UNIVERSITY OF GRANADA

Intelligent Systems for Function Approximation and the Integration of Heterogeneous Biological Data

by

Javier Pérez Florido

Department of Computer Architecture and Computer Technology

Granada, September 2011

Editor: Editorial de la Universidad de Granada
Autor: Javier Pérez Florido
D.L. : GR 1564-2012
ISBN: 978-84-9028-024-9

UNIVERSITY OF GRANADA

Intelligent Systems for Function Approximation and the Integration of Heterogeneous Biological Data

(Sistemas inteligentes para aproximación funcional y la integración de datos biológicos heterogéneos)

Dissertation presented by

Javier Pérez Florido

To apply for the

European PhD degree in Computer Science

Signed. Javier Pérez Florido

D. Héctor Pomares Cintas y D. Ignacio Rojas Ruiz, Profesor Titular y Catedrático de Universidad respectivamente del Departamento de Arquitectura y Tecnología de Computadores

CERTIFICAN

Que la memoria titulada "Intelligent Systems for Function Approximation and the Integration of Heterogeneous Biological Data" ha sido realizada por D. Javier Pérez Florido, bajo nuestra dirección en el Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada para optar al grado de Doctor Europeo en Ingeniería Informática.

Granada, a de Septiembre de 2011

Fdo. Héctor Pomares Cintas
Director de la Tesis

Fdo. Ignacio Rojas Ruiz
Director de la Tesis

Agradecimientos

Me gustaría aprovechar esta páginas para agradecer a todas las personas que me han apoyado en el desarrollo de este trabajo.

En primer lugar a Héctor Pomares Cintas e Ignacio Rojas Ruiz, mis directores de tesis, por su buena labor a lo largo de estos cuatros años. Gracias por darme la libertad de realizar todas aquellas cosas que, de alguna forma, pensaba que eran interesantes y que podrían aportar algo a esta sociedad. Al fin y al cabo, la investigación tiene que estar al servicio de la misma.

Al departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada, por su constante apoyo en mi carrera investigadora y a todos los componentes del mismo, por el fantástico ambiente de trabajo existente.

A Fran, Curro, Pablo, Jose, Natalia, Antonio y todos los demás compañeros del CITIC-UGR.

A Zlatko Trajanoski, de la sección de Bioinformática de la Universidad de Innsbruck, por permitirme realizar una fructífera estancia en el Instituto de Genómica y Bioinformática de la Universidad Técnica de Graz cuando era el responsable del mismo.

A Yu Xia, de la Universidad de Boston, donde realicé parte de mi trabajo de investigación.

También quiero agradecer a mis familiares y amigos, por todo el apoyo y cariño que me han brindado durante esa dura etapa. A Jesús por hablar de mis temas de investigación con quien fuera y obligarme, de vez en cuando, a salir de la pantalla del ordenador. También quiero agradecer a mis abuelos, Juan y Carmen, su cariño a lo largo de todos estos años: nunca olvidaré nuestros almuerzos juntos. Este trabajo va también dedicado a vosotros.

Por último, agraceder a mis padres su constante preocupación, ayuda y buenos consejos. Gracias a vosotros soy como soy.

A Aída, por su paciencia, ayuda, ánimos y amor que me brinda cada día.

La realización de este trabajo no habría sido posible sin la beca de Formación de Profesorado Universitario del Ministerio de Educación que disfruté durante los años 2008-2011 (AP2007-03009) y los proyectos del ministerio de Ciencia e innovación TIN2007-60587 and SAF2010-20558.

Contents

List of Figures

List of Tables

Abbreviations

DNA	the DeoxyriboNucleic Acid
RNA	RiboNucleic Acid
FLN	Functional Linkage Network
MO-GP	Multi-Objective-Genetic Programming
RBFNN	Radial Basis Function Neural Network
CLL	Chronic Lymphocytic Leukemia
PPI	Protein Protein Interaction
GI	Gene Interaction
GF	Gene Fusion
GC	Gene Cluster
PP	Phylogenetic Profile
GN	Gene Neighbor
SS	Sequence Similarity
PDS	Protein Domain Sharing
GO	Gene Ontology
SSBP	Smallest Shared Biological Process
SSMF	Smallest Shared Molecular Function
SSCC	Smallest Shared Cellular Component
GSP	Gold Standard Positive
GSN	Gold Standard Negative
KEGG	Kyoto Encyclopedia of Genes and Genomes

MIPS	the Munich Information center for Protein Sequences
COG	Clusters of Orthologous Groups
ROC	Receiver Operating Characteristic
PPV	Positive Predictive Value
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative
AUC	Area Under Curve
AUPRC	Area Under Precision Recall Curve
LR	Likelihood Ratio
LLR	Log Likelihood Ratio
ANN	Artificial Neural Network
MLP	Multilayer Perceptron
DT	Decision Tree
RF	Random Forest
LR	Linear Regression
SVM	Support Vector Machine
RVM	Relevance Vector Machine
EA	Evolutionary Algorithm
GP	Genetic Programming
NSGA	Non-dominated Sorting Genetic Algorithm
MO	Multi-Objective
MOEA	Multi-Objective Evolutionary Algorithm
MOGP	Multi-Objective Genetic Programming
POCR	Pareto Optimal Classification Rules
DNF	Disjuctive Normal Form
TM	Threshold Moving
MMC	Minimization of Misclassification Costs

MLP-TM	Multi-Layer Perceptron - Threshold Moving
MLP-MMC	Multi-Layer Perceptron - Minimization of Misclassification Costs
ANOVA	ANalysis Of VAriance
LS-SVM	Least Square - Support Vector Machine
OMIM	Online Mendelian Inheritance Man database
PRC	Pareto Classification Rules
CGH	Comparative Genomic Hybridization
RLE	Relative Log Expression
NUSE	Normalized Unscaled Standard Error
VSN	Variance StabilizatioN
MAS	MicroArray Suite
RMA	Robust Multichip Average
RBF	Radial Basis Function
ARIMA	Auto-Regressive Integrated Moving Average
ARMAX	Auto-Regressive Moving Average with eXogenous inputs
ARX	Auto-Regressive with eXogeneous inputs
NN	Neural Network
KL	Kullback Leibler
CFA	Clustering for Function Approximation
NNO	Nearest Neighbor Out
MSE	Mean Squared Error
NRMSE	Normalized Root Mean Square Error
AIC	Akaike's Information Criterion
BIC	Bayesian Information Criterion
MDL	Minimum Description Length
ICFA	Improved Clustering for Function Approximation
ANFIS	Adaptive Network-based Fuzzy Inference System

Dedicado a mis padres

Abstract

This dissertation presents a set of contributions that can be grouped into three parts. The first part of the thesis is related to the integration of heterogeneous biological data for the prediction of functional associations between proteins. This topic has become in the last years one of the major goals of current biological studies. In the literature, there exist several machine learning methods applied to the integration of heterogeneous biological data sources. However, all of them suffer from the same common problem: interpretability and simplicity for the decision maker. Due to this, it is proposed a data integration methodology based on interpretable and simple IF-THEN rules that reflect the contributions of different types of evidences or data sources toward the prediction of functional associations between proteins. Through a multi-objective genetic programming (MO-GP) approach run in parallel architectures, a set of pareto optimal IF-THEN classification rules are provided and each rule can be used to build an functional linkage network (FLN) with a given level of accuracy. Furthermore, the decision maker does not have to specify partial preferences on the desired accuracy of the FLN, since covering the entire pareto, different FLNs are obtained, each one with a different level of accuracy.

The second part of the dissertation is related to the automation of Affymetrix 3' microarray data analysis. Microarray data are commonly used in the data integration previously described so that it is proposed a microarray data analysis tool with the following features: (1) automated detection of low quality microarrays so that the decision maker is able to decide whether one or more arrays are defective or not based on a full set of quantitative and qualitative measures, (2) automated selection of the best pre-processing methods among several ones for a given data set through objective quality metrics and (3) automated generation of confident and

complete lists of differentially expressed genes according to the set of best pre-processing methods selected before. This automation means an important advance in microarray data analysis and a great help to the decision maker, since the automatic detection of low quality microarrays and the automatic selection of the best pre-processing methods will avoid that posterior phases on microarray data analysis, such as classification, are affected by low quality arrays and/or an incorrect choice of pre-processing methods.

The third part of the dissertation is related to the problem of distributing the original data set (input/output data) into two representative and balanced sets for function approximation tasks and to the problem of model selection. Two contributions are proposed. The first one is related to one of the most common methodologies to evaluate models built by supervised learning algorithms. Such methodology consists in partitioning the original data set (input/output data) into two sets: learning and test. The learning set is used for building models that capture the relationships between inputs and outputs while the test set is used for checking models' generalization ability with data not used in the learning process. Usually, in the literature, the partition into learning and test sets does not usually take into account the variability and geometry of the original data. This might lead to non-balanced and unrepresentative learning and test sets and, thus, to wrong conclusions in the accuracy of the learning algorithm. Thus, it is proposed a new deterministic data mining approach to distributing a given data set (input/output data) into two representative and balanced sets of roughly equal size to be used in function approximation problems. The distribution takes into account the variability of the data set with the purpose of allowing both a fair evaluation of learning's accuracy and to make reproducible machine learning experiments usually based on random distributions.

The second contribution is associated to one of the problems related to the selection of the best model for Radial Basis Function Neural Network (RBFNN) in time series prediction tasks. This problem is given by the methodology commonly used in the literature to select the best structure model. Such methodology is based on *K-fold cross-validation* model evaluation strategy which has some drawbacks, such as its random nature and the subjective decision for a proper value of K. Thus, it is proposed a new deterministic model selection methodology with applications for incremental Radial Basis Function Neural Network (RBFNN) construction in time

series prediction problems. Such model selection approach is a combined algorithm which takes advantage of balanced and representative training and validation sets obtained through the data distribution approach previously described for their use in all the steps of the RBFNN design: initialization, optimization and network model evaluation. This way, the model prediction accuracy is improved, reducing the computation time spent in selecting the model and avoiding random and computationally expensive model selection methodologies based on *K-fold cross-validation* procedures.

Resumen

Esta tesis doctoral presenta un conjunto de aportaciones que pueden ser agrupadas en tres partes. El primer bloque de la tesis está relacionado con la integración de diversas fuentes biológicas heterogéneas para la predicción de nuevas relaciones funcionales entre proteínas. Esta línea de investigación es uno de los mayores retos actuales de la biología. En la literatura, existen numerosas herramientas de aprendizaje aplicadas a la integración de diversas fuentes heterogéneas de datos. Sin embargo, todas ellas adolecen de un problema común: interpretabilidad y sencillez de cara al investigador. Debido a esto, se ha desarrollado una metodología de integración basada en reglas SI-ENTONCES que reflejan de forma clara y sencilla las contribuciones de las diferentes fuentes de datos integradas para la tarea de predicción de asociaciones funcionales. A través de una metodología basada en programación genética multi-objetivo ejecutada en arquitecturas paralelas, un conjunto de reglas pareto-óptimas es obtenido, donde cada regla se puede utilizar para construir una red de asociaciones funcionales de proteínas (*Functional Linkage Network*, FLN) y diferentes reglas del pareto dan lugar a diferentes redes de asociaciones en términos de precisión. Además, a través de esta nueva metodología, el investigador no tiene que especificar preferencias en cuanto a la precisión de la red deseada, ya que, cubriendo todo el pareto de soluciones, se obtienen diferentes redes de asociaciones funcionales, cada una de ellas con un nivel de precisión.

El segundo bloque de esta Tesis Doctoral está relacionado con la automatización del análisis de datos de microarrays de Affymetrix 3'. Los datos de microarrays son utilizados comúnmente en la integración de datos previamente descrita. Así, se ha desarrollado una herramienta de análisis de microarrays que ofrece las siguientes funcionalidades originales: (1) automatización del análisis de calidad de un

conjunto de datos de microarrays de forma que el investigador es capaz de decidir la eliminación de muestras defectuosas de acuerdo a un conjunto completo de métricas cualitativas y cuantitativas, (2) selección automática de los mejores métodos de pre-procesamiento de entre decenas de ellos para el conjunto de datos dado a través de métricas objetivas de calidad, (3) generación automática de dos listas de genes expresados diferencialmente: una con candidatos fiables y otra con candidatos posibles. Esta automatización supone un gran avance en el análisis de microarrays, ya que ayuda al usuario, posiblemente no experto, en la toma de decisiones en el análisis de microarrays, de forma que la detección de microarrays de baja calidad y/o la selección automática de los mejores métodos de pre-procesamiento evitará que posteriores etapas de dicho análisis, como la clasificación, se vean afectadas por la presencia de muestras defectuosas y/o una elección incorrecta del método de pre-procesamiento.

La tercera parte de la tesis doctoral está relacionada con el problema de la distribución del conjunto de datos original (datos de entrada/salida) en dos conjuntos representativos y balanceados para tareas de aproximación funcional y con el problema de selección del modelo. Se proponen dos aportaciones. La primera de ellas está relacionada con una de las formas más utilizadas en la literatura para evaluar un algoritmo de aprendizaje y que consiste en la partición del conjunto de datos de entrada/salida en dos conjuntos: aprendizaje, utilizado para construir modelos que reflejen la relación entre entradas y salidas, y test, para la evaluación del modelo generado. De forma general, en la literatura, no se tiene en cuenta la variabilidad y la geometría del conjunto de datos en la partición, de forma que se pueden obtener conclusiones erróneas acerca del algoritmo de aprendizaje. Así, se ha desarrollado una metodología determinista para una partición del conjunto de datos en dos conjuntos balanceados y representativos teniendo en cuenta la variabilidad de dicho conjunto de datos original con el propósito de permitir una evaluación justa de los algoritmos de aprendizaje y realizar experimentos reproducibles de aprendizaje normalmente basados en distribuciones aleatorias.

La segunda contribución está asociada con uno de los problemas relacionados con la selección del mejor modelo para redes neuronales de base radial (*Radial Basis Function Neural Networks*, RBFNNs) en tareas de predicción de series temporales. Dicho problema viene dado por la elección comúnmente utilizada en la

literatura para seleccionar el mejor modelo de red, la cual está basada en estrategias de validación cruzada K-veces que tiene como principales inconvenientes su naturaleza aleatoria y la elección de un valor adecuado de K. Así, se ha propuesto una nueva metodología de selección del modelo con aplicaciones en RBFNNs en problemas de predicción de series temporales. La metodología propuesta está basada en un algoritmo combinado que hace uso de conjuntos balanceados y representativos, como los obtenidos en la aportación anterior, en todas las etapas de diseño de RBFNN: inicialización, optimización y evaluación del modelo de red. De esta forma, la precisión en las predicciones del modelo es mejorada, reduciendo, además, el tiempo de computación empleado en la selección del modelo y, evitando de esta forma, metodologías de selección del modelo computacionalmente costosas y aleatorias como las basadas en la validación cruzada K-veces.

Introducción

Esta introducción contiene una versión en español del Capítulo 1 y ha sido incluida para cumplir con los requerimientos necesarios para poder optar a la mención de *Doctorado Europeo*.

Antecedentes

El término ”Sistemas Inteligentes” es utilizado para describir sistemas y métodos que simulan ciertos aspectos del comportamiento inteligente de los seres humanos para diseñar modelos computacionales. El principal objetivo de los sistemas inteligentes es, por tanto, construir modelos que puedan representar su propio conocimiento y razonar sobre él, que puedan planificar y actuar y que puedan asimilar nuevo conocimiento de la experiencia y de la interacción con el entorno.

Los sistemas inteligentes son el paradigma clave en muchas de las aplicaciones actuales, como por ejemplo el diagnóstico médico, el control robótico, la predicción del tiempo, la predicción de valores de bolsa, los procesos industriales, el control de plantas, la industria financiera, los sistemas de visión, el análisis de secuencias genómicas, la predicción de funciones/estructuras de proteínas, etc.

Esta tesis, presenta una serie de aportaciones dentro de la extensa área de las aplicaciones de los sistemas inteligentes. Concretamente, la tesis se centra en el desarrollo/aplicación de sistemas inteligentes en dos campos: (1) la integración de diversas fuentes biológicas heterogéneas para predecir asociaciones/relaciones funcionales entre proteínas y (2) la aproximación funcional.

Integración de fuentes biológicas heterogéneas para predecir asociaciones funcionales entre proteínas

La predicción de relaciones funcionales entre genes/proteínas es, hoy día, uno de los mayores objetivos de los estudios relacionados con la biología, debido a que las tareas celulares complejas de un organismo normalmente dependen de asociaciones entre proteínas que colaboran entre ellas como un bloque o módulo funcional, el cual puede estar representado por un proceso biológico concreto o un pathway específico. Las asociaciones funcionales puede ser inferidas o extraídas a partir de fuentes heterogéneas de datos como los experimentos de micromatrices o microarrays, los cuales detectan co-expresión entre genes, o los datos de secuencias a partir de los que se pueden extraer correlaciones de perfiles filogenéticos. El resultado de una búsqueda de relaciones o asociaciones funcionales entre proteínas es representado como un grafo denominado *Red de relaciones funcionales* (Functional Linkage Network, FLN, en inglés). En este grafo, los nodos representan proteínas y las uniones entre ellas representan un grado de relación o similitud funcional. Sin embargo, cada fuente de datos únicamente revela una cierta perspectiva del genoma completo y del mecanismo biológico existente subyacente. Es más, algunas de las fuentes de datos son criticadas por su ruido y baja fiabilidad y, por tanto, carecen del grado de especificidad requerido para predicciones precisas de relaciones funcionales. Debido a estos problemas, en los últimos años, la integración de fuentes heterogéneas de datos para predecir relaciones funcionales entre proteínas haciendo uso de metodologías de aprendizaje automático (*machine learning* en inglés), representan una forma prometedora de superar todos estos problemas, proporcionando una red de relaciones funcionales más completa donde las predicciones de relaciones funcionales entre proteínas son más precisas.

En la literatura, diversas metodologías se han aplicado con éxito a la integración de evidencias a partir de datos genómicos y proteómicos heterogéneos. No obstante, y a pesar del incremento en la precisión de las predicciones de relaciones funcionales cuando diversas fuentes son integradas, todavía existen diversas mejoras que pueden ser tenidas en cuenta en dichas metodologías, como la interpretabilidad y la sencillez de los modelos construídos por el método de integración y la posibilidad de proporcionar, sin un incremento excesivo del coste computacional, diversas

FLNs con diferentes niveles de precisión ya que, a priori, el nivel de precisión deseado por un investigador en la relaciones funcionales predichas es desconocido.

Por otro lado, una red de relaciones funcionales (FLN) construída a partir de la integración de diversas fuentes biológicas heterogéneas, puede, además, ser utilizada para predecir nuevos genes asociados con una determinada enfermedad. Para este fin y para una enfermedad determinada, los genes conocidos asociados a esa enfermedad son marcados en la FLN y el resto de genes relacionados con los genes enfermos marcados son identificados, de forma que estos últimos pueden asociarse con dicha enfermedad con un nivel de confianza dado por el grado de asociación con los genes enfermos. Los genes relacionados con una determinada enfermedad y que son utilizados para ser marcados en la FLN pueden ser extraídos a partir de experimentos a medida como los datos de microarray (o micromatrices) y, para este fin, se debe llevar a cabo un análisis organizado de los datos de microarrays en diferentes etapas como: análisis de calidad, pre-procesamiento de los datos y la detección de genes expresados diferencialmente. Este análisis multi-etapa puede llevarse a cabo utilizando numerosas herramientas de análisis de microarrays existentes en la literatura. Sin embargo y a pesar de ser herramientas muy completas, todavía existe la necesidad de ayudar al usuario, probablemente no experto, en algunas etapas del análisis mencionado. Por ejemplo, en la detección automática de microarrays de baja calidad o en la selección automática de los mejores y más apropiados métodos de pre-procesamiento para un experimento de datos dado. De esta forma, los posibles errores presentes en etapas posteriores del análisis de microarrays, como la clasificación, debido a la presencia de microarrays de baja calidad y/o una selección incorrecta de los métodos de pre-procesamiento, pueden ser reducidos.

Aproximación funcional

Un problema de aproximación funcional puede definirse de la siguiente forma: dado un conjuno de observaciones o datos de entrada/salida extraídos de una función o sistema desconocido F, es deseado obtener un modelo $F^* \approx F$ a partir del cual se puedan proporcionar salidas precisas a partir de datos de entrada del conjunto y, además, con buena capacidad de predicción para nuevos datos de entrada.

Los problemas de aproximación funcional han recibido gran atención en áreas como la predicción de series temporales y la identificación del sistema. La predicción de series temporales consiste en la predicción de valores futuros de una secuencia a partir de valores pasados y es un reto en numerosos campos, como la predicción de valores de bolsa o del consumo de electricidad. Por otro lado, la identificación del sistema es uno de los aspectos más importantes en campos como el control de plantas, las comunicaciones o el reconocimiento de patrones.

Existe una cuestión o asunto no tratado apropiadamente en la comunidad científica en tareas de aproximación funcional. Este asunto está relacionado con una de las metodologías comúnmente utilizadas para evaluar modelos construídos por algoritmos de aprendizaje supervisados. Esta metodología consiste en la partición del conjunto de datos original (datos de entrada/salida) en dos conjuntos: aprendizaje y test. El conjunto de aprendizaje, el cual puede ser dividido a su vez en entrenamiento y validación, es utilizado para construir modelos que capturan las relaciones entre las entradas y las salidas. Por otro lado, el conjunto de test es utilizado para comprobar la capacidad de generalización de los modelos construídos haciendo uso de datos no utilizados en el proceso de aprendizaje. Normalmente, en la literatura, la partición o división del conjunto de datos original en los conjuntos de learning y test es realizada de forma aleatoria o con algún tipo de muestreado uniforme en un dominio determinado, es decir, la partición en aprendizaje y test no tiene en cuenta la variabilidad y la geometría de los datos originales. Esto puede causar conjuntos de aprendizaje y test no balanceados y no representativos y, por tanto, conclusiones erróneas en la precisión del algoritmo de aprendizaje. Es más, las comparaciones entre el rendimiento obtenido por diversos algoritmos de aprendizaje en diferentes experimentos son complicadas si la distribución o partición se ha obtenido de forma aleatoria debido a la necesidad de utilizar numerosas particiones aleatorias para obtener una estimación representativa de la calidad de los algoritmos de aprendizaje. Así, cómo se realiza la partición o distribución es esencial y es más importante cuando el conjunto de datos original es pequeño, debido a la necesidad de reducir los efectos negativos debidos a la eliminación de muestras del conjunto de datos original.

Por otro lado, dicha partición no sólo es útil para la evaluación de la precisión

o rendimiento de un algoritmo de aprendizaje y para realizar experimentos reproducibles de aprendizaje. También es útil en el contexto de la selección de la mejor estructura de un modelo (por ejemplo, de diferente complejidades, diferentes número de neuronas en la capa oculta para un perceptrón multicapa, etc) para un problema determinado. Esto es conocido como selección del modelo. Como previamente se ha comentado, el conjunto de aprendizaje puede ser dividido a su vez en dos conjuntos: entrenamiento, utilizado para la estimación de los parámetros de una estructura determinada del modelo, y validación, utilizado para evaluar el modelo entrenado, obteniendo así un error de validación. La selección del modelo está, por tanto, relacionada con la tarea de comparar numerosas estructuras del modelo de acuerdo a estimaciones de sus errores de validación con el objetivo de seleccionar la estructura del modelo más apropiada para un problema determinado.

Normalmente, en la literaura, la estimación del error de validación es obtenida utilizando la estrategia de evaluación del modelo denominada *validación cruzada K-veces* (*K-fold cross-validation* en inglés). Sin embargo, esta estrategia tiene numerosos inconvenientes, por ejemplo, su naturaleza aleatoria (no tiene en cuenta la variabilidad y la geometría del conjunto de datos de aprendizaje cuando se construyen los conjuntos de entrenamiento y validación) y la decisión subjetiva de un valor apropiado para K, proporcionando un sesgo elevado para valores bajos y varianza y coste computacional elevados para valores altos. Por tanto, es deseable una metodologǵia de selección del modelo basada en una estrategia de evaluación del modelo con las siguientes características: (1) sesgo y varianza bajos, (2) sin aleatoriedad, (3) con un coste computacional bajo y (4) que utilice conjuntos de entrenamiento y validación balanceados y representativos.

Aportaciones de la tesis

Esta tesis presenta un conjunto de cuatro aportaciones que tratan de dar una solución a las necesidades descritas previamente. Así y relacionado con el tema de la integración de datos biológicos, proponemos dos contribuciones:

- Una nueva metodología para la integración de fuentes biológicas heterogéneas para predecir relaciones funcionales entre proteínas. La metodología desarrollada es interpretable ya que está basada en reglas de clasificación sencillas de la forma SI-ENTONCES. Estas reglas reflejan las contribuciones de los diferentes tipos de fuentes utilizadas en la tarea de predicción (relaciones funcionales). A través de una propuesta basada en programación genética multi-objetivo (*multi-objective genetic programming*, MO-GP, en inglés) ejecutada en arquitecturas paralelas, se proporciona un conjunto de reglas de clasificación SI-ENTONCES pareto-óptimas de forma que cada regla es utilizada para construir una FLN con un nivel de precisión dado. Así, la propuesta MO-GP evoluciona simultáneamente múltiples reglas de forma que no se incrementa significativamente el tiempo de aprendizaje cuando se compara con otras metodologías de integración de datos. Es más, el investigador no tiene que especificar preferencias en la precisión de la FLN, ya que cubriendo el pareto completo, se obtienen diferentes FLNs, cada una con un nivel de precisión.

- Una nueva herramienta para las primeras tres etapas del análsis de datos de microarrays del tipo Affymetrix 3': análisis de calidad, pre-procesamiento y detección de genes expresados diferencialmente. Esta nueva herramienta, proporciona las siguientes características:

  - Detección automática de microarrays de baja calidad, de forma que el investigador es capaz de decidir si elimina un microarray defectuoso o no, de acuerdo a numerosas métricas cuantitativas y cualitativas de calidad.
  - Selección automática de los mejores métodos de pre- procesamiento para un conjunto de datos dado a través de métricas de calidad objetivas. El objetivo es liberar al investigador de la ardua tarea de la selección de uno o más métodos de pre-procesamiento.
  - Generación automática de listas fiables y completas de genes expresados diferencialmente de acuerdo a los mejores métodos de pre- procesamiento obtenidos en el paso anterior.

Por medio de esta herramienta, los genes relacionados con una determinada enfermedad a partir de datos de microarrays, pueden ser extraídos y marcados en la FLN para predecir nuevos genes que potencialmente pueden estar asociados con la enfermedad bajo estudio.

Respecto al tema de aproximación funcional, proponemos dos contribuciones:

- Una nueva metodología determinista para distribuir un conjunto de datos de entrada/salida dado en dos conjuntos representativos y balanceados de, aproximadamente el mismo tamaño para ser utilizado en problemas de aproximación funcional. La distribución tiene en cuenta la variabilidad del conjunto de datos con el objetivo de permitir una evaluación justa de la precisión o el rendimiento de un algoritmo de aprendizaje y para realizar experimentos reproducibles de aprendizaje automático normalmente basados en distribuciones aleatorias. Los conjuntos son generados por medio de una combinación de un procedimiento de clustering, especialmente diseñado para problemas de aproximación funcional, y de un algoritmo de distribución que distribuye, para cada cluster, un conjunto de datos en dos conjuntos de acuerdo a una metodología de vecinos más cercanos.

- Una nueva metodología de selección del modelo con aplicación directa a la construcción incremental de redes neuronales de base radial (Radial Basis Function Neural Network, RBFNN, en inglés) en problemas de predicción de series temporales. Dicha metodología es un algoritmo combinado que hace uso de los conjuntos balanceados y representativos de entrenamiento y validación obtenidos mediante la metodología de distribución propuesta anteriormente. Estos conjuntos son utilizados en la inicialización de la RBFNN, en su optimización y en la evalución del modelo de red, mejorando, de esta forma, la capacidad de predicción del modelo, obteniendo sesgo y varianza bajos, reduciendo el tiempo empleado en seleccionar el modelo y evitando metodologías de selección del modelo aleatorias y computacionalmente costosas como las basadas en procedimientos de *validación cruzada K-veces*.

Estructura de la tesis

La tesis está estructurada en tres partes. La primera parte contiene tres capítulos dedicados al problema de la integración de fuentes biológicas heterogéneas para predecir relaciones funcionales entre proteínas. El capítulo 2 proporciona una introducción al problema de la predicción de relaciones funcionales entre proteínas y motiva el desarrollo del trabajo desarrollado en este bloque de la tesis. El capítulo 3 introduce algunos aspectos teóricos como una introducción básica a la biología y a la bioinformática, la cual constituye el campo de investigación de esta parte de la tesis. Seguidamente, se revisan las metodologías existentes en la literatura para la predicción de relaciones funcionales entre proteínas utilizando fuentes individuales. Después, describimos las aproximaciones desarrollados por otros autores para la misma tarea de predicción pero integrando diversas fuentes o evidencias. El capítulo 4 presenta en detalle la metodología MO-GP propuesta que proporciona reglas de clasificación pareto óptimas para la integración de fuentes biológicas heterogéneas de datos para predecir relaciones funcionales entre proteínas. En este capítulo, se lleva a cabo una evaluación de la metodología propuesta así como una comparación con otras metodologías existentes.

El segundo bloque de la tesis está compuesto de tres capítulos, todos ellos dedicados al análisis de datos de microarrays. El capítulo 5 proporciona una introducción a las diferentes etapas que componen el análisis de datos de microarrays y motiva el desarrollo del trabajo desarrollado en esta parte de la tesis. El capítulo 6 proporciona una visión general de la tecnología de microarrays, una descripción de las diferentes etapas que componen el análisis de microarrays y una enumeración de algunas de las herramientas que existen en la literatura para analizar microarrays. La herramienta de análisis de microarrays propuesta en esta tesis es decrita en detalle en el capítulo 7, así como su aplicación al conjunto de datos *Chronic Lymphocytic Leukemia, CLL.*

La tercera parte de la tesis contiene tres capítulos dedicados al problema de la distribución de un conjunto de datos de entrada/salida en dos conjuntos balanceados y representativos para aproximación funcional y al problema de la selección del

modelo. El capítulo 8 proporciona una introducción a estos dos problemas, motivando el desarrollo de las metodologías propuestas en este bloque de la tesis. El capítulo 9 presenta la metodología propuesta para el particionamiento de un conjunto de datos de entrada/salida en dos conjuntos representativos y balanceados de, aproximadamente, el mismo tamaño para ser utilizados en problemas de aproximación funcional. La nueva metodología es evaluada y comparada con otras metodologías existentes. Finalmente, la nueva metodología de selección del modelo para RBFNNs en problemas de predicción de series temporales es presentada en el capítulo 10, con aplicaciones a varios problemas de series temporales así como una comparativa con otras metodologías de predicción de series temporales.

Finalmente, las principales conclusiones de la tesis son discutidas en el capítulo 11. Las publicaciones relacionadas con las contribuciones de esa tesis son, además, enumeradas.

Con el objetivo de hacer más sencilla la lectura de tesis, la siguiente guía puede ser tenida en cuenta:

- Cada parte está compuesta de capítulos. Debido a que se proporciona un conjunto de contribuciones diferentes, se proporciona una introducción en cada parte o bloque de la tesis para su mejor entendimiento.
- Cada capítulo está dividido en secciones, que a su vez contienen subsecciones y sub-subsecciones. Cuando proporcionamos una referencia hacia cualquier parte del presente documento, la referencia incluye un número de capítulo, seguido de un número de sección, subsección, etc.
- Las figuras y tablas son numeradas por capítulos. Por ejemplo, Figura 4.1.
- Las expresiones matemáticas siguen el mismo esquema que los capítulos y tienen un número asociado. Por ejemplo, ecuación 9.1
- Las referencias bibliográficas se indican mediante el apellido de los autores seguido del año de publicación si el trabajo tiene al menos 3 autores y por el apellido del principal autor seguido de la abreviación *et al.*, si el trabajo tiene más autores. Ejemplo:[Gonzalez et al., 2003],[Paul and Kumar, 2002].

Chapter 1

Introduction

1.1 Antecedents

The term ”Intelligent Systems” is used to describe systems and methods that simulate certain aspects of the intelligent behavior of a human being in order to design computational models. The main goal of intelligent systems is therefore to build models that can represent and reason knowledge, plan, act and assimilate new knowledge from the experience and the environment.

Intelligent systems are the key paradigm in many of today's applications, such as medical diagnosis, robot control, weather forecasting, stock market indexes prediction, industrial processes, plant control, financial industry, vision systems, genomic sequence analysis, protein structure/function prediction, etc.

This dissertation presents a set of contributions inside the open-wide area of intelligent system applications. Specifically, this thesis is focused on the development/application of intelligent systems to two fields: (1) the integration of heterogeneous biological data to predict functional associations between proteins and (2) function approximation.

1.1.1 Integration of heterogeneous biological data to predict functional associations between proteins

The prediction of functional associations between genes/proteins has become in the last years one of the major goals of current biological studies, since the complicated cellular tasks of an organism frequently rely on associations among proteins that collaborate with each other as a functional module, which can be represented by a particular biological process or a specific pathway. Functional associations can be inferred from different heterogeneous data sources such as microarray experiments, which detect co-expression among genes, or sequence data, from which correlations of phylogenetic profiles can be extracted. The result of a search for such associations is conveniently displayed as a graph, the so called Functional Linkage Network (FLN), where the nodes represent proteins and the edges between them expressing the degree of functional similarity. However, each data source can only reveal a certain perspective of both the whole genome and the underlying complex biological mechanism. Moreover, some of the data sources are criticized for their noise and low reliability and thus may lack the degree of specificity required for an accurate prediction of functional association. Due to these reasons, in recent years, the integration of evidences from heterogeneous data sources to predict functional associations by means of machine learning methods represents a promising way to overcome these drawbacks, providing a complete genome-wide functional network and more accurate inferences of new functional relationships between proteins.

Several methodologies have successfully been applied in the literature for the integration of evidences from heterogeneous genomic and proteomic data. Nevertheless and despite the increase in the prediction power of functional associations when several data evidences are integrated, improvements of such methodologies are still needed such as the interpretability and simplicity of the model built by a data integration method and the possibility of providing, without a dramatic increase in the computational cost, several FLNs with different levels of accuracy since it is unknown, a priori, the partial preferences of a decision maker on the accuracy of the functional associations predicted.

A functional linkage network (FLN) constructed by integrating several heterogeneous biological data sources can also be used to predict or prioritize new genes

that are potentially associated with a given disease. For this purpose and given a particular disease, genes known to be associated with this disease are labelled as seeds in the FLN and all other genes are prioritized in terms of their association with the disease based on the sum of the weights of their network links to the seed genes. One can extract/discover genes related to a given disease from custom experiments such as microarray data and, for this purpose, an organized analysis of such data must be accomplished, including steps of quality data analysis, data pre-processing and the detection of differentially expressed genes. This multistep analysis can be carried out using several analysis tools available in the literature. However and despite being complete enough tools for microarray data analysis, there is still a need of helping the (non-expert) decision maker in some steps of the analysis pipeline. For example, in the automatic detection of low quality arrays or the automatic selection of the best and more suitable pre-processing methods for a given data experiment. This way, the possible errors in posterior microarray analysis phases, such as classification, due to the presence of low quality arrays and/or incorrect choice of pre-processing methods can be reduced.

1.1.2 Function approximation

A function approximation problem can be defined as follows: given a set of observations or input/output data sampled from an unknown function or system F, it is desired to obtain a model $F^* \approx F$ by which accurate outputs from input data specified in the original data set can be provided and with good predictive performance for new input data.

Function approximation problems have received great attention in areas such as time series prediction or system identification. Time series forecasting consists in the prediction of future values of a sequence of past values and is a challenge in many fields, such as the prediction of stock market indexes or electricity consumption. On the other hand, system identification is one of the most important aspects in plant control, communication, pattern recognition and fault analysis fields.

There exists an issue not properly discussed or treated in the research community in function approximation tasks. This issue is related to one of the most common

methodologies to evaluate models built by supervised learning algorithms. Such methodology consists in partitioning the original data set (input/output data) into two sets: learning and test. The learning set, which can be in turn divided into training and validation sets, is used for building models that capture the relationships between inputs and outputs. On the other hand, the test set is used for checking models' generalization ability with data not used in the learning process. Usually, in the literature, the partition into learning and test sets is made randomly or with some kind of sampling uniformly in some domain, that is, the partition into learning and test sets does not usually take into account the variability and geometry of the original data. This might lead to non-balanced and unrepresentative learning and test sets and, thus, to wrong conclusions in the accuracy of the learning algorithm. Moreover, comparisons between performances of several learning algorithms in different experiments are difficult if randomness is present in the distribution due to the need of using several random splits to get a reliable estimate of the quality of the learning algorithms. How the partitioning is made is therefore a key issue and becomes more important when the data set is small due to the need of reducing the pessimistic effects caused by the removal of instances from the original data set.

On the other hand, such proper partition is not only useful for the evaluation of learning's accuracy and to make reproducible machine learning experiments, but also in the context of the selection of the best model structure (for example of different complexities, different number of neurons in the hidden layer of multilayer perceptrons, etc) for a given problem, which is known as model selection. As previously stated, the learning set can be, in turn, split into two sets: training, used for parameter estimation for a given model structure, and validation, used for evaluating the trained model, obtaining a validation error. Model selection is, therefore, related to the task of comparing several model structures based on estimations of their validation errors in order to select the most suitable model structure for a given problem.

Usually, in the literature, the estimation of the validation error is obtained using the *K-fold cross-validation* model evaluation strategy. However, this approach has some drawbacks, such as its random nature (it does not take into account the variability and geometry of the learning data when building the training and validation

sets) and the subjective decision for a proper value of K, resulting in large bias for low values and high variance and computational cost for high values. So, it is desirable that a model selection approach is based on a model evaluation strategy with the following features: (1) low variance and bias, (2) no randomness, (3) low computational cost and (4) use of balanced and representative training and validation sets.

1.2 Contributions of the Dissertation

This thesis presents a set of four contributions that try to bring a solution to the set of needs described above. Related to the topic of biological data integration, we propose two contributions:

- A new methodology for the integration of evidences from heterogeneous biological data sources to predict functional associations between proteins. The methodology developed is an interpretable approach since it is based on simple IF-THEN classification rules that reflect the contributions of different types of evidences toward the prediction task, in this case, functional associations. Through a multi-objective genetic programming (MO-GP) approach run in parallel architectures, a set or pareto optimal IF-THEN classification rules are provided and each rule can be used to build an FLN with a given level of accuracy. This way, the MO-GP approach simultaneously evolves toward multiple pareto optimal rules and does not dramatically increase the learning time when compared to other data integration methodologies. Furthermore, the decision maker does not have to specify partial preferences on the desired accuracy of the FLN, since covering the entire pareto, different FLNs are obtained, each one with a different level of accuracy.

- A new tool for the first three steps of standard Affymetrix 3' expression arrays data analysis pipeline: quality assessment, pre-processing and the detection of differentially expressed genes. This new tool provides the following features:

1. Automated detection of low quality microarrays so that the decision maker is able to decide whether one or more arrays are defective or not based on a full set of quantitative and qualitative measures.
2. Automated selection of the best pre-processing methods among several ones for a given data set through objective quality metrics. The aim is to free the researcher from taking a decision about the pre-processing method(s) to be used.
3. Automated generation of confident and complete lists of differentially expressed genes according to the set of best pre-processing methods selected before.

By means of this tool, genes related to a given disease from a custom microarray data experiment can be extracted and labelled in an FLN to predict new genes that are potentially associated with the disease under study.

With regard to the topic of function approximation, two contributions are proposed:

- A new deterministic data mining approach to distributing a given data set (input/output data) into two representative and balanced sets of roughly equal size to be used in function approximation problems. The distribution takes into account the variability of the data set with the purpose of allowing both a fair evaluation of learning's accuracy and to make reproducible machine learning experiments usually based on random distributions. The sets are generated using a combination of a clustering procedure, especially suited for function approximation problems, and a distribution algorithm which distributes the data set into two sets within each cluster based on a nearest-neighbor approach.
- A new deterministic model selection methodology with applications for incremental Radial Basis Function Neural Network (RBFNN) construction in time series prediction problems. Such model selection approach is a combined algorithm which takes advantage of balanced and representative training and validation sets obtained through the data distribution approach previously proposed. These balanced and representative sets are used in RBFNN

initialization, optimization and network model evaluation, improving, this way, the model prediction accuracy, getting small variance and bias, reducing the computation time spent in selecting the model and avoiding random and computationally expensive model selection methodologies based on *K-fold cross-validation* procedures.

1.3 Structure of the Dissertation

The dissertation is arranged into three parts. The first part contains the three chapters dedicated to the problem of integrating evidences from heterogeneous biological data sources to predict functional associations between proteins. Chapter 2 gives an introduction to the problem of predicting functional associations between proteins and motivates the development of the work done in this part of the dissertation. Chapter 3 introduces some theoretical basis such as a basic introduction to some biological concepts and an overview of bioinformatics, which constitutes the research field of this part of the dissertation. Then, the existing methodologies in the literature for the prediction of functional associations between proteins using single sources are reviewed. Next, we describe the approaches developed by other authors for the same prediction task when several data sources or evidences are integrated. Chapter 4 presents in detail the proposed MO-GP approach to developing pareto optimal classification rules for the integration of evidences from heterogeneous data sources to predict functional associations between proteins. An evaluation of the proposed approach and a comparison with other existing methodologies are also performed.

The second part of the dissertation is composed of three chapters, devoted to microarray data analysis pipeline. Chapter 5 gives an introduction to the different steps involved in microarray data analysis and motivates the development of the work done in Part II. Chapter 6 provides an overview to microarray technology, an explanation of the different steps involved in microarray data analysis pipeline and the description of some tools available in the literature to analyze microarray experiments. In Chapter 7, the microarray data analysis tool proposed in this part,

A^2**TOOL**, is presented and applied to the Chronic Lymphocytic Leukemia (CLL) data set.

The third part of this thesis contains the three chapters dedicated to the problem of distributing the original data set (input/output data) into two representative and balanced sets for function approximation tasks and the problem of model selection. Chapter 8 provides an introduction to these problems, motivating the development of the methodologies proposed in this part. Chapter 9 presents the proposed data mining approach for a distribution of a data set (input/output data) into two representative and balanced sets of roughly equal size to be used in function approximation problems. An evaluation of such methodology and a comparison with other existing ones are also provided. Finally the new deterministic model selection methodology for incremental RBFNN construction in time series prediction problems is presented in Chapter 10 with applications to several time series prediction problems and a comparison with other existing time series prediction techniques.

Finally, the main conclusions of the dissertation are discussed in Chapter 11. The publications supporting the contributions of this thesis are also enumerated.

In order to make easier the reading of this thesis, the following conventions were taken:

- Each part is composed of chapters. Since a set of different contributions are given in this dissertation, an introduction is provided in each part for its better understanding.
- Each chapter is divided into sections, containing subsections and sub- subsections. When a reference is given to any part of the text, this reference will include the number of the chapter, followed by the number of section, subsection and so on.
- Figures and tables are numbered by chapter. For example, figure 4.1.
- Mathematical expressions follow the same scheme that chapters and are showed by their numbers. For example, equation 9.1.
- The bibliographic references are indicated by using the surname of the authors followed by the year of publication if the work has at most 3 authors,

and by the surname of the main author followed by the abbreviation et al. if the work has more authors. Example:[Gonzalez et al., 2003],[Paul and Kumar, 2002].

Part I

Intelligent Systems for the integration of biological data to predict functional associations between proteins

Chapter 2

Introduction

A protein in an organism does not fulfill its function independently. The complicated cellular tasks of an organism frequently rely on associations among proteins [Xu et al., 2010],[Lees et al., 2011] that collaborate with each other as a functional module, which can be represented by a particular biological process or a specific pathway [Ravasz et al., 2002],[Hartwell et al., 1999]. For example, proteins are functionally related when sharing a substrate in a common metabolic pathway, when regulating each other transcriptionally or when participating in larger multiprotein assemblies [Mering et al., 2003].

In biology, however, functional relationships among proteins often transcend direct physical interactions [Lee et al., 2010]. Many proteins can be important for common biological processes without physically interacting. For example, proteins functioning in the same biosynthesis pathway, but at different biochemical steps, may never physically contact each other but are functionally associated because they act in the same biological process.

Thus, given the importance of functional associations between proteins to the explanations of cellular processes, the discovering of new functional relationships between proteins is one of the major goals of current biological studies and it is likely to be an important challenge for many years to come [Lysenko et al., 2011],[Lee et al., 2010],[Janga et al., 2011],[You et al., 2010],[Wang et al., 2009b],[Costello et al., 2009],[Wu et al., 2010],[Bradford et al., 2010], [Mostafavi and Morris,

2010],[Li et al., 2006],[Linghu et al., 2008],[Lee et al., 2004],[Lee et al., 2007] and [Hwang et al., 2005].

Several quantities of high-throughput biological data have become available in recent years to provide diverse insights of protein functions such as phenotypic profiles, gene expression microarrays, protein sequences, protein-protein interaction data, protein phylogenetic profiles and Rosetta Stone sequence, among others [Ray et al., 2009]. For most of them, various analytical techniques can be applied to extract evidences of functional correlation/similarity between proteins [Li et al., 2006] and the result of a search for such correlations is conveniently displayed as a graph where the nodes represent proteins and the edges between them expressing a correlation [Linghu et al., 2008]. The links are generally weighted, reflecting the degree of functional similarity based on some of the data sources described above. When the average number of edges per protein is sufficiently large, the result will be a network of associations, the so-called functional linkage network (FLN) as described in [Linghu et al., 2008] and also mentioned in [Lee et al., 2010],[Janga et al., 2011], [Costello et al., 2009],[Lee et al., 2004],[Linghu et al., 2009],[Lee et al., 2007].

An FLN can be constructed by using any of the data sources described above, however there are two main drawbacks to using each data type in isolation:

- Each of these distinct data sources provides a different, partly independent and complementary view of the whole genome [Hamid et al., 2009] and, thus, each single data source often can only reveal a certain perspective of the underlying complex biological mechanism [Li et al., 2006].
- Many single-source-based approaches are criticized for their noise and low reliability [Li et al., 2006] and thus may lack the degree of specificity requiered for an accurate prediction of functional associations [Bradford et al., 2010]

In recent years, the integration of evidences for functional association from heterogeneous data sources by means of machine learning procedures is believed to provide a means to overcome these drawbacks, and thereby benefit studies of genomic

functions, to cross-validate noisy data sets and to gain broad interdisciplinary views of large genomic and proteomic data sets [Hackl et al., 2010],[Lysenko et al., 2011],[Lee et al., 2010],[Janga et al., 2011],[You et al., 2010],[Re and Valentini, 2010],[Xiong et al., 2006], [Yao and Ruzzo, 2006],[Li et al., 2006],[Linghu et al., 2009],[Linghu et al., 2008],[Lee et al., 2004],[Lee et al., 2007],[Hamid et al., 2009], [Ray et al., 2009]. By combining multiple forms of evidences, it is expected to provide a complete genome-wide FLN and more accurate inferences of new functional relationships between proteins [Li et al., 2006]. Moreover, through this network, a further step can be taken: to infer individual proteins' functions on the basis of linked neighbors, that is, a decision rule for transfering the function of annotated proteins to unannotated proteins using direct methods [Lee et al., 2004],[Linghu et al., 2008],[Janga et al., 2011],[Costello et al., 2009],[Bradford et al., 2010], [Xiong et al., 2006],[Lee et al., 2007] although this topic is out of the scope of this work.

Several methodologies have successfully been applied in the literature for the integration of evidences from heterogeneous genomic and proteomic data such as Bayesian models [Xu et al., 2010],[Lee et al., 2004],[Lee et al., 2007],[Lee et al., 2010],[Wang et al., 2009b],[Costello et al., 2009], [Bradford et al., 2010],[Li et al., 2006],[Linghu et al., 2008],[Linghu et al., 2009], Artificial Neural Networks [Linghu et al., 2008],[Xiong et al., 2006], Fisher's method [Hwang et al., 2005], Decision Trees [Qi et al., 2006], Random Forests [Qi et al., 2006], Logistic Regression [Qi et al., 2006], Kernel methods [Lanckriet et al., 2004] [Wu et al., 2010] or Random walks on a graph [Lees et al., 2011], most of them being supervised-based learning methodologies that require a gold standard, which is a trusted representation of the functional information one might hope to discover. A gold standard generally consists of sets of samples grouped as either "positive" or "negative" examples. In spite of the increase in the prediction accuracy when several data sources or evidences are integrated, all of these methodologies suffer from at least one of the following problems:

- The computational cost of obtaining several FLNs with different levels of accuracy. From a biological viewpoint, it is desirable to obtain several FLNs

with different levels of accuracy since it is unknown, a priori, the partial preferences of a decision maker (i.e. the researcher) on the desired accuracy of the functional associations predicted. For supervised-based learning and in the case of prediction of functional relationships between proteins, the set of negative samples is much greater than the set of positive ones, so, to obtain several FLNs with different levels of accuracy, a cost-sensitive learning approach should be used, in which the varying costs of different misclassification types are considered. However, since such costs are usually unknown, different cost setups have to be explored within a reasonable range and, for each cost setup, a model must be learned to obtain an FLN with a given level of accuracy which, obviously, introduces extra computational cost.

- Interpretability. It is also desirable, from the biological point of view, for the model built by the data integration approach to be interpretable in the sense that (i) simple rules are provided to predict functional associations between proteins and (ii) the contributions of different types of evidences in the integration process toward predicting such functional associations are given.

Thus, in this part of the dissertation, it is proposed a new methodology for the integration of evidences from heterogeneous biological data sources to predict functional associations between proteins that tries to overcome these problems. The proposed methodology is a interpretable approach since it is based on simple IF-THEN rules that reflect the contributions of different types of evidences toward the prediction task. Through a multi-objective genetic programming (MO-GP) approach that takes into account simultaneously different misclassification costs, a set or pareto optimal IF-THEN classification rules are provided and each rule can be used to build an FLN with a given level of accuracy. This way, the MO-GP approach simultaneously evolves toward multiple pareto optimal rules and does not dramatically increase the learning time when compared to other integration methodologies where several costs have to be explored. Moreover, the decision maker does not have to specify partial preferences on the desired accuracy of the FLN, since covering the entire pareto, different FLNs are obtained, each one with a different level of accuracy.

This part of the thesis is structured as follows. Chapter 3 introduces some theoretical background for the understanding of this part such as a basic introduction to some biological concepts necessary for computer scientists and an overview of bioinformatics, which constitutes the research field of this part of the dissertation. Then, the methodologies present in the literature for the prediction of functional associations between proteins using single sources are reviewed. Next, we describe the approaches developed by other authors for the prediction of functional associations when several data sources or evidences are integrated. Chapter 4 presents the proposed MO-GP approach to developing pareto optimal classification rules for data integration.

Chapter 3

Prediction of functional associations between proteins: concepts and methodologies

The aim of this chapter is three-fold. First, to introduce some basic concepts of biology and an overview of bioinformatics (section 3.1). Second, to review the methodologies present in the literature for the prediction of functional associations between proteins using single sources (Section 3.2) and third, to describe the approaches developed by other authors for the prediction task when several data sources are integrated (Section 3.3).

3.1 Biology and Bioinformatics

In this section, the universal characteristics of all the living organisms are outlined. We briefly discuss the cellular diversity, the Deoxyribonucleic Acid (DNA) genetic material and the central dogma of gene expression. Then, a global vision of Bioinformatics is introduced.

3.1.1 Biology

3.1.1.1 Prokaryotic and Eukaryotic Cells

The biological universe consists of two types of cells: prokaryotic and eukaryotic. Prokaryotic cells consist of a single closed compartment that is surrounded by the plasma membrane, lacks a defined nucleus, and has a relatively simple internal organization (Figure 3.1). Bacteria, the most numerous prokaryotes, are single-celled organisms; the cyanobacteria, or blue-green algae, can be unicellular or filamentous chains of cells. Although bacterial cells do not have membrane-bounded compartments, many proteins are precisely localized in their aqueous interior, or cytosol, indicating the presence of internal organization [Lodish et al., 2007].

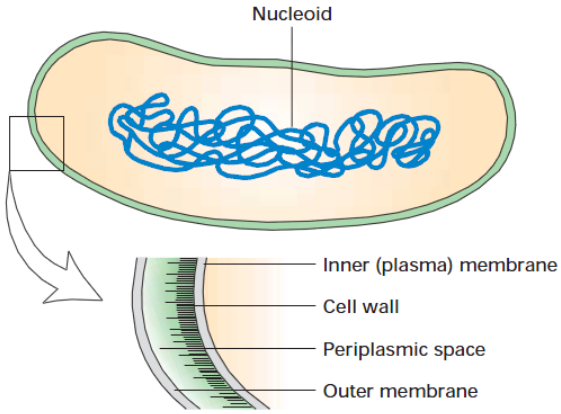


FIGURE 3.1: A Prokaryotic cell. The nucleoid, consisting of the bacterial DNA, is not enclosed within a membrane. E. coli and some other bacteria are surrounded by two membranes separated by the periplasmic space. The thin cell wall is adjacent to the inner membrane [Lodish et al., 2007].

Eukaryotic cells, unlike prokaryotic cells, contain a defined membrane-bound nucleus and extensive internal membranes that enclose other compartments, the organelles (Figure 3.2). The region of the cell lying between the plasma membrane and the nucleus is the cytoplasm, comprising the cytosol (aqueous phase) and the organelles. Eukaryotes comprise all members of the plant and animal kingdoms, including the fungi, which exist in both multicellular forms (molds) and unicellular forms (yeasts), and the protozoans (*proto*, primitive; *zoan*, animal), which are

exclusively unicellular. Eukaryotic cells are commonly about $10 - 100\mu m$ across, generally much larger than bacteria. A typical human fibroblast, a connective tissue cell, might be about $15\mu m$ across with a volume and dry weight some thousands of times those of an *E. coli* bacterial cell. An amoeba, a single-celled protozoan, can be more than 0.5 mm long. An ostrich egg begins as a single cell that is even larger and easily visible to the naked eye.

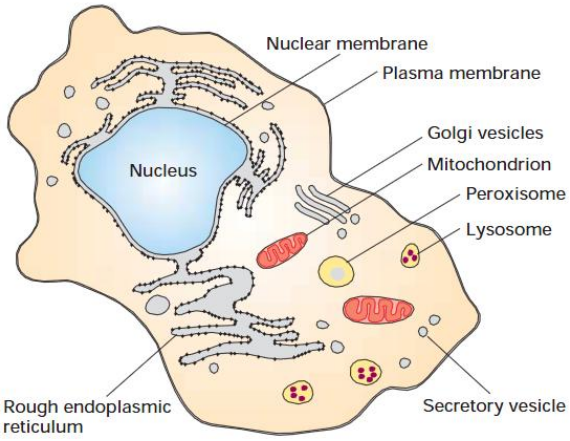


FIGURE 3.2: A Eukaryotic cell. Only a single membrane (the plasma membrane) surrounds the cell, but the interior contains many membrane-limited compartments, or organelles. The defining characteristic of eukaryotic cells is segregation of the cellular DNA within a defined nucleus, which is bounded by a double membrane. The outer nuclear membrane is continuous with the rough endoplasmic reticulum, a factory for assembling proteins. Golgi vesicles process and modify proteins, mitochondria generate energy, lysosomes digest cell materials to recycle them, peroxisomes process molecules using oxygen, and secretory vesicles carry cell materials to the surface to release them [Lodish et al., 2007].

All cells are thought to have evolved from a common progenitor because the structures and molecules in all cells have so many similarities. In recent years, detailed analysis of the DNA sequences from a variety of prokaryotic organisms has revealed two distinct types: the so-called ”true” bacteria, or eubacteria, and archaea (also called archaebacteria or archaeans). Working on the assumption that organisms with more similar genes evolved from a common progenitor more recently than those with more dissimilar genes, researchers have developed the evolutionary lineage tree shown in Figure 3.3. According to this tree, the archaea and the eukaryotes diverged from the true bacteria before they diverged from each other.

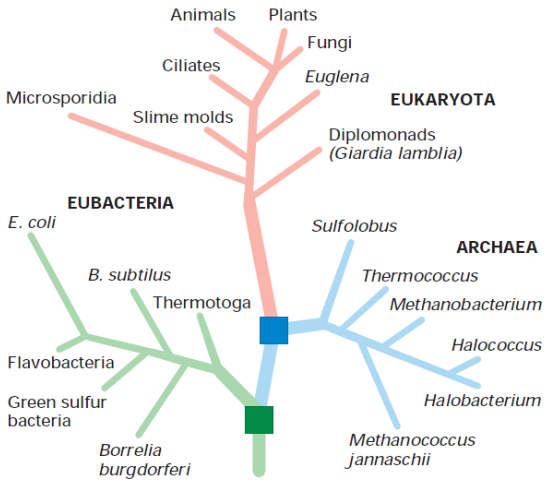


FIGURE 3.3: Evolutionary lineage tree from simple bacteria to complex mammals. This family tree depicts the evolutionary relations among the three major lineages of organisms. The structure of the tree was initially ascertained from morphological criteria: Creatures that look alike were put close together. More recently the sequences of DNA and proteins have been examined as a more information-rich criterion for assigning relationships. The greater the similarities in these macromolecular sequences, the more closely related organisms are thought to be. The trees based on morphological comparisons and the fossil record generally agree well with those based on molecular data. Although all organisms in the eubacterial and archaean lineages are prokaryotes, archaea are more similar to eukaryotes than to eubacteria (”true” bacteria) in some respects [Lodish et al., 2007].

3.1.1.2 The Deoxyribonucleic Acid (DNA)

The information about how, when, and where to produce each kind of protein is carried in the genetic material, a polymer called deoxyribonucleic acid (DNA). The three-dimensional structure of DNA consists of two long helical strands that are coiled around a common axis, forming a double helix. DNA strands are composed of monomers called nucleotides; these often are referred to as bases because their structures contain cyclic organic bases.

Four different nucleotides, abbreviated A (Adenine), T (Thymine), C (Cytosine), and G (Guanine), are joined end to end in a DNA strand, with the base parts projecting out from the helical backbone of the strand. Each DNA double helix has a

simple construction: wherever there is an A in one strand there is a T in the other, and each C is matched with a G (Figure 3.4).

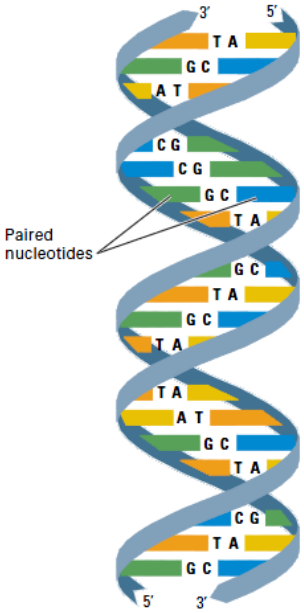


FIGURE 3.4: DNA: A diagram highlighting the helical strands around the outside of the molecule and the A-T and G-C base pairs inside [Lodish et al., 2007].

This complementary matching of the two strands is so strong that if complementary strands are separated, they will spontaneously zip back together in the right salt and temperature conditions. Such hybridization is extremely useful for detecting one strand using the other. For example, if one strand is purified and attached to a piece of paper, soaking the paper in a solution containing the other complementary strand will lead to zippering, even if the solution also contains many other DNA strands that do not match.

3.1.1.3 Gene expression: The Central Dogma

The genetic information carried by DNA resides in its sequence, the linear order of nucleotides along a strand. The information-bearing portion of DNA is divided into discrete functional units, the genes, which typically are 5000 to 100,000 nucleotides long. Most bacteria have a few thousand genes; humans, about 40,000.

The genes that carry instructions for making proteins commonly contain two parts: a coding region that specifies the amino acid sequence of a protein and a regulatory region that controls when and in which cells the protein is made.

The indirect route of information from DNA to proteins is known as the **central dogma** of molecular genetics and the cells use two processes in series to convert the coded information in DNA into proteins (Figure 3.5). In the first, called transcription, the coding region of a gene is copied into a single-stranded ribonucleic acid (RNA) version of the double-stranded DNA. A large enzyme, RNA polymerase, catalyzes the linkage of nucleotides into a RNA chain using DNA as a template. In eukaryotic cells, the initial RNA product is processed into a smaller messenger RNA (mRNA) molecule, which moves to the cytoplasm. Here the ribosome, an enormously complex molecular machine composed of both RNA and protein, carries out the second process, called translation. During translation, the ribosome assembles and links together amino acids in the precise order dictated by the mRNA sequence according to the nearly universal genetic code.

3.1.2 Bioinformatics

Bioinformatics is a interdisciplinary research area at the interface between computer science and biological science. A variety of definitions exist in the literature and on the world wide web, but one might define Bioinformatics as the union of biology and informatics: *bioinformatics* involves the technology that uses computers for storage, retrieval, manipulation, and distribution of information related to biological macromolecules such as DNA, RNA and proteins [Xiong, 2006].

Bioinformatics differs from a related field known as *computational biology*. Bioinformatics is limited to sequence, structural, and functional analysis of genes and genomes and their corresponding products. Computational biology encompasses all biological areas that involve computation. For example, mathematical modeling of ecosystems, population dynamics, application of the game theory in behavioral studies and phylogenetic construction using fossil records all employ computational tools, but do not necessarily involve biological macromolecules.

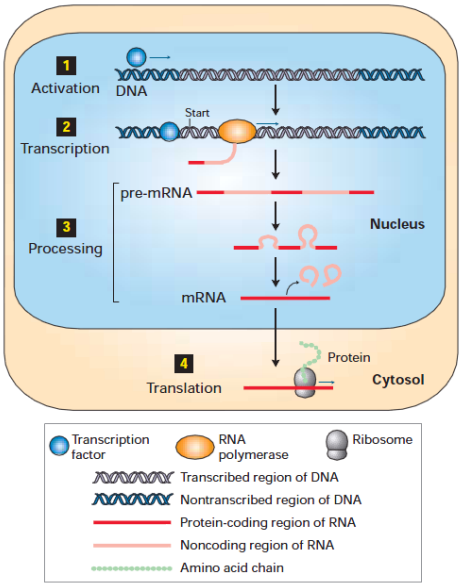


FIGURE 3.5: The coded information in DNA is converted into the amino acid sequences of proteins by a multistep process. Step 1: Transcription factors bind to the regulatory regions of the specific genes they control and activate them. Step 2: Following assembly of a multiprotein initiation complex bound to the DNA, RNA polymerase begins transcription of an activated gene at a specific location, the start site. The polymerase moves along the DNA linking nucleotides into a single-stranded pre-mRNA transcript using one of the DNA strands as a template. Step 3: The transcript is processed to remove noncoding sequences. Step 4: In a eukaryotic cell, the mature messenger RNA (mRNA) moves to the cytoplasm, where it is bound by ribosomes that read its sequence and assemble a protein by chemically linking amino acids into a linear chain [Lodish et al., 2007].

Bioinformatics [Xiong, 2006] consists of two subfields: the development of computational tools and databases and the application of these tools and databases in generating biological knowledge to better understand living systems. These two subfields are complementary to each other. The tool development includes writing software for sequence, structural and functional analysis, as well as the construction and curating of biological databases. These tools are used in three areas of genomic and molecular biological research: molecular sequence analysis, molecular structural analysis and molecular functional analysis.

The areas of sequence analysis include sequence alignment, sequence database searching, motif and pattern discovery, gene and promoter finding, reconstruction

of evolutionary relationships and genome assembly and comparison. Structural analysis include protein and nucleic acid structure analysis, comparison, classification and prediction. The functional analysis include gene expression profiling, protein-protein interaction prediction, protein subcellular localization prediction, metabolic pathway reconstruction and simulation.

Bioinformatics has not only become essential for basic genomic and molecular biology research, but is having a major impact on many areas of biotechnology and biomedical sciences. It has applications, for example, in knowledge-based drug design, forensic DNA analysis, and agricultural biotechnology. Computational studies of protein-ligand interactions provide a rational basis for the rapid identification of novel leads for synthetic drugs. Knowledge of the three-dimensional structures of proteins allows molecules to be designed that are capable of binding to the receptor site of a target protein with great affinity and specificity. This informatics-based approach significantly reduces the time and cost necessary to develop drugs with higher potency, fewer side effects and less toxicity than using the traditional trial-and-error approach. There is no doubt that bioinformatics is a field that holds great potential for revolutionizing biological research in the coming decades.

3.2 Prediction of functional associations between proteins by means of single sources

In recent years, several biological data sources have become available from which new evidences for functional relationships between proteins can be predicted. It is possible to distinguish different classes of evidences based on the type of data they use and we are going to describe some of the commonly used in the literature: experimental, sequence and literature-based evidences.

3.2.1 Experimental-based evidences

The first class of evidences for functional associations is based on high-throughput data, such as protein-protein interactions, microarray data or gene interactions.

3.2.1.1 Protein-protein interactions (PPI)

Physical interactions between proteins identified by mass spectrometry or one of the hybrid approaches are used to generate protein interaction maps on a large-scale which are used to infer functional associations between proteins [Janga et al., 2011]. It is based on the assumption that proteins that physically interact are likely to be functionally related [Janga et al., 2011], [Lee et al., 2004],[Linghu et al., 2008],[Linghu et al., 2009].

3.2.1.2 Microarray profiles (Co-exp)

Microarray data are used to measure the expression levels of large numbers of genes simultaneously or to genotype multiple regions of a genome. It has been demonstrated that genes with high co-expression across different conditions, are more likely to be functionally related than randomly chosen [Lee et al., 2004]. The strengh of co-expression is usually measured using the Pearson Correlation coefficient [Li et al., 2006],[Linghu et al., 2008],[Lee et al., 2007] or the Mutual Information [Li et al., 2006].

3.2.1.3 Genetic interactions (GI)

Genetic interactions are also used as a evidence for discovering functional linkage relationships between proteins [Linghu et al., 2008]. In these approaches, relationships between genes are constructed by linking gene pairs which show significantly reduced fitness when both genes are knocked out compared to when each gene is knocked out independently. These lethality assays are carried out on a high-throughput scale to construct genome-scale relationships [Janga et al., 2011].

Table 3.1 shows some data sources available from which the relationships between proteins can be extracted based on experimental data.

TABLE 3.1: Experimental data for generating functional linkage associations between proteins and their sources

Approach	**Data source**
PPI	HPRD (http://www.hprd.org)
	IntAct (http://www.ebi.ac.uk/intact)
	MINT (http://cbm.bio.uniroma2.it/mint/)
	BioGRID (http://www.thebiogrid.org)
	DIP (http://dip.doe-mbi.ucla.edu/dip/Main.cgi)
	MIPS (http://mips.gsf.de/proj/ppi)
Microarray profiles	GEO (http://www.ncbi.nlm.nih.gov/geo)
	SMD (http://genome-www5.stanford.edu)
	ArrayExpress (http://www.ebi.ac.uk/arrayexpress)
	caArray (http://caarraydb.nci.nih.gov/caarray)
	ATLAS (http://www.ebi.ac.uk/gxa/array/U133A)
	STRING (http://string.embl.de)
GI	BioGRID (http://www.thebiogrid.org)
	DRYGIN (http://drygin.ccbr.utoronto.ca)
	IM Browser (http://proteome.wayne.edu/PIMdb.html)
	SGD (http://www.yeastgenome.org/)

3.2.2 Sequence-based evidences

Several computational methods have been proposed for discovering associations between proteins from sequence data alone. Three different groups of sequence-based evidences are going to be described: genomic context, sequence similarity and protein domain sharing.

3.2.2.1 Genomic context evidences

These evidences include gene fusion, gene cluster or operon method and phylogenetic profiles and gene neighbor.

Gene fusion (GF) This approach, also known as the Rosetta Stone method, tries to detect the fusion of two genes into a single protein coding gene in one of the sequenced genomes and thereby links them as a strong functional association [Linghu et al., 2009].

Gene cluster or operon method (GC) Within bacteria, proteins of closely related function are often transcribed from a single functional unit known as an operon. Operons contain two or more closely spaced genes located on the same DNA strand. These genes are often in proximity to a transcriptional promoter that regulates operon expression [Bowers et al., 2004]. A series of genes are considered functionally linked if the nucleotide distance between genes in the same orientation was less than or equal to a specified distance threshold [Strong et al., 2003].

Phylogenetic profiles (PP) In this case, a vector of presence/absence of a protein across all the analyzed genomes is constructed and compared to identify proteins showing correlated profiles, as a measure of functional linkage. The rationale is that two proteins showing similar profiles are expected to be functionally related [Bowers et al., 2004],[Lee et al., 2004],[Linghu et al., 2008],[Linghu et al., 2009].

Gene neighbor (GN) This evidences is based on the assumption that if two genes are found to be chromosomal neighbors in several different genomes, a functional linkage can be inferred between the proteins they encode [Bowers et al., 2004][Linghu et al., 2009].

Relationships between proteins based on the genomic context evidences can be extracted from STRING [Szklarczyk et al., 2011] or Prolinks [Bowers et al., 2004] databases.

3.2.2.2 Sequence similarity (SS) evidence

The similarity between the aminoacid sequences of two given proteins can also be used as a evidence for functional relationships. It is based on the assumption that the more similar the sequences of two proteins are, the more likely the functional relationship is between them [Linghu et al., 2008]. To extract evidences for functional associations based on sequences similarity, the *blastp* program can be used

[Altschul et al., 1990]. This program finds regions of local similarity between protein sequences and calculates the statistical significance of matches, which can be used as a score of functional relationship between proteins.

3.2.2.3 Protein domain sharing (PDS)

The rationale behind protein domain sharing is that proteins containing the same protein domains tend to have similar function. Domains represent modular protein subunits that are often repeated in various combinations throughout the genome. To retrieve protein domain information, the Interpro database can be used [Mulder et al., 2005].

3.2.3 Literature-derived evidences

3.2.3.1 Text mining (TextM)

Information on functional associations between proteins are also available from Pubmed and other online resources. Evidences for protein associations can be obtained by searching for statistically significant co-occurrences between protein names, based on the assumption that the higher the frequency two proteins occur in the same sentence/paragraph/abstract or article the more likely their functional association [Bowers et al., 2004], [Linghu et al., 2009].

Examples of databases where functional relationships between proteins are extracted based on text mining are String [Szklarczyk et al., 2011] and Prolinks [Bowers et al., 2004].

3.2.3.2 Functional semantic similarity in biomedical ontologies

The Gene Ontology (GO) [Ashburner et al., 2000] is a controlled vocabulary used to describe various attributes of genes including their functions. GO organizes the information by means of three different ontologies: cellular component, biological process and molecular function. The ontologies of GO are structured as a Direct

Acyclic Graph (DAG), with terms as nodes in the graph and the relations between the terms as arcs. Where it is appropriate to talk about a parent-child relationship between nodes, parent refers to the node closer to the root of the graph and child to that closer to the leaf nodes; the parent would be a broader GO term and the child would be a more specific term. The terms themselves do not describe specific genes or gene products. Genes and gene products are annotated by collaborating databases in one or more terms at the most specific level possible, but are considered to share the attributes of all the parent nodes.

Various methods have sought to derive evidences for functional associations between proteins using their associated GO terms. For example, Xia et al. [Xia et al., 2006] and Rhodes et al. [Rhodes et al., 2005] used the following procedure using the biological process ontology: (1) identify all biological process terms shared by two proteins (2) count how many other proteins were assigned to each of the shared terms as well (3) identify the shared biological process term with the smallest count (Smallest Shared Biological Process, SSBP). In general, the smaller the SSBP count, the more specific the biological process term and, therefore, the higher degree of functional association between two proteins. The same procedure has also been used in [Linghu et al., 2009] to get functionally related proteins but using the molecular function ontology (Smallest Shared Molecular Function, SSMF) and the cellular component ontology (Smallest Shared Cellular Component, SSCC).

3.3 Prediction of functional associations between proteins by means of the integration of evidences from heterogeneous data sources

As stated in the previous section, there are several heterogeneous data sources from which evidences for functional relationships between proteins can be extracted. However, each source has bias and errors. On the other hand, it is unlikely, given the potential number of possible protein relationships, that two independent evidences for functional association or prediction methods will give rise to the same

false positive prediction. Due to this fact, it is expected an increase in both prediction accuracy and coverage when several independent approaches or evidences supporting the association between proteins are present. Due to these reasons, in recent years, the integration of evidences from multiple heterogeneous data sources by means of state-of-the-art procedures is believed to provide a means to overcome the drawbacks of using data sources in isolation. These approaches learn rules from existing knowledge, which can then be applied to make predictions on new data, revealing relationships between proteins not distinguishable within single datasets.

In the literature, many approaches have integrated multiple type of evidences to predict functional associations between proteins, namely, Bayesian integration, Artificial Neural Networks, Fisher's method, Decision Trees, Random forest classifiers, Logistic regression, Kernel methods and Random walks on a graph.

Most of these evidence/data integration methodologies require a gold standard, that is, they are supervised-based learning methodologies. A gold standard is a trusted representation of the functional information one might hope to discover. Such standard, coupled with an effective means of evaluation, can be used to assess the performance of a data integration method and serves as a basis for comparison among different integration approaches [Myers et al., 2006].

First of all, some issues about the integration of evidences from multiple data sources such as gold standards and the metrics for the evaluation of data integration methods, will be described (section 3.3.1.1). Then, some of the state-of-the-art data integration methodologies will be explained, focusing with more detail on the approaches used in this dissertation (section 3.3.2).

3.3.1 Gold standards and metrics for data integration evaluation

3.3.1.1 Gold standards

As previously stated, a gold standard is a trusted representation of the information one wants to discover, in our case, functional associations. There are several issues relating to gold standards such as their type or their size.

Existing gold standards A number of different gold standards to be used in data integration methodologies have been proposed in the literature. Each standard generally consists of sets of gene or protein pairs grouped as either ”positive” (Gold Standard Positive, GSP) or ”negative” (Gold Standard Negative, GSN) examples.

GSPs are usually derived from functional classification schemes that capture associations of genes or proteins with specific biological processes as reported in the literature [Lee et al., 2004],[Lee et al., 2007],[Linghu et al., 2008],[Linghu et al., 2009]. Such classifications are available from multiple sources including the Gene Ontology (GO) [Ashburner et al., 2000], KEGG [Ogata et al., 1999], FunCat [Ruepp et al., 2004], the Munich Information Center for Protein Sequences (MIPS) [Mewes et al., 2002] or the clusters of orthologous group (GOG) [Tatusov et al., 1997].

However, choosing a proper GSN set is a challenging problem, due to the difficulty in specifying genes/proteins that are not functionally related. In the literature, there are several examples to generate a GSN set:

- The GSN generated is based on different cellular compartments [Li et al., 2010],[Jansen and Gerstein, 2004]. However, localization data is likely not representative of ”typical” unrelated protein pairs, since a pathway or a biological process can be composed of proteins located in different compartments [Myers et al., 2006].

- Two genes/proteins are defined as belonging to the GSN set if they are not functionally related in the GSP set [Li et al., 2010],[Lee et al., 2004],[Linghu et al., 2009]. This method generates a large number of negative relationships, but some of them may represent potentially positive relationships. Instead of using the whole GSN set, a subset of negative protein pairs are selected randomly from that GSN set. The main drawback of this procedure is that the integration method must be run several times, each with a different subset of GSN to get an average evaluation of the data integration methodology.

- Wu et al. [Wu et al., 2010] proposed a method to generate a GSN set based on a graph approach in which a network is built using the defined GSP. The most distant gene pairs in the network can be defined as a GSN set.

Relative size of gold standard positive/negative sets Another issue about the gold standard set is the relative size between GSP and GSN sets [Myers et al., 2006]. The expected number of proteins involved in any particular biological process is a small percentage of the proteome. This imbalance is particularly problematic in data integration methods based on pairwise associations between proteins, where the expected number of protein pairs sharing functional relationships is an even smaller fraction of all possible protein combinations. For instance, of the 18 million possible protein pairs in yeast, it is expected that less than 1 million are functionally related. So, to get a representative measure of how well one could expect a data integration method to perform on whole-genome tasks, the ratio of positive to negative examples in the gold standard sets should match that in the application domain as closely as possible [Myers et al., 2006],[Linghu et al., 2009],[Lee et al., 2004],[Lee et al., 2007] instead of using equi-sized GSP and GSN sets as in [Wu et al., 2010] and [Linghu et al., 2008].

3.3.1.2 Metrics for data integration evaluation: ROC and precision-recall curves

Sensitivity-specificity and precision-recall analysis are two approaches to measuring the predictive accuracy of data from two classes given the class labels (positives and negatives). Sensitivity and specificity are typically computed over a range of thresholds and plotted with respect to one another, leading to the Receiver Operating Characteristic (ROC) curve and portrays the trade-off between sensitivity and specificity. Each value of the threshold yields one point on the curve by considering protein pairs whose association in the data exceeds the threshold value to be positive predictions and other pairs to be negative. Precision-recall analysis is done in the same way, but with precision (or Positive Predictive Value, PPV) replacing specificity. Precision is commonly used to evaluate data integration methodologies [Linghu et al., 2008],[Lee et al., 2004],[Lee et al., 2007],[Wu et al., 2010] [Linghu et al., 2009] because it is more informative than specificity when the size of the GSN set is much greater than the size of the GSP set and it rewards methods that

generate firm positive predictions, without regard to the accuracy of negative predictions, which are less helpful in guiding laboratory experiments [Myers et al., 2006].

Specificity, sensitivity, precision and recall are defined as:

$$\begin{aligned} Specificity &= \frac{TN}{TN + FP} \\ Sensitivity &= \frac{TP}{TP + FN} \end{aligned} \tag{3.1}$$

$$\begin{aligned} Precision &= \frac{TP}{TP + FP} \\ Recall &= \frac{TP}{TP + FN} \end{aligned} \tag{3.2}$$

where true positives (TP) and false positives (FP) are protein pairs associated by the data integration approach and annotated in GSP and GSN respectively. True negatives (TN) and False negatives (FN) are protein pairs not associated by the data integration methodology and annotated in GSN and GSP respectively.

Any point in both the ROC and Precision-Recall curves corresponds to a different trade-off between FPs and FNs. ROC and precision-recall curves can be summarized with a single statistic: the area under the curve (AUC). Precision-recall characteristics can be summarized with a similar measure which is usually referred as the AUPRC.

3.3.2 Data integration methods. State-of-the-art

In the literature, many approaches have integrated multiple source of evidences to predict functional associations between proteins (i.e. to construct the FLN network). In this section, some of the more relevant are described.

3.3.2.1 Bayesian integration

The Bayesian integration method is a probabilistic model and it is the most widely used strategy for the integration of evidences to predict functional associations between proteins [Wang et al., 2009b], [Bradford et al., 2010], [Linghu et al., 2008], [Linghu et al., 2009], [Lee et al., 2004],[Lee et al., 2007] and [Lee et al., 2010]. Given the GPS and GSN sets and following the Bayes theorem, the prior odds O_{prior} of finding a functional relationship between proteins can be calculated as:

$$O_{prior} = \frac{P_{GSP}}{P_{GSN}} \tag{3.3}$$

where P_{GSP} is the probability that a pair of proteins are functionally related within all the possible protein pairs while P_{GSN} stands for the possibility that a pair of proteins are not related. When considering the given n evidences (E), i.e. the n predictive features or data sets to be integrated, the posterior odds $O_{posterior}$ of an functional relationship can be computed as:

$$O_{posterior} = \frac{P(GSP|E_1, \ldots, E_n)}{P(GSN|E_1, \ldots, E_n)} \tag{3.4}$$

Let the likelihood ratio (LR) be defined as:

$$LR_{(E_1,\ldots,E_n)} = \frac{P(E_1, \ldots, E_n|GSP)}{P(E_1, \ldots, E_n|GSN)}, \tag{3.5}$$

which represents the probability of observing the values in the evidence data sets given that a pair of proteins are functionally related divided by the probability of observing the values given that the pair is not functionally related. That is, it is a score to quantify the degree of functional association between protein pairs.

The posterior odds of an interaction can be calculated as the product of the prior odds and the likelihood ratio :

$$O_{posterior} = O_{prior} \times LR_{(E_1,\ldots,E_n)} \tag{3.6}$$

If the evidence types to be integrated are independent (or non-redundant), the likelihood ratio can be calculated simply as the product of individual likelihood ratios from the respective evidence types:

$$LR_{(E_1,...,E_n)} = \prod_{i=1}^{n} LR(E_i), \tag{3.7}$$

which is called the naïve Bayesian model [Jansen and Gerstein, 2004],[Rhodes et al., 2005],[Linghu et al., 2009]. Equation 3.7 is equivalent to the following, where LLR represent log likelihood ratios:

$$LLR_{(E_1,...,E_n)} = LLR(E_1) + \ldots LLR(E_n), \tag{3.8}$$

this way, scores greater than zero indicate that the evidences tend to functionally link proteins, with higher scores indicating more confident linkages. Calculating LLR for each feature means adjusting different evidences to a common benchmark. That makes the different scores comparable even if they initially are of a different nature.

As the prior odds is a constant, the composite LLR corresponding to a specific biological evidence can be used to measure the predictive power or confidence degree for predicting functional relationships [Linghu et al., 2009]. A cutoff of log likelihood ratio (LLR_{cutoff}) is represented as an indicator whether a protein pair functionally relates (that is, yes if the composite $LLR_{(E_1,...,E_n)}$ is above LLR_{cutoff}, no if not). By filtering in the Naïve Bayesian model, the resulting protein pairs with $LLR_{(E_1,...,E_n)}$ above LLR_{cutoff} are identified as functionally related and used to construct the Functional Linkage Network (FLN). By varying the LLR cutoff, FLNs with different levels of accuracy can be obtained:

- A low LLR cutoff will correspond to a dense and relative noisy network in which most of the predictions are positives (TPs and FPs).
- A high LLR cutoff will correspond to a sparse and relative confident network in which most of the predictions are negatives (TNs and FNs).

Therefore, different LLR cutoffs will give several trade-offs between precision and recall values (different ratios of FP/FN).

Bayesian integration has several features that make it suitable for data integration. Each individual evidence is implicitly weighted according to its reliability, and hence it is easy to interpret the probability relationships for each evidence. Also, this method can accommodate missing data.

On the other hand, as previously stated, Naïve Bayesian integration assumes the evidence types to be integrated are independent or non-redundant, that is, the conditional probability of a evidence given a class (functionally related or non-functionally related) is assumed to be independent from conditional probabilities of other evidences given that class. Due to this assumption, the predictive power of Naïve Bayes may be reduced if evidence dependencies are present. We may have three choices when this dependence exists:

- To use the fully-connected Bayes network (eq. 3.5) which benefits from the advantages of the Naïve Bayes approach and does not assume independence between features. However, a fully-connected BN may be computationally expensive (learning a Bayes Network structure is an NP-hard problem) [Li et al., 2006].
- To make a correction for conditional dependencies [Elefsinioti et al., 2009].
- To use a heuristic modification to the strict Bayesian approach by incorporating the relative weighting of the data as well as capturing simple aspects of their relative independence [Lee et al., 2004],[Lee et al., 2007].

However, the relative independence of the various evidences can be difficult to estimate in the Bayesian framework [Lee et al., 2004]. Some other works have shown that correcting for conditional dependencies does not significantly improve the performance [Lu et al., 2005] and that the Naïve Bayes classifier can still be applied even when the independence assumption is not strictly satisfied [Linghu et al., 2009].

3.3.2.2 Artificial Neural Networks

Artificial neural networks (ANNs) [Smith, 1993] are signal processing systems that try to emulate the behavior of biological nervous systems by providing a mathematical model of numerous neurons connected in a network.

The various models of ANNs are designated by:

- The network topology, which refers to the structure of interconnections among the various nodes (neurons) in terms of layers and/or feedback or feedforward links.
- Node characteristics, which mainly specify the operations it can perform, such as summing weighted inputs incident on it and then amplifying or applying some aggregation operators on it.
- The updating rules, which may be for weights and/or states of the neurons.

Thus, ANNs can be viewed as weighted directed graphs in which artificial neurons are nodes and directed edges (with weights) are connections between neuron outputs and neuron inputs.

Most practical applications of ANNs involve a multilayer structure, such as the popular multilayer perceptron (MLP) model [Haykin, 1998], grouped into the category of Feedforward networks in which there are no loops in the architecture. MLPs consist of a number of interconnected processing elements (neurons) arranged in layers. Assuming, for example, a MLP with two hidden layers, the outputs of the neurons in the first hidden layer, second hidden layer, and output layer, respectively can be written as:

$$
\begin{aligned}
p_k &= f\left(\sum_{j=1}^{n} w_{kj}^{(p)} x_j\right), k = 1, ..., L; \\
q_k &= g\left(\sum_{j=1}^{L} w_{kj}^{(q)} p_j\right), k = 1, ..., S; \\
o_k &= h\left(\sum_{j=1}^{S} w_{kj}^{(o)} q_j\right), k = 1, ..., C
\end{aligned}
\tag{3.9}
$$

where x_j is the j-th input variable to the MLP (there are n inputs), p_k and q_k are the outputs of the k-th neuron in the first and second hidden layers respectively and o_k is the real-value output of the kth neuron in the output layer. In standard neural classifiers, the class returned is $\arg\max_k o_k, k = 1, \ldots, C$ [Zhou and Liu, 2006]. $f(.), g(.)$ and $h(.)$ are activation functions, such as the sigmoid function, a Gaussian function or a linear function, which transform the activation level of a neuron into an output signal. L and S are the number of neurons in the first and second hidden layers, respectively and C is the number of neurons in the output layer. The network size (i.e., L and S) and the network weights (i.e. $w_{kj}^{(p)}s$, $w_{kj}^{(q)}s$ and $w_{kj}^{(o)}$) are adjusted or trained to achieve a desired overall behavior of the network in terms of predicting the phenotype of samples in the training set. The back-propagation algorithm is commonly used to train MLP networks [Ressom et al., 2008].

With regard to the integration of evidences from heterogeneous biological data sources to predict functional associations between proteins, each input variable x_j is related to one evidence and the output layer has two nodes, one related to functional relationship and the other related to non-functional relationship between proteins [Linghu et al., 2008].

However, if FLNs with both different sizes and different levels of accuracy are to be constructed, a more sophisticated methodology must be followed in which the FP/FN cost ratio must be varied. This way, a FP/FN cost ratio close to 0 will give a classifier with most of the predictions positives (a dense and relative noisy FLN network) whereas a large FP/FN cost ratio will provide a classifier with most of the predictions negatives (a sparse and relative confident network). For this

purpose, a cost-sensitive neural network can be used [Zhou and Liu, 2006] in which different misclassification costs (FP/FN cost ratios) must be taken into account to get different trade-offs between precision and recall.

There are several approaches to train neural networks with different misclassification costs. Two of the most successfully applied in the literature are *threshold-moving* [Zhou and Liu, 2006] and the *minimization of the misclassification costs* [Kukar and Kononenko, 1998].

Threshold-moving This methodology moves the output threshold toward inexpensive classes so that examples with higher cost become harder to be misclassified. The approach trains an artificial neural network once as usual and the cost-sensitivity is introduced in the test phase.

Suppose there are C classes (i.e. C neurons in the ouput layer) and let $Cost[k, c]$, with $k, c \in \{1..C\}$, be the cost of misclassifying a k-class instance as a c-class instance with $Cost[k, k]$=0.

In *threshold-moving* and assuming that $\sum_{k=1}^{C} o_k = 1$ and $0 \leq o_k \leq 1$, instead of returning the class given by $\arg\max_k o_k$, the class returned is $\arg\max_k o_k^*$ where:

$$o_k^* = \mu \sum_{c=1}^{C} o_k \cdot Cost[k, c] \tag{3.10}$$

where μ is a normalization term such that $\sum_{k=1}^{C} o_k^* = 1$ and $0 \leq o_k^* \leq 1$.

Minimization of the misclassification costs This approach changes the error function that is being minimized by the artificial neural network. Instead of minimizing the squared error, as usual in multilayer perceptrons, the misclassification cost is minimized. Let us suppose a set of m samples $\{p_i\}_{i=1}^{m}$ belonging to C different classes. Assuming that the MLP output layer has C nodes, if $class(p_i) = c$, $c \in \{1, \ldots, C\}$, then the desired output for the c-th output neuron for sample p_i is $o_c^{'}(p_i) = 1$ and the desired output for the j-th output neuron for sample p_i is $o_j^{'}(p_i) = 0$, $\forall j \in [1, \ldots, C]$ and $j \neq c$. As defined in [Kukar and Kononenko,

1998], the error function is corrected by introducing the factor $\Phi[class(p_i), k]$ so that:

$$Error = \sum_{i=1}^{m} \cdot \sum_{k=1}^{C} ((o_k^{'}(p_i) - o_k(p_i)) \cdot \Phi[class(p_i), k])^2 \tag{3.11}$$

where $o_k(p_i) \in [0, 1]$. If $class(p_i) = c$, two cases are considered:

1. When dealing with output neuron k such as $k = c$, the difference between the desired and the real output of the neuron c is $o_c^{'}(p_i) - o_c(p_i) = 1 - o_c(p_i)$ and can be interpreted as a probability of misclassifying the training example p_i into any of the incorrect classes. In the ideal case $o_c(p_i) = 1$ and the misclassification probability is 0. The misclassification probability should be weighted with the expected misclassification cost of the class c, that is $CostVector[c]$.

2. For all other output neurons $k \in [1, \ldots, C]$ with $k \neq c$, the difference between the desired and the real output of the neuron k is $o_k^{'}(p_i) - o_k(p_i) = 0 - o_j(p_i) = -o_j(p_i)$ (actually, this expression is squared, so its sign can be ignored). This can be interpreted as a probability that the example p_i that belongs to the class c will be incorrectly classified into the class k. This probability should be weighted with the cost of misclassifying a c-class instance as a k-class instance, that is, $Cost[c, k]$.

The factor $\Phi[class(p_i), k]$ should therefore be defined as follows:

$$\Phi[class(p_i), k] = \begin{cases} CostVector[c], & \text{if } k = c \\ Cost[c, k], & \text{otherwise} \end{cases} \tag{3.12}$$

where $CostVector[c]$ represents the expected cost of misclassifying an example that belongs to the c-th class:

$$CostVector[c] = \frac{1}{1 - P(c)} \sum_{k=1}^{C} P(k) Cost[c, k], \text{ with } k \neq c \tag{3.13}$$

where $P(c)$ is an estimate of the prior probability that an example belongs to class c. For a two-class problem, $CostVector[c] = Cost[c, k]$.

As can be noticed, this approach trains a different artificial neural network for each FP/FN cost ratio, where the cost of FP is given by $Cost[-, +]$ and the cost of FN is given by $Cost[+, -]$.

3.3.2.3 Other approaches for data integration

There are also other methodologies used to integrate several evidences to predict functional associations between proteins. Here, some of them are described together with their advantages and disadvantages.

Fisher's method This methodology [Hwang et al., 2005] uses an optimization algorithm to minimize the number of false positives and false negatives and it makes no assumptions about the number of data sets to be integrated; on the contrary, it is for general purposes and may be applied to integrate from any existing technologies. The method is able to deal with the low overlap between source data sets and does not need trained or supervised based on experimental gold standard data sets of protein associations [Hackl et al., 2010]. Therefore, if only genomic context evidences (see section 3.2.2.1) are used, Fisher's predictions can be considered independent of the public repositories of protein associations [Lees et al., 2011]. A weighted version of Fisher method provides the ability to optimize contributions from each data source [Hwang et al., 2005].

Decision Trees Decision trees (DTs) [Quinlan, 1993] are trees where the nonleaf nodes are labeled with attributes (in our case, the data sources or evidences used in the integration process), the arcs out of a node are labeled with each of the possible values of the attribute and the leaves of the tree are labeled with classifications (in our case, functional relationship or non-functional relationship). A decision tree learns a classification function to predict the value of a dependent response (variable) given the values of the input attributes. DTs have been used in the prediction

of functional associations between proteins [Qi et al., 2006] and it is a useful approach because it presents all the rules used in the prediction task, showing which attribute combinations are most informative. However, in general, rules that are read directly off a decision tree are far more complex than necessary [Witten and Frank, 2005]. For example, when integrating diverse biological datasets for the prediction of co-complexed protein pairs, the decision tree obtained has tens of leaves [Zhang et al., 2004], difficulting the interpretability of the model.

Random Forest Random Forests (RFs) classifiers [Breiman, 2001] "grow" many Decision Trees simultaneously where each node uses a random subset of the data sources or evidences. They have also been applied to the integration of evidences from heterogeneous biological data sources for the prediction of functional relationships between proteins [Qi et al., 2006]. To classify, for example, a pair of proteins, the different evidences for that pair are subjected to analysis by each of the trees in the forest. Each tree provides (votes) a classification output (functional or non-functional relationship) and the forest chooses the classification based on majority vote over all the trees in the forest. RFs provide an efficient means of increasing performance with respect to decision trees, however they are computationally more expensive [Lin et al., 2004] and prone to overfitting for noisy datasets [Segal, 2004] as this might be a problem for the integration of evidences from heterogeneous biological data sources since it is well known that some of them are characterized for their noise and low reliability [Li et al., 2006].

Logistic regression Logistic Regression (LR) [Agresti, 2006] is a generalized linear statistical model that can predict a discrete outcome from a set of variables that may be continuous, discrete, dichotomous or a mixture of these. It applies maximum likelihood estimation after transforming the dependent variable into a logit variable (the natural log of the odds of the dependent variable occurring or not). In this way, logistic regression can be used to estimate the probability of a certain event, for example, functional relationships between proteins. This approach has been used to predict physical protein-protein interactions, co-complex relationship and functional relationships between proteins [Qi et al., 2006], although being outperformed by random forests.

Kernel methods Kernel-based statistical methods are also used for integrating evidences from heterogeneous biological data types [Qiu and Noble, 2008], [Lanckriet et al., 2004], [Wu et al., 2010]. One can describe kernel-based statistical learning approaches using two basic steps: (1) to choose the right kernel for each evidence or data set to build the so-called kernel matrix, which essentially constitutes similarity measures between pairs of entities (proteins, for example) and (2) to combine the kernels from the different data sources to give a complete representation of the available data for a given statistical task. Basic mathematical operations such as multiplication, addition, and exponentiation preserve properties of kernel matrices and hence produce valid kernels. Lanckriet et al. [Lanckriet et al., 2004] used kernel-based Support Vector Machine (SVM) methods to predict functional associations between proteins and Wu et al. [Wu et al., 2010] used a kernel-based Relevance Vector Machine (RVM) approach for the same task. Although one of the advantages of kernel methods is that kernel matrices can be defined for any type of data as well as their ability to incorporate prior knowledge through the kernel function [Qiu and Noble, 2008], one of the main drawbacks of kernel-based approaches is the prohibitive computational cost associated with SVMs/RVMs for problems with large data size, which is still a challenge [Wu et al., 2010].

Random walks on a graph A random walk on a graph describes the sequence of steps taken by a walker who moves from one node to a randomly selected adjacent node with a probability proportional to the weight associated with the edge connecting the two nodes [Lovasz, 1993]. They have been used in two ways for the integration of evidences from heterogeneous data sources. In the first, each data set is considered separately as a graph from which a random walk-based similarity is derived and used to rank the genes or proteins. A rank aggregation method is then used for the data integration step. The second integration approach consists in merging the different source data sets into one graph from which a random walk-based measure of similarity is derived and used for ranking genes. Although these approaches have been essentially used to predict disease genes from heterogenoeus biological data sources [Li and Patra, 2010a] [Li and Patra, 2010b], they could be applicable to the prediction of functional relationships between proteins.

Chapter 4

A multi-objective GP approach to developing pareto optimal IF-THEN classification rules for data integration

4.1 Motivation and goals

All the methodologies described in section 3.3 for integrating several evidences from heterogeneous data sources to predict functional associations between proteins have their advantages and disadvantages in their own right and we are going to discuss them in terms of the following issues: (i) the use of unbalanced GSP and GSN sets, (ii) the computational cost of obtaining several FLNs with different levels of accuracy and (iii) the interpretability of the model built by the data integration method:

- **Use of unbalanced GSP and GSN sets**. An important issue about the gold standard set is the relative size between GSP and GSN sets (see section 3.3.1.1): it is desirable that the ratio of positive to negative examples in

the gold standard matches that in the application domain as closely as possible, as stated in [Myers et al., 2006]. In the case of predicting functional associations between proteins, the expected number of protein pairs sharing functional relationships is a very small fraction of all possible protein combinations so, ideally, the GSP and GSN sets have to be unbalanced with the size of GSN much greater than the size of GSP. This imbalance will avoid misleading evaluations of different data integration methodologies, so that an evaluation on gold standards will be a representative measure of how well one could expect a method to perform on whole-genome data. Moreover, it is important to use all the information available instead of using equi-sized GSP and GSN sets as sometimes carried out in the literature due to the computational cost of data integration methodologies [Wu et al., 2010] [Linghu et al., 2008] when unbalanced GSP and GSN sets are used. All data integration methodologies described in the previous chapter, but Naïve Bayes, increase their computational cost when unbalanced GSP and GSN sets are used during learning and the computational requirements can be sometimes prohibitive such as in the case of kernel-based methods (SVMs or RVMs).

- **The computational cost of obtaining several FLNs with different levels of accuracy**. From a biological viewpoint, it is desirable to obtain several FLNs with different levels of accuracy, that is, from a dense and relative noisy network, in which most of the predictions are positives, to a sparse and relative confident network in which most of the predictions are negatives. It is unknown, a priori, the partial preferences of a decision maker (i.e. the researcher) on the accuracy of the functional associations predicted (i.e. the accuracy of the FLN), so that, a proper data integration approach must offer, without dramatically increasing the computational cost, several FLNs with different levels of accuracy. According to this fact, threshold-moving-based ANNs and Naïve Bayes are the only approaches that offer several FLNs with different levels of accuracy with low additional computational cost. In the case of threshold-moving-based ANNs, the cost-sensitivity is introduced once the artificial neural network has been trained so that different FLNs can be obtained at relative low computational cost. In Naïve Bayes, by varying the LLR cutoff, different quality FLNs are obtained, providing several

trade-offs between precision and recall values (different FP/FN ratios). This is because the classification cutoff used (LLR cutoff) is completely independent of the likelihood ratios learned during training. For the rest of methodologies, a cost-sensitive learning approach can be used in which different FP/FN cost ratios must be taken into account to build different FLNs from noisy networks to confident ones. This can be prohibitively time consuming, since several FP/FN cost ratios have to be explored within a reasonable range and, for each cost ratio, a model must be learned to obtain an FLN with a given accuracy level.

- **Interpretability**. It is also desirable, from the biological point of view, for the model built by the data integration approach to be interpretable in the sense that (i) simple rules are provided to predict functional associations between proteins and (ii) the contributions of different types of evidences in the integration process toward predicting such functional associations are given. This can greatly help us to gain insight of the underlying biological relationships. Decision Trees represent an appropriate approach for these issues [Zhang et al., 2004] [Qi et al., 2006], however they have been outperformed by other data integration methodologies [Qi et al., 2006] and usually offers a tree (a set of rules) with tens of leaves [Zhang et al., 2004] which difficult the interpretability of the results [Witten and Frank, 2005].

It must be noticed that Fisher's method does not need to be trained or supervised based on gold standards, however, it does not provide functional associations between proteins in the form of simple rules.

According to these issues, it is desirable a data integration approach that fulfills the following requirements: (i) handling of high unbalanced GSP and GSN sets whose size reflects the application domain as closely as possible; (ii) possibility of providing the decision maker with several FLNs with different levels of accuracy without dramatically increasing the computational cost and (iii) construction of an interpretable model with simple rules in which the contributions of different types of data or evidences integrated for the prediction task at hand are provided. The first requirement is not directly related to the data integration methodology, since it depends on the selection made for the GS sets, but it is indirectly related,

since some of the data integration methodologies described in section 3.3.2 force to balance the size of GSP and GSN sets due to the increase in the computational cost when high unbalanced sets are used. So, it is desirable that the data integration methodology is not significantly affected by large-scale gold standard sets in terms of computational cost.

So, in this chapter a new methodology for the integration of evidences from heterogeneous biological data sources to predict functional associations between proteins is presented. This methodology overcomes both the drawbacks given when using isolated data sources for building FLNs and the problems described for the current methodologies of data integration described above. The proposed approach is based on a multi-objective genetic programming algorithm to developing pareto optimal IF-THEN classification rules. These rules are interpretable and reflect the contributions of different types of evidences toward the discovery of functional associations. The MO-GP approach simultaneously evolves toward multiple pareto optimal rules and does not dramatically increase the learning time as compared to other integration methodologies where several FP/FN costs ratios have to be explored. This way, the decision maker does not have to specify partial preferences on the desired accuracy of the FLN, since covering the entire pareto, different FLNs are obtained, each one with a different level of accuracy (each rule is better than any other on at least one of two conflicting objectives, in our case, precision and recall).

This chapter is structured as follows. Section 4.2 reviews some basic concepts about classification rules, genetic programming and multi-objective evolutionary optimization. The architecture of the proposed approach is presented and explained in detail in section 4.3. In section 4.4, the results obtained through our approach when integrating ten evidences from heterogeneous data sources for the uncovering of functional associations between proteins in *Saccharomices Cerevisiae* (Baker's yeast) organism are presented. A comparison of the proposed methodology with respect to others present in the literature is also reported through a statistical analysis of the results. Moreover, the flexibility and the interpretability of the designed methodology together with its suitability to be executed in parallel architectures are explained in detail. Finally, some conclusions and future work are presented in section 4.5.

4.2 Classification rules and Genetic Programming

4.2.1 Classification rules

Classification rules are a simple and easily interpretable way to represent knowledge [Witten and Frank, 2005]. A classification rule has the following format:

$$IF < antecedent > THEN < consequent >$$

The rule antecedent contains a combination of conditions for the predicting attributes. Usually, a condition is composed of a binary relational operator ($=, \neq, >, <, \leq, \geq$) comparing the value of an attribute with a constant or another attribute. Usually conditions form a conjunction by means of the AND logical operator, but in general any logical operator can be used to connect elemental conditions. The rule consequent contains the value predicted for the class. This way, a rule assigns a data instance to the class pointed out by the consequent if the values of the predicting attributes satisfy the conditions expressed in the antecedent. From a data mining viewpoint, this kind of knowledge representation has the advantage of being intuitively comprehensible and interpretable for the user, as long as the number of discovered rules and the number of terms in the rule antecedent are not large [Espejo et al., 2010].

4.2.2 Evolutionary Algorithms. Genetic Programming

The evolutionary algorithm (EA) paradigm is based on the use of probabilistic search algorithms inspired by certain points in the Darwinian theory of evolution [Spears et al., 1993]. Several different techniques are grouped under the generic denomination of EA. The essential features shared by all EAs are [Espejo et al., 2010]:

1. The use of a population (a group) of individuals (candidate or partial solutions) instead of just one of them.

2. A generational inheritance method. Genetic operators are applied to the individuals of a population to give birth to a new population of individuals (the next generation). The main genetic operators are crossover (recombination) and mutation. Crossover swaps a part of the genetic material of two individuals, whereas mutation randomly changes a small portion of the genetic material of one individual.

3. A fitness-biased selection method. A fitness function is used in order to measure the quality of an individual. The better the fitness of an individual, the higher its probability of being selected to take part in the breeding of the next generation of individuals, thus, increasing the probability that its genetic material will survive throughout the evolutionary process.

The general scheme of an evolutionary algorithm is given in Fig. 4.1.

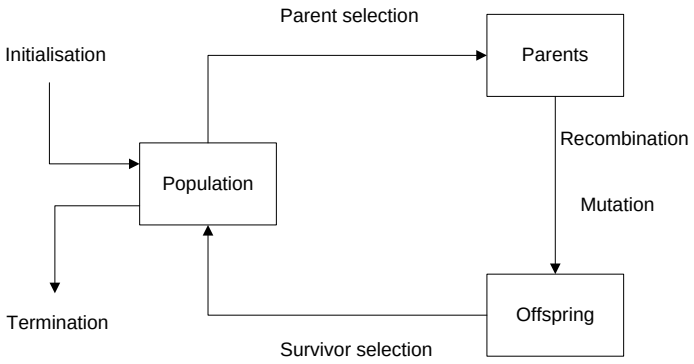


FIGURE 4.1: The general scheme of an evolutionary algorithm as a flow-chart (Introduction to Evolutionary Computing, [Eiben and Smith, 2008])

Genetic Programming (GP) is essentially considered to be a variant of EAs that uses a complex representation language to codify individuals. The most commonly used representation schema is based on trees. The original goal of GP was the evolution of computer programs but, nowadays, GP is used to evolve other abstractions of knowledge, like mathematical expressions or rule-based systems. GP individuals are usually seen as trees, where leaves correspond to terminal symbols (variables and constants) and internal nodes correspond to non-terminals (operators and functions). The set of all the nonterminal symbols allowed is called the *function*

set, whereas the terminal symbols allowed constitute the *terminal set*. Two conditions must be satisfied to ensure that GP can be successfully applied to a specific problem: sufficiency and closure. Sufficiency states that the terminals and nonterminals (in combination) must be capable of representing a solution to the problem. Closure requires that each function of the *nonterminal set* should be able to handle all values it might receive as input [Espejo et al., 2010].

GP offers a wide range of possibilities in the classifier induction task: GP is a search and optimization algorithm and, thus, it can be employed to search in the space of classifiers to find the best one. The fitness function of GP is used as the preference criterion that drives the search process and the flexibility of the GP representation allows it to employ many different kinds of models such as decision trees, discriminant functions or classification rules like the ones described in the previous section.

Regardless of the model used, the quality of classifiers must be measured by means of the fitness function. Usually, quality is based on accuracy, and it is often measured as the ratio between the number of correctly classified examples and the total number of examples, but other possibilities exist for measuring accuracy, such as precision, support, confidence, recall, sensitivity and specificity (see section 3.3.1.2).

4.2.2.1 Multiobjective evolutionary optimization. Non-dominated Sorting Genetic Algorithm II (NSGA-II)

In many problems, there are several goals which have to be optimized simultaneously. These goals are often conflicting, so that the optimization of one of the performance measures implies an unacceptably poor performance for other measures.

In this kind of problem, known as a multiobjective optimization (MO) problem, there is usually not a single solution, but instead a set of equivalent nondominated solutions, known as a Pareto front, composed of all the solutions where it is not possible to enhance some objectives without degrading some others. To clarify and for a set of multiple conflicting objectives $f_1, f_2, \ldots, f_M$ that have to be minimized

(or maximized) simultaneously, a solution x_1 is said to dominate another solution x_2 if x_1 is at least as good as x_2 on every objective and is better than x_2 on at least one objective. Formally, $\forall i = 1, 2, \ldots, M : f_i(x_1) \leq f_i(x_2) \wedge \exists i = 1, 2, \ldots, M : f_i(x_1) < f_i(x_2)$. A solution x is said to be *non-dominated* or *Pareto optimal* if there is no other feasible solution that dominates x. The set of all Pareto optimal solutions is referred to as the *Pareto front*.

EAs can be applied to MO problems easily and suitably, since different individuals in the population can search for different solutions in parallel. When EAs are applied to this kind of problem, the term employed is multiobjective optimization evolutionary algorithm (MOEA) (MOGP if the EA is a GP) [Espejo et al., 2010] [Deb et al., 2000]. For example, in [Zhao, 2007] a multi-objective optimization Genetic Programming system is used to obtain pareto optimal decision trees for classification, in which each decision tree is better than any other on at least one of two conflicting objectives, e.g. minimizing false negative rate vs minimizing false positive rate.

Several MOEAs have been proposed in recent years, considering SPEA2 [Zitzler et al., 2002] and NSGA-II [Deb et al., 2000] as the most representative ones. Briefly, NSGA-II has two key features [Gacto et al., 2008]:

- The fitness evaluation of each solution is based on Pareto ranking and a crowding measure. Each solution in the current population is evaluated in the following manner. First, Rank 1 is assigned to all non-dominated solutions in the current population. All solutions with Rank 1 are removed from the current population. Then, Rank 2 is assigned to all non-dominated solutions in the reduced current population and, again, all solutions with Rank 2 are removed from the reduced current population. This procedure is iterated until all solutions are removed from the current population. This way, a different rank is assigned to each solution where solutions with smaller ranks are viewed as being better than those with larger ranks. Among solutions with the same rank, the so called crowding measure criterion is taken into account. For a given solution, the crowding measure calculates the distance between its adjacent solutions with the same rank in the objective space. Less crowded solutions with larger values of the crowding measure are viewed as

being better than more crowded solutions with smaller values of the crowding measure.

- The elitist generation update procedure. When the next population is to be constructed, the current and offspring populations are combined into a merged population. Each solution in the merged population is evaluated using both the Pareto ranking and the crowding measure. The next population is constructed by choosing a specified number (i.e., population size) of the best solutions from this merged population.

4.3 A multi-objective GP approach to developing pareto optimal classification rules for the integration of evidences to predict functional associations between proteins

As previously stated, the approach proposed in this chapter is based on a multi-objective genetic programming (MO-GP) methodology to developing **P**areto **O**ptimal IF-THEN **C**lassification **R**ules (POCR). These rules are said to be Pareto optimal, in that each rule is better than any other on at least one of the conflicting objectives, in our case, precision and recall. The methodology satisfies the requirements for a data integration methodology:

- It can deal with large gold standard sets, so that high unbalanced GSP and GSN sets can be used.
- It provides a set of simple classification rules since the GP methodology can simultaneously evolve toward multiple alternative non-dominated solutions. Each rule in the pareto reflects the evidences integrated and is used to build an FLN (i.e. to predict functional associations between proteins) with a given level of accuracy. Covering the entire Pareto, different FLNs are obtained, each one with a different quality (trade-off between precision and recall) and, thus, different size. This way, the computational cost of obtaining different FLNs is dramatically reduced.

- Each FLN is based on a single rule. This means, that the FLN can be described in terms of an interpretable and comprehensive rule. Moreover, the rule provides the contributions of the different types of data or evidences used in the integration process toward predicting functional associations.

4.3.1 Classification rules for the integration of evidences to predict functional associations between proteins

In section 3.2, there are several data sources from which evidences for functional associations between proteins can be extracted. For example, when microarray data is used, it has been demonstrated that genes with high co-expression across different conditions, are more likely to be functionally related than randomly chosen [Lee et al., 2004]. So, the greater the co-expression value between two proteins is, the more likely a functional association between them exists.

In table 4.1 some of the evidences for functional association described in section 3.2 together with the assumption or hypothesis assumed for the evidence to predict such associations are shown. As can be observed, some evidences are continuous and others are discrete.

The problem of predicting functional associations between proteins is a binary classification problem, that is, the class predicted is either ”functional relationship” or ”non-functional relationship”. This would simplify the optimization process of the multi-objective genetic programming approach, since the rules to be optimized are the ones that provide the class predicted ”functional relationship”. The instances not covered by these rules, are assigned to the other class (”non-functional relationship”).

To justify the rules to be used, let us suppose that we are dealing with just a single data source or evidence, co-expression from microarray data (Co-exp). According to this evidence (see table 4.1, *Type* column), the greater the co-expression score is, the more likely the inferred functional linkage between proteins will be true positives. The following is an example of classification rule for predicting functional associations between proteins when Co-exp evidence is used:

TABLE 4.1: Evidences commonly used in the literature for predicting functional associations (see details in section 3.2) Co-exp, correlated gene expression; GF, gene fusion; PP, phylogenetic profile; GN, gene neighbor; SS, sequence similarity; TextM, text mining; SSMF, Smallest Shared Molecular Function (GO ontology); SSCC, Smallest Shared Cellular Component (GO ontology). *Type* column refers to the type of hypothesis assumed for the evidence to predict functional association between proteins: (a) *ascendant* and *descendant* mean that the greater and lower respectively the evidence score is, the more likely the functional relationship between two proteins is a true positive, (b) *existence* means that when the evidence score is *1* (existence), the more likely the functional relationship between two proteins is a true positive. *Domain* column refers to the continuous or discrete interval allowed for the evidence score

Evidence	**Description**	**Type**	**Domain**
Co-exp	Expression correlation from multiple microarray data	ascendant	$[a_{Co-exp}, b_{Co-exp}]$
PPI	Protein-protein interactions	existence	$\{0, 1\}$
GI	Genetic interactions	existence	$\{0, 1\}$
GF	Protein pairs fused into one single protein in other species	ascendant	$[a_{GF}, b_{GF}]$
PP	Protein pairs having correlated phylogenetic profiles	ascendant	$[a_{PP}, b_{PP}]$
GN	Gene pairs located close to each other along the chromosome	ascendant	$[a_{GN}, b_{GN}]$
SS	Protein pairs having similar aminoacid sequences	descendant	$[a_{SS}, b_{SS}]$
TextM	Co-ocurrence in PubMed abstracts	ascendant	$[a_{TextM}, b_{TextM}]$
SSMF	Protein pairs sharing same molecular function terms in GO	descendant	$[a_{SSMF}, b_{SSMF}]$
SSCC	Protein pairs sharing same cellular component terms in GO	descendant	$[a_{SSCC}, b_{SSCC}]$

$$\textit{IF Co-exp} \geq \alpha_{Co-exp} \textit{ THEN functional relationship}$$

where *Co-exp* is the co-expression score for two proteins and α_{Co-exp} is the co-expression value used as threshold for predicting functional associations (*Co-exp*, $\alpha_{Co-exp} \in [a_{Co-exp}, b_{Co-exp}]$). This classification rule will provide a curve with different values of precision and recall by varying α_{Co-exp} in $[a_{Co-exp}, b_{Co-exp}]$. When $\alpha_{Co-exp} = a_{Co-exp}$ all functional association predictions will be positives (true and false), providing high recall and low precision. On the other hand, when $\alpha_{Co-exp} = b_{Co-exp}$, most of functional association predictions will be negatives (true and false), providing low recall and high precision.

Since it is desired to integrate diverse evidences from heterogeneous data sources to infer functional linkages between proteins, the rules will be in the form of:

$$IF\ E_1 \geq \alpha_{E_1}\ AND/OR\ E_2 \leq \alpha_{E_2}\ AND/OR\ \ldots E_n \geq \alpha_{E_n}$$
$$THEN\ functional\ relationship$$

where n data sources or evidences E_i are being used (or integrated) to infer a functional linkage between proteins. Notice that each condition is made up of a binary relational operator comparing the score of an evidence E_i with a threshold α_{E_i}. The relational operator in a condition depends on the type of evidence used in that condition (see table 4.1). Evidences of type "ascendant" have the relational operator $\geq$ in the condition. Evidences of type "descendant", have the relational operator $\leq$ in the condition and evidences of type "existence" will have the relational operator =.

As can be observed, the rule antecedent contains a combination of conditions for the predicting evidences. The conditions can form a conjunction by means of the AND logical operators or a disjunction by means of the OR operator. For example, if all conditions form a conjunction, the rules will be in the form of:

$$IF\ E_1 \geq \alpha_{E_1}\ AND\ E_2 \leq \alpha_{E_2} \ldots\ AND\ \ldots E_n \geq \alpha_{E_n}$$
$$THEN\ functional\ relationship$$

where a functional relationship between proteins is inferred when it is supported by the n evidences used. This is not a valid approach, since all the evidences are considered for predicting a functional link and, due to the noisy nature of several sources, it is likely that a given relationship is supported by a (unknown) subset of evidences.

On the other hand, if all conditions form a disjunction, the rules will be in the form of:

$$IF\ E_1 \geq \alpha_{E_1}\ OR\ E_2 \leq \alpha_{E_2} \ldots\ OR\ \ldots E_n \geq \alpha_{E_n}$$
$$THEN\ functional\ relationship$$

This time, for predicting a functional relationship between proteins it is enough if the link is supported by at least one source, which is considered very permissive, assuming, again, the noisy nature of the data sources. This kind or rules will produce a very high number of false positives.

So, it seems that a proper rule will lie in between these two formats: the rule antecedent is made up of a set of compound conditions connected by a logical OR operator and each compound condition is made up of a set of single conditions connected by a logical AND operator:

$$\begin{aligned} &IF\ (E_1 \geq \alpha_{E_1}\ AND\ E_2 \leq \alpha_{E_2}\ AND\ \ldots\ AND\ E_k \geq \alpha_{E_k}) \ldots\ OR \\ &\ldots (E_{n-p} \leq \alpha_{E_{n-p}}\ AND\ \ldots\ AND\ E_{n-1} \geq \alpha_{E_{n-1}}\ AND\ E_n \geq \alpha_{E_n}) \\ &THEN\ functional\ relationship \end{aligned} \tag{4.1}$$

this way, each compound condition of the rule restricts the prediction of functional association to a subset of evidences at the same time. By connecting the compound conditions by a logical OR operator, we are also given the possibility to study several subsets of evidences to predict associations between proteins. This rule represents an entire solution for the classification problem at hand. For example, the following rule:

$$IF\ (Co\text{-}exp \geq 0.7\ AND\ PPI = 1)\ OR\ (GF \geq 0.5\ AND\ SS \leq 0.2)$$
$$THEN\ functional\ relationship$$

predicts functional relationship for a given pair of proteins if their microarray co-expression score is greater or equal than 0.7 AND they physically interact OR their gene fusion score is greater or equal than 0.5 and the similarity between their aminoacid sequences is lower or equal than 0.2. Otherwise, no functional relationship is predicted between such pair of proteins. In this example, it is assumed that the scores of the continuous evidences are normalized in some interval, for example [0,1].

However, the following rule:

$$IF\ (Co\text{-}exp \geq 0\ AND\ PPI = 1)\ OR\ (GF \geq 0.5\ AND\ SS \leq 0.2)$$
$$THEN\ functional\ relationship$$

is equivalent to:

$$IF\ PPI = 1\ OR\ (GF \geq 0.5\ AND\ SS \leq 0.2)$$
$$THEN\ functional\ relationship$$

since, the prediction of a functional associations between two proteins is supported enough when there is a protein-protein interaction between them (it does not matter the value of their microarray co-expression score) OR their gene fusion score is greater or equal than 0.5 AND the similarity between their aminoacid sequences is lower or equal than 0.2.

As can be observed, for evidences with score values in a continuous interval, if the threshold for that evidence in the condition takes the minimum possible value, a functional relationship is supported by that evidence regardless of its score value. This is fulfilled for evidences of type ”ascendant”. In the case of evidences of type ”descendant”, the threshold in the condition must take the maximum possible value. In consonance with this type of evidences, the relational operator of evidences whose score is discrete, such as PPI and GI, is changed to $\geq$. This way, when, for example, we have the following rule:

$$IF\ (Co\text{-}exp \geq 0.7\ AND\ PPI \geq 1)\ OR\ (GF \geq 0.5\ AND\ SS \leq 0.2)$$
$$THEN\ functional\ relationship$$

it predicts functional relationship for a given pair of proteins if their microarray co-expression score is greater or equal than 0.7 AND they physically interact OR their gene fusion score is greater or equal than 0.5 and the similarity between their aminoacid sequences is lower or equal than 0.2.

However, when the rule is:

$$IF\ (Co\text{-}exp \geq 0.4\ AND\ PPI \geq 0)\ OR\ (GF \geq 0.5\ AND\ SS \leq 0.2)$$
$$THEN\ functional\ relationship$$

this is equivalent to:

$$IF\ Co\text{-}exp \geq 0.4\ OR\ (GF \geq 0.5\ AND\ SS \leq 0.2)$$
$$THEN\ functional\ relationship$$

since the prediction of functional association between two proteins is supported enough when their co-expression score is greater or equal than 0.4 (it does not matter whether they interact physically or not) OR their gene fusion score is greater or equal than 0.5 and the similarity between their aminoacid sequences is lower or equal than 0.2. Notice that the domain of PPI is still $\{0, 1\}$. The only difference is related to the relational operator, which is changed from = to $\geq$, since the greater the value for PPI evidence score (a true physical interaction, 1), the more likely a functional association between a pair of proteins is a true positive. This way, if the threshold for PPI evidence in the condition takes the minimum possible value

(zero) a functional relationship is supported by PPI regardless of its score value. The same applies to Gene Interaction (GI).

4.3.2 Proposed MO-GP approach

As previously stated, a multi-objective genetic programming approach is used to simultaneously evolve multiple alternative non-dominated classification rules. Each rule in the pareto represents an entire solution for the prediction of functional associations between proteins so that an FLN with a given level of accuracy is built. Covering the entire pareto, different FLNs are obtained, each one with a different quality (trade-off between precision and recall) and, thus, different network size.

In the following subsections, the components of the multi-objective genetic programming approach are described in detail, namely, solution encoding or representation of individuals (section 4.3.2.1), population (section 4.3.2.2), evaluation or fitness function (section 4.3.2.3) and the genetic operators: recombination and mutation (section 4.3.2.4). Although the size of the individuals is fixed in the GP approach, some evidences present in the final pareto optimal classification rules may not be necessary for the prediction task. So, once the optimization of the MO-GP is done, a post-procedure is applied to detect irrelevant evidences in the set of pareto rules (section 4.3.2.5).

4.3.2.1 Solution encoding

The rules to be used in the multi-objective Genetic Programming approach will be the one described in the expression 4.4.3.3:

$$IF\ (E_1 \geq \alpha_{E_1}\ AND\ E_2 \leq \alpha_{E_2}\ AND\ \ldots\ AND\ E_k \geq \alpha_{E_k}) \ldots\ OR$$
$$\ldots (E_{n-p} \leq \alpha_{E_{n-p}}\ AND\ \ldots\ AND\ E_{n-1} \geq \alpha_{E_{n-1}}\ AND\ E_n \geq \alpha_{E_n})$$
$$THEN\ functional\ relationship$$

where $\alpha_{E_i} \in [a_{E_i}, b_{E_i}], \forall i = 1, \ldots, n$ and the rule antecedent is made up of a set of compound conditions connected by a logical OR operator and each compound condition is made up of a set of single conditions connected by a logical AND operator. This rule can be represented naturally as a tree in the MO-GP approach to codify individuals. For example, the following rule

$$IF\ (E_1 \geq 0.3\ AND\ E_2 \leq 0.7)\ OR\ (E_4 \geq 0.1\ AND\ E_7 \geq 0.9)\ OR$$
$$(E_2 \leq 0.6\ AND\ E_8 \geq 0.55)\ THEN\ functional\ relationship$$

is composed of three compound conditions linked by the OR operator and each compound condition is made up of two conditions linked by the AND operator. This rule is represented as a tree in Fig. 4.2.

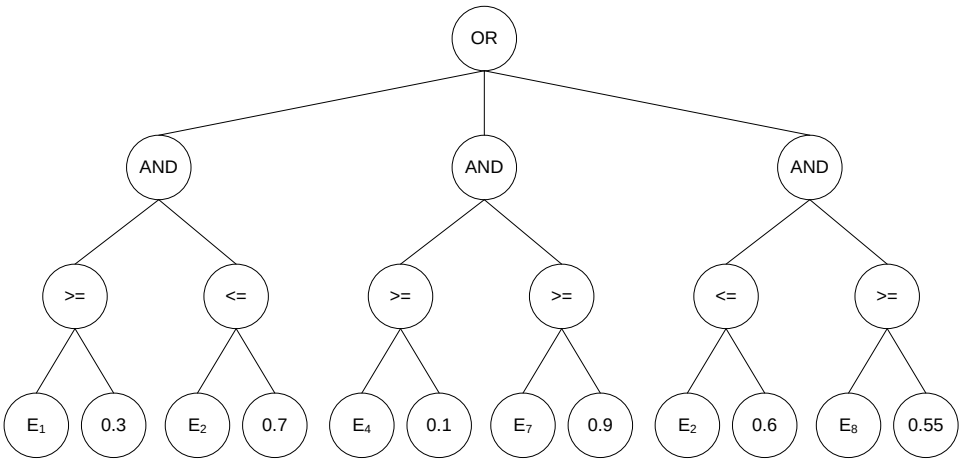


FIGURE 4.2: Example of an individual in the MO-GP approach

As can be noticed from Fig. 4.2, the rule consequent is not encoded into the genetic material of the individual, since it is a binary or two-class problem. The instances not covered by this rule, are assigned to the other class ("non-functional relationship"). An individual represents an entire solution to the prediction problem of functional relationships between proteins.

In GP, the user must specify all the functions, variables and constants that can be used as nodes in a tree. Functions, variables and constants which require no

arguments become the leaves of the tree and are called *terminals*. Functions which require arguments form the branches of the trees are called *non-terminals*. The set of all terminals is called the *terminal set* and the set of all non-terminals is called *non-terminal set* or sometimes called *function* [Koza, 1992].

For our individual representation (Fig. 4.2), the terminal set consists of the different evidences for functional associations E_i and the threshold values α_{E_i}. The function or non-terminal set consists of logical operators (AND,OR) and relational operators (”≤”,”≥”).

Next, we specifiy syntactical and semantic constraints associated with the individual representation. It should be stressed that some constraints are used in the generation of individuals in the initial population and the production of new individuals via crossover or mutation (sections 4.3.2.2 and 4.3.2.4).

In conventional GP systems, the property of closure must be satisfied [Koza, 1992], which means that all the functions (non-terminals) must accept arguments of a single data type (i.e. float) and return values of the same data type. This means that all non-terminals return values that can be used as arguemnts for any other non-terminal or function. This property is satisfied, for instance, if the *function set* contains only mathematical operators (like +,-, /,*) and all terminal symbols are real-valued variables or constants. However, in a data mining scenario, the situation is more complex, since one wants to mine a data set with a mixing of logical and relational operators.

Thus, the individual representation includes several constraints useful for data mining applications, such as classification rules. First, for each function of the *function set*, it is specified what the data types valid for the input arguments and the output of the function are. As said before, the *function set* of the GP consists of logical operators (AND,OR) and relational operators (”≤”,”≥”). The valid data types for the input arguments and output of these operators as well as the arity (i.e. the number of input arguments is takes) are shown in table 4.2. This table refers, without loss of generalization, to the individual described in Fig. 4.2.

The data type restrictions specified in table 4.2 naturally suggest the individual representation based on hierarchy of functions. At the deepest level in the tree, the

TABLE 4.2: Valid data types for each function's input arguments and output in a GP individual

Function	**Arity**	**Input arguments**	**Output**
OR	3	(Boolean,Boolean,Boolean)	Boolean
AND	2	(Boolean,Boolean)	Boolean
"≤","≥"	2	(real,real)	Boolean

leaf nodes are terminal symbols (evidences and their thresholds). One level up (see Fig.4.2), there are internal or non-terminal nodes containing relational operators. Each one of these nodes has two nodes as offspring, one of them containing an evidence and the other containing a threshold, both of them in the domain of that evidence. If the type of evidence in question is "ascendant" or "existence", then the internal node is "≥". On the other hand, if type of evidence is "descendant", then the internal node is "≤". In any case, the output of such an internal node is a Boolean value (yes or no). Notice that the relational operator, the evidence and the threshold form a single condition in the rule.

One level up, the internal nodes contain only AND operators and in the highest level, there is a unique node (or root node) which contains an OR operator. Therefore, the output of the tree (i.e. the output of the root node) will be a boolean value, indicating whether a pair of proteins satisfies the rule antecedent encoded in the individual. Notice that an AND node cannot be the ancestor of an OR node. This way, the individuals represent a set compound antecedents in disjunctive normal form (DNF) - i.e. an individual consists of a logical disjunction of compound conditions, where each compound condition is a logical conjunction of conditions (evidence-relational operator-threshold).

To summarize, the hierarchical individual representation has the following syntactic constraints:

- If a terminal node is an evidence, it shares its parent with another terminal node which is the related threshold for that evidence.
- If a terminal node is a threshold, it shares its parent with another terminal node which is the related evidence for that threshold.

- A terminal node can have as parent only a relational operator: ”≤” if the terminal node (evidence or threshold) is of type ”descendant” and ”≥” if the terminal is of type ”ascendant” or ”existence”.
- An internal node containing a relational operator (”≥”,”≤”) must have as parent only an AND logical operator.
- An internal node containing an AND operator must have as parent only an OR operator.
- An internal node containing an OR operator does not have any parent, it is the root node.

Notice that these constraints, implicitly, ensure that the depth of a tree has exactly four levels:

- The first level contains only the root node (OR node).
- The second level holds only nodes with AND logical operators.
- The third level contains only nodes with relational operators (≤ or ≥).
- The fourth level holds terminal nodes with either evidences or thresholds related to evidences.

However, the above constraints do not restrict the number of nodes in the second level and the number of nodes in the third level. This makes it possible to use rules with any number of compound conditions (i.e. any number of nodes in the second level of the tree) and any number of single conditions for each compound condition (i.e. any number of nodes in the third level).

To simplify the MO-GP procedure, the size of the individuals (the number of nodes in the second and third levels) is fixed. However, as previously described, the following rule

$$\textit{IF (Co-exp} \geq 0 \textit{ AND PPI} \geq 1) \textit{ OR (GF} \geq 0.5 \textit{ AND SS} \leq 0.2)$$
$$\textit{THEN functional relationship}$$

is equivalent to:

$$IF\ PPI \geq 1\ OR\ (GF \geq 0.5\ AND\ SS \leq 0.2)\ THEN\ functional\ relationship$$

which is shorter than the first case. However, the MO-GP does not remove the condition *Co-exp* ≥ 0 from the rule during the optimization process for simplicity. Instead, a post-procedure (a pruning approach) method is run once the MO-GP has finished (see section 4.3.2.5) to remove useless conditions. If conditions are removed from rules, the final pareto classification rules may have variable length. To increase the probability of obtaining final rules of variable length, the domain of the threshold parameter can be modified. For example, assuming that *Co-exp* $\in [0, 1]$ the following rule:

$$IF\ (Co\text{-}exp \geq 1.1\ AND\ PPI \geq 1)\ OR\ (GF \geq 0.5\ AND\ SS \leq 0.2)$$
$$THEN\ functional\ relationship$$

is equivalent to:

$$IF\ GF \geq 0.5\ AND\ SS \leq 0.2\ THEN\ functional\ relationship$$

since $\alpha_{Co-exp} \in [0, 1+\vartheta]$. The whole compound condition *Co-exp* ≥ 1.1 *AND PPI* ≥ 1 is removed from the rule, since there are no protein pairs whose co-expression score is greater or equal than 1.1 (*Co-exp* $\in [0, 1]$) and, at the same time, physically interact.

So, by extending the domain of thresholds α_{E_i} to values out of the domain of the related evidence, the probability of obtaining pareto optimal rules of variable length after the pruning approach is higher.

To clarify things, when an individual is created and during the optimization of the MO-GP approach:

- $\alpha_{E_i} \in [a_{E_i}, b_{E_i}+\vartheta_i], \vartheta_i > 0$ when the type of evidence is "ascendant" and $E_i \in [a_{E_i}, b_{E_i}]$ (see table 4.1). When $\alpha_{E_i} = b_{E_i} + \vartheta_i$ it means that the threshold is

out of the domain of its related evidence $[a_{E_i}, b_{E_i}]$. In this case, the threshold value is defined as *strict*, since there are no protein pairs whose evidence score is greater or equal than $b_{E_i} + \vartheta_i$. On the other hand, when $\alpha_{E_i} = a_{E_i}$, the threshold value is defined as *permissive*, since all protein pairs have evidence score greater or equal than a_{E_i}. Evidences of type "existence", such as PPI and GI will be treated as evidences of type ascendant in which their related thresholds take a value from a discrete range $\{a_{E_i}, b_{E_i}, b_{E_i} + \vartheta_i\}, \vartheta_i > 0$.

- $\alpha_{E_i} \in [a_{E_i} - \vartheta_i, b_{E_i}], \vartheta_i > 0$ when the type of evidence is "descendant" and $E_i \in [a_{E_i}, b_{E_i}]$ (see table 4.1). When $\alpha_{E_i} = a_{E_i} - \vartheta$ it means that the threshold is out of the domain of its related evidence $[a_{E_i}, b_{E_i}]$. In this case, the threshold value is defined as *strict*, since there are no protein pairs whose evidence score is lower or equal than $a_{E_i} - \vartheta$. On the other hand, when $\alpha_{E_i} = b_{E_i}$, the threshold value is defined as *permissive*, since all protein pairs have evidence score lower or equal than b_{E_i}.

This way, a variable size of the individuals (length of rules) is introduced implicitly in the MO-GP by means of the modified domain for the thresholds although, again, the individual size is fixed during optimization.

4.3.2.2 Initial population

In the previous section, the tree structure of an individual and their syntactical constraints have been described. The initial population is composed of a set of individuals, each representing an entire solution for the classification task.

The are two things one has to decide before the initial population is created:

1. The number of compound conditions linked by the OR operator or, in other words, the number of nodes at level 2 in the tree containing the logical operator AND (see Fig.4.2).

2. The number of single conditions linked by the AND operator for each compound condition, that is, the number of nodes at level 3 in the tree containing relational operators ("≥","≤").

The more the number of nodes at level 2 and 3 are, the more complex the individual is (the classification rule will be less interpretable). The number of compound conditions and the number of single conditions for each compound condition is the same for all the individuals of the population.

Thus, once the number of nodes in levels 2 and 3 has been chosen, for each individual of the population, the two offspring of nodes at level 3 are created. For each internal node at level 3 (a relational node) an evidence E_i and a threshold value α_{E_i} are created randomly as offspring. The constraints for individuals have already been described in section 4.3.2.1.

There is a constraint of uniqueness of an evidence in a given compound condition of the rule when an individual is created. This means that a given evidence or data source must be unique in a given compound condition, but can be present in another compound condition of the same rule (i.e. must be unique in a subtree rooted at an AND node).

4.3.2.3 Fitness evaluation and objective trade-offs

The fitness function is used in order to measure the quality of an individual. In a MO-GP system, the fitness is assigned on the basis of its relative non-dominance and for NSGA-II, it is based on Pareto ranking and crowding measure (see section 4.2.2.1). In our MOGP approach, we use the following two-objective problem:

$$\text{Maximize } f_1(x_i) \text{ and } f_2(x_i)$$

where $f_1(x_i)$ and $f_2(x_i)$ are the precision and recall respectively of training patterns by the *i-th* individual (x_i), representing a classification rule. (See section 3.3.1.2 for a definition of precision and recall).

Therefore, the goal of the MO-GP approach is to maximize two conflicting objectives at the same time, precision and recall, to obtain Pareto optimal classification rules in which each rule (individual) is better than any other on at least one of two conflicting objectives. A similar approach has been followed in [Zhao, 2007] to developing Pareto optimal decision trees in other type of classification problems.

4.3.2.4 Genetic operations: recombination and mutation

There are two major genetic operations: recombination (or crossover) and mutation.

Recombination Recombination in GP creates offspring by swapping genetic material among the selected parents. In technical terms, it is a binary operator creating two child trees from two parent trees. The most common implementation is *subtree crossover*, which works by interchanging the subtrees starting at two randomly selected nodes in the given parents [Eiben and Smith, 2008].

Nevertheless, in the NSGA-II implementation we used in this dissertation, a single offspring is created from two parent trees given, so, a modified version of *subtree crossover* is used as follows.

1. Select the level of the subtree to be modified. In our proposal, we offer two types of subtrees: a subtree whose root node contains an AND operator or a subtree rooted in a node containing a relational operator. So the type of subtree is selected with equal probability (i.e. it is chosen between level 2 or 3 of the individual). This way we avoid to choose internal nodes that contain relational operators with more probability.

2. Select randomly a crossover point (node) in the level chosen in the previous step. This node is selected in one of the parent individuals, here called the first parent.

3. Select randomly a node in the same level in the other parent individual (the second parent).

4. The crossover is performed by replacing at the crossover point in the first parent, the subtree rooted at the selected point in the second parent.

This way, it is ensured that the crossover always produces valid individuals and guarantees that all tree nodes have inputs and output of a valid data type as specified in table 4.2. Moreover, the crossover provides individuals with exactly the

same tree structure (the same number of levels in the tree). This will avoid the phenomena known as *bloat* in GP, in which the average tree size is growing during the GP run [Eiben and Smith, 2008].

In Fig. 4.3, an example of a recombination is shown. Note that the individuals in this example has only two AND nodes.

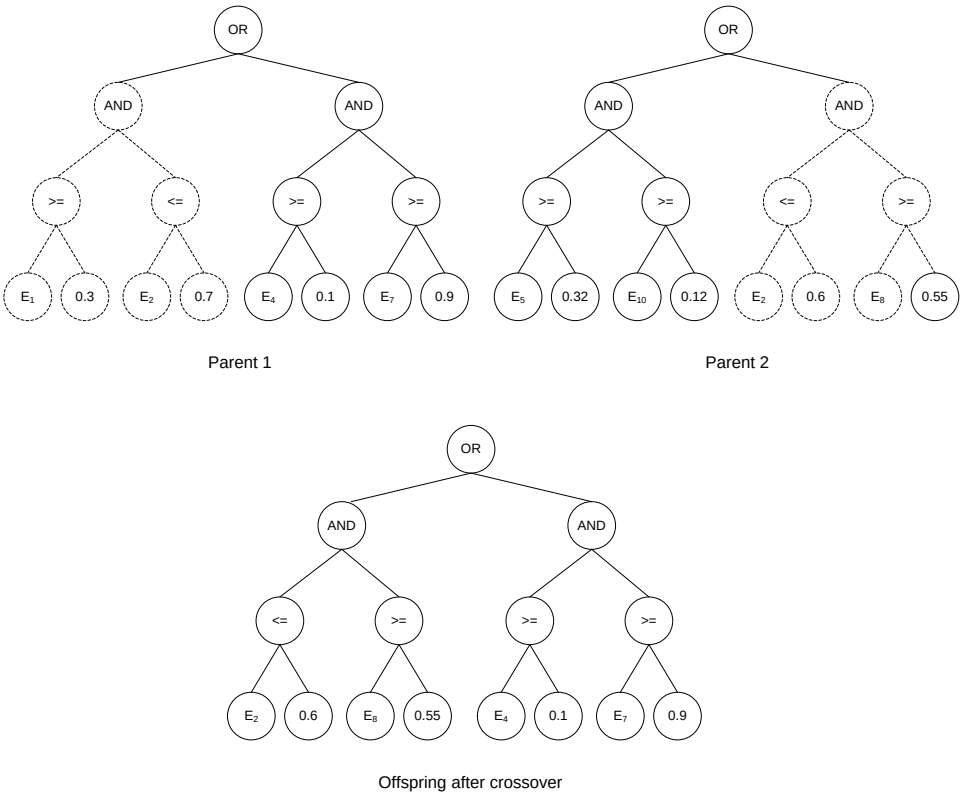


FIGURE 4.3: Example of recombination. 1) Level 2 (internal nodes that contain AND operators) is randomly chosen, 2) The first node of Level 2 is randomly selected in the first parent, 3) The second node of Level 2 is randomly chosen in the second parent, 4) The subtree rooted at the node selected in the second parent is used to replace the subtree rooted at the crossover point in the first parent

Mutation The task of mutation in GP consists in creating a new individual from an old one through some small random variation. The most common implementation works by replacing the subtree starting at a randomly selected node by a randomly generated tree.

In our case, the mutation procedure in an individual is described as follows. Select, with equal probability, among levels 2, 3 and 4 of an individual. Three cases are possible:

- If the selected level is 4, we randomly select a leaf node containing a threshold. That threshold is randomly changed according to its domain.
- If the selected level is 3, an internal node of this level which contains a relational operator, is randomly selected and the subtree rooted at this selected node is randomly generated. This means that its two offspring (an evidence and its related threshold) are randomly changed. It must be taken into account that the relational operator may also change according to the assumption of functional relationship related to the new evidence selected (see table 4.1).
- If the selected level is 2, an internal node of this level, which contains an AND operator, is randomly selected and the subtree rooted at this selected node is randomly generated. The procedure previously described when the selected level is 3 is applied for each of the offspring of the AND node.

In Fig. 4.4, an example of a mutation is shown. Note that the individual in this example has only two AND nodes.

4.3.2.5 Pruning approach

Once the MO-GP methodology has finished and produced pareto optimal classification rules, a pruning approach is applied to each optimal individual (rule) to remove useless evidences.

The pruning procedure works at follows:

1. For each AND node of level 2, check the evidences (offspring nodes of level 4) related to that node according to the following possibilities:
    - If one (or both) of the evidences have a strict threshold value in their conditions, the subtree rooted at the AND node is removed.

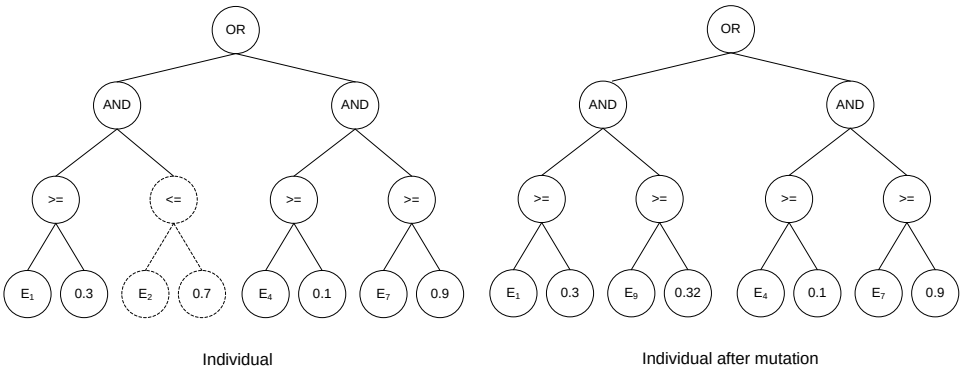


FIGURE 4.4: Example of mutation. 1) Level 3 (internal nodes that contain a relational operators) is randomly chosen. 2) The second node of Level 3 is randomly selected and the subtree rooted at this node is randomly generated: evidence 9 and a threshold of 0.32 are chosen. Assuming that evidence 9 is based on an ascendant assumption of functional linkage, the relational node is changed to " >= "

- If both evidences are in conditions with permissive thresholds values, the whole tree is reduced to the single rule : *IF any evidence has any value THEN functional relationship*
- If both evidences are equal, two cases are possible:
  - If the evidences are of type *ascendant*, remove the evidence whose related threshold is lower. From the evidence to be removed, its parent (relational node) and its related threshold are also removed.
  - If the evidences are of type *descendant*, remove the evidence whose related threshold is higher. From the evidence to be removed, its parent (relational node) and its related threshold are also removed.

  The subtree rooted at the ancestor of the other evidence (a relational node) is now a direct offspring of the root node OR.

2. Get pairs of evidences of level 4 that are repeated under more than one AND node (level 2) along the individual. Three cases are possible:

   - Each pair of evidences is of type *ascendant*. If the lowest threshold values are given for a pair of evidences under the same AND node, the remaining subtrees in the individual rooted at the AND nodes with the same pair of evidences as offspring are removed.

- One of the evidences is of type *ascendant* and the other is of type *descendant* in all pairs. If the lowest threshold value and the highest threshold value are given, respectively, for an evidence of type *ascendant* and an evidence of type *descendant* under the same AND node, the remaining subtrees in the individual rooted at the AND nodes with the same pair of evidences as offspring are removed.
- Each pair of evidences is of type *descendant*. If the highest threshold values are given for a pair of evidences under the same AND node, the remaining subtrees in the individual rooted at the AND nodes with the same pair of evidences as offspring are removed.

3. For each remaining AND node of level 2, check again the pair of evidences related to that node (offspring of level 4). If one of the evidences has a permissive threshold, remove that evidence and its related relational node (parent) and threshold. The subtree rooted at the ancestor of the other evidence (a relational node) is now a direct offspring of the root node OR.

4. Check all evidence nodes whose ancestor (a relational node) is a direct offspring of the root node OR. If there are repeated evidences and:

   - they are of type *ascendant*, keep the one with the lowest threshold and remove the other repeated evidences and their related relational nodes and thresholds.
   - they are of type *descendant*, keep the one with the highest threshold and remove the other repeated evidences and their related relational nodes and thresholds.

   If the evidence kept has a *permissive* threshold, then the whole tree is reduced to the single rule :*IF any evidence has any value THEN functional relationship*

In Fig. 4.5, an example of the pruning approach is shown. Note that the individual in this example has three compound conditions (three AND nodes) and two conditions for each compound condition.

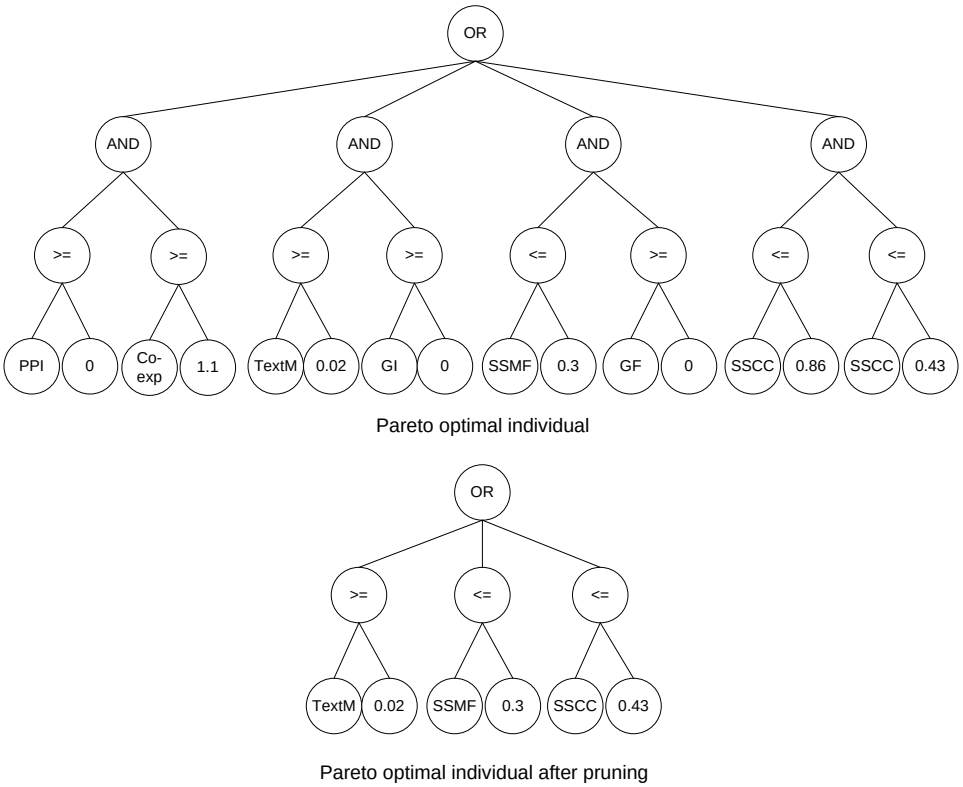


FIGURE 4.5: Example of the pruning approach once the MO-GP is finished. The methodology is applied to each individual belonging to the set of pareto optimal classification rules (individuals)

4.4 Experiments and results

This section presents the experimental design and results of our study for the integration of evidences from heterogeneous biological data sources to predict functional associations between proteins in the *Saccharomyces Cerevisiae* (Baker's yeast) organism. Examples of discovering functional relationships between proteins using this organism are widely study in the literature [Lee et al., 2004],[Lee et al., 2007],[Lee et al., 2010], [Mostafavi and Morris, 2010], [Xiong et al., 2006], [Linghu et al., 2008].

Our goal is to predict new functional relationships between proteins (i.e. to construct the FLN) by integrating genomic and proteomic features from disparate data sources to establish functional links between yeast proteins. Since each evidence

or data source usually characterizes only one type of functional association between proteins and covers a limited set of proteins, functional relationships from several evidences need to be combined to attain maximal accuracy and coverage. In this dissertation, 10 different features or evidences are assembled and comprise experimental evidences (Microarray profiles, Protein-protein interactions and Gene interactions), Sequence-based evidences (Gene fusion, phylogenetic profiles, gene neighbor, sequence similarity) and literature-derived evidences (co-ocurrence in PubMed abstracts and protein pairs sharing same molecular function terms and same cellular components in GO). These evidences are summarized in table 4.3.

TABLE 4.3: Evidences assembled in this disseration for FLN construction (see details in section 4.4.1.2) grouped by experimental, sequence-based and literature-derived evidences. Co-exp, correlated gene expression; GF, gene fusion; PP, phylegenetic profile; GN, gene neighbor; SS, sequence similarity; TextM, text mining; MF, molecular function; CC, cellular component

Group	**Evidences**	**Description**	**Downloaded/ extracted from**
Experim.	Co-exp	Expression correlation from multiple large-scale expression datasets	STRING[1]
	PPI	Protein-protein interactions	SGD[2]
	GI	Genetic interactions	SGD[2]
Sequence-based	GF	Protein pairs fused into one single protein in other species	STRING[1]
	PP	Protein pairs having correlated phyloegenetic profiles	STRING[1]
	GN	Gene pairs located close to each other along the chromosome	STRING[1]
	SS	Protein pairs having similar aminoacid sequences	*blastp*[3]
Literature-derived	TextM	Co-ocurrence in PubMed abstracts	STRING[1]
	SSMF	Protein pairs sharing same molecular function terms in GO	SGD[2]
	SSCC	Protein pairs sharing same cellular component terms in GO	SGD[2]

[1]STRING database [Szklarczyk et al., 2011]
[2]SGD database [Cherry et al., 1997]
[3]*blastp* [Altschul et al., 1990]

The data integration approach proposed in this dissertation, Pareto Optimal Classification Rules (POCR) obtained by a multi-objective Genetic Programming methodology, is applied to the integration of the evidences mentioned for predicting functional association between proteins. For comparison purposes, this methodology will be compared to other approaches:

1. Naïve Bayes, which is the most widely used strategy for the integration of evidences to predict functional associations between proteins [Wang et al., 2009b],[Bradford et al., 2010][Linghu et al., 2008][Linghu et al., 2009] [Lee et al., 2004][Lee et al., 2007] [Lee et al., 2010] (section 3.3.2.1).

2. Multilayer Perceptrons using two different approaches to train neural networks with different misclassification costs: *Threshold-Moving*, TM and *Minimization of the Misclassification Costs*, MMC (see section 3.3.2.2) which are named MLP-TM and MLP-MMC respectively from now on. In this dissertation, these cost-sensitive MLP methodologies are applied to the data integration problem for comparisons purposes.

First, the gold standard, input set and evidences and their scoring used in the experiments will be explained in section 4.4.1. Then, the methodology used to evaluate a data integration approach is briefly described (section 4.4.2). The results from Naïve Bayes, Multilayer Perceptrons (MLP-TM and MLP-MMC), and the proposed data integration methodology are reported, together with a statistical comparison among them in section 4.4.3. The suitability of the designed methodology for being executed in parallel architectures are explained in detail in section 4.4.4.

4.4.1 Gold Standard, input set and evidences and their scoring

4.4.1.1 Gold Standard

KEGG [Ogata et al., 1999] is chosen as our functional ontology or Gold Standard Set because its endpoint, pathway presence, is relatively well defined. KEGG has also been used in other works as gold standard [Linghu et al., 2008],[Lee et al., 2004].

Data downloaded from KEGG [KEGG, 2010] contains 1543 unique yeast proteins annotated to 93 different pathways. Of this set of proteins, 237,991 pairs co-occur in at least one KEGG pathway, and similar to [Linghu et al., 2008], this set can be the Gold Standard for True Positives (GSP) . Since there is no standard guideline to define the True Negative Gold Standard (GSN) set (see 3.3.1.1), we followed

TABLE 4.4: Fraction of total KEGG proteins (first column) or protein pairs (second column) by each KEGG pathway. The first ten more annotated pathways are shown

KEGG proteins fraction (%)	**KEGG protein pairs fraction** (%)	**Pathway**
40.83	83.25	Metabolic pathways
14.78	10.87	Biosynthesis of secondary metabolites
9.33	4.33	Ribosome
8.23	3.36	Meiosis - yeast
8.10	3.25	Cell cycle - yeast
5.96	1.76	Purine metabolism
4.93	1.20	Oxidative phosphorylation
4.47	0.99	Pyrimidine metabolism
4.34	0.93	Spliceosome
3.56	0.62	RNA degradation

the approach described in [Lee et al., 2004] [Lee et al., 2007] and [Linghu et al., 2009] in which the GSN is composed of the collection of protein pairs that: (1) are annotated in KEGG and (2) never occur in the same KEGG pathway based on current knowledge. There are 951,662 such pairs.

It is interesting to examine the distribution of pathway terms to check whether GSP pairs are skewed toward particular pathways. It can be observed from table 4.4 that the KEGG pathways "Metabolic pathways" and "Biosynthesis of secondary metabolites" are the two most populous pathways with 83.25% and 10.87% of the GSP respectively. So, the frequency distribution of annotated proteins are heavily biased toward these two pathways. To avoid that such bias affects the data integration methodologies and similarly to other works [Lee et al., 2007],[Linghu et al., 2009], these two pathways are excluded from the KEGG annotation data.

After removing these biased KEGG terms and since there are some proteins that are only annotated to one of these pathways, the number of proteins annotated in KEGG drops from 1543 to 1536. Thus, the Gold standard sets must be built again, yielding to 52,335 GSP pairs and 1,126,545 GSN pairs.

4.4.1.2 Input set, evidences and their scoring

Since only a subset of the *Saccharomyces Cerevisiae* genome is annotated in KEGG, we need to define the whole input set (annotated and unannotated proteins). This input set includes only yeast proteins which have sequences in the Reference Sequence (RefSeq) [Pruitt et al., 2009],[RefSeq, 2010] and measurements from both sequence-based and experimental evidences (see section 3.2) since the sequence data and experimental data have the largest proteome coverage. The same procedure has been followed in [Linghu et al., 2008] and ensures the integration by providing at least eight input features or evidences (see table 4.3).

Table 4.5 shows the number of unique yeast proteins in *RefSeq* and in the evidences downloaded from different databases in this dissertation (see table 4.1).

TABLE 4.5: Evidences (databases) and number of unique yeast proteins contained in each evidence. Datasets were downloaded in August, 2010

Evidences (database)	**Number of unique proteins**
RefSeq	5880
Co-exp,GF,PP,GN,TextM (STRING)	6142
PPI (SGD)	5569
GI (SGD)	5311
SSMF (SGD)	3080
SSCC (SGD)	4663

As previously said, the input set only includes yeast proteins that have sequences in the Reference Sequence (RefSeq) and measurements from both sequence-based evidences (Gene fusion, Phylogentic Profiles, Gene Neighbor, Sequence Similarity) and experimental evidences (co-expression from microarray data, PPI and GI). This input set will be then the intersection of the yeast proteins contained in RefSeq (5880), STRING database (6142), proteins contained in PPI predictions from the SGD database (5569) and proteins contained in GI predictions from the SGD database (5311) (see table 4.5). The intersection set, or final input set, contains 5079 proteins or nearly 13 million protein pairs.

Of these 5079 proteins, 1443 are annotated in KEGG (28.41% of the input set) and after removing both the KEGG proteins not contained in the input set and the two most biased pathways, the GSP and GSN sets contain 46,636 and 993,767 protein pairs respectively, and these would be our final gold standard sets. Notice that, for our gold standard sets, it is assumed that the ratio of functionally related protein pairs to non-functionally related protein pairs is approximately 1:22 (from 22 protein pairs in our gold standard set, one pair is expected to be functionally related and 21 pairs are not, i.e. they are unbalanced sets). In [Myers et al., 2006], they assume that of the 18 million possible pairs in yeast, it is expected than, approximately, 900,000 are functionally related. This corresponds to a ratio of functionally related proteins to non-functionally related of 1:20. As stated in the same work and in section 4.1, it is desirable that the ratio of positive to negative examples in the gold standard matches that in the application domain as closely as possible so that an evaluation of a data integration methodology on gold standards will be a representative measure of how well one could expect such methodology to perform on whole-genome data. Although our GSP and GSN sets are not perfect, the ratio of positive to negative associations is quite close to the expected ratio of positive to negative relationships at whole-genome level.

As previously described in table 4.3, we are going to use ten different data sources from which functional relationship evidences between proteins can be predicted. Each of the ten evidences contributes one component to the feature vector characterizing a protein pair:

1. **Microarray profiles, Co-exp**. We downloaded protein pairs with correlated gene expression and their associated correlation scores from STRING database [Szklarczyk et al., 2011] [STRING, 2010]. A total of 58,444 pairs of proteins have an score greater than zero in 2066 unique proteins. Pairs not having correlation score take the default value of 0.

2. **Protein-protein interactions, PPI**. Protein-protein interactions are downloaded from the *Sachharomyces Genome Database* [SGD, 2010], [Cherry et al., 1997]. Similar to [Linghu et al., 2008], the following PPI subtypes are included: Two-hybrid, Affinity Capture-MS, Affinity Capture-Western, Co-purification, Co-localization, Affynity Capture-Luminescence, Affynity

Capture-RNA, Biochemical Activity, Co-crystal Structure, Co-fractionation, FRET, Far Westerns, PCA and Reconstituted Complex. A binary value serves as the input feature denoting existence (1) or absence (0) of an interaction for a pair of proteins. In total, 49,536 pairs among 5073 proteins are known to interact, with self interaction and redundant interactions removed.

3. **Genetic interactions, GI**. Genetic interactions are also downloaded from the *Sachharomyces Genome Database* [SGD, 2010], [Cherry et al., 1997]. Similar to [Linghu et al., 2008], the following GI subtypes are included: Synthetic Lethality, Synthetic Growth Defect, Dosage Lethality, Dosage Growth Defect, Dosage Rescue, Negative Genetic, Phenotypic Enhancement, Phenotypic Supression, Positive Genetic, Synthetic Haploinsufficiency and Synthetic Rescue. We also used a binary value serving as the input feature denoting existence (1) or absence (0) of a genetic interaction for a given pair of proteins. In total, 104,347 pairs among 5076 proteins are known to interact genetically, with self interaction and redundant interactions removed.

4. **Gene fusion, GF**. Protein pairs with domain fusions and their scores are downloaded from STRING database [Szklarczyk et al., 2011],[STRING, 2010]. A total of 986 pairs have a score greater than zero in 847 proteins. Pairs not having fusion events, take the default value of 0.

5. **Phylogenetic profiles, PP**. Protein pairs with correlated phylogenetic profiles and their associated correlation scores are downloaded from STRING database [Szklarczyk et al., 2011],[STRING, 2010]. In total, 2353 pairs have non zero phylogenetic profile score among 1082 unique proteins. Pairs not having phylogenetic profile score take the default value of 0.

6. **Genomic neighborhood, GN**. Protein pairs identified by gene neighbor approach and their associated interaction scores are downloaded from STRING database [Szklarczyk et al., 2011],[STRING, 2010]. A total of 13,623 pairs have a score greater than zero in 1033 proteins. Protein pairs not having genomic neighborhood events, take the default value of 0.

7. **Sequence similarity, SS**. Protein sequences from RefSeq [Pruitt et al., 2009],[RefSeq, 2010] are downloaded and *blastp* in blast 2.2.23 is used to perform an

all against all blast within the proteome. As in [Linghu et al., 2008], pairs are filtered by requiring that their best alignment has an *E-value* lower than 0.1 and the smaller protein aligns to the larger in at least 50% of its length. The *E-value* serves as the input feature and pairs not passing this filter take the default *E-value* of 1.0. A total of 3384 protein pairs in 1786 proteins passed the filter.

8. **Text-mining, TextM**. We downloaded yeast text mining data from STRING database [Szklarczyk et al., 2011],[STRING, 2010] with each protein pair associated with a corresponding text mining score. A total of 64,761 protein pairs in 4217 proteins have a text-mining score greater than 0. Pairs not having a text-mining score take the default value of 0.

9. **Sharing molecular-function terms in Gene Ontology**. This approch makes use of the Gene Ontology database [Ashburner et al., 2000] to extract functional associations between proteins. We downloaded all GO molecular function annotations for yeast from the *Sachharomyces Genome Database*, SGD [SGD, 2010], [Cherry et al., 1997]. To avoid potential circularity with the rest of data sources to be integrated and, thus, achieve the strongest evidence, only the most reliable GO annotations, namely, IDA (Inferred from Direct Assays), IMP (Inferred from Mutant Phenotype) and TAS (Traceable Author Statement) are taken into account. Also, all GO terms labelled as "not" associated with the respective gene product were ignored, similar to other studies [Bradford et al., 2010]. As a measure of functional association for two proteins with one or more shared GO Molecular Function (GO-MF) terms, the Smallest Shared Molecular Function (SSMF) count is obtained, similar to the SSBP count described in section 3.2.3.2. A lower SSMF score corresponds to higher degree of functional association between a pair of proteins. A total of 1,643,935 protein pairs corresponding to 2893 proteins have a score less than 5079, which is the number of proteins that are present in the root GO term. Pairs of proteins in the input set that do not share any GO term take the default SSMF score of 5079.

10. **Sharing cellular-component terms in Gene Ontology**. The same procedure described for GO-MF is applied to GO Cellular Component (GO-CC)

ontology, in which the Smallest Shared Cellular Component (SSCC) count is obtained as a measure of functional association for two proteins. In total, 9,324,840 protein pairs corresponding to 4326 proteins have a score less than 5079, which is the number of proteins that are present in the root GO term. Pairs of proteins in the input set that do not share any GO term take the default SSCC score of 5079.

4.4.2 Evaluation methodology

In this section, the methodology used to evaluate the different data integration methodologies used is described.

Regardless of the data integration methodology used (Naïve Bayes, MLP-MMC, MLP-TM or POCR), to evaluate the overall performance of the prediction of functional relationship (pathway sharing) for protein pairs, we did a stratified ten-fold cross-validation (10-CV). First, both GSP and GSN sets are randomly divided into ten disjoint subsets of approximately equal size, in which the ratio of positive to negative examples is kept in all subsets. Then, nine of the ten subsets are used as the learning set to train the data integration approach and the remaining subset is used as the validation set to identify the positive (pathway sharing) and negatives (non-pathway sharing). We ran this process ten times so that each of the ten subsets was a validation set and the remaining nine constituted the learning set (10-CV). Depending on the data integration methodology, the overall performance of the prediction of functional relationships between proteins is obtained in different ways:

- Naïve Bayes. At the end of the 10-CV process, the number of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN) are averaged to get the precision (TP/(TP+FP)) and recall (TP/(TP+FN)) values (see section 3.3.1.2) under different log likelihood ratio cutoffs (LLR_{cutoff}). In this context, precision is defined as the fraction of the GS protein pairs that belong to the GSP set, and recall or sensitivity is defined as the fraction of the GSP pairs that are functionally related. By varying the LLR cutoff,

different tradeoffs between precision and recall values are obtained and plotted in the linkage precision versus linkage recall plot. Each point in the curve represents an FLN with a given size and quality. The area under the precision-recall curve (AUPRC) is used to summarize the precision-recall curve with a single statistic and, thus, to get a single quality measure of the data integration approach. This statistic is calculated as in [Davis and Goadrich, 2006].

- MLPs. Two different approaches are followed depending on how the cost-sensitivity is introduced in MLPs (MLP-TM or MLP-MMC):

  - MLP-TM. At the end of the 10-CV process, the number of TP, FP, TN and FN are averaged to get the precision and recall values under different FP/FN cost ratios. In MLP-TM, the FP/FN cost ratio is introduced in the validation phase (see Eq.3.10) of each trained network in the 10-CV procedure and varying the FP/FN cost ratio from very large to very small, different tradeoffs between precision and recall values are obtained. Again, each point in the precision-recall curve represents an FLN with a given size and quality.
  - MLP-MMC. In this case, for each FP/FN cost ratio, the 10-CV procedure is run, since the cost-sensitivity is introduced in the training phase of an MLP (see Eq.3.11). Varying the FP/FN cost ratio from very large to very small, different tradeoffs between precision and recall values are obtained, being each point in the precision-recall curve an FLN.

  In both cases, the AUPRC statistic is used to get a single quality measure of the data integration approach.

- POCR. For each learning set in the 10-CV procedure, the multi-objective genetic programming (MO-GP) approach is run, providing pareto optimal classification IF-THEN rules (POCR) which are evaluated through the validation data to obtain a set of precision-recall values. Again, each point in the curve corresponds to an FLN with a given size and quality. Notice that each IF-THEN rule in the pareto represents an entire solution for the prediction of functional associations between proteins. Since different FP/FN

ratios are considered in the MO-GP procedure, to get the AUPRC statistic, an AUPRC value is calculated for each learning/validation run of the 10-CV procedure and a final AUPRC value is obtained by averaging the different AUPRC values obtained in the 10-CV process.

Keeping the initial partition of the GSP and GSN sets, the 10-CV procedure described above is repeated 10 times for MLP-TM, MLP-MMC and POCR approaches to get average results, due to both the pseudo-random nature of MLPs when they are initialized and the random nature of the MO-GP approach to obtain POCR. We call this process as 10-times 10-CV.

It must also be noted that the initial division/partition of GSP and GSN into ten subsets to be used in the 10-times 10-CV procedure is the same for all the data integration methodologies used in the experiments section. This way, it is ensured that a data integration methodology is evaluated under the same conditions. Moreover, to ensure that such initial random partition does not affect the accuracy of a data integration method, the 10-times 10-CV procedure described above (or the single 10-CV procedure in the case of Naïve Bayes) will be run another 10 times, each one with a different partition of the GSP and GSN sets into ten subsets. This way, the data integration methodologies are studied under different random partitions of GSP and GSN to get an average overview of their accuracy.

In order to demonstrate the usefulness of integration of several evidences for predicting functional relationships, the relations extracted by the different data integration methods are compared with those supported by each evidence on isolation. In this case, protein pairs are ranked by the score of the evidence (for example, correlated gene expression score) and the ranking is then thresholded with several cutoffss to form predictions. Discrete evidences, such as PPI and GI, represent a binary relation, so, a single point in the precision-recall curve is shown.

All the evidences used in this dissertation are normalized so that they lie in the interval $[0, 1]$. Evidences extracted from STRING database (Co-exp, GF, PP, GN, TextM) and others such as PPI, SGD and SS are already normalized to $[0, 1]$. However, SS evidence has pair of proteins with very low values (1e-250,1e-200,1e-150...), so, scores from SS have been re-scaled to manageable values to be used

by the data integration methods in the interval [0, 1]. SSMF and SSCC have been normalized to [0, 1] since their original score values are in the interval [2, 5079].

4.4.3 Results

The results given by the different data integration methodologies are reported in this section. First, the results of the approaches commonly used in the literature are given, followed by the results of the proposed approach and a global comparison among all of methodologies.

4.4.3.1 Results from Naïve Bayes

In the Naïve Bayesian model (section 3.3.2.1), the likelihood ratio can be calculated as the product of individual likelihood ratios from the respective evidence types:

$$LR_{(E_1,\ldots,E_n)} = \prod_{i=1}^{n} LR(E_i), \tag{4.2}$$

where Eq. 4.2 is equivalent to the following where LLR represent log likelihood ratios:

$$LLR_{(E_1,\ldots,E_n)} = LLR(E_1) + \ldots + LLR(E_n), \tag{4.3}$$

This composite LLR corresponding to a specific biological evidence can be used to measure the predictive power or confidence degree for predicting functional relationships [Linghu et al., 2009].

First of all, the predictive power of each of the individual evidences in predicting functional associations between proteins is examined. For this purpose, it is enough to calculate $LLR(E_i)$ for each of the ten evidences. We have to distinguish between evidences with binary scores (PPI and GI) and evidences with continuous scores (the remaining evidences). Therefore, the calculation of log likelihood ratios (LLR) for evidences E_i with binary scores is:

$$LLR(E_i = 1) = log\frac{P(E_i = 1|GSP)}{P(E_i = 1|GSN)}$$
$$LLR(E_i = 0) = log\frac{P(E_i = 0|GSP)}{P(E_i = 0|GSN)} \tag{4.4}$$

where GSP and GSN denote belonging to Gold Standard Positive and Gold Santandard Negative Set, respectively.

For evidences E_i with continuous values, the scores must be binned into consecutive intervals and then calculate a LLR for each individual bin. We define $lbound_{ij}$ and $ubound_{ij}$ as the lower bound and upper bound for interval j of evidence i ($lbound_{ij} < ubound_{ij}$) respectively. Thus

$$LLR(E_i\_bin_j) = log\frac{P(lbound_{ij} \leq E_i < ubound_{ij}|GSP)}{P(lbound_{ij} \leq E_i < ubound_{ij}|GSN)} \tag{4.5}$$

where GSP and GSN denote belonging to Gold Standard Positive and Gold Santandard Negative Set, respectively and $E_i\_bin_j$ denotes the $j-th$ bin for E_i.

The LLR value for each bin was calculated over all evidences according to the GSP and GSN sets. Contingency tables in appendix A, illustrate the correlations between the evidences scores and the corresponding LLRs.

Clear correlations can be observerd in all the evidences used in this dissertation, which indicates that the LLRs can be taken as a relative measure for predicting functional linkages (pathway sharing) between proteins in Naïve Bayes. Co-expression score from microarray data (Co-exp, tableA.1), correlated phylogenetic profile (PP, tableA.5), Gene neighbor (GN, tableA.6) and Sequence similarity correlation (SS, tableA.7) have the most predictive power, whereas PPI and GI are the less predictive evidences. For example, for microarray data a significant correlation between the co-expression score and the LLR is found when the score is above 0.8 and for sequence similarity a significant correlation between the *E-value* and the LLR is found when $E\text{-}value < 1e-50$. This way, LLR values greater than zero, indicate that the evidences tends to functionally link proteins, with higher LLR scores indicating more confidente linkages.

Therefore, all the evidences (a total of ten) are going to be integrated according to Eq.4.3. In principle, Naïve Bayes integration assumes conditional independence among the different evidences to be integrated, but, as explained in section 3.3.2.1, a Naïve Bayes classifier can still be applied even when the independence assumption is not strictly satisfied [Linghu et al., 2009].

First of all, let us compare whether the initial random partition of GSP and GSN sets into ten sets to be used by the 10-CV procedure affects the accuracy of Naïve Bayes. From table 4.6, it can be observed that the Naïve Bayes integration method performs comparably when different random partitions of GSP and GSN sets are used.

TABLE 4.6: AUPRC values of data integration by means of Naïve Bayes approach. 10 different random partitions of GSP and GSN sets into ten sets used by the 10-CV procedure are used. Each cell corresponds to the AUPRC value for a given partition

Partition	**Naïve Bayes**
1	0.40889
2	0.40885
3	0.40888
4	0.40887
5	0.40889
6	0.40892
7	0.40886
8	0.40889
9	0.40891
10	0.40888

From Fig.4.6, it can be observed that data integration by means of Naïve Bayes outperforms individual evidences in terms of quantifying functional links between yeast proteins (i.e. in terms of predicting pathway sharing). So, it has been demonstrated the importance of data integration. In this plot, a random control curve is also used. To build this curve, the class labels in the gold standard data sets are randomized and then the Naïve Bayes is run as usual. In Fig. 4.6b), it is shown how each individual evidence or feature has different quality in terms of predicting functional associations. For example, for a given recall value, 0.1, the noisiest evidence is SSMF (predicts more FPs), whereas the most confident evidence is SSCC (has better precision value). Another interesting example is sequence similarity

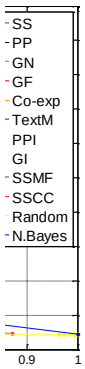
(SS) evidence: As can be observed in Fig. 4.6b), different FLNs can be obtained by varying the linkage weight cutoff and all of them are quite sparse due to the fraction of TPs present in the network (low recall value). However, most of them have a precision value greater than 0.8, which makes the FLNs very confident but sparse. So, it has been demonstrated that each data source or evidence has different quality in terms of predicting functional relationships between proteins and, probably, provides a complementary view of the yeast genome. By combining multiple evidences, in this case by means of Naïve Bayes, complete genome-wide functional linkage network and more accurate inferences of new functional relationships between proteins are provided.

4.4.3.2 Results from MLPs

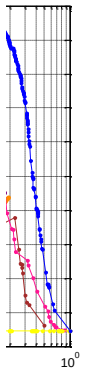
For comparison purposes, as stated at the beginning of section 4.4, a Multi-layer Perceptron (MLP) is used to recognize whether two given proteins are in the same pathway (functionally related) based on the ten evidences described in section 4.4.1.2. An MLP with one hidden layer is assumed. Each input node of the input layer is related to one evidence and the output layer has two nodes, one corresponding to functional relationship and the other related to non-functional relationship between proteins, similar to [Linghu et al., 2008]. Since the best number of neurons in the hidden layer is unknown beforehand, different number of neurons for the hidden layer is explored: 2, 5, 10, 15, 20 and 25 neurons.

As aforementioned, to obtain FLNs with different accuracy level, two cost-sensitivity approaches are used in MLPs: *minimization of the misclassification costs* [Kukar and Kononenko, 1998] (MLP-MMC) and *Threshold-moving* [Zhou and Liu, 2006] (MLP-TM). In any case, the Neural Network Toolbox [MATLAB, 2010b] from MATLAB software [MATLAB, 2010c] is used to train an MLP, using the *Levenberg Marquardt* as the backpropagation algorithm [Parker, 1987].

MLP with minimization of the misclassification costs as cost-sensitivity approach, MLP-MMC As stated in section 3.3.2.2, through this approach the error function to be minimized is the misclassification cost. For a given number of



(a)



(b)

FIGURE 4.6: Naïve Bayes Data integration vs individual evidences in terms of quantifying functional links between yeast proteins. Naïve Bayes results corresponds with one of the random partitions of GSP and GSN sets into ten sets used by the 10-CV procedure.(a) x-axis on decimal scale; (b) x-axis on log scale

neurons in the hidden layer, this approach trains a different MLP for each FP/FN cost ratio.

Since this fact makes this approach computationally expensive and to check its suitability for predicting functional linkages between proteins, only five evidences are used as inputs to the input layer: SS, PP, Co-exp, PPI and SSMF. These evidences have strong predictive power in predicting functional associations between proteins according to Tables A.1, A.2, A.5, A.7 and A.9. A range of different cost

values on either the positive or negative class are used in Eq.3.12 to get different values of precision and recall rates:

- The cost of misclassifying a negative example as a positive example, which is referred as $Cost[GSN, GSP]$, is varied in the range $[1, 30]$ while maintaining the cost of misclassifying a positive example as a negative example, $Cost[GSP, GSN] = 1$. This run is named *Cost(-)*.
- $Cost[GSP, GSN]$ is varied in the range $[1, 150]$ while maintaining the cost of misclassifying a negative example as a positive example $Cost[GSN, GSP] = 1$. This run is named *Cost(+)*.

This way, we try to cover the range from almost all predictions being true and false negatives (a sparse and relative confident FLN network) to most of the predictions being true and false positives (a dense and relative noisy FLN network). By the use of different cost values, a Precision-Recall (PRC) curve is obtained.

From Fig. 4.7, it can be observed a strange behavior when applying MLP-MMC to data integration, regardless of the number of neurons in the hidden layer. For example, with recall values greater than 0.2, the precision of the FLNs obtained is very low for any number of neurons in the hidden layer. When recall is below 0.2, FLNs with disparate accuracy (precision) are built so it seems that predictions made by the MLP-MMC approach for any number of neurons in the hidden layer are random.

To confirm the hypothesis of random predictions, instead of using Precision-recall curves to evaluate the performance of the MLP-MMC approach, ROC curves are used (see section 3.3.1.2) in which the relative trade-offs between the true positives and the false positives are highlighted. It can be observed from Fig.4.8 that, regardless of the number of neurons in the hidden layer, MLP-MMC appear very close to the diagonal, which represents a random guess prediction, in our case, random predictions of pathway sharing between pair of proteins.

So, it has been demonstrated that MLPs which are made cost-sensitive by means of the minimization of the misclassification costs (MLP-MMC), are not proper

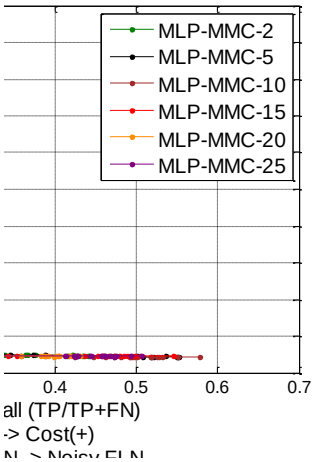


FIGURE 4.7: Data integration by means of MLP-MMC approach to predict functional links between yeast proteins. Five evidences are integrated: SS, PP, Co-exp, PPI and SSMF. Different number of neurons in the hidden layer are used. The results shown corresponds to one of the random partitions of GSP and GSN sets into ten sets used by the 10-CV procedure

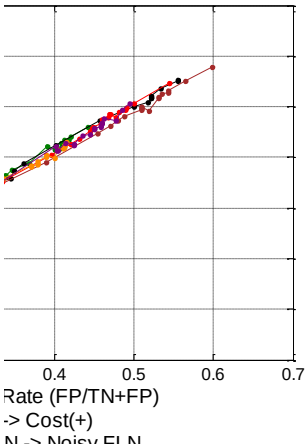


FIGURE 4.8: Data integration by means of MLP-MMC approach to predict functional links between yeast proteins. Five evidences are integrated: SS, PP, Co-exp, PPI and SSMF. Different number of neurons in the hidden layer are used. The results shown corresponds to one of the random partitions of GSP and GSN sets into ten sets used by the 10-CV procedure

approaches for integrating several evidences to predict functional associations between proteins probably for two main reasons:

- Its computational cost. For a given number of neurons in the hidden layer, a different MLP must be trained for each FP/FN cost used.
- The distribution of evidence scores in the input domain.

The last issue can probably be the main reason of poor performance for MLP-MMC when predicting functional associations between proteins. If we have a look at the contingency table detailing the distribution of SSCC scores (SSCC evidence) in the input domain (table 4.7), it can be observed that the majority of protein pairs (including GSP and GSN pairs) have an SSCC score that lie in the interval $[1000, 5079]$, which is considered, for Naïve Bayes methodology, the worst bin in terms of predicting functional associations taking into account this evidence ($LLR < 0$, see tableA.10).

TABLE 4.7: Contingency table detailing the distribution of SSCC scores (SSCC evidence) in the input domain

Bin	GSP	GSN	TOTAL
$[0, 10[$	417	18	2222
$[10, 50[$	2860	1060	24243
$[50, 100[$	3704	2949	61429
$[100, 500[$	7955	40232	508104
$[500, 1000[$	5910	100254	969801
$[1000, 5079[$	25790	849254	11329782

In the case of other evidences (see Appendix A), the same tendency is observed: for each evidence, the majority of protein pairs have a score that lies in the interval in which a functional association is less likely to exist according to the Naïve Bayes methodology ($LLR < 0$). Artificial Neural Networks whose error function is the minimization of misclassification cost are not suitable for this kind of data, since the majority of true positive cases (GSP) are in the same interval or bin where the majority of true negative cases (GSN) are, i.e. the evidence score by itself cannot be used as a predictor for functional links between proteins. This is, probably,

the main reason that this type of cost-sensitive MLP makes random predictions of functional association between proteins for several FP/FN cost ratio values.

MLP with threshold-moving as cost-sensitivity approach, MLP-TM Due to the problems described for MLPs when the cost-sensitivity is introduced in the training phase, an alternative approach is *threshold-moving*, in which the cost-sensitivity is introduced in the validation phase where the outputs of the MLP are manipulated.

For a given number of neurons in the hidden layer, this approach trains a neural network once and the cost-sensitivity is introduced in the validation phase. In this validation phase, a range of different cost values on either the positive or negative class are used to get different values of precision and recall rates (i.e. different quality for the FLNs built):

- The cost of misclassifying a negative example as a positive example, which is reffered as $Cost[GSN, GSP]$, is varied in the range $[1, 400]$ while maintaining the cost of misclassifying a positive example as a negative one, $Cost[GSP, GSN] = 1$.
- $Cost[GSP, GSN]$ is varied in the range $[1, 400]$ while maintaining the cost of misclassifying a negative example as a positive $Cost[GSN, GSP] = 1$. This run is named *Cost(+)*.

The AUPRC scores of the MLP-TM approach with different number of neurons in the hidden layer and different initial partitions of GSP and GSN sets for the 10 times 10-CV procedure are shown in table 4.8.

It will be checked whether the AUPRC values are affected by two factors: the hidden layer size (number of neurons in that layer) and the partition of the GSP and GSN sets to be used in the 10 times 10-CV procedure. For this purpose, a two-way ANalysis Of VAriance (ANOVA) analysis [Box et al., 1978] is performed, since it is a useful test when it is suspected that one or more factors affect a response. The statistical parameter considered in this test is the significant level and if it is lower than 0.05, then the corresponding levels of the factor are statistically significant

TABLE 4.8: AUPRC values of data integration by means of a cost-Sensitive MLP. Threshold-moving is used to make the MLP cost-sensitive (MLP-TM). 10 different random partitions of GSP and GSN sets into ten sets are used by the 10 times 10-CV procedure. Each cell corresponds to the average AUPRC value for a given partition and a given number of neurons in the hidden layer

Partition	**MLP-TM-2**	**MLP-TM-5**	**MLP-TM-10**
1	0.349 ± 0.014	0.400 ± 0.016	0.412 ± 0.016
2	0.354 ± 0.011	0.389 ± 0.018	0.407 ± 0.016
3	0.356 ± 0.025	0.389 ± 0.015	0.4193±0.0043
4	0.298 ± 0.045	0.402 ± 0.014	0.403 ± 0.019
5	0.350 ± 0.017	0.395 ± 0.015	0.4198 ± 0.0037
6	0.345 ± 0.022	0.4066 ± 0.0072	0.4184 ± 0.0028
7	0.359 ± 0.014	0.4065 ± 0.0074	0.4206 ± 0.0014
8	0.3636 ± 0.0042	0.399 ± 0.015	0.4186 ± 0.0068
9	0.3569 ± 0.0096	0.396 ± 0.013	0.413 ± 0.015
10	0.338 ± 0.022	0.399 ± 0.015	0.411 ± 0.017
Partition	**MLP-TM-15**	**MLP-TM-20**	**MLP-TM-25**
1	0.408 ± 0.021	0.41 ± 0.02	0.41 ± 0.02
2	0.4248 ± 0.0016	0.403 ± 0.034	0.4193 ± 0.0092
3	0.4252 ± 0.0025	0.4206 ± 0.0095	0.421 ± 0.016
4	0.419 ± 0.016	0.412 ± 0.035	0.40 ± 0.02
5	0.410 ± 0.021	0.42607 ± 0.00092	0.412 ± 0.022
6	0.416 ± 0.017	0.408 ± 0.034	0.411 ± 0.019
7	0.417 ± 0.016	0.413 ± 0.018	0.420 ± 0.017
8	0.41 ± 0.02	0.4231 ± 0.0021	0.418 ± 0.016
9	0.417 ± 0.016	0.4253 ± 0.0015	0.417 ± 0.017
10	0.407 ± 0.035	0.415 ± 0.016	0.418 ± 0.017

with a confidence level of 95%. The dependent or response variable is the AUPRC score and the factors are the hidden layer size, whose levels are 2,5,10,15,20 and 25 neurons, and the partition, whose levels are the different partitions used (a total of 10). The normality, independence of populations and homocedasticity assumptions for ANOVA test are accomplished. The results of the ANOVA test for the AUPRC response variable can be observed in table 4.9, in which the hidden layer size factor presents the greatest statistical relevance ($P < 0.05$), which means that at least one of the levels of the hidden layer size factor affects the AUPRC score or, in other words, AUPRC values statistically depend on the number of neurons in the hidden layer size. The partition factor is not statistically significant, which means that AUPRC scores do not statistically depend on the partition performed.

From Fig.4.9, it can be observed that when the MLP has 10 or more neurons in the hidden layer, it achieves statistically better results (AUPRC values) than MLPs

TABLE 4.9: ANOVA table for the analysis of the main variables (hidden layer size and partition) for the AUPRC response when threshold-moving MLP (MLP-TM) is used as a data integration approach

Main factors	Sum of squares	D.F.	Mean square	F-Ratio	Sig.level
H.Layer size	0.037	5	0.007	88.11	0
Partition	0.003	9	0	1.74	0.11

with a hidden layer size of 2 or 5 neurons. On the other hand, there are no statistical differences ($P > 0.05$) among MLPs with 10 or more neurons in the hidden layer. So, among these MLPs, it is preferable to choose the simplest one according to the parsimony principle, i.e. 10 neurons in the hidden layer.

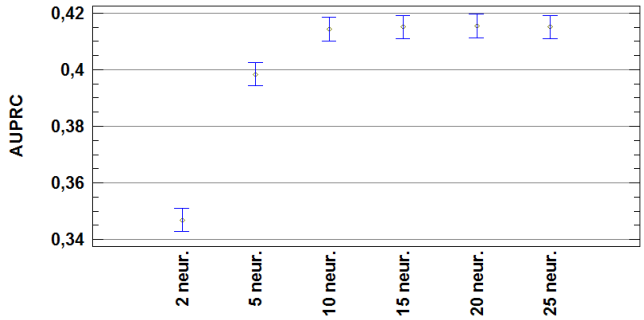


FIGURE 4.9: Means and 95% Least Significant Differences (LSD) intervals of the different sizes of hidden layer through the AUPRC values

From Fig.4.10, it can be observed that data integration by means of a cost-sensitive MLP through the *Threshold-moving* approach (MLP-TM) outperforms individual evidences in terms of quantifying functional links between yeast proteins (i.e. in terms of predicting pathway sharing). In this plot, a random control curve is also used. To build this curve, the class labels in the gold standard data sets are randomized and then the MLP-TM approach with 10 nodes in the hidden layer is run as usual.

It has been demonstrated that when the cost-sensitivity is introduced in the validation phase, that is, the outputs of the MLP are manipulated, data integration by

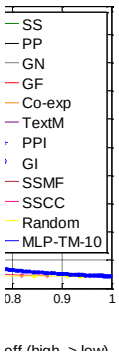


FIGURE 4.10: Data integration by means of MLP-TM with 10 neurons in the hidden layer vs individual evidences in terms of quantifying functional links between yeast proteins. Results from MLP corresponds to one of the random partitions of GSP and GSN sets used by the 10 times 10-CV procedure

means of MLPs outperforms individual evidences in terms of quantifying functional links between yeast proteins. It is easy to manipulate the outputs of MLPs and, this way, the problem of the distribution of evidence scores in the input domain can be avoided.

4.4.3.3 Results from pareto optimal classification rules (POCR) obtained by a multi-objective genetic programming approach

In this section, classification rules that use several evidences to predict functional relationships between proteins are applied. The multi-objective genetic programming (MO-GP) approach described in section 4.3.2 is run to extract such Pareto optimal classification rules (POCR).

As explained in section 4.3.2.1, the rules to be used in the MO-GP approach will be of the form:

$$IF\ (E_1 \geq \alpha_{E_1}\ AND\ E_2 \leq \alpha_{E_2}\ AND\ \ldots\ AND\ E_k \geq \alpha_{E_k}) \ldots\ OR$$
$$\ldots (E_{n-p} \leq \alpha_{E_{n-p}}\ AND\ \ldots\ AND\ E_{n-1} \geq \alpha_{E_{n-1}}\ AND\ E_n \geq \alpha_{E_n})$$
$$THEN\ functional\ relationship$$

in which the rule antecedent is made up of compound conditions linked by a logical OR operator and each compound condition is made up of single conditions connected by a logical AND operator.

In this dissertation, five compound conditions linked by a logical OR operator are used. For each compound condition, two single conditions connected by a logical AND operator are selected, that is:

$$IF\ (E_1 >= \alpha_{E_1}\ AND\ E_2 <= \alpha_{E_2})\ OR\ (E_3 >= \alpha_{E_3}\ AND\ E_4 >= \alpha_{E_4})\ OR$$
$$(E_5 >= \alpha_{E_5}\ AND\ E_6 >= \alpha_{E_6})\ OR\ (E_7 >= \alpha_{E_7}\ AND\ E_8 >= \alpha_{E_8})\ OR$$
$$(E_9 >= \alpha_{E_9}\ AND\ E_{10} >= \alpha_{E_{10}})\ THEN\ functional\ relationship$$

Using this subtype of rules, each compound condition in the antecedent of the rule restricts the prediction of functional associations to two evidences at the same time. By connecting the compound conditions by a logical OR operator, it is also given the possibility to study several subsets of two evidences to predict associations between proteins. Only two single conditions are used in each compound condition since, to keep high interpretability, a very low number of conditions are recommended when they are linked by an AND operator [Mendel, 2001]. On the other hand, interpretability is not significantly affected when several conditions are linked by OR operators [Mendel, 2001]. We still consider rules as interpretable when 5 compound conditions are linked by an OR operator. Thus, protein pairs covered by this subtype of rules, are predicted to be functionally related if this prediction is supported by, at least, two evidences, that is, a single rule represents an entire solution for the classification problem at hand. Using this structure in the rule, all evidences used in the experiments (see table 4.3) may be present in the rule at the same time.

As described in section 4.3.2.2, this rule can be represented naturally as a tree in the GP approach to codify individuals. This structure does not change during the execution of the MO-GP approach and the relational operator between an evidence and its related threshold depends on the type of evidence (see table 4.10): $\geq$ for evidences of type "ascendant" and $\leq$ for evidences of type "descendant". According to the guidelines described in section 4.3.2.1, the domain of the evidences and the threshold parameter are shown in table 4.10

TABLE 4.10: Domain of the evidences and thresholds used in the experiments for the classification rules, $\vartheta > 0$

Evidences, E_i	**Domain of** E_i	**Type of evidence**	**Domain of** α_{E_i}
Co-exp	$[0, 1]$	ascendant	$[0, 1 + \vartheta]$
PPI	$\{0, 1\}$	ascendant	$\{0, 1, 1 + \vartheta\}$
GI	$\{0, 1\}$	ascendant	$\{0, 1, 1 + \vartheta\}$
GF	$[0, 1]$	ascendant	$[0, 1 + \vartheta]$
PP	$[0, 1]$	ascendant	$[0, 1 + \vartheta]$
GN	$[0, 1]$	ascendant	$[0, 1 + \vartheta]$
SS	$[0, 1]$	descendant	$[0 - \vartheta, 1]$
TextM	$[0, 1]$	ascendant	$[0, 1 + \vartheta]$
GO-MF	$[0, 1]$	descendant	$[0 - \vartheta, 1]$
GO-CC	$[0, 1]$	descendant	$[0 - \vartheta, 1]$

In this dissertation, the multi-objective evolutionary implementation of NSGA-II in MATLAB's Global Optimization toolbox ^{TM}[MATLAB, 2010a] is used. The most relevant parameters of the MO-GP approach are shown in table 4.11. Several population sizes are explored: 150, 200, 250, 300, 350, 400, 450, 500, 550 and 600 individuals are used in the experiments. The maximum number of generations and the stall generation limit are chosen to be high enough (in fact, we checked that none of the MO-GP executions achieve the limits of these parameters). The rest of parameters has default values. It must also be emphasized, that the number of pareto optimal rules (individuals) returned by MATLAB's multi-objective evolutionary approach is equal to $populationSize * paretoFraction$.

In table 4.12, it is shown the AUPRC scores of data integration by means of the pareto optimal classification rules. Each cell corresponds to a given population size used in the MO-GP approach and a given partition of GSP and GSN sets to be used in the 10 times 10-CV procedure.

TABLE 4.11: Parameters of the MO-GP system to obtain pareto optimal classification rules for data integration

Parameter	**Description**	**Value**
Population size	The number of individuals in the population	150-600
Crossover fraction	The fraction of the population that is created by the crossover function	0.8
Mutation fraction	The fraction of the population that is created by the mutation function	0.2
Max. no. of generations	Maximum number of iterations before the algorithm halts	1000
Pareto fraction	Fraction of individuals to keep on the first pareto front while the solver selects individuals from lower fronts	0.35
StallGenLimit	The algorithm stops if there is no improvement in the objective function for StallGenLimit consecutive generations	100
Selection function	Function that selects parents of crossover and mutation children	Tournament

Again, it will be checked whether the AUPRC values are affected by two factors: the population size (number of individuals) of the MO-GP approach and the partition of the GSP and GSN sets to be used in the 10 times 10-CV procedure. For this purpose, a two-way ANOVA analysis is performed. The statistical parameter considered in this test is the significant level and if it is lower than 0.05, then the corresponding levels of the factor are statistically significant with a confidence level of 95%. The dependent or reponse variable is AUPRC score and the factors are the population size of the MO-GP, whose levels are 150,200,...,600, and the partition factor, whose levels are the different partitions used (a total of 10). The normality, independence of populations and homocedasticity assumptions for ANOVA test [Box et al., 1978] are accomplished. The results of the ANOVA test for the AUPRC response variable can be observed in table 4.13, in which the population size factor presents the greatest statistical relevance ($P < 0.05$), which means that AUPRC scores statistically depend on the number of individuals of the population in the MO-GP approach. The parittion factor is not statistically significant, which means AUPRC scores do not statistically depend on the partition performed.

TABLE 4.12: Average AUPRC values of data integration by means of pareto optimal classification rules (POCR) obtained by the MO-GP approach. 10 different random partitions of GSP and GSN sets are used in the 10 times 10-CV procedure. Each cell corresponds to the AUPRC value for a given partition and a population size of the MO-GP approach

Partition			**POCR**		
	MO-GP-150	**MO-GP-200**	**MO-GP-250**	**MO-GP-300**	**MO-GP-350**
1	0.3999 ± 0.0014	0.4057 ± 0.0013	0.4076 ± 0.0011	0.4102 ± 0.0038	0.4114 ± 0.0011
2	0.401 ± 0.001	0.4050 ± 0.0015	0.4084 ± 0.0011	0.41027 ± 0.00045	0.4118 ± 0.0011
3	0.3999 ± 0.0011	0.40479 ± 0.00093	0.4089 ± 0.0009	0.41007 ± 0.00073	0.41108 ± 0.00072
4	0.4002 ± 0.0013	0.406 ± 0.001	0.4086 ± 0.0011	0.41012 ± 0.00043	0.411 ± 0.001
5	0.4011 ± 0.0018	0.40471 ± 0.00082	0.40834 ± 0.00077	0.40969 ± 0.00078	0.41175 ± 0.00042
6	0.3997 ± 0.0015	0.405 ± 0.001	0.40825 ± 0.00079	0.41021 ± 0.00051	0.4119 ± 0.0011
7	0.4001 ± 0.0019	0.4067 ± 0.0013	0.4083 ± 0.0014	0.40980 ± 0.00053	0.41169 ± 0.00052
8	0.40056 ± 0.00062	0.4055 ± 0.0012	0.4090 ± 0.0011	0.41022 ± 0.00083	0.4112 ± 0.0011
9	0.4023 ± 0.0016	0.405 ± 0.002	0.40897 ± 0.00098	0.41041 ± 0.00081	0.4115 ± 0.0013
10	0.4019 ± 0.0014	0.4055 ± 0.0018	0.4089 ± 0.0011	0.41017 ± 0.00081	0.411 ± 0.001
Partition			**MOGP-CR**		
	MO-GP-400	**MO-GP-450**	**MO-GP-500**	**MO-GP-550**	**MO-GP-600**
1	0.41229 ± 0.00066	0.41316 ± 0.00049	0.41341 ± 0.00077	0.41380 ± 0.00045	0.41453 ± 0.00031
2	0.41212 ± 0.00044	0.41309 ± 0.00073	0.41318 ± 0.00084	0.41419 ± 0.00035	0.41464 ± 0.00061
3	0.41148 ± 0.00063	0.41262 ± 0.00087	0.41348 ± 0.00055	0.41390 ± 0.00035	0.41423 ± 0.00059
4	0.41175 ± 0.00043	0.41337 ± 0.00086	0.41356 ± 0.00032	0.4135 ± 0.0014	0.41398 ± 0.00031
5	0.41237 ± 0.00066	0.41305 ± 0.00048	0.41349 ± 0.00037	0.41393 ± 0.00041	0.41427 ± 0.00042
6	0.41234 ± 0.00068	0.41274 ± 0.00049	0.41347 ± 0.00039	0.41403 ± 0.00072	0.41401 ± 0.00054
7	0.41157 ± 0.00065	0.41335 ± 0.00054	0.41347 ± 0.00089	0.41415 ± 0.00072	0.41404 ± 0.00061
8	0.41160 ± 0.00053	0.41246 ± 0.00014	0.41329 ± 0.00017	0.41415 ± 0.00021	0.41433 ± 0.00061
9	0.41169 ± 0.00059	0.41297 ± 0.00096	0.41342 ± 0.00055	0.41364 ± 0.00057	0.41473 ± 0.00033
10	0.41089 ± 0.00047	0.41289 ± 0.00075	0.41340 ± 0.00052	0.41388 ± 0.00051	0.41413 ± 0.00046

TABLE 4.13: ANOVA table for the analysis of the main variables (hidden layer size and partition) for the AUPRC response when POCR is used is used as a data integration approach for predicting functional linkages between proteins

Main factors	**Sum of squares**	**D.F.**	**Mean**	**F-Ratio**	**Sig.level**
Population size	1.70e-3	9	1.89e-4	921.01	0
Partition	1.45e-6	9	1.61e-7	0.78	0.63

From Fig.4.11 it can be observed that when the MO-GP approach used to obtain POCR has 550 or 600 individuals in the population size, achieves statistically better results (AUPRC values) than population size with 500 individuals or less. On the other hand, there are no statistical differences ($P > 0.05$) among MO-GPs with 550 or 600 individuals. Among these, it is preferable to choose the MO-GP approach with less number of individuals (550) according to the parsimony principle.

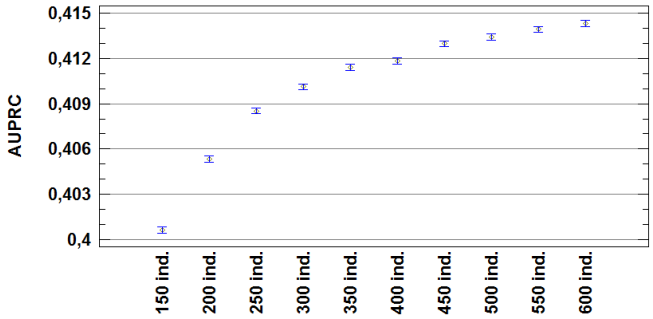


FIGURE 4.11: Means and 95% Least Significant Differences (LSD) intervals of the different sizes of population in MO-GP approach through the AUPRC values

From Fig.4.12, it can be observed that data integration by means of the POCR approach proposed in this dissertation outperforms individual evidences in terms of quantifying functional links between yeast proteins (i.e. in terms of predicting pathway sharing). Since the FP/FN ratios are considered in the MO-GP procedure, to plot a single precision-recall curve in Fig.4.12, we merged all the precision-recall points on validation data obtained in the 10 times 10-CV procedure and a single curve was approximated from this set of points by Least-Squares Support Vector Machine (LS-SVM) [Suykens et al., 2002] using the MATLAB package LS-SVMLab [Pelckmans et al., 2002]. In this plot, a random control curve is

also used. To build this curve, the class labels in the gold standard data sets are randomized and then the MO-GP approach with the same settings described in table 4.11 is run to obtain POCR. The procedure to obtain a single precision-recall curve described above is also applied.

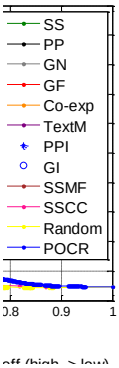


FIGURE 4.12: Data integration by means of POCR obtained by a MO-GP approach with 550 individuals vs individual evidences in terms of quantifying functional links between yeast proteins. Results from POCR corresponds with one of the random partitions of GSP and GSN sets used in the 10 times 10-CV procedure

4.4.3.4 Comparison among Naïve Bayes, MLP-TM and POCR

In this section, the POCR obtained by the MO-GP approach will be compared to the other data integration approaches used in the experiments section, namely, Naïve Bayes and MLP with *threshold-moving* to introduce cost-sensitivity, MLP-TM . The suitability of these methods for quantifying links between yeast proteins over individual evidences has been demonstrated in section 4.4.3.1 and 4.4.3.2 respectively.

Among MLP-TMs with different number of neurons in the hidden layer and according to the discussion given in section 4.4.3.2 the results provided by a MLP-TM with 10 neurons in the hidden layer, MLP-TM-10, are used in this comparison (see table 4.9). With regard to Naïve Bayes, the results obtained in section 4.4.3.1 are taken for the comparison (table 4.6). In the case of POCR and according to

the discussion given in section 4.4.3.3, the results reported in this comparison are given by the POCR when 550 individuals are used as the population size in the MO-GP approach (see table 4.12).

Thus, the three data integration approaches (Naïve Bayes, MLP-TM-10 and POCR with MO-GP-550) will be evaluated in terms of quantifying functional links between yeast proteins when several evidences are integrated. Table 4.14 shows the AUPRC values of these methodologies for several partitions of the GSP and GSN sets into subsets to be used by the 10 times 10-CV procedure.

TABLE 4.14: Average AUPRC values of different data integration approaches compared in this dissertation: Naïve Bayes, MLP-TM and POCR (MO-GP-550). 10 different random partitions of GSP and GSN sets to be used by the 10 times 10-CV procedure (single 10-CV procedure in the case of Naïve Bayes) are used. Each cell corresponds to the AUPRC value for a given partition and a given data integration approach

Partition	**N.Bayes**	**MLP-TM-10**	**POCR MO-GP-550**
1	0.40889	0.412 ± 0.016	0.41380 ± 0.00045
2	0.40885	0.407 ± 0.016	0.41419 ± 0.00035
3	0.40888	0.4193 ± 0.0043	0.41390 ± 0.00035
4	0.40887	0.403 ± 0.019	0.4135 ± 0.0014
5	0.40889	0.4198 ± 0.0037	0.41393 ± 0.00041
6	0.40892	0.4184 ± 0.0028	0.41403 ± 0.00072
7	0.40886	0.4206 ± 0.0014	0.41415 ± 0.00072
8	0.40889	0.4186 ± 0.0068	0.41415 ± 0.00021
9	0.40891	0.413 ± 0.015	0.41364 ± 0.00057
10	0.40888	0.411 ± 0.017	0.41388 ± 0.00051

It would be interesting to check whether the differences in the AUPRC values among the different data integration methodologies and the different partitions used are due to chance. For this purpose, a two-way ANOVA test could be useful as in previous sections, but this time, the homocedasticity assumption is not accomplished. Thus, the equivalent non-parametric test has to be used, in this case, the Friedman test [Friedman, 1937]. This test will check only for effects of one factor (main factor) after adjusting for possible effects of other factor, providing a ranking of the levels of the main factor and a related statistic. Since the statistic given by the Friedman test produces a conservative undersirably effect [García et al., 2009], Iman and Davenport statistic [Iman and Davenport, 1980], which is a derivation

from the Friedman's statistic, is also used to detect significant differences of the main factor under study.

Since our main interest is related to differences among the different data integration methodologies, this will be the main factor under study for Friedman test. Table 4.15 shows the average ranks obtained by each data integration algorithm in the Friedman test. At can be observed, the POCR approach achieves the best rank, followed by MLP-TM-10 and Naïve Bayes approaches.

TABLE 4.15: Average ranks obtained by each data integration method in the Friedman test for the AUPRC response variable

Data integration algorithm	**Ranking**
N.Bayes	2.8
MLP-TM-10	1.7
POCR (MO-GP-550)	1.5

Table 4.16 shows the statistic related to Friedman's test and also the statistic of applying the Iman-Davenport test. The table shows the Friedman and ImanDavenport values, χ^2 and F, respectively, and it relates them with the corresponding critical values for each distribution by using a level of significance $\alpha = 0.05$. The p-value obtained is also reported for each test. Given that the p-value of Friedman and Iman-Davenport are lower than the level of significance considered $\alpha = 0.05$, there are significant differences among the different data algorithms (Naïve Bayes, MLP-TM-10 and POCR) with a confidence level of 95%.

TABLE 4.16: Results of the Friedman and Iman-Davenport tests ($\alpha = 0.05$) for the analysis of the main factor (algorithm) for the AUCPRC response

Factor	**Friedman's statistic**	**Value in χ^2**	**p-value**
Data integration algorithm	9.8	5.99	0.0074
	Iman-Davenport's statistic	**Value in F**	**p-value**
Data integration algorithm	8.65	3.55	0.0023

Since the Friedman and Iman-Davenport test only inform us about the presence of statistical differences among all samples of result compared, a post-hoc statistical

analysis is needed to detect the existing differences among the data integration algorithms. A control algorithm, in our case the one with the best rank (POCR) is chosen so that the post-hoc procedure proceeds to compare this control algorithm with the remaining algorithms, in our case Naïve Bayes and MLP-TM-10. The post-hoc procedure chosen is Holm [Holm, 1979], which sequentially checks the hypothesis of no differences between algorithms ordered according to their significance. It is one of the most powerful post-hoc methods [García et al., 2009].

Table 4.17 shows all the adjusted p values obtained for Holm's procedure (Holm p) for each comparison which involves the control algorithm (POCR). The unadjusted and the adjusted p values are indicated in each comparison, considering a level of significance $\alpha = 0.05$. According to this table, POCR is not statistically better than MLP-TM-10 (Holm $p > 0.05$), but statistically better than Naïve Bayes (Holm $p < 0.05$) with a confidence level of 95%.

TABLE 4.17: Holm's post-hoc procedure to detect differences among the data integration algorithms using POCR as the control algorithm ($\alpha = 0.05$). z is the statistic for comparing two algorithms

i	**Data integration algorithm**	$z = (R_0 - R_i)/SE$	**Unadjusted p**	**Holm p**
2	N.Bayes	2.9069	0.0037	0.0073
1	MLP-TM-10	0.4472	0.6547	0.6547

Table 4.18 shows the average learning time for each of the data integration methodologies compared in this section. Not surprisingly, Naïve Bayes is the fastest data integration methodology. Once the different bins to be used with continuous evidences have been decided, to build contingency tables (Equations 4.4 and 4.5) simple calculations of probabilities are needed. However, MLP-TM and the MO-GP approach to obtain pareto optimal classification rules, are more computationally expensive, being the POCR obtained by the MO-GP approach the data integration methodology with the highest values.

In spite of the computational cost needed to obtain POCR, the methodology proposed in this dissertation can be considered a valid alternative not only in terms of accuracy (see table 4.17) but also in terms of flexibility and interpretability as will be examined in the following section.

TABLE 4.18: Average learning time (in sec.) of the different data integration approaches compared in this dissertation: Naïve Bayes, MLP-TM and POCR (MO-GP-550). For Naïve Bayes, each cell corresponds to the average time in the 10-CV procedure needed to learn *log likelihood ratios* $LLR(E_i), \forall i$ (i.e. time needed to build the contingency tables for all evidences, see Equations 4.4 and 4.5). For MLP-TM-10, each cell corresponds to the average time in the 10-CV procedure needed to learn the network. For POCR (MO-GP-550), each cell corresponds to the average time in the 10-CV needed by the MO-GP approach to optimize a set of classification rules (i.e. time needed to obtain pareto optimal classification rules).

Partition	**N.Bayes**	**MLP-TM-10**	**POCR MO-GP-550**
1	1.94 ± 0.08	2400 ± 350	5420 ± 100
2	1.81 ± 0.10	2040 ± 270	5382 ± 38
3	1.91 ± 0.10	2420 ± 260	5358 ± 43
4	1.97 ± 0.09	2320 ± 270	5415 ± 35
5	1.81 ± 0.10	2350 ± 130	5363 ± 22
6	1.80 ± 0.15	2080 ± 260	5400 ± 58
7	1.96 ± 0.12	2380 ± 140	5397 ± 57
8	1.86 ± 0.18	2096 ± 82	5363 ± 21
9	1.82 ± 0.15	2280 ± 450	5387 ± 47
10	1.91 ± 0.13	2080 ± 220	5383 ± 73

4.4.3.5 Flexibility and interpretability of the POCR approach

The MO-GP approach to developing POCR for data integration proposed in this dissertation fulfills the desirable requirements for a data integration methodology described in section 4.1 since:

- It can deal with large gold standard sets, so that high unbalanced GSP and GSN sets can be used.
- It provides a set of simple classification rules since the GP methodology can simultaneously evolve toward multiple alternative non-dominated solutions. Each rule in the pareto reflects the evidences integrated and is used to build an FLN (i.e. to predict functional associations between proteins) with a given level of accuracy. Covering the entire Pareto, different FLNs are obtained, each one with a different quality (trade-off between precision and recall) and, thus, different size.

- Each FLN is based on a single rule. This means, that the FLN can be described in terms of an interpretable and comprehensive rule. Moreover, the rule provides the contributions of the different types of data or evidences used in the integration process toward predicting functional associations.

To demonstrate these advantages, the MO-GP approach is run to obtain POCR using the whole GSP and GSN sets available. The number of individuals in the population for the MO-GP is 550 (the best results were achieved through this population size in section 4.4.3.3) and the parameters of the MO-GP approach are set according to table 4.11. The MO-GP approach has been run 10 times (10 different POCR sets are obtained) and the execution with the best AUPRC value is kept as a final solution.

In Fig. 4.13 the precision-recall curve for FLNs through the best POCR set is shown. Its AUPRC value is 0.4159.

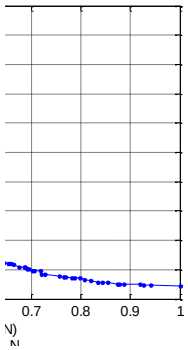


FIGURE 4.13: Data integration by means of pareto optimal classification rules obtained by a multiobjective genetic programming approach with 550 individuals. Full GSP and GSN sets are used for data integration.

One of the main advantages of the proposed approach is that a single rule, and thus an FLN built using that rule, can be obtained from a specified precision or recall value. For example, if the decision maker has a partial preference, let's say, a recall of 0.8, a single rule can be obtained through the pareto optimal classification rules obtained:

$$IF\ SS \leq 0.83\ OR\ TextM \geq 0.02\ OR\ SSMF \leq 0.3\ OR$$
$$SSCC \leq 0.24\ OR\ Co\text{-}exp \geq 0.01\ THEN\ functional\ relationship$$

this rule achieves a recall of 0.7972 and a precision of 0.0708 and five evidences are used. In this case, all the conditions in the rule form a disjunction by means of the OR operator. Obviously, the pruning procedure described in section 4.3.2.5 has been applied to obtain this rule. This way useless evidences are removed and the rules are more interpretable.

Another example is given by the following rule:

$$IF\ Co\text{-}exp \geq 0.01\ OR\ SSMF \leq 0.013\ OR\ TextM \geq 0.02$$
$$OR\ SSCC \leq 0.17\ THEN\ functional\ relationship$$

in which four evidences are used to predict functional association and this rule achieves a recall of 0.5309 and a precision of 0.2103.

If the decision maker wants the rule with the highest precision value, it is given by:

$$IF\ (Co\text{-}exp \geq 0.5\ AND\ SSMF \leq 0.0003)\ OR\ (Co\text{-}exp \geq 0.77\ AND\ SSCC \leq 0.2)$$
$$OR\ (TextM \geq 0.99\ AND\ SSCC \leq 0.12)\ OR\ (Co\text{-}exp \geq 0.47\ AND\ SSCC \leq 0.009)$$
$$OR\ (PP \geq 0.32\ AND\ TextM \geq 0.62)\ THEN\ functional\ relationship$$

which provides a precision of 1 and a recall of 0.0338.

On the other hand, it is also useful to examine different issues regarding the pareto optimal classification rules obtained:

- The number of compound conditions or terms linked by the OR operator, which will provide information about the complexity of the rule. A compound term is made up of two conditions linked by the AND logical operator, so, the number of compound terms linked by the OR operator will

provide the number of AND operators present in the rule. It is possible that there are less or no AND operators in the rule, in which case it is interesting to measure the number of single condition terms linked by the OR operator. For example, the following rule:

$$IF\ TextM \geq 0.28\ OR\ SSCC \leq 0.013\ OR\ (Co\text{-}exp \geq 0.01\ AND\ PPI \geq 1)\ OR$$
$$(SSMF \leq 0.03\ AND\ SSCC \leq 0.17)\ OR\ (Co\text{-}exp \geq 0.13\ AND\ SSCC \leq 0.21)$$
$$THEN\ functional\ relationship$$

has three compound terms linked by the OR operator and two single condition terms linked by the OR operator.

Fig. 4.14 shows the distribution of compound and single condition terms linked by the OR operator along different values of recall. It can be observed that, for very low recall values (high precision scores), the number of compound terms is very high. This is obvious because high precision with low recall values means the prediction of a small number of positive functional associations, but most of them being true (a compound term restricts the prediction to two evidences). To this end, the rules are more complex since it is made up of a set of compound terms and the threshold values are more strict. On the other hand, for very high recall values (low precision), there are no compound terms in the rules obtained and those are made of single condition terms linked by the OR operator. In this case, rules are easier and the threshold values are more permissive: most of the predictions are positives (true and false) and the rule does not have to restrict the results by means of a set of compound terms with high threshold values. As can be observed, there is, in general, a correlation between the recall values and the average number of compound and single condition terms. As the recall value increases, the average number of compound terms decreases as well, whereas the average number of single condition terms increases. In other words, as the recall value increases, the rule complexity decreases.

- Average number of evidences per rule. This issue is related to the number of the compound terms linked by the OR operator previously described. The

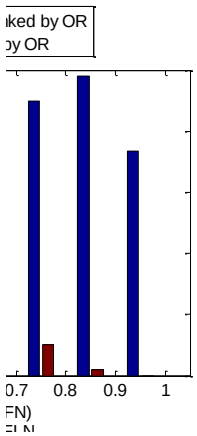


FIGURE 4.14: Distribution of compound and single condition terms linked by the OR operator along different values of recall

more number of compound terms in a rule, the more number of evidences since the rule is more complex. Fig. 4.15 shows the distribution of the average number of evidences per rule along different recall intervals. For each interval under study, how often a single evidence appears in the classification rules of that interval is also provided. It can be observed that, in general, less number of evidences are used as the recall increases, which means that the rules are more complex for low recall values (high precision) and simpler for high recall values (low precision). These results are consistent with the ones given from Fig.4.14.

Moreover, Fig. 4.15 highlights the importance of Co-exp, TextM, SSMF and SSCC evidences in the POCR set. This means that these evidences have a strong predictive power in terms of quantifying functional associations for any level of recall or precision. It can also be concluded that gene neighbor (GN) evidence is useful only for very low values of recall and that gene interaction (GI) evidence is suitable for rules with medium-high values of recall.

- The presence of evidences in the POCR. It is checked the proportion of classification rules of the pareto that contains a given evidence. From Fig.4.16, it can be observed that in the POCR obtained, a total of 193 rules, Co-exp,

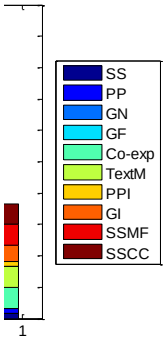


FIGURE 4.15: Distribution of the average number of evidences per rule along different recall intervals.

SSMF and SSCC evidences are present in more than 95% of the classification rules and TextM evidence is present in around 92% of the rules. On the other hand, sequence similarity (SS), phylogenetic profile (PP), gene neighbor (GN) and gene fusion (GF) are present in less than 10% of the rules. These results are consistent with the ones given in Fig.4.15.

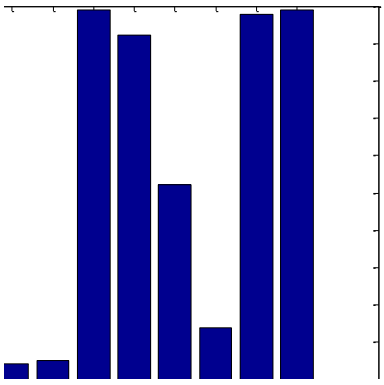


FIGURE 4.16: Presence of evidences in the pareto optimal classification rules

- Total number of times that a given evidence is referenced in the pareto optimal classification rules. In our case, in the set of 193 pareto optimal classification rules, there are 1337 referenced evidences (this includes repetitions of evidences). Around 30% of them correspond to the SSCC evidence

(see Fig.4.17). This means that such evidence appears around 407 times and since the number of classification rules in the pareto is 193, this suggests that SSCC evidence appears more than once in several classification rules. The second and third most referenced evidences are Co-exp and SSMF respectively (around 22% for Go-exp and 17% for SSMF of referenced evidences). This also means that these evidences appear more than once in some rules. On the other hand, sequence similarity (SS), phylogenetic profile (PP), gene neighbor (GN), gene fusion (GF) and genetic interactions (GI) evidences are the least referenced evidences (less than 2% each).

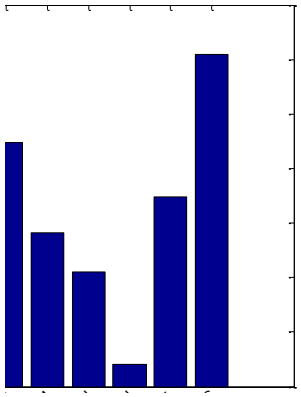


FIGURE 4.17: Number of times (in %) that a given evidence is referenced in the pareto optimal classification rules

4.4.4 High-performance computing for the MO-GP approach

It must be emphasized that the executions of the MO-GP approach to developing POCR, have been done in the context of high-performance computing through the use of parallel architectures. To be more precise, the computer cluster of our department, called BIOATC, has been used to take advantage of its several computational resources. BIOATC offers several possibilities for our MO-GP approach since, it is well known, that such type of optimization methodologies are quite demanding both in time and in space.

From Fig.4.18, the procedure to run the MO-GP approach with different population size is shown. BIOATC cluster is made up of 19 hosts, providing up to 304 CPUs and 304 GB of memory. The main node is in charge of distributing MO-GPs with different population size to several nodes, each one running the MO-GP approach with a given population size. In our case, 10 nodes were needed, since 10 different MO-GP algorithms were run (see table 4.12). For each node, the MO-GP was run through the multi-objective evolutionary algorithm provided by the MATLAB optimization toolbox. This toolbox offers the possibility of using a pool of MATLAB workers connected to a given MATLAB client providing parallel computing. This feature was used in our MO-GP for fitness evaluation of a population. It is well known that a fitness function is computationally demanding since it evaluates each individual on each generation, so, in our case, the MATLAB client of a given node, creates a pool of MATLAB workers. Each of these workers is in charge of evaluating a subpopulation so that different subpopulations of the total population are evaluated by the fitness function in parallel. Fig.4.18 shows the details for node number 3, but the same is applied to the rest of nodes. Initially, MATLAB allows as many workers as CPUs available in a node, however, the license used at BIOATC, allows up to 8 workers.

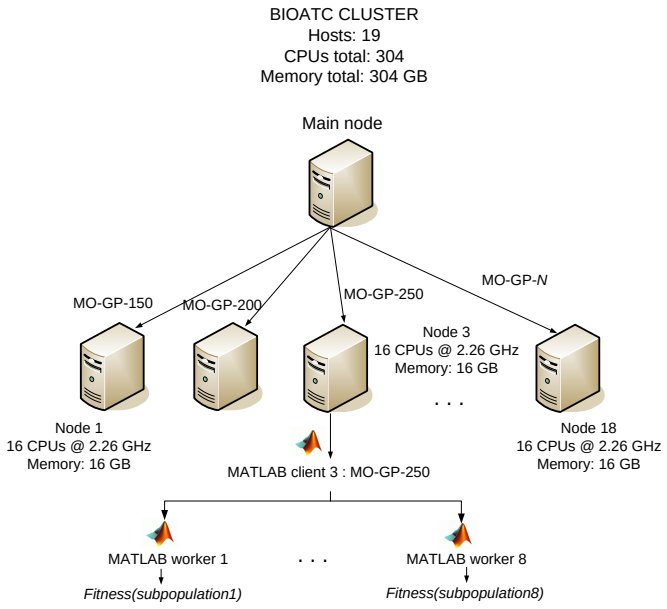


FIGURE 4.18: High-performance computing for the MO-GP approach

This procedure can be seen as a task-parallelism and data-parallelism approach in the following sense: task-parallelism due to the distribution of different MO-GP algorithms to different nodes and data-parallelism due to the distribution of subpopulations to different workers for their evaluation. Nevertheless and, strictly speaking, data-parallelism refers to the distribution of data across different parallel computing resources and, in our case, we are referring to a set of individuals as data.

For the MLP-TM data integration methodology, we also took advantage of the cluster, distributing MLP-TMs with different number of neurons to several nodes. However, MATLAB does not offer the possibility of parallelizing the training of an artificial neural network.

4.5 Conclusion and future work

In this part of the dissertation, we have demonstrated the merit of the integration of ten evidences or sources to predict functional associations between proteins in *Saccharomyces Cerevisiae* organism. Weak evidences from multiple sources can be combined to provide strong evidences for a relation. To demonstrate this fact, several data integration approaches have been applied:

- Naïve Bayesian model. It is one of the most widely used strategies for the integration of evidences to predict functional associations between proteins. It is a very fast methodology to provide FLNs with different levels of accuracy, although it does not provide interpretable rules in which the contributions of the evidences used toward the prediction task can be observed. Its accuracy has been demonstrated in the literature and it does not suffer from the problem of the distribution of the protein pairs (evidence scores) in the input domain: for each evidence, the majority of GSP protein pairs are in the same bin where the majority of GSN pairs are as well. For a given bin, Naïve Bayes measures the probability of observing the values in the evidence data set given that a pair of proteins are functionally related (GSP) divided by the probability of observing the values given that the pair is not functionally related (GSN). Thus, for this methodology, the absolute number of GSP pairs

in the worst bin is irrelevant: although the absolute number of GSP pairs in the worst bin are large (see Appendix A), the absolute number of GSN pairs in the worst bin are much larger.

- Multilayer perceptrons using two different approaches to train neural networks with different misclassification costs: *Minimization of the Misclassification Costs* and *Threshold-Moving*. It has been demonstrated that the first approach is not suitable for data integration due to the problem of the distribution of protein pairs in the input domain previously described: the evidence score by itself cannot be used as a predictor for functional links between proteins, since, as previously described, the majority of true positive cases (GSP) are in the same interval or bin where the majority of true negative cases (GSN) are. This fact makes the neural network to produce random predictions of functional relationships, regardless of the different misclassification costs introduced during learning. Moreover, even if the problem of the distribution of protein pairs in the input domain does not exist, a different network has to be learned for every FP/FN cost ratio to obtain FLNs with different levels of accuracy. MLPs with threshold-moving as cost-sensitivity approach has been proved to be effective for data integration, since by manipulating the outputs of the MLP, the problem of the distribution of protein pairs in the input domain is overcome. It is a relative fast approach in the sense that the outputs are modified to obtain several FLNs with different levels of accuracy, although the training can be quite time consuming. On the other hand, the model built is not interpretable (i.e. in the form of simple rules) for the decision maker and a reasonable range of FP/FN cost ratios must be explored explicitly.

- Pareto optimal IF-THEN classification rules obtained by a multi-objective genetic programming approach. This methodology proposed in the present dissertation has been proved to be an effective data integration methodology in terms of: (i) the handling of high unbalanced GSP and GSN sets that reflects the ratio of positive to negative examples in the application domain as closely as possible; (ii) accuracy, improving the results given by the Naïve Bayesian model, (iii) interpretability, since an FLN is constructed through simple and understandable rules in which the evidences used for predicting

functional associations are given and (iv) flexibility, since the decision maker does not have to specify partial preferences on the desired accuracy of the FLN, owing to the fact that covering the entire pareto, different FLNs are obtained, each one with a different level of accuracy. Although the computational cost of the proposed methodology is higher than other data integration approaches, such increase in the learning time is not dramatic, since the MO-GP method simultaneously evolves toward multiple pareto optimal rules or solutions (i.e. multiple FLNs).

Due to the interpretability of pareto optimal classification rules obtained, it has been demonstrated the importance of several evidences when predicting functional associations between proteins. In this sense, co-expression (Co-exp), text mining (TextM), sharing molecular function and cellular components terms in Gene Ontology (SSMF and SSCC respectively) evidences have the best predictive power toward functional associations for any level of recall or precision. Moreover, Gene Neighbor (GN) plays a key role in predicting associations at high precision. Providing the decision maker with the role of each evidence when functional associations are uncovered is a valuable information from the biological point of view.

The MO-GP approach has been demonstrated to be very useful as a search and optimization algorithm to find a set of classification rules by optimizing at the same time conflicting objectives such as precision and recall. However, more conflicting objectives can be added, such as false negative vs. false positive or sensitivity vs. specificity.

As a future work, we are planning to explore other ways to generate the GSN set (see section 3.3.1.1) and check how the predicted functional associations are affected by the generation of such set. Also, we are developing several ways of assigning a weight or degree to the functional associations predicted so that the greater the link weight, the greater the tendency of functional association. For example, for two protein pairs and a rule given, the compound terms in the rule that cover the protein pairs are taken. For each compound term, one can measure the euclidean distance between the evidence scores of the pair of proteins and the threshold values in the compound term. Then, the maximum value of the distance measures given by all compound terms (they are linked by an OR operator), can

be taken as a degree of functional association. Another possibility could be to ”fuzzify” the link weight, so that the degree of functional association could be ”low”, ”medium” or ”high”. This way, once a weight is defined for each functional association predicted, individual protein functions on the basis of linked neighbors can be inferred by means of decision rules for transferring the function of annotated proteins to unannotated proteins [Linghu et al., 2008].

Other ways to extract rules can also be explored. For example, rules can be extracted from trained neural network models, such as the *orthogonal search-based rule extraction*, OSRE [Etchells and Lisboa, 2006]. This methodology could be applied to the MLP-TM approach proposed in this dissertation to extract rules to be compared with the ones obtained by the MO-GP approach in terms of accuracy and interpretability.

The methodology proposed in this part of the thesis can easily be applied to other prediction tasks, such as the prediction of protein-protein interactions [Rhodes et al., 2005] or even to prioritize new genes that are potentially associated with a given disease or to explore the inter-relationships between diverse disease revealed by considering functional associations between genes associated with different diseases [Linghu et al., 2009]. Moreover, other types of data can also been integrated, such as clinical, environmental and demographic data for the identification of new cancer biomarkers and targets for therapy [Hackl et al., 2010].

Part II

A2TOOL: Affymetrix Microarray Analysis TOOL

Chapter 5

Introduction

A functional linkage network (FLN) constructed by integrating several heterogeneous biological data sources, can be used to predict or prioritize new (not previously recognized) genes that are potentially associated with a given disease; and to explore the inter-relationships between diverse diseases revealed by considering functional associations between genes associated with different diseases. This approach is based on the idea that genes associated with the same or related disease phenotypes tend to participate in common functional modules.

The procedure is quite straightforward [Linghu et al., 2009]: functional associations between genes are retrieved from diverse data sources and such functional associations are then integrated into one single FLN using a data integration methodology such as the multiobjective genetic programming approach described in previous sections. The nodes, in the FLN, represent individual genes and the weighted edges represent the degree of their overall functional association upon combining all contributing data sources:

- For candidate disease gene prioritization and given a particular disease, genes known to be associated with this disease are labelled as seeds and all other genes are prioritized in terms of their association with the disease based on the sum of the weights of their network links to the seed genes.

- For quantifying the disease-disease associations, genes known to be associated with different diseases are labelled and quantify the associations between any two diseases are quantified based on the degree of association between the two corresponding disease gene sets within the FLN.

Known disease genes can be obtained from the Online Mendelian Inheritance in Man Database (OMIM) database [Hamosh et al., 2005], however, one also can extract/discover genes related to a given disease from custom experiments using microarray data. For example, interesting genes associated to pancreatic adenocarcinoma can be extracted through a microarray data analysis [Caba et al., 2010] and, then, genes identified as being related to such disease, can be labelled in the FLN for either prioritizing new genes that are potentially associated with pancreatic adenocarcinoma or exploring the inter-relationships between such disease and other cancer-based diseases.

To extract genes related to a given disease from custom microarray data experiments, a proper and organized analysis of such data must be carried out. A typical microarray data analysis pipeline consists of:

1. Quality data analysis, which analyzes the quality of the microarray data set with the aim of making the best use of the information produced by the arrays [Gentleman et al., 2005].

2. Data pre-processing, which consists in removing technical variations which affects the measured gene expression levels while maintaining the effect due to the treatment under investigation [Lim et al., 2007].

3. Detection of differentially expressed genes. It allows to identify differentially expressed genes with the purpose of, for example, detecting genes associated with different disease phenotypes [Gentleman et al., 2005] or allowing researchers to elucidate related biological processes [Xu et al., 2009].

4. High-level analysis, such as cluster analysis, classification, GO-analysis or Gene Set Enrichment Analysis

In the last few years, the Bioconductor project [Gentleman et al., 2004] has become the reference tool for the analysis of microarray data. It is an open development software project, based on the statistical programming language R, for the analysis and comprehension of genomic data and, currently, hundreds of Bioconductor packages have been developed, providing comprehensive functionalities for all aspects of microarray data analysis

However, for scientists without adequate programming experience, the command line usage of R and Bioconductor is too awkward and difficult. To overcome these pitfalls, many analysis tools with graphical user interfaces and powerful computing servers have been developed, including some well-known tools such as CARMAweb [Rainer et al., 2006], GEPAS [Vaquerizas et al., 2005], MAGMA [Rehrauer et al., 2007], EzArray (currently called BxArrays) [Zhu et al., 2008], dChip [Wong, 2011], RMAexpress [Bolstad, 2011b] or mu-CS [Guzzi and Cannataro, 2010].

Quality assessment and pre-processing have an important role in the earlier stage of microarray data analysis pipeline:

- Quality assessment has the purpose of analyze, using different metrics, the quality of the microarray data set with the aim of making the best use of the information produced by the arrays [Gentleman et al., 2005] and remove those samples with low quality that may affect the rest of the analysis such as the detection of differentially expressed genes or the high level analysis. Some of the tools described above provide many metrics for quality assessment, but they leave to the researcher the decision of removing a sample of the experiment if *it seems* to be defective given the quality plots used with no clear guidelines. Thus, if the user has no experience on microarrays, this task can be cumbersome.

- Pre-processing removes technical variations present in the experiment. The tools described above provide a set (or subset) of standard pre-processing methods and some of them also provide custom pre-processing methods, but, in any case, they leave to the (unexperienced) user to decide which

pre-processing method(s) to use, which may also have an effect in posterior stages such as the detection of differentially expressed genes [Bolstad et al., 2003], [Ritchie et al., 2007], phenotype classification [Florido et al., 2010b],[Wu et al., 2005] or the construction of reverse engineering networks [Lim et al., 2007]. In the case of Affymetrix microarray technology, pre-processing consists of four stages: background correction, normalization, PM correction and summarization and the user has to decide, for custom pre-processing, which combination for the four stages to be used among tens of them.

Due to these problems, it is proposed a new tool for the first three steps of microarray data analysis pipeline: quality assessment, pre-processing and the detection of differentially expressed genes. The tool, which has been developed for the commonly used standard Affymetrix 3' expression arrays, provides the following original features:

1. Automated detection of low quality microarrays so that the decision maker is able to decide whether one or more arrays are defective or not based on a full set of quantitative and qualitative measures.
2. Automated selection of the best pre-processing methods among several ones for a given data set through objective quality measures. The aim is to free the researcher from taking a decision about the pre-processing method to be used.
3. Automated generation of confident and complete lists of differentially expressed genes.

This automation on both quality assessment and pre-processing, (i) avoids to waste time for searching the proper quality assessment and pre-processing methods and (ii) reduces the possible errors in further analysis phases due to the presence of low quality arrays and/or incorrect choice of pre-processing methods.

This part of the dissertation is structured as follows. Chapter 6 provides an overview to microarray technology (section 6.1), an explanation of the different steps involved in microarray data analysis pipeline (section 6.2) and the description of

some tools available in the literature to analyze microarray experiments (section 6.3). In Chapter 7, the microarray analysis tool proposed in this part of the thesis, A^2**TOOL**, is presented and applied to the Chronic Lymphocytic Leukemia (CLL) data set.

Chapter 6

Microarray technology: concepts and tools

6.1 Introduction to Microarray Technology

The fundamental basis of DNA microarrays is the process of *hybridization*. Two DNA strands hybridize if they are complementary to each other. Complementarity reflects the Watson-Crick rule that adenine (A) binds to thymine (T) and cytosine (C) binds to guanine (G) (Fig.6.1).

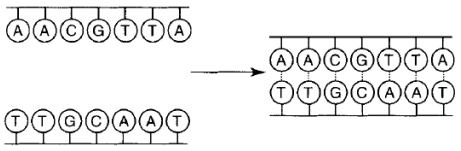


FIGURE 6.1: Hybridization of two DNA molecules

Hybridization has for decades been used in molecular biology as the basis for such techniques as Southern blotting and Northern blotting. DNA arrays are a massively parallel version of Northern and Southern blotting. Instead of distributing the oligonucleotide probes (small strings of DNA) over a gel containing samples of RNA or DNA, the oligonucleotide probes are attached to a surface. Different

probes can be attached within micrometers of each other, so it is possible to place many of them on a small surface of one square centimeter, forming a DNA array. The sample is labeled fluorescently and added to the array. After washing away excess unhybridized material, the hybridized material is excited by a laser and is detected by a light scanner that scans the surface of the chip. Because it is known the location of each oligonucleotide probe, it can be quantified the amount of sample hybridized to it from the image generated by the scan [Knudsen, 2006]. There is some contention in the literature on the use of the word ”probe” in relation to microarrays. Throughout this chapter, the word ”probe” will be used to refer to what is attached to the microarray surface and the word ”target” will be used to refer to what is hybridized to the probes.

DNA arrays are often used to study *all* known messengers of an organism. This has opened the possibility of an entirely new, systematic view of how cells react in response to certain stimuli. It is also an entirely way to study human disease by viewing how it affects the expression of all genes inside the cell.

6.1.1 The technology behind DNA microarrays

When DNA microarrays are used for measuring the concentration of messenger RNA in living cells, a *probe* of one DNA strand that matches a particular messenger RNA in cell is used. The concentration of a particular messenger is a result of *expression* of its corresponding gene, so this application is often referred to as *expression analysis*. When different probes matching all messenger RNAs in a cell are used, a snapshot of the total messenger RNA pool of a living cell or tissue can be obtained. This is often referred to as an *expression profile*, because it reflects the expression of every single measured gene at that particular moment. Expression profile is also sometimes used to describe the expression of a single gene over a number of conditions.

For expression analysis, the field has been dominated in the past by two major technologies: (1) Spotted arrays or custom-made chips, which use a robot to spot cDNA, oligonucleotides, or PCR products on a glass slide or membrane and (2) the Affymetrix GeneChip system, which uses prefabricated oligonucleotide chips and

In spotted arrays, the probes, which are cDNA, PCR product or oligonucleotides, are synthesized prior to deposition on the array surface and are then ”spotted” onto glass. A common approach utilizes an array of fine pins or needles controlled by a robotic arm that is dipped into wells containing DNA probes and then depositing each probe at designated locations on the array surface. The resulting ”grid” of probes represents the nucleic acid profiles of the prepared probes and is ready to receive complementary cDNA or cRNA ”targets” derived from experimental or clinical samples (see Fig.6.2). This technique is used by research scientists around the world to produce *in−house* printed microarrays from their own labs, which provides a relatively low-cost microarray that may be customized for each study, and avoids the costs of purchasing often more expensive commercial arrays that may represent vast numbers of genes that are not of interest to the investigator [Knudsen, 2006].

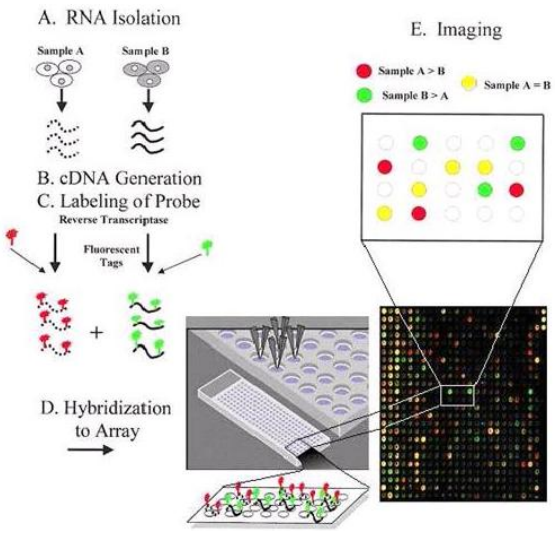


FIGURE 6.2: The spotted array technology. A robot is used to transfer probes in solution from a micro titer plate to a glass slide where they are dried. Extracted mRNA from cells is converted to cDNA and labeled fluorescently. Reference sample is labeled red and test sample is labeled green. After mixing, they are hybridized to the probes on the glass slide. After washing away unhybridized material, the chip is scanned with a confocal laser and the image analyzed by the computer [Knudsen, 2006]

Standard Affymetrix 3’ expression arrays or classical Affymetrix chips, use a set of features (often referred to as ”spots”) designed to recognize each molecule of

interest. Each feature consists of millions of identical single-stranded 25-mer nucleotide probes, each designed to hybridize to a specific transcript. On a gene-level array, such as the HGU133plus2 chip, each of these Perfect Match (PM) features is accompanied by an adjacent Mis-Match (MM) feature in which the middle residue is changed. Hybridization conditions are designed to maximize binding to the PM features while minimizing binding to the MM ones. The rationale for including MM probes in a probe set is that their intensities were thought to account for non-specific signals that affect both PM and MM probes in the same way (see Fig.6.3), although some authors claim that MM probes are not adequate controls for non-specific hybridization [Pozhitkov et al., 2007]. Multiple PM/MM pairs are used for each transcript. On most gene-level arrays, 11 PM/MM pairs are used per transcript, and the complete set of 22 features is referred to as a probeset.

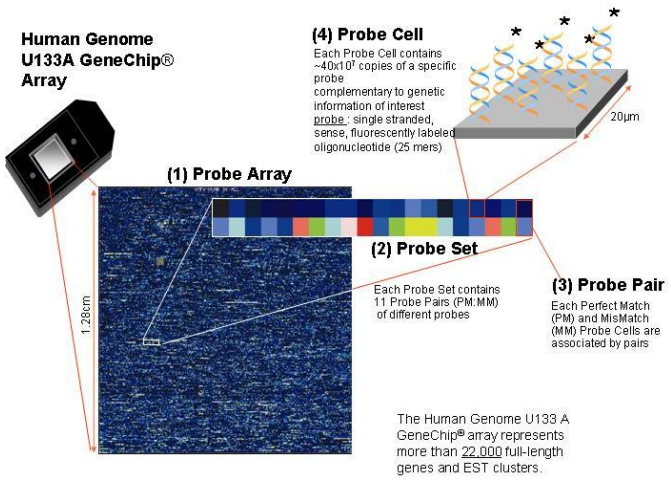


FIGURE 6.3: The Affymetrix Genechip technology (http://www.weizmann.ac.il/home/ligivol/research_interests.html)

6.1.2 Applications of microarrays

The types and numbers of applications for microarray experiments are quite variable and constantly increasing [Knudsen, 2006], for example:

- To monitor the expression level of genes between two conditions. This type of study, termed gene expression profiling, can be used to determine the

function of particular genes during a particular state, such as nutrition, temperature, or chemical environment. Such results could be observed as up- or down-regulation, or unchanged during particular conditions.

- To discern the mechanism of action of therapeutic agents and as a corollary to develop new drug targets, which is known as pharmacological studies. The guiding principle in this endeavor is that genes regulated by therapeutic agents result from the actions of the drug. Identification of the genes that are regulated by a certain drug could potentially provide insight into the mechanism of action of the drug, prediction of toxicological properties, and new drug targets.

- The diagnosis of clinically relevant diseases. The oncology field has been especially active and to an extent successful in using microarrays to differentiate between cancer cell types. The ability to identify cancer cells based on gene expression represents a novel methodology that has real benefits. In difficult cases where a morphological or an antigen marker is not available or reliable enough to distinguish cancer cell types, gene expression profiling using microarrays can be extremely valuable.

- Comparative genomic analysis. Microarrays have been used as a shortcut to both characterize the genes within an organism (structural genomics) and also to determine whether those genes are expressed in a similar way to a reference organism (functional genomics).

- Comparative genomic hybridization (CGH), in which gene copy numbers are compared between two samples and genomic DNA rather than RNA transcripts are labeled. CGH can detect gene amplification or deletion events underlying tumor genesis and it can also be used to detect genomic rearrangement or abnormality events with serious health consequences in humans such as trisomies.

6.2 Microarray data analysis pipeline

A typical microarray data analysis pipeline consists of (Fig. 6.4):

1. Quality data analysis. This step consists of analyzing the quality of the microarray data set with the aim of making the best use of the information produced by the arrays [Gentleman et al., 2005].

2. Data pre-processing. It consists of removing technical variations which affects the measured gene expression levels while maintaining the effect due to the treatment under investigation. In Affymetrix Genechips, this step consists of four stages: background correction, normalization, PM-MM correction and summarization [Lim et al., 2007].

3. Detection of differentially expressed genes. It allows to identify differentially expressed genes with the purpose of, for example, detecting genes associated with different disease phenotypes [Gentleman et al., 2005] or allowing researchers to elucidate related biological processes [Xu et al., 2009].

4. High-level analysis, such as:

    - Cluster analysis, whose main purpose is to identify and group together similarly expressed genes and then try to correlate the observations to biology. Genes and/or samples can be grouped according to their expression similarity using the hierarchical clustering algorithm (HCL), the *k*-means clustering method or self-organizing maps (SOM) [Rainer et al., 2006].
    - Classification, in which given a collection of gene expression profiles for tissue samples belonging to various disease types, the goal is to build a classifier to automatically determine the disease type of a new sample at high precision. The most well known methodologies to perform classifications are *k*-Nearest Neighbor, Support Vector Machines and backpropagation neural networks [Florido et al., 2010b].
    - GO-Analysis. The GO analysis aims to assist in the biological interpretation of the results by finding GO terms that are significantly often associated to genes in a given gene list.
    - Gene Set Enrichment Analysis, which determines whether a given gene set is significantly enriched in a list of gene markers ranked by their

correlation with a phenotype of interest. The method has been successfully used to discover metabolic pathways altered in human diabetes and reveal more consistency between independent lung cancer outcome datasets at the gene set level than at the single gene level, among many other applications [Subramanian et al., 2007].

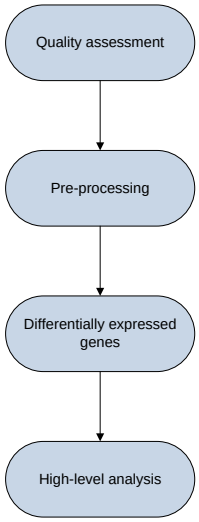


FIGURE 6.4: Microarray data analysis workflow

In the following subsections, the first three steps (quality data analysis, data preprocessing and the detection of differentially expressed genes) will be explained in detail. The fourth step (high-level analysis) will be briefly introduced in terms of Classification.

6.2.1 Quality data analysis

Obtaining gene expression measures for biological samples through the use of Affymetrix GeneChip microarrays is an elaborate process with many potential sources of variation. Therefore, it is critical to make the best use of the information produced by the arrays, and to ascertain the quality of this information. Thus, an initial examination of the data is needed to show evidence of possible quality problems and, in this case, low quality arrays should be removed from the data set. In this subsection, various graphical tools that can be used to facilitate the decision

of whether to remove an array from further analysis are presented (Fig.6.5). From the viewpoint of a user of GeneChip expression values, lower variability data, with all other things being equal, should be judged to be of higher quality [Gentleman et al., 2005].

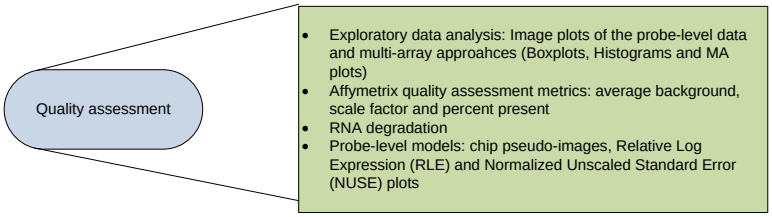


FIGURE 6.5: Microarray quality assessment tools

6.2.1.1 Exploratory data analysis

Exploratory data analysis consists of Image plots of the probe-level data and multi-array approaches.

Image plots of the (PM and MM) probe-level data A typical first step is to look at image plots of the log intensities of the raw probe-level data. In general, one looks for spatial artifacts such rings, shadows, etc (see Fig. 6.6) or other non homogeneous patterns in the image plots across all arrays (a potentially defective array may appear lighter or darker than the others) [Gentleman et al., 2005], [Alvord et al., 2007]. In Fig.6.6, it can be observed a shadow in the first array, which may indicate a defective sample.

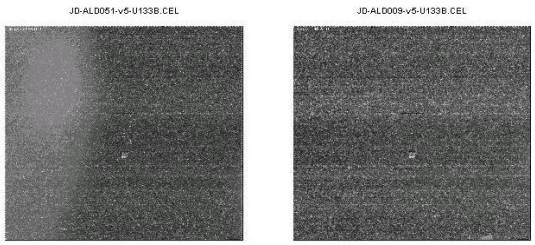


FIGURE 6.6: A subset of the MLL.B arrays from a large acute lymphoblastic leukemia (ALL) study [Ross et al., 2003]

Multi-array approaches Looking at the distribution of probe intensities across all arrays at once can sometimes demonstrate that one array is not like the others. These approaches are *boxplots*, *histograms* and *MAplots* [Gentleman et al., 2005], [Alvord et al., 2007].

Boxplots give a simple summary of the distribution of probes [Gentleman et al., 2005] and are a convenient way of graphically depicting groups of numerical data through their five-number summaries: the smallest observation (sample minimum), lower quartile (Q1), median (Q2), upper quartile (Q3), and largest observation (sample maximum). Low quality arrays have boxplots that stand out from the others for a given group of replicates, as evidenced for example by distinctly different ranges or displaced boxes (interquartile ranges, IQR) [Alvord et al., 2007]. For example, in Fig.6.7, an array has a boxplot that clearly stands out from the rest.

Histograms or density plots are used to plot density of data. Low quality arrays have densities that are moved from the others, or that display bimodalities, show uniquely different shapes or other abnormalities [Alvord et al., 2007]. In Fig. 6.7, it can be observed an array whose histogram is moved from the rest and another which has a bimodal shape.

Another exploratory plot for quality assessment is the MA plot for the probe-level data. When two microarrays are being compared, the difference of their log intensities for each probe on each gene (usually denoted 'M') are plotted against their average (usually denoted 'A'). When it is desired to compare more than two arrays, a synthetic array is created by taking the probe wise medians across all arrays. The plot, adds a loess curve fitted to the scatter-plot to summarize any non-linear relationship. Quality problems are most apparent from an MA-plot in cases where the loess smoother oscillates wildly or if the variability of the M values appears to be greater in one or more arrays relative to the others [Gentleman et al., 2005],[Alvord et al., 2007]. For example, in Fig. 6.7, JD-ALD384-v5-U133B.CEL has a lowess curve whose median (M values) is greater then the other arrays.

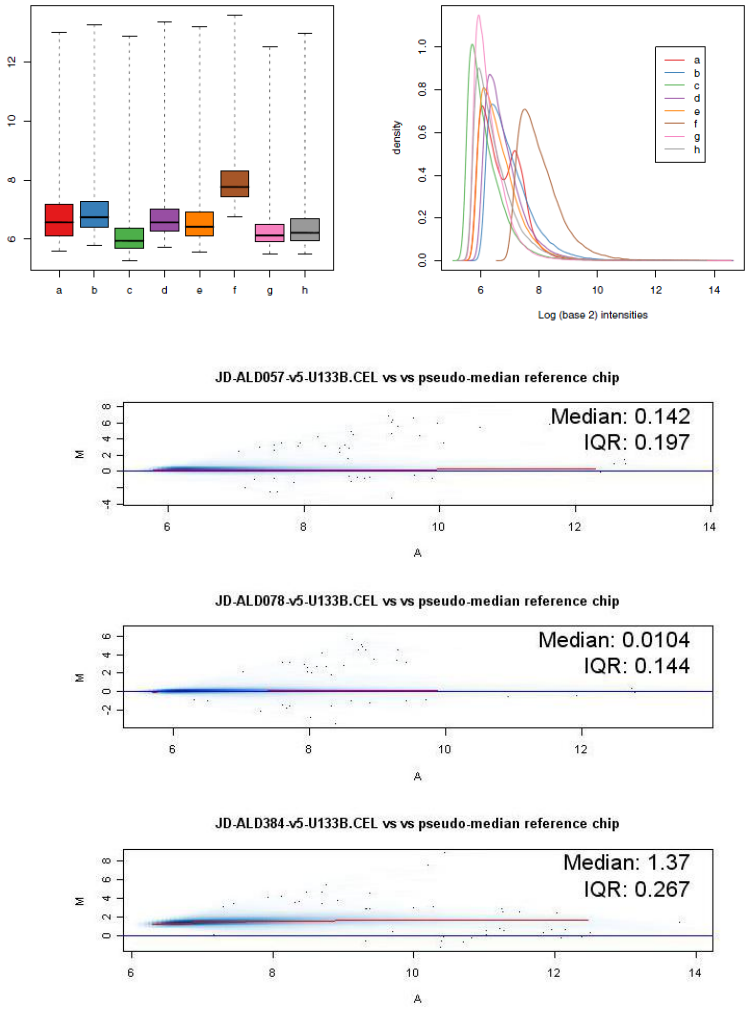


FIGURE 6.7: Boxplots (top-left) and Histograms (top-right) for the MLL.B subset. MAplots (bottom) for some of the samples of MLL.B subset [Ross et al., 2003]

6.2.1.2 Affymetrix quality assessment metrics

Affymetrix software produces a number of quantities for quality assessment of GeneChip data [Affymetrix, 2001] such as the average background, the scale factor and Percent present.

The average background This approach is outlined in the Statistical Algorithms Description Document by Affymetrix [Affymetrix, 2001] and used in the MAS 5.0 software. The chip is divided into a grid of k (default k = 16) rectangular regions.

For each region, the lowest 2% of probe intensities are used to compute a background value for that grid. Then each probe intensity is adjusted based upon a weighted average of each of the background values. According to the guidelines recommended by Affymetrix, the average background values should be ”comparable” to each other.

Scale factors The second quantity is the ”scale factors”. It is accomplished by taking a baseline array to which all other arrays are scaled to have the same mean intensity. The same procedure is followed for each array, so, the scale factor refers to the constant by which every intensity on the chip is multiplied by in the scaling normalization [Alvord et al., 2007]. Affymetrix recommends that the scale factors be within 3-fold of each other.

Percent Present The third quantity is ”percent present”. The Affymetrix detection algorithm generates Present/Marginal/Absent calls by looking at the difference between PM and MM values for each probe pair in a probeset. Probesets are flagged marginal or Absent when the PM values for that probeset are not considered to be significantly above the MM probes [Affymetrix, 2001]. The percent present values should be ”similar” among samples with extremely low values being a possible indication of poor quality.

6.2.1.3 RNA degradation

Through this metric, for every GeneChip probe set, the individual probes are numbered sequentially from the 5’ end of the targeted transcript. When RNA degradation is sufficiently advanced, PM probe intensities should be systematically elevated at the 3’ end of a probe set, when compared to the 5’ end.It has been observed that the 3’/5’ trend is roughly linear for the middle probe positions and lower at the ends. There are no clear guidelines to know how large the slope is to consider an array to have too much degradation and it depends on the chip type [Alvord et al., 2007], although in Fig. 6.8, it can be observed a very different slope for one of the arrays, which may be indicative of poor quality

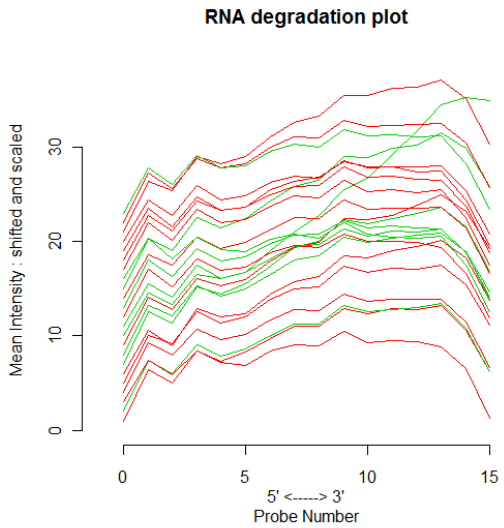


FIGURE 6.8: RNA degradation plot for the CLL subset [Whalen and Gentleman, 2011]

6.2.1.4 Probe-level models

Probe-level models (PLM) may be useful in determining the quality of Affymetrix chips [Alvord et al., 2007]. A PLM is a model that is fit to probe-intensity data [Bolstad, 2011a]. Specifically, a PLM provides parameter estimates for probe sets and chips (arrays) on a probe-set by probe-set (i.e. gene by gene) basis.

Having performed the convolution background correction and normalization procedures (see subsections 6.2.2.1 and 6.2.2.2), the following linear PLM for the background adjusted normalized probe-level data, S_{gij}, may be stated:

$$log_2(S_{gij}) = \theta_{gi} + \phi_{gj} + \epsilon_{gij} \tag{6.1}$$

where, θ_{gi} represents the log-scale expression level for the g-th gene on the i-th array, ϕ_{gj} represents the effect of the j-th probe representing gene g-th gene and ϵ_{gij} represents the measurement error. Note that, following background correction and normalization procedures, the signal S_{gij}, is, in fact, the PM (perfect match) value for the j-th probe on the g-th gene on the i-th array. The fitted object contains

information regarding the parameter estimates, standard errors, weights, residuals and signed residuals [Bolstad, 2011a].

Numerous useful quality assessment tools can be derived from the PLM fitted data such as chip pseudo-images and Relative Log Expression (RLE) and Normalized Unscaled Standard Error (NUSE) plots.

Chip pseudo-images Chip pseudo-images show the weights, residual and signed residuals of the fitted PLM data and may help to discover artifacts in the data set [Bolstad, 2011a]. Thus, for each array of a data set, one looks for any anomalies, artifacts or non homogeneous patterns in the following plots [Gentleman et al., 2005], [Alvord et al., 2007]:

- Gray scale image of the log intensities (the same as the exploratory data analysis).
- PLM weights plot. They use topographical coloring so that light areas indicate high weights and dark areas (green in color plots) indicate significant down-weighting of mis-performing probes .
- PLM residual plot. Images based on residuals are "dark" for negative residuals (blue in color plots) and "light" for positive residuals (red in color plots). It must be checked that the positive and negative residuals are homogeneously spread out across the image, otherwise, it may be indicative of a defective array.
- PLM signed residuals. The image of the signs of the residuals report either +1 or -1 depending on whether the residual is positive (red) or negative (blue). This can sometimes make visible effects that might not be apparent in the other plots. This image highlights the power of the PLM procedures at detecting a subtle artifact that might otherwise be missed completely.

For example Fig. 6.9 displays the resulting chip pseudo-images for a given array of the *AmpAffyExample* package [Irizarry, 2011b]. Fig.6.9 (top-left) is an image of the log intensities with no obvious spatial artifact. However, Fig.6.9 (top-right,

bottom-left and bottom-right) shows PLMweights and a ”ring” artifact is clearly visible. This artifact can be seen in all the PLMimages.

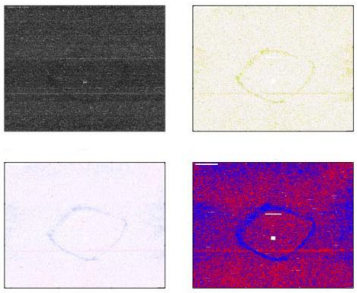


FIGURE 6.9: Chip pseudo-images based on PLM fit make visible subtle artifacts [Gentleman et al., 2005].

With larger artifacts, an issue of concern is whether data from a particular array is of poorer quality relative to other arrays in the data set. This question can be addressed using other procedures provided by the fitted PLM object such as the *Relative Log Expression* (RLE) and the *Normalized Unscaled Standard Error* Plot (NUSE).

Relative Log Expression (RLE) plot This plot is constructed as follows: first, start with the log scale estimates of expression $\hat{\theta}_{gi}$ for each gene g on each array i obtained from the PLM fit . Next, compute the median value across arrays for each gene, $m(g)$, and define the RLE as

$$RLE_{gi} = \hat{\theta}_{gi} - m(g) \tag{6.2}$$

That is, the median value of the $g - th$ gene is subtracted from each gene g on each array i. These relative expressions are then displayed with a boxplot for each array. It is reasonable to assume that the majority of genes are not changing in expression among conditions. The majority of these non-differential genes are displayed on the RLE plot by the boxes (IQRs). Ideally, these boxes should have small spread and be centered at $RLE = 0$. An array with quality problems may result in a box that has relatively greater spread or that is not centered near $RLE = 0$ [Gentleman et al., 2005].

For example, using the whole *MLL.B* example from the *ALLMLL* data set [Ross et al., 2003], it can be observed in Fig.6.10 an array that deviates considerably and has more spread relative to the remaining boxes (number 2, JD-ALD051-v5-U133B.CEL).

Normalized Unscaled Standard Error (NUSE) plot Another graphical tool is the NUSE plot. For this plot, one begins with the standard error estimates obtained for each gene g on each array i from the PLM fit. Call this $SE(\hat{\theta}_{gi})$. Variability may differ considerably among genes. To correct for this, one may standardize these standard error estimates such that the median standard error across arrays is 1 for each gene [Bolstad, 2011a]. Specifically:

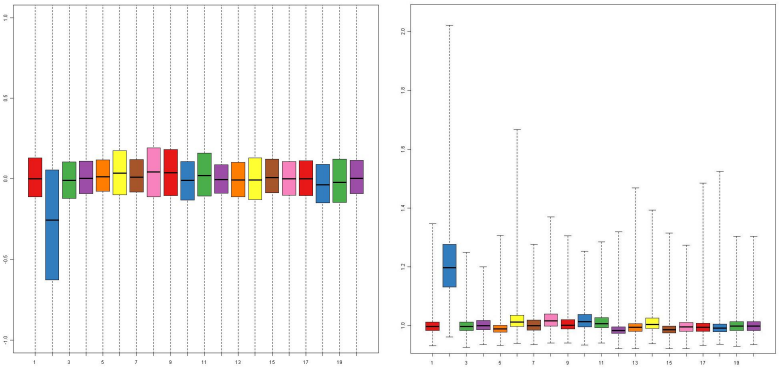
$$NUSE(\hat{\theta}_{gi}) = \frac{SE(\hat{\theta}_{gi})}{med_i\{SE(\hat{\theta}_{gi})\}} \tag{6.3}$$

A low quality array on this plot might be indicated by a box that is significantly elevated with respect to $NUSE = 1$ or shows more spread relative to other arrays [Bolstad, 2011a].

Using the same example, *MLL.B* data set, it can be observed in Fig.6.10 an array that deviates considerably and has more spread relative to other boxes (number 2, JD-ALD051-v5-U133B.CEL).

6.2.2 Data pre-processing

A typical microarray experiment has many different sources of variation which can be attributed to biological and technical causes. Biological variation results from tissue heterogeneity, genetic polymorphism, and changes in mRNA levels within cells and among individuals due to sex, age, race, genotype-environment interactions and other ”living” factors. Biological variation is then of interest to investigators. On the other hand, preparation of samples, labeling, hybridization, and other steps of microarray experiment can contribute to technical variation, which can significantly impact the quality of array data. To ensure highly reproducible microarray data, technical variation should be minimized by controlling the quality



(a) (b)

FIGURE 6.10: RLE (a) and NUSE (b) plots for the 20 HGU-133B arrays in the ALLMLL data set

of the RNA samples, and by efficient labeling and hybridization, but in most of the cases, controlling these tasks are not enough to avoid such type of variation. Therefore, since those systematic non-biological sources of variation are always present and can mask real biological variation, significant pre-processing is required [Bolstad et al., 2003], although this pre-processing must be handled with caution, due to the risk of masking the real biological variation.

Pre-processing has an important role in the earlier stage of microarray data analysis pipeline, because different normalization procedures can lead to different expression data and, therefore, may have an effect in posterior stages, such as the detection of differentially expressed genes Bolstad et al. [2003][Ritchie et al., 2007], phenotype classification [Florido et al., 2010b],[Wu et al., 2005] or the construction of reverse engineering networks [Lim et al., 2007]. Therefore, pre-processing is a critical initial step in the analysis of a microarray experiment, where the goal is to balance the individual signal intensity levels across the experimental factors, while maintaining the effect due to the treatment under investigation, e.g., keep the biological variation as much as possible.

Since our work is focused on Affymetrix Genechips, we are going to describe in detail each step involved in the pre-processing task: background correction, normalization, PM correction and summarization [Lim et al., 2007],[Florido et al., 2010b],[Florido et al., 2009c] (see Fig.6.11).

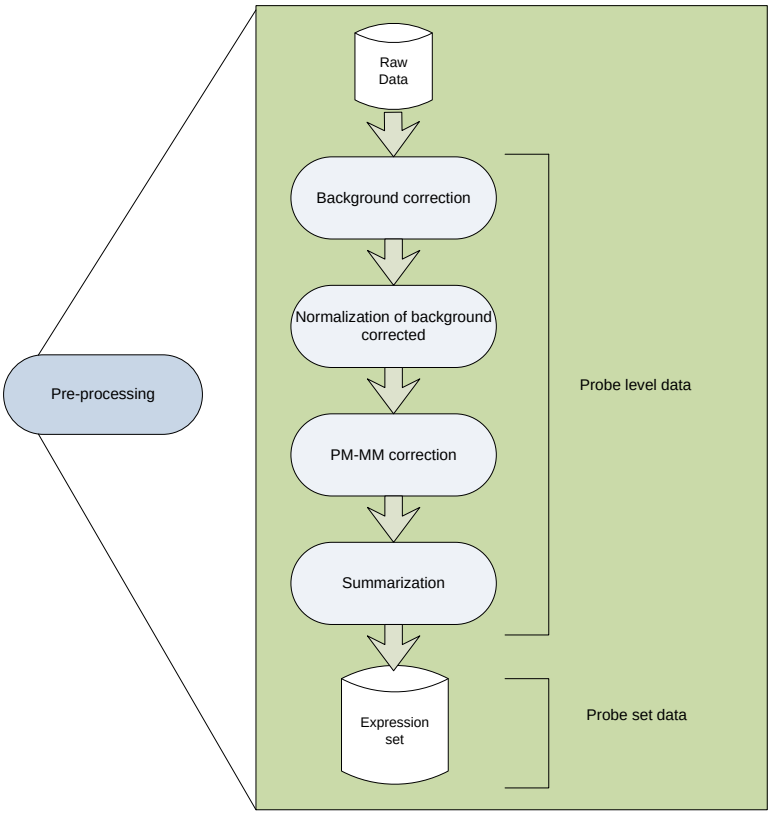


FIGURE 6.11: Pre-processing workflow for Affymetrix Genechips

6.2.2.1 Background correction

This is the first step of pre-processing and removes unspecific background intensities of scanner images. There is some amount of background noise in every scanner image. Sterile water can be labeled and hybridized to a microarray and even though there is no RNA in the sample, the scanner will detect low levels of fluorescence on the chip. An estimate of the background signal, which is the signal due to nonspecific binding of cluorescent molecules or the autofluorescence of the chip surface [Choe et al., 2005], can be estimated using three possible metrics: (1) *MAS* background adjustment, which is used in the MAS 5.0 software [Affymetrix, 2001] and corrects both PM and MM probes; (2) *Robust Multichip Average* (RMA) convolution, which is an implementation of the background adjustment carried out as part of the RMA pre-processing method [Irizarry et al., 2003] which corrects only the

PM signals and (3) *gcrma* [Wu et al., 2004], which it is designed to account for background noise, as well as non-specific binding.

6.2.2.2 Normalization

Even if the exact same sample is used on each of several chips, there will be chip to chip differences in the overall distribution of probe intensity values. Normalization procedures attempt to detect and correct systematic differences between chips so that data from different chips can be directly compared. Studies show that the normalizing procedure has a marked impact on the final expression measures [Bolstad et al., 2003],[Wu et al., 2005]. A number of normalization procedures for Affymetrix GeneChips have been proposed:

- *Scaling* (Constant), which is the method taken by MAS 5.0 Affymetrix software [Affymetrix, 2001]. The approach chooses a baseline array (the array having the median of the median intensities) and all arrays are then normalized to this ”baseline”.
- *Quantile*. The goal of the quantile method is to make the distribution of probe intensities for each array in a set of arrays the same [Bolstad et al., 2003].
- Cyclic *Loess*. This method is extended from cDNA microarray data normalization method [Bolstad et al., 2003] and works on all distinct pair-wise combinations of arrays, which is time consuming.
- *Invariant set*. It is a nonlinear, intensity-dependent normalization approach based on a subset of probes which have similar ranks (the rank-invariant set) between two chips [Schadt et al., 2001]. dChip software [Wong, 2011] uses this ”rank invariant set” for the normalization of the summarized gene-level intensity. .
- *Variance Stabilization Method* (VSN). This approach [Huber et al., 2002] assumes that most of the genes on the arrays are not differentially expressed in a given experiment and utilizes the arcsine rather than log transformation

to stabilize the variance so as to remove the dependence of the variance on the total intensity.

6.2.2.3 PM-MM correction

Mismatch probes are included on Affymetrix GeneChips to quantify non-specific and cross-hybridization. The aim of this step is to correct for cross-hybridization or non-specific binding. Theoretically, the intensity measured in the MM-probe gives us an estimation of the RNA that binds in the PM-probe having just a partial match. Therefore, the difference between the PM-measure and the MM-measure may yield a better index to estimate the true expression of the gene. We emphasize two possible methods: (1) *MAS* [Affymetrix, 2001], in which the MM probe signal is substracted and (2) *PMonly*, in which the MM value is not substracted, using uncorrected PM probes alone [Gautier et al., 2004].

6.2.2.4 Summarization

In Affymetrix GeneChip arrays each gene is represented by a set of several PM and MM probe pairs. Thus, probe intensities for each probe set should be summarized to define a measure of expression representing the amount of the corresponding mRNA species. Several model-based approaches to this problem have been proposed: (1) *Tukey-biweight* (MAS), which is followed by Affymetrix [Affymetrix, 2001] and takes into account PM and MM probes; (2) *MBEI* [Li and Wong, 2001] used in the dChip software [Wong, 2011], which also takes into account PM and MM probes; (3) *Median polish*, which is the summarization used in the RMA expression summary Irizarry et al. [2003] and only uses information of PM probes; (4) *Avgdiff*, used in MAS 4.0 Affymetrix software [Affymetrix, 1999]; (5) *Factor Analysis for Robust Microarray Summarization* (FARMS) method [Hochreiter et al., 2006] and (6) *Distribution Free Weighted* (DFW) method [Chen et al., 2007b].

As mentioned above, a pre-processing method is then a combination of methods of background correction, normalization, PM correction and summarization. Not all

combinations of methods are allowed, for example, RMA background correction method adjusts only PM probe intensities (*rma*) and so they should only be used in conjunction with the *PM only* PM-MM correction [Irizarry et al., 2003]. Some pre-processing methods are standard in the literature for a specific combination of background correction, normalization, PM correction and summarization methods, for example, RMA pre-processing method runs *rma* as background correction, *quantiles* as normalization, *PM only* as PM correction and *medianpolish* as summarization (see table B.1 in Appendix B). Non-standard pre-processing methods are known in the literature as "custom" [Zhu et al., 2008].

6.2.2.5 Evaluating a pre-processing method

As can be noticed, there are many procedures for each pre-processing step and, thus, many possible combinations of *background correction, normalization, PM correction* and *summarization* methodologies. Different pre-processing procedures can lead to different expression data and, therefore, may have an effect in posterior stages, such as the detection of differentially expressed genes [Ritchie et al., 2007], phenotype classification [Florido et al., 2010b],[Wu et al., 2005] or the construction of reverse engineering networks [Lim et al., 2007] as previously stated. Therefore, pre-processing is a critical initial step in the analysis of a microarray experiment and, for this reason, an objective way of evaluating a pre-processing method is needed with the aim of choosing the best ones. For this purpose, there are some works in the literature that try to deal with this problem, for example, J.P.Florido et al. [Florido et al., 2009c] evaluate the performance of the pre-processing methods using three different metrics:

- **Replicate variability**. This criterion has been used in works such as in [Xiong et al., 2008] and it is based on the assumption that the expression level of a gene should ideally remain the same across multiple replicated slides. For m replicated slides, the variability of m values for each gene can be used to compare normalization methods. The mean of the standard deviation over all genes is a global measure of the normalization methods. A smaller mean is indicative of better performance of the pre-processing procedure.

- **Kolmogorov-Smirnov (K-S) test**. The K-S test is a goodness-of-fit test of two continuous distributions and has also been adopted in [Xiong et al., 2008]. When using this statistic for evaluating the quality of a pre-processing method, we base on the hypothesis that an effective pre-processing procedure should result in two similar, ideally identical, distributions with a small, ideally zero-valued K-S statistic between two replicated slides. On the other hand, two different distributions will generate a large K-S statistic.

- **Spearman Rank Correlation Coefficient**. It is a nonparametric (distribution-free) rank statistic which measures the strength of the associations between two variables. Lim et al. [Lim et al., 2007] have also used this statistic to evaluate the quality of a pre-processing method. This approach is based on the assumption that, given an experiment, the correlation coefficient among replicated slides will be increased after the pre-processing stage.

On the other hand, Lim et al., [Lim et al., 2007] conclude that the choice of a pre-processing method strongly affects the correlation structure in the data, that is, correlation artifacts can be introduced in it. This seriously undermines the utilization of pre-processing procedures, at least in their standard form, upstream in microarray data analysis pipeline. More precisely, they found that *GCRMA* pre-processing method introduces correlation artifacts for gene pairs that are not expected to be co-expressed. As a result, this method is not suitable to the reconstruction of cellular networks from expression profile data, including the inference of networks topological properties and gene functional relationships based on co-expression measurements. Although *GCRMA* method has been fixed and does not introduce correlations any more, it is important not only to evaluate a pre-processing method in terms of the quality metrics (replicate variability, K-S test and Spearman Correlation Coefficient), but also to check whether a pre-processing method introduces correlation artifacts by applying the quality metrics to uninformative raw data, in which raw signal intensities for each probe pairs in the data set experiment are randomly permuted [Lim et al., 2007].

6.2.3 Detection of differentially expressed genes

This is the next step in the microarray analysis pipeline. Fundamental to the task of analyzing gene expression data, is the need to identify genes whose patterns of expression differ according to phenotype or experimental condition. Gene expression is a well coordinated system, and hence measurements on different genes are in general not independent. For this reason, statistical tests to assess differential expression [Gentleman et al., 2005] are used and one may distinguish between parametric tests, such as the *t-test*, and non-parametric tests, such as the *Mann-Whitney* test or *permutation* tests. However, these approaches have a number of drawbacks: most important is the fact that a large number of hypothesis tests is carried out, potentially leading to a large number of falsely significant results. *Multiple testing* procedures have been proposed to deal with this problem and allow one to assess the overall significance of the results of a family of hypothesis tests. They focus on specificity by controlling type I (false positive) error rates such as the *family-wise error rate* or the *false discovery rate* [Gentleman et al., 2005]. Still, multiple hypothesis testing remains a problem, because an increase in specificity, as provided by *p-value* adjustment methods, is coupled with a loss of sensitivity, that is, a reduced chance of detecting true positives. Furthermore, the genes with the most drastic changes in expression are not necessarily the ”key players” in the relevant biological processes. This problem can only be addressed by incorporating prior biological knowledge into the analysis of microarray data, which may lead to focusing the analysis on a specific set of genes. Other approaches such the significant analysis of microarrays (SAM) statistic [Tusher et al., 2001], have been proposed for the problem of large number of falsely significant results.

On the other hand, the number of hypotheses to be tested can often be reasonably reduced by removing control genes and/or performing non-specific filtering procedures, whose aim is to remove control genes which are not of interest of the researcher and perform a non-specific filtering, which consists in removing genes that, e.g., due to their low overall intensity or variability, are unlikely to carry information about the phenotypes under investigation.

Many microarray experiments involve only few replicates per condition, which makes it difficult to estimate the gene-specific variances that are used, e.g., in the

t-test. It is important to notice that, using the *t-statistic* of *t-test*, the statistical significance, evaluates the average difference between the two population values *relative* to their variances. Small average differences in the context of large variances could result in a large, and therefore, *non significant P-values*. On the other hand, small average differences with small variances could result in *significant P-values*. Large average differences with small variances might result in significant P-values. And, finally, large average differences with large variances could result in *non significant P-values*. In summary, it is possible to have large log-fold changes that are not statistically significant because the populations exhibit much variability. It is also possible to have small log-fold changes that are highly statistically significant because the populations exhibit little variability [Alvord et al., 2007].

For these reasons some researchers have proposed alternative statistics that borrow information about variability across all genes to obtain a more stable estimate of gene specific variance. These procedures attempt to minimize the impact of genes with very large and very small variances. The resulting statistics are referred to as *modified, penalized, attenuated* or *regularized t-statistics*. An example of such a modified t-statistic is given by the *moderated t-statistic* [Smyth, 2004], which is obtained through an Empirical Bayes approach which employs a global variance estimator computed on the basis of all genes' variances.

6.2.4 High-level analysis: Classification

The last level of microarray data analysis pipeline is related to the high-level analysis, such as Cluster analysis, Classification, GO-Analysis or Gene Set Enrichment Analysis [Rainer et al., 2006], [Florido et al., 2010b], [Subramanian et al., 2007].

In this subsection, some basic concepts about classification analysis are described, since one of the major use of microarrays is related to phenotype classification via expression-based classifiers. Classifying cancer tissues based on their gene expression profiles has the promise of providing more reliable means to diagnose and predict various types of a disease [Xiong et al., 2007].

The most well known methodologies to perform microarray-based classification are k-Nearest Neighbors, backpropagation and probabilistic neural networks, weighted

voting methods, decision trees and kernel methods such as Support Vector Machines (SVMs) [Statnikov et al., 2005]. SVM is usually preferred in microarray-based classification [Chu and Wang, 2005] due to its outperformance compared to other paradigms and its suitability to two special aspects of microarray data: high dimensionality and small sample size. Kernel methods represent one way to cope with the curse of dimensionality [Xiong et al., 2007].

6.2.4.1 Support Vector Machine-based classifiers

Support vector machines (SVMs) are learning kernel-based systems [Ressom et al., 2008] that use a hypothesis space of linear functions in high dimensional feature spaces. Unlike artificial neural networks, which try to define complex functions in the input feature space, the kernel methods such as SVMs perform a nonlinear mapping of the complex data into high dimensional feature spaces and then use simple linear function to create linear decision boundaries. Thus, the problem is related to choose a suitable kernel for the data projection. Parameters of an SVM model are determined based on structural risk minimization to search for one target known as the optimal hyperplane. Given a training set of instance label pairs $\{(x_k, y_k); k = 1, ..., n\}$, where $x_k \in \mathbb{R}^p$ and $y_k \in \{1, -1\}$, SVMs require the solution of the following optimization problem [Cortes and Vapnik, 1995]:

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i$$

subject to

$$y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i, \xi_i \geq 0$$

where w and b are the weight vector and bias of the hyperplane and $\xi_i' s$ are non-negative scalar variables called slack variables that measure the deviation of a data point from the ideal condition of pattern separability. Here, training vectors x_i are mapped into a higher dimensional space by the function ϕ, where $\phi(x_i)^T \phi(x_j)$ is called the kernel function, denoted by $K(x_i, x_j)$. The most commonly used kernels are:

- Linear: $K(x_i, x_j) = x_i^T x_j$
- Polynomial: $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \gamma > 0$
- RBF: $K(x_i, x_j) = exp(-\gamma \|x_i - x_j\|^2), \gamma > 0$

where γ, r and d are the kernel parameters. For a given set of training samples and kernel, SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space. $C > 0$ is the penalty parameter of the error term.

During the operation phase, the optimal hyperplane $w \cdot x + b$ and the corresponding decision function $d(x) = w \cdot x + b$ are used to determine the class labels for new samples. Here, w and b are the optimal values obtained by solving the above optimization problem.

The one vs. all (OVA) approach extends the functionality of SVM from binary to multi-class classification. The OVA constructs a binary classifier for each group. Thus, for a k-group classification, k binary SVMs are needed. Each binary SVM classifier creates a decision boundary that separates the group it represents from all other groups. The k binary SVM classifiers compete to categorize an unknown spectrum into their corresponding group. The SVM with the highest decision value (farthest from the decision boundary) "wins" the competition, assigning the unknown spectrum to its group [Ressom et al., 2008].

6.2.4.2 Gene selection

Because the sample size is much smaller than the dimensionality, too many genes may not be helpful, sometimes may even be harmful, for the class discrimination. Selecting the most discriminatory genes and removing the rest not only reduce the computational complexity, but also substantially improve the performance of microarray-based classifies [Xiong et al., 2007]. Several approaches have been proposed in the literature for gene selection, such as the SVM-based recursive feature elimination [Guyon et al., 2002], the BW ratio [Ye et al., 2004], the Principal component analysis (PCA) or the simpler, although commonly used, *t-test*-based approach [Chu and Wang, 2005], which is used to measure how large the difference is between the distribution of two groups of samples.

6.3 Tools for microarray data analysis

As stated in the introduction, Bioconductor provides hundreds of packages for all aspects of microarray data analysis. For example, *affy* [Irizarry, 2011a] is often used for quality assessment and pre-processing of Affymetrix GeneChip data, while *limma* [Smyth, 2011] is useful in detecting differentially expressed genes.

However, for scientists without adequate programming experience, several GUI tools are available on the world wide web to analyze microarray experiments in a user-friendly environment. Some of them are:

- **EzArray**. EzArray [Zhu et al., 2008] is a web-based Affymetrix expression array data management and analysis system, which includes three highly automated and seamlessly integrated data analysis programs named:
  - PreQ, for quality assessment through Boxplots and MAplots (section 6.2.1.1), quality assessment metrics from Affymetrix (section 6.2.1.2), RNA degradation (section 6.2.1.3) and plots from Probe-level models (section 6.2.1.4). PreQ also offers tools for pre-processing, with several choices for background correction, normalization, PM correction and summarization for custom or standard pre-processing methods (section 6.2.2)
  - ProS, which implements several statistical procedures for detecting differentially expressed genes (section 6.2.3).
  - RepA, for report generating and gene annotation.

  Microarray data can be from users' experiments (Custom Array Data), published raw array data (deposited CEL supplementary files in GEO), or GEO curated DataSets (GDS records). In addition, a number of standalone tools have been included in EzArray, including tools for gene annotation, array probe search, R shell for interactive execution of R scripts, and R batch for batch execution of R scripts. The last version of *EzArray* is called *BxArrays* (http://bioinforx.com/lims/microarray-gene-expression-data-analysis/bxarrays), which allows different microarray technologies such as Nimblegen, Affymetrix, Agilent or Illumina.

- **CARMAweb**. It is a web application [Rainer et al., 2006] designed for the analysis of microarray data. The following features are provided:

  1. Support for Affymetrix, two-color and ABI microarrays.
  2. Import of raw data from a variety of imaging software tools for two-color microarrays (Agilent Feature Extraction, ArrayVision, BlueFuse, GenePix, ImaGene, QuantArray, SPOT or raw data files from the Stanford Microarray Database).
  3. A complete analytical pipeline for Affymetrix, two-color and ABI microarrays including modules for preprocessing (custom or standard preprocessing methods), detection of differentially expressed genes, clustering and visualization, as well as GO mapping. Microarray quality is assessed through Boxplots and Histograms (section 6.2.1.1).
  4. Generation of comprehensive analysis report files.

- **GEPAS**. It is a web-based tool [Vaquerizas et al., 2005] for the analysis of genomic data. This suite of programs include tools for data quality assessment (Raw images of data set, Boxplots, Histograms, RNA degradation plot, and MA plots), normalization of data for different platforms (Affymetrix, Agilent, Codelink, etc.) using custom pre-processing methods, class discovery in genes or experiments by diverse clustering methods, differential gene expression methods for class comparison (including two-class, multiclass, continuous parameters -such as the level of a metabolite-, and survival analysis), class prediction (predictors can be built and further applied to class prediction in new samples), methods for analysing time course and dose response experiments and the possibility of different types of genomic analysis by array CGH. GEPAS provides a direct access to different functional profiling and functional annotation facilities

- **MAGMA**. The web application MAGMA [Rehrauer et al., 2007] provides tools for quality assessment (MAplots), pre-processing, analysis for differentially expressed genes using linear models and biological annotation on two-channel microarray data.

- **dChip**. DNA-Chip Analyzer (dChip) (http://www.dchip.org) is a Windows-based software package for probe-level (e.g. Affymetrix platform) and high-level analysis of gene expression microarrays and SNP microarrays [Wong, 2011].

- **RMAExpress**. It is a standalone GUI program for Windows and Linux to compute gene expression summary values for Affymetrix Genechips data using the Robust Multichip Average expression summary (RMA pre-processing method) and to carry out quality assessment using probe-level metrics, box-plots and histograms [Bolstad, 2011b]

- **Microarray Cel file Summarizer** (mu-CS). This software [Guzzi and Cannataro, 2010] is a cross-platform tool, based on a client-server architecture, for the automatic normalization, summarization and annotation of Affymetrix data. It enables users to read, pre-process and analyse microarray data, avoiding the manual invocation of external tools the manual loading of pre-processing libraries, and the management of intermediate files.

Let us compare these tools in terms of the first two steps of microarray data analysis pipeline:

- Experiment quality assessment. *MAGMA* and *CARMAweb* provides limited quality assessment of the experiment. *dChip* and *mu-CS* have not quality assessment and although *EzArray*, *GEPAS* and *RMAexpress* provide many tools for quality assessment, they leave to the researcher the decision of removing a sample of the experiment if *it seems* to be defective given the quality plots used with no clear guidelines. Thus, if the user has no experience on microarrays, this task can be cumbersome.

- Pre-processing. All the tools described above provide a set (or subset) of standard pre-processing methods (*CARMAweb*, *EzArray*, *MAGMA*, *dChip*, *RMAexpress*, *mu-CS*) and some of them (*CARMAweb*, *EzArray*, *GEPAS*) also provide custom pre-processing methods, but, in any case, they leave to the (unexperienced) user to decide which pre-processing method to use.

Chapter 7

A2TOOL - Affymetrix microarray Analysis Tool

7.1 Goals

In this chapter, the tool developed for this dissertation entitled A^2**TOOL: Affymetrix microarray Analysis Tool** is described. This tool has been designed for the three first steps described in Section 6.2 for microarray data analysis pipeline: Quality data analysis (section 6.2.1), Data pre-processing (section 6.2.2) and the detection of differentially expressed genes (section 6.2.3). Given an Affymetrix raw data set experiment, this tool is outlined as follows (see Fig.7.1):

1. The quality of the raw data set is analyzed through the quality metrics described in section 6.2.1: Image plots of the probe level data, Box plots, Histograms, MA plots, Affymetrix quality assessment metrics (Average Background, Scale factors and Percent Present), RNA degradation and Probe-level models-based quality tools. A^2**TOOL** performs an automated quality assessment of the raw data given in the experiment and points defective arrays (if any) according to the criteria described in section 6.2.1. At the end of this step, a report in PDF format is generated to help the researcher to take

a decision: either remove or keep defective arrays (if any) and go on with the next phase of the analysis.

2. The whole raw data set, or a subset of it depending on the decision taken in the previous step, is pre-processed using all possible and allowed combinations of Background correction, Normalization, PM correction and Summarization methods described in section 6.2.2. Let us denote all possible pre-processing methods as the set P. The goal is to select the best pre-processing methods for a given raw data set according to the quality metrics described in section 6.2.2.5: *replicate variability*, *Kolmogorov-Smirnov test* and *Spearman Rank Correlation Coefficient* [Florido et al., 2009c], that is, a subset $P' \subseteq P$ of preprocessing methods are obtained, i.e., a set of pre-processed data (expression sets) P' is selected.

3. The next step carried out by our tool is to detect differentially expressed genes on each pre-processed data set (expression set) obtained as a result of applying the selected pre-processing methods in P' to raw data. At the end of this stage, a second report in PDF format is generated with a list of differentially expressed genes detected in each pre-processed data set and an intersection and union lists obtained through the intersection and union respectively of differentially expressed genes detected in all pre-processed data sets. The goal is to provide the most reliable genes selected as differentially expressed and a complete list of candidate genes that are likely to be differentially expressed.

A^2**TOOL** is implemented in R and all the packages used are from *Bioconductor* [Gentleman et al., 2004] repository.

This chapter is structured as follows: the tool proposed is presented and explained in detail in section 7.2.1 in terms of quality data analysis, data pre-processing and the detection of differentially expressed genes. In section 7.3 the results of applying A^2**TOOL** to the Chronic Lymphocytic Leukemia (CLL) data set are shown (section 7.3.1). Furthermore, the results of studying the effect of pre-processing methods on microarray-based cancer classifiers are also presented in section 7.3.2.

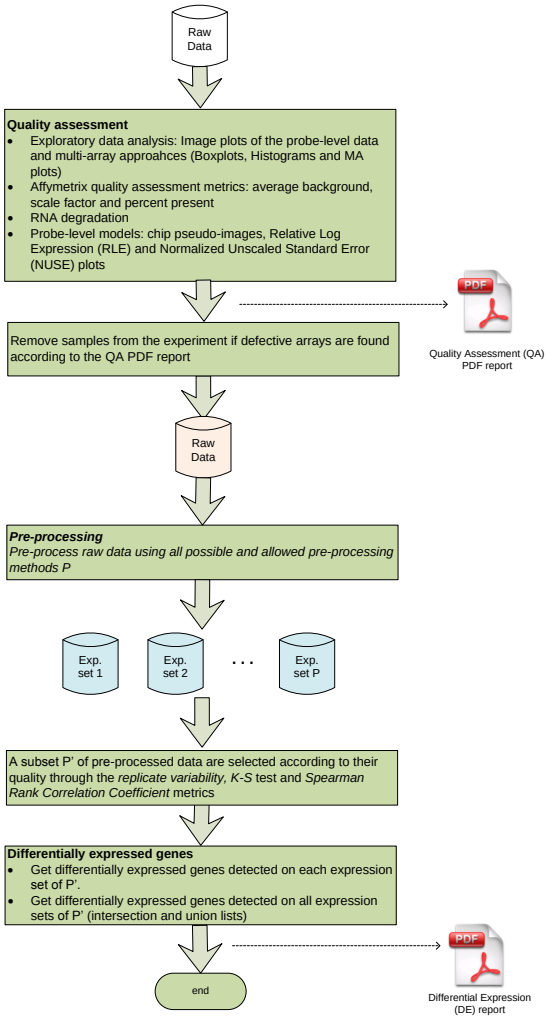


FIGURE 7.1: A^2TOOL workflow

7.2 A^2TOOL

7.2.1 A^2TOOL: Quality data analysis

The first step carried out by **A^2TOOL** is to analyze according to the criteria given in section 6.2.1 the quality of the experiment data set $A = \{a_1, a_2, ..., a_n\}$ where n is the number of arrays or samples in the experiment.

7.2.1.1 Exploratory data analysis: image plots and multi-array approaches

Image plots of the probe-level data The user has to examine the image plots of the log intensities for each microarray contained in the experiment to look for spatial artifacts or other non homogeneous patterns in the image plots across all arrays such as lighter or darker arrays and/or spatial artifacts such as rings or shadows [Gentleman et al., 2005]. For plotting images of a raw data set, the *affy* R package [Irizarry, 2011a],[Gautier et al., 2004] has been used.

Boxplots As stated in subsection 6.2.1.1, we look for boxplots that stand out from the others for a given group of replicates, as evidenced by distinctly different range or displaced boxes (interquartile ranges, IQR) [Alvord et al., 2007]. For plotting the boxplots of a raw data set, *affy* R package [Irizarry, 2011a] is used. If the experiment data set (raw data) has k group of replicates, $G = \{group_1, group_2, ..., group_k\}$, then for each $group_j \in G$, the procedure used by A^2**TOOL**, to detect defective arrays is:

1. Get the set of arrays belonging to $group_j$, that is, $A_j = \{a_1, a_2, ..., a_m\} \in group_j$.
2. Get the interquartile range (IQR) of $A_j = \{a_1, a_2, ..., a_m\} \in group_j$, that is, $IQR_j = \{iqr_1, iqr_2, ..., iqr_m\}$.
3. Get the mean value of the lower quartile (Q1), μ_{Q1}, and the mean value of the upper quartile (Q3), μ_{Q3}, of IQR_j.
4. For each $iqr_i \in IQR_j$, get its lower boxplot quartile (Q1), $Q1_i$, and its upper boxplot quartile (Q3), $Q3_i$. Check whether $Q1_i > \mu_{Q3}$ or $Q3_i < \mu_{Q1}$. If so, the array i is detected as defective.

Histograms According to the criterion described in [Alvord et al., 2007], we look for densities that display bimodalities, are moved excessive from the others, show uniquely different shapes or other abnormalities. For this purpose, the "bimodality index" from *ClassDiscovery* R package [Wang et al., 2009a],[Coombes,

2010] is used. This index is a continuous measure of the extent to which a set of data fits a two-component mixture model, that is, it restricts to trying to decide if the data is better described by a one-component (unimodal) or two-component (bimodal) distribution through the parameter *bi*. Larger values of *bi* represent "more strongly" bimodal distributions. The rule-of-thumb is that with at least 200 samples, then $bi > 1.1$ represents believable bimodality. Thus A^2**TOOL** computes the bimodal index for each array $a_i \in A$, obtaining $BI = \{bi_1, bi_2, ..., bi_n\}$. Since raw microarray data have more than 200 samples per microarray, when $bi > 1.1$ for a given array a_i, A^2**TOOL** detects a bimodal distribution on that sample. Some experiments run using this quality metric reveals that smooth bimodalities are not detected, but strong do. So, we suggest the user to inspect the histogram plot to detect small bimodalities, different shapes or other bimodalities. Therefore, the histogram plot is a more manual criterion rather than automatic.

MA plots A^2**TOOL** runs a different MA plot for each condition, treatment or group of replicates $group_j$, in which each array $a_i \in group_j$ is compared to a *synthetic* array built from the samples that belong to $group_j$. Quality problems are most apparent from an MA-plot in cases where the loess smoother oscillates wildly or if the variability of the M values appears to be greater in one or more arrays relative to the others [Alvord et al., 2007]. To automate the detection of defective arrays, A^2**TOOL** proceeds the same as in the array quality metrics package [Kauffmann and Huber, 2011],[Kauffmann et al., 2009], in which a boxplot of the mean values of M for each sample is calculated and those arrays classified as outliers using this nonparametric statistic will be marked as defective samples. Thus, if the experiment data set has k group of replicates, that is $G = \{group_1, group_2, ..., group_k\}$, then for each $group_j \in G$:

1. Get the set of arrays belonging to $group_j$, that is, $A_j = \{a_1, a_2, ..., a_m\} \in group_j$. The *synthetic* array is created taking the probe wise medians across all arrays $a_i \in A_j$. This is the reference array, ref_j.

2. The difference between each array $a_i \in A_j$ and the reference array ref_j for each probe on each gene, denoted as m_i, is computed, that is, $M = \{m_1, m_2, ..., m_m\}$ where $m_i = a_i - ref_j$.

3. The mean of the absolute values of m_i, μ_{m_i} for each array is computed, that is, $\mu M = \{\mu m_1, \mu m_2, ..., \mu m_m\}$ where $\mu m_i = mean(|m_i|), m_i \in M$.

4. A boxplot of the μM set is built to detect outliers and those arrays whose $\mu_{m_i} \in M$ value is lower or greater than the smallest and largest observation respectively in the boxplot are pointed as defective, that is, an array $a_i \in A_j$ is considered of low quality if $\mu m_i < Q1 - 1.5 \cdot IQR$ or $\mu m_i > Q3 + 1.5 \cdot IQR$, where $Q1$ and $Q3$ are the lower and upper quartile respectively of the boxplot and IQR is the inter-quartile range $IQR = Q3 - Q1$.

7.2.1.2 Affymetrix quality assessment metrics

Affymetrix quality assessment metrics are implemented in the *simpleaffy* R package [Wilson and Miller, 2005], [Wilson and Miller, 2011]:

The average background According to the guidelines recommended by Affymetrix [Affymetrix, 2001], the average background values should be "comparable" to each other. Assuming normality for the average background values, all must fall within two standard deviation of the mean [Alvord et al., 2007], thus, A^2**TOOL** detects any microarray whose average background value is outside of two standard deviation of the mean, that is, given the average background values of all arrays in the experiment $AB = \{ab_1, ab_2, ..., ab_n\}$, our tool checks whether:

$$\begin{gathered} ab_i > \mu_{AB} + 2\sigma_{AB} \\ or \\ ab_i < \mu_{AB} - 2\sigma_{AB} \end{gathered} \tag{7.1}$$

$\forall ab_i \in AB$, where μ_{AB} and σ_{AB} are the mean and standard deviation respectively of AB set and i is the $i-th$ array of the experiment data set $A = \{a_1, a_2, ..., a_n\}$.

Scale factor Affymetrix [Affymetrix, 2001] recommends that the scale factors be within 3-fold of each other. The procedure taken by A^2**TOOL** to detect defective arrays using this quality metric is the same as the one followed in the *simpleaffy* R package:

1. Get scale factor values for each array in the experiment in log_2 scale, that is, $log(SC) = \{log_2(sc_1), log_2(sc_2), ..., log_2(sc_n)\}$ where n is the number of arrays in the experiment.

2. Get the mean of all scale factor values, μ_{SC}, from $log(SC)$.

3. Get the upper,U_{sc}, and lower, L_{sc}, bound values used to decide whether an array is defective or not: $U_{sc} = (3/2 + \mu_{SC})$ and $L_{sc} = (-3/2 + \mu_{SC})$

4. If $log_2(sc_i) > U_{sc}$ or $log_2(sc_i) < L_{sc}, \forall log_2(sc_i) \in log(SC)$, then, the array $a_i \in A$ is detected as defective.

Percent present As stated in subsection 6.2.1.2, the percent present values should be "similar" among samples with extremely low values being a possible indication of poor quality. An array $a_i \in A$ is detected as defective by A^2**TOOL** if it has more than 10% of difference with respect to the maximum value of percent present, which is considered as the best quality [Alvord et al., 2007] [Wilson and Miller, 2011]. The procedure is as follows:

1. Get the percent present values for all arrays of the experiment A in %: $PP = \{pp_1, pp_2, ..., pp_n\}$.

2. Get the maximum value of percent present, that is, $max_{PP} = max(PP)$

3. An array $a_i \in A$ is considered of low quality if $|pp_i - max_{PP}| > 10, \forall pp_i \in PP$

7.2.1.3 RNA degradation

As stated in subsection 6.2.1.3 when there is RNA degradation, PM (perfect match) probe intensities are elevated at the 3' end of a probe set, when compared to the 5' end [Gentleman et al., 2005][Alvord et al., 2007].

Since there are no clear guidelines to know how large the slope is to consider an array to have too much degradation and, also, the result depends on the chip type [Alvord et al., 2007], A^2**TOOL** adopted the difference in slopes among chips to detect RNA degradation: if the arrays have similar slopes, then comparisons within

genes across arrays may still be valid. On the other hand, a plot with one or more arrays with very different slope is indicative of a possible problem [Gentleman et al., 2005]. The procedure that A^2**TOOL** follows to detect defective arrays is the following:

1. Get the set of slopes for every chip $a_i \in A$ in the experiment, $SL = \{sl_1, sl_2, ..., sl_n\}$
2. Build a boxplot using the set of slopes SL.
3. Identify those arrays $a_i \in A$ identified as outliers, that is, if $sl_i > Q3 + 1.5 \cdot IQR, \forall sl_i \in SL$, where $Q3$ is the upper quartile of the boxplot and IQR is the inter-quartile range $IQR = Q3 - Q1$, the array $a_i \in A$ is considered as defective. Higher slope is indicative of larger RNA degradation [Gentleman et al., 2005].

7.2.1.4 Probe-level models

The Bioconductor R package *affyPLM* [Bolstad, 2011a] provides functions to build Probe-level models from probe-intensity data. As described in subsection 6.2.1.4, three useful quality assessment tools can be obtained from the output of the PLM fitting procedure (eq.6.2.1.4): Chip pseudo-images, Relative Log Expression (RLE) and Normalized Unscaled Standard Error plot (NUSE), which are used by A^2**TOOL**.

Chip pseudo-images For each array of a data set $a_i \in A$, A^2**TOOL** builds the following plots for each array of the experiment: Gray scale image of the log intensities and PLM weights, PLM residual and PLM signed residual plots. This quality metric needs the inspection of the user, who has to look the plots for any anomalies, artifacts or non homogeneous patterns in them, for example, small blemishes, rings or circles [Gentleman et al., 2005].

Relative Log Expression (RLE) plot A^2**TOOL** plots the RLE figure and assuming that, ideally, the boxes in the RLE plot should have small spread and be centered near $RLE = 0$ (see section 6.2.1.4), it detects automatically the existence of defective arrays according to the following procedure:

1. Get the numerical summaries based on the RLE plot for all the samples in the data set A: $B_{RLE} = \{b_{RLE-1}, b_{RLE-2}, ..., b_{RLE-n}\}$ and $V_{RLE} = \{v_{RLE-1}, v_{RLE-2}, ..., v_{RLE-n}\}$, where $b_{RLE-i} \in B_{RLE}$ and $v_{RLE-i} \in V_{RLE}$ are the median (bias measure) and the IQR (variability measure) respectively of the RLE value for the i-th array.

2. Build a boxplot, $Boxplot_{B_{RLE}}$, through the set of absolute median values $|B_{RLE}|$. Those arrays whose $|b_{RLE-i}| \in |B_{RLE}|$ value is identified as outlier by the boxplot, $Boxplot_{B_{RLE}}$, are considered defective samples. Absolute values are used because it does not matter the sign of the b_{RLE-i} values, that is, b_{RLE-i} must be centered at $RLE = 0$.

3. Build a boxplot, $Boxplot_{V_{RLE}}$, through the set of the variability values V_{RLE}. Our tool points as defective those samples whose $v_{RLE-i} \in V_{RLE}$ value is identified as outlier in the largest observation side of the boxplot, $Boxplot_{V_{RLE}}$, that is, if $v_{RLE-i} > Q3 + 1.5 \cdot IQR, \forall v_{RLE-i} \in V_{RLE}$ where Q3 is the upper quartile of $Boxplot_{V_{RLE}}$ and IQR is the inter-quartile range $IQR = Q3 - Q1$ of $Boxplot_{V_{RLE}}$, then the array $a_i \in A$ is considered as defective. Notice that we are looking for arrays with big spread.

Normalized Unscaled Standard Error (NUSE) plot A^2**TOOL** plots the NUSE figure and assuming that, ideally, the boxes in the NUSE plot should have small spread and be centered near $NUSE = 1$ (see section 6.2.1.4), it detects automatically the existence of defective arrays according to the following procedure:

1. Get the numerical summaries based on the NUSE plot for all samples in the data set: $B_{NUSE} = \{b_{NUSE-1}, b_{NUSE-2}, ..., b_{NUSE-n}\}$ and $V_{NUSE} = \{v_{NUSE-1}$,$v_{NUSE-2}, ..., v_{NUSE-n}\}$ where $b_{NUSE-i} \in B_{NUSE}$ and $v_{NUSE-i} \in V_{NUSE}$ are the median (bias measure) and the IQR (variability measure) respectively of the NUSE values for the i-th array.

2. Build a boxplot, $Boxplot_{B_{NUSE}}$, through the set of absolute median values B_{NUSE}. Notice that for B_{NUSE} we would like to have the values centered at one. Those arrays whose $|b_{NUSE-i}| \in |B_{NUSE}|$ value is identified as outlier by the boxplot ,$Boxplot_{B_{NUSE}}$, are considered defective samples. Although

in [Gentleman et al., 2005] the authors consider lower quality arrays those whose $b_{NUSE-i} \in B_{NUSE}$ values are significantly elevated, we consider also defective arrays those samples whose median value is significantly displaced under the value of $NUSE = 1$, therefore, absolute values are considered.

3. Build a boxplot, $Boxplot_{V_{NUSE}}$, through the set of variability values V_{NUSE}. Our tool points as defective those samples whose $v_{NUSE-i} \in V_{NUSE}$ value is identified as outlier in the largest observation side of the boxplot, $Boxplot_{V_{NUSE}}$, that is, if $v_{NUSE-i} > Q3 + 1.5 \cdot IQR$, $v_{NUSE-i} \in V_{NUSE}$ where Q3 is the upper quartile of $Boxplot_{V_{NUSE}}$ and IQR is the inter-quartile range $IQR = Q3 - Q1$ of $Boxplot_{V_{NUSE}}$, then the array $a_i \in A$ is considered as defective. Notice that, again, we are looking for arrays with big spread.

At the end of this stage, A^2**TOOL** generates a report in PDF format, from which the researcher can check the results of applying all quality metrics described in this section to the experiment being analyzed. Besides the results given for each quality metric, a summary is presented where, for each array in the experiment $a_i \in A$, it is informed whether the array has been detected as defective or not in each quality metric.

But, what should the researcher do if abnormalities are found through the quality metrics? The researcher has the final decision, but we recommend to remove a chip from the analysis if it was detected as defective in two or more quality metrics.

7.2.2 A^2TOOL: Data pre-processing

The second step carried out by A^2**TOOL** is to pre-process the whole raw data set, or a subset of it if some arrays have been removed due to their low quality, using all possible and allowed combinations of Background correction, Normalization, PM correction and Summarization methods described in section 6.2.2. From the set of all possible pre-processing methods, P, the goal is to select the best of them for a given raw data set according to the quality metrics described in [Florido et al.,

2009c]: *replicate variability*, *Kolmogorov-Smirnov test* and *Spearman Rank Correlation Coefficient*, that is, a subset $P' \subseteq P$ of preprocessing methods is obtained.

Since A^2**TOOL** works on Affymetrix Genechips, pre-processing is split into four steps: background correction (subsection 6.2.2.1), normalization (subsection 6.2.2.2), PM correction (subsection 6.2.2.3) and summarization (subsection 6.2.2.4) [Lim et al., 2007],[Florido et al., 2010b],[Florido et al., 2009c].

For each step, our tool is able to execute the following procedures:

1. **Background correction**: *mas* [Affymetrix, 2001], (rma) [Irizarry et al., 2003], and *gcrma* [Wu et al., 2004].
2. **Normalization**: *scaling* (constant) [Affymetrix, 2001], *quantile* [Bolstad et al., 2003], *cyclic loess* [Bolstad et al., 2003], *invariant set* [Schadt et al., 2001],[Wong, 2011] and *variance stabilization method* (VSN) [Huber et al., 2002].
3. **PM-MM correction**: *MAS* [Affymetrix, 2001] and *PMonly* [Gautier et al., 2004].
4. **Summarization**: *Tukey-biweight* (mas) [Affymetrix, 2001], *MBEI* [Li and Wong, 2001],[Wong, 2011], *Median polish* [Irizarry et al., 2003], *Avgdiff* [Affymetrix, 1999], *Factor Analysis for Robust Microarray Summarization* (FARMS) [Hochreiter et al., 2006] and *Distribution Free Weighted* (DFW) [Chen et al., 2007b].

A^2**TOOL** will execute all possible and allowed combinations of Background correction, Normalization, PM correction and Summarization methods using the Bioconductor [Gentleman et al., 2004] R packages available. Notice that, as stated in section 6.2.2, not all possible combinations are allowed. In Appendix B, table B.1 shows the pre-processing methods used by A^2**TOOL**.

7.2.2.1 On selecting the best pre-processing methods through A^2**TOOL**

As mentioned at the beginning of section 7.2.2, A^2**TOOL** executes all pre-processing methods contained in table B.1 for a given raw data set obtaining a set of pre-processed data (expression set). Since the expression data may have an effect in posterior stages of microarray data analysis pipeline (see section 6.2.3), each expression set must be analyzed in terms of the metrics proposed in subsection 6.2.2.5: Replicate variability, K-S test and Spearman Rank Correlation Coefficient to evaluate its quality.

As stated in section 6.2.2.5, it is important to check whether a pre-processing method introduces correlation artifacts in the pre-processed data set. For this purpose, A^2**TOOL** proceeds the same as in [Lim et al., 2007]: each pre-processing procedure is run not only on the original raw data, but also on uninformative raw data, that is, raw signal intensities for each probe pairs in the data set experiment were randomly permuted to create uninformative CEL files. The relative position between PM and MM for every probe pairs are retained, in order to ensure fair comparison between normalization procedures that utilize MM information to correct for non-specific binding and those that rely entirely on PM intensities. Shuffling the probe pairs has been sufficient to destroy real signal of the probe sets as they now consist of random probes values. Then, the quality metrics (Replicate variability, K-S test and Spearman Rank Correlation Coefficient) are also applied to the pre-processed uninformative data set.

Therefore, for each pre-processed informative (original) and uninformative data set, A^2**TOOL** computes three different quality metrics according to [Florido et al., 2009c]:

- Replicate variability. It is based on the assumption that expression level of a gene should ideally remain the same across multiple replicated slides. Variability is measured by the mean of the standard deviation over all genes. Smaller mean is indicative of better pre-processing [Xiong et al., 2008]. Thus, if the experiment data set has k group of replicates, that is $G = \{group_1, group_2, ..., group_k\}$, then for each $group_j \in G$:

1. Get the set of arrays belonging to $group_j$, that is, $A_j = \{a_1, a_2, ..., a_m\} \in group_j$.
2. Compute the standard deviation for gene $\hat{\sigma_g}$ across $A_j \in group_j$:

$$\hat{\sigma_g} = \sqrt{\frac{1}{m-1}\sum_{i=1}^{m}(expr_{gi} - \mu_j)^2} \tag{7.2}$$

where m is the number of arrays in $group_j$, $expr_{gi}$ is the value for gene g in slide $a_i \in A_j$, while μ_j is the average expression value over A_j for gene g. A smaller $\hat{\sigma_g}$ is indicative of a more effective normalization procedure. The mean of such estimates over all genes for a given group of replicates $group_j$, $\hat{\mu_j}$, is obtained.

Thus, $G_{sd} = \{\hat{\mu_1}, \hat{\mu_2}, ..., \hat{\mu_k}\}$ is the set of means of each group of replicates and $\mu_{G_{sd}} = \frac{1}{k}\sum_{j=1}^{k}\hat{\mu_j}$ is a global measure of the performance of the pre-processing method, with smaller mean indicative of better performance.

- Kolmogorov-Smirnov (K-S) test. It is based on the hypothesis that an effective pre-processing method should result in two similar, ideally identical, distributions with a small, ideally zero-valued K-S statistic [Xiong et al., 2008]. On the other hand, two different distributions will generate a large K-S statistic. If the experiment data set has k group of replicates, that is $G = \{group_1, group_2, ..., group_k\}$, then for each $group_j \in G$:

1. Get the set of arrays belonging to $group_j$, that is, $A_j = \{a_1, a_2, ..., a_m\} \in group_j$.
2. Compute KS-test on every pairwise of arrays $KS(a_p, a_q), a_p$ and $a_q \in A_j$ and $p \neq q$. A total of $\frac{m(m-1)}{2}$ tests are run for A_j.
3. The mean of such tests, $\hat{\mu_j}$, for the given group of replicates A_j is computed

Thus, $G_{KS} = \{\hat{\mu_1}, \hat{\mu_2}, ..., \hat{\mu_k}\}$ is the set of means of each group of replicates and $\mu_{G_{KS}} = \frac{1}{k}\sum_{j=1}^{k}\hat{\mu_j}$ is a global measure of the performance of the pre-processing method, with smaller mean indicative of better performance.

- Spearman Rank Correlation Coefficient. It is based on the comparison of the correlation coefficient between replicated slides assuming that, given an experiment, the correlation coefficient among replicated slides will be increased after the pre-processing stage [Lim et al., 2007]. If the experiment data set has k group of replicates, that is $G = \{group_1, group_2, ..., group_k\}$, then for each $group_j \in G$:

  1. Get the set of arrays belonging to $group_j$, that is, $A_j = \{a_1, a_2, ..., a_m\} \in group_j$.
  2. Compute Spearman Rank Correlation Coefficient on every pairwise of arrays $corr(a_p, a_q), a_p$ and $a_q \in A_j$ and $p \neq q$. A total of $\frac{m(m-1)}{2}$ tests are run for A_j.
  3. The mean of such tests, $\hat{\mu_j}$, for the given group of replicates A_j is computed

  Thus, $G_{corr} = \{\hat{\mu_1}, \hat{\mu_2}, ..., \hat{\mu_k}\}$ is the set of means of each group of replicates and $\mu_{G_{corr}} = \frac{1}{k}\sum_{j=1}^{k}\hat{\mu_j}$ is a global measure of the performance of the pre-processing method, with bigger mean indicative of better performance.

To select the best pre-processing methods for a given raw data set *rawDataO*, A^2**TOOL** the procedure proposed by J.P.Florido et al. [Florido et al., 2010a] is adopted (see Fig.7.2):

1. Set the best pre-processing methods set to empty: $P' = \emptyset$
2. Get pre-processed raw expression data using *RAW* preprocessing method[1]:

   esetRawO=RAW(rawDataO)
   esetRawC=RAW(rawDataC)

   where *rawDataO* is the original raw data set and *rawDataC* is the control raw data set (uninformative raw data set).

[1]RAW pre-processing method (first entry in table B.1) corresponds to raw expression data (which is different from raw data): a summarization method is needed because in Affymetrix Genechips, each gene is represented by a set of several PM and MM probe pairs and the probe intensities for each probe set (gene) should be summarized to define a measure of expression value. *Medianpolish* has been chosen as summarization method for raw expression data due to its outperforming to other summarization methodologies [Florido et al., 2009c]

3. Compute quality metrics on the expression sets obtained:

   - Replicate variability: $\mu_{G_{sd}}(esetRawC)$ and $\mu_{G_{sd}}(esetRawO)$
   - KS-test: $\mu_{G_{KS}}(esetRawC)$ and $\mu_{G_{KS}}(esetRawO)$
   - Spearman Rank Correlation Coefficient: $\mu_{G_{corr}}(esetRawC)$ and $\mu_{G_{corr}}(esetRawO)$

4. For each pre-processing method $p_i \in P$, with $p_i \neq RAW$:

   (a) Get pre-processed expression data using p_i preprocessing method:

   $$esetO_i = p_i(rawDataO)$$
   $$esetC_i = p_i(rawDataC)$$

   where *rawDataO* is the original raw data set and *rawDataC* is the control raw data set.

   (b) Compute quality metrics on the expression sets obtained:

   - Replicate variability: $\mu_{G_{sd}}(esetC_i)$ and $\mu_{G_{sd}}(esetO_i)$
   - KS-test: $\mu_{G_{KS}}(esetC_i)$ and $\mu_{G_{KS}}(esetO_i)$
   - Spearman Rank Correlation Coefficient: $\mu_{G_{corr}}(esetC_i)$ and $\mu_{G_{corr}}(esetO_i)$

   (c) Check whether the pre-processed expression data given by p_i improves quality metrics with respect to raw expression data. If so, the method p_i is selected if it does not introduce correlation artifacts or, at most, introduces a number of correlation artifacts which is below a threshold, that is:

   if $(\mu_{G_{sd}}(esetO_i) < \mu_{G_{sd}}(esetRawO)$ *&&* $\mu_{G_{KS}}(esetO_i) < \mu_{G_{KS}}(esetRawO)$
   && $\mu_{G_{corr}}(esetO_i) > \mu_{G_{corr}}(esetRawO))$
   { # the pre-processed expression data improves raw data (esetRawO)
   if $(\mu_{G_{sd}}(esetC_i) > \mu_{G_{sd}}(esetRawC)$ *&&*
   $\mu_{G_{corr}}(esetC_i) < \mu_{G_{corr}}(esetRawC))$
   $P' = \{P' \cup p_i\}$ # p_i does not introduce correlation artifacts
   else
   {
   $\mu_{G_{sd}}(O) = \mu_{G_{sd}}(esetO_i) - \mu_{G_{sd}}(esetRawO)$
   $\mu_{G_{sd}}(C) = \mu_{G_{sd}}(esetC_i) - \mu_{G_{sd}}(esetRawC)$
   $\mu_{G_{corr}}(O) = \mu_{G_{corr}}(esetO_i) - \mu_{G_{corr}}(esetRawO)$

$\mu_{G_{corr}}(C) = \mu_{G_{corr}}(esetC_i) - \mu_{G_{corr}}(esetRawC)$
if($\frac{\mu_{G_{sd}}(C)}{\mu_{G_{sd}}(O)} < threshold$ *&&* $\frac{\mu_{G_{corr}}(C)}{\mu_{G_{corr}}(O)} < threshold$)
$P' = \{P' \cup p_i\}$# p_i introduces correlation artifacts
#below a threshold
}
}

5. Once the set of best pre-processing methods have been selected in P', sort this list according to the values of the quality metrics in the following way:

   - Sort P' in ascending order according to the variability value $\mu_{G_{sd}}(esetO_i)$, $\forall p_i \in P'$, obtaining P'_{sd}. Assign 1 to $P'_{sd}(1)$, 2 to $P'_{sd}(2)$ and so on.
   - Sort P' in ascending order according to the KS-statistic value $\mu_{G_{KS}}(esetO_i)$, $\forall p_i \in P'$, obtaining P'_{KS}. Assign 1 to $P'_{KS}(1)$, 2 to $P'_{KS}(2)$ and so on.
   - Sort P' in descending order according to the Spearman coefficient value $\mu_{G_{corr}}(esetO_i)$, $\forall p_i \in P'$, obtaining P'_{corr}. Assign 1 to $P'_{corr}(1)$, 2 to $P'_{corr}(2)$ and so on.

   Sort P' in ascending order according to $P'_{sd} + P'_{corr} + P'_{KS}$ scores. Thus, the pre-processing methods selected are sorted according to its quality: the lower the total score value, the higher the quality of a pre-processing method.

Notice that the use of the uninformative raw data set ($esetC_i$) is very important to detect whether a pre-processing method has introduced correlation artifacts. Its used is based on the following hypothesis: the replicate variability and Spearman Correlation Coefficient are expected to have low quality values in uninformative raw expression data (*esetRawC*), that is, high replicate variability and low Spearman Correlation. For a pre-processed uninformative raw data ($esetC_i$), it is expected to have even worst values for these quality metrics than the ones related to the uninformative raw expression data (*esetRawC*), because uninformative raw data has no sense and has destroyed the original information of microarray (second *if* of step 4.c of the procedure). However, one can also expect an improvement and, if so, it must be measured whether this improvement is significant or not (else in

step 4.c). If it is significant, A^2**TOOL** considers that the pre-processing method has introduced many correlation artifacts. If not, the method would be accepted (last *if*). This procedure is outlined as follows:

1. Quantify, through the quality metrics, the improvement in the pre-processed data set with respect to the raw expression data set for both informative (original) and uninformative data sets: $\mu_{G_{sd}}(O), \mu_{G_{sd}}(C)$,$\mu_{G_{corr}}(O), \mu_{G_{corr}}(C)$ (see *else* of step 4.c).

2. If for both quality metrics the improvement in the uninformative data with respect to the improvement in the original data is below a threshold, the method is accepted. Otherwise, the method is rejected, i.e., the pre-processing method introduces several correlation artifacts.

Notice that when quantifying the number of correlation artifacts introduced, the K-S statistic is not taken into account. This is because K-S statistic is a measure based on the comparison of two empirical distribution functions and we consider that there are not many differences between distributions based on informative data and distributions based on uninformative data.

In Fig.7.2, steps from 1 to 4 are schematized.

7.2.3 A^2TOOL: Detection of differentially expressed genes

The third and the last step carried out by A2TOOL is to detect differentially expressed genes using each pre-processed data (expression data) given by all pre-processing methods contained in P'. Moreover, our tool also builds a list of intersection and a list of union genes detected in all pre-processed data sets with the aim of providing confident and complete lists of differentially expressed genes. It is possible to build such lists only when more than a pre-processing method has been selected.

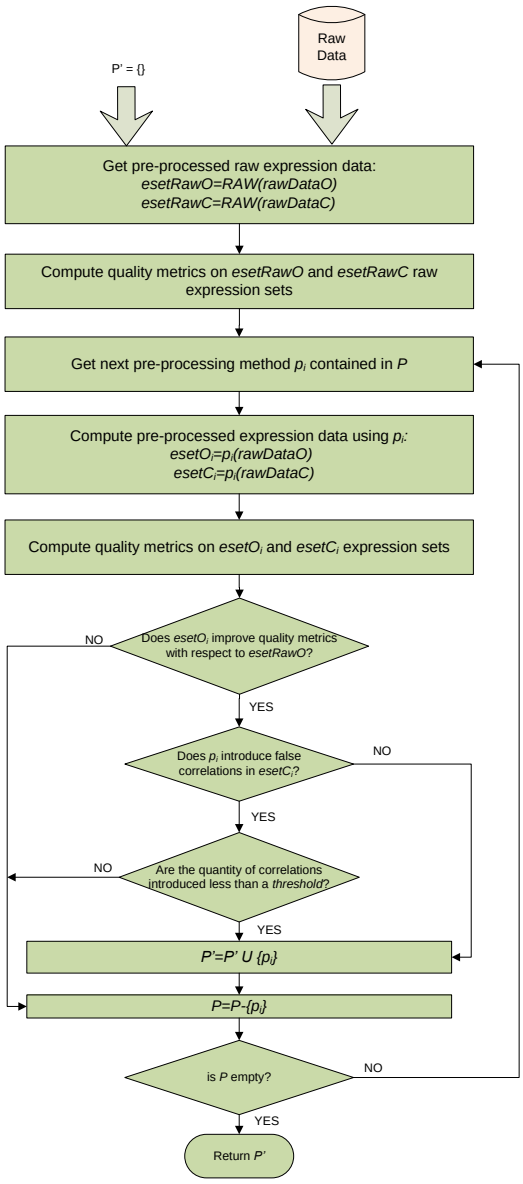


FIGURE 7.2: On selecting the best pre-processing methods for a given raw data set

7.2.3.1 Differential expression

For each pre-processing method $p_i \in P'$, the following two-step procedure to obtain differentially expressed genes is applied by A^2**TOOL** to the expression set obtained when pre-processing with p_i, that is, $esetO_i \leftarrow p_i(rawData)$ where *rawData* is the original raw data set.

- Apply nonspecific filtering to $esetO_i$ using the *geneFilter* package [Gentleman et al., 2011], obtaining a reduced expression set $esetO'_i$.
- Apply a moderated *t*-test analysis to $esetO'_i$ through the *limma* package [Smyth, 2004].

Nonspecific filtering To alleviate the loss of power from the formidable multiplicity of gene-by-gene hypothesis testing, it is important to carry out nonspecific filtering (see subsection 6.2.3). By *nonspecific* we mean that it is done without reference to the parameters or conditions of the tested RNA samples. Its aim is to remove from consideration that set of probes whose genes are not differentially expressed under any comparison [Gentleman et al., 2011].

For this purpose, A^2**TOOL** uses *genefilter* package and a gene is removed by A^2TOOL from the experiment if:

- it is an Affymetrix control gene (starts with AFFX)
- it has a variability across samples below a predefined user threshold. The IQR per-feature filtering statistic is used and the default value for this filter is 0.5, which is a strong filter, i.e. the majority of the unchanged probe sets are removed (the top 50% of genes are selected on the basis of variability).
- it has the same EntrezID as other gene. Is this case, the gene with the greatest variability is retained.

A reduced expression set $esetO'_i$ is obtained once this filter is applied.

Statistics for Differential Expression The *moderated t-statistic* is computed for each gene and for each contrast through the *limma* package [Smyth, 2004] (see section 6.2.3).

A^2**TOOL** applies *limma* to the reduced expression set, $esetO'_i$, and each gene of this expression set is ranked according to the *adjusted p*-value. Only those genes whose $logFC$ and *adjusted p*-values are above and below a threshold respectively, are listed as differentially expressed.

For each gene listed as differentially expressed, A^2**TOOL** shows the following statistics:

- ID: Affymetrix gene ID.
- $EntrezID$.
- $logFC$: log_2-fold change for the contrast being processed (the default threshold for the test is 0.585, which corresponds to a fold-change of 1.5)
- $AveExpr$: average log_2-expression level for that gene across all the arrays and channels in the experiment.
- t: moderated t-statistic.
- adj.P.Val: p-value adjusted for multiple testing. It is the probability that an expression change is due to chance. The default value for the test is $p_{threshold} = 0.05$. A^2**TOOL** uses two different correction methods: Benjamini and Hochberg False Discovery Rate (FDR) correction and Bonferroni Step-down (Holm) correction. Thus, for a given gene, its *adjusted p*-value ($adj.P.val$) corresponding to the FDR correction is shown and if $adj.P.val$ of *Bonferroni* correction is called significant, the gene is also marked.
- B: it is the log-odds that the gene is differentially expressed. For example, a B-statistic of zero corresponds to a 50-50 chance that the gene is differentially expressed. The bigger this value, the bigger the probability of being differentially expressed.

It must be noticed, that a list of differentially expressed genes is shown for each contrast or each pair of conditions in the experiment. For example, if an experiment has three conditions: *Control, Pre* and *Pos*, three lists will be shown: *Control vs Pre*, *Control vs Pos* and *Pre vs Pos*, corresponding to three contrasts.

Thus, for a given reduced pre-processed data $esetO'_i$ and a set of conditions in the experiment $C = \{condition_1, condition_2, ..., condition_k\}$, a set of differentially expressed genes $Genes_i = \{\{genes_{i_1}\}, \{genes_{i_2}\}, ..., \{genes_{i_{numContrasts}}\}\}$ are obtained, where $numContrasts = \frac{k(k-1)}{2}$ and $\{genes_{i_j}\}$ represents the set of differentially expressed genes detected in the reduced expression set $esetO'_i$ related to the pre-processing method p_i and a contrast $j \in \{j = 1, ..., numContrasts\}$.

7.2.3.2 Merging results: intersection and union lists

A^2**TOOL** has computed the list of differentially expressed genes detected in each reduced expression set $esetO'_i, \forall p_i \in P'$. If the number of pre-processing methods in P' is low, say, 4, one can have a look to each list of genes to have an idea about the genes that have been selected, but, if the length of P' is large and our experiment has more than two conditions, to get conclusions from the results can be cumbersome and difficult. If P' has 15 methods and our experiment has 3 conditions, up to 45 lists of differentially expressed genes can be obtained, which is not useful.

To overcome this problem and if the number of methods in P' is greater than 1, for each contrast $j \in \{1, ..., numContrasts\}$, A^2TOOL generates two lists:

- The intersection list. It is obtained through the intersection of differentially expressed genes detected in all reduced expression sets $esetO'_i, \forall p_i \in P'$. For an initial value of $I_{genes_j} = \emptyset$, this list is obtained as:

  $$I_{genes_j} = I_{genes_j} \cap \{genes_{ij}\} \text{ for } i = \{1, ..., |P'|\}$$

  where $|P'|$ is the number of pre-processing methods (or the number of pre-processed data) in P'. Through this list, it is expected to obtain the most reliable genes as differentially expressed, since they are expressed regardless of the pre-processing method used.

- The union list. It is obtained through the union of differentially expressed genes detected in all reduced expression sets $esetO'_i, \forall p_i \in P'$. For an initial value of $U_{genes_j} = \emptyset$ this list is obtained as follows:

$$U_{genes_j} = U_{genes_j} \cup \{genes_{ij}\} \text{ for } i = \{1, ..., |P'|\}$$

  where $|P'|$ is the number of pre-processing methods in P'. In this case, it is expected to obtain a long list of genes. This list may contain differentially expressed genes detected only in an expression set, but also may contain genes that have been detected in all expression sets but one. Thus, it is an interesting and complete list of candidate genes that are likely to be differentially expressed.

7.3 Experimental results

In this section the results of applying A^2**TOOL** to the Chronic Lymphocytic Leukemia (CLL) data set are presented. Since different pre-processing methods can lead to different expression data and, therefore, may have an effect in posterior stages such as phenotype classification [Florido et al., 2010b],[Wu et al., 2005] the results of studying the effect of the pre-processing methods commonly used in the literature on microarray-based cancer classifiers are also presented in section 7.3.2.

7.3.1 Results given by A^2TOOL on CLL data set

In this section, A^2**TOOL** is applied to the Chronic Lymphocytic Leukemia (CLL) data set [Whalen and Gentleman, 2011]. This data set has 23 samples that were either classified as progressive or stable in regards to disease progression. Thus, there are two conditions or group of replicates. Some useful information about this data set is given below:

- Size of arrays: 640 x 640
- Number of genes: 12625

- Number of samples: 23
- Array platform: hgu95av2

The experimental design of CLL experiment is shown in table 7.1.

TABLE 7.1: Experimental design of CLL experiment

id	FileName	Condition
1	CLL11.CEL	progressive
2	CLL12.CEL	stable
3	CLL13.CEL	progressive
4	CLL14.CEL	progressive
5	CLL15.CEL	progressive
6	CLL16.CEL	progressive
7	CLL17.CEL	stable
8	CLL18.CEL	stable
9	CLL19.CEL	progressive
10	CLL1.CEL	stable
11	CLL20.CEL	stable
12	CLL21.CEL	progressive
13	CLL22.CEL	stable
14	CLL23.CEL	progressive
15	CLL24.CEL	stable
16	CLL2.CEL	stable
17	CLL3.CEL	progressive
18	CLL4.CEL	progressive
19	CLL5.CEL	progressive
20	CLL6.CEL	progressive
21	CLL7.CEL	progressive
22	CLL8.CEL	progressive
23	CLL9.CEL	stable

According to the guidelines given in section 7.2, A^2**TOOL** will analyze this data set in terms of (i) its quality (section 7.2.1), (ii) data pre-processing (section 7.2.2) and (iii) the detection of differentially expressed genes (section 7.2.3).

7.3.1.1 Quality data analysis

This is the first step carried out by A^2**TOOL** and consists in analyzing the quality of the experiment data set $A = \{a_1, a_2, ..., a_{23}\} = \{CLL1.CEL, CLL2.CEL, ..., CLL24.CEL\}$.

Image plots For the CLL experiment data set, images of probe-level intensities using intensities in logarithmic scale are created by A^2**TOOL**. For brevity purposes, only image plots of 6 arrays are shown (Fig. 7.3). The remaining image plots are quite similar and there are no clear spatial artifacts or other non-homogeneous patterns on these plots.

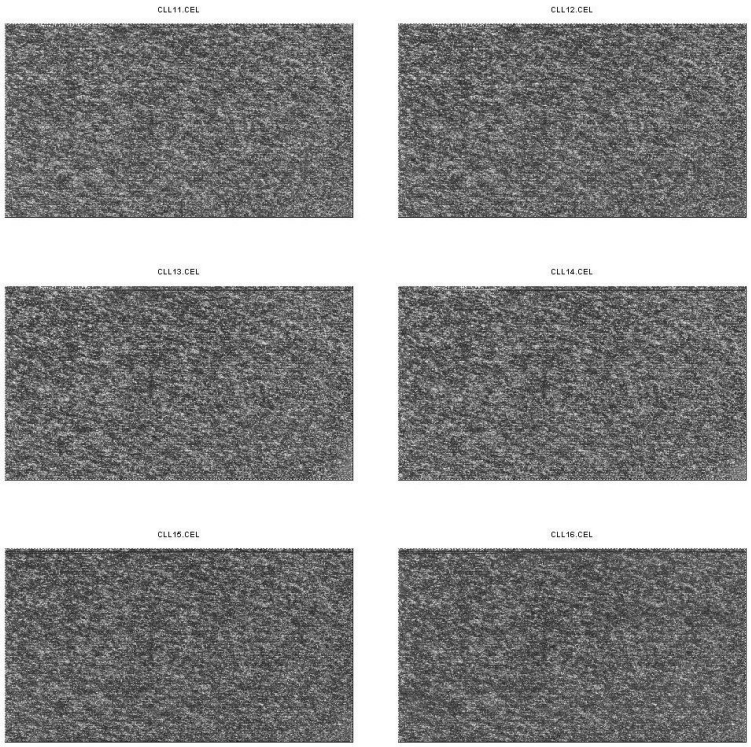


FIGURE 7.3: Image of probe-level intensities using logarithmically transformed intensities for the CLL experiment. A subset of CEL files is shown

Multi-array approaches *Boxplots*, *histograms* and *MAplots* approaches are shown to look at the distribution of probe intensities across all arrays of the CLL experiment.

1. *Boxplots*. In Fig. 7.4, boxplots of CLL data set are shown. Red color corresponds to the *progressive* condition and green color to the *stable* condition. It can be observed that there are no boxplots that stand out from the others for a given group of replicates.

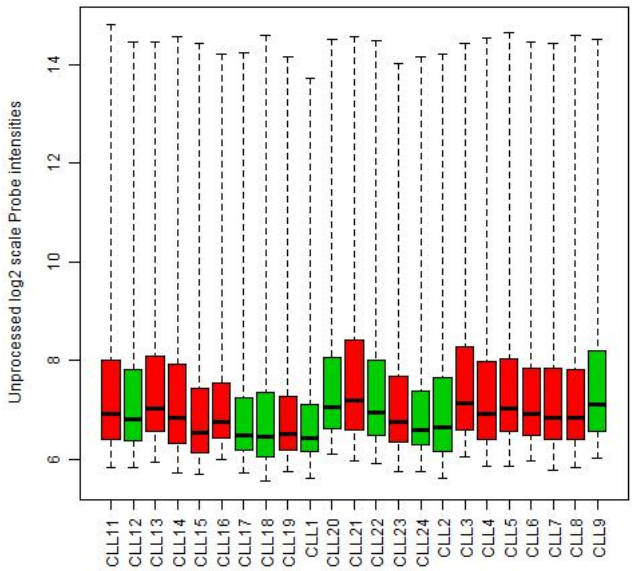


FIGURE 7.4: Boxplots of unprocessed log scale probe intensities for the CLL experiment. Replicates have the same color

In table 7.2, the lower quartile (Q1), the upper quartile (Q3) and the mean value of the lower and upper quartile for each group of replicates, μ_{Q1} and μ_{Q3}, are shown.

According to the guidelines given in section 7.2.1.1, for a given group of replicates (progressive or stable), $Q1_i < \mu_{Q3}$ and $Q3_i > \mu_{Q1}, \forall i$. Therefore, there are no defective arrays according to the boxplot quality metric.

2. *Histograms*. In fig.7.5 the smooth histograms for the CLL experiment are shown.

   A^2**TOOL** uses the bimodal index to check whether a set of data fits a two-component mixture model. If the bimodal index $bi > 1.1$ for a given array, it would be considered as defective. In table 7.3, the bimodal index of each sample is shown and, as can be observed, none of the arrays has a bimodal index greater than 1.1. However, the bi-modal index of Class Discovery package, detects only big bi-modalities. So, visual inspection on Fig. 7.5 is also needed to detect small bimodalities. It can be observed that none of the arrays has a little bimodality or a different shape.

TABLE 7.2: Lower and upper quartiles for each sample and mean lower and upper quartiles for each condition of the CLL experiment

id	Sample Name	Condition	$Q1_i$	$Q3_i$	μ_{Q1}	μ_{Q3}
1	CLL11.CEL	progressive	6.409	8.015		
3	CLL13.CEL	progressive	6.559	8.079		
4	CLL14.CEL	progressive	6.321	7.936		
5	CLL15.CEL	progressive	6.139	7.442		
6	CLL16.CEL	progressive	6.426	7.546		
9	CLL19.CEL	progressive	6.189	7.268		
12	CLL21.CEL	progressive	6.599	8.413	6.416	7.874
14	CLL23.CEL	progressive	6.357	7.679		
17	CLL3.CEL	progressive	6.584	8.276		
18	CLL4.CEL	progressive	6.409	7.995		
19	CLL5.CEL	progressive	6.554	8.049		
20	CLL6.CEL	progressive	6.475	7.857		
21	CLL7.CEL	progressive	6.409	7.857		
22	CLL8.CEL	progressive	6.392	7.826		
2	CLL12.CEL	stable	6.375	7.820		
7	CLL17.CEL	stable	6.185	7.242		
8	CLL18.CEL	stable	6.044	7.367		
10	CLL1.CEL	stable	6.149	7.114		
11	CLL20.CEL	stable	6.622	8.060	6.320	7.651
13	CLL22.CEL	stable	6.472	8.010		
15	CLL24.CEL	stable	6.285	7.390		
16	CLL2.CEL	stable	6.169	7.658		
23	CLL9.CEL	stable	6.577	8.194		

3. *MA plots*. The MA plot is used for each condition or treatment (progressive or stable), in which each array a belonging to that condition is compared to a *synthetic* array built from the samples that belong to the same condition of a. In Fig. 7.6, the MA plots for the chips of progressive level (CLL11.CEL, CLL13.CEL and CLL14.CEL) versus the synthetic (median) array of the same level are shown. For brevity purposes, only the MAplots for these chips are provided. Neither these MAplots nor the rest of them, present quality problems that occur when the loess smoother oscillates wildly or the variability of the M values appears to be greater in one or more arrays relative to the others.

   Remember from section 7.2.1.1, that, for a given group of replicates, a boxplot is built from the mean of the absolute values of m_i, μ_{m_i}. The outliers detected in that boxplot are considered as defective by A^2**TOOL**. As can be observed from table 7.4 and, for each group of replicates or condition,

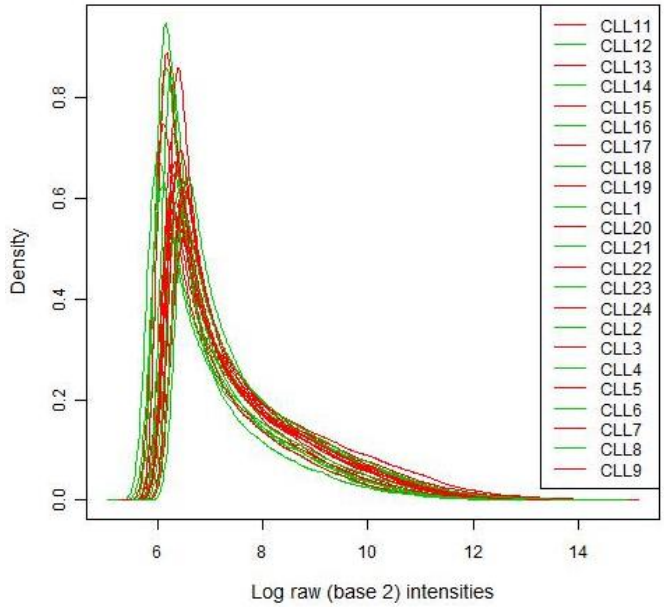


FIGURE 7.5: Smoothed histograms for the CLL experiment. Replicates have the same color

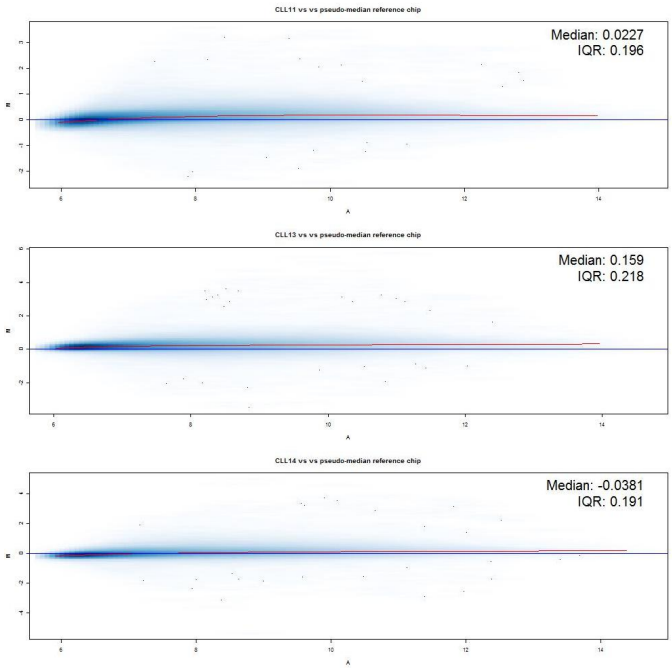


FIGURE 7.6: MA plot for the chips of progressive level (CLL11.CEL, CLL13.CEL and CLL14.CEL) versus the synthetic (median) array of the same level. A loess regression line is added to the plot

TABLE 7.3: Bi-modal indexes for the histograms in the CLL experiment

Sample	Bi-modal index (bi_i)
CLL11.CEL	0.066
CLL12.CEL	0.064
CLL13.CEL	0.069
CLL14.CEL	0.066
CLL15.CEL	0.051
CLL16.CEL	0.055
CLL17.CEL	0.049
CLL18.CEL	0.056
CLL19.CEL	0.053
CLL1.CEL	0.051
CLL20.CEL	0.065
CLL21.CEL	0.073
CLL22.CEL	0.067
CLL23.CEL	0.067
CLL24.CEL	0.052
CLL2.CEL	0.062
CLL3.CEL	0.068
CLL4.CEL	0.067
CLL5.CEL	0.066
CLL6.CEL	0.063
CLL7.CEL	0.064
CLL8.CEL	0.061
CLL9.CEL	0.066

$\mu m_i > Q1 - 1.5 \cdot IQR$ and $\mu m_i < Q3 + 1.5 \cdot IQR, \forall i$, that is, there are no defective arrays.

Affymetrix quality assessment metrics

- *The average background.* According to the guidelines given in section 7.2.1.2, the average background values must fall within two standard deviation of the mean. Table 7.5 shows the average background values for the CLL experiment. As can be observed, CLL18.CEL is detected by A^2**TOOL** as defective array, since $ab_{CLL18.CEL} < \mu_{AB} - 2 \cdot \sigma_{AB}$ where μ_{AB} and σ_{AB} are the mean and standard deviation respectively of $AB = \{ab_1, \ldots, ab_n\}$. In this case, $\mu_{AB} - 2 \cdot \sigma_{AB} = 64.83 - 2 \cdot 6.62 = 51.59$.

TABLE 7.4: Statistics for the MA plots for two group of replicates: progressive and stable (CLL data set)

id	Sample Name	Condition	μ_{m_i}	$Q1 - 1.5 \cdot IQR$	$Q3 + 1.5 \cdot IQR$
1	CLL11.CEL	progressive	0.139		
3	CLL13.CEL	progressive	0.207		
4	CLL14.CEL	progressive	0.143		
5	CLL15.CEL	progressive	0.362		
6	CLL16.CEL	progressive	0.226		
9	CLL19.CEL	progressive	0.416		
12	CLL21.CEL	progressive	0.380	-0.083	0.519
14	CLL23.CEL	progressive	0.215		
17	CLL3.CEL	progressive	0.293		
18	CLL4.CEL	progressive	0.134		
19	CLL5.CEL	progressive	0.189		
20	CLL6.CEL	progressive	0.150		
21	CLL7.CEL	progressive	0.142		
22	CLL8.CEL	progressive	0.133		
2	CLL12.CEL	stable	0.145		
7	CLL17.CEL	stable	0.296		
8	CLL18.CEL	stable	0.306		
10	CLL1.CEL	stable	0.395		
11	CLL20.CEL	stable	0.352	-0.033	0.583
13	CLL22.CEL	stable	0.261		
15	CLL24.CEL	stable	0.197		
16	CLL2.CEL	stable	0.179		
23	CLL9.CEL	stable	0.397		

- *Scale factors*. According to section 7.2.1.2, it is recommended that the scale factors be within 3-fold of each other (see section 7.2.1.2). As can be observed from Table 7.5, CLL1.CEL is detected as defective array since $log_2(sc_{CLL1.CEL}) = 2.398 > U_{sc}$, where μ_{SC} os the mean of all scale factor values. In this case, $U_{sc} = (3/2 + \mu_{SC}) = (3/2 + 0.81) = 2.31$.

- *Percent present*. According to section 7.2.1.2, an array $a_i \in A$ is detected as defective by A^2**TOOL** if it has more than 10% of difference with respect to the maximum value of percent present, which is considered as the best quality. From table 7.5, column *percent present*, CLL1.CEL is detected as defective array since $|pp_{CLL1.CEL} - max_{PP}| > 10$, where $PP = \{pp_1, \ldots, pp_n\}$. In this case, $|25.04 - 43.93| = 18.89 > 10$.

TABLE 7.5: Average background values, scale factors and percent present values for the CLL experiment

Chip	Average Background (ab_i)	Scale factor (sc_i)	$log_2(scalefactor)$ ($log_2(sc_i)$)	Percent present (pp_i)
CLL11.CEL	63.26	1.306	0.386	41.30
CLL12.CEL	63.59	1.516	0.601	40.60
CLL13.CEL	72.59	1.250	0.322	41.49
CLL14.CEL	61.70	1.260	0.333	43.93
CLL15.CEL	56.09	2.088	1.062	38.89
CLL16.CEL	69.77	2.550	1.350	34.23
CLL17.CEL	58.85	2.944	1.558	36.51
CLL18.CEL	51.12	2.240	1.164	39.56
CLL19.CEL	59.31	3.192	1.674	34.00
CLL1.CEL	56.11	5.271	2.398	25.04
CLL20.CEL	75.41	1.422	0.508	39.03
CLL21.CEL	71.42	0.845	-0.244	43.35
CLL22.CEL	68.19	1.236	0.306	42.40
CLL23.CEL	64.87	1.988	0.991	39.32
CLL24.CEL	62.13	2.869	1.520	34.39
CLL2.CEL	54.59	1.839	0.879	39.59
CLL3.CEL	72.55	1.217	0.283	38.24
CLL4.CEL	64.88	1.302	0.381	41.99
CLL5.CEL	71.80	1.590	0.669	37.31
CLL6.CEL	69.82	1.881	0.911	36.65
CLL7.CEL	65.99	1.456	0.542	40.93
CLL8.CEL	65.51	1.489	0.575	41.28
CLL9.CEL	71.59	1.275	0.350	40.57

RNA degradation As stated in section 7.2.1.3, A^2**TOOL** adopted the difference in slopes among chips to detect RNA degradation. At can be observed in Fig.7.7, there is an array with a very different slope to the rest of samples and its probe intensities are strongly elevated at the 3'end of a probe set when compared to the 5'end.

Table 7.6 shows the slope values for RNA degradation for the CLL experiment. As can be observed, CLL1.CEL is detected as defective by A^2**TOOL**, since $sl_{CLL1.CEL} > Q3 + 1.5 \cdot IQR$, where Q_3 is the upper quartile of the boxplot built using $SL = \{sl_1, \ldots, sl_n\}$ and IQR os the interquartile range of the boxplot. In this case, $sl_{CLL1.CEL} = 1.728 > 1.335$ and this slope is not similar to others in the experiment, so, this array is detected by A^2**TOOL** as defective.

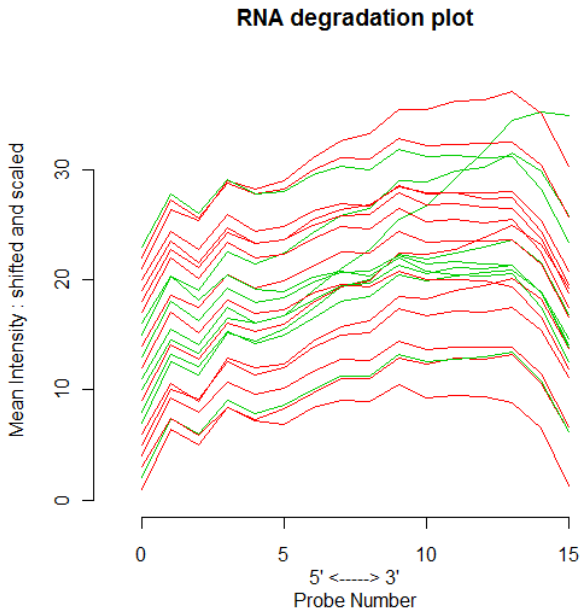


FIGURE 7.7: RNA degradation plot. Each line represents one of the chip from the CLL experiment. Red line corresponds to progressive condition and the green line is related to stable condition

Probe-level models

- *Chip pseudo-images*. Fig. 7.8, shows the gray scale image of the log intensities (the same as the exploratory data analysis) and weights, residual and signed residuals of the fitted PLM object. For brevity purposes, only chip pseudo-images of two arrays are shown: CLL14.CEL and CLL1.CEL. The rest of pseudo-images are very similar to the ones related to CLL14.CEL, with no circles, rings or small blemishes. However, CLL1.CEL shows a weight plot with a color pattern quite different to the rest of arrays. Thus, in this case, it can be considered as defective.

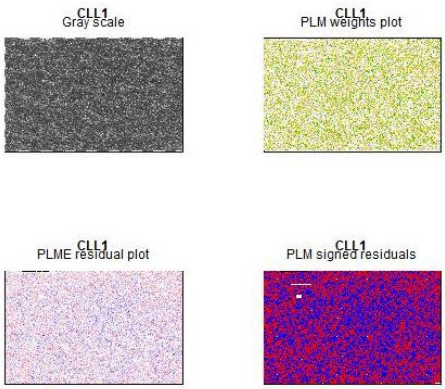
- *Relative Log Expression (RLE) plot*. The RLE plot is shown in Fig. 7.9a) and it can be observed that CLL1.CEL array is not centered at RLE=0 and has greater spread than the other arrays, which indicates that the array has quality problems. According to the guidelines given in section 7.2.1.4, this fact is confirmed in table 7.7 where CLL1.CEL is detected as defective array

TABLE 7.6: Slope values for RNA degradation for the CLL experiment

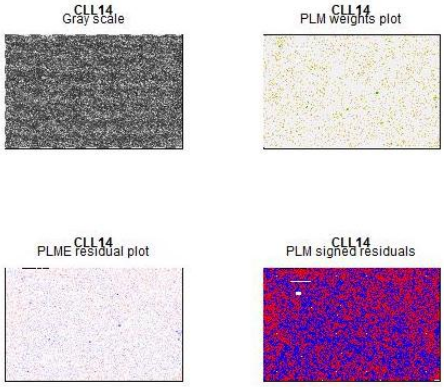
Chip	slope of RNA degradation sl_i
CLL11.CEL	0.485
CLL12.CEL	0.424
CLL13.CEL	0.056
CLL14.CEL	0.317
CLL15.CEL	0.544
CLL16.CEL	0.719
CLL17.CEL	0.843
CLL18.CEL	0.586
CLL19.CEL	0.843
CLL1.CEL	1.728
CLL20.CEL	0.392
CLL21.CEL	0.138
CLL22.CEL	0.214
CLL23.CEL	0.336
CLL24.CEL	0.882
CLL2.CEL	-0.056
CLL3.CEL	0.200
CLL4.CEL	0.365
CLL5.CEL	0.175
CLL6.CEL	0.154
CLL7.CEL	0.809
CLL8.CEL	0.388
CLL9.CEL	0.195

by A^2**TOOL**, since $|b_{CLL1.CEL-RLE}| > Q3_{B_{RLE}} + 1.5 \cdot IQR_{B_{RLE}}$, where $Q3_{B_{RLE}}$ and $IQR_{B_{RLE}}$ are the upper quartile and the interquartile range respectively of $Boxplot_{B_{RLE}}$. In this case, $|b_{CLL1.CEL-RLE}| = 0.163 > 0.070$. On the other hand $v_{CLL1.CEL-RLE} > Q3_{V_{RLE}} + 1.5 \cdot IQR_{V_{RLE}}$ where $Q3_{V_{RLE}}$ and $IQR_{V_{RLE}}$ are the upper quartile and the interquartile range respectively of $Boxplot_{V_{RLE}}$. In this case, $|v_{CLL1.CEL-RLE}| = 0.566 > 0.352$.

- *Normalized Unscaled Standard Error (NUSE) plot.* The NUSE plot is shown in Fig. 7.9b) and it can be observed that CLL1.CEL array has a box which is significantly elevated with respect to $NUSE = 1$. The box also shows more spread relative to other arrays, so it might indicate a quality problem. This fact is confirmed in table 7.7 where CLL1.CEL is detected as defective array by A^2**TOOL**, since $|b_{CLL1.CEL-NUSE}| > Q3_{B_{NUSE}} + 1.5 \cdot IQR_{B_{NUSE}}$, where



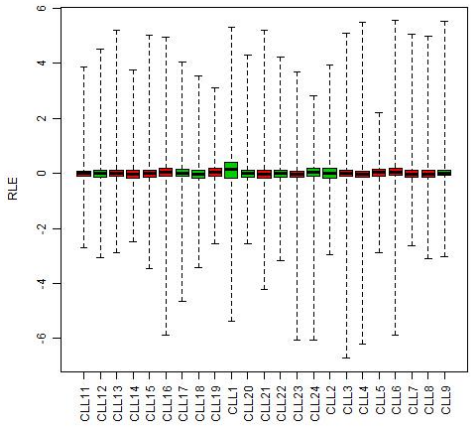
(a)



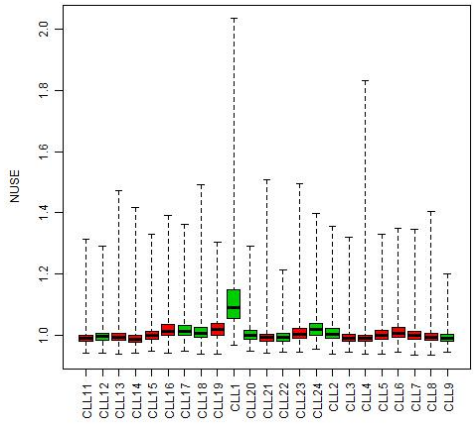
(b)

FIGURE 7.8: Chip pseudo-images based on PLM fit on arrays CLL1.CEL (a) and CLL14.CEL (b). For each array, there are four plots: log transformed gray scale image (top left), PLM weights plot (top right); PLM residual plot (bottom left); PLM signed residuals (bottom right) signed residuals

$Q3_{B_{NUSE}}$ and $IQR_{B_{NUSE}}$ are the upper quartile and the interquartile range respectively of $Boxplot_{B_{NUSE}}$. In this case, $|b_{CLL1.CEL-NUSE}| = 1.090 > 1.028$. On the other hand, $v_{CLL1.CEL} > Q3_{V_{NUSE}} + 1.5 \cdot IQR_{V_{NUSE}}$, where $Q3_{V_{NUSE}}$ and $IQR_{V_{NUSE}}$ are the upper quartile and the interquartile range respectively of $Boxplot_{V_{NUSE}}$. In the case of CLL1.CEL sample, that is, $|v_{CLL1.CEL-NUSE}| = 0.095 > 0.048$.



(a)



(b)

FIGURE 7.9: Relative Log Expression (RLE) plot (a) and Normalized Unscaled Standard Error (NUSE) plot (b) for the CLL experiment

Quality metrics Summary The following table shows whether a sample from the CLL experiment has been detected as defective taking into account all the quality metrics described above.

TABLE 7.7: Relative Log Expression (RLE) and Normalized Unscaled Standard Error (NUSE) values for the CLL experiment

Chip	b_{i-RLE}	v_{i-RLE}	b_{i-NUSE}	v_{i-NUSE}
CLL11.CEL	-0.006	0.185	0.989	0.020
CLL12.CEL	-0.005	0.255	0.995	0.023
CLL13.CEL	-0.001	0.217	0.994	0.022
CLL14.CEL	-0.045	0.259	0.988	0.021
CLL15.CEL	-0.006	0.238	1	0.025
CLL16.CEL	0.044	0.280	1.014	0.035
CLL17.CEL	0.011	0.258	1.013	0.034
CLL18.CEL	-0.029	0.285	1.007	0.031
CLL19.CEL	0.031	0.289	1.018	0.038
CLL1.CEL	0.163	0.566	1.090	0.095
CLL20.CEL	0	0.256	1	0.027
CLL21.CEL	-0.045	0.272	0.992	0.023
CLL22.CEL	-0.011	0.224	0.993	0.023
CLL23.CEL	-0.018	0.212	1.002	0.032
CLL24.CEL	0.034	0.294	1.018	0.039
CLL2.CEL	0.009	0.346	1.002	0.031
CLL3.CEL	0.015	0.227	0.990	0.022
CLL4.CEL	-0.031	0.213	0.989	0.020
CLL5.CEL	0.029	0.240	1	0.027
CLL6.CEL	0.052	0.252	1.007	0.033
CLL7.CEL	-0.018	0.239	0.999	0.025
CLL8.CEL	-0.016	0.226	0.994	0.022
CLL9.CEL	0	0.180	0.991	0.021

TABLE 7.8: Summary of quality metrics results on CLL experiment

Chip	Img*	Boxp	Hist*	MA plots*	AB	SF	PP	RNAdeg	Chip PsI*	RLE	NUSE
CLL11.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL12.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL13.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL14.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL15.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL16.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL17.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL18.CEL	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓
CLL19.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL1.CEL	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗
CLL20.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL21.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL22.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL23.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL24.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL2.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL3.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL4.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL5.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL6.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL7.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL8.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLL9.CEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Img: *Image plots*; Boxp: *Boxplots*; Hist: *Histograms*; MA plots: *MA plots*
AB: *Average Background*; SF: *Scale Factors*; PP: *Percent Present*; RNAdeg: *RNA degradation*
Chip PsI: *Chip Pseudo-images*; RLE: *Relative Log Expression*;NUSE: *Normalized Unscaled Standard Error*
*: this quality metric needed visual inspection

As can be observed from table 7.8, CLL18.CEL has been detected as defective in the *Average Background* quality metric and CLL1 has been detected as defective in *Scale Factors, Percent Present, RNA degradation, Chip Pseudo-images, RLE* plot and *NUSE* plot. A^2**TOOL** (see section 7.2.1) recommends to remove a chip from the experiment if it was detected as defective in two or more quality metrics and since CLL1.CEL has been detected defective in six metrics (five of them automatically by A^2**TOOL**), it is recommended to remove such sample from the experiment. This results are similar to the ones given in [Gentleman et al., 2005], in which a quality analysis was run on the same experiment. The authors recommended to remove CLL1.CEL sample from the experiment, since this sample could be a problem for further analysis (detection of differentially expressed genes, classification, clustering, etc).

7.3.1.2 Data pre-processing

According to the results given in the previous section, we decided to remove CLL1.CEL sample from the experiment due to its low quality. Thus, our experiment has now 22 samples.

The second step carried out by A^2**TOOL** is to pre-process the reduced raw data set using all possible and allowed combinations of Background correction, Normalization, PM correction and Summarization methods described in section 6.2.2 (see table B.1). From the set of all possible pre-processing methods, P, the goal is to select the best of them for the given reduced CLL raw data set according to the procedure described in section 7.2.2.1.

Best pre-processing methods selected Among the 79 different pre-processing methods run (table B.1), a total of 7 (table 7.9) have fulfilled the quality requirements described in section 7.2.2.1.

The score column in table7.9 is a quality measure of the pre-processing method, which is the sum of the ranks obtained when ordering the selected methods according to the replicate variability, K-S statistic and Spearman Rank Correlation

TABLE 7.9: Best pre-processing methods, P', selected for the CLL experiment

Method	known as	R function	Bioc. package	Score
rma, quantiles, pmonly, medianpolish	RMA	rma	AFFY	6
none, invariantset, pmonly, medianpolish	Custom 54	expresso	AFFY	8
-,vsn,pmonly,medianpolish	VSN	vsnrma	VSN	10
mas, quantile, pmonly, medianpolish	CP	threestep	AFFYPLM	12
rma, scaling, pmonly, medianpolish	Custom 13	threestep	AFFYPLM	14
rma, invariantset, pmonly, medianpolish	Custom 17	expresso	AFFY	15
mas, scaling, pmonly, medianpolish	Custom 24	threestep	AFFYPLM	19

Coefficient values. The lower value, the better the method (see section 7.2.2.1, step 5 of the procedure).

It must be noticed that A^2**TOOL** selects pre-processing methods that not only improve the quality metrics with respect to raw expression data, but also do not introduce correlation artifacts on the uninformative data sets or if it does, the quantity of artifacts must be below a threshold. Thus, first of all, let us have a look to the pre-processing methods that improve raw expression data and do not introduce correlation artifacts (table 7.10).

TABLE 7.10: Preprocessing methods (a total of 6) that improve raw expression data and do not introduce correlations artifacts (CLL experiment)

known as	Replicate variability	Spearman Corr. Coefficient	K-S test
RAW	0.297±0.038	0.9690±0.0031	0.179±0.044
VSN	0.201±0.012	0.9717±0.0031	0.0188±0.0008
Custom 13	0.280±0.019	0.9755±0.0025	0.0416±0.0096
Custom 17	0.285±0.004	0.9753±0.0025	0.0393±0.0055
RMA	0.255±0.012	0.9759±0.0026	0.0136±0.0002
Custom 24	0.266±0.017	0.9691±0.0032	0.0422±0.0089
CP	0.240±0.011	0.9694±0.0031	0.0159±0.0004

It can be observed from table 7.10 that all methods selected in table 7.9 do not introduce correlation artifacts except *Custom 54* method. In table 7.11, it can be observed the quality metrics for this pre-processing method with regard to raw expression data.

Let us reproduce for *Custom 54* method the step 4.c of the procedure described in section 7.2.2.1. The quality metrics related to both the uninformative pre-processed

TABLE 7.11: Preprocessing methods (a total of 1) that improve raw expression data and introduce false correlations below the threshold (CLL experiment)

known as	Replicate variability	Spearman Corr. Coefficient	K-S test
RAW	0.297±0.038	0.9690±0.0031	0.179±0.044
Custom 54	0.153±0.009	0.9714±0.0032	0.0159±0.0005

data using this pre-processing method and the uninformative raw expression data are:

- Replicate variability: $\mu_{G_{sd}}(esetC_{Custom54}) = 0.43$, $\mu_{G_{sd}}(esetRawC) = 0.42$
- Spearman Rank Correlation Coefficient: $\mu_{G_{corr}}(esetC_{Custom54}) = 0.029$ and $\mu_{G_{corr}}(esetRawC) = 0.028$

It is clear from table 7.11 that the quality metrics given by *custom 54* on informative pre-processed data are improved with respect to informative pre-processed raw expression data. Now, let us check whether the method introduces correlation artifacts. It can be observed (second *if* of the procedure) that:

$$\mu_{G_{sd}}(esetC_{Custom54}) > \mu_{G_{sd}}(esetRawC), \text{ that is, } 0.43 > 0.42$$
$$\mu_{G_{corr}}(esetC_{Custom54}) > \mu_{G_{corr}}(esetRawC), \text{ that is, } 0.029 > 0.028$$

which means that this pre-processing method does not introduce artifacts when the variability metric is used (the pre-processed uninformative data set is worst than the pre-processed uninformative raw expression data) but it does when the Spearman Coefficient is taken into account: the coefficient is improved for the pre-processed uninformative data set with regard to the pre-processed uninformative raw expression data. Then, we have to measure whether this improvement is significant or not (*else* part of the procedure):

$$\mu_{G_{sd}}(O) = \mu_{G_{sd}}(esetO_{Custom54}) - \mu_{G_{sd}}(esetRawO)\text{=0.153-0.297=-0.144}$$
$$\mu_{G_{sd}}(C) = \mu_{G_{sd}}(esetC_{Custom54}) - \mu_{G_{sd}}(esetRawC)\text{=0.43-0.42=0.01}$$
$$\mu_{G_{corr}}(O) = \mu_{G_{corr}}(esetO_{Custom54}) - \mu_{G_{corr}}(esetRawO)\text{=0.9714-0.9690=0.0024}$$
$$\mu_{G_{corr}}(C) = \mu_{G_{corr}}(esetC_{Custom54}) - \mu_{G_{corr}}(esetRawC)\text{=0.029-0.028=0.001}$$

This means that:

$$\frac{\mu_{G_{sd}}(C)}{\mu_{G_{sd}}(O)} < threshold, \text{ that is, } \frac{0.01}{-0.144} = -0.07 < 0.5$$
$$\frac{\mu_{G_{corr}}(C)}{\mu_{G_{corr}}(O)} < threshold, \text{ that is, } \frac{0.0010}{0.0024} = 0.4 < 0.5$$

As previously stated, there are no artifacts introduced when variability metric is used, but, for the Spearman Correlation Coefficient, the improvement made by the pre-processing method on the uninformative data set, $\mu_{G_{corr}}(C)$, with regard to the improvement made on the informative data set, $\mu_{G_{corr}}(O)$, is not significant (it is below the default threshold which is equal to 0.5) . Thus, this pre-processing method is accepted.

In table 7.12, it can be observed the list of pre-processing methods that, although improve in terms of the qualty metrics with respect to raw data, they introduce many artifacts on the pre-processed data set, that is, the improvement made by the pre-processing method on the uninformative data set with regard to the improvement made on the informative data set is significant (it is above the default threshold). Therefore, they are not accepted.

TABLE 7.12: Preprocessing methods (a total of 9) that improve raw data and introduce correlation artifacts above the threshold (CLL experiment)

known as	Replicate variability	Spearman Corr. Coefficient	K-S test
RAW	0.297±0.038	0.9690±0.0031	0.179±0.044
justPlier	0±0	1±0	0±0
DFW	0.0013±0.0001	1±0	0.0009±0.0001
custom 8	0.1629±0.0082	0.9691±0.0028	0.0121±0.0081
Custom 19	0.1236±0.0073	0.9725±0.0022	0.0114±0.0007
Custom 22	0.1241±0.0071	0.9718±0.0021	0.0113±0.0005
Custom 26	0.1206±0.0066	0.9712±0.0025	0.017±0.002
Custom 33	0.1121±0.0069	0.9772±0.0019	0.0110±0.0005
Custom 40	0.1112±0.0066	0.9730±0.0021	0.0114±0.0004
Custom 61	0.1281±0.0073	0.9712±0.0031	0.0173±0.0004

Among the methods accepted, (table 7.9), let us compare them in terms of the quality metrics:

- **Replicate variability**. Through this metric, the lowest variability is achieved by the *Custom 54* method. This result is expected because its normalization

step is *Invariant set*, which uses a "rank invariant set" for the normalization, keeping the expression ratio values between chips under consideration unchanged by forcing the selected non-differentially expressed genes to have equal values. Thus, it is expected low variability, since the majority of genes in an experiment are expected to be non-expressed. On the other hand, VSN method achieves the second lowest value, which is not unexpected, because this normalization aims to stabilize the variance across the replicated arrays [Florido et al., 2009c] (see section 6.2.2.2).

- **Kolmogorov-Smirnov test**. The lowest K-S statistic is achieved by the RMA method. This is due to the use of the *Quantile* normalization procedure (see section 6.2.2.2), which forces the empirical distributions in different slides to be identical. Similar results are achieved in [Florido et al., 2009c]. Notice that the second lowest value corresponds to CP and Custom 54 methods. The former also uses the *Quantile* normalization method and the latter uses *Invariant set*, in which forcing the selected non-differentially expressed genes to have equal values would produce similar empirical distributions.

- **Rank Spearman Correlation Coefficient**. RMA achieves the best value for this metric, although the differences among all pre-processing methods in terms of this quality metric do not seem to be significant, similar to the results given in [Florido et al., 2009c].

According to these quality metrics, the best method selected by A^2**TOOL** is RMA (see table 7.9), which achieves the best results in two quality metrics (Spearman coefficient and K-S statistic) as in [Florido et al., 2009c]. This method is one of the most well known in the literature [Bolstad et al., 2003]. The second method is a custom one (Custom 54), which achieves the best results in terms of replicate variability and a very good value in terms of the K-S statistic.

On the other hand, it can be observed in terms of the overall score (table 7.9) the strong influence of changing a single step in the pre-processing method as stated in [Harr and Schlötterer, 2006]. In table 7.13, the differences between pairs of pre-processing methods are shown in terms of the overall score when a single step in

the pre-processing method is changed. For example, RMA and Custom 17 methods differ only in the normalization step and the difference in the overall score is quite high.

TABLE 7.13: Differences between the selected pre-processing methods in terms of their overall score when a single step in the pre-processing method is changed

Methods	Differences in	Score difference
RMA - Custom 17	Normalization	-9
Custom 54 - Custom 17	Background correction	-7
CP - Custom 24	Normalization	-7
RMA - CP	Background Correction	-6
Custom 13 - Custom 24	Background correction	-5

Some other important conclusions can be extracted from the pre-processing methods selected for the CLL experiment (table 7.9). For example, the PM-MM correction and summarization methods used by the selected pre-processing methods are *pmonly* and *median polish* respectively, which only use information of the PM probes. Although the MM probe is supposed to distinguish noise caused by non-specific hybridization from the specific hybridization signal, these results might suggest that the MM probes are useless in these pre-processing steps, which are in tune with the suggestions of some researchers that recommend avoiding the use of MM probes [Gautier et al., 2004]. This might also suggest that the most critical steps when pre-processing are background correction and normalization. On the other hand, it can also be concluded that it is important not only to use the standard pre-processing methods, such as RMA, VSN and CP, but also custom pre-processing methods that may provide good pre-processing to a given experiment. Remember that each experiment is completely different and has many sources of variations. Thus, it is important to search for the best pre-processing method for a given experiment through objective quality metrics.

7.3.1.3 Detection of differentially expressed genes

Given the best pre-processing methods selected in the previous section $P' = \{$RMA, Custom 54, VSN, CP, Custom 13, Custom 17, Custom 24$\}$, the next step carried out by A^2**TOOL** is to detect differentially expressed genes through each pre-processed data or expression set $esetO_i$ where $esetO_i \leftarrow p_i(rawData), \forall p_i \in P'$. According

to the guidelines given in section 7.2.3, for each $esetO_i$, the following two-step procedure to obtain differentially expressed genes is applied by A^2**TOOL**:

- Apply nonspecific filtering to $esetO_i$ using the *geneFilter* package, obtaining a reduced expression set $esetO'_i$ (see section 7.2.3.1). Affymetrix control genes are removed, the IQR filtering statistic is applied with a default value of 0.5 and genes with the same EntrezID are removed, keeping the one with the greatest variability. The number of genes that remain after nonspecific filtering is applied is 4399 for $esetO'_i, \forall i$
- Apply a moderated *t*-test analysis to $esetO'_i$ through the *limma* package. For each gene listed as differentially expressed, the following statistics are presented: *ID*, *EntrezID*, *logFC*, *AveExpr*, *t*, *adj.P.Val* and *B* (see section 7.2.3.1 for further details) with default values for the following thresholds: $logFC = 0.585$ and $p_{threshold} = 0.05$ with FDR correction.

Thus, in table 7.14, the genes detected as differentially expressed on each reduced pre-processed data, $esetO'_i$, are shown:

Merging results: intersection and union lists Two more lists of differentially expressed genes are constructed by A^2**TOOL**. The first one is the set of genes that are very likely to be differentially expressed (a reliable list). This list is constructed through the intersection of differentially expressed genes detected in the different expression sets (table 7.15).

The second list is the set of genes that are likely to be differentially expressed. It is constructed through the union of differentially expressed genes detected in the different expression sets (table 7.16). This list is a complete list of genes and it is useful in cases where a gene is detected as differentially expressed in all the expression sets but one. For example, the gene whose *Entrez ID* is 1303 is detected in all expression sets but the one generated by *Custom 54* pre-processing method. Moreover, the significance of that gene is very high (very low *p-value*) and, without the use of this union list, it would be lost.

TABLE 7.14: Differentially expressed genes for the selected pre-processing methods P' and progressive-stable contrast in the CLL experiment. Genes marked with * are also selected by the Bonferroni Multiple Testing Correction

ID	EntrezID	logFC	AveExpr	t	adj.P.Val	B
RMA						
1303_at *	6452	-1.07	5.57	-6.22	0.0081	4.71
33791_at *	10301	1.44	6.3	5.61	0.015	3.43
36939_at	2823	2.08	6.29	5.42	0.015	3.02
37636_at	9767	1.13	5.91	5.4	0.015	2.98
36131_at	1192	0.77	9.48	5.33	0.015	2.83
41776_at	475	0.79	7.74	4.82	0.039	1.72
Custom 54						
36131_at	1192	0.7	10.4	5.54	0.021	3.21
36939_at	2823	1.15	7.94	5.21	0.021	2.51
37636_at	9767	0.59	7.64	5.18	0.021	2.44
33791_at	10301	0.81	7.89	4.87	0.039	1.77
37676_at	5151	-0.62	8.78	-4.65	0.046	1.29
39967_at	23641	0.68	7.64	4.55	0.046	1.08
VSN						
1303_at *	6452	-0.66	6.33	-6.21	0.0087	4.65
36939_at	2823	1.41	6.87	5.42	0.030	2.99
36131_at	1192	0.78	9.49	5.27	0.030	2.66
37636_at	9767	0.73	6.56	5.09	0.031	2.28
33791_at	10301	0.94	6.84	5.06	0.031	2.21
CP						
1303_at *	6452	-0.87	5.99	-6.07	0.011	4.49
33791_at *	10301	1.17	6.62	5.53	0.015	3.32
37636_at *	9767	0.98	6.27	5.48	0.015	3.22
36131_at	1192	0.77	9.51	5.39	0.015	3.01
36939_at	2823	1.8	6.61	5.28	0.016	2.78
39967_at	23641	1.04	6.26	4.69	0.039	1.48
41776_at	475	0.67	7.85	4.69	0.039	1.46
Custom 13						
1303_at *	6452	-1.08	5.54	-6.41	0.0045	5.07
33791_at *	10301	1.42	6.28	5.49	0.017	3.16
36939_at *	2823	2.07	6.29	5.46	0.017	3.1
36131_at	1192	0.79	9.48	5.19	0.019	2.53
37636_at	9767	1.1	5.91	5.19	0.019	2.52
41776_at	475	0.78	7.73	4.82	0.044	1.7
Custom 17						
1303_at *	6452	-1.1	6.35	-6.24	0.0062	4.89
33791_at *	10301	1.46	7.1	5.74	0.011	3.81
36939_at *	2823	2.12	7.09	5.52	0.013	3.33
37636_at	9767	1.12	6.72	5.4	0.014	3.07
36131_at	1192	0.76	10.3	5.15	0.021	2.51
41776_at	475	0.80	8.54	4.78	0.045	1.71
39967_at	23641	1.29	6.6	4.72	0.045	1.57
Custom 24						
1303_at *	6452	-0.9	6.01	-6.36	0.0042	5.14
33791_at *	10301	1.13	6.64	5.46	0.019	3.23
36939_at	2823	1.77	6.61	5.33	0.019	2.93
36131_at	1192	0.78	9.51	5.25	0.019	2.75
37636_at	9767	0.92	6.28	5.06	0.025	2.33

TABLE 7.15: Intersection of differentially expressed genes detected in the different expression sets for the CLL experiment. The average log2 expression level of each gene is shown

ID	EntrezID	Average log_2 expression level						
		RMA	Cust.54	VSN	CP	Cust.13	Cust.17	Cust.24
33791_at	10301	6.3	7.89	6.84	6.62	6.28	7.1	6.64
36939_at	2823	6.29	7.94	6.87	6.61	6.29	7.09	6.61
37636_at	9767	5.91	7.64	6.56	6.27	5.91	6.72	6.28
36131_at	1192	9.48	10.4	9.49	9.51	9.48	10.3	9.51

TABLE 7.16: Union of differentially expressed genes detected in the different expression sets for the CLL experiment. The average log2 expression level of each gene is shown

ID	EntrezID	Average log_2 expression level						
		RMA	Cust.54	VSN	CP	Cust.13	Cust.17	Cust.24
33791_at	10301	6.3	7.89	6.84	6.62	6.28	7.1	6.64
36939_at	2823	6.29	7.94	6.87	6.61	6.29	7.09	6.61
37636_at	9767	5.91	7.64	6.56	6.27	5.91	6.72	6.28
36131_at	1192	9.48	10.4	9.49	9.51	9.48	10.3	9.51
1303_at	6452	5.57	-	6.33	5.99	5.54	6.35	6.01
41776_at	475	7.74	-	-	7.85	6.28	8.54	-
37676_at	5151	-	8.78	-	-	-	-	
39967_at	23641	-	7.64	-	6.26	-	6.6	-

Also, through the intersection and union tables, the average expression value in log_2 scale of genes are shown. It can be observed differences in the expression values, which means that the expression sets generated by different pre-processing methods have some differences among them. Thus, through this example, it is demonstrated that the selection of a pre-processing method affects the detection of differentially expressed genes as also stated in [Bolstad, 2004] and [Ritchie et al., 2007]. However, A^2**TOOL** alleviates this problem, providing two lists of differentially expressed genes constructed from several expression sets given by the best pre-processing methods according to objective quality metrics.

7.3.1.4 Discussion

According to these results, it has been demonstrated the uselfulness of A^2**TOOL** in terms of:

1. **Quality data analysis**. Through this analysis, the tool detected CLL1.CEL sample as defective. Removing this array from the experiment will guarantee to make the best use of the information produced by the rest of arrays.

2. **Data pre-processing**. Seven pre-processing methods (RMA, Custom 54, VSN, CP, Custom 13, Custom 17 and Custom 24) were selected according to three objective quality metrics. This guarantees that the pre-processing methods used have the best quality.

3. **Detection of differentially expressed genes**. A set of genes detected as differentially expressed on each pre-processed data selected is presented. To get a reliable and a complete list of differentially expressed genes, intersection and union lists were built.

A^2**TOOL** has the following advantages:

- Automated quality assessment of the experiment to detect defective arrays according to quantitative and qualitative measures.
- Automated selection of the best pre-processing methods among several ones for a given data set through objective quality measures. This way, the researcher does not take the decision about the pre-processing method(s) to be used.
- Two lists of differentially expressed genes are obtained: a reliable list and a complete one.

However, A^2**TOOL** has the following deficiencies:

- It does not support other microarray technologies, such as Agilent or Genepix or new Affymetrix-based technologies, such Gene or Exon arrays.
- It does not support other methodologies for detecting differentially expressed genes, such as SAM or ANOVA.
- It does not support high level analysis, such as clustering or classification.
- It is not a user-friendly GUI.

7.3.2 Effect of pre-processing methods on Microarray-based SVM classifiers

According to the results given in previous section, the selection of a pre-processing method affects the detection of differentially expressed genes. However, such selection may affect other posterior stages of the microarray data analysis pipeline, such as phenotype classification, since different pre-processing methods can lead to different expression data. In this section, the influence of five different pre-processing methods commonly used in the literature (RMA, GCRMA, VSN, dChip and MAS5, see table B.1) are assessed in Support Vector Machine-based classification methodologies with different kernels (linear, Radial Basis Functions and polynomial) on several cancer microarray data sets. The results presented in this section are obtained from the work developed by Florido et al. [Florido et al., 2010b].

7.3.2.1 Data sets

Seven publicly available microarray data sets are chosen for the study. The basic information about these data sets are summarized in table 7.17 and all of them are available as raw data.

1. *ALL-MLL-AML Leukemia Data*. This leukemia microarray data set [Armstrong et al., 2002] includes 72 human leukemia samples, 24 of them belonging to acute lymphoblastic leukemia (ALL), 20 of them to mixed lineage leukemia (MLL), a subset of human acute leukemia with a chromosomal translocation, and 28 of the samples are acute myelogenous leukemia (AML).

2. *Bladder cancer data*. Bladder cancer is a common malignant disease characterized by frequent recurrences. The stage of disease at diagnosis and the presence of surrounding carcinoma in situ are important in determining the disease course of an affected individual. In this context, the authors [Dyrskjot et al., 2003] report the identification of clinically relevant subclasses of bladder carcinoma using expression microarray analysis: Ta, T1 and T2+.

3. *Colon cancer data*. This data set [Laiho et al., 2006] is composed of 37 microarrays, 8 of them belong to serrated colorectal carcinomas (CRCs) and 29 conventional CRCs. Serrated colorectal carcinomas are morphologically different from conventional CRCs and have been proposed to follow a distinct pathway of CRC formation.

4. *Brain Cancer data sets*. In the first data set [Nutt et al., 2003] microarray analysis was used to determine the expression of approximately 12,000 genes in a set of 50 gliomas: 28 glioblastomas (G) and 22 anaplastic oligodendrogliomas (O), which were classified as classic (C) or non-classic (N). Thus, there are 14 Classic Glioblastomas (CG), 7 Classic Oligodendrogliomas (CO), 14 Non-classic Glioblastomas (NG) and 15 Non-classic OligoDendrogliomas (NO). In the second dataset [Pomeroy et al., 2002], embryonal tumors of the central nervous system represent a heterogeneous group of tumors whose diagnosis, on the basis of morphologic appearance alone, is controversial. Medulloblastomas (MD), for example, are the most common malignant brain tumor of childhood, but their pathogenesis is unknown and patients' response to therapy is difficult to predict. The authors demonstrate that medulloblastomas (MD, 10 samples) are molecularly distinct from other brain tumors including primitive neuroectodermal tumors (PNETs, 8 samples), atypical teratoid/rhabdoid tumors (Rhab, 10 samples) and malignant gliomas (Mglio, 10 samples). Also, 4 normal tissues were considered (Ncer).

5. *Blood Cancer data set*. In this data set [Shipp et al., 2002], the authors investigated whether a supervised learning algorithm could generate a classifier able to discriminate tumors within a single (B-cell) lineage. Specifically, they asked weather the classifier could distinguish diffuse large B-cell lymphoma (DLBCL, 58 samples) from a related GC B-cell lymphoma, follicular (FL, 19 samples). Although these two malignancies have very different clinical presentation, natural histories and responses to therapy, FLs often evolve over time and acquire the morphologic and clinical features of DLBCLs.

6. *Prostate Cancer data set*. In this data set [Singh et al., 2002], the authors studied gene expression patterns from 52 prostate tumors (PR) and 50 normal prostate specimens (N) in order to ask whether such patterns could be

used to predict common clinical and pathological phenotypes relevant to the treatment of men diagnosed with this disease.

TABLE 7.17: Number of samples, number of classes, distribution of samples within classes and number of original genes in cancer data sets

Data set	#samples	#C	Dist.Classes	#Genes
Leukemia	72	3	24,20,28	12626
Bladder cancer	40	3	20,11,9	7129
Colon cancer	37	2	8,29	22283
Brain cancer-CG	50	4	14,7,14,15	12625
Brain cancer-MD	42	5	10,10,4,8,10	7129
Boold cancer	77	2	58,19	7129
Prostate cancer	102	2	52,50	12625

7.3.2.2 Experimental settings and results

The pre-processing methods selected are those commonly used in the literature: RMA, GCRMA, VSN, dChip and MAS5 [2]. The CMA package [Slawski et al., 2008] was used for the SVM classifier. We applied 10-fold cross-validation [Kohavi, 1995] to estimate the performance of the classification algorithm. In order to optimize SVM parameters, we adopted another "nested" loop of cross-validation by further splitting each of the 10 original learning sets into smaller training and validation sets. For each combination of the classifier parameters, the cross-validation performance is obtained and we selected the best performing parameters inside this inner loop of cross-validation. Then, a classification model is built with the best parameters on the learning set and applied this model to the testing set. Three kernels are used for the SVM classifier: linear, radial (RBFs) and polynomial. The parameters (see section 6.2.4.1) to be optimized by nested cross-validation are:

- Linear: C values {0.1, 1, 5, 10, 50, 100, 500}
- Polynomial: C values {0.1, 1, 5, 10, 50, 100, 500} and d values {2,3,4}. γ and r values are fixed ($\gamma = 1$ and $r = 1$).

[2] It must be noticed that, in this study, the introduction of correlation artifacts by any of the pre-processing methods has not been assessed

- Radial: C values {0.1, 1, 5, 10, 50, 100, 500} and γ values {0.25, 0.5, 1, 2, 4}

For multi-category data, the "one vs all" (OVA) works better in SVMs for multi-class data [Statnikov et al., 2008], so we adopted this method for the analysis of multi-category data sets. The performance measure used for SVM is the misclassificaton rate.

Since the sample size is much smaller than the dimensionality of the data sets, too many genes may not be helpful for the class discrimination. In this experiment and because this comparison is not intended to compare different gene selection methodologies, the *t-test*-based gene selection approach is used because its simplicity. To find the genes that contribute most to the classification, a score based on *t-test* (named *t*-score or *TS*) is calculated for each gene. Then, all the genes are rearranged according to their *TSs* and only some top genes in the list are used for classification. The number of genes selected in this list is also varied according to the following values $\{0, 5, 10, 25, 50, 100, 200, 400, 600, 800, 1000, 2000\}$. Raw data is also used in the comparison, that is, the data sets were pre-processed with no background correction, no normalization and no PM correction. Just the summarization step is needed to obtain gene expression values and, according to [Florido et al., 2009c], *median-polish* summarization is used.

Thus, for each data set, pre-processing method, number of genes selected and kernel function, the SVM classifier is applied, using 10 different partitions of the 10-fold cross-validation procedure. The results are analyzed through a four-way ANalysis Of VAriance (ANOVA) test [Box et al., 1978], which evaluates whether the observed difference in classification performance is statistically significant or simply due to chance. The factors for this test are the data set, the pre-processing method, the number of genes selected and the type of kernel used in the SVM-based classifier, the dependent or response variable is the misclassification rate. The statistical parameter considered is the significant level and if its lower than 0.05, then the corresponding levels of the factor are statistically significant with a confidence level of 95%. The normality, independence of populations and homocedasticity assumptions for ANOVA test are accomplished.

It can be observed from table 7.18, that all the factors (pre-processing method, number of genes selected, kernel function and data set), present the greatest statistical relevance, which means that the misclassification rate statistically depends on these factors.

TABLE 7.18: ANOVA table for the analysis of the main factors for misclassification rate response variable

Main factors	Sum of squares	D.F.	Mean square	F-Ratio	Sig.level
Prep. method	0.16	5	0.03	9.68	0
Num. genes	0.14	11	0.01	3.68	0
Kernel	4.48	2	2.24	666.67	0
Data set	59.09	6	9.85	2934.33	0

Figure 7.10 shows that there are no statistical differences ($P > 0.05$) among dChip, RMA, VSN and MAS5 methods but there are statistical differences between them and the GCRMA and Raw data ($P < 0.05$). This suggests that a pre-processing method is important not only in terms of differentially expressed genes, but also in terms of classification performance. Also, the GCRMA gives a misclassification rate very similar to raw data. Thus, GCRMA method is not recommended as pre-processing method for higher-order tasks such as classification, because its performance is similar to perform no pre-procesing step, that is, to use the raw data directly as input to the classifier. It is obvious that the under-performance of GCRMA is due to its background correction step (see Table B.1), since it is the only step where GCRMA and RMA differ: the background correction used in GCRMA is designed to account for background noise, as well as non-specific binding [Wu et al., 2004]. It seems that these differences with respect to RMA make the misclassification rate achieves worse results.

Although this comparison was intended to study the effect of pre-processing methods in terms of classification rate, it would be also interesting to study whether the number of genes selected in the feature selection step and the kernel method used in the SVM classifier affect the results.

From Fig.7.11, it can be observed that the accuracy of SVM is affected by the number of genes selected by *t-test*. There are no statistical differences ($P > 0.05$) when the number of genes selected varies from 10 to 400. On the other hand,

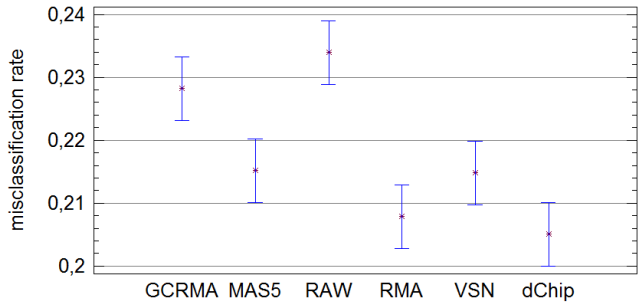


FIGURE 7.10: Means and 95% Least Significant Differences (LSD) intervals of the different pre-processing methods through the misclassification rate response variable

when very few genes (5) are selected or the number is large (600-2000 and the whole chip) SVM's performance gets worse. In the first case, the data does not contain enough discriminative information and, in the second case, a large number of irrelevant genes may be harmful for the class discrimination, acting as "noise" and affecting SVM's accuracy [Xiong et al., 2007].

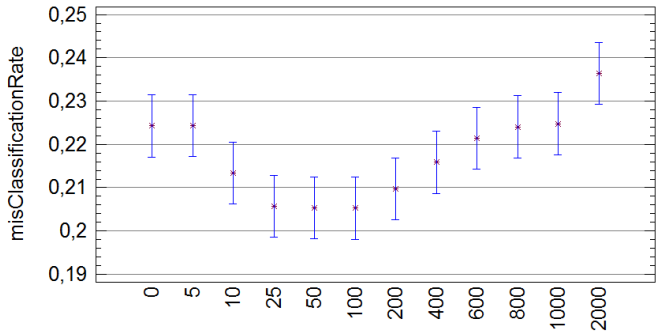


FIGURE 7.11: Means and 95% Least Significant Differences (LSD) intervals of the different number of genes selected in the feature selection step through the misclassification rate response variable

According to the kernel used (Fig.7.12), polynomial kernel performs statistically worse ($P < 0.05$) than linear and radial kernels, whereas there are no statistical differences ($P > 0.05$) between the latter. These results suggest that the problems are intrinsically linear and, therefore, the radial kernel chooses parameters gamma that make the learned decision boundary almost linear. This conclusion is also

consistent with the one given in a benchmark study [Pochet et al., 2004] in which well-tuned RBF kernels achieve results as good as their linear counterparts.

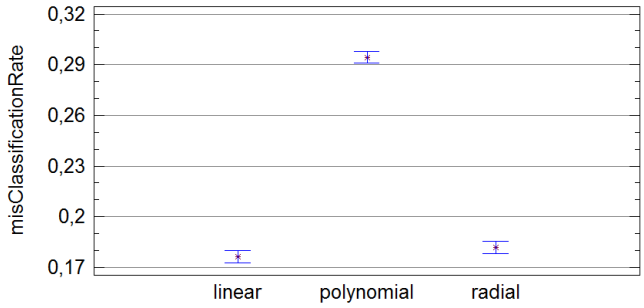


FIGURE 7.12: Means and 95% Least Significant Differences (LSD) intervals of the different kernels through the misclassification rate response variable

These results are also compared to the accuracy of pre-processing methods in terms of the quality metrics described in [Florido et al., 2009c] and also used by A^2**TOOL**: data variability, similarity in data distributions and correlation coefficient among replicates. As stated in section 6.2.2.5 smaller variability, similar distributions (measured as Kolmogorov-Smirnov statistic) and higher Spearman correlation coefficient among replicates should be expected after pre-processing step. A two-way ANOVA is run in this case where the factors are the data set and the pre-processing method and the dependent or response variables are the different quality metrics. It can be observed from Fig. 7.13a) that, in terms of data variability, VSN and dChip perform statistically better ($P < 0.05$) than the others. The VSN performance was not unexpected because it specifically aims to stabilize the variance across the replicated arrays [Huber et al., 2002].

With respect to Kolmogorov-Smirnov statistic (Fig.7.13b), RMA performs statistically better ($P < 0.05$) than the others. The performance of RMA and is justified because the use of Quantile normalization algorithm (see table B.1) which forces the empirical distributions in different slides to be identical. According to the Spearman Coefficient (Fig.7.13c), RMA performs statistically better ($P < 0.05$) than the others.

So, according to these quality metrics, RMA, VSN and dChip methods are the best ones and, this fact, is consistent with the results given in terms of misclassification

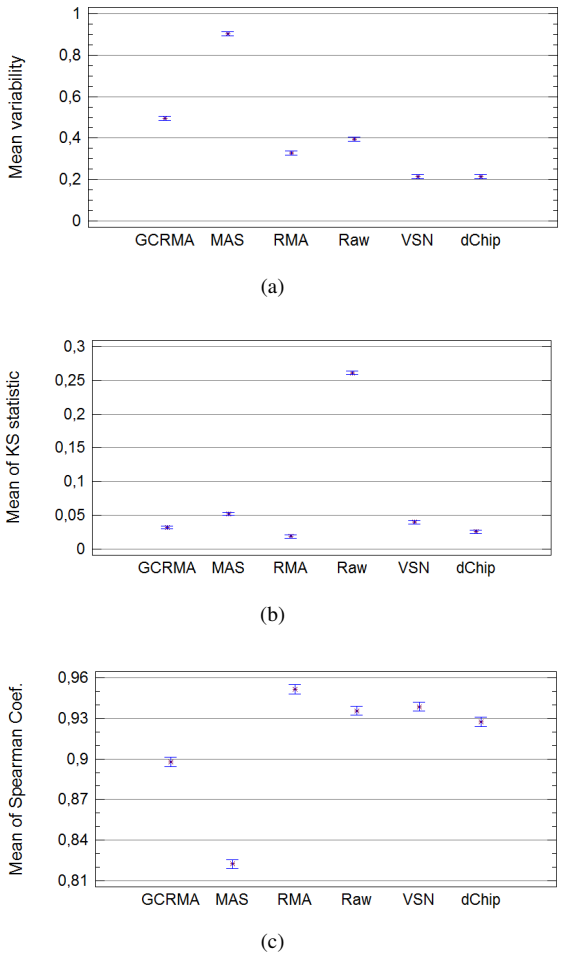


FIGURE 7.13: Means and 95% Least Significant Differences (LSD) intervals of the different pre-processing methods through the mean variability (a), the mean of K-S statistic (b) and the Spearman Coefficient (c) response variables

rate (Fig.7.13). However, MAS shows very poor performance in terms of quality metrics, contrary to its behavior in terms of misclassification.

7.4 Conclusions and future work

It has been demonstrated that A^2**TOOL** is a useful approach for the first three steps of microarray data analysis workflow. Some important conclusions are outlined below:

1. It is important to make the best use of the information produced by the microarray experiments by removing defective arrays if they are detected. This way, the results will have more statistical significance and biological meaning.

2. Some pre-processing methods introduce correlation artifacts on the data set even if they improve raw data. Thus, it is important to detect such pre-processing methods and avoid their use.

3. The performance of a pre-processing method is highly dependent on the selection of each pre-processing step. Moreover, the most critical steps are Background Correction and Normalization.

4. The selection of a pre-processing method has an influence on the list of differentially expressed genes.

5. It is justified the use of two lists of differentially expressed genes: the intersection list provides a reliable list of differentially expressed genes regardless of the pre-processing method selected and the union list shows a complete list of candidate genes that are likely to be differentially expressed. This way, the set of candidate genes does not rely on a single expression data and the effect that the selection of a pre-processing method may have on the detection of differentially expressed genes is alleviated.

As a future work, we can propose the following improvements for A^2**TOOL** tool:

- To develop either a web-based application or a desktop application to avoid the use of R console.
- The use of *Aroma.Affymetrix* package [Bengtsson et al., 2011], which pre-processes Affymetrix microarray data with less computational cost.
- To compare with more experiments. This way, a list of the best pre-processing methods selected in several experiments could be obtained.
- The design of a more complex system, such as supervised learning models, with the aim of recognize data patterns to decide the best-preprocessing method for a given data set.

- The development of an approach that combines the three quality metrics used to evaluate a given pre-processing method.
- Support for the new arrays for Affymetrix: Gene and Exon arrays.

Furthermore, it would be interesting to extract genes related to a given disease from a custom microarray data experiment by means of this tool. Once extracted, they can be labelled in a FLN obtained using the approach described in section 4.3 to predict new genes that are potentially associated with the disease under study.

With respect to the effect of pre-processing methods on microarray-based SVM classifiers, it has been demonstrated that there are no statistical differences among RMA, VSN, dChip and MAS5 pre-processing methods in terms of misclassification rate, but the GCRMA method shows the same performance, statistically speaking, as raw data. It has also been shown that the SVM classifier is sensitive to both feature selection and kernel function: when very few/large number of genes are selected or the polynomial kernel is chosen, SVM's accuracy goes down. On the other hand, well-tuned RBF kernels give similar results to the linear ones. More investigation is needed as a future work to understand the interplay between pre-processing (which improves data quality), feature selection (which improves the classifier by throwing away non-informative data), kernel function (linear vs nonlinear) and their optimized parameters to ascertain pre-processing strategies to produce an optimal classifier.

Part III

Intelligent Systems for Function Approximation

Chapter 8

Introduction

At present, many engineering problems are based on the processing of input/output data sets, namely, feature selection [Patrinos et al., 2010], classification [Chen et al., 2007a] and function approximation tasks [Gonzalez et al., 2003],[Paul and Kumar, 2002],[Huang et al., 2005] such as time series prediction [Sorjamaa et al., 2007] or system identification problems [Ghaffari et al., 2007].

Most of the learning algorithms or techniques proposed in the literature to build a model for feature selection, classification or function approximation problems, split the original input/output data set into two groups: learning and test. The learning set, which can be in turn divided into training and validation sets, is used for building models that capture the relationships between inputs and outputs. On the other hand, the test set is used for checking models' generalization ability with data not used in the learning process [Chen et al., 2007a], [Gonzalez et al., 2003],[Sorjamaa et al., 2007] and [Ghaffari et al., 2007].

When the partition into learning and test sets does not take into account the variability and geometry of the original data, it might lead to non-balanced and unrepresentative learning and test sets (known as the dataset shift problem [Moreno-Torres et al., 2011]) and, thus, to wrong conclusions in the accuracy of the learning algorithm. How the partitioning is made is therefore a key issue and becomes more important when the data set is small due to the need of reducing the pessimistic effects caused by the removal of instances from the original data set.

Thus, in this part of the dissertation, we propose a deterministic data mining approach for a distribution of a data set (input/output data) into two representative and balanced sets of roughly equal size to be used in function approximation problems [Florido et al., 2011a]. The distribution takes into account the variability of the data set with the purpose of allowing both a fair evaluation of learning's accuracy and to make reproducible machine learning experiments usually based on random distributions. The sets are generated using a combination of a clustering procedure, especially suited for function approximation problems, and a distribution algorithm which distributes the data set into two sets within each cluster based on a nearest-neighbor approach.

This methodology is not only useful for the evaluation of learning's accuracy and to make reproducible machine learning experiments, but also in the context of the selection of the best model structure (for example of different complexities, different number of neurons in the hidden layer of multi-layer perceptrons, etc.) for a given problem, which is known as model selection. The evaluation of a given model structure is usually carried out through the splitting of the learning set into two sets: training, used for parameter estimation for a given model structure, and validation, used for evaluating the trained model, obtaining a validation error. Model selection is, therefore, related to the task of comparing several model structures based on estimations of their validation errors in order to select the most suitable model structure for a given problem.

One of most widely used model selection approaches in the literature are those based on the *K-fold cross-validation* [Lendasse et al., 2003], [Aran et al., 2009], [Constantinopoulos and Likas, 2006] model evaluation strategy. However, such model selection approaches have some drawbacks: its random nature (it does not take the variability and geometry of the learning data into consideration when building the training and validation sets) and the subjective decision for a proper value of K, resulting in large bias for low values and high variance and computational cost for high values.

In this sense, we also demonstrate the usefulness of balanced and representative training and validation sets in the context of a new deterministic model selection methodology for incremental Radial Basis Function Neural Network (RBFNN)

construction in time series prediction problems [Florido et al., 2011b]. The development of such special designed methodology is motivated by the problems that arise when using a *K-fold cross-validation*-based model selection methodology described above.

This part of the thesis is structured as follows. Chapter 9 presents the data mining approach for a distribution of a data set (input/output data) into two representative and balanced sets of roughly equal size to be used in function approximation problems. The new deterministic model selection methodology for incremental RBFNN construction in time series prediction problems is presented in Chapter 10.

Chapter 9

Methodologies for generating balanced learning and test sets for function approximation problems

9.1 Motivation and goals

Nowadays, many engineering problems are based on the processing of input/output data sets, for example, function approximation tasks [Gonzalez et al., 2003],[Paul and Kumar, 2002],[Huang et al., 2005] such as time series prediction [Sorjamaa et al., 2007] or system identification problems [Ghaffari et al., 2007].

Time series prediction [Weigend and Gershenfeld, 1994] consists in the prediction of future values of a sequence of past values. The general equation model for time series prediction is:

$$y_{t+h} = F(y_t, y_{t-1}, \ldots, y_{t-p}) \tag{9.1}$$

where $y_t, y_{t-1}, \ldots, y_{t-p}$ are the current and past values of the series and h is the prediction horizon. Thus, the time series prediction problem can be regarded as a problem of estimating F given the input data $y_t, y_{t-1}, \ldots, y_{t-p}$ and the output data y_{t+h} or, in other words, a function approximation problem.

System identification is one of the most important aspects in control, communication, pattern recognition and fault analysis fields. In general, when modeling discrete nonlinear systems, the current output of a dynamic system is a function of both its previous outputs and inputs, which makes the identification problem more complex than for static systems [Ghaffari et al., 2007]. The general equation model for system identification is:

$$y_{t+h} = F(y_t, y_{t-1}, \ldots, y_{t-p}, x_1, x_2, \ldots, x'_p) \tag{9.2}$$

where $y_t, y_{t-1}, \ldots, y_{t-p}$ are the current and past values of the system, h is the identification horizon and $x_1, x_2, \ldots, x'_p$ are exogenous or external values which can influence the plant output. Thus, system identification can be regarded as a problem of estimating an unknown model F given the input data $y_t, y_{t-1}, \ldots, y_{t-p}$, the external data $x_1, x_2, \ldots, x'_p$ and the output data y_{t+h}. Therefore, it can also be considered as a function approximation problem.

Formally, a function approximation problem can be formulated as follows: given a set of observations or input/output vectors $S_{total} = \{x_k, y_k; k = 1, \ldots, n\}$ sampled from an unknown function or system F with $x_k \in \mathbb{R}^d$ and $y_k \in \mathbb{R}$, it is desired to obtain a model F^* so that $y_k^* = F^*(x_k) \in \mathbb{R}$, providing accurate outputs from input data specified in the original data set, and with good predictive performance for new input vectors.

To build a model for function approximation problems, many techniques exist in the literature: there are classical models that try to build a linear model of the process such as the Auto-Regressive Integrated Moving Average (ARIMA), the Auto-Regressive with eXogenous input (ARX) or the Auto-Regressive Moving Average with eXogenous input (ARMAX) models [Ljung, 1986] and non linear models such as Fuzzy Systems [Pomares et al., 2004], Neuro-Fuzzy Systems [Theodoridis et al., 2010], Least-Squares Support Vector Machines (LS-SVM)[Suykens et al., 2002], Nearest Neighbor(NN)-based models [Sorjamaa et al., 2007], Radial Basis Function Neural Networks (RBFNNs) [Park and Sandberg, 1991], Multilayer Perceptrons (MLPs) [Balaguer et al., 2008] or Recurrent Neural Networks [Puscasu et al., 2009].

Most of these techniques usually split the original input/output data S_{total} into two groups: learning, S_{learn}, and test S_{test}. In the literature, the variability and the geometry of S_{total} is usually not taken into account in the partition. This fact might lead to non-balanced and unrepresentative learning and test sets (known as the dataset shift problem [Moreno-Torres et al., 2011]) and, thus, wrong conclusions in the accuracy of the technique chosen.

Thus, our goal is to distribute homogeneously n input/output data into two mutually exclusive, balanced and representative sets by sampling all instances from the original data, S_{total}, without replacement. Let us call these sets, without loss of generalization, learning, S_{learn}, and test,S_{test}, sets.

The following assumptions must be fulfilled:

- $S_{learn} \subseteq S_{total}$ and $S_{test} \subseteq S_{total}$
- $S_{learn} \cap S_{test} = \emptyset$ and $S_{learn} \cup S_{test} = S_{total}$

The learning set, S_{learn}, is used for building a model that captures the relationships between inputs and outputs and the S_{test} set is used for checking the generalization ability of the model built. Both sets should be representative from S_{total} so that they could be interchangeable, that is, S_{test} could be used as a learning set and S_{learn} could be used as a testing set. The sets are generated using a combination of a clustering procedure, especially suited for function approximation problems, and a distribution algorithm which distributes the data set into two sets within each cluster based on a nearest-neighbor approach. The sets obtained result in a reduction of the pessimistic effects caused by the removal of instances from the original data set whose importance is evident when evaluating function approximation models, specially in small data sets. It is expected therefore to allow both a fair evaluation of learning's accuracy and to make reproducible machine learning experiments usually based on random distributions.

This chapter is structured as follows, section 9.2 reviews different ways in the literature for distributing the original data set into two groups or sets. The architecture of the proposed approach is presented and explained in detail in section 9.3. Section 9.4 describes different ways of assessing the quality of the distribution of the

original data set into two sets and, in section 9.5, an experimental comparison of the new algorithm with respect to others present in the literature is reported through a statistical analysis of its performance. Finally, some conclusions and future work are presented in section 9.6.

9.2 State-of-the art

The distribution of the original data set into two groups or sets, namely learning and test sets, can be carried out in several ways and, in the literature, there are different approaches. First of all, the most common approaches are described and then, some attempts to build balanced and representative learning and test sets will be explained.

9.2.1 Common approaches

The most common methodology to partitioning the original data set into learning and test sets is randomly [Paul and Kumar, 2002],[Huang et al., 2005],[Kohavi, 1995], [Dreyfus, 2005]. For example, in [Dreyfus, 2005], the author proposed a methodology in which several random partitions of the original data set are run and the partition for which the Kullback-Leibler (KL) divergence between the learning and test sets is smallest is retained.

Another methodology is to build the sets by sampling uniformly in some domain [Wen and Ma, 2008],[Gonzalez et al., 2003],[Yu and Li, 2004]. This approach is normally not feasible when a set of input/output data is given, due to the difficult task of setting a sampling rate in a multidimensional space. Another approach is to set a division point in the original data set. The data contained before and after the division point are selected as the learning and the test sets respectively [Paul and Kumar, 2002], [Gonzalez et al., 2003], [Sorjamaa et al., 2007], [Ghaffari et al., 2007]. One of the drawbacks is given by the decision of setting the division point, which is based on the subjective judgment of the researcher.

All these approaches have some common problems:

- The behavior of the estimation method can be affected, leading to gross underestimation of the variance of the estimator and to the wrong conclusion that a learning algorithm is significantly better when it is not, as described in [Nadeau and Bengio, 2003].
- The variability and geometry of the data set is not taken into account, i.e. they are unsupervised procedures, and may produce unrepresentative sets, biasing the estimator [Diamantidis et al., 2000].
- There is no guarantee to obtain a representative and balanced distribution.

Furthermore, when the random approach is followed, an additional problem arises: comparisons between performances of several learning algorithms in different experiments are difficult when randomness is present in the distribution due to the need of using several random splits to get a reliable estimate of the quality of the learning algorithms, which is computationally expensive.

9.2.2 Attempts to build representative and balanced learning and test sets

Due to the problems described above, there is a need to develop new algorithms that take into account the features of the data set with the aim of building representative and balanced groups, reducing the pessimistic effects caused by the removal of instances from the data set, which becomes more important in small data sets, and making reproducible machine learning experiments usually based on random distributions.

In this sense, the majority of works have been developed for classification and little work has been done for function approximation problems.

9.2.2.1 Approaches for classification problems

Zeng et al. [Zeng and Martinez, 2000], proposed a method that provides a balanced intra-class distribution when partitioning a data set into multiple sets using a nearest

neighbor approach but, when the sets are being built, the algorithm only takes into account the last instances introduced in the sets to distribute new ones and ignores all the instances that are already in the sets.

Dupret et al. [Dupret and Koda, 2001], developed an approach based on the well-known bootstrap analysis technique and on the Bayesian optimal classifier methodology, but it is suitable only to binary classification.

Diamantidis et al. [Diamantidis et al., 2000] developed methods that exploit the distribution of instances in the instance space in an unsupervised manner, ignoring the class. They refer to this methods as unsupervised stratification approaches and they were developed to get a distribution of the original set into several subsets. Nevertheless, they can be applied to the problem of distributing an original set into two sets: learning and test. Due to both its originality and its importance in the state-of-the-art, the details of the methodologies are described in the following subsections.

Unsupervised stratification methods In [Diamantidis et al., 2000], the authors mainly developed two methods to partitioning an original set into several subsets taking into account the variability and the geometry of the original data set.

In order to discover regions having similar instances in the instance space, the authors define a *similarity measure* between two input instances (x_i, x_j) with d attributes. This similarity measure is defined as:

$$Sim(x_i, x_j) = -\sqrt{\sum_{l=1}^{d} dist(x_{il}, x_{kl})} \tag{9.3}$$

where $dist(x_{il}, x_{kl})$ is the *Euclidean distance* for continuous attributes. For discrete attributes is defined as:

$$dist(x_{il}, x_{kl}) = \begin{cases} 0, & \text{if } x_{il} = x_{jl} \\ 1, & \text{otherwise} \end{cases} \tag{9.4}$$

A first approach to the deterministic ordering of instances is to order them according to their similarity to the center of the instance space. In this approach the authors assume that instances close to the center of the instance space are themselves close. For discrete attributes, the value with maximum frequency is considered the attribute value of the center. For continuous attributes the center attribute value is the mean of the known attribute values of all instances. This method (Fig. 9.1) is named by the authors as 1C-CV (single center method).

```
Find the Center of the instance space
Sort instances according to their similarity to the center
For each instance x_i (in order)
        p=i mod numSets
        if p=0 then p=numSets
        Assign instance x_i to set p
End For
```

FIGURE 9.1: The 1C-CV method [Diamantidis et al., 2000]. *numSets* is the number of sets in which the original data set is partitioned

The implicit assumption of the 1C-CV approach is that instances having identical similarities to the center are themselves close. Obviously this assumption does not hold in general, especially for datasets with continuous attributes. Thus, the authors developed a method based on clustering analysis that tries to overcome this problem. This clustering-based method, which is called CL-CV by the authors, is combination of a modified version of *K-means* clustering and a distribution of instances using the clusters obtained.

K-means [Duda et al., 2001] clustering partitions the data into a specified number of clusters in an iterative manner. The method is performed in three steps:

1. Select the K initial centers.
2. Assign each instance x_i to the cluster with the closest center.
3. Refine clusters centers. If there are significant changes to the cluster centers, repeat from 2

For the *K-means* clustering method, the authors selected the initial centers (Step 1), according to the procedure shown in Fig.9.2. The aim is the selection of distant instances for the initial centers.

```
/*C_r,C_f,C_h denote cluster centers */
For f=1 to K
        C_f=x_f
End For
For i=K+1 to n
        C_r= the closest center to x_i
        S1=Sim(x_i,C_r)
        /*compute the similarity S2 between the two closest centers */
        S2=max{Sim(C_f,C_h),f=1,...,K,h=1,...,K, with f not equal to h}
        /*find the largest similarity between the center C_r and all
        other centers*/
        S3=max{Sim(C_r,C_f),f=1,...,K, with f not equal to r}
        if (S1<S2) or (S1<S3) then
                C_r=x_i
        End if
End For
```

FIGURE 9.2: Selection of initial centers for K-means clustering in CL-CV method [Diamantidis et al., 2000]. K is the number of clusters and n is the number of input instances ($\{x_i\}_{i=1}^{n}$)

In Step 3 the cluster centers are refined after the end of each repetition. The authors finish the clustering procedure when the largest change in the similarity of cluster centers is lower than 2% of the largest similarity between the cluster centers of the previous iteration.

After the application of the clustering procedure described above, the instance space is partitioned into clusters. The next step performed by the authors is to order the clusters according to the similarity between cluster centers and the center of the first cluster. After the clusters have been ordered, the instances within each cluster are sorted according to their similarity to the cluster center and are distributed in different sets iteratively (see Fig.9.3).

As can be observed, both 1C-CV and CL-CV methods are deterministic approaches.

```
Sort clusters according to their similarity to the first cluster
For each cluster
        Sort the instances of the cluster according to their
        similarity to the cluster center
        For each instance x_i in current cluster
                p=i mod numSets
                if p=0 then p=numSets
                Assign instance x_i to set p
        End For
End For
```

FIGURE 9.3: Ordering instances with clustering in the CL-CV method [Diamantidis et al., 2000]. *numSets* is the number of sets in which the original data set is partitioned and n is the number of input instances ($\{x_i\}_{i=1}^{n}$)

9.2.2.2 Approaches for function approximation problems

As previously said, there are few attemps to the distribution of the original data set into two representative and balanced sets for function approximation tasks. In this sense, the work described by Vasquez et al. [Vasquez and Janaqi, 2001] is emphasized. In that paper, a tabu search algorithm is developed to obtain a homogeneous partition, but this approach has some disadvantages, namely, the tabu list length should be tuned properly for each target problem and thesearch strongly relies on the initial solution.

9.3 A new methodology for generating balanced learning an test sets

As described in the previous section, most of the approaches developed for generating balanced and representative sets from the original data set have been applied to classification problems. However, these methodologies, in principle, are not suitable for function approximation tasks due to the following reasons [Gonzalez et al., 2002]:

1. In classification problems, the output variable takes values in a finite label set which is defined a priori, while in function approximation problems the output variable can take any of the infinite values within an interval of real numbers. In other words, the output variable is discrete for classification and continuous for function approximation problems.

2. In a function approximation problem, output values different from the optimum ones may be accepted if they are "sufficiently close" to them. These variations might only produce a worthless approximation error. This behavior is not desirable for a classifier. In classification problems an output response different from the correct one may be unacceptable.

3. Clustering techniques, such as the *K-means* used in section 9.2.2.1, do not take into account the interpolation properties of the approximator system, since it is not necessary in a classification problem.

Due to both these reasons and the little work made for function approximation problems, it is presented a contribution to the problem of distributing the original data set $S_{total} = \{(x_k, y_k), k = 1, \ldots, n\}$ into two representative and balanced sets for function approximation tasks, such as system identification or time series prediction, to allow both a fair evaluation of learning algorithm's accuracy and to make reproducible machine learning experiments usually based on random distributions. The proposed method, described in [Florido et al., 2011a], takes into account the variability and the geometry of the data set and is based on a combination of a clustering procedure, especially suited for function approximation problems, and a distribution algorithm which distributes the data set into two sets within each cluster based on a nearest-neighbor approach. Furthermore, the proposed approach is deterministic, avoiding the problems described in section 9.2.1 when randomness is present. The main idea of the proposed method is:

1. To split the input/output data pairs into m groups using the *Clustering for Function Approximation* (CFA) approach [Gonzalez et al., 2002] (Section 9.3.1).

2. To apply a distribution algorithm to the input/output data pairs contained in each group obtained in the previous step in order to distribute them into

two subsets per group: learning and test (Section 9.3.2). This algorithm is named *Nearest Neighbor Out* (NNO) since it is based on a nearest neighbor approach.

3. Once the distribution algorithm has been applied to each group, to merge all learning and test subsets obtained for each group into a unique learning set and a unique test set (Section 9.3.3).

Steps (1) to (3), see Fig. 9.4, are repeated varying the number of clusters from $m = 1$ until the stopping criterion is met (Sec.9.3.4). The proposed method is named NNO-CFA, since it is a combination of the CFA clustering approach and the NNO distribution algorithm.

9.3.1 The clustering approach: the CFA algorithm

The first step of the proposed method is to organize the input/output data pairs into groups using a clustering approach. The clustering procedure used in this work is named Clustering for Function Approximation (CFA) algorithm [Gonzalez et al., 2002] and is specially designed for function approximation problems. This approach is more adequate for these kind of problems compared to other input-output clustering techniques and traditional input clustering algorithms (unsupervised clustering) that are oriented toward classification problems owning to the fact that they do not take into account the interpolation properties of the approximator system, since interpolation has no sense in classification problems.

CFA uses the information provided by the target function output in such way that, during the clustering process, the algorithm increases the density of cluster centers in the input areas where the target function presents a more variable response, rather than just in zones where there are more input examples. To carry out this task and being m the number of centers or clusters, the CFA algorithm incorporates a set $O = \{o_i\}_{i=1}^{m}$ that represents an estimation of the output that can be associated to each cluster C_i. The latter is defined as

$$C_i = \{x \in X_{total} / \|x - c_i\| < \|x - c_j\|, \forall j = 1, \ldots, m, j \neq i\} \tag{9.5}$$

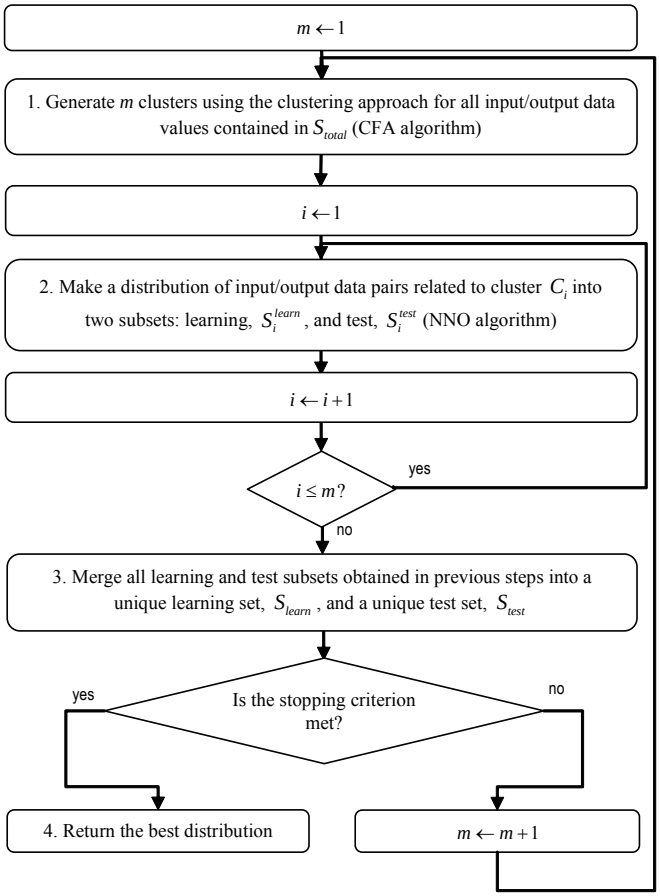


FIGURE 9.4: General description of the proposed NNO-CFA method

where c_i is the center of cluster C_i and X_{total} is the set of all input values of the original data set S_{total}. The value of each o_i is calculated as a weighted average of the output responses of the input data belonging to cluster C_i.

CFA defines a distortion function that has to be minimized in order to converge to a solution:

$$d = \frac{\sum_{i=1}^{m} \sum_{x_j \in C_i} \|x_j - c_i\|^2 \omega_{ji}}{\sum_{i=1}^{m} \sum_{x_j \in C_i} \omega_{ji}} \tag{9.6}$$

where ω_{ji} weights the influence of each input vector in the final position of the ith center. This distortion gives us an idea about the existing controversy between the expected output for the cluster center c_i and the output of the input vector x_j. The

greater the distance between the output of x_j and the estimated output of the cluster C_i it belongs to, the greater the influence of x_j in the final result. Mathematically, ω_{ji} is defined as:

$$\omega_{ji} = \frac{|F(x_j) - o_i|}{max_{k=1}^{n}\{F(x_k)\} - min_{k=1}^{n}\{F(x_k)\}} + \vartheta_{min}, \vartheta > 0 \tag{9.7}$$

The first addend in this expression calculates a normalized distance (in the interval $[0, 1]$) between $F(x_j)$ and o_i, the second addend is a minimum contribution threshold. As ϑ_{min} decreases, CFA forces the cluster centers to concentrate on input zones where the output variability is greater.

The basic organization of the CFA algorithm consists in the iteration of three main steps: partition of the data, the updating of the cluster centers and their estimated outputs and the migration of centers.

The partition is performed as it is done in *K-means* [Duda et al., 2001] thus, a Voronoi partition of the data is obtained. Once the input vectors are partitioned, the cluster centers and their estimated outputs are updated. This process is done iteratively using the equations shown below:

$$\begin{aligned} c_i &= \frac{\sum_{x_j \in C_i} x_j \omega_{ji}}{\sum_{x_j \in C_i} \omega_{ji}} \\ o_i &= \frac{\sum_{x_j \in C_i} F(x_j) \omega_{ji}}{\sum_{x_j \in C_i} \omega_{ji}} \end{aligned} \tag{9.8}$$

To update the cluster centers and their estimated outputs, the algorithm has an internal loop that iterates until the total distortion of the partition is not decreased significantly.

The migration step is introduced in the algorithm to avoid local minima. This step moves cluster centers allocated in input zones where the target function is stable, to zones where the output variability is higher. CFA tries to find an optimal vector quantization where each cluster center makes an equal contribution to the total distortion. This means that the migration step will iterate, moving centers that make a small contribution to the total distortion to areas where centers make a bigger contribution.

Once CFA is applied, a set of disjoint clusters $\{C_i\}_{i=1}^{m}$ are obtained. Taking into account the corresponding output of each input value, the original input/output data set (S_{total}) is then structured into groups.

9.3.2 *The Nearest Neighbor Out* (NNO) distribution algorithm

Once the CFA clustering process has finished, all the input/output data pairs contained in S_{total} are split into groups, that is, all data values contained in each group are related to each other making it easier to local distribute them into two subsets, namely learning and test. The distribution algorithm proposed in this dissertation, named *Nearest Neighbor Out* (NNO) method, is designed for distributing a set of input/output data values into two mutually exclusive subsets according to the Euclidean Distance. It is important to remind that the distribution of a data set into two sets is not trivial, especially for multidimensional data, due to the reasons already mentioned in section 9.2.1, such as setting a sampling rate or a division point in a multidimensional space.

The NNO algorithm (Fig. 9.5) is performed for each group obtained in the previous clustering step (Sec. 9.3.1) and, iteratively, distributes a pair of data values contained in the group into two subsets. The key idea is to distribute into different subsets, namely learning and test, the data values that are themselves close with the aim of building representative subsets. For every cluster C_i obtained by the CFA algorithm, the steps of the NNO algorithm are described as follows.

Stage 1 The first stage consists in selecting the first two input data values contained in the current cluster C_i to start from, according to the following steps (step 1 in Fig.9.5):

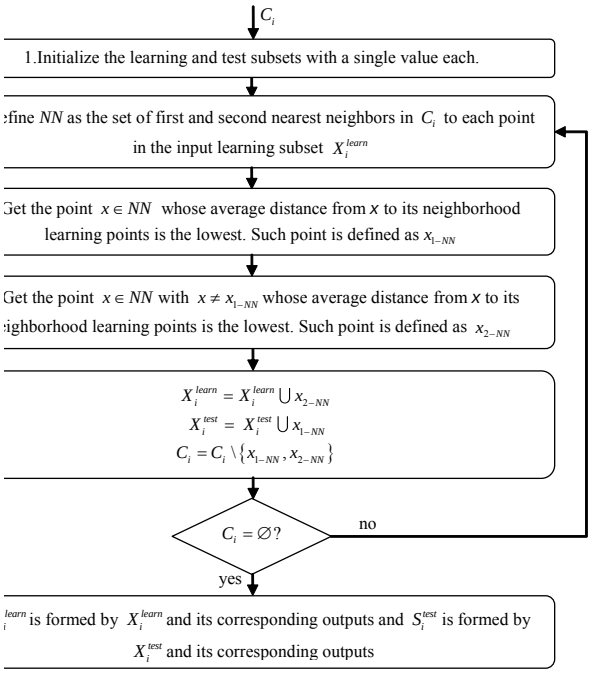


FIGURE 9.5: General description of the proposed NNO distribution algorithm

1. Select the first and the second closest input data values to the cluster center, c_i, and contained in C_i :

$$\begin{aligned} x_a &= \arg\min_{x \in C_i} \{\|c_i - x\|\} \\ x_b &= \arg\min_{\substack{x \in C_i \\ x \neq x_a}} \{\|c_i - x\|\} \end{aligned} \tag{9.9}$$

2. The first and second closest input data values to the cluster center, and their corresponding outputs, are assigned to the learning S_i^{learn} and test S_i^{test} subsets respectively. Such input data values are removed from C_i:

$$\begin{aligned} S_i^{learn} = (x_a, y_a),\ S_i^{test} = (x_b, y_b) \\ C_i = C_i - \{x_a, x_b\} \end{aligned} \tag{9.10}$$

At the end of this step, the learning and test subsets each contain a single input/output data value.

Stage 2 For each iteration of the NNO algorithm, two new input data values are selected from the current cluster C_i for distributing them into the learning and the test subsets (steps 2 to 5 in Fig.9.5) until all data values contained in C_i have been distributed. The aim is to keep the closest points of C_i in different subsets and, for this purpose, the distribution algorithm must take into account **all** input data values present in the learning subset. In order to do so, let us define X_i^{learn} as the set of input values of S_i^{learn}, X_i^{test} as the set of input values of S_i^{test} and the functions that, for a given data $x_j \in X_i^{learn}$, search its first (9.11) and second (9.12) nearest neighbor in cluster C_i:

$$NN_1(x_j, C_i) = \arg\min_{x \in C_i} \{\|x_j - x\|\} \text{ with } x_j \in X_i^{learn} \tag{9.11}$$

$$NN_2(x_j, C_i) = \arg\min_{\substack{x \in C_i \\ x \neq NN_1(x_j, C_i)}} \{\|x_j - x\|\} \text{ with } x_j \in X_i^{learn} \tag{9.12}$$

For each point in the input learning subset X_i^{learn}, its first and second nearest neighbors in C_i are calculated (step 2 in Fig.9.5):

$$NN = \{NN_1(x_j, C_i) \cup NN_2(x_j, C_i)\}, \forall x_j \in X_i^{learn} \tag{9.13}$$

Thus, NN is a subset of values of the cluster set C_i. Notice that there might be elements in NN that are the first and/or the second nearest neighbor to several data values contained in the input learning subset X_i^{learn}. Taking into account this premise, for each point $x \in NN$, the average distance from x to the points $x_j \in X_i^{learn}$ with $NN_1(x_j, C_i)) = x$ or $NN_2(x_j, C_i)) = x$ is measured. The point $x \in NN$ with the lowest average distance will be selected (step 3 in Fig.9.5):

$$x_{1-NN} = \arg\min_{x \in NN} AvgDist(X_i^{learn}, x) \tag{9.14}$$

where

$$AvgDist(X_i^{learn}, x) = \frac{1}{n_{learn}} \sum_{x_j \in X_i^{learn}} \|x_j - NN_k(x_j, C_i)\| \tag{9.15}$$

subject to

$$NN_k(x_j, C_i) = x, \text{ with } k = \{1, 2\} \tag{9.16}$$

From Eq.9.15, calculations are only made on those points in the input learning subset X_i^{learn} whose first ($k = 1$) or second ($k = 2$) nearest neighbor in C_i is x (Eq.9.16). Let us call such points as "neighborhood learning points" of x. Their number is defined as n_{learn} and the average distance from them to x is measured (9.15).

Then, for each point $x \in NN$ and $x \neq x_{1-NN}$, the average distance from x to the points $x_j \in X_i^{learn}$ with $NN_1(x_j, C_i) = x$ or $NN_2(x_j, C_i)) = x$ is measured. The point $x \in NN \backslash \{x_{1-NN}\}$ with the lowest average distance will be selected (step 4 in Fig.9.5) :

$$x_{2-NN} = \underset{\substack{x \in NN \\ x \neq x_{1-NN}}}{\arg\min} AvgDist(X_i^{learn}, x) \tag{9.17}$$

where $AvgDist$ is defined in eq.(9.15) and is subject to Eq. 9.16.

This way and according to the equations (9.14) and (9.17), two new points of C_i and that are the closest ones, in average, to a set of points in the input learning subset X_i^{learn} have been selected. Then, the point selected in (9.14) is assigned to X_i^{test} and the point selected in (9.17) is assigned to X_i^{learn} (step 5 in Fig.9.5):

$$\begin{aligned} X_i^{learn} &= X_i^{learn} \cup \{(x_{2-NN})\} \\ X_i^{test} &= X_i^{test} \cup \{(x_{1-NN})\} \end{aligned} \tag{9.18}$$

where $\{x_{1-NN}, x_{2-NN}\} \in C_i$. These input points are removed from the cluster set C_i:

$$C_i = C_i - \{x_{1-NN}, x_{2-NN}\} \tag{9.19}$$

The key idea of this procedure is to keep close points in the space away from each other and, therefore, build representative learning and test subsets. For this reason, from the set NN of first and second nearest neighbors in C_i (9.13), the point $x_{1-NN} = x \in NN$ with the lowest average distance to a given set of points in the input learning subset X_i^{learn} will be assigned to the input test subset X_i^{test} (9.18). Since this point is placed in an space area which is enough represented by such set of learning points (it is the closest one in average distance), it must be assigned to the input test subset due to the fact that the goal is to build balanced and representative learning and test subsets from C_i. The point $x_{2-NN} = x \in NN$

and $x \neq x_{1-NN}$ with the lowest average distance to a given set of points in the input learning subset will be assigned to the input learning subset (9.18). Equations 9.13-9.19 are repeated until there are no data values to be distributed in C_i. Finally, the learning subset S_i^{learn} is formed by X_i^{learn} and its corresponding outputs and the test subset S_i^{test} is formed by X_i^{test} and its corresponding outputs (step 6 in Fig.9.5). As can be noticed, the subsets are of, approximately,equal size.

Notice that the distribution algorithm is repeated for each cluster C_i obtained in the clustering approach (Sec.9.3.1). The main advantage of distributing the input/output data values within each cluster is that data set's variability is studied locally, which improves the accuracy of the distribution due to the reduction of the input space.

Thus, once the distribution algorithm has been performed for each group, a set of learning and test subsets is arranged:

$$\{S_i^{learn}, S_i^{test}\}, i = 1, \ldots, m \tag{9.20}$$

where m is the number of clusters obtained in the clustering approach.

9.3.3 Merging all learning and test subsets

At this step of the algorithm, a set of learning and test subsets has been built. Each pair of subsets reflects a local representative and balanced distribution of input/output data values. The goal is to get a global balanced distribution and for this purpose, all the learning and test subsets obtained for each group are merged into a unique learning set and a unique test set of, approximately, equal size:

$$S_{learn} = \bigcup_{i=1}^{m} S_i^{learn}; S_{test} = \bigcup_{i=1}^{m} S_i^{test} \tag{9.21}$$

where m is the number of clusters.

9.3.4 Stopping criterion: determining the number of clusters

The NNO-CFA method is a clustering approach, thus, the number of clusters to be used must be chosen. The approach described in [Mojena, 1977] has been followed and it is based on examining the distribution of the *cross-validation error* coefficients (section 9.4.1). This coefficient is one of the quality metrics used in our dissertation to evaluate the partition of the original data set into two sets. A critical value is given by the first stage m in which the obtained coefficient is greater than a value derived from the distribution of coefficients, that is:

$$\alpha_m > \bar{\alpha} + 2\sigma_\alpha \tag{9.22}$$

where α_m is the coefficient (the *cross-validation error*) given when the number of clusters used by the NNO-CFA method is m; $\bar{\alpha}$ and σ_α are the mean and the standard deviation of the $m-1$ coefficients. The value of the coefficient at stage m indicates that the last number of clusters gives a coefficient, α_m, that lies in the upper tail of the distribution and, therefore, it is considered a worse partition of the original data set into two sets with respect to the partitions obtained so far. Thus, when Eq. 9.22 is fulfilled, the best distribution of the original data set into two sets is the one given by the lowest α_k value, that is, the lowest *cross-validation* error in the range $1 \leq k \leq m-1$. Therefore, the number of clusters selected is k.

At the end of the proposed NNO-CFA method, there are two balanced and representative sets from S_{total}, the learning set S_{learn}, which is used for building models that capture the relationships between inputs and outputs, and the test set S_{test}, which is used for checking models' generalization ability with data not used in the learning process. The sets are of, approximately, equal size and are interchangeable, that is, S_{test} can be used as a learning set and S_{learn} can be used as a test set.

9.4 Assessing the quality of the distribution

Assessing the quality of the distribution of the original data set into learning and test sets means to check if the sets are representative and balanced. For this purpose three different criteria have been followed: the *cross-validation error* (section 9.4.1), the *J-divergence* (section 9.4.2) and the *average of generalization errors* (section 9.4.3).

9.4.1 The *cross-validation error*

As stated above, the original data set $S_{total} = \{(x_k, y_k); k = 1, \ldots, n\}$ has been distributed into two sets of roughly equal size. For checking the quality of the distribution, the *cross-validation error* is used and it is obtained in a *2-fold cross-validation* [Kohavi, 1995] procedure according to the following steps:

1. The input/output data values of the first set form the learning set, S_{learn}, and the other set forms the test set, S_{test}.

2. The training of the model F^* is done using S_{learn} and the *Mean Square Error*, $MSE_{test1}(F^*)$, is calculated:

$$MSE_{test1}(F^*) = \frac{1}{n_1} \sum_{(x_i^{test}, y_i^{test}) \in S_{test}} (F^*(x_i^{test}) - y_i^{test})^2 \tag{9.23}$$

   where $(x_i^{test}, y_i^{test}) \in S_{test}$, $F^*(x_i^{test})$ is the approximation of y_i^{test} by model F^* and n_1 is the number of input/output data values contained in the test set S_{test}.

3. The input/output data values of the second set become now the learning set S_{learn} and the first set becomes the test set S_{test}. The step 2 is repeated, obtaining another *Mean Square Error*, $MSE_{test2}(F^*)$.

4. The *mean square cross-validation error* (MSE_{C-V}) is computed according to:

$$MSE_{C-V} = \frac{1}{2}(MSE_{test1}(F^*) + MSE_{test2}(F^*)) \tag{9.24}$$

MSE_{C-V} has the drawback that it is sensitive to changes in the range of data. A more suitable index is the Normalized Root Mean Square cross-validation Error, $NRMSE_{C-V}$, which is defined as:

$$NRMSE_{C-V} = \sqrt{\frac{MSE_{C-V}}{\sigma_y^2}} \tag{9.25}$$

where σ_y^2 is the variance of the output data of the original data set S_{total}.

In the experiments section, $NRMSE_{C-V}$ will be the index selected to measure the cross-validation error.

9.4.2 The *J-divergence*

The *J-divergence* is a symmetrized version of the Kullback-Leibler (K-L) divergence [Lefebvre et al., 2010] and reflects the divergence between two probability distributions. In our case, the *J-divergence* between the histograms of the learning and test sets is used for checking the quality of the distribution of the original input/output data set into two sets.

A fixed hypercube space histogram is defined by dividing the region of interest into a set of equisized hypercubes. Therefore, given a distribution of the original input/output data set into two sets, the main interest is about checking if the number of observations is equally distributed in the same hypercube in the learning and test sets for all hypercubes using histograms. For this purpose, the histograms from the learning and test sets are obtained and the *J-divergence* between them is calculated as the sum of the K-L in both directions:

$$J\text{-}divergence = \sum_{j=1}^{p} H_{learn}(j) \log_2 \frac{H_{learn}(j)}{H_{test}(j)} + \sum_{j=1}^{p} H_{test}(j) \log_2 \frac{H_{test}(j)}{H_{learn}(j)} \tag{9.26}$$

where p is the number of hypercubes and $H_{learn}(j)$ and $H_{test}(j)$ are the fraction of observations falling in the jth hypercube in the learning and test sets respectively.

From Eq.9.26, when *J-divergence=0*, the number of observations is equally distributed between the sets in the same hypercube for all hyper cubes, that is, the

learning and test sets are balanced in terms of the number of observations, which means that the histograms are equal. The bigger the *J-divergence*, the worse the distribution of the number of observations between the sets is.

It is important to notice that histograms are a valuable tool for measuring the number of observations between the same hypercube in the learning and test sets for all hyper cubes throughout the input domain. However, they are not suitable for building two sets from the original input/output data set in function approximation problems, since they do not take into account, in the distribution process, either the output data or the possible presence of noise in the data, contrary to the NNO-CFA algorithm.

9.4.3 The *average of generalization errors*

It is also interesting to observe how the models trained with the learning and test sets perform with data not used in the distribution procedure. For this purpose, the generalization ability of the learning and test sets through a huge generalization data set, S_{gen} is measured:

1. Two models F^*_{learn} and F^*_{test} are trained using S_{learn} and S_{test} sets respectively. The Mean Square Errors, $MSE_{gen1}(F^*_{learn})$ and $MSE_{gen2}(F^*_{test})$, are calculated:

$$\begin{aligned} MSE_{gen1}(F^*_{learn}) &= \frac{1}{n_{gen}} \sum_{(x_i^{gen}, y_i^{gen}) \in S_{gen}} (F^*_{learn}(x_i^{gen}) - y_i^{gen})^2 \\ MSE_{gen2}(F^*_{test}) &= \frac{1}{n_{gen}} \sum_{(x_i^{gen}, y_i^{gen}) \in S_{gen}} (F^*_{test}(x_i^{gen}) - y_i^{gen})^2 \end{aligned} \tag{9.27}$$

   where $(x_i^{gen}, y_i^{gen}) \in S_{gen}$, $F^*_{learn}(x_i^{gen})$ and $F^*_{test}(x_i^{gen})$ are the approximation of y_i^{gen} by models F^*_{learn} and F^*_{test} respectively and n_{gen} is the number of input/output data values contained in the generalization set S_{gen}.

2. A normalized version of the errors obtained in the previous step are obtained (Normalized Root Mean Square Errors, $NRMSE_{gen1}$ and $NRMSE_{gen2}$):

$$\begin{aligned} NRMSE_{gen1} &= \sqrt{\frac{MSE_{gen1}}{\sigma_y^2}} \\ NRMSE_{gen2} &= \sqrt{\frac{MSE_{gen2}}{\sigma_y^2}} \end{aligned} \tag{9.28}$$

3. The average of $NRMSE_{gen1}$ and $NRMSE_{gen2}$ is defined as:

$$NRMSE_{gen} = \frac{1}{2}(NRMSE_{gen1} + NRMSE_{gen2}) \tag{9.29}$$

$NRMSE_{gen}$ is the index selected for measuring the average of generalization errors in the experiments section.

It is important to notice the differences between the *NRMSE* values, in the form of *2-fold cross-validation error* ($NRMSE_{C-V}$), the *average of generalization errors* ($NRMSE_{gen}$) and the *J-divergence*. The latter only measures how the number of observations has been distributed between the same hypercube in the sets for all hyper cubes throughout the input domain, whereas the $NRMSE_{C-V}$ and $NRMSE_{gen}$ values reflect a learning/ testing procedure and takes into account the output value. Thus, for a given hypercube, it is more important how representative the observations are than their number, although the latter is also important. In any case, both ways for checking the quality of the data set distribution are useful and complementary.

9.5 Experimental results

In order to show the performance of the proposed method, several different data sets have been used (subsection 9.5.1). The results of the proposed method compared to other present in the literature are reported in subsection 9.5.2 through a statistical analysis of variance.

9.5.1 Data sets

In order to make a thorough comparative study, several different examples have been selected to cover many possible practical function approximation situations.

9.5.1.1 Nonlinear chaotic time series

These time series present some chaotic behaviour in order to be a non trivial approximation problem.

The Hénon map The Hénon map [Hènon, 1976] runs through the plane following equation 9.30. When the parameters are set to the values $a = 1.4$ and $b = 0.3$, the result is called the Canonical Hénon map and exhibits chaotic behaviour. The values $\{y_i\}_1^n$ of the Canonical Hénon map were taken as the time series data (see Fig. 9.6a).

$$\begin{aligned} x_{t+1} &= y_t - ax_t^2 \\ y_{t+1} &= bx_t \end{aligned} \tag{9.30}$$

The Logistic map The logistic map was a demographic model, that has been popularized by May [May, 1976] as an example of simple nonlinear system that exhibits complex, chaotic behaviour (Fig. 9.6b). It is drawn from the following equation:

$$y_t = 4y_{t-1}(1 - y_{t-1}) \tag{9.31}$$

Mackey-Glass time series The Mackey-Glass time series [Mackey and Glass, 1977] is approximated from the differential equation 9.32 and models the dynamics of white blood cell production in the human body. It is a widely used benchmark for generalization abilities of time series prediction methods. When $a = 0.25$, $b = 0.1$ and $\tau > 17$ the series is chaotic (Fig.9.6c). The series is continuous and it is obtained by integrating 9.32 with a numerical integration method such as fourth

order Runge-Kutta method.

$$\frac{dx(t)}{dt} = \frac{ax(t-\tau)}{1+x^{10}(t-\tau)} - bx(t) \tag{9.32}$$

For Mackey-Glass, the public accessible data given in MATLAB^{TM}software [MATLAB, 2010c] has been used (*mgdata.dat*). This time series consists of 1200 samples from which the first 200 are discarded due to their random nature, as it is usual in the literature.

9.5.1.2 Real time series prediction

These problems are difficult to model since they are normally noisy, non-linear and stochastic. The one chosen for this work is *dailysap* which is the Daily Standard & Poor's index (S&P) of stocks from January 1st, 1980 to October 8th, 1992 (Fig.9.6d) [Korsan, 1993].

9.5.1.3 Artificial function approximation problems

There have been created two artificial functions with different sizes and number of inputs. In function f_1 [Cherkassky et al., 1996] x_1 and x_2 are uniformly generated in the range]-2,2[(Fig.9.7a) and in function f_2 [Pomares, 2000] x is uniformly sampled in the range]0,1[(Fig.9.7b):

$$\begin{aligned} f_1(x_1, x_2) &= sin(x_1 \cdot x_2) \\ f_2(x) &= exp(-5x) \cdot sin(2\pi x) \end{aligned} \tag{9.33}$$

9.5.1.4 Nonlinear dynamic systems

The problem selected as dynamic system is the one described in [Narendra and Parthasarathy, 1990]. It is given by Eq.9.34 where the output y non-linearly depends on both its two previous values and the control variable u, which is uniformly selected in the range $[-2, 2]$ and represents the actuator's output (Fig.9.7c).

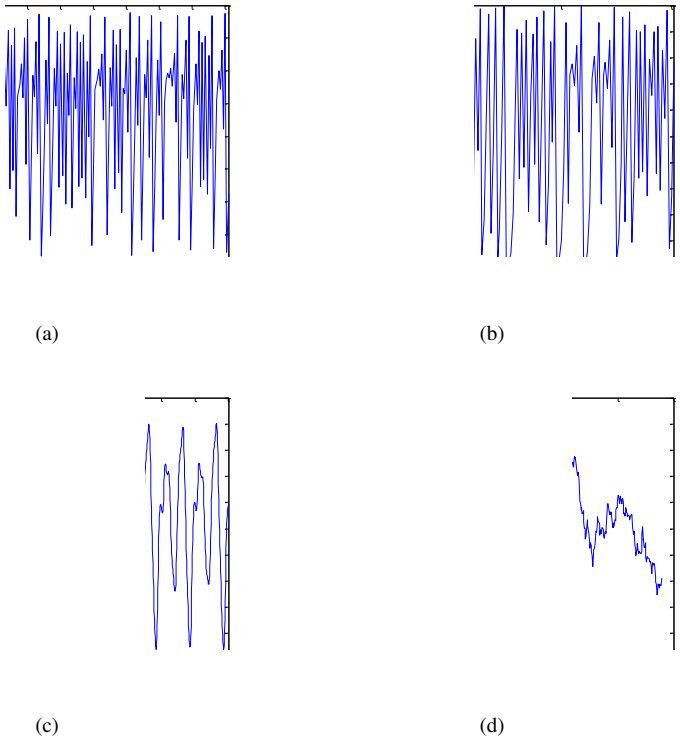
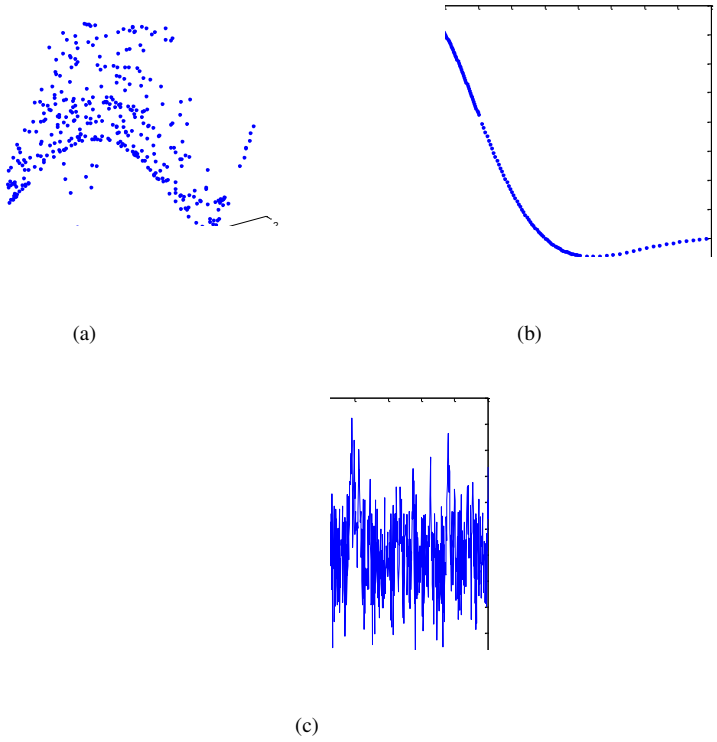


FIGURE 9.6: Nonlinear chaotic and real time series used in the experiments. (a) Hénon map; (b) Logistic Map; (c) Mackey-Glass; (d) Daily Standard & Poor's index (S & P)

The equilibrium state of the unforced system is reached when $y = 0$ and $u = 0$.

$$y(n) = \frac{y(n-1) \cdot y(n-2) \cdot (y(n-1) + 2.5)}{1 + y^2(n-1) + y^2(n-2)} + u(n-1) \tag{9.34}$$

For each example, the model that describes the input-output relationship is shown in Table 9.1 as well as the size of the original data set S_{total} and the size of the generalization set S_{gen}. Notice that a generalization set has only been generated for those data sets whose function is available to generate as many data as desired. Standard & Poor's time series is limited from 1980 to 1992. With regard to Mackey-Glass time series, a set of generalization data can not be generated since the time series used is the one given in MATLAB software.



(a) (b)

(c)

FIGURE 9.7: Artificial function approximation problems and nonlinear dynamic systems used in the experiments. (a) f_1; (b) f_2; (c) Nonlinear dynamic system

For the Hénon map, Logistic map and Mackey-Glass time series, the inputs selected are the ones commonly used in the literature [Jayawardena et al., 2006], [Yao and Liu, 1997], [Chen et al., 2005]. For f_1 and f_2 artificial functions and for the nonlinear dynamic system, the inputs chosen were the ones used in [Cherkassky et al., 1996], [Pomares, 2000] and [Narendra and Parthasarathy, 1990] respectively. For S&P time series, the inputs chosen are those that minimize the statistical independence among variables and they were selected using the approach proposed in [Herrera et al., 2006].

TABLE 9.1: Data set models and their size

Data set	**Model**	$\|S_{total}\|$	$\|S_{gen}\|$
Hénon map	$y_t = F(y_{t-1}, y_{t-2})$	200	4000
Logistic map	$y_t = F(y_{t-1})$	150	10000
Mackey Glass	$y_t = F(y_{t-6}, y_{t-12}, y_{t-18}, y_{t-24})$	976	$-^a$
S&P	$y_t = F(t_{t-2}, y_{t-36}, y_{t-65}, y_{t-79})$	500	$-^a$
f_1	$y_t = F(x_{1t}, x_{2t})$	400	4000
f_2	$y_t = F(x_t)$	198	10000
Dynamic system	$y_t = F(y_{t-1}, y_{t-2}, u_{t-1})$	1000	1500

[a] The number of data is limited

9.5.2 Results

In this section, we are going to evaluate the quality of the distribution of the original data set into two sets given by the proposed method (NNO-CFA), the random approach and the deterministic *single center* and *k-means clustering* methods which have been described in section 9.2.2.1. They are referred by the authors as 1C-CV and CL-CV respectively and have been implemented for function approximation tasks for comparison purposes in our work. The quality will be measured through the quality metrics described in Sec.9.4. Least-squares-support vector machines (LS-SVMs), which are widely used in time series prediction problems [Herrera et al., 2006], are chosen as a paradigm for building the validation models. We have made use of the LS-SVM implementation in MATLAB, LS-SVMLab [Pelckmans et al., 2002] to evaluate the performance of the application of the LS-SVM model to the examples present in this section. The values of the hyperparameters, σ (the width of the kernel) and γ (the regularization parameter) are set according to [Cherkassky and Ma, 2004].

The single center method 1C-CV is not a clustering procedure. To make a comparison when no clustering is considered, the NNO distribution algorithm was run alone with a small variation: the starting point from which the data is distributed is changed to the input space center of the data set (see stage 1, Sec.9.3.2) since the CFA algorithm is not applied and, therefore, there is no cluster center. This variation is referred in the results as the NNO method. Therefore, five algorithms will be compared: the random algorithm, which has been executed over 20 runs, the NNO

and 1C-CV non-clustering algorithms and the NNO-CFA and CL-CV clustering algorithms. For the CL-CV method, the same approach described in Sec.9.3.4 for selecting the number of clusters has been followed.

For the evaluation of our proposed methodologies, it would be interesting to test them through different fractions of the original size of the examples. For this purpose, the sample size of every data set will be classified into three groups: small, medium and big:

$$group \leftarrow \begin{cases} small & \text{if } size \leq 60 \\ medium & \text{if } 60 > size \leq 100 \\ big & \text{if } size > 100 \end{cases} \tag{9.35}$$

where

$$size = \frac{fraction \cdot |S_{total}|}{d} \tag{9.36}$$

$|S_{total}|$ is the original size (table 9.1), d is the number of inputs for each example (column *Model* in table 9.1) and $fraction = 0.3, 0.6, 0.9$ is the sample size fraction of $|S_{total}|$.

Therefore, it will be checked whether the results (the response of the system) are affected by the algorithm (NNO, NNO-CFA, 1C-CV, CL-CV and RANDOM), the example (Hénon map, Logistic map, Mackey-Glass, S&P, f_1, f_2, and Dynamic System) and the sample size factors (small, medium and big). For this purpose, the ANalysis Of VAriance (ANOVA) test [Box et al., 1978] is used, since it is one of the most widely statistical techniques. The statistical parameter considered in this test is the significant level and if it is lower than 0.05, then the corresponding levels of the factor are statistically significant with a confidence level of 95%. For each example and each different fraction value, each algorithm has been executed 10 times. It must be noticed that each run is based on a different random selection of a fraction of the original samples for a given example. This way, the results obtained will be representative in terms of the samples selected for each example and fraction. For instance, for the Mackey-Glass time series and a fraction of 90%, each run takes randomly $976 \cdot 0.9 = 878$ samples from the original set. The same

selection of samples is then used by all the algorithms with the aim of providing a fair comparison among them.

The response variables to be used in the ANOVA test are the *cross-validation error* ($NRMSE_{C-V}$), which measures the quality of the distribution of the data set into two sets using the *2-fold cross-validation procedure* (Sec.9.4.1), the *J-divergence*, which measures how the number of observations has been distributed between the same hypercube in the sets for all hyper cubes throughout the input domain (Sec.9.4.2) and the *average of generalization errors* ($NRMSE_{gen}$) which measures the generalization ability of the learning and test sets through a huge generalization data set S_{gen} (Sec.9.4.3). The normality, independence of populations and homocedasticity assumptions for ANOVA test are accomplished and the values of the dependent variables were normalized to $[0, 1]$ to allow a fair comparison among the values of the different factors.

According to the $NRMSE_{C-V}$ response variable (Table 9.2), the algorithm and the example factors present the greatest statistical relevance, which means that the result statistically depends on the algorithm chosen (NNO-CFA, NNO, CL-CV, 1C-CV and RANDOM) and the example (data set), which is obvious due to the different coverage of examples. The sample size factor is not statistically significant, which means that the results do not statistically depend on the sample size chosen. Figure 9.8 shows that the methods proposed here, NNO and NNO-CFA, outperforms statistically better ($P < 0.05$) than the others. Among the former, NNO-CFA gets statistically the best results ($P < 0.05$) and the random algorithm throws the worst $NRMSE_{C-V}$ value.

TABLE 9.2: ANOVA table for the analysis of the main factors for the *cross-validation* error, $NRMSE_{C-V}$ response variable

Main factors	Sum of squares	D.F.	Mean square	F-Ratio	Sig.level
Algorithm	19.22	4	4.80	107.55	0
Example	3.35	6	0.56	12.49	0
Sample size	0.09	2	0.04	1.00	0.37

According to the *J-divergence* response variable (Table 9.3), the same conclusions

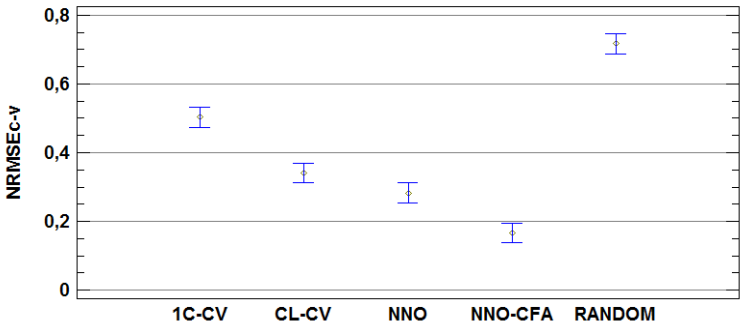


FIGURE 9.8: Means and 95% Least Significant Differences (LSD) intervals of the different algorithms through the *cross-validation error*, $NRMSE_{C-V}$ response variable

are obtained: the algorithm and the example factors present the greatest statistical relevance and the sample size factor is not statistically significant. NNO-CFA method shows statistically the best results ($P < 0.05$) than the remaining ones as can be observed in Fig.9.9. NNO method also outperforms statistically better ($P < 0.05$) than the other methodologies but NNO-CFA.

TABLE 9.3: ANOVA table for the analysis of the main factors for the *J-divergence* response variable

Main factors	Sum of squares	D.F.	Mean square	F-Ratio	Sig.level
Algorithm	25.37	4	6.34	130.25	0
Example	1.96	6	0.33	6.71	0
Sample size	0.09	2	0.05	1.01	0.36

Related to the *average of generalization errors* ($NRMSE_{gen}$), the algorithm has the greatest statistical relevance and the example and sample size factors are not statistically significant (Table 9.4). On the other hand there is no statistical difference ($P > 0.05$) among NNO-CFA, NNO and CL-CV methods, although our proposed methodologies (NNO-CFA and NNO) perform better on average (Fig.9.10) and thus improve the generalization ability of the models. It must be noticed, that both NNO and NNO-CFA outperforms statistically better ($P < 0.05$) than 1C-CV and Random approaches.

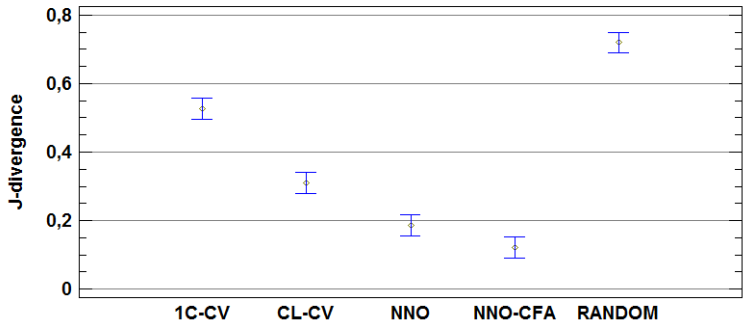


FIGURE 9.9: Means and 95% Least Significant Differences (LSD) intervals of the different algorithms through the *J-divergence* response variable

TABLE 9.4: ANOVA table for the analysis of the main variables for the $NRMSE_{gen}$ response variable

Main factors	Sum of squares	D.F.	Mean square	F-Ratio	Sig.level
Algorithm	1.61	4	0.4	4.81	0
Example	0.39	4	0.09	1.19	0.32
Sample size	0.17	2	0.09	1.06	0.35

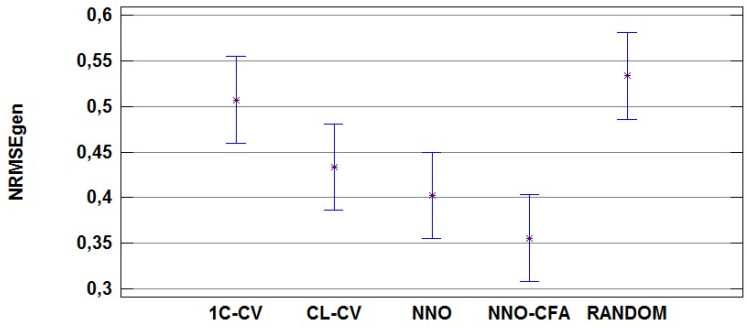


FIGURE 9.10: Means and 95% Least Significant Differences (LSD) intervals of the different algorithms through the *Average of Generalization Errors*, $NRMSE_{gen}$ response variable

On the other hand, it is also useful to check the number of clusters selected by NNO-CFA and CL-CV methods. It can be observed from Fig. 9.11 that, in most of the cases, the number of clusters used by the NNO-CFA method is lower than the ones used by CL-CV. This means that NNO-CFA methodology needs less number of clusters to achieve better results, which decreases the complexity of the model.

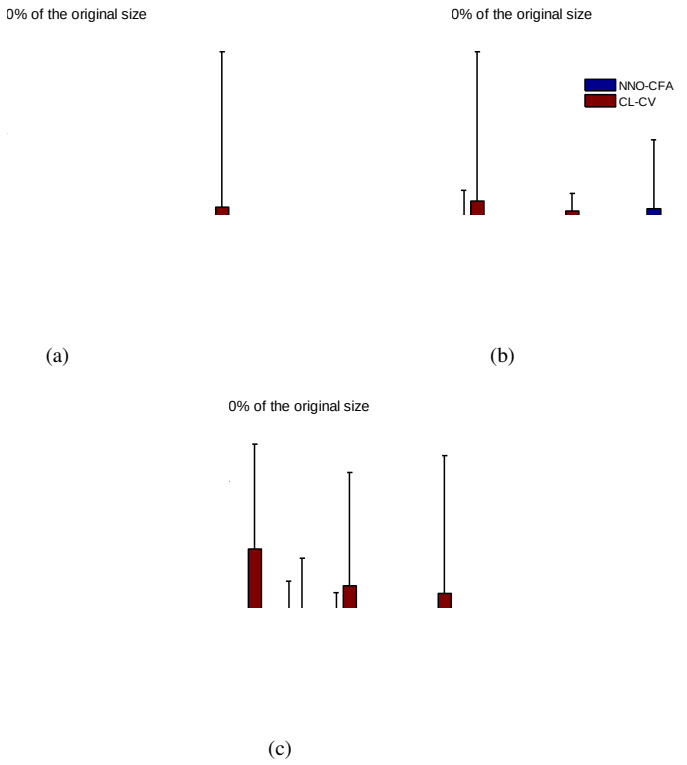


FIGURE 9.11: Mean and standard deviation number of clusters for the clustering methods CL-CV and NNO-CFA for the fraction of the original size of a) 90%, b) 60%, c) 30%

With respect to the time complexity of the methodologies described in this work, Table 9.5 shows the time in seconds for each algorithm and each fraction of the original size: 30%, 60% and 90%. Random method is not shown because the distribution into two sets is negligible in time. From Table 9.5, it can be observed that although our methods, NNO and NNO-CFA, are more computationally expensive, the time employed is not so high (around 75 seconds for NNO and 24 seconds for NNO-CFA in the Dynamic System Example when the fraction of the original size

is 90%), and it is worth executing them because they produce more representative and balanced sets according to the results given before.

TABLE 9.5: Mean and standard deviation time in seconds for the NNO, 1C-CV, CL-CV and NNO-CFA methods depending on the fraction of the original size: 30%, 60% and 90%

Data set	**Algorithm**	**Fraction of the original size**		
		30%	**60%**	**90%**
Hénon map	NNO	0.08 ± 0.01	0.25 ± 0.02	0.48 ± 0.03
	1C-CV	0.02 ± 0.01	0.02 ± 0.01	0.05 ± 0.01
	NNO-CFA	0.12 ± 0.06	0.24 ± 0.03	0.56 ± 0.15
	CL-CV	0.05 ± 0.05	0.19 ± 0.20	0.18 ± 0.25
Logistic map	NNO	0.04 ± 0.01	0.09 ± 0.01	0.16 ± 0.01
	1C-CV	0.01 ± 0.01	0.01 ± 0.01	0.03 ± 0.01
	NNO-CFA	0.07 ± 0.04	0.20 ± 0.22	0.25 ± 0.13
	CL-CV	0.02 ± 0.02	0.04 ± 0.01	0.04 ± 0.01
Mackey-Glass	NNO	1.86 ± 0.17	11.31 ± 0.65	36.35 ± 3.39
	1C-CV	0.13 ± 0.01	0.55 ± 0.01	1.40 ± 0.01
	NNO-CFA	1.80 ± 0.74	10.83 ± 1.30	17.02 ± 9.14
	CL-CV	0.56 ± 0.39	1.73 ± 2.43	1.97 ± 0.74
S&P	NNO	0.49 ± 0.03	1.81 ± 0.09	5.25 ± 0.39
	1C-CV	0.04 ± 0.01	0.13 ± 0.01	0.30 ± 0.01
	NNO-CFA	0.92 ± 0.58	2.09 ± 0.90	5.55 ± 2.09
	CL-CV	0.11 ± 0.14	0.17 ± 0.07	0.55 ± 0.31
f_1	NNO	0.39 ± 0.03	1.62 ± 0.04	4.63 ± 0.42
	1C-CV	0.03 ± 0.01	0.08 ± 0.01	0.20 ± 0.01
	NNO-CFA	0.58 ± 0.44	1.84 ± 0.49	3.72 ± 1.40
	CL-CV	0.14 ± 0.15	0.11 ± 0.05	0.25 ± 0.05
f_2	NNO	0.06 ± 0.01	0.12 ± 0.02	0.23 ± 0.01
	1C-CV	0.01 ± 0.01	0.03 ± 0.01	0.03 ± 0.01
	NNO-CFA	0.10 ± 0.06	0.19 ± 0.09	0.40 ± 0.34
	CL-CV	0.02 ± 0.01	0.03 ± 0.02	0.04 ± 0.01
Dynamic system	NNO	4.42 ± 0.13	27.27 ± 1.94	74.78 ± 2.56
	1C-CV	0.11 ± 0.01	0.47 ± 0.01	1.18 ± 0.01
	NNO-CFA	3.99 ± 1.02	14.80 ± 8.44	23.47 ± 10.37
	CL-CV	0.47 ± 0.70	0.56 ± 0.16	10.10 ± 19.32

Finally, it must be emphasized that, although the ANOVA test evaluates the results in terms of the different factors involved in the experiments (algorithms, examples and sample size), for brevity purposes, only figures related to the algorithm factor are shown, because they reflect the differences between the methodologies compared.

9.6 Conclusions and future work

In this chapter, we have proposed a deterministic approach for a distribution of the original input/output data set into two representative and balanced sets of roughly equal size, namely learning and test sets, based on a combination of a clustering approach and a nearest-neighbor-based distribution procedure. The sets obtained result in a reduction of the pessimistic effects caused by the removal of instances from the original data set whose importance is evident when evaluating function approximation models, especially in small data sets. The goal is to allow both a fair evaluation of learning's accuracy and to make reproducible machine learning experiments usually based on random distributions.

The Analysis of Variance (ANOVA) statistical test has examined the effects of several factors on three quantitative responses, the *cross-validation error* ($NRMSE_{C-V}$), the *J-divergence* and the *average of generalization errors* ($NRMSE_{gen}$), and showed the superiority of the NNO-CFA algorithm over the other methodologies (NNO, 1C-CV, CL-CV and RANDOM) regardless of the examples and sample size used due to the following reasons:

- It is based on a clustering procedure, the CFA, which is specially suited and adequate for function approximation problems due to the analysis of the output variability of the target function and, thus, its adaptation to the instance space with singularities during the clustering process.
- The distribution of the data values is made within clusters using the NNO approach, taking into account all data values contained in a cluster. This fact allows the study of the variability and the geometry of the data set locally,which produces a reduction of the input space and, therefore, an improvement of the distribution's quality.

The results have also demonstrated that the most common approach used in the literature for distributing the original data set into two groups, the random algorithm, should not be recommended for this kind of problems due to the following reasons:

- It may lead to wrong conclusions about a learning algorithm due to both its random nature and the distribution itself, since the variability of the data set is not taken into account and may produce unrepresentative sets.
- Comparisons between performances of several learning algorithms in different experiments are difficult due to the need of using several random splits to get a reliable estimate of the quality of the learning algorithms. This is computationally expensive because it involves several repetitions of the random splitting and learning process.

The method proposed in this chapter is not only useful for the distribution of the original data set into learning and test sets, but also for the distribution of the learning set into training and validation sets in the context of the selection of the best model structure for a given problem, which is known as model selection. Such model selection approaches could be a valid alternative to those based on *k-fold cross-validation* strategies, which are random-based and can be computationally expensive when the complexity of the learning algorithm is high.

The NNO-CFA method is freely available, together with the examples used in this dissertation, at the website http://atc.ugr.es/~hector/NNO-CFA/index.html.

Chapter 10

Model selection for RBFNNs in time series forecasting

10.1 Motivation and goals

As previously stated, most of the learning algorithms proposed in the literature split the original input/output data set into two groups: learning and test. The learning set is used for building models that capture the relationships between inputs and outputs and the test set is used for checking model's generalization ability.

The learning set can be, in turn, split into two sets: training, used for parameter estimation for a given model structure, and validation, used for evaluating the trained model, obtaining a validation error. Model selection is, therefore, related to the task of comparing several model structures (for example, different number of neurons in the hidden layer of multilayer perceptrons) based on estimations of their validation errors in order to select the most suitable model structure for a given problem.

One of the most widely used model selection approaches in the literature are those based on the *K-fold cross-validation* [Lendasse et al., 2003],[Kohavi, 1995] model evaluation strategy. However, such model selection approaches have some drawbacks, such as its random nature (it does not take into account the variability and

geometry of the learning data when building the training and validation sets) and the subjective decision for a proper value of K, resulting in large bias for low values and high variance and computational cost for high values [Kohavi, 1995]. So, it is desirable that a model selection approach is based on a model evaluation strategy with the following features: (1) low variance and bias, (2) no randomness, (3) low computational cost and (4) use of balanced and representative training and validation sets.

Thus, it is expected that the use of two balanced training and validation sets obtained by the data distribution methodology proposed in section 9.3.2 and integrated in a new model selection approach will prevent the problems appeared when model selection methodologies based on *K-fold cross-validation* are used.

In this sense, the balanced and representative training and validation sets will be applied in the context of a new deterministic model selection methodology for incremental Radial Basis Function Neural Network (RBFNN) construction in time series prediction problems. Such model selection approach is a combined algorithm which takes advantage of such balanced and representative training and validation sets for their use in RBFNN initialization, optimization and network model evaluation. This way, the model prediction accuracy is improved, getting small variance and bias, reducing the computation time spent in selecting the model and avoiding random and computationally expensive model selection methodologies based on *K-fold cross-validation* procedures.

This chapter is structured as follows. Section 10.2 briefly introduces RBFNNs. Section 10.3 reviews the common approaches that search for the most suitable RBFNN structure as well as the model evaluation methodologies existing in the literature. Section 10.4 presents the model selection proposed in this chapter. In section 10.5, an experimental comparison of the new algorithm with respect to *K-fold cross-validation*-based model selection methods in three well-known time series prediction benchmarks is reported. An ANOVA-based statistical study of the results and a comparison of the RBFNN model selected to other series prediction methodologies are also presented. Finally, some conclusions and future work are drawn in section 10.6.

10.2 Radial Basis Function Neural Networks, RBFNNs

RBFNNs [Park and Sandberg, 1991] are one of the most widely applied ANNs in time series forecasting tasks in recent years [Qi and Zhang, 2001], [Du and Zhang, 2008], due to their simple architecture and learning scheme and the possibility of incorporating the qualitative aspects of human experience in the model selection and training [Du and Zhang, 2008].

RBFNNs are Neural Networks composed of two layers: the hidden layer, which is at the same time composed of m radial functions, and an output layer, which performs a weighted addition of neuron's activation in the previous layer. An RBFNN $\mathcal{F}$ can approximate an unknown function F with n inputs and one output from a set of input/output data values $S_{total} = \{(x_k, y_k); k = 1, ..., n\}$ with $x_k \in \mathbb{R}^d$ and $y_k = F(x_k) \in \mathbb{R}$. RBFNNs are universal approximators, functionally equivalent to a nonlinear regression model and have a set of numerous parameters that have to be optimized: the centers, the radii and the weights of the RBFs Guillén et al. [2008]. They are defined as:

$$\mathcal{F}(x_k; C, R, \Omega) = \sum_{i=1}^{m} \phi(x_k; c_i, r_i) \cdot \Omega_i \tag{10.1}$$

where $C = \{c_i\}_{i=1}^m$, $c_i \in \mathbb{R}^d$, is the set of RBF centers, $R = \{r_i\}_{i=1}^m$, $r_i \in \mathbb{R}$, is the set of values for each RBF radius, $\Omega = \{\Omega_i\}_{i=1}^m$, $\Omega_i \in \mathbb{R}$, is the set of weights and $\phi(x_k; c_i, r_i)$ represents an RBF. The most commonly used RBF is the Gaussian function because it is continuous, differentiable, it provides a smoother output and improves the interpolation capabilities:

$$\phi(x_k; c_i, r_i) = exp\left(\frac{||x_k - c_i||^2}{r_i^2}\right) \tag{10.2}$$

One of the most common procedures to design an RBFNN for functional approximation problems, such as time series prediction, is shown below:

1. Initialize RBF centers C
2. Initialize the radius R for each RBF

3. Calculate the optimum value for the weights Ω
4. Apply local search algorithm to adjust centers and radii

10.2.1 Centers initialization

The initialization of the centers is very important because, if an incorrect initialization of the centers is performed, the approximation error could be increased. The centers can be initialized using several approaches specially designed for function approximation problems such as the *clustering for function approximation* (CFA) [Gonzalez et al., 2002] or the *improved clustering for function approximation* (ICFA) [Guillén et al., 2007]. The latter introduces a fuzzy partition of the data and analyzes the output variability of the target function during the clustering process and augments the number of centers in those input zones where the target function is more variable, increasing the variance explained by the approximator. Therefore, the ICFA uses the information provided by the function output in order to make a better placement of the centers of the RBFs. This change in the behavior of the clustering algorithm improves the performance of the RBFNN, compared to other models derived from traditional classification oriented clustering algorithms [Guillén et al., 2008].

10.2.2 Radii initialization

When initializing the radii, two choices are possible. One is that all radii have the same value. In [Park and Sandberg, 1991], the authors demonstrated that if all the radii have the same value, the network can still be a universal approximator, reducing the number of non-linear parameters of the model and, thus, simplifying the network training. The other choice is that each center can define its own value for the radius. This way, the performance of the network can be increased [Benoudjit and Verleysen, 2003] but at the expense of increasing the complexity of the training.

10.2.3 Weights initialization

Since the output of the RBFNN is linear with respect to the weights (see Eq. 10.1), it is possible to optimally obtain these parameters through a linear equation system that can be robustly solved using the Singular Value Decomposition (SVD), the Cholesky decomposition or the orthogonal least squares (OLS) method [Gonzalez et al., 2003].

10.2.4 Apply local search algorithm to adjust centers and radii

A local search training algorithm can be used to set the parameters concerning each RBF (its center and radius) in order to minimize an error criterion. In the literature, several training methods to identify these parameters have been published. The most popularly used training method is the backpropagation algorithm which is essentially a gradient steepest descent method. However, it suffers the problems of slow convergence, inefficiency, and lack of robustness and can be very sensitive to the choice of the learning rate [Zou et al., 2007]. In light of the weakness of the conventional backpropagation algorithm, a number of variations or modifications of backpropagation, such as the adaptive method [Jacobs, 1987], quickprop [Fahlman, 1988] and second-order methods [Parker, 1987] have been proposed. Among them, the second-order methods (such as BFGS and Levenberg−Marquardt methods) are more efficient nonlinear optimization methods and are used in most optimization packages. Their faster convergence, robustness, and the ability to find good local minima make them attractive in ANN training.

10.3 Searching for the most suitable RBFNN structure: model selection

In spite of the advantages of RBFNNs previously described, one of the critical issues of RBFNNs is the selection of the best network structure for a forecasting task, due to the numerous parameters to be estimated and optimized. A network smaller than the optimal architecture underfits and fails to learn the data well (bias

is high and variance is low) and a large network suffers the overfitting problem, resulting in poor generalization (bias is low and variance is high). Thus, the optimal architecture is the one with low bias and low variance so that the network learns the function underlying the data and not the existing noise [Aran et al., 2009].

Common approaches that search for the RBFNN structure are the well known greedy techniques (*growing/constructive/incremental* and *pruning/destructive/ decremental*) and genetic-algorithm-based evolutionary algorithms. Incremental techniques start with a small network, adding additional hidden units and weights until a satisfactory solution is found [Kaminski and Strumillo, 1997], [Gomm and Yu, 2000]. On the other hand, pruning approaches use a larger network at the beginning which shrinks during learning [Paetz, 2004]. There are also a combination of the two greedy strategies: incremental and pruning [Alpaydin, 1994]. The incremental approach is generally preferred over the pruning approach [Kwok and Yeung, 1997]: it is straightforward to specify an initial network and it is less computationally expensive since small network solutions are searched first. Nevertheless, one of their drawbacks is to determine when to stop the addition of hidden units.

Genetic-algorithm-based evolutionary algorithms [Leung et al., 2002] consider the process of selecting the RBFNN structure as a search problem within all possible network architectures. However, such methods are usually quite demanding both in time and memory requirements and a good representation of the network structure and good design of the genetic operators are required [Kwok and Yeung, 1997].

Regardless of the approach chosen, when the network structure changes, an evaluation is needed to check the suitability of the new network structure/model [Aran et al., 2009]. For this purpose, there are information theory-based model evaluation methodologies such as the Akaike's Information Criterion (AIC) [Akaike, 1973], the Bayesian Information Criterion (BIC) [Schwarz, 1978] and the Minimum description length (MDL) [Rissanen, 1978]. These approaches are not quite useful in neural network time series forecasting because they are originally derived for traditional statistical models where the number of parameters is usually small [Qi and Zhang, 2001].

On the other hand, there are other methodologies to evaluate a given network model, such as those that split the available data into two groups: learning, used for

ANN model building, and test, for out-of-sample evaluation. The learning set is in turn split into two sets: training, used for parameter estimation for a given structure, and validation, used for evaluating the trained network, obtaining a validation error.

ANN model selection is, therefore, related to the task of comparing several network structures/models based on estimations of their validation errors in order to select the most suitable network structure for a given problem. The estimation of the validation error can be obtained using the following well-known model evaluation methods:

- *K-fold cross-validation*: it is widely used in the literature [Lendasse et al., 2003],[Aran et al., 2009],[Constantinopoulos and Likas, 2006], but the value of K is based on a subjective judgment: a low value results in small variance and large bias and a high value makes the evaluation computationally expensive [Kohavi, 1995].

- *Leave-One-Out (LOO)*: it is less biased, but its variance [Lendasse et al., 2003] and computational cost are unacceptable.

- *Bootstrap*: it is downward biased and has a very low variance [Lendasse et al., 2003] but its computational load is high. The .632 bootstrap [Efron and Tibshirani, 1997], is almost unbiased and has a low variance, but it favors overfitting classifiers, which makes it not suitable for model evaluation [Kohavi, 1995].

All these model evaluation strategies are computationally expensive, which becomes a problem in ANN model selection since several optimizations and evaluations of different structure networks are required. In other words, a model evaluation method is run several times because one wants to select the best ANN model or structure among several ones, i.e., ANN model selection. Moreover, these model evaluation strategies are random approaches, since none of them takes the variability and geometry of the learning data into consideration when building the training and validation sets [Lendasse et al., 2003],[Constantinopoulos and Likas, 2006]. For example, one can use the functions such as *divideind, dividerand, divideblock*

and *divideint* from the MATLAB's Neural Networks toolbox^{TM}[MATLAB, 2010c] for building training and validation sets to be used in *K-fold cross-validation* model evaluation, but none of these functions take the variability of the data set into account to build such sets.

So, from the point of view of bias and variance, reproducibility and computational complexity, it is desirable that a model selection approach for ANNs is based on a model evaluation strategy with the following features: (1) low variance and bias, (2) no randomness, (3) low computational cost and (4) use of balanced and representative training and validation sets. Moreover, the use of such sets must play an essential role in all the steps related to the design of an RBFNN for time series prediction: initialization, optimization and network model evaluation. Such desirable model selection approach is proposed in the following chapter.

10.4 The *Mo*del *Se*lection algorithm for incremental RBFNN construction (MoSe)

According to the problems described in the previous section when using a *K-fold cross-validation*-based model selection methodology, it is presented a fast and new model selection methodology for incremental RBFNN construction, which is generally preferred over other strategies [Kwok and Yeung, 1997], in time series prediction tasks. The proposed method, described in [Florido et al., 2011b], is based on a network model evaluation strategy that is not random, uses balanced and representative training and validation sets from the learning set and has low variance, bias and computational cost.

First of all, the original data set $S_{total} = \{(x_k, y_k); k = 1, ..., n\}$ with $x_k \in \mathbb{R}^d$ and $y_k = F(x_k) \in \mathbb{R}$ is split into two parts: the first one forms the learning set S_{learn} which is used for the RBFNN model selection approach, and the second part forms the test set, S_{test}, for genuine out-of-sample evaluation in which the validity and usefulness of the model selected is checked. Assuming that the inputs of the network have been previously selected, the proposed *Mo*del *Se*lection (MoSe) algorithm consists of the proper combination of three main parts (see Fig.10.1):

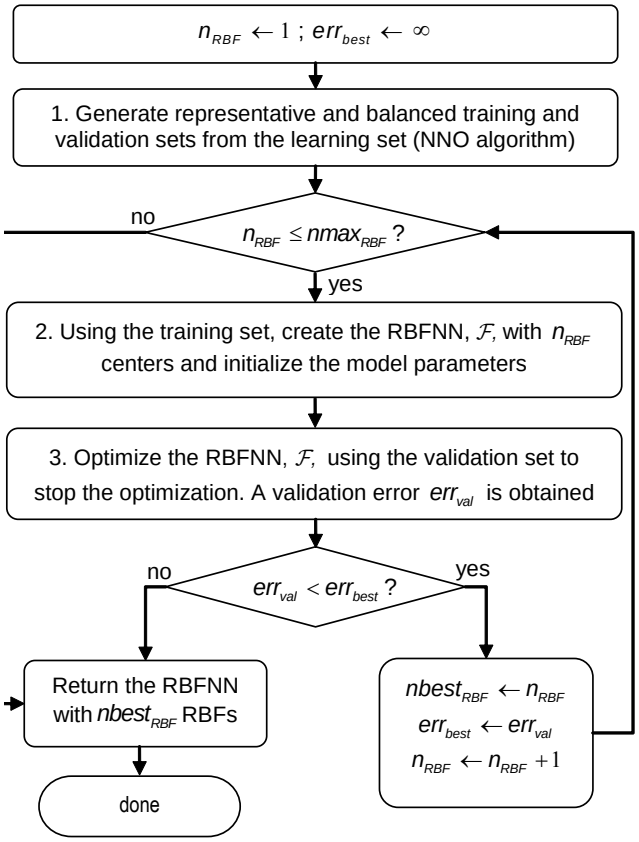


FIGURE 10.1: General description of the MoSe model selection algorithm

1. A deterministic generation of representative and balanced training, S_{train}, and validation, S_{val} sets from the learning set, S_{learn}, using the NNO algorithm proposed in section 9.3.2. The aim of generating the sets this way is three-fold: (1) create and initialize an RBFNN through a representative training set, (2) reduce the overfitting problem in the RBFNN training/optimization process and (3) avoid a *cross-validation*-based procedure for network model evaluation (section 10.4.1).

2. The creation of the RBFNN $\mathcal{F}$ using the representative training set S_{train} with centers, weights and radii with initial values. The RBF centers are initialized using the ICFA algorithm [Guillén et al., 2007] (section 10.4.2).

3. A training/optimization procedure for the RBFNN $\mathcal{F}$, based on the Levenberg-Marquardt local search algorithm. This optimization procedure has been designed for using the representative validation set S_{val} created in step 1 to stop the optimization when overfitting appears. At the end of the optimization process, a validation error is given as a result of evaluating the optimized network $\mathcal{F}^*$ using the validation set S_{val}. The validation error is unique, since the validation set is enough representative from the learning set S_{learn}. This serves as a *non-cross-validation* model evaluation for the network $\mathcal{F}^*$ (section 10.4.3).

The model selection algorithm runs steps 2 and 3 varying the number of nodes or RBFs starting at 1 until the validation error is not improved or a maximum number of RBFs is reached. The final RBFNN forecasting model selected is the one related to the lowest validation error found (section 10.4.4).

Thus, the novelty of this work is the whole model selection procedure. It is a methodology that combines, in a suitable way, the use of different parts and, among them, a deterministic distribution of the learning set into training and validation sets is used. This distribution is then integrated in the remaining parts of the combined algorithm to get a fast model selection methodology for RBFNN.

10.4.1 A deterministic generation of representative and balanced training and validation sets

The first step of the proposed method consists in distributing the learning set, S_{learn}, into two mutually exclusive sets: the training set, S_{train}, and the validation set, S_{val}, according to the euclidean distance. This distribution is not random and takes into account the variability and the geometry of the learning set when building the training and validation sets. The distribution algorithm used is a slight variation of the *Nearest Neighbor out (NNO)* methodology described in section 9.3.2. Two important issues must be emphasized:

- Although the original distribution algorithm described in section 9.3 (the NNO-CFA algorithm) is a combination of a clustering procedure (CFA) and

a distribution algorithm (NNO), in the context of RBFNN model selection, it is preferable to use only the NNO algorithm as the distribution algorithm with the aim of avoiding higher computational costs when selecting the best RBFNN model. In section 9.5.2, it has been demonstrated that although the best results are achieved for the NNO-CFA algorithm, the NNO algorithm itself is a valid alternative to NNO-CFA when generating balanced and representative sets in terms of computational cost without sacrifying the quality of the distribution in high degree.

- The original NNO method described in section 9.3.2, departs from a cluster C_i and, iteratively, distributes a pair of data values contained in the cluster into two subsets: learning and test. In the context of the MoSe algorithm, a slight variation is introduced in the following way: the NNO approach departs from the set of input values of S_{learn}, which is referred as X_{learn}. For the stage 1, the training and validation inputs sets are initialized with a single value each: the starting point from which the data is distributed is changed to the space center of the input learning set X_{learn}. Then, the stage 2 iteratively distributes a pair of data values contained in that set into the input training set, X_{train}, and the input validation set, X_{val}. At the end of the procedure, the training set S_{train} is formed by X_{train} and its corresponding outputs and the validation set S_{val} is formed by X_{val} and its corresponding outputs. Thus, S_{train} is the set of input/output values used as training and S_{val} is the set of input/output values used as validation (see Fig.10.2).

The purpose of using the distribution algorithm is three-fold: (1) create and initialize the RBFNN using a representative training set, (2) reduce the overfitting problem in the RBFNN training/optimization process and, therefore, improve the network forecasting accuracy and (3) avoid a *cross-validation*-based model evaluation procedure for a given network structure, which means a reduction in the computation time employed in selecting the RBFNN model.

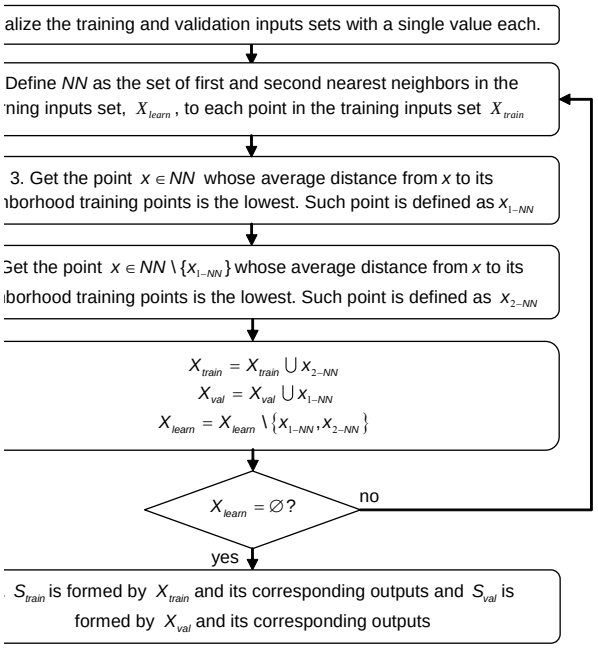


FIGURE 10.2: The Nearest Neighbor Out (NNO) distribution algorithm in the context of MoSe methodology

10.4.2 Create the RBFNN $\mathcal{F}$

Given a number of nodes or RBFs and the training set S_{train}, the next step of the proposed method consists in creating the RBFNN, $\mathcal{F}$, with initial values for the model parameters: centers (C), radii (R) and weights (Ω). The RBFNN initialization is an important task since the network optimization strongly depends on the initial values of the model parameters [Guillén et al., 2007]. Due to this importance, it is important to take advantage of a representative training set for such initial values.

10.4.2.1 Centers initialization

The centers are initialized using the deterministic ICFA algorithm [Guillén et al., 2007] described in section 10.2.1. This method has proved to outperform other approaches for the initialization of the RBFNN centers [Guillén et al., 2007]. Moreover, the ICFA algorithm effectiveness is expected to improve, since it is run in a representative training set.

10.4.2.2 Radii initialization

In this dissertation, the radii are the same for each radial basis function (RBF) with the aim of reducing the nonlinear parameters. The initial value of the radii is based on the nearest neighbor heuristic described in [Karayiannis and Mi, 1997]: the width of each RBF is set to the euclidean distance of its nearest neighbor function. Since in our work the radii are the same for all RBFs, the width taken is the maximum value of all RBF widths calculated this way.

10.4.2.3 Weights initialization

Since the output of the RBFNN is linear with respect to the weights (Eq.10.1), it is possible to optimally obtain these parameters through a linear equation system that can be robustly solved. In our case the Singular Value Decomposition (SVD) method [Gonzalez et al., 2003] is used.

10.4.3 RBFNN training/optimization and model evaluation

Once the RBFNN $\mathcal{F}$ has been created with initial model parameters for a given number of RBFs, the model parameters must be optimized. The optimization procedure has been designed for using, within the well-known Levenberg-Marquardt local search algorithm [Parker, 1987], the representative training set to train/optimize the network $\mathcal{F}$ and the validation set to evaluate the trained network during the optimization process, stopping the latter when overfitting appears. This way, it is expected to improve the forecasting accuracy of the model. At the end of

the optimization process, a validation error is given as a result of evaluating the optimized network $\mathcal{F}^*$ using the representative validation set (network/model evaluation). This serves as a non-cross-validation model evaluation for the network.

The use of balanced and representative training and validation sets (section 10.4.1) in this context has several advantages. First, both the network optimization and the evaluation are done on these sets which might reduce the overfitting problem in the RBFNN training process and improve the forecasting accuracy. Secondly, a given network structure is optimized once and the validation error is both unique and a representative estimation of the network forecasting accuracy. On the contrary, in a *K-fold cross-validation* model evaluation strategy, a given network structure is optimized K times using K different training sets, obtaining an average validation error. In other words, our methodology runs just one optimization process whereas a *K-fold cross-validation* model evaluation strategy executes K optimization processes where K is the number of different training sets. This way, the computation time employed in evaluating a model is reduced and, consequently, the running time of the model selection process.

10.4.4 RBFNN model selection

According to Fig. 10.1, a different RBFNN $\mathcal{F}$ is built each time varying the number of RBFs in a greedy incremental approach, obtaining a different validation error for each optimized RBFNN $\mathcal{F}*$. Since the validation error obtained for each RBFNN architecture is a representative estimation of the network forecasting accuracy, the iterative procedure is stopped once the validation error is not improved with respect to the previous number of RBFs used or the maximum number of RBFs is reached. This means that, if the validation error is worse when increasing the number of RBFs, this number of nodes used in the network does not improve its accuracy on the validation set. Thus, the RBFNN used as the final forecasting model has the number of RBFs given in the previous iteration.

10.5 Experiments and results

In this section, and for comparison purposes, other strategies will be explained as alternative model selection for the incremental RBFNNs construction (section 10.5.1). The quality of all these methodologies will be evaluated through three well known time series prediction benchmarks (section 10.5.2). The results of the RBFNN model performances are reported in section 10.5.3.1 and statistically analyzed in section 10.5.3.2. The RBFNN forecasting model selected by the proposed method, MoSe, is compared, in terms of forecasting accuracy, to other traditional methods for time series prediction in section 10.5.3.3. Finally, the results obtained in the time series competition MINCODA'09 using a slight variation of the RBFNN model selection proposed are briefly explained in section 10.5.3.4.

10.5.1 Other RBFNN model selection strategies

As described above, other model selection strategies for incremental RBFNN construction will be used for comparison purposes. They can be classified into two categories: variations of the MoSe algorithm and strategies based on a *K-fold cross-validation* model evaluation methodology.

10.5.1.1 Variations of the MoSe algorithm

Two variations of the MoSe algorithm in terms of the generation of the training and validation sets from the learning set (step 1 of MoSe algorithm, section 10.4.1) are proposed:

- *MoSe-R*: the learning set is randomly split into training and validation sets.
- *MoSe-D*: the first and the second half of the learning set are used as training and validation sets respectively. Since there is an order in time series data, this partition makes sense.

This way, we will evaluate which methodology used when building the training and validation sets from the learning set is better: the one used by MoSe (section 10.4.1), Mose-R or MoSe-D.

10.5.1.2 Strategies based on a *K-fold cross-validation* methodology

Two model selection approaches based on a *K-fold cross-validation* (K-CV) model evaluation methodology, which is commonly used in the literature [Aran et al., 2009],[Lendasse et al., 2003],[Constantinopoulos and Likas, 2006], are also applied for comparison purposes. The main differences with respect to MoSe, MoSe-R and MoSe-D model selection methodologies are in terms of the generation of the training and validation sets, the RBFNN optimization process and the iterative procedure typical of a cross-validation approach to evaluate a given model/structure network (Fig.10.3):

- *K-CV-R*. The learning set is randomly split into K roughly equal parts, each one being used successively as a validation set. With regard to the RBFNN optimization process, the validation set is not used during the optimization process and once the network is optimized, it is evaluated using the validation set. The model evaluation for a given number of RBFs consists, therefore, in optimizing K networks using a different training set each time, obtaining K different validation errors. The final validation error corresponds to the mean of these errors.

- *K-CV*. This approach differs from *K-CV-R* only in the partition of the learning set into K parts. In this case, the learning set is split in a non-random way, that is, the first set corresponds to the first part of the learning set, the second set to the second part, etc. The K^{th} set is related to the K^{th} part of the learning set.

It is important to emphasize that the distribution strategy used by the MoSe algorithm (section 10.4.1) is the only one that takes into account the variability and the geometry of the learning set when building the training and validation sets.

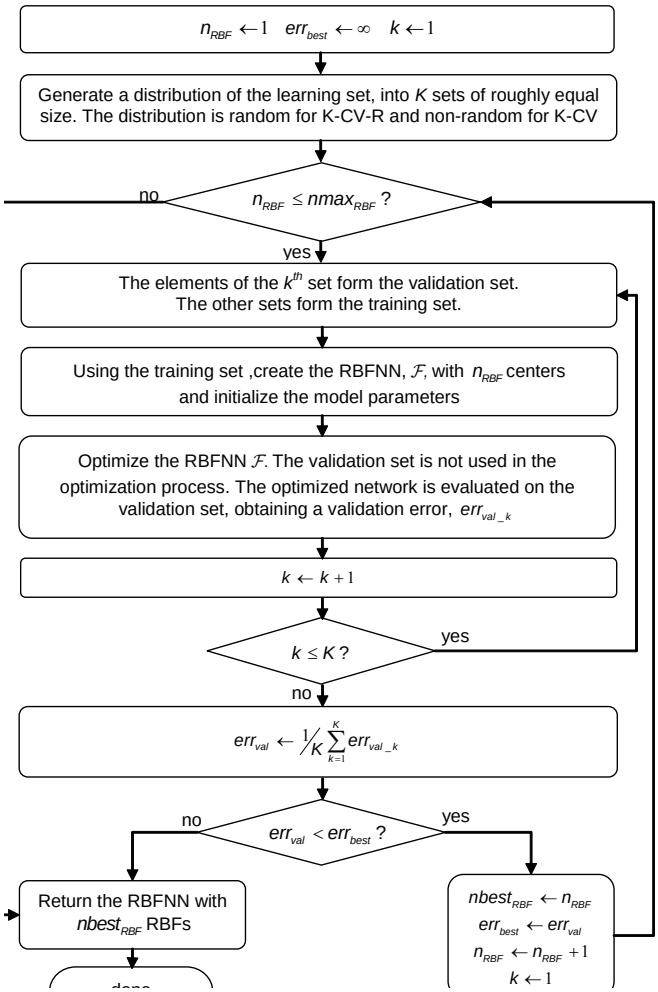


FIGURE 10.3: General description of the K-CV-R and K-CV model selection procedures based on *K-fold cross-validation*

10.5.2 Time series prediction benchmarks

Each model selection methodology has been applied on three small nonlinear chaotic time series predictions benchmarks with different size of samples and number of inputs. The required goal is to check the prediction accuracy of the RBFNN selected by each model selection strategy. The time series are the Logistic map, the Hénon map and the Mackey-Glass time series, which have been already described

in section 9.5.1.1. The inputs, which are the same as the ones described in that section, and the size of each time series are shown in Table 10.1.

TABLE 10.1: Size and models of the time series benchmarks

Data set	**Model**	**Size**
Hénon map	$y_t = F(y_{t-1}, y_{t-2})$	200
Logistic map	$y_t = F(y_{t-1})$	150
Mackey Glass	$y_t = F(y_{t-6}, y_{t-12}, y_{t-18}, y_{t-24})$	250

The small size of the time series have been chosen deliberately, with the aim of demonstrating the suitability and the importance of the distribution algorithm (section 10.4.1) when small data sets are used. Since our main interest is focused on the model selection strategy proposed in time series prediction problems, each time series will be split into two parts: the first 70% corresponds to the learning set S_{learn}, which is used by the MoSe algorithm and the other model selection strategies to select the best RBFNN model, and the remaining 30% corresponds to the test set, S_{test}, to evaluate the forecasting accuracy of the model selected.

10.5.3 Results

10.5.3.1 Results for the RBFNN model selection algorithms

A total of 21 model selection algorithms have been suggested in previous sections: the one proposed in this chapter, *MoSe*, its variations, *MoSe-R* and *MoSe-D*, and the ones based on a *K-fold cross-validation* methodologies,*K-CV-R* and *K-CV* for $K = [2, 10]$. For each time series benchmark described in section 10.5.2, the 21 algorithms will be run with the aim of checking their quality in terms of the prediction accuracy in the test set S_{test}, given by the *Mean Square Error* (MSE_{test}):

$$MSE_{test}(\mathcal{F}^*) = \frac{1}{n_{test}} \sum_{(x_i^{test}, y_i^{test}) \in S_{test}} ((\mathcal{F}^*(x_i^{test}) - y_i^{test})^2 \tag{10.3}$$

where $(x_i^{test}, y_i^{test}) \in S_{test}$, $\mathcal{F}^*(x_i^{test})$ is the approximation of y_i^{test} by the RBFNN $\mathcal{F}^*$ chosen by a model selection algorithm and n_{test} is the number of input/output data values contained in the test set S_{test}.

The 21 algorithms will be also evaluated in terms of the computation time t, which is the time needed by a methodology to find the best RBFNN model.

The results of applying the methodologies to the time series benchmarks, will be shown in three different tables, one for each time series, where it can be compared the quality of the RBFNN models selected for each algorithm. This way, it is easy to see the individual behavior of each algorithm when comparing to the rest in the context of time series benchmarks with different size and number of inputs. Due to the random nature of *MoSe-R*, and *K-CV-R* methods, they have been run 20 times in each time series and their results have been averaged and the standard deviations have also been computed.

Table 10.2 shows that the best prediction accuracy, lowest MSE_{test} value, for the Logistic Map time series is achieved by the *MoSe* method when the number of RBFs is 8. The time needed to find the RBFNN model is approximately, in average, 14.8 seconds. Although other model selection methods (*MoSe-R*, *MoSe-D*, *2-CV-R* and *2-CV*) find their best RBFNN model quicker, their prediction accuracy on the test set is not better than the one given by the *MoSe* method. Other two methodologies (*5-CV* and *6-CV*) achieve the same MSE_{test}, but the number of folds used in this case increases the computation time up to 39.6 and 52 seconds respectively.

From table 10.3 (Hénon map time series), the RBFNN model selected by the *5-CV* algorithm achieves the best prediction accuracy, but its computation time is much greater (4.7 times) than the one given by the *MoSe* method, which provides the second lowest MSE_{test} value. This fact shows that the *MoSe* algorithm, with lower computation time, produces a prediction error very similar to the one given by the *5-CV* method and, therefore, can be selected as a fast and reliable RBFNN model selection method. Other algorithms, *8-CV* and *9-CV*, also provide the second best prediction accuracy but their computation time is 5.59 and 6.49 times respectively greater than the one given by the *MoSe* algorithm.

TABLE 10.2: Results for logistic map time series by different RBFNN model selection strategies

Algorithm	#RBFs	MSE_{test}	t(sec)
MoSe	**8**	**1.4E-4**	**14.8 ± 0.1**
MoSe-R	5.3 ± 1.9	0.94E-3 ± 1.41E-3	6.9 ± 3.5
MoSe-D	7	3.8E-4	11.3 ± 0.1
2-CV-R	6.1 ± 1.1	4.4E-4 ± 1.4E-4	12.7 ± 4.4
2-CV	6	3.9E-4	10.7 ± 0.1
3-CV-R	6.4 ± 1.1	4.2E-4 ± 1.3E-4	18.7 ± 7.1
3-CV	7	2.7E-3	48.9 ± 0.1
4-CV-R	7.2 ± 1.2	4.4E-4 ± 5.6E-4	31.2 ± 10.7
4-CV	7	3.8E-4	29.2 ± 0.1
5-CV-R	7.2 ± 1.4	4.6E-4 ± 5.5E-4	38.2 ± 15.3
5-CV	**8**	**1.4E-4**	**39.6 ± 0.1**
6-CV-R	7.2 ± 1.2	3.3E-4 ± 1.4E-4	43.3 ± 13.9
6-CV	**8**	**1.4E-4**	**52.0 ± 0.1**
7-CV-R	7.6 ± 1.0	2.8E-4 ± 1.1E-4	56.2 ± 15.3
7-CV	7	3.8E-4	48.3 ± 0.1
8-CV-R	7.3 ± 0.8	3.2E-4 ± 1.0E-04	58.0 ± 12.6
8-CV	7	3.8E-4	52.7 ± 0.1
9-CV-R	7.4 ± 0.7	4.5E-4 ± 5.5E-4	66.2 ± 15.7
9-CV	7	3.8E-4	58.3 ± 0.1
10-CV-R	7.1 ± 1.1	4.5E-4 ± 5.5E-4	69.1 ± 20.9
10-CV	7	3.8E-4	68.1 ± 0.1

Table 10.4 shows that the best prediction accuracy for the Mackey-Glass time series is given by the *MoSe* method and no other methodology can achieve the same accuracy. Moreover, the *MoSe* computation time is very satisfactory compared to the rest of algorithms.

These results confirm the superiority of the *MoSe* algorithm in terms of prediction accuracy and computation time over the rest of RBFNN model selection methodologies. From tables 10.2 to 10.4, it can also be observed how those model selection methodologies based on random distributions of the learning set into training and validation sets (*MoSe-R* and *K-CV-R* for $K = [2, 10]$), estimate a variable number of RBFs. Moreover, these algorithms show a high variability in terms of both the MSE_{test} value and computation time, due to this heterogeneous number of estimated RBFs. On the other hand, *MoSe-R* and *MoSe-D* are, in general, not suitable methods for RBFNN model selection, since their MSE_{test} values are not as good as the ones achieved with *K-CV-R* or *K-CV* for low values of K. This fact suggests

TABLE 10.3: Results for Hénon map time series by different RBFNN model selection strategies

Algorithm	#RBFs	MSE_{test}	t (sec)
MoSe	**6**	**9.2E-5**	**8.1 ± 0.1**
MoSe-R	4.4 ± 1.6	2.3E-3 ± 1.5E-3	4.7 ± 3.5
MoSe-D	3	3.5E-3	2.4 ± 0.1
2-CV-R	3.7 ± 1.2	2.9E-3 ± 1.3E-3	5.6 ± 3.7
2-CV	3	3.5E-3	4.1 ± 0.1
3-CV-R	4.4 ± 1.4	2.1E-3 ± 1.5E-3	10.9 ± 6.2
3-CV	4	2.8E-3	10.6 ± 0.1
4-CV-R	4.2 ± 1.2	2.1E-3 ± 1.5E-3	14.2 ± 7.1
4-CV	3	3.5E-3	8.3 ± 0.1
5-CV-R	3.9 ± 1.1	2.7E-3 ± 1.2E-3	14.9 ± 7.9
5-CV	**7**	**5.2E-05**	**38.3 ± 0.1**
6-CV-R	4.3 ± 1.4	2.1E-3 ± 1.5E-3	21.9 ± 12.6
6-CV	3	3.5E-3	12.8 ± 0.1
7-CV-R	4.2 ± 1.5	2.1E-3 ± 1.7E-3	23.9 ± 13.6
7-CV	5	1.7E-4	28.5 ± 0.1
8-CV-R	4.8 ± 1.6	1.4E-3 ± 1.7E-3	33.9 ± 17.3
8-CV	**6**	**9.2E-5**	**45.3 ± 0.1**
9-CV-R	4.3 ± 1.5	2.1E-3 ± 1.7E-3	33.2 ± 19.1
9-CV	**6**	**9.2E-5**	**52.6 ± 0.1**
10-CV-R	4.3 ± 1.7	2.3E-3 ± 1.6E-3	37.3 ± 26.1
10-CV	3	3.5E-3	19.3 ± 0.1

that the deterministic distribution of data proposed in section 10.4.1 is essential for RBFNN model selection since it distributes the learning set into two balanced and representative sets, training and validation, taking into account the variability and the geometry of the learning data.

10.5.3.2 Statistical tests: the Analysis of Variance (ANOVA)

When applying different algorithms to several examples, it is interesting to examine the effects of the algorithms and the examples on the results, either in terms of the prediction accuracy, MSE_{test} or the computation time. More important, it is essential to check whether the differences of the results among the RBFNN model selection methods described in section 10.5.3.1 are due to chance or not. For this purpose, the analysis of variance (ANOVA) test Box et al. [1978] is useful when it is suspected that one or more factors affect a response. The statistical parameter

TABLE 10.4: Results for Mackey-Glass time series by different RBFNN model selection strategies

Algorithm	#RBFs	MSE_{test}	t (sec)
MoSe	**5**	**6.6E-5**	**10.2 ± 0.1**
MoSe-R	4.1 ± 0.8	4.2E-4 ± 8.4E-4	6.7 ± 1.9
MoSe-D	2	2.4E-3	2.8 ± 0.1
2-CV-R	3.6 ± 1	0.78E-3 ± 1.11E-3	9.1 ± 3.3
2-CV	4	8.2E-5	12.3 ± 0.1
3-CV-R	3.9 ± 1.2	6.8E-4 ± 1.0E-3	15.3 ± 6.8
3-CV	2	2.4E-3	4.4 ± 0.1
4-CV-R	4.2 ± 1.3	5.8E-4 ± 9.6E-4	23.2 ± 10.2
4-CV	2	2.4E-3	7.2 ± 0.1
5-CV-R	3.6 ± 0.9	6.7E-4 ± 1.0E-3	22.6 ± 8
5-CV	4	8.2E-5	27.9 ± 0.1
6-CV-R	3.2 ± 1.1	1.1E-3 ± 1.2E-3	23.2 ± 11.4
6-CV	2	2.4E-3	12.3 ± 0.1
7-CV-R	3.3 ± 1.1	1.0E-3 ± 1.2E-3	28.7 ± 12.9
7-CV	2	2.4E-3	16.0 ± 0.1
8-CV-R	3.6 ± 1.3	0.96E-3 ± 1.11E-3	36.6 ± 18.3
8-CV	4	8.2E-5	44.0 ± 0.1
9-CV-R	3.1 ± 1.3	1.4E-3 ± 1.2E-3	35 ± 21
9-CV	2	2.4E-3	16.6 ± 0.1
10-CV-R	3.9 ± 1.2	0.67E-3 ± 1.12E-3	52.7 ± 21.4
10-CV	4	8.2E-5	54.9 ± 0.1

considered in this test is the significant level and if it is lower than 0.05, then the corresponding levels of the factor are statistically significant with a confidence level of 95%. In the case that a response is affected by one or more factors, a more profound study must be carried out to classify the levels of the significant factors through the *multiple range test* Rojas et al. [2000]. The levels of a factor that are not statistically different form a homogeneous group, and therefore the selection between the various levels belonging to a given homogeneous group has no significant repercussion on the response.

Therefore, in the statistical study performed in this section, the factors considered are the RBFNN model selection algorithm and the time series benchmark (example). The levels of the algorithm factor are *MoSe*, *MoSe-R*, *MoSe-D* and *K-CV-R* and *K-CV* for $K = [2, 10]$ and the levels for the example factor are Logistic map, Hénon map and Mackey-Glass. It is suspected that these factors affect the response variables, which are the forecasting accuracy of the RBFNN model selected in the

test set MSE_{test} and the computation time t which is the time needed by a method to find the best RBFNN model. Thus, two different statistical studies are performed, one for each response variable and, for this purpose, for each time series benchmark (example), each model selection algorithm has been run 10 times. The normality, independence of populations and homoscedasticity assumptions for ANOVA test [Box et al., 1978] are accomplished and for each example, the results have been normalized to the interval $[0, 1]$ to allow a fair comparison among the values of the different factors. For brevity purposes, only the hypothesis or the levels of the algorithm and example factors will be analyzed.

When the response variable in the ANOVA test is the MSE_{test} value, it can be observed from table 10.5 that the significante level of both factors, algorithm and example, is lower than 0.05, which means that at least one of the levels of the analyzed factors affects the MSE_{test} value (the null hypothesis is rejected).

TABLE 10.5: ANOVA table for the analysis of the main factors for the MSE_{test} response

Main factors	Sum of squares	D.F.	Mean square	F-Ratio	Sig.level
Algorithm	40.26	20	2.01	21.33	0
Example	40.20	2	20.10	212.93	0

Table 10.6 classifies the levels of the significant *algorithm* factor through the multiple range test. It can be observed five different homogeneous groups (A,B,C,D and E) and the algorithms present within each group are not statistically different. The best prediction results are achieved for the *5-CV*, *MoSe* and *8-CV* model selection methodologies (homogeneous group A), although from tables 10.2 to 10.4 it has been demonstrated that the computation load of *MoSe* is very low when compared to *5-CV* and *8-CV*. From table 10.6, it can also be observed that the two variation of *MoSe* algorithm, *MoSe-D* and *MoSe-R* (section 10.5.1.1), get high mean when compared to *MoSE*. This fact demonstrates the improvement of the prediction accuracy when the deterministic distribution of data proposed in section 10.4.1 is used in the model selection procedure. Clearly, the prediction accuracy gets worse when applying a random (*MoSe-R*) or a classic distribution (*MoSe-D*) before the RBFNN is built and optimized.

TABLE 10.6: Multiple range test for the factor *algorithm* when using MSE_{test} as response

Factor levels	Mean squares	Homogeneous groups
5-CV	0.0022	A
MoSe	0.0024	A
8-CV	0.0179	A
8-CV-R	0.2289	B
4-CV-R	0.2385	BC
3-CV-R	0.2464	BCD
MoSe-R	0.2534	BCD
10-CV-R	0.2625	BCD
10-CV	0.2648	BCD
2-CV	0.2655	BCD
5-CV-R	0.2908	BCD
7-CV-R	0.2911	BCD
6-CV-R	0.3118	BCD
2-CV-R	0.3202	BCD
9-CV-R	0.3463	CD
9-CV	0.3490	D
7-CV	0.3543	D
6-CV	0.5832	E
4-CV	0.5959	E
MoSe-D	0.5959	E
3-CV	0.6673	E

Table 10.7 classifies the levels of the significant *example* factor for the MSE_{test} response. Two different homogeneous groups are found: the one given by logistic map time series and the other given by both the Hénon map and Mackey-Glass time series. The best accuracy prediction result (lower mean value) is achieved for the logistic map time series. It seems that, in terms of accuracy prediction, better results are achieved for small time series with less inputs (Logistic map is the one with the smallest number of samples and inputs), but further analysis is needed for this assumption.

When the response variable in the ANOVA test is the computation time value t, it can be observed from table 10.8 that the significance level of both factors, algorithm and example, is lower than 0.05, which means that at least one of the levels of the analyzed factors affects the computation time t (the null hypothesis is rejected).

Table 10.9 classifies the levels of the significant *algorithm* factor. It can be observed

TABLE 10.7: Multiple range test for the factor *example* when using MSE_{test} as response

Factor levels	Mean squares	Homogeneous groups
Logistic map	0.0564	A
Hénon map	0.4313	B
Mackey-Glass	0.4392	B

TABLE 10.8: ANOVA table for the analysis of the main factors for the computation time *t* response

Main factors	Sum of squares	D.F.	Mean square	F-Ratio	Sig.level
Algorithm	26.02	20	1.30	90.80	0
Example	2.54	2	1.27	88.59	0

eleven different homogeneous groups (each letter represents a homogeneous group) and the algorithms present within each group are not statistically different. The best computation time results are achieved for the *MoSe-D*, *MoSe-R* and *2-CV-R* model selection algorithms. This is obvious because these algorithms require less computation time than others with higher number of folds ($K > 2$),but please remind from table 10.6 that these methodologies do not achieve the best prediction accuracy values MSE_{test} and none of them take into account the variability and the geometry of the learning set when building the training and validation sets. The *MoSe* method, which belongs to the group of the best model selection methodologies as shown in table 10.6, is present in two homogeneous groups: one given by the *2-CV-R*, *2-CV* and *4-CV* algorithms (group C) and the other given by *4-CV* and *3-CV-R* algorithms (group D), which means that the proposed *MoSe* algorithm is equivalent in terms of computation time to these methodologies with low number of folds ($K <= 4$). Moreover, from table 10.9, it can also be noticed how the computation time increases as the number of folds, K, becomes larger. This fact proves how the *K-fold cross-validation*-based model selection methodology for high values of K is computationally expensive.

Finally, table 10.10 classifies the levels of the significant *example* factor for the time *t* response. Each time series (example) forms a group, that is, there are statistical differences among these time series in terms of computation time, which is obvious,

TABLE 10.9: Multiple range test for the factor *algorithm* when using the computation time t as response

Factor levels	Mean squares	Homogeneous groups
MoSe-D	0.0350	A
MoSe-R	0.0480	AB
2-CV-R	0.0753	ABC
2-CV	0.0798	BC
MoSe	0.0934	CD
4-CV	0.1168	CD
3-CV-R	0.1342	DE
3-CV	0.1610	E
6-CV	0.2098	F
4-CV-R	0.2099	F
5-CV-R	0.2269	FG
6-CV-R	0.2643	G
7-CV	0.2697	G
7-CV-R	0.3262	H
5-CV	0.3272	H
9-CV	0.3702	I
8-CV-R	0.3966	I
9-CV-R	0.4076	I
8-CV	0.4516	J
10-CV	0.4542	J
10-CV-R	0.5043	K

since each one has both different number of samples and number of input values (see table 10.1).

TABLE 10.10: Multiple range test for the factor *example* when using the computation time t as response

Factor levels	Mean squares	Homogeneous groups
Hénon map	0.1859	A
Mackey-Glass	0.2576	B
Logistic map	0.2939	C

Thus, the statistical analysis performed in this section guides to the following conclusions:

- The superiority of the proposed *MoSe* model selection methodology when considering both the prediction accuracy (table 10.6) and the computation

time (table 10.9) regardless of the time series benchmark chosen.

- The drawbacks when using model selection procedures based on *K-fold cross-validation*. A researcher has two subjective choices: first, to decide *a priori* the value of K with the risk of choosing an inappropriate RBFNN model or second, execute *K-fold cross-validation* for a range of values for K, which is computationally expensive.

- The usefulness of the deterministic generation of representative and balanced training and validation sets (section 10.4.1) for the RBFNN model selection problem.

- The suitability of the ANOVA test for analyzing the effect of one or more factors on a response and to classify the levels of significant factors.

10.5.3.3 MoSe RBFNN versus other time series prediction methodologies

It is also interesting to check the forecasting accuracy of the RBFNN model selected by the proposed methodology, *MoSe*, when compared to other traditional methods for time series prediction. For this purpose, we have chosen classical linear models such as *Auto- Regressive Moving Average* (ARMA) models [Balaguer et al., 2008] and non linear models such as *Multilayer Perceptron* Neural Networks (MLP) [Shiblee et al., 2009], *Adaptive Network-based Fuzzy Inference Systems* (ANFIS) [Boyacioglu and Avci, 2010] and *Weighted K-Nearest Neighbors* (WKNN), which are a variant of the K-Nearest Neighbors technique [Sorjamaa et al., 2005].

Thus, for each time series benchmark described in section 10.5.2, the ARMA, MLP, ANFIS and WKNN models, with different parameters and/or structures, will be trained on the learning set S_{learn} and the forecasting model built will be evaluated using the test set S_{test}, obtaining the corresponding MSE_{test} error, which is the forecasting accuracy. This error will be compared to the one given by the final RBFNN forecasting model selected by the *MoSe* method, referred to as M-RBFNN. This error has already been estimated in section 10.5.3 (tables 10.2 to 10.4).

Table 10.11 shows the comparison between the M-RBFNN model and ARMA models of order (m, n). For each time series benchmark, ARMA models of order (m, n) with $m = [1, 10]$ and $n = [1, 10]$, have been run. Since the number of possible ARMA models is 100, the one with lowest Akaike's Information Criterion (AIC) value [Akaike, 1973] on the learning set S_{learn} is shown. The AIC criterion is commonly used in traditional statistical models where the number of parameters is small [Qi and Zhang, 2001]. It can be observed from table 10.11 that, for any time series benchmark, the M-RBFNN forecasting model is superior to the ARMA models in terms of prediction accuracy. Not surprisingly, linear ARMA models are not proper for modeling highly non linear time series, such as the Logistic and the Hénon maps. The forecasting accuracy obtained by ARMA(1,1) on Mackey-Glass time series is nevertheless quite good.

TABLE 10.11: Comparison between M-RBFNN and ARMA models

Example	TSP method	MSE_{test}
Logistic map	**M-RBFNN**	**1.4E-4**
	ARMA(1,1)	0.2
Hénon map	**M-RBFNN**	**9.2E-5**
	ARMA(9,7)	4.9E-2
Mackey-Glass	**M-RBFNN**	**6.6E-5**
	ARMA(1,1)	5.2E-4

Table 10.12 shows the comparison between the M-RBFNN model and *Multilayer Perceptron* Neural Network (MLP) models. In this case, several MLP architectures with different number of hidden layers, $n_l = [1, 3]$, and different number of neurons per layer, $n_n = [1, 20]$, have been run. Since MLP models are not deterministic, the prediction accuracy has been averaged from 10 different executions for a given number of layers and neurons. It can be observed from table 10.12 that, for any example, the M-RBFNN model achieves the best prediction accuracy when compared to MLPs. Also, the time needed to learn the network, t_{learn}, for a given architecture (number of hidden layers/neurons per layer for MLPs and number of neurons for a single hidden layer for M-RBFNN), is lower for M-RBFNN. Although MLPs are considered universal approximators [Park and Sandberg, 1991], it is well known that any lack of success in applications of MLPs to time series prediction/function approximation, arises from inadequate learning or insufficient numbers of hidden units [Hornik et al., 1989]. This might be the reason of the bad MLP's performance

compared to M-RBFNN, specially for the Hénon map and Mackey-Glass time series. An increase in the number of hidden units might provide better results, but at the expense of high computational time.

TABLE 10.12: Comparison between M-RBFNN and MLP models. n_l is the number of hidden layers and n_n is the number of neurons per layer

Example	TSP method	n_l-n_n	MSE_{test}	t_{learn} (sec)
Logistic map	**M-RBFNN**	**1-8**	**1.4E-4**	**3.9 ± 0.1**
	MLP	1-2	0.5 ± 1.3	3.2 ± 0.7
		1-5	5.2E-4 ± 4.2E-4	3.7 ± 0.1
		1-10	5.9E-3 ± 2.0E-3	4.0 ± 0.1
		1-15	1.5E-2 ± 6.1E-3	4.0 ± 0.1
		1-20	4.3E-3 ± 1.0E-3	4.3 ± 0.1
		2-2	0.6E-3 ± 1.1E-3	3.6 ± 0.1
		2-5	5.9E-3 ± 4.2E-3	3.9 ± 0.1
		2-10	1.5E-2 ± 0.6E-2	5.3 ± 0.2
		2-15	4.5E-3 ± 1.4E-3	8.6 ± 0.2
		2-20	1.3E-2 ± 0.6E-2	21.1 ± 1.5
		3-2	0.3 ± 0.7	3.9 ± 0.2
		3-5	4.6E-3 ± 6.6E-3	4.3 ± 0.1
		3-10	7.4E-3 ± 4.9E-3	8.0 ± 0.3
		3-15	3.9E-3 ± 1.7E-3	23.3 ± 0.1
		3-20	1.8E-2 ± 1.5E-2	80.7 ± 0.3
Hénon map	**M-RBFNN**	**1-6**	**9.2E-5**	**2.9 ± 0.1**
	MLP	1-2	3.2E-1 ± 8.9E-1	4.2 ± 0.9
		1-5	1.1E-2 ± 1.4E-2	3.8 ± 0.1
		1-10	1.6E-3 ± 1.1E-3	3.9 ± 0.1
		1-15	1.9E-3 ± 0.8E-3	4.6 ± 0.3
		1-20	2.7E-3 ± 1.3E-3	4.6 ± 0.1
		2-2	4.1E-2 ± 5.1E-1	4.0 ± 0.1
		2-5	4.6E-3 ± 2.8E-3	4.3 ± 0.1
		2-10	2.8E-3 ± 1.7E-3	5.6 ± 0.2
		2-15	2.8E-3 ± 1.9E-3	8.3 ± 0.3
		2-20	1.6E-3 ± 1.1E-3	19.9 ± 0.2
		3-2	3.2E-1 ± 4.8E-1	3.9 ± 0.1
		3-5	3.9E-2 ± 9.8E-2	4.2 ± 0.1
		3-10	2.8E-3 ± 1.8E-3	7.2 ± 0.2
		3-15	2.8E-3 ± 1.9E-3	32.3 ± 0.2
		3-20	6.4E-3 ± 3.6E-3	80.6 ± 0.2
Mackey-Glass	**M-RBFNN**	**1-5**	**6.6E-5**	**2.1 ±0.1**
	MLP	1-2	0.6E-1 ± 1.4E-1	3.6 ± 0.1
		1-5	7.3E-3 ± 2.3E-3	3.7 ± 0.1
		1-10	7.3E-3 ± 2.9E-3	3.9 ± 0.1
		1-15	5.6E-3 ± 1.9E-3	4.1 ± 0.1
		1-20	4.1E-3 ± 2.2E-3	4.6 ± 0.1
		2-2	1.9E-1 ± 1.9E-1	3.9 ± 0.2
		2-5	1.1E-2 ± 0.4E-2	4.0 ± 0.1
		2-10	4.7E-3 ± 2.1E-3	5.3 ± 0.1
		2-15	4.9E-3 ± 1.1E-3	9.5 ± 0.2
		2-20	5.4E-3 ± 2.2E-3	23.6 ± 0.4
		3-2	3.7E-2 ± 2.9E-2	4.0 ± 0.1
		3-5	9.4E-3 ± 6.2E-3	4.4 ± 0.1
		3-10	6.1E-3 ± 3.2E-3	7.8 ± 0.1
		3-15	5.5E-3 ± 2.5E-3	26.7 ± 0.4
		3-20	7.9E-3 ± 3.4E-3	80.8 ± 0.3

In table 10.13, it is shown the comparison between the M-RBFNN model and WKNN models with different neighborhood size ($K = [1, 10]$). It can be observed that the best results are achieved for medium neighborhood size ($K = 4, 5$). Again, for all time series, the prediction accuracy of M-RBFNN model is superior to WKNN for any value of K. The key idea behind the WKNN is that similar input data values have similar output values [Sorjamaa et al., 2005] and, for the case of function approximation problems with a single input (logistic time series, see table 10.1), this idea works well. Since there is no "real" learning of data in WKNN models, the learning time is not shown.

TABLE 10.13: Comparison between M-RBFNN and WKNN models

TSP method	K	MSE_{test}		
		Logistic map	Hénon map	Mackey-Glass
M-RBFNN	-	**1.4E-4**	**9.2E-5**	**6.6E-5**
WKNN	1	6.3E-4	3.5E-4	3.8E-4
	2	5.3E-4	2.6E-4	1.9E-4
	3	2.6E-4	2.3E-4	1.7E-4
	4	2.6E-4	2.0E-4	1.4E-4
	5	2.7E-4	1.8E-4	1.6E-4
	6	3.3E-4	1.9E-4	1.5E-4
	7	3.8E-4	2.0E-4	1.8E-4
	8	4.5E-4	2.0E-4	2.0E-4
	9	5.8E-4	2.2E-4	2.7E-4
	10	6.5E-4	2.6E-4	2.4E-4

Finally, table 10.14 shows the comparison between the M-RBFNN model and ANFIS models with different number of membership functions, $n_{mf} = [2, 7]$, per input variable. This way, the number of rules in the Sugeno-type fuzzy inference system is given by $n_r = n_{mf}^d$ where d is the number of inputs for a given benchmark (see table 10.1). According to the results given, the superiority of the proposed M-RBFNN method in terms of prediction accuracy is proved in all the cases, except for Hénon time series when the number of membership functions 7. Only in this case, the prediction accuracy is slightly better than the one given by M-RBFNN. Although ANFIS models have been successfully applied to time series forecasting [Boyacioglu and Avci, 2010], one of their main drawbacks is their learning computational load, t_{learn}, when applied to function approximation problems with

medium-high number of inputs, such as the Mackey-Glass example. On the contrary, ANFIS models are quite fast when the number of inputs is low (Logistic and Hénon maps).

TABLE 10.14: Comparison between M-RBFNN and WKNN models

Example	TSP method	$n_{mf} - n_r$	MSE_{test}	t_{learn} (sec)
Logistic map	**M-RBFNN**	-	**1.4E-4**	**3.9 ± 0.1**
	ANFIS	2-2	1.1E-1	1.4E-2 ± 0.1E-2
		3-3	1.1E-2	7.9E-3 ± 0.1E-3
		4-4	4.0E-3	1.5E-2 ± 0.1E-2
		5-5	1.3E-3	1.2E-2 ± 0.1E-2
		6-6	6.0E-4	1.3E-2 ± 0.1E-2
		7-7	7.4E-4	1.6E-2 ± 0.1E-2
Hénon map	**M-RBFNN**	-	**9.2E-5**	**2.9 ± 0.1**
	ANFIS	2-4	3.7E-2	1.3E-2 ± 0.1E-2
		3-9	1.8E-3	2.4E-2 ± 0.1E-2
		4-16	8.2E-4	3.7E-2 ± 0.1E-2
		5-25	2.5E-4	5.7E-2 ± 0.1E-2
		6-36	1.7E-4	8.7E-2 ± 0.1E-2
		7-49	**7.6E-5**	**1.3E-1 ± 0.1E-1**
Mackey-Glass	**M-RBFNN**	-	**6.6E-5**	**2.1 ± 0.1**
	ANFIS	2-16	2.0E-3	4.8E-2 ± 0.1E-2
		3-81	3.8E-4	3.5E-1 ± 0.1E-1
		4-256	2.1E-3	3.0 ± 0.1
		5-625	1.7E-3	25.1 ± 0.1
		6-1296	1.3E-2	121.9 ± 0.1
		7-2401	9.4E-3	495.5 ± 0.1

According to the results given in this section, the effectiveness of the RBFNN model selected by the *MoSe* algorithm (M-RBFNN) has been demonstrated in terms of prediction accuracy when compared to other traditional methods for time series forecasting in the recent literatures such as ARMA, MLP, ANFIS and WKNN methodologies.

10.5.3.4 RBFNNs in time series competitions: MINCODA'09 and SICO'10

In [Florido et al., 2009b] and [Florido et al., 2009a], RBFNNs models are applied to time series competitions held at the *First International Workshop on Mining and Non-Conventional Data* (MINCODA'09) and the *III Simposio de Inteligencia Computacional* (SICO'10) respectively. The work described in [Florido et al.,

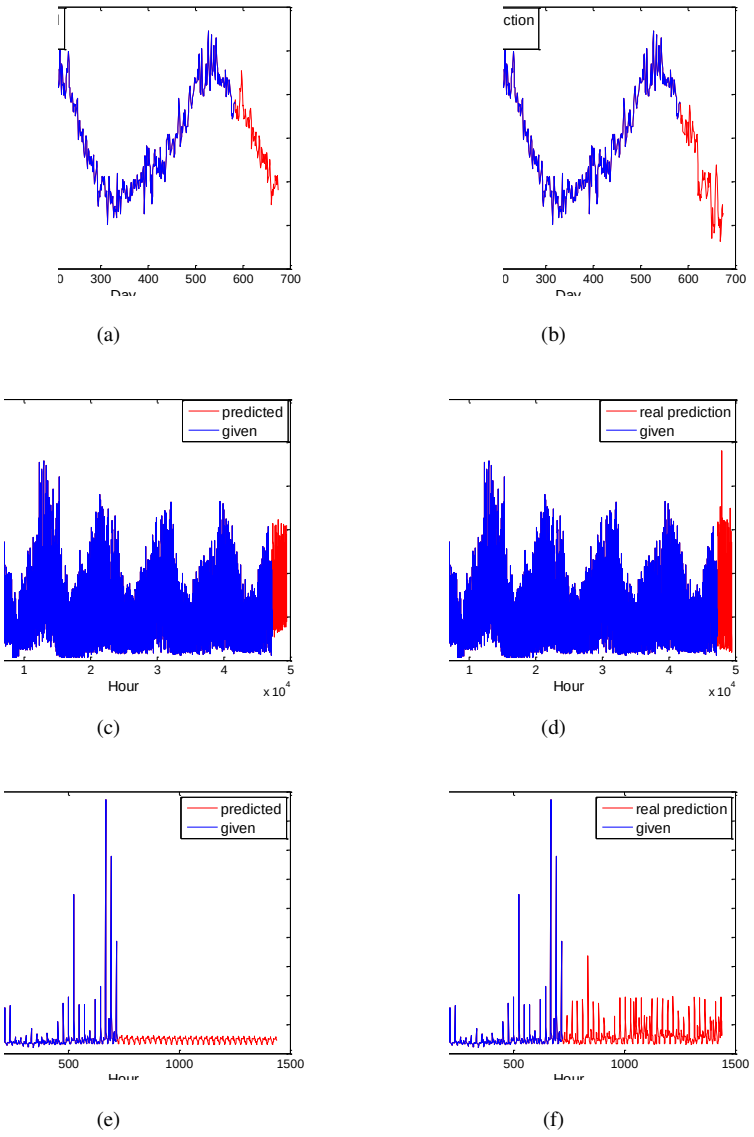
2009b] is emphasized in which a global methodology for the prediction of time series is proposed. This methodology is composed of an input variable selection criteria and the selection of the best RBFNN model for a forecasting task. The input variable selection is carried out through the concept of Mutual Information [Herrera et al., 2006] and the selection of the best RBFNN network is based on a slight variation of the MoSe method proposed in section 10.4. This variation is related to the deterministic generation of representative and balanced training and validation sets (section 10.4.1). Instead of using the NNO algorithm proposed, the training and validation sets are generated by means of a *Minimum Spanning Tree* (MST). MSTs have two important properties that make them suitable for the distribution of data into two sets: (1) they connect all the nodes with $n-1$ edges and (2) the node pairs defining the edges represent points that tend to be close together.

This methodology was applied to the Time Series Contest carried out at MINCODA'09, obtaining the best average forecasting for three time series: temperature, electricity prices and ozone levels (see Fig.10.4).

10.6 Conclusions and future work

Model selection approaches for incremental RBFNN construction based on *K-fold cross-validation* have some drawbacks: its random nature and the subjective decision for a proper value of K, resulting in large bias for low values and high variance and computational cost for high values. In order to prevent these problems, a deterministic model selection approach for incremental RBFNN construction applied to time series prediction problems has been proposed as a valid alternative from the point of view of bias and variance, reproducibility and computational complexity to those model selection approaches based on *K-fold cross-validation*. Given the inputs, the method selects the network structure (number of nodes in the hidden layers) and the model parameters (centers, radii and weights) for a given forecasting task.

The results of applying the proposed algorithm (MoSe) in different examples confirms that its performance, considering the prediction accuracy and the computational load, is better than other model selection approaches commonly used such



(a) (b) (c) (d) (e) (f)

FIGURE 10.4: Time series at MINCODA'09 contest. Predictions made by the methodology proposed (left) and real predictions (right) (a) Prediction for temperature; (b) Real prediction for temperature; (c)Prediction for ozone levels ; (d)Real prediction for ozone levels ; (e) Prediction for electricity prices; (f) Real prediction for electricity prices

as those based on *K-fold cross-validation* due to the following reasons: (1) it is deterministic; (2) the distribution of the learning set into two sets, training and validation, is accomplished taking into account the variability and the geometry of the learning set. This fact reduces the pessimistic effects caused by the removal of data from the original learning set since the distribution of the observations is representative and balanced. This distribution is used in all the steps related to the design of an RBFNN and it can be easily integrated in the Neural Network toolbox of MATLAB^{TM}software as an alternative to the functions that split randomly the data into training and validation sets and whose drawbacks were described in the introduction; (3) the computation time is reduced due to the use of only a validation set in the RBFNN optimization and model evaluation, instead of using a cross-validation procedure for evaluating a given RBFNN; (4) it is not necessary to choose a value for K, such as in *K-fold cross-validation* procedure, which is unknown a priori for RBFNN model selection. Moreover, the RBFNN model selected by the proposed methodology can be considered a valid alternative for time series forecasting to other linear and non-linear traditional prediction methods such ARMA, MLP, WKNN and ANFIS.

It must be emphasized that, since the proposed methodology is a combined algorithm, its main parts may be changed. For example, the RBFNN centers can be initialized using other methodologies such as Fuzzy C-means or the distribution of the learning set into training and validation sets may take into account not only the inputs, but also the output value. On the other hand, some of the parts may be integrated in other forecasting methodologies. For example, in ANFIS training, learning algorithms of neural networks are used and it might be expected an improvement in the prediction accuracy and a reduction of time when training through representative and balanced training and validation sets obtained from the deterministic distribution approach.

Chapter 11

Conclusions of the dissertation and list of publications

11.1 Conclusions

This section presents the general and most relevant conclusions of the dissertation, despite the specific conclusions of each contribution were previously given at the end of each corresponding chapter.

11.1.1 Part I. Intelligent Systems for Bioinformatics

This part has been devoted to the problem of predicting functional associations between proteins by means of the integration of evidences from heterogeneous biological data sources. After introducing the problem of predicting functional relationships between proteins (chapter 2) and some of the existing methodologies in the literature for such prediction task using data sources in isolation or by means of a proper integration of them (chapter 3), it was proposed a multi-objective genetic programming (MO-GP) approach to developing pareto optimal IF-THEN classification rules for the problem at hand (chapter 4).

To demonstrate the usefulness of our methodology, ten different evidences or sources to predict functional associations between proteins in *Saccharomyces Cerevisiae* organism were used. The results demonstrated that pareto optimal IF-THEN rules obtained by a MO-GP approach is an effective data integration methodology in terms of: (i) the handling of high unbalanced GSP and GSN sets that reflects the ratio of positive to negative examples in the application domain as closely as possible; (ii) accuracy, improving statistically the results given by the Naïve Bayesian model, (iii) interpretability, since an FLN is constructed through simple and understandable rules in which the evidences used for predicting functional associations are given and (iv) flexibility, since the decision maker does not have to specify partial preferences on the desired accuracy of the FLN, owing to the fact that covering the entire pareto, different FLNs are obtained, each one with a different level of accuracy. Although the computational cost of the proposed methodology is higher than other data integration approaches, such increase in the learning time is not dramatic, since the MO-GP method simultaneously evolves toward multiple pareto optimal rules or solutions (i.e. multiple FLNs). It must be also emphasized, that the MO-GP has been run in parallel architectures (computer cluster): several MO-GPs with different population size are run in different nodes of the cluster and for each MO-GP, different subpopulations of the total population are evaluated by the fitness function in parallel through several processors.

Due to the use of simple rules for predicting functional associations between proteins, the importance of the evidences used in the integration process can be extracted at any level of accuracy. Thus, we may conclude that co-expression (Co-exp) and co-ocurrence in PubMed abstracts (TextM) evidences and the ones related to the molecular function and cellular componentes in Gene Ontology (SSMF and SSCC respectively), have the best predictive power toward functional associations for any level of accuracy. Moreover, other evidences such as Gene Neighbor plays a key role in predicting associations for sparse and relative high confident FLNs. Providing the decision maker with the role of each evidence when functional associations are uncovered is a valuable information from the biological point of view.

Other important conclusions are related to the other methodologies applied to the integration of biological data for the prediction task. For example, multilayer perceptrons are very sensitive to the methodology used to introduce cost-sensitivity.

When the *minimization of the misclassification costs* is used, the neural network produces random predictions of functional relationships due to the presence of the majority of true positive and negative cases (GSP and GSN respectively) in the same interval of the input domain: the evidence score by itself cannot be used as a predictor for functional links between proteins. When *threshold-moving* is used as a cost-sensitivity approach, a multilayer perceptron has been proved to be effective for data integration, since by manipulating the outputs of the MLP, the problem of the distribution of protein pairs in the input domain is overcome. In the case of Naïve Bayes, it is a very fast methodology and does not suffer from the problem of the distribution of the protein pairs in the input domain previously described, since it measures the probability of observing the values in the evidence data set given that a pair of proteins are functionally related (GSP) divided by the probability of observing the values given that the pair is not functionally related (GSN). All of these alternative methodologies do not provide interpretable rules.

11.1.2 Part II. A2TOOL - Affymetrix microarray Analysis Tool

This part of the dissertation was related to microarray data analysis pipeline. After introducing the different steps involved in microarray data analysis (chapter 5) and the existing tools and metrics for such pipeline (chapter 6), the proposed tool for analyzing standard Affymetrix 3' expression arrays was explained in chapter 7. This tool has been developed for the first three steps of the pipeline: (1) automated quality assessment of the experiment to detect defective arrays according to quantitative and qualitative measures, (2) automated selection of the best pre-processing methods among tens of them for a given data set through objective quality metrics (replicate variability, Kolmogorov-Smirnov test and rank Spearman correlation coefficient) and (3) automated generation of confident and complete lists of differentially expressed genes according to the set of best pre-processing methods selected before.

The usefulness of the tool has been proved when applied to the Chronic Lymphocytic Leukemia (CLL) data set. Through this data set, a low quality array was automatically detected as defective in six quality metrics. Seven pre-processing methods (most of them standard in the literature and some of them custom) were

selected as the more suitable for the data set, discarding the pre-processing methods that did not fulfill the quality metrics or that introduced correlation artifacts in the pre-processed data. Confident and complete lists of differentially expressed genes were built according to this set of pre-processing methods, proving that the selection of a pre-processing method has an influence on the list of differentially expressed genes. As can be noticed, the whole process is automatic, so that the (non-expert) decision maker is aided. This automation (i) avoids to waste time for searching the proper quality assessment and pre-processing methods and (ii) reduces the possible errors in further analysis phases due to the presence of low quality arrays and/or incorrect choice of pre-processing methods.

According to these results, some other important conclusions can be drawn:

1. It is important to make the best use of the information produced by the microarray experiments by removing defective or low quality arrays if they are detected. This way, the results will have more statistical significance and biological meaning.

2. Some pre-processing methods introduce correlation artifacts on the data set even if they improve raw data. Thus, it is important to detect such pre-processing methods and avoid their use.

3. The performance of a pre-processing method is highly dependent on the selection of each pre-processing step. Moreover, the most critical steps are Background Correction and Normalization.

4. It is justified the use of two lists of differentially expressed genes: the intersection list provides a reliable list of differentially expressed genes regardless of the pre-processing method selected and the union list shows a complete list of candidate genes that are likely to be differentially expressed. This way, the set of candidate genes does not rely on a single expression data and the effect that the selection of a pre-processing method may have on the detection of differentially expressed genes is alleviated.

As demonstrated, the selection of a pre-processing method affects the detection of differentially expressed genes. We also assessed whether such selection may affect

other posterior stages of the microarray data analysis pipeline, such as phenotype classification. So, we studied the effect of a subset of standard pre-processing methods (RMA, VSN, dChip, MAS5 and GCRMA) on microarray-based SVM classifiers, demonstrating that there are no statistical differences among RMA, VSN, dChip and MAS5 pre-processing methods in terms of misclassification rate, but the GCRMA method shows the same performance, statistically speaking, as raw data. Through these results, we also demonstrated that SVM classifiers are sensitive to both feature selection and kernel function: when very few/large number of genes are selected or the polynomial kernel is chosen, SVM's accuracy goes down. On the other hand, well-tuned RBF kernels gave similar results to the linear ones.

11.1.3 Part III. Intelligent systems for function approximation

This part of the dissertation was devoted to the problem of distributing the original data set (input/output data) into two representative and balanced sets for function approximation tasks and the problem of model selection. An introduction to these problems were briefly described in chapter 8.

Chapter 9 presented a new data mining approach, NNO-CFA, for a distribution of a data set (input/output data) into two representative and balanced sets of roughly equal size to be used in function approximation problems with the aim of (i) allowing both a fair evaluation of learning's accuracy and (ii) making reproducible machine learning experiments usually based on random distributions. The main features of the NNO-CFA methodology are:

- It is based on a clustering procedure, the CFA, which is specially suited and adequate for function approximation problems due to the analysis of the output variability of the target function and, thus, its adaptation to the instance space with singularities during the clustering process.
- The distribution of the data values is made within clusters using the NNO approach, taking into account all data values contained in a cluster. This fact allows the study of the variability and the geometry of the data set locally,which produces a reduction of the input space and, therefore, an improvement of the distribution's quality.

The proposed methodology was assessed through several function approximation problems, such as nonlinear chaotic time series, real time series problems, artificial function approximation problems and nonlinear dynamic systems and compared to other existing methodologies in the literature. The results analyzed through an ANOVA test demonstrated the superiority of our approach according to three quality metrics: the *cross-validation error*, the *J-divergence* and the *average of generalization errors*.

We also demonstrated through the results that the random approach, which is commonly used in the literature for distributing the original data set into two groups, should not be recommended for this kind of problems due to the following reasons:

- It may lead to wrong conclusions about a learning algorithm due to both its random nature and the distribution itself, since the variability of the data set is not taken into account and may produce unrepresentative sets.
- Comparisons between performances of several learning algorithms in different experiments are difficult due to the need of using several random splits to get a reliable estimate of the quality of the learning algorithms. This is computationally expensive because it involves several repetitions of the random splitting and learning process.

The NNO-CFA method is freely available, together with the examples used in this dissertation, at the website http://atc.ugr.es/~hector/NNO-CFA/index.html.

Finally, chapter 10 presented a new deterministic model selection methodology for incremental RBFNN construction in time series prediction problems (MoSe). Such model selection approach is a combined algorithm which takes advantage of balanced and representative training and validation sets for their use in RBFNN initialization, optimization and network model evaluation.

The results of applying the proposed algorithm in different examples confirms that its performance, considering the prediction accuracy and the computational load, is better than other model selection approaches commonly used such as those based on *K-fold cross-validation* due to the following reasons: (1) it is deterministic; (2) the distribution of the learning set into two sets, training and validation, is

accomplished taking into account the variability and the geometry of the learning set. This fact reduces the pessimistic effects caused by the removal of data from the original learning set since the distribution of the observations is representative and balanced. This distribution is used in all the steps related to the design of an RBFNN and it can be easily integrated in the Neural Network toolbox of MATLAB^{TM}software as an alternative to the functions that split randomly the data into training and validation sets and whose drawbacks were described in the introduction; (3) the computation time is reduced due to the use of only a validation set in the RBFNN optimization and model evaluation, instead of using a cross-validation procedure for evaluating a given RBFNN; (4) it is not necessary to choose a value for K, such as in *K-fold cross-validation* procedure, which is unknown a priori for RBFNN model selection. Moreover, the RBFNN model selected by the proposed methodology can be considered a valid alternative for time series forecasting to other linear and non-linear traditional prediction methods such ARMA, MLP, WKNN and ANFIS.

11.2 List of publications

The different studies included in this dissertation have been presented in the following publications:

1. Florido, J.P., Pomares, Rojas, I., Urquiza, J.M, Ortuño, F. (2011). Prediction of functional associations between proteins by means of a cost-sensitive artificial neural network. *In Proceedings of the 11th International Work-Conference on Artificial Neural Networks, IWANN'11. Lecture Notes in Computer Science*, vol.6692, pp.194-201.

2. Florido, J.P., Pomares, H., Rojas, I. (2011). Generating balanced learning and test sets for function approximation problems. *International Journal of Neural Systems*, 21(3): 247-263.

3. Florido, J.P., Pomares H., Rojas, I., Lopez-Gordo, M., Urquiza, J.M. (2011). A deterministic model selection scheme for incremental RBFNN construction in time series forecasting. *Neural Computing & Applications*. In Press. DOI: 10.1007/s00521-010-0466-5

4. Florido, J.P., Claros, M., Pomares, H., Rojas, I. (2010). Ranking affymetrix microarray-based pre-processing methods. *In 10th Spanish Symposium on Bioinformatics*, Malaga, Spain, pp.226-229.

5. Florido, J.P., Pomares, Rojas, I., Urquiza, J.M, Herrera, L.J., Claros, M.G. (2010). Effect of pre-processing methods on microarray-based SVM classifiers in Affymetrix genechips. *In the 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, España*. pp.1-6

6. Florido, J.P., Pomares, Rojas, I., Urquiza, J.M. (2010). Applications of computational intelligence to the integration of heterogeneous biological data sources. *In III Simposio de Inteligencia Computacional, SICO'10. Congreso Español de Informática, CEDI 2010, Valencia, España*, pp.289-294.

7. Florido, J.P., Pomares, H., Herrera, L.J., Rojas, I. Urquiza, J.M.(2009). Adressing RBFNNs for time series prediction: inputs and network model selection. *In III Simposio de Inteligencia Computacional, SICO'10. Congreso Español de Informática, CEDI 2010, Valencia, España*, pp.229-236.

8. Florido, J.P., Pomares, H., Herrera, L.J, Rojas, I., Urquiza, J.M (2009). Time series prediction using mutual information and RBFNNs. *In I International Workshop on Mining and Non-Conventional Data (MINCODA'09), 13 Conference of the Spanish Association for Artificial Intelligence (CAEPIA'09)*, pp.25-32. **Winner of the time series prediction contest**

9. Florido, J.P., Pomares, Rojas, I., Calvo, J.C., Urquiza, J.M, Claros, M.G. (2009). On selecting the best pre-processing methods for affymetrix genechips. *In Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part I: Bio-Inspired Systems: Computational and Ambient Intelligence, IWANN 2009, Salamanca, España. Lecture Notes in Computer Science*, vol.5517, pp.845-852.

10. Florido, J.P., Pomares, Urquiza, J.M., Lopez-Gordo, M.A., Damas, M. (2007). Data Distribution for Cross-Validation using a Genetic Algorithm for Time Series Prediction. *In II Simposio de Ingeligencia Computacional, SICO 2007. Congreso Español de Informática, Zaragoza*, pp.373-380.

The following references are publications in which I have collaborated, but not as the main author. The research topics of these publications are related to time series prediction and other areas of Bioinformatics.

1. Urquiza, J., Rojas, I., Pomares, H., Herrera, L.J., Pérez-Florido, J. (2011). Using machine learning techniques and genomic/proteomic information for protein-protein classification. *In Termis EU 2011 Annual Meeting, Tissue Engineering & Regenerative Medicine International Society, Granada, España. Histology and Histopathology, Cellular and Molecular Biology, Vol.26, Sup. 1, pp.300.*

2. Ortuño, F., Rojas, I., Pomares, H., Urquiza, J.M., Florido, J.P. (2011). Emerging methodologies in multiple sequence alignment using high throughput data. *In 5th International Conference on Practical Applications of Computational Biology & Bioinformatics, Salamanca, España. Advances in Intelligent and Soft Computing*, Vol. 93, pp.184-190

3. Urquiza, J.M., Rojas, I., Pomares, H., Herrera, L.J., Florido, J.P., Ortuño, F. (2011). Using Machine Learning Techniques and Genomic/Proteomic Information from Known Databases for PPI prediction. *In 5th International Conference on Practical Applications of Computational Biology & Bioinformatics, Salamanca, España. Advances in Intelligent and Soft Computing*, Vol. 93, pp.373-380.

4. Caba, O., Álvarez-Aránega, P., Prados, J.C., Ortiz, R., Melguizo-Alonso, C., Rodriguez-Serrano, F., Delgado, J.R., Rojas, I. Pérez-Florido, J., Prieto, A., Aránega, A. (2010). Global gene expression profiling of pheripheral blood pancreas adenocarcinoma patients to determine potential biomarkers to treatment response. *MEDMOL Research Report*, DOI: 10.4428/MMRR.a. 201005003

5. Caba, O., Prados, J.C., Álvarez-Aránega, P., Rojas, I., Florido, J.P., Torres, C., Perales, S., Delgado, J.R., Linares, A., Aránega, A. (2010). Estudio de patrones de expresin gnica en sangre periférica de pacientes con adenocarcinoma de páncreas para determinar potenciales biomarcadores de respuesta a tratamiento. *In IX Jornadas Andaluzas "Salud Investiga", Cádiz*

6. Pomares, H., Rojas, I., Guillén, A., Herrera, L.J., González, J., Damas, M., Florido, J.P., Urquiza, J., Prieto, B., Garcia-Garcia, C., Use of Evolutionary Algorithms for Two-level Minimization of Boolean Functions. *In III Simposio de Inteligencia Computacional, SICO'10. Congreso Español de Informática, CEDI 2010, Valencia, España*, pp.223-226.

7. Herrera, L.J., Pomares, H., Rojas, I., Rubio, G., Guillén, A., Floriddo, J.P. (2010). Recursive Prediction for Long Term Time series Forecasting. *In Forecasting models: Methods & Applications, Gabriel Fung and Jia Zhu (editors), pp.125-144, iConcept Press, Australia*

8. López, M.A., Pomares, H., Pelayo, F. Urquiza, J., Pérez, J. (2009). Evidences of cognitive effects over auditory steady-state responses by means of artificial neural networks and its use in brain-computer interfaces. *Neurocomputing*, vol. 72, issue: 16-18, pp.3617-3623.

9. Urquiza, J.M., Rojas, I., Pomares, H., Herrera, L.J., Rubio, G., Florido, J.P. (2009). Prediction of Protein-Protein Interactions in Yeast using SVMs with Genomics/Proteomics Information and Feature Selection. *In 24th International Symposium on Computer and Information Sciences, ISCIS 2009*, North Cyprus, pp.645-650.

10. Urquiza, J.M., Rojas, I., Pomares, H. Rubio, G., Herrera, L.J., Calvo, J.C., Florido, J.P. (2009). Method for Prediction of Protein-Protein Interactions in Yeast using Genomics/Proteomics Information and Feature Selection. *In Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part I: Bio-Inspired Systems: Computational and Ambient Intelligence, IWANN 2009, Salamanca, España. Lecture Notes in Computer Science*, vol.5517, pp.853-860.

11. Calvo, J.C., Ortega, J., Anguita, M., Urquiza, J.M, Florido, J.P. (2009). Protein Structure Prediction by Evolutionary Multi-objective Optimization: Search Space Reduction by Using Rotamers. *In Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part I: Bio-Inspired Systems: Computational and Ambient Intelligence, IWANN 2009, Salamanca, España. Lecture Notes in Computer Science*, vol.5517, pp.861-868.

12. Urquiza, J.M., Pomares, Florido, J.P., Lopez-Gordo, M.A. (2007). Prediction of water consumption through GA-based variable selection method. *In II Simposio de Ingeligencia Computacional, SICO 2007. Congreso Español de Informãtica, Zaragoza*, pp.389-396.

Other publications not related to the topics covered in this dissertation that contributed to my training in Computer Architecture and Computer Technology are:

1. Pomares, H., Rojas, I., Guillén, A., González, J., Valenzuela, O., Florido, J.P., Urquiza, J.M, Cara, A.B. Lopez-Mansilla, L., Egea-Serrano, S. (2011). Desarrollo de un entorno integrado para un computador didáctico elemental para la asignatura de Fundamentos de Informática del nuevo grado en Ingeniería de Tecnologías de Telecomunicaciones. *In Enseñanza y aprendizaje de Ingeniería de Computadores*, vol. 1, pp.43-49.

2. Florido, J.P., Pomares, H., Rojas, I., Rodriguez-Diaz, F.J. (2009). CMUcam2 communication system design to facilitate the teaching in robotics: telesurveillance case study. *In V Internacional Conference on Multimedia and Information and Communication Technologies in Education Research, Lisbon, Portugal. Reflections and Innovations in Integrating ICT in Education*, Vol.2. pp.935-939

3. Florido, J.P., Pomares, H., Rojas, I., Ferrer-Garcia, J., (2009). How to make the teaching in robotics easier through the CMUcam2 communication system. *In I Jornadas Andaluzas de Informtica (JAI'09), Canillas del Aceituno, Mãlaga, España*, pp.66-70.

4. Pomares, H., Rojas, I., Guillén, A., Herrera, L.J., Rubio, G., Florido, J.P., Urquiza, J.M, Cara, A.B. Lopez-Mansilla, L., Egea-Serrano, S. (2009). Implementation of a didactic interpreter for CODE-2. *In V Internacional Conference on Multimedia and Information and Communication Technologies in Education Research, Lisbon, Portugal. Reflections and Innovations in Integrating ICT in Education*, Vol.2. pp.960-964.

5. Pomares, H., Garcia-Garcia, C., Rojas, I., Damas, M., González, J., Florido, J.P., Urquiza, J.M, Cara, A.B. Lopez-Mansilla, L., Egea-Serrano, S. (2009). Teaching Digital Systems Design with a New Didactic Environment through the Internet. *In V Internacional Conference on Multimedia and Information and Communication Technologies in Education Research, Lisbon, Portugal. Reflections and Innovations in Integrating ICT in Education*, Vol.2. pp.1021-1026.

Conclusiones de la Tesis Doctoral y lista de publicaciones

Estas conclusiones contienen una versión en español del Capítulo 11 y han sido incluídas para cumplir con los requerimientos necesarios para poder optar a la mención de *Doctorado Europeo*.

Conclusiones

Esta sección presenta las conclusiones generales y más relevantes de la Tesis Doctoral.

Parte I. Sistemas Inteligentes para Bioinformática

Este bloque de la tesis ha sido dedicado al problema de la predicción de relaciones funcionales entre proteínas mediante la integración de fuentes biológicas heterogéneas. Después de introducir el problema de la predicción de relaciones funcionales entre proteínas (capítulo 2) y algunas de las metodologías existentes en la literatura para dicha tarea de predicción utilizando fuentes de datos de forma aislada o por medio de una integración de ellas (capítulo 3), se ha propuesto una metodología basada en programación genética multi-objetivo (MO-GP) para obtener reglas pareto-óptimas de clasificación del tipo SI-ENTONCES para el problema de predicción (capítulo 4). Para demostrar la utilidad de la metodología

propuesta, diez fuentes de datos han sido utilizadas para prededir relaciones funcionales entre proteínas en el organismo de la levadura *Saccharomyces Cerevisiae*. Los resultados han demostrado que las reglas pareto-óptimas del tipo SI-ENTONCES obtenidas mediante una metodología MO-GP conforman una propuesta de integración de datos efectiva en términos de: (i) uso de conjuntos GSP y GSN no balanceados que reflejan la proporción de ejemplos positivos y negativos existentes en el dominio de aplicación; (ii) precisión, mejorando estadísticamente los resultados dados por el modelo Naïve Bayes; (iii) interpretabilidad, ya que la FLN es construída por medio de reglas sencillas e interpretables en las que las diferentes fuentes utilizadas en la tarea de predicción son proporcionadas y (iv) flexibilidad, ya que el investigador no tiene que especificar preferencias en el nivel de precisión de la FLN, debido a que, cubriendo el pareto completo, se obtienen diferentes FLNs, cada una con un nivel de precisión determinado. Aunque el coste computacional de la metodología propuesta es mayor que el de otras metodologías de integración, dicho incremento en el proceso de aprendizaje no es dramático, debido a que el método MO-GP evoluciona simultáneamente diferentes reglas o soluciones (es decir, múltiples FLNs). También hay que destacar que la metodología MO-GP ha sido ejecutada en arquitecturas paralelas (cluster de computadores) donde diversos MO-GPs con diferentes tamaños de la población son ejecutados en diferentes nodos del cluster y, para cada MO-GP, diferentes subpoblaciones de invididuos son evaluados por la función fitness en paralelo en diferentes procesadores.

Debido al uso de reglas sencillas para la predicción de relaciones funcionales entre proteínas, queda reflejada la importancia de las diversas fuentes utilizadas en el proceso de integración para cualquier nivel de precisión. Así, podemos concluir que las fuentes de datos relacionadas con la co-expresión (Co-exp), la co-ocurrencia de resúmenes en PubMed (TexM) y las relacionadas con las funciones moleculares y los componentes celulares en Gene Ontology (SSMF y SSCC respectivamente), tienen la mayor capacidad de predicción de relaciones funcionales para cualquier nivel de precisión. Es más, otras fuentes como *Gene neighbor* (GN), juegan un papel importante en la predicción de relaciones funcionales en FLNs pequeñas y precisas. Así, proporcionar este tipo de información al investigador es de un valor incalculable desde el punto de vista biológico.

Otras conclusiones importantes están relacionadas con otras metodologías aplicadas a la integración de fuentes biológicas heterogéneas para la tarea de predicción. Por ejemplo, los perceptrones multicapa son muy sensibles a la metodología utilizada para introducir costes de clasificación errónea. Cuando se utiliza la *minimización del coste de clasificación errónea* (*minimization of the misclassification cost* en inglés), la red neuronal produce predicciones de relaciones funcionales de forma aleatoria, debido a la presencia en el mismo intervalo del dominio de entrada de la mayoría de los casos de verdaderos positivos y verdaderos negativos (GSP y GSN respectivamente) de forma que el valor de la fuente por sí mismo no puede ser utilizado como evidencia para predecir relaciones funcionales entre proteínas. Cuando se utiliza *threshold-moving* como técnica para introducir costes de clasificación errónea, el perceptrón multicapa es efectivo en la integración de datos, debido a que se manipulan las salidas de la red neuronal, de forma que se evita el problema de la distribución de los pares de proteínas en el dominio de entrada. En el caso de Naïve Bayes, es una metodología muy rápida y no sufre del poblema de la distribución de los pares de proteínas en el dominio de entrada anteriormente comentado, ya que mide la probabilidad de observar los valores de una fuente de datos determinada dado que el par de proteínas está relacionado funcionalmente (GSP) dividido por la probabilidad de observar los valores dado que el par de proteínas no está relacionado funcionalmente (GSN). Todas estas metodologías alternativas no proporcionan reglas interpretables.

Parte II. A2TOOL - Herramienta de análisis de microarrays de Affymetrix

Este bloque de la tesis está relacionado con el análisis de datos de microarrays. Después de introducir las diferentes etapas involucradas en el análisis de datos de microarrays (capítulo 5) y las diferentes herramientas y métricas para dicho análisis (capítulo 6), la herramienta propuesta para analizar microarrays del tipo Affymetrix 3' es descrita con detalle en el capítulo 7. Esta herramienta ha sido desarrollada para las primeras tres etapas del análisis: (1) evaluación automática de la calidad del experimento para detectar microarrays de baja calidad de acuerdo a diversas métricas cuantitativas y cualitativas, (2) selección automática de los mejores métodos de pre-procesamiento para un conjunto de datos determinado haciendo

uso de métricas de calidad objetivas (variabilidad de réplicas, test de Kolmogorov-Smirnov y coeficiente de correlación de Spearman) y (3) generación automática de listas fiables y completas de genes expresados diferencialmente de acuerdo a los mejores métodos de pre-procesamiento seleccionados anteriormente. La utilidad de la herramienta ha sido demostrada mediante su aplicación a los datos *Chronic Lymphocytic Leukemia*, CLL. A través de este conjunto de datos, un microarray de baja calidad fue detectado automáticamente en seis métricas de calidad. Siete métodos de pre-procesamiento (la mayoría de ellos estándar en la literatura y algunos de ellos personalizados) fueron seleccionados como los más apropiados para el conjunto de datos, descartando los métodos de pre-procesamiento que no cumplían con las métricas de calidad o que introducían efectos artificiales de correlación en los datos pre-procesados. Por otro lado, la herramienta proporcionó listas fiables y completas de genes expresados diferencialmente de acuerdo a los métodos de pre-procesamiento seleccionados anteriormente, demostrando que los métodos de pre-procesamiento tienen influencia en la lista de genes expresados diferencialmente. Como se ha podido comprobar, el proceso completo es automático, de forma que el usuario, probablemente no experto, es ayudado en todo momento. Esta automatización (i) evita la pérdida de tiempo en la búsqueda de los métodos más apropiados para el análisis de calidad y el pre-procesamiento y (ii) reduce los posibles errores presentes en etapas posteriores del análisis de microarrays, como la clasificación, debidos a la presencia de microarrays de baja calidad y/o una selección incorrecta de los métodos de pre-procesamiento.

De acuerdo a estos resultados, podemos extraer algunas conclusiones interesantes:

- Es importante hacer el mejor uso posible de la información producida por los experimentos de microarrays, eliminando microarrays de baja calidad si éstos son detectados. De esta forma, los resultados tendrán más significado desde el punto de vista biológico.
- Algunos métodos de pre-procesamiento introducen efectos artificiales de correlación en el conjunto de datos, incluso mejorando a los datos crudos. Por tanto, es esencial detectar dichos métodos de pre-procesamiento y evitar su uso.

- El rendimiento de un método de pre-procesamiento es altamente dependiente de la selección de cada etapa de pre-procesamiento. Es más, las etapas más críticas corresponden a la normalización y a la correción del ruido de fondo.

- Está justificado el uso de dos listas de genes expresados diferencialmente: la lista *intersección* que proporciona una lista fiable de genes expresados diferencialmente independientemente del método de pre-procesamiento seleccionado y la lista *unión*, que muestra una lista completa de genes candidatos que, con mucha probabilidad, están expresados diferencialmente. De esta forma, el conjunto de genes candidatos no se basa en un sólo conjunto de expresión y el efecto que los métodos de pre-procesamiento tienen en la detección de genes expresados diferencialmente es reducido.

Como se ha podido demostrar, la selección de un método de pre-procesamiento afecta a la detección de los genes expresados diferencialmente. También se ha estudiado si dicha selección tiene un efecto en etapas posteriores del análisis de microarrays, como la clasificación. Así, hemos estudiado el efecto de un subconjunto de métodos de pre-procesamiento estándar (RMA, VSN, dChip, MAS5 y GCRMA) en clasificadores de microarrays basados en Máquinas de Vector Soporte (*Support Vector Machine*, SVM, en inglés), demostrando que no existen diferencias significativas entre RMA, VSN, dChip y MAS5 en términos de errores cometidos en la clasificación. Sin embargo, se pudo demostrar que el método GCRMA muestra el mismo rendimiento, estadísticamente hablando, que los datos crudos. A través de estos resultados, demostramos también que los clasificadores SVM son sensibles a la selección de genes y a la función kernel: cuando son seleccionados muy pocos o muchos genes o se escoge un kernel de tipo polimonial, la precisión de SVM desciende. Por otro lado, kernels basados en funciones de base radial proporcionan resultados similares a los kernels de tipo lineal.

Parte III. Sistemas inteligentes para aproximación funcional

Este bloque de la tesis ha sido dedicado al problema de la distribución de un conjunto original de datos de entrada/salida en dos conjuntos representativos y balanceados para aproximación funcional y al problema de la selección del modelo.

Una introducción a estos problemas se proporcionó en el capítulo 8.

El capítulo 9 presentó una nueva metodología denominada NNO-CFA para la distribución de un conjunto de datos de entrada/salida en dos conjuntos representativos y balanceados de, aproximadamente, el mismo tamaño para ser utilizados en problemas de aproximación funcional con el objetivo de (i) permitir una evaluación justa de la precisión de un algoritmo de aprendizaje y (ii) realizar experimentos reproducibles de aprendizaje normalmente basados en distribuciones aleatorias. Las principales características de la metodología NNO-CFA son:

- Está basada en un procedimiento de clustering, el CFA, que está especialmente diseñado para problemas de aproximación funcional debido al análisis de la variabilidad de la salida de la función y, por tanto, su adaptación al espacio con singularidades durante el proceso de clustering.
- La distribución de los datos se realiza para cada cluster por medio del algoritmo NNO, teniendo en cuenta todos los datos contenidos en un cluster. Este hecho permite el estudio de la variabilidad y la geometría de los datos de forma local, proporcionando una reducción del espacio de entrada y, por lo tanto, una mejora en la calidad de la distribución.

La metodología propuesta ha sido evaluada mediante numerosos problemas de aproximación funcional, como series temporales caóticas no lineales, series temporales reales, problemas de aproximación funcional artificiales o sistemas dinámicos no lineales. Además, la metodología ha sido comparada con otras propuestas en la literatura. Los resultados fueron analizados a través de un test ANOVA demostrando la superioridad de la metodología propuesta de acuerdo a tres métricas de calidad: el *error de validación cruzada*, la *J-divergencia* y el *error medio de generalización*.

También demostramos que la metodología de distribución aleatoria comúnmente utilizada en la literatura para distribuir el conjunto de datos original en dos conjuntos, no es recomendable para este tipo de problemas debido a la siguientes razones:

- Puede dar lugar a conclusiones erróneas sobre el algoritmo de aprendizaje, debido a su naturaleza aleatoria y a la propia distribución, ya que la variabilidad del conjunto de datos no es tenida en cuenta y puede, por lo tanto, producir conjuntos no representativos.

- Las comparaciones entre el rendimiento obtenido por diferentes algoritmos de aprendizaje en diferentes experimentos son difiíciles debido a la necesidad de utilizar numerosas particiones aleatorias para obtener una estimación representativa de la calidad de los algoritmos de aprendizaje. Esto es computacionalmente costoso ya que implica numerosas repeticiones del particionamiento o distribución aleatoria y del proceso de aprendizaje.

El método NNO-CFA está disponible de forma libre junto con los ejemplos utilizados en esta tesis, en el sitio web http://atc.ugr.es/˜hector/NNO-CFA/index.html.

Finalmente, el capítulo 10 presentó una nueva metodología de selección del modelo para la construcción de RBFNNs incremental en problemas de predicción de series temporales denominado *MoSe*. Dicha metodología de selección del modelo es un algoritmo combinado que aprovecha conjuntos de entrenamiento y validación balanceados y representativos para su uso en la inicialización de la RBFNN, su optimización y en la evaluación del modelo de red.

Los resultados de aplicar el algoritmo propuesto en diferentes ejemplos confirman que su rendimiento, considerando la capacidad de predicción y el coste computacional, es mejor que otras metodologías de selección del modelo comúnmente utilizadas en la literatura, como las basadas en la *validación cruzada K-veces*, debido a los siguientes motivos: (1) es determinista; (2) la distribución del conjunto de aprendizaje en dos conjuntos, entrenamiento y validación, se lleva a cabo teniendo en cuenta la variabilidad y la geometría del conjunto de aprendizaje. Este hecho reduce los efectos pesimistas causados por la eliminación de datos del conjunto de aprendizaje debido a que la distribución de los datos es representativa y balanceada. Esta distribución es utilizada en todas las etapas relacionadas con el diseño de la RBFNN y puede ser fácilmente integrada en la toolbox de Redes Neuronales del software de MATLAB^{TM}como una alternativa a las funciones que dividen aleatoriamente el conjunto de datos en entrenamiento y validación y cuyas desventajas

ya fueron descritas en la introducción; (3) el tiempo de cómputo es reducido debido al uso de un solo conjunto de validación en la optimización y en la evaluación de la RBFNN; (4) no es necesario elegir un valor para K, como en los procedimientos basados en la *validación cruzada K-veces*. El valor de K es desconocido a priori para la selección del mejor modelo de RBFNN. Es más, el modelo de RBFNN seleccionado por la metodología propuesta puede ser considerado una alternativa válida para predicción de series temporales comparada con otros métodos de predicción, como ARMA, MLP, WKNN y ANFIS.

Lista de publicaciones

Los diferentes estudios incluídos en esta tesis doctoral han sido presentados en las siguientes publicaciones:

1. Florido, J.P., Pomares, Rojas, I., Urquiza, J.M, Ortuño, F. (2011). Prediction of functional associations between proteins by means of a cost-sensitive artificial neural network. *In Proceedings of the 11th International Work-Conference on Artificial Neural Networks, IWANN'11. Lecture Notes in Computer Science*, vol.6692, pp.194-201.

2. Florido, J.P., Pomares, H., Rojas, I. (2011). Generating balanced learning and test sets for function approximation problems. *International Journal of Neural Systems*, 21(3): 247-263.

3. Florido, J.P., Pomares H., Rojas, I., Lopez-Gordo, M., Urquiza, J.M. (2011). A deterministic model selection scheme for incremental RBFNN construction in time series forecasting. *Neural Computing & Applications*. In Press. DOI: 10.1007/s00521-010-0466-5

4. Florido, J.P., Claros, M., Pomares, H., Rojas, I. (2010). Ranking affymetrix microarray-based pre-processing methods. *In 10th Spanish Symposium on Bioinformatics*, Malaga, Spain, pp.226-229.

5. Florido, J.P., Pomares, Rojas, I., Urquiza, J.M, Herrera, L.J., Claros, M.G. (2010). Effect of pre-processing methods on microarray-based SVM classifiers in Affymetrix genechips. *In the 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, España*. pp.1-6

6. Florido, J.P., Pomares, Rojas, I., Urquiza, J.M. (2010). Applications of computational intelligence to the integration of heterogeneous biological data sources. *In III Simposio de Inteligencia Computacional, SICO'10. Congreso Español de Informática, CEDI 2010, Valencia, España*, pp.289-294.

7. Florido, J.P., Pomares, H., Herrera, L.J., Rojas, I. Urquiza, J.M.(2009). Adressing RBFNNs for time series prediction: inputs and network model selection. *In III Simposio de Inteligencia Computacional, SICO'10. Congreso Español de Informática, CEDI 2010, Valencia, España*, pp.229-236.

8. Florido, J.P., Pomares, H., Herrera, L.J, Rojas, I., Urquiza, J.M (2009). Time series prediction using mutual information and RBFNNs. *In I International Workshop on Mining and Non-Conventional Data (MINCODA'09), 13 Conference of the Spanish Association for Artificial Intelligence (CAEPIA'09)*, pp.25-32. **Ganador del concurso de predicción de series temporales**

9. Florido, J.P., Pomares, Rojas, I., Calvo, J.C., Urquiza, J.M, Claros, M.G. (2009). On selecting the best pre-processing methods for affymetrix genechips. *In Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part I: Bio-Inspired Systems: Computational and Ambient Intelligence, IWANN 2009, Salamanca, España. Lecture Notes in Computer Science*, vol.5517, pp.845-852.

10. Florido, J.P., Pomares, Urquiza, J.M., Lopez-Gordo, M.A., Damas, M. (2007). Data Distribution for Cross-Validation using a Genetic Algorithm for Time Series Prediction. *In II Simposio de Ingeligencia Computacional, SICO 2007. Congreso Español de Informática, Zaragoza*, pp.373-380.

Las siguientes referencias son publicaciones en las que he colaborado como coautor. Las áreas de investigación de las siguientes publicaciones están relacionadas con la predicción de series temporales y otros campos de la Bioinformática:

1. Urquiza, J., Rojas, I., Pomares, H., Herrera, L.J., Pérez-Florido, J. (2011). Using machine learning techniques and genomic/proteomic information for protein-protein classification. *In Termis EU 2011 Annual Meeting, Tissue Engineering & Regenerative Medicine International Society, Granada, España. Histology and Histopathology, Cellular and Molecular Biology, Vol.26, Sup. 1, pp.300.*

2. Ortuño, F., Rojas, I., Pomares, H., Urquiza, J.M., Florido, J.P. (2011). Emerging methodologies in multiple sequence alignment using high throughput data. *In 5th International Conference on Practical Applications of Computational Biology & Bioinformatics, Salamanca, España. Advances in Intelligent and Soft Computing*, Vol. 93, pp.184-190

3. Urquiza, J.M., Rojas, I., Pomares, H., Herrera, L.J., Florido, J.P., Ortuño, F. (2011). Using Machine Learning Techniques and Genomic/Proteomic Information from Known Databases for PPI prediction. *In 5th International Conference on Practical Applications of Computational Biology & Bioinformatics, Salamanca, España. Advances in Intelligent and Soft Computing*, Vol. 93, pp.373-380.

4. Caba, O., Álvarez-Aránega, P., Prados, J.C., Ortiz, R., Melguizo-Alonso, C., Rodriguez-Serrano, F., Delgado, J.R., Rojas, I. Pérez-Florido, J., Prieto, A., Aránega, A. (2010). Global gene expression profiling of pheripheral blood pancreas adenocarcinoma patients to determine potential biomarkers to treatment response. *MEDMOL Research Report*, DOI: 10.4428/MMRR.a. 201005003

5. Caba, O., Prados, J.C., Álvarez-Aránega, P., Rojas, I., Florido, J.P., Torres, C., Perales, S., Delgado, J.R., Linares, A., Aránega, A. (2010). Estudio de patrones de expresin gnica en sangre periférica de pacientes con adenocarcinoma de páncreas para determinar potenciales biomarcadores de respuesta a tratamiento. *In IX Jornadas Andaluzas "Salud Investiga", Cádiz*

6. Pomares, H., Rojas, I., Guillén, A., Herrera, L.J., González, J., Damas, M., Florido, J.P., Urquiza, J., Prieto, B., Garcia-Garcia, C., Use of Evolutionary Algorithms for Two-level Minimization of Boolean Functions. *In III*

Simposio de Inteligencia Computacional, SICO'10. Congreso Español de Informática, CEDI 2010, Valencia, España, pp.223-226.

7. Herrera, L.J., Pomares, H., Rojas, I., Rubio, G., Guillén, A., Floriddo, J.P. (2010). Recursive Prediction for Long Term Time series Forecasting. *In Forecasting models: Methods & Applications, Gabriel Fung and Jia Zhu (editors), pp.125-144, iConcept Press, Australia*

8. López, M.A., Pomares, H., Pelayo, F. Urquiza, J., Pérez, J. (2009). Evidences of cognitive effects over auditory steady-state responses by means of artificial neural networks and its use in brain-computer interfaces. *Neurocomputing*, vol. 72, issue: 16-18, pp.3617-3623.

9. Urquiza, J.M., Rojas, I., Pomares, H., Herrera, L.J., Rubio, G., Florido, J.P. (2009). Prediction of Protein-Protein Interactions in Yeast using SVMs with Genomics/Proteomics Information and Feature Selection. *In 24th International Symposium on Computer and Information Sciences, ISCIS 2009*, North Cyprus, pp.645-650.

10. Urquiza, J.M., Rojas, I., Pomares, H. Rubio, G., Herrera, L.J., Calvo, J.C., Florido, J.P. (2009). Method for Prediction of Protein-Protein Interactions in Yeast using Genomics/Proteomics Information and Feature Selection. *In Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part I: Bio-Inspired Systems: Computational and Ambient Intelligence, IWANN 2009, Salamanca, España. Lecture Notes in Computer Science*, vol.5517, pp.853-860.

11. Calvo, J.C., Ortega, J., Anguita, M., Urquiza, J.M, Florido, J.P. (2009). Protein Structure Prediction by Evolutionary Multi-objective Optimization: Search Space Reduction by Using Rotamers. *In Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part I: Bio-Inspired Systems: Computational and Ambient Intelligence, IWANN 2009, Salamanca, España. Lecture Notes in Computer Science*, vol.5517, pp.861-868.

12. Urquiza, J.M., Pomares, Florido, J.P., Lopez-Gordo, M.A. (2007). Prediction of water consumption through GA-based variable selection method. *In*

II Simposio de Ingeligencia Computacional, SICO 2007. Congreso Español de Informãtica, Zaragoza, pp.389-396.

Otras publicaciones no relacionadas con las líneas de investigación de esta tesis doctoral pero que contribuyeron a mi formación en la Arquitectura de Computadores y la Tecnología de Computadores:

1. Pomares, H., Rojas, I., Guillén, A., González, J., Valenzuela, O., Florido, J.P., Urquiza, J.M, Cara, A.B. Lopez-Mansilla, L., Egea-Serrano, S. (2011). Desarrollo de un entorno integrado para un computador didáctico elemental para la asignatura de Fundamentos de Informática del nuevo grado en Ingeniería de Tecnologías de Telecomunicaciones. *In Enseñanza y aprendizaje de Ingeniería de Computadores*, vol. 1, pp.43-49.

2. Florido, J.P., Pomares, H., Rojas, I., Rodriguez-Diaz, F.J. (2009). CMUcam2 communication system design to facilitate the teaching in robotics: telesurveillance case study. *In V Internacional Conference on Multimedia and Information and Communication Technologies in Education Research, Lisbon, Portugal. Reflections and Innovations in Integrating ICT in Education*, Vol.2. pp.935-939

3. Florido, J.P., Pomares, H., Rojas, I., Ferrer-Garcia, J., (2009). How to make the teaching in robotics easier through the CMUcam2 communication system. *In I Jornadas Andaluzas de Informtica (JAI'09), Canillas del Aceituno, Mãlaga, España*, pp.66-70.

4. Pomares, H., Rojas, I., Guillén, A., Herrera, L.J., Rubio, G., Florido, J.P., Urquiza, J.M, Cara, A.B. Lopez-Mansilla, L., Egea-Serrano, S. (2009). Implementation of a didactic interpreter for CODE-2. *In V Internacional Conference on Multimedia and Information and Communication Technologies in Education Research, Lisbon, Portugal. Reflections and Innovations in Integrating ICT in Education*, Vol.2. pp.960-964.

5. Pomares, H., Garcia-Garcia, C., Rojas, I., Damas, M., González, J., Florido, J.P., Urquiza, J.M, Cara, A.B. Lopez-Mansilla, L., Egea-Serrano, S. (2009). Teaching Digital Systems Design with a New Didactic Environment through

the Internet. *In V Internacional Conference on Multimedia and Information and Communication Technologies in Education Research, Lisbon, Portugal. Reflections and Innovations in Integrating ICT in Education*, Vol.2. pp.1021-1026.

Appendix A

Supplementary material for Naïve Bayes data integration

A.1 Contingency tables for Naïve Bayes

In this appendix, the LLR value for each bin over all evidences assembled (a total of 10) by the Naïve Bayes data integration methodology is calculated. LLR values for discrete evidences (PPI and GI) were calculated according to equation 4.4 and LLR values for continuous evidences (Co-exp, GF, PP, GN, SS, TextM, SSMF and SSCC) were calculated according to equation 4.5. Contingency tables for these evidences are in tables A.1 - A.10.

TABLE A.1: Contingency table detailing the intersection of predicted functional linkages with the GSP and GSN sets and the resultant log-likelihood ratios for the Co-exp evidence

Bin	GSP	GSN	TOTAL	$P(bin\|GSP)$	$P(bin\|GSN)$	LLR
$[0.1, 0.2[$	1624	4673	22545	0.034822884	0.0047023095	2.002221
$[0.2, 0.4[$	3452	5142	25074	0.074020070	0.0051742511	2.660641
$[0.4, 0.6[$	1390	1038	5957	0.029805301	0.0010445104	3.35113
$[0.6, 0.7[$	460	197	1332	0.009863624	0.0001982356	3.907152
$[0.7, 0.8[$	440	132	887	0.009434771	0.0001328279	4.263102
$[0.8, 0.9[$	323	42	457	0.006925979	4.2263428e-005	5.099112
$[0.9, 1]$	891	30	1093	0.019105412	3.0188163e-005	6.450277

TABLE A.2: Contingency table detailing the intersection of predicted functional linkages with the GSP and GSN sets and the resultant log-likelihood ratios for the PPI evidence

Interaction	GSP	GSN	TOTAL	$P(bin\|GSP)$	$P(bin\|GSN)$	LLR
Yes	4756	3804	49531	0.1019	0.0038	3.2824
No	41880	989963	12840972	0.8980	0.9961	-0.1037

TABLE A.3: Contingency table detailing the intersection of predicted functional linkages with the GSP and GSN sets and the resultant log-likelihood ratios for the GI evidence

Interaction	GSP	GSN	TOTAL	$P(bin\|GSP)$	$P(bin\|GSN)$	LLR
Yes	2954	8683	104344	0.0633	0.00873	1.9809
No	43682	985084	12786159	0.9366	0.99126	-0.0566

TABLE A.4: Contingency table detailing the intersection of predicted functional linkages with the GSP and GSN sets and the resultant log-likelihood ratios for the GF evidence

Bin	GSP	GSN	TOTAL	$P(bin\|GSP)$	$P(bin\|GSN)$	LLR
$[0, 0.2[$	46559	993743	12895237	0.99834892	0.99997585	-0.001628
$[0.2, 0.4[$	21	14	110	0.00045029	1.40878e-005	3.4645951
$[0.4, 0.6[$	19	4	71	0.00040741	4.02509e-006	4.6172746
$[0.6, 1[$	37	6	163	0.00079337	6.03763e-006	4.8782884

TABLE A.5: Contingency table detailing the intersection of predicted functional linkages with the GSP and GSN sets and the resultant log-likelihood ratios for the PP evidence

Bin	GSP	GSN	TOTAL	$P(bin\|GSP)$	$P(bin\|GSN)$	LLR
$[0, 0.1[$	46377	993566	12893883	0.99444635	0.99979774	-0.005367
$[0.1, 0.3[$	156	174	1419	0.00334505	0.00017509	2.9499307
$[0.3, 0.4[$	44	23	170	0.00094347	2.31443e-005	3.7078254
$[0.4, 0.45[$	17	2	44	0.00036452	2.01254e-006	5.1991962
$[0.45, 1]$	42	2	65	0.00090059	2.01254e-006	6.1036524

TABLE A.6: Contingency table detailing the intersection of predicted functional linkages with the GSP and GSN sets and the resultant log-likelihood ratios for the GN evidence

Bin	GSP	GSN	TOTAL	$P(bin\|GSP)$	$P(bin\|GSN)$	LLR
[0, 0.1[	44878	988784	12882766	0.9623038	0.99498575	-0.033398
[0.1, 0.3[	711	3343	8098	0.0152457	0.00336396	1.5111785
[0.3, 0.4[	165	732	1725	0.0035380	0.00073659	1.569295
[0.4, 0.5[	122	400	979	0.0026160	0.00040250	1.8716865
[0.5, 0.6[	129	234	620	0.0027661	0.00023546	2.4636213
[0.6, 0.7[	207	149	546	0.0044386	0.00014993	3.3879025
[0.7, 0.8[	352	120	708	0.0075478	0.00012075	4.1352694
[0.8, 1]	72	5	139	0.0015438	5.03136e-006	5.7263582

TABLE A.7: Contingency table detailing the intersection of predicted functional linkages with the GSP and GSN sets and the resultant log-likelihood ratios for the SS evidence

Bin	GSP	GSN	TOTAL	$P(bin\|GSP)$	$P(bin\|GSN)$	LLR
[0,1e-250[	137	6	274	0.0029	6.037e-006	6.187
[1e-250,1e-150[	52	3	110	0.0011	3.018e-006	5.911
[1e-150,1e-50[	210	14	825	0.0045	1.408e-005	5.767
[1e-50,1e-1[	244	147	2175	0.0052	0.00015	3.565
[1e-1,1]	45993	993597	12887119	0.9862	0.9998	-0.014

TABLE A.8: Contingency table detailing the intersection of predicted functional linkages with the GSP and GSN sets and the resultant log-likelihood ratios for the TextM evidence

Bin	GSP	GSN	TOTAL	$P(bin\|GSP)$	$P(bin\|GSN)$	LLR
[0, 0.1[	39533	984436	12833065	0.84769277	0.99061048	-0.155803
[0.1, 0.3[	2477	5257	22859	0.05311347	0.00528997	2.3066176
[0.3, 0.4[	742	1341	8818	0.01591045	0.00134941	2.4673084
[0.4, 0.5[	818	1247	9806	0.01754009	0.00125482	2.6374964
[0.5, 0.6[	715	653	7620	0.01533150	0.00065709	3.1498354
[0.6, 0.7[	547	356	4746	0.01172913	0.00035823	3.4886481
[0.7, 0.8[	438	210	2477	0.00939188	0.00021131	3.7942414
[0.8, 0.9[	397	174	2859	0.00851273	0.00017509	3.884011
[0.9, 1]	969	93	3331	0.02077794	9.35833e-005	5.4027951

TABLE A.9: Contingency table detailing the intersection of predicted functional linkages with the GSP and GSN sets and the resultant log-likelihood ratios for the SSMF evidence

Bin	GSP	GSN	TOTAL	$P(bin\|GSP)$	$P(bin\|GSN)$	LLR
[0, 10[	586	340	3310	0.0125654	0.0003421	3.6035042
[10, 50[	2587	3187	33423	0.0554721	0.0032069	2.8505489
[50, 100[	1869	6475	38291	0.0400763	0.0065156	1.8165849
[100, 500[	7951	37228	293664	0.1704906	0.0374614	1.5153665
[500, 1000[	2667	66533	474884	0.0571875	0.0669503	-0.157613
[1000, 5079[	30976	880004	12052009	0.6642079	0.8855234	-0.287583

TABLE A.10: contingency table detailing the intersection of predicted functional linkages with the GSP and GSN sets and the resultant log-likelihood ratios for the SSCC evidence

Bin	GSP	GSN	TOTAL	$P(bin\|GSP)$	$P(bin\|GSN)$	LLR
[0, 10[	417	18	2222	0.0089415	1.81129e-005	6.201844
[10, 50[	2860	1060	24243	0.0613260	0.0010666	4.051682
[50, 100[	3704	2949	61429	0.0794236	0.0029674	3.287077
[100, 500[	7955	40232	508104	0.1705763	0.0404843	1.438268
[500, 1000[	5910	100254	969801	0.1267261	0.1008828	0.228068
[1000, 5079[	25790	849254	11329782	0.5530062	0.8545806	-0.435241

Appendix B

Supplementary material for A^2TOOL

B.1 Preprocessing methods used by the A^2TOOL

In this appendix, pre-processing methods used by A^2**TOOL** are presented. Table B.1 shows such pre-processing methods in terms of:

- The method used in each step of pre-processing: Background correction, Normalization, PM correction and Summarization.
- The name of the pre-processing method known in the literature. Some pre-processing methods are standard in the literature for a specific combination of background correction, normalization, PM correction and summarization methods (l.farms, justPlier, mmgmos, MBEI(PM-MM model), MBEI(PM only model), AD, DFW, VSN, GCRMA, RMA, MAS5 and CP). The non-standard pre-processing methods are pointed out as ”custom”.
- The name of the function by which the pre-processing method is called within R.
- The Bioconductor package that implements such pre-processing method.

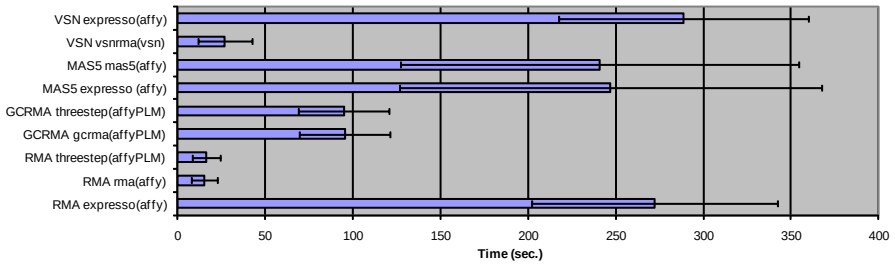


FIGURE B.1: Average and standard deviation running time of R functions running the same pre-processing method in several Bioconductor packages. Five different data sets are used and y axes show the standard name of the preprocessing method, followed by the R function that implements such method and the Bioconductor package[Florido et al., 2009c].

In some cases, one can run the same pre-processing method (that is, the same combination of background correction, normalization, PM correction and summarization procedures) using different R functions in different Bioconductor packages. For example, *expresso*, *rma* (affy package) and *threestep* (affyPLM package) R functions may run the same standard pre-processing method (RMA). When such situation arises, the function with the lowest computational cost is chosen. For example, in Fig. B.1, VSN, MAS5 and GCRMA standard pre-processing methods can be run using two different R functions each and RMA standard pre-processing method can be run using three different R functions in different Bioconductor packages [Florido et al., 2009c].

TABLE B.1: Preprocessing methods used by A^2**TOOL**

Background correction	**Normalization**	**PM correction**	**Summarization**	**known as**	**R function**	**Package**
none	none	pmonly	medianpolish	RAW	threestep	affyPLM
none	loess	pmonly	farms	l.farms	l.farms	farms
-	-	-	-	justPlier	justplier	plier
-	-	-	-	mmgmos	mmgmos	puma
none	invariantset	subtractmm	liwong	MBEI (PM-MM model)	expresso	AFFY
none	invariantset	subtractmm	avgdiff	AD	expresso	AFFY
none	quantiles	pmonly	dfw	DFW	expresso	AFFY
-	vsn	pmonly	medianpolish	VSN	vsnrma	VSN
gcrma	scaling	pmonly	tukey.biweight	custom 1	threestep	AFFYPLM
gcrma	scaling	pmonly	median.polish	custom 2	threestep	AFFYPLM
gcrma	scaling	pmonly	liwong	custom 3	expresso	AFFY
gcrma	scaling	pmonly	farms	custom 4	exp.farms	FARMS
gcrma	invariantset	pmonly	tukey.biweight	custom 5	expresso	AFFY
gcrma	invariantset	pmonly	medianpolish	custom 6	expresso	AFFY
gcrma	invariantset	pmonly	liwong	custom 7	expresso	AFFY
gcrma	invariantset	pmonly	farms	custom 8	exp.farms	FARMS
gcrma	quantile	pmonly	tukey.biweight	custom 9	threestep	AFFYPLM
gcrma	quantile	pmonly	median.polish	GCRMA	threestep	AFFYPLM
gcrma	quantiles	pmonly	liwong	Custom 10	expresso	AFFY
gcrma	quantiles	pmonly	farms	Custom 11	exp.farms	FARMS
rma	scaling	pmonly	tukey.biweight	Custom 12	threestep	AFFYPLM
rma	scaling	pmonly	median.polish	Custom 13	threestep	AFFYPLM
rma	scaling	pmonly	liwong	Custom 14	expresso	AFFY
rma	scaling	pmonly	farms	Custom 15	exp.farms	FARMS
rma	invariantset	pmonly	tukey.biweight	Custom 16	expresso	AFFY
rma	invariantset	pmonly	medianpolish	Custom 17	expresso	AFFY
rma	invariantset	pmonly	liwong	Custom 18	expresso	AFFY
rma	invariantset	pmonly	farms	Custom 19	exp.farms	FARMS

rma	quantile	pmonly	tukey.biweight	Custom 20	threestep	AFFYPLM
rma	quantiles	pmonly	medianpolish	RMA	rma	AFFY
rma	quantiles	pmonly	liwong	Custom 21	expresso	AFFY
rma	quantiles	pmonly	farms	Custom 22	exp.farms	FARMS
mas	scaling	pmonly	tukey.biweight	Custom 23	threestep	AFFYPLM
mas	scaling	pmonly	median.polish	Custom 24	threestep	AFFYPLM
mas	scaling	pmonly	liwong	Custom 25	expresso	AFFY
mas	scaling	pmonly	farms	Custom 26	exp.farms	FARMS
mas	scaling	mas	tukey.biweight	MAS5	mas5	AFFY
mas	scaling	mas	medianpolish	Custom 27	expresso	AFFY
mas	scaling	mas	liwong	Custom 28	expresso	AFFY
mas	scaling	mas	farms	Custom 29	exp.farms	FARMS
mas	invariantset	pmonly	tukey.biweight	Custom 30	expresso	AFFY
mas	invariantset	pmonly	medianpolish	Custom 31	expresso	AFFY
mas	invariantset	pmonly	liwong	Custom 32	expresso	AFFY
mas	invariantset	pmonly	farms	Custom 33	exp.farms	FARMS
mas	invariantset	mas	tukey.biweight	Custom 34	expresso	AFFY
mas	invariantset	mas	medianpolish	Custom 35	expresso	AFFY
mas	invariantset	mas	liwong	Custom 36	expresso	AFFY
mas	invariantset	mas	farms	Custom 37	exp.farms	FARMS
mas	quantiles	pmonly	tukey.biweight	Custom 38	threestep	AFFYPLM
mas	quantile	pmonly	median.polish	CP	threestep	AFFYPLM
mas	quantiles	pmonly	liwong	Custom 39	expresso	AFFY
mas	quantiles	pmonly	farms	Custom 40	expresso	AFFY
mas	quantiles	mas	tukey.biweight	Custom 41	expresso	AFFY
mas	quantiles	mas	medianpolish	Custom 42	expresso	AFFY
mas	quantiles	mas	liwong	Custom 43	expresso	AFFY
mas	quantiles	mas	farms	Custom 44	exp.farms	FARMS
none	scaling	pmonly	tukey.biweight	Custom 45	threestep	AFFYPLM
none	scaling	pmonly	median.polish	Custom 46	threestep	AFFYPLM
none	scaling	pmonly	liwong	Custom 47	expresso	AFFY
none	scaling	pmonly	farms	Custom 48	exp.farms	FARMS
none	scaling	mas	tukey.biweight	Custom 49	expresso	AFFY
none	scaling	mas	medianpolish	Custom 50	expresso	AFFY
none	scaling	mas	liwong	Custom 51	expresso	AFFY

none	scaling	mas	farms	Custom 52	exp.farms	FARMS
none	invariantset	pmonly	tukey.biweight	Custom 53	expresso	AFFY
none	invariantset	pmonly	medianpolish	Custom 54	expresso	AFFY
none	invariantset	pmonly	liwong	MBEI (PMonly model)	expresso	AFFY
none	invariantset	pmonly	farms	Custom 55	exp.farms	FARMS
none	invariantset	mas	tukey.biweight	Custom 56	expresso	AFFY
none	invariantset	mas	medianpolish	Custom 57	expresso	AFFY
none	invariantset	mas	liwong	Custom 58	expresso	AFFY
none	invariantset	mas	farms	Custom 59	exp.farms	FARMS
none	quantile	pmonly	tukey.biweight	Custom 60	threestep	AFFYPLM
none	quantile	pmonly	median.polish	Custom 61	threestep	AFFYPLM
none	quantiles	pmonly	liwong	Custom 62	expresso	AFFY
none	quantiles	pmonly	farms	Custom 63	q.farms	FARMS
none	quantiles	mas	tukey.biweight	Custom 64	expresso	AFFY
none	quantiles	mas	medianpolish	Custom 65	expresso	AFFY
none	quantiles	mas	liwong	Custom 66	expresso	AFFY
none	quantiles	mas	farms	Custom 67	exp.farms	FARMS

AFFY package: [Irizarry, 2011a],[Gautier et al., 2004]
affyPLM package: [Bolstad, 2011a]
vsn package: [Huber et al., 2002], [Huber, 2011]
farms package: [Hochreiter et al., 2006], [Hochreiter et al., 2011]
plier package: [Miller, 2011]
puma package: [Pearson et al., 2009],[Pearson et al., 2011]

Bibliography

Affymetrix (1999). *Affymetrix Microarray Suite Users Guide, v.4.0*. Affymetrix, Santa Clara, CA.

Affymetrix (2001). *Affymetrix Microarray Suite Users Guide, v.5.0*. Affymetrix, Santa Clara, CA.

Agresti, A. (2006). *Building and Applying Logistic Regression Models*, pages 137–172. John Wiley & Sons, Inc.

Akaike, H. (1973). *Information theory and an extension of the maximum likelihood principle*, volume 1, pages 267–281. Akademiai Kiado.

Alpaydin, E. (1994). Gal: Networks that grow when they learn and shrink when they forget. *International Journal of Pattern Recognition and Artificial Intelligence*, pages 391–414.

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403 – 410.

Alvord, W., Roayaei, J., Quiñones, O., and Schneider, K. (2007). A microarray analysis for differential gene expression in the soybean genome using bioconductor and r. *Briefings in Bioinformatics*, 8(6):415–431.

Aran, O., Yildiz, O. T., and Alpaydin, E. (2009). An incremental framework based on cross-validation for estimating the architecture of a multilayer perceptron. *IJPRAI*, 23(2):159–190.

Armstrong, S., Staunton, J., Silverman, L., Pieters, R., den Boer, M., Minden, M., Sallan, S., Lander, E., Golub, T., and Korsmeyer, S. (2002). MLL translocations

specify a distinct gene expression profile that distinguishes a unique leukemia. *Nat Genet*, 30:41–47.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., and Sherlock, G. (2000). Gene ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–29. PMID: 10802651 PMCID: 3037419.

Balaguer, E., Palomares, A., Soria-Olivas, E., and Martín-Guerrero, J. D. (2008). Predicting service request in support centers based on nonlinear dynamics, arma modeling and neural networks. *Expert Syst. Appl.*, 34(1):665–672.

Bengtsson, H., Simpson, K., Bullard, J., and Hansen, K. (2011). aroma.affymetrix: a generic framework in r for analyzing small to very large affymetrix data sets in bounded memory, tech report 745, department of statistics, university of california, berkeley. http://www.aroma-project.org/.

Benoudjit, N. and Verleysen, M. (2003). On the kernel widths in radial-basis function networks. *Neural Process. Lett.*, 18:139–154.

Bolstad, B. (2004). *Low level analysis of high-density oligonucleotide array data: background, normalization and summarization*. PhD thesis, University of California, Berkeley.

Bolstad, B. (2011a). affyplm: methods for fitting probe-level models, bioconductor package, v.1.28.5. http://bioconductor.org/packages/2.8/bioc/html/affyPLM.html.

Bolstad, B. (2011b). Rmaexpress software, a standalone gui program to compute gene expression summary values for affymetrix genechips. http://rmaexpress.bmbolstad.com/.

Bolstad, B., Irizarry, R., Astrand, M., and Speed, T. (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193.

Bowers, P., Pellegrini, M., Thompson, M., Fierro, J., Yeates, T., and Eisenberg, D. (2004). Prolinks: a database of protein functional linkages derived from coevolution. *Genome Biology*, 5(5):R35.

Box, G. E. P., Hunter, W. G., Hunter, J. S., and Hunter, W. G. (1978). *Statistics for Experimenters: An Introduction to Design, Data Analysis, and Model Building*. John Wiley & Sons.

Boyacioglu, M. and Avci, D. (2010). An adaptive network-based fuzzy inference system (anfis) for the prediction of stock market return: The case of the istanbul stock exchange. *Expert Systems with Applications*, 37(12):7908–7912.

Bradford, J. R., Needham, C. J., Tedder, P., Care, M. A., Bulpitt, A. J., and Westhead, D. R. (2010). Go-at: in silico prediction of gene function in arabidopsis thaliana by combining heterogeneous data. *The Plant Journal*, 61(4):713–721.

Breiman, L. (2001). Random forests. *Machine Learning*, 45:5–32. 10.1023/A:1010933404324.

Caba, O., Alvarez, P., Prados, J., Ortiz, R., Melguizo, C., Rodriguez-Serrano, F., Delgado, J., Rojas, I., Florido, J. P., Prieto, A., and Aranega, A. (2010). Global gene expression profiling of peripheral blood pancreas adenocarcinoma patiens to determine potential biomarkers to treatment response. http://www.medmol.es/publications/documents/7/.

Chen, Y., Yang, B., Dong, J., and Abraham, A. (2005). Time-series forecasting using flexible neural tree model. *Inf. Sci.*, 174:219–235.

Chen, Z., Li, J., and Wei, L. (2007a). A multiple kernel support vector machine scheme for feature selection and rule extraction from gene expression data of cancer tissue. *Artif. Intell. Med.*, 41:161–175.

Chen, Z., McGee, M., Liu, Q., and Scheuermann, R. (2007b). A distribution free summarization method for Affymetrix GeneChip arrays. *Bioinformatics*, 23(3):321–327.

Cherkassky, V., Gehring, D., and Mulier, F. (1996). Comparison of adaptive methods for function estimation from samples. *Neural Networks, IEEE Transactions on*, 7(4):969 –984.

Cherkassky, V. and Ma, Y. (2004). Practical selection of svm parameters and noise estimation for svm regression. *Neural Netw.*, 17:113–126.

Cherry, J. M., Ball, C., Weng, S., Juvik, G., Schmidt, R., Adler, C., Dunn, B., Dwight, S., Riles, L., Mortimer, R. K., and Botstein, D. (1997). Genetic and physical maps of Saccharomyces cerevisiae. *Nature*, 387(6632 Suppl):67–73.

Choe, S., Boutros, M., Michelson, A., Church, G., and Halfon, M. (2005). Preferred analysis methods for Affymetrix GeneChips revealed by a wholly defined control dataset. *Genome Biology*, 6(2):R16+.

Chu, F. and Wang, L. (2005). Applications of support vector machines to cancer classification with microarray data. *International Journal of Neural Systems,*, 15(6):475–484.

Constantinopoulos, C. and Likas, A. (2006). An incremental training method for the probabilistic rbf network. *IEEE Transactions on Neural Networks*, 17(4):966–974.

Coombes, K. (2010). Classdiscovery package of the object-oriented microarray and proteomic analysis (oompa). http://bioinformatics.mdanderson.org/Software/OOMPA/.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Mach. Learn.*, 20:273–297.

Costello, J., Dalkilic, M., Beason, S., Gehlhausen, J., Patwardhan, R., Middha, S., Eads, B., and Andrews, J. (2009). Gene networks in drosophila melanogaster: integrating experimental data to predict gene function. *Genome Biology*, 10(9):R97.

Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 233–240, New York, NY, USA. ACM.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2000). A Fast Elitist Multi-Objective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197.

Diamantidis, N. A., Karlis, D., and Giakoumakis, E. A. (2000). Unsupervised stratification of cross-validation for accuracy estimation. *Artif. Intell.*, 116:1–16.

Dreyfus, G. (2005). *Neural networks - methodology and applications*. Springer.

Du, H. and Zhang, N. (2008). Time series prediction using evolving radial basis function networks with new encoding scheme. *Neurocomput.*, 71:1388–1400.

Duda, R. O., Hart, P. E., and Stork, D. G. (2001). *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2 edition.

Dupret, G. and Koda, M. (2001). Bootstrap re-sampling for unbalanced data in supervised learning. *European Journal of Operational Research*, 134(1):141–156.

Dyrskjot, L., Thykjaer, T., Kruhoffer, M., Jensen, J., Marcussen, N., Wolf, H., and Orntoft, T. (2003). Identifying distinct classes of bladder carcinoma using microarrays. *Nat Genet*, 33(1):90–96.

Efron, B. and Tibshirani, R. (1997). Improvements on Cross-Validation: The .632+ Bootstrap Method. *Journal of the American Statistical Association*, 92(438):548–560.

Eiben, A. E. and Smith, J. E. (2008). *Introduction to Evolutionary Computing (Natural Computing Series)*. Springer.

Elefsinioti, A., Ackermann, M., and Beyer, A. (2009). Accounting for redundancy when integrating gene interaction databases. *PLoS ONE*, 4(10):e7492.

Espejo, P. G., Ventura, S., and Herrera, F. (2010). A Survey on the Application of Genetic Programming to Classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(2):121–144.

Etchells, T. A. and Lisboa, P. J. G. (2006). Orthogonal search-based rule extraction (OSRE) for trained neural networks: a practical and efficient approach. *Neural Networks, IEEE Transactions on*, 17(2):374–384.

Fahlman, S. E. (1988). Faster-learning variations on Back-propagation: An empirical study. In *Proceedings of the 1988 Connectionist Summer School*, pages 1–19, San Mateo, CA. Morgan Kaufmann.

Florido, J. P., Claros, M., Pomares, H., and Rojas, I. (2010a). Ranking affymetrix microarray-based pre-processing methods. In *10th Spanish Symposium on Bioinformatics, Málaga,Spain*, pages 226–229.

Florido, J. P., Pomares, H., Herrera, L., Rojas, I., and Urquiza, J. (2009a). Addressing rbfnns for time series prediction: inputs and network model selection. In *III Simposio de Inteligencia Computacional, SICO'10, Congreso Espanol de Informatica, CEDI 2010*, pages 229–236.

Florido, J. P., Pomares, H., Herrera, L., Rojas, I., and Urquiza, J. (2009b). Time series prediction using mutual information and rbfnns. In *I International Workshop on Mining and Non-Conventional Data (MINCODA'09), 13rd Conference of the Spanish Association for Artificial Intelligence (CAEPIA'09)*, pages 25–32.

Florido, J. P., Pomares, H., and Rojas, I. (2011a). Generating balanced learning and test sets for function approximation problems. *International Journal of Neural Systems*, 21(3):247–263.

Florido, J. P., Pomares, H., Rojas, I., Calvo, J. C., Urquiza, J. M., and Claros, M. G. (2009c). On selecting the best pre-processing method for affymetrix genechips. In *Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part I: Bio-Inspired Systems: Computational and Ambient Intelligence, IWANN'09*, volume 5517 of *Lecture Notes in Computer Science*, pages 845–852, Berlin, Heidelberg. Springer-Verlag.

Florido, J. P., Pomares, H., Rojas, I., Lopez-Gordo, M., and Urquiza, J. (2011b). A deterministic model selection scheme for incremental rbfnn construction in time series forecasting. *Neural Computing and Applications*, In Press, Corrected Proof.

Florido, J. P., Pomares, H., Rojas, I., Urquiza, J., Herrera, L., and Claros, M. (2010b). Effect of pre-processing methods on microarray-based svm classifiers in affymetrix genechips. In *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pages 1 –6.

Friedman, M. (1937). The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *Journal of the American Statistical Association*, 32(200):675–701.

Gacto, M., Alcala, R., and Herrera, F. (2008). Adaptation and application of multi-objective evolutionary algorithms for rule reduction and parameter tuning of fuzzy rule-based systems. *Soft Comput.*, 13:419–436.

García, S., Molina, D., Lozano, M., and Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: acase study ontheCEC'2005 Special Session onReal Parameter Optimization. *Journal of Heuristics*.

Gautier, L., Cope, L., Bolstad, B., and Irizarry, R. (2004). affy—analysis of affymetrix genechip data at the probe level. *Bioinformatics*, 20:307–315.

Gentleman, R., Carey, V., Bates, D., Bolstad, B., Dettling, M., Dudoit, S., Ellis, B., Gautier, L., Ge, Y., Gentry, J., Hornik, K., Hothorn, T., Huber, W., Iacus, S., Irizarry, R., Leisch, F., Li, C., Maechler, M., Rossini, A., Sawitzki, G., Smith, C., Smyth, G., Tierney, L., Yang, J., and Zhang, J. (2004). Bioconductor: open software development for computational biology and bioinformatics. *Genome Biology*, 5(10):R80+.

Gentleman, R., Carey, V., Huber, W., and Hahne, F. (2011). genefilter: methods for filtering genes from microarray experiments, bioconductor package, v.1.34.0. http://www.bioconductor.org/packages/2.8/bioc/html/genefilter.html.

Gentleman, R., Carey, V., Huber, W., Irizarry, R., and Dudoit, S. (2005). *Bioinformatics and Computational Biology Solutions Using R and Bioconductor (Statistics for Biology and Health)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Ghaffari, A., Mehrabian, A. R., and Mohammad-Zaheri, M. (2007). Identification and control of power plant de-superheater using soft computing techniques. *Eng. Appl. Artif. Intell.*, 20:273–287.

Gomm, J. and Yu, D. (2000). Selecting radial basis function network centers with recursive orthogonal least squares training. *Neural Networks, IEEE Transactions on*, 11(2):306 –314.

Gonzalez, J., Rojas, H., Ortega, J., and Prieto, A. (2002). A new clustering technique for function approximation. *IEEE Trans Neural Netw*, 13(1):132–42.

Gonzalez, J., Rojas, I., Ortega, J., Pomares, H., Fernandez, F., and Diaz, A. (2003). Multiobjective evolutionary optimization of the size, shape, and position parameters of radial basis function networks for function approximation. *Neural Networks, IEEE Transactions on*, 14(6):1478 – 1495.

Guillén, A., Gonzlez, J., Rojas, I., Pomares, H., Herrera, L. J., Valenzuela, O., and Prieto, A. (2007). Using fuzzy logic to improve a clustering technique for function approximation. *Neurocomputing*, 70:2853–2860.

Guillén, A., Pomares, H., Rojas, I., Gonzlez, J., Herrera, L. J., Rojas, F., and Valenzuela, O. (2008). Studying possibility in a clustering algorithm for rbfnn design for function approximation. *Neural Computing and Applications*, pages 75–89.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Mach. Learn.*, 46:389–422.

Guzzi, P. and Cannataro, M. (2010). mu-CS: an extension of the TM4 platform to manage Affymetrix binary data. *BMC bioinformatics*, 11(1):315+.

Hackl, H., Stocker, G., Charoentong, P., Mlecnik, B., Bindea, G., Galon, J., and Trajanoski, Z. (2010). Information technology solutions for integration of biomolecular and clinical data in the identification of new cancer biomarkers and targets for therapy. *Pharmacology and Therapeutics*, 128(3):488 – 498.

Hamid, J. S., Pingzhao, H., Roslin, N. M., Ling, V., Greenwood, C. M. T., and Beyene, J. (2009). Data integration in genetics and genomics: Methods and challenges. *Human Genomics and Proteomics*, 2009(869093):1–13.

Hamosh, A., Scott, A., Amberger, J., Bocchini, C., and McKusick, V. (2005). Online mendelian inheritance in man (omim), a knowledgebase of human genes and genetic disorders. *Nucleic Acids Research*, 33(suppl 1):D514–D517.

Harr, B. and Schlötterer, C. (2006). Comparison of algorithms for the analysis of Affymetrix microarray data as evaluated by co-expression of genes in known operons. *Nucleic Acids Research*, 34(2):e8.

Hartwell, L. H., Hopfield, J. J., Leibler, S., and Murray, A. W. (1999). From molecular to modular cell biology. *Nature*, 402(6761):47–52.

Haykin, S. (1998). *Neural Networks, A comprehensive Foundation*. Prentice Hall, Inc., New Yersey, USA.

Hènon, M. (1976). A two-dimensional mapping with a strange attractor. *Comm. in Mathematical Physics*, 50:69–77.

Herrera, L., Pomares, H., Rojas, I., Verleysen, M., and Guilén, A. (2006). Effective Input Variable Selection for Function Approximation. In *Proceedings of the International Conference on Artificial Neural Networks*, pages 41–50.

Hochreiter, S., Clevert, D., and Obermayer, K. (2006). A new summarization method for affymetrix probe level data. *Bioinformatics*, 22:943–949.

Hochreiter, S., Clevert, D., and Obermayer, K. (2011). Farms package, v.1.4.1. http://www.bioinf.jku.at/software/farms/farms.html.

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6:65–70.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366.

Huang, G., Saratch, P., and Sundararajan, N. (2005). A generalized growing and pruning rbf (ggap-rbf) neural network for function approximation. *IEEE Transactions on Neural Networks*, 16:57–67.

Huber, W. (2011). Variance stabilization and calibration for microarray data, bioconductor package, v.3.20.0. http://www.bioconductor.org/packages/2.8/bioc/html/vsn.html.

Huber, W., von Heydebreck, A., Saltmann, H., Poustka, A., and Vingron, M. (2002). Variance stabilization applied to microarray data calibration and to the quantification of differential expression. *Bioinformatics*, 18(suppl 1):S96–S104.

Hwang, D., Rust, A. G., Ramsey, S., Smith, J. J., Leslie, D. M., Weston, A. D., de Atauri, P., Aitchison, J. D., Hood, L., Siegel, A. F., and Bolouri, H. (2005). A

data integration methodology for systems biology. *Proceedings of the National Academy of Sciences of the United States of America*, 102(48):17296–17301.

Iman, R. and Davenport, J. (1980). Approximations of the critical region of the Friedman statistic. *Communications in Statistics*, 9(6):571–595.

Irizarry, R. (2011a). Affy: Methods for affymetrix oligonucleotide arrays, bioconductor package, v.1.30. http://www.bioconductor.org/packages/2.8/bioc/html/affy.html.

Irizarry, R. (2011b). Ampaffyexample package. example of amplified data, bioconductor package, v.1.2.5. http://www.bioconductor.org/packages/2.8/data/experiment/html/AmpAffyExample.html.

Irizarry, R. A., Hobbs, B., Collin, F.and Beazer-Barclay, Y., Antonellis, K., Scherf, U., and Speed, T. (2003). Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics (Oxford, England)*, 4(2):249–264.

Jacobs, R. A. (1987). Increased rates of convergence through learning rate adaptation. Technical report, University of Massachusetts, Amherst, MA, USA.

Janga, S., Daz-Meja, J. J., and Moreno-Hagelsieb, G. (2011). Network-based function prediction and interactomics: The case for metabolic enzymes. *Metabolic Engineering*, 13(1):1 – 10.

Jansen, R. and Gerstein, M. (2004). Analyzing protein function on a genomic scale: the importance of gold-standard positives and negatives for network prediction. *Current Opinion in Microbiology*, 7(5):535 – 545.

Jayawardena, A., Xu, P., Tsang, F., and Li, W. (2006). Determining the structure of a radial basis function network for prediction of nonlinear hydrological time series. *HYDROLOGICAL SCIENCES JOURNAL-JOURNAL DES SCIENCES HYDROLOGIQUES*, 51(1):21–44.

Kaminski, W. and Strumillo, P. (1997). Kernel orthonormalization in radial basis function neural networks. *Neural Networks, IEEE Transactions on*, 8(5):1177 –1183.

Karayiannis, N. and Mi, G. (1997). Growing radial basis neural networks: Merging supervised and unsupervised learning with network growth techniques. *IEEE Transactions on Neural Networks*, 8:1492–1506.

Kauffmann, A., Gentleman, R., and Huber, W. (2009). arrayQualityMetrics–a bioconductor package for quality assessment of microarray data. *Bioinformatics (Oxford, England)*, 25(3):415–416.

Kauffmann, A. and Huber, W. (2011). arrayqualitymetrics: Array quality metrics on microarray data sets, bioconductor package, v.3.8.0. http://www.bioconductor.org/packages/2.8/bioc/html/arrayQualityMetrics.html.

KEGG (2010). Kyoto encyclopedia of genes and genomes. http://www.genome.jp/kegg/download/.

Knudsen, S. (2006). *Cancer diagnostics with DNA Microarrays*. John Wiley & Sons, Inc., New York, NY, USA.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proc. of the 14th Int. Joint Conference on Artificial Intelligence (Montreal, Quebec, Canada, 1995)*, pages 1137–1143. Morgan Kaufmann.

Korsan, R. (1993). Fractals and time series analysis. *Mathematica*, 3(1):39–47.

Koza, J. R. (1992). Genetic programming: on the programming of computers by means of natural selection. *Statistics and Computing*.

Kukar, M. and Kononenko, I. (1998). Cost-sensitive learning with neural networks. In *ECAI'98*, pages 445–449.

Kwok, T. and Yeung, D. (1997). Constructive algorithms for structure learning in feedforward neural networks for regression problems. *IEEE Transactions on Neural Networks*, 8:630–645.

Laiho, P., Kokko, A., Vanharanta, S., Salovaara, R., Sammalkorpi, H., Jarvinen, H., Mecklin, J. P., Karttunen, T. J., Tuppurainen, K., Davalos, V., Schwartz, S., Arango, D., Makinen, M. J., and Aaltonen, L. A. (2006). Serrated carcinomas

form a subclass of colorectal cancer with distinct molecular basis. *Oncogene*, 26(2):312–320.

Lanckriet, G., Deng, M., Cristianini, N., Jordan, M., and Noble, W. (2004). Kernel-based data fusion and its application to protein function prediction in yeast. In *Biocomputing 2004, Proceedings of the Pacific Symposium, Hawaii, USA*, pages 300–311. World Scientific.

Lee, I., Date, S. V., Adai, A. T., and Marcotte, E. M. (2004). A probabilistic functional network of yeast genes. *Science*, 306(5701):1555–1558.

Lee, I., Lehner, B., Vavouri, T., Shin, J., Fraser, A. G., and Marcotte, E. M. (2010). Predicting genetic modifier loci using functional gene networks. *Genome Research*, 20(8):1143–1153.

Lee, I., Li, Z., and Marcotte, E. M. (2007). An improved, bias-reduced probabilistic functional gene network of baker's yeast, ¡italic¿saccharomyces cerevisiae¡/italic¿. *PLoS ONE*, 2(10):e988.

Lees, J. G., Heriche, J. K., Morilla, I., Ranea, J. A., and Orengo, C. A. (2011). Systematic computational prediction of protein interaction networks. *Physical Biology*, 8(3):035008.

Lefebvre, G., Steele, R., and Vandal, A. C. (2010). A path sampling identity for computing the kullback-leibler and j divergences. *Comput. Stat. Data Anal.*, 54:1719–1731.

Lendasse, A., Wertz, V., and Verleysen, M. (2003). Model selection with cross-validations and bootstraps - application to time series prediction with rbfn models. In *Artificial Neural Networks and Neural Information Processing ICANN/ICONIP 2003*, pages 573–580. Springer-Verlag.

Leung, H., Dubash, N., and Xie, N. (2002). Detection of small objects in clutter using a ga-rbf neural network. *Aerospace and Electronic Systems, IEEE Transactions on*, 38(1):98 –118.

Li, C. and Wong, W. (2001). Model-based analysis of oligonucleotide arrays: model validation, design issues and standard error application. *Genome Biology*, 2(8):research0032.1–research0032.11.

Li, J., Li, X., Su, H., Chen, H., and Galbraith, D. W. (2006). A framework of integrating gene relations from heterogeneous data sources: an experiment on arabidopsis thaliana. *Bioinformatics*, 22(16):2037–2043.

Li, X., Cai, H., Xu, J., Ying, S., and Zhang, Y. (2010). A mouse protein interactome through combined literature mining with multiple sources of interaction evidence. *Amino Acids*, 38(4):1237–52.

Li, Y. and Patra, J. (2010a). Integration of multiple data sources to prioritize candidate genes using discounted rating system. *BMC Bioinformatics*, 11(Suppl 1):S20.

Li, Y. and Patra, J. C. (2010b). Genome-wide inferring genephenotype relationship by walking on the heterogeneous network. *Bioinformatics*, 26(9):1219–1224.

Lim, W. K., Wang, K., Lefebvre, C., and Califano, A. (2007). Comparative analysis of microarray normalization procedures: effects on reverse engineering gene networks. *Bioinformatics*, 23(13):i282–i288.

Lin, N., Wu, B., Jansen, R., Gerstein, M., and Zhao, H. (2004). Information assessment on predicting protein-protein interactions. *BMC Bioinformatics*, 5(1):154.

Linghu, B., Snitkin, E., Holloway, D., Gustafson, A., Xia, Y., and DeLisi, C. (2008). High-precision high-coverage functional inference from integrated data sources. *BMC Bioinformatics*, 9(1):119.

Linghu, B., Snitkin, E., Hu, Z., Xia, Y., and DeLisi, C. (2009). Genome-wide prioritization of disease genes and identification of disease-disease associations from an integrated human functional linkage network. *Genome Biology*, 10(9):R91.

Ljung, L. (1986). *System identification: theory for the user*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Lodish, H., Berk, A., Kaiser, C. A., Krieger, M., Scott, M. P., Bretscher, A., Ploegh, H., and Matsudaira, P. (2007). *Molecular Cell Biology (Lodish, Molecular Cell Biology)*. W. H. Freeman, 6th edition.

Lovasz, L. (1993). Random walks on graphs: A survey.

Lu, L. J., Xia, Y., Paccanaro, A., Yu, H., and Gerstein, M. (2005). Assessing the limits of genomic data integration for predicting protein networks. *Genome Research*, 15(7):945–953.

Lysenko, A., Defoin-Platel, M., Hassani-Pak, K., Taubert, J., Hodgman, C., Rawlings, C., and Saqi, M. (2011). Assessing the functional coherence of modules found in multiple-evidence networks from arabidopsis. *BMC Bioinformatics*, 12(1):203.

Mackey, M. C. and Glass, L. (1977). Oscillation and chaos in physiological control systems. *Science*, 197(4300):287–289.

MATLAB (2010a). Global optimization toolbox 3 user's guide.

MATLAB (2010b). Neural network toolbox 7 user's guide.

MATLAB (2010c). *version 7.11.0 (R2010b)*. The MathWorks Inc., Natick, Massachusetts.

May, R. M. (1976). Simple mathematical models with very complicated dynamics. *Nature*, 261(5560):459–467.

Mendel, J. M. (2001). *Uncertain rule-based fuzzy logic system: introduction and new directions*. Prentice–Hall PTR.

Mering, C. v., Huynen, M., Jaeggi, D., Schmidt, S., Bork, P., and Snel, B. (2003). String: a database of predicted functional associations between proteins. *Nucleic Acids Research*, 31(1):258–261.

Mewes, H. W., Frishman, D., Gldener, U., Mannhaupt, G., Mayer, K., Mokrejs, M., Morgenstern, B., Mnsterktter, M., Rudd, S., and Weil, B. (2002). Mips: a database for genomes and protein sequences. *Nucleic Acids Research*, 30(1):31–34.

Miller, C. (2011). Affymetrix plier package, bioconductor package, v.1.22.0. http://www.bioconductor.org/packages/2.8/bioc/html/plier.html.

Mojena, R. (1977). Hierarchical grouping methods and stopping rules: an evaluation. *The Computer Journal*, 20(4):359–363.

Moreno-Torres, J. G., Raeder, T., Alaiz-Rodrguez, R., Chawla, N. V., and Herrera, F. (2011). A unifying view on dataset shift in classification. *Pattern Recognition*, In Press, Corrected Proof:–.

Mostafavi, S. and Morris, Q. (2010). Fast integration of heterogeneous data sources for predicting gene function with limited annotation. *Bioinformatics*, 26(14):1759–1765.

Mulder, N. J., Apweiler, R., Attwood, T. K., Bairoch, A., Bateman, A., Binns, D., Bradley, P., Bork, P., Bucher, P., Cerutti, L., Copley, R., Courcelle, E., Das, U., Durbin, R., Fleischmann, W., Gough, J., Haft, D., Harte, N., Hulo, N., Kahn, D., Kanapin, A., Krestyaninova, M., Lonsdale, D., Lopez, R., Letunic, I., Madera, M., Maslen, J., McDowall, J., Mitchell, A., Nikolskaya, A. N., Orchard, S., Pagni, M., Ponting, C. P., Quevillon, E., Selengut, J., Sigrist, C. J. A., Silventoinen, V., Studholme, D. J., Vaughan, R., and Wu, C. H. (2005). Interpro, progress and status in 2005. *Nucleic Acids Research*, 33(suppl 1):D201–D205.

Myers, C., Barrett, D., Hibbs, M., Huttenhower, C., and Troyanskaya, O. (2006). Finding function: evaluation methods for functional genomic data. *BMC Genomics*, 7(1):187.

Nadeau, C. and Bengio, Y. (2003). Inference for the generalization error. *Mach. Learn.*, 52:239–281.

Narendra, K. S. and Parthasarathy, K. (1990). Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1(1):4–27.

Nutt, C. L., Mani, D. R., Betensky, R. A., Tamayo, P., Cairncross, J. G., Ladd, C., Pohl, U., Hartmann, C., and Mclaughlin, M. E. (2003). Gene expression-based classification of malignant gliomas correlates better with survival than histological classification. *Cancer Res.*, 63:1602–1607.

Ogata, H., Goto, S., Sato, K., Fujibuchi, W., Bono, H., and Kanehisa, M. (1999). Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 27(1):29–34.

Paetz, J. (2004). Reducing the number of neurons in radial basis function networks with dynamic decay adjustment. *Neurocomputing*, 62:79–91.

Park, J. and Sandberg, I. W. (1991). Universal approximation using radial-basis-function networks. *Neural Comput.*, 3:246–257.

Parker, D. (1987). Optimal algorithm for adaptive networks: second order back propagation, second order direct propagation and second order hebbian learning. In *Proc. of the IEEE International Conference on Neural Networks*, pages 593–600.

Patrinos, P., Alexandridis, A., Ninos, K., and Sarimveis, H. (2010). Variable selection in nonlinear modeling based on rbf networks and evolutionary computation. *Int. J. Neural Syst.*, 20(5):365–379.

Paul, S. and Kumar, S. (2002). Subsethood-product fuzzy neural inference system (supfunis). *IEEE Trans Neural Netw*, 13(3):578–99.

Pearson, R., Liu, X., Sanguinetti, G., Milo, M., Lawrence, N., and Rattray, M. (2009). puma: a bioconductor package for propagating uncertainty in microarray analysis. *BMC Bioinformatics*, 10(1):211.

Pearson, R., Liu, X., Sanguinetti, G., Milo, M., Lawrence, N., and Rattray, M. (2011). puma: propagating uncertainty in microarray analysis, bioconductor package, v.2.4.0. http://www.bioconductor.org/packages/2.8/bioc/html/puma.html.

Pelckmans, K., Suykens, J. A. K., Van Gestel, T., De Brabanter, J., Lukas, L., Hamers, B., De Moor, B., and Vandewalle, J. (2002). LS-SVMlab: a Matlab/C toolbox for least squares support vector machines. Technical Report 02-44, ESAT-SISTA, K.U. Leuven, Leuven, Belgium.

Pochet, N., De Smet, F., Suykens, J., and De Moor, B. (2004). Systematic benchmarking of microarray data classification: assessing the role of non-linearity and dimensionality reduction. *Bioinformatics*, 20:3185–3195.

Pomares, H. (2000). *New methodologies for fuzzy systems design*. PhD thesis, University of Granada.

Pomares, H., Ruiz, I. R., González, J., Damas, M., Pino, B., and Prieto, A. (2004). Online global learning in direct fuzzy controllers. *IEEE T. Fuzzy Systems*, 12(2):218–229.

Pomeroy, S., Tamayo, P., Gaasenbeek, M., Sturla, L., Angelo, M., McLaughlin, M. E., Kim, J., Goumnerova, L., Black, P., Lau, C., Allen, J., Zagzag, D., Olson, J., Curran, T., Wetmore, C., Biegel, J., Poggio, T., Mukherjee, S., Rifkin, R., Califano, A., Stolovitzky, G., Louis, D., Mesirov, J., Lander, E., and Golub, T. (2002). Prediction of central nervous system embryonal tumour outcome based on gene expression. *Nature*, 415(6870):436–442.

Pozhitkov, A., Tautz, D., and Noble, P. (2007). Oligonucleotide microarrays: widely appliedpoorly understood. *Briefings in Functional Genomics & Proteomics*, 6(2):141–148.

Pruitt, K. D., Tatusova, T., Klimke, W., and Maglott, D. R. (2009). Ncbi reference sequences: current status, policy and new initiatives. *Nucleic Acids Research*, 37(suppl 1):D32–D36.

Puscasu, G., Codres, B., Stancu, A., and Murariu, G. (2009). Nonlinear system identification based on internal recurrent neural networks. *Int. J. Neural Syst.*, 19(2):115–125.

Qi, M. and Zhang, G. (2001). An investigation of model selection criteria for neural network time series forecasting. *European Journal of Operational Research*, 132:666–680.

Qi, Y., Bar-Joseph, Z., and Klein-Seetharaman, J. (2006). Evaluation of different biological data and computational classification methods for use in protein interaction prediction. *Proteins: Structure, Function, and Bioinformatics*, 63(3):490–500.

Qiu, J. and Noble, W. S. (2008). Predicting co-complexed protein pairs from heterogeneous data. *PLoS Comput Biol*, 4(4):e1000054.

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Rainer, J., Sanchez-Cabo, F., Stocker, G., Sturn, A., and Trajanoski, Z. (2006). CARMAweb: comprehensive R- and bioconductor-based web service for microarray data analysis. *Nucleic acids research*, 34(Web Server issue).

Ravasz, E., Somera, A. L., Mongru, D. A., Oltvai, Z. N., and Barabsi, A.-L. (2002). Hierarchical organization of modularity in metabolic networks. *Science*, 297(5586):1551–1555.

Ray, S. S., Bandyopadhyay, S., and K.Pal, S. (2009). Combining multisource information through functional-annotation-based weighting: gene function prediction in yeast. *IEEE Transactions on biomedical engineering*, 56(2):229–236.

Re, M. and Valentini, G. (2010). Integration of heterogeneous data sources for gene function prediction using decision templates and ensembles of learning machines. *Neurocomputing*, 73(7-9):1533–1537.

RefSeq (2010). The reference sequence collection. ftp://ftp.ncbi.nih.gov/genomes/.

Rehrauer, H., Zoller, S., and Schlapbach, R. (2007). Magma: analysis of two-channel microarrays made easy. *Nucleic Acids Research*, 35(Web-Server-Issue):86–90.

Ressom, H., Varghese, R., Zhang, Z., Xuan, J., and Clarke, R. (2008). Classification algorithms for phenotype prediction in genomics and proteomics. *Frontiers in bioscience : a journal and virtual library*, 13:691–708.

Rhodes, D. R., Tomlins, S. A., Varambally, S., Mahavisno, V., Barrette, T., Kalyana-Sundaram, S., Ghosh, D., Pandey, A., and Chinnaiyan, A. M. (2005). Probabilistic model of the human protein-protein interaction network. *Nat Biotech*, 23(8):951–959.

Rissanen, J. (1978). Modeling By Shortest Data Description. *Automatica*, 14:465–471.

Ritchie, M., Silver, J., Oshlack, A., Holmes, M., Diyagama, D., Holloway, A., and Smyth, G. (2007). A comparison of background correction methods for two-colour microarrays. *Bioinformatics*, 23:2700–2707.

Rojas, I., Pomares, H., Gonzáles, J., Bernier, J. L., Ros, E., Pelayo, F. J., and Prieto, A. (2000). Analysis of the functional block involved in the design of radial basis function networks. *Neural Process. Lett.*, 12:1–17.

Ross, M. E., Zhou, X., Song, G., Shurtleff, S. A., Girtman, K., Williams, W. K., Liu, H.-C., Mahfouz, R., Raimondi, S. C., Lenny, N., Patel, A., and Downing, J. R. (2003). Classification of pediatric acute lymphoblastic leukemia by gene expression profiling. *Blood*, 102(8):2951–2959.

Ruepp, A., Zollner, A., Maier, D., Albermann, K., Hani, J., Mokrejs, M., Tetko, I., Gldener, U., Mannhaupt, G., Mnsterktter, M., and Mewes, H. W. (2004). The funcat, a functional annotation scheme for systematic classification of proteins from whole genomes. *Nucleic Acids Research*, 32(18):5539–5545.

Schadt, E., Li, C., Ellis, B., and Wong, W. (2001). Feature extraction and normalization algorithms for high-density oligonucleotide gene expression array data. *JCell Bioachm Suppl.*, 37:120–125.

Schwarz, G. (1978). Estimating the Dimension of a Model. *The Annals of Statistics*, 6(2):461–464.

Segal, M. R. (2004). Machine learning benchmarks and random forest regression.

SGD (2010). Saccharomyces genome database. http://downloads.yeastgenome.org/.

Shiblee, M., Kalra, P. K., and Chandra, B. (2009). Time Series Prediction with Multilayer Perceptron (MLP): A New Generalized Error Based Approach. In Koppen, M and Kasbov, N and Coghill, G, editor, *ADVANCES IN NEURO-INFORMATION PROCESSING, PT II*, volume 5507 of *Lecture Notes in Computer Science*, pages 37–44. SPRINGER-VERLAG BERLIN.

Shipp, M., Ross, K., Tamayo, P., Weng, A., Kutok, J., Aguiar, R., Gaasenbeek, M., Angelo, M., Reich, M., Pinkus, G., Ray, T., Koval, M., Last, K., Norton, A., Lister, T. A., Mesirov, J., Neuberg, D., Lander, E., Aster, J., and Golub, T. (2002). Diffuse large b-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning. *Nat Med*, 8(1):68–74.

Singh, D., Febbo, P., Ross, K., Jackson, D., Manola, J., Ladd, C., Tamayo, P., Renshaw, A., D'Amico, A., and Richie, J. (2002). Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, 1(2):203–209.

Slawski, M., Daumer, M., and Boulesteix, A. L. (2008). CMA - a comprehensive Bioconductor package for supervised classification with high dimensional data. *BMC Bioinformatics*, 9(1):439+.

Smith, M. (1993). *Neural Networks for Statistical Modeling*. John Wiley & Sons, Inc., New York, NY, USA.

Smyth, G. (2004). Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Statistical applications in genetics and molecular biology*, 3(1).

Smyth, G. (2011). Limma: linear models for microarray data, bioconductor package, v.3.8.3. http://www.bioconductor.org/packages/2.8/bioc/html/limma.html.

Sorjamaa, A., Hao, J., and Lendasse, A. (2005). Mutual information and k-nearest neighbors approximator for time series prediction. In *Proceedings of the 15th international conference on Artificial neural networks: formal models and their applications - Volume Part II*, ICANN'05, pages 553–558, Berlin, Heidelberg. Springer-Verlag.

Sorjamaa, A., Hao, J., Reyhani, N., Ji, Y., and Lendasse, A. (2007). Methodology for long-term prediction of time series. *Neurocomput.*, 70:2861–2869.

Spears, W. M., Jong, K. A. D., Bck, T., Fogel, D. B., and De Garis, H. (1993). An overview of evolutionary computation. *Lecture Notes in Computer Science*, 667(1):442–459.

Statnikov, A., Aliferis, C., Tsamardinos, I., Hardin, D., and Levy, S. (2005). A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics (Oxford, England)*, 21(5):631–643.

Statnikov, A., Wang, L., and Aliferis, C. (2008). A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification. *BMC Bioinformatics*, 9(1):319+.

STRING (2010). String: Known and predicted protein-protein interactions. http://string-db.org/.

Strong, M., Mallick, P., Pellegrini, M., Thompson, M., and Eisenberg, D. (2003). Inference of protein function and protein linkages in mycobacterium tuberculosis based on prokaryotic genome organization: a combined computational approach. *Genome Biology*, 4(9):R59.

Subramanian, A., Kuehn, H., Gould, J., Tamayo, P., and Mesirov, J. (2007). Gsea-p: a desktop application for gene set enrichment analysis. *Bioinformatics*, 23(23):3251–3253.

Suykens, J., Van Gestel, T., De Brabanter, J., De Moor, B., and Vandewalle, J. (2002). *Least Squares Support Vector Machines*. World Scientific., Singapore.

Szklarczyk, D., Franceschini, A., Kuhn, M., Simonovic, M., Roth, A., Minguez, P., Doerks, T., Stark, M., Muller, J., Bork, P., Jensen, L. J., and Mering, C. v. (2011). The string database in 2011: functional interaction networks of proteins, globally integrated and scored. *Nucleic Acids Research*, 39(suppl 1):D561–D568.

Tatusov, R. L., Koonin, E. V., and Lipman, D. J. (1997). A genomic perspective on protein families. *Science*, 278(5338):631–637.

Theodoridis, D. C., Boutalis, Y. S., and Christodoulou, M. A. (2010). Indirect adaptive control of unknown multi variable nonlinear systems with parametric and dynamic uncertainties using a new neuro-fuzzy system description. *Int. J. Neural Syst.*, 20(2):129–148.

Tusher, V. G., Tibshirani, R., and Chu, G. (2001). Significance analysis of microarrays applied to the ionizing radiation response. *Proceedings of the National Academy of Sciences*, 98(9):5116–5121.

Vaquerizas, J. M., Conde, L., Yankilevich, P., Cabezón, A., Minguez, P., Díaz-Uriarte, R., Al-Shahrour, F., Herrero, J., and Dopazo, J. (2005). GEPAS, an

experiment-oriented pipeline for the analysis of microarray gene expression data. *Nucleic Acids Res*, 33(Web Server issue).

Vasquez, M. and Janaqi, S. (2001). A tabu algorithm for homogeneous partition of samples. In *Proc. of the 4th Metaheuristics Int. Conference (Porto, Portugal,2001)*, pages 155–158. .

Wang, J., Wen, S., Symmans, W., Pusztai, L., and Coombes, K. (2009a). The bimodality index: a criterion for discovering and ranking bimodal signatures from cancer gene expression profiling data. *Cancer Informatics*, 7:199–216. PMID: 19718451.

Wang, Y., Zhang, X.-S., and Xia, Y. (2009b). Predicting eukaryotic transcriptional cooperativity by bayesian network integration of genome-wide data. *Nucleic Acids Research*, 37(18):5943–5958.

Weigend, A. S. and Gershenfeld, N. A., editors (1994). *Time series prediction: Forecasting the future and understanding the past*.

Wen, C. and Ma, X. (2008). A max-piecewise-linear neural network for function approximation. *Neurocomputing*, 71(4-6):843–852.

Whalen, E. and Gentleman, R. (2011). Chronic lymphocytic leukemia (cll) gene expression data, bioconductor package, v.1.2.8. http://bioconductor.org/packages/2.8/data/experiment/html/CLL.html.

Wilson, C. and Miller, C. (2011). simpleaffy: a bioconductor package for affymetrix quality contro and data analysis, bioconductor package, v.2.28.0. http://www.bioconductor.org/packages/2.8/bioc/html/simpleaffy.html.

Wilson, C. L. and Miller, C. (2005). Simpleaffy: a BioConductor package for Affymetrix Quality Control and data analysis. *Bioinformatics (Oxford, England)*, 21(18):3683–3685.

Witten, I. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann.

Wong, W. (2011). dchip software. https://sites.google.com/site/dchipsoft/.

Wu, C.-C., Asgharzadeh, S., Triche, T. J., and D'Argenio, D. Z. (2010). Prediction of human functional genetic networks from heterogeneous data using rvm-based ensemble learning. *Bioinformatics*, 26(6):807–813.

Wu, W., Xing, E., Myers, C., Mian, I., and Bissell, M. (2005). Evaluation of normalization methods for cdna microarray data by -nn classification. *BMC Bioinformatics*, 6:–1–1.

Wu, Z., Irizarry, R., Gentleman, R.and Martinez-Murillo, F., and Spencer, F. (2004). A Model-Based Background Adjustment for Oligonucleotide Expression Arrays. *Journal of the American Statistical Association*, 99(468):909–917.

Xia, K., Dong, D., and Han, J. (2006). IntNetDB v1.0: an integrated protein-protein interaction network database generated by a probabilistic model. *BMC Bioinformatics*, 7(1):508.

Xiong, H., Zhang, D., Martyniuk, C., Trudeau, V., and Xia, X. (2008). Using generalized procrustes analysis (gpa) for normalization of cdna microarray data. *BMC Bioinformatics*, 9.

Xiong, H., Zhang, Y., and Chen, X. (2007). Data-dependent kernel machines for microarray data classification. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 4(4):583–595.

Xiong, J. (2006). *Essential Bioinformatics*. Cambridge University Press, 1 edition.

Xiong, J., Rayner, S., Luo, K., Li, Y., and Chen, S. (2006). Genome wide prediction of protein function via a generic knowledge discovery approach based on evidence integration. *BMC Bioinformatics*, 7(1):268.

Xu, F., Li, G., Zhao, C., Li, Y., Li, P., Cui, J., Deng, Y., and Shi, T. (2010). Global protein interactome exploration through mining genome-scale data in arabidopsis thaliana. *BMC Genomics*, 11(Suppl 2):S2.

Xu, T., Gu, J., Zhou, Y., and Du, L. (2009). Improving detection of differentially expressed gene sets by applying cluster enrichment analysis to Gene Ontology. *BMC bioinformatics*, 10(1):240+.

Yao, X. and Liu, Y. (1997). Epnet for chaotic time-series prediction. In *Selected papers from the First Asia-Pacific Conference on Simulated Evolution and Learning*, SEAL'96, pages 146–156, London, UK. Springer-Verlag.

Yao, Z. and Ruzzo, W. (2006). A regression-based k nearest neighbor algorithm for gene function prediction from heterogeneous data. *BMC Bioinformatics*, 7(Suppl 1):S11.

Ye, J., Li, T., Xiong, T., and Janardan, R. (2004). Using uncorrelated discriminant analysis for tissue classification with gene expression data. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 1:181–190.

You, Z.-H., Yin, Z., Han, K., Huang, D.-S., and Zhou, X. (2010). A semi-supervised learning approach to predict synthetic genetic interactions by combining functional and topological properties of functional gene network. *BMC Bioinformatics*, 11(1):343.

Yu, W. and Li, X. (2004). Fuzzy identification using fuzzy neural networks with stable learning algorithms. *IEEE T. Fuzzy Systems*, 12(3):411–420.

Zeng, X. and Martinez, T. R. (2000). Distribution-balanced stratified cross-validation for accuracy estimation. *JETAI*, 12(1):1–12.

Zhang, L., Wong, S., King, O., and Roth, F. (2004). Predicting co-complexed protein pairs using genomic and proteomic data integration. *BMC Bioinformatics*, 5(1):38.

Zhao, H. (2007). A multi-objective genetic programming approach to developing pareto optimal decision trees. *Decision Support Systems*, 43(3):809 – 826. Integrated Decision Support.

Zhou, Z.-H. and Liu, X.-Y. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *Knowledge and Data Engineering, IEEE Transactions on*, 18(1):63 – 77.

Zhu, Y., Zhu, Y., and Xu, W. (2008). EzArray: a web-based highly automated Affymetrix expression array data management and analysis system. *BMC bioinformatics*, 9(1).

Zitzler, E., Laumanns, M., and Thiele, L. (2002). Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. *Evolutionary Methods for Design Optimisation and Control with Application to Industrial Problems EUROGEN 2001*, pages 95–100.

Zou, H. F., Xia, G. P., Yang, F. T., and Wang, H. Y. (2007). An investigation and comparison of artificial neural network and time series models for chinese food grain price forecasting. *Neurocomput.*, 70:2913–2923.