UNIVERSIDAD DE GRANADA

E.T.S. DE INGENIERÍAS INFORMÁTICA Y DE TELECOMUNICACIÓN

Programa de Doctorado en Tecnologías de la Información y la Comuncación

Uso de Técnicas de Minería de Datos para el Control de Autoridades Automatizado

Tesis Doctoral

Irene Díaz Valenzuela

Directores: María José Martín-Bautista y María Amparo Vila Miranda

Editorial: Universidad de Granada. Tesis Doctorales
Autora: Irene Díaz Valenzuela
ISBN: 978-84-9125-208-5
URI: http://hdl.handle.net/10481/40660

La memoria titulada "Uso de Técnicas de Minería de Datos para el Control de Autoridades Automatizado", que presenta Dña. Irene Díaz Valenzuela, para optar al grado de Doctor, ha sido realizada en el Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada bajo la dirección de las Doctoras María José Martín Bautista y María Amparo Vila Miranda.

La doctoranda Dña. Irene Díaz Valenzuela y las directoras de la tesis Doctoras María José Martín Bautista y María Amparo Vila Miranda. Garantizamos, al firmar esta tesis doctoral, que el trabajo ha sido realizado por la doctoranda bajo la dirección de las directoras de la tesis y hasta donde nuestro conocimiento alcanza, en la realización del trabajo, se han respetado los derechos de otros autores a ser citados, cuando se han utilizado sus resultados o publicaciones.

Granada, Junio de 2015

La doctoranda	Las directoras	
[illegible]	[illegible]	[illegible]
Irene Díaz Valenzuela	María José Martín Bautista	M. Amparo Vila Miranda

Agradecimientos

It could be worse. It could be raining.

Gene Wilder

Quiero comenzar esta tesis agradeciendo a toda la gente que me ha acompañado durante estos cuatro años. Habéis sido muchos y sin vosotros, esto no habría sido posible.

A mis directoras, las Doctoras María Amparo Vila y María José Martín Bautista, porque gracias a vosotras esta tesis es una realidad.

Al Profesor Vincezo Loia y su equipo. Gracias por el buen trato recibido durante mi estancia en Salerno. Especialmente, me gustaría agradecer a la Profesora Sabrina Senatore por todo el tiempo que estuvimos trabajando "mano a mano", por todas las ideas y por todos los buenos ratos que pasamos juntas.

A todos los compañeros que me han acompañado en las distintas etapas del desarrollo de esta tesis. A la gente que compartimos despacho en Orquídea: *Dani*, *Juanan*, *María*, *Michella*, *Pablo*, *Rafa*, *Sara*, *Sergio* y *Victoria*. No quiero dejar pasar esta oportunidad de agradecer a *Manu García* todo lo que ha hecho por mi durante este tiempo, incluyendo el día que pensó que era una buena idea que en mi mesa llovieran polvorones.

A mis compañeros de despacho en el CITIC: *Niceto*, *Paco*, *Richard*, *Jesús* y *Pablo*. Han sido muchas horas, muchas risas y muchos cafés juntos. Habéis conseguido que se diluya la frontera entre compañeros de trabajo y amigos. Al personal de servicios del CITIC, principalmente a *Encarni*, porque sin ti el cactus no habría sobrevivido y yo hace tiempo que habría sido sepultada por papeles.

A *Mariló*, *María*, *Migue*, *Juan*, *Gonzalo*, *Rita* y tantos otros becarios del departamento. Muchas gracias por enseñarme como funciona esto, por compartir tantas comidas y tantos buenos ratos. A *Jesús Campaña*, porque la lista de cosas que has hecho por mi es muy larga y no cabe en este documento.

A la gente de CORISA: *Antonio*, *Carmine*, *Pasquale*, *Paola*, *Pietro* y *Susanna*. Gracias por hacerme sentir como una más en el laboratorio, por enseñarme italiano, por las pizzas, por el árbol de Navidad y por todos los buenos ratos que hemos pasado juntos.

Al *Milk Club*: *Alex*, *Juanga*, *Matteo*, *Pablo*, *Rafa*, *Rob* y *Roberto*. Sin vosotros esta última etapa no habría sido tan divertida.

A *Yahira*, *Julia* y *Sandra* por ayudarme a liberarme del stress y darme la dosis de buen rollo diaria para seguir adelante.

Y por supuesto a mi familia. A mis padres que siempre me han apoyado y han creído en mi. A *Marina*, mi hermanita y la persona más increíble del mundo. Finalmente, *last but not least*, a *Juan Ramón*, que es quién siempre me ayuda a ponerlo todo en perspectiva. Gracias de corazón por haber estado a mi lado.

Lo dicho, ¡GRACIAS A TODOS!

Contents

Resumen Extendido

Esta tesis aborda el problema del "Uso de técnicas de Minería de datos para el Control de Autoridades automatizado". Este problema puede definirse como el proceso de *reconocer diferentes representaciones del mismo concepto dentro de los registros de una biblioteca*[86].

El control de autoridades puede aplicarse a distintos tipos de datos asociados a un registro, como los lugares de publicación o las series. Para esta tesis, nos vamos a centrar únicamente en el *Control de Autoridades de Nombres.* Este caso concreto puede definirse como la detección de las distintas representaciones para el nombre de un autor.

Motivación: El Control de Autoridades

Tradicionalmente, el control de autoridades se ha venido realizando por los bibliotecarios, dado que son los encargados de las bibliotecas y conocen los registros que allí se encuentran. Sin embargo, conforme aumenta el número de registros almacenados en una biblioteca, este proceso puede volverse muy complejo y tedioso.

En los últimos tiempos, como complemento a las bibliotecas tradicionales, se ha popularizado el uso de Bibliotecas Digitales. Este tipo de servicios pueden definirse como sitios web que contienen registros similares a los de una biblioteca, tales como libros, audio, vídeos, publicaciones científicas, etc. En esta tesis abordaremos la Automatización del Control de Autoridades en Bibliotecas Digitales, centrándonos en aquellas dedicadas a publicaciones científicas, tales como CiteSeerX, DBLP, o INSPEC. Comúnmente, estas bibliotecas proporcionan además de las publicaciones, su información relacionada y servicios de búsqueda que permiten la consulta de los registros contenidos en ellas.

Realizar un buen proceso de control de autoridades no es una tarea sencilla, especialmente en bibliotecas digitales donde no suele haber un bibliotecario encargado. Por ese motivo, el uso de sus servicios de búsqueda puede resultar muy complejo, especialmente si se trata de encontrar todas las publicaciones asociadas a un autor concreto. Dado que no existe uniformidad en los nombres, realizar una búsqueda usando este campo sólo devolverá aquellos resultados que encajen con el término buscado. Por ejemplo, si se desean encontrar las publicaciones de un investigador llamado *José García*, buscando esa cadena, aquellas publicaciones que aparezcan bajo *J. García*, o cualquier otra representación de su nombre, no serán devueltas por la búsqueda. Además, si hay más de un autor llamado así, los registros aparecerán mezclados, siendo muy difícil distinguir entre ellos.

Para paliar este tipo de problemas, proponemos automatizar el proceso de Control de Autoridades usando técnicas de Minería de Datos y que éste pueda ser aplicado a bibliotecas digitales de publicaciones científicas mediante un sistema automático capaz de agrupar las distintas publicaciones correspondientes a un mismo autor. Para ello, nos hemos basado en técnicas de agrupamiento (clustering) enfocando el problema como un proceso de "resolución de entidades". Este problema puede verse como una generalización del control de autoridades y definirse como tal la identificación de las distintas representaciones que una entidad pueda presentar en un conjunto de textos. Si se considera que un autor es una entidad, la identificación de sus posibles firmas puede verse como un caso concreto de resolución de entidades.

Como posible solución a este problema, proponemos un modelo teórico genérico de *resolución de entidades*, descrito en la Sección 2.2.1 el cual, aplicado a nuestro problema, nos proporciona un marco de representación que nos permite expresarlo desde una perspectiva de *Agrupamiento*. Debido a que cuándo se realiza una búsqueda por autor en una biblioteca digital se desconoce el número de autores al que corresponden, proponemos el uso de técnicas de agrupamiento jerárquico.

Para comprobar la calidad de las autoridades obtenidas se requiere de conocimiento experto. Sin embargo, la información proporcionada por este experto parecía demasiado útil como para ser utilizada únicamente en la validación. Esto nos llevó a la idea del uso del *Agrupamiento Semi-supervisado*.

Un nuevo enfoque de Agrupamiento Semisupervisado Difuso

El *Agrupamiento Semi-supervisado* es un tipo de técnica de Minería de Datos que se encuentra a medio camino entre las técnicas de agrupamiento supervisado y no supervisado. Este tipo de técnicas están basadas en la inclusión de una pequeña porción de información externa en el proceso de agrupamiento, la cual siempre será menor que la requerida para el agrupamiento supervisado.

Dentro de los distintos tipos de técnicas de agrupamiento semi-supervisado, son de especial interés las llamadas *restricciones a nivel de instancia* propuestas por Wagstaff y Cardie [108]. Estas técnicas introducen información externa en el agrupamiento indicando si algunos pares de instancias están o no en el mismo grupo. Dentro de estas restricciones se distinguen dos tipos: *must-link* (puede agruparse) y *cannot-link* (no puede agruparse) que indican si dos instancias dentro del proceso de agrupamiento deben o no aparecer en el mismo grupo.

Por tanto, proponemos el uso de restricciones a nivel de instancia dentro de un proceso de agrupamiento jerárquico. Para ello presentamos el algoritmo Crisp Hierarchical Semi-Supervised, Crisp HSS, el cual aplica este tipo de restricciones de forma distinta a como se venía utilizando tradicionalmente. Crisp HSS parte de un método de agrupamiento jerárquico y utiliza las restricciones para encontrar el número de grupos óptimo que puede encontrarse en la salida de dicho método. Este algoritmo se encuentra explicado con detalle en la Sección 3.2.

Sin embargo, para determinados problemas, puede resultar difícil encontrar un experto que disponga del conocimiento suficiente para proporcionar restricciones a nivel de instancia nítidas, las cuales indican de forma cierta si dos elementos están o no en el mismo grupo. Es por esto que proponemos relajar esta restricción mediante el uso de *restricciones a nivel de instancia difusas*. A diferencia de las restricciones nítidas, estas restricciones difusas le dan al experto la flexibilidad de indicar un grado de creencia en la información que está proporcionando. Dentro de este tipo de restricciones proponemos dos tipos: *fuzzy must-link* (puede agruparse difuso) y *fuzzy cannot-link* (no puede agruparse difuso). Usando este nuevo tipo de restricciones, proponemos el algoritmo Fuzzy Hierarchical Semi-supervised, *Fuzzy HSS*, el cual encuentra el número de grupos óptimo dentro de un proceso de agrupamiento jerárquico usando restricciones difusas. Este algoritmo se encuentra explicado con detalle en la Sección 3.3.

Observando las propiedades de la restricciones a nivel de instancia difusas, descubrimos que es posible determinar la cantidad de información adicional que será necesaria en el proceso de agrupamiento mediante el estudio de su entropía difusa. Por lo tanto, en la Sección 3.4 describimos un método para determinar esta cantidad.

Los algoritmos *Crisp HSS* y *Fuzzy HSS* han sido probados experimentalmente tanto en problemas de propósito general como dentro del control de autoridades.

Generación automática de restricciones a nivel de instancia

Como extensión a los algoritmos de agrupamiento semi-supervisado propuestos en esta tesis proponemos una metodología de generación de restricciones a nivel de instancia.

Para ello, estudiamos la naturaleza de los datos usando un proceso de agrupamiento basado en las k-medias, con un número de grupos k mayor del esperado. Esto nos permite encontrar relaciones entre los datos que nos generan estas restricciones. Esta metodología, descrita con detalle en el Capítulo 4 ha sido probada con éxito en problemas de agrupamiento de documentos.

Conclusiones

Durante el desarrollo de esta tesis hemos llegado a las siguientes conclusiones:

- Las técnicas de Minería de Datos, específicamente, las técnicas de agrupamiento son una buena herramienta para abordar el problema del Control de Autoridades Automatizado. En el capítulo 2 hemos propuesto un sistema para automatizar este problema. Para ello ha sido necesario:

  - Proporcionar una *representación intermedia* considerando el problema como un caso específico de *Resolución de Entidades*, definiendo este otro problema como la búsqueda de los distintos nombres de una entidad en un texto. Dentro de este modelo general, una *autoridad* puede ser considerada como una entidad con diferentes nombres.
  - Proponer una medida de distancia específica para comparar los distintos elementos del problema. Esta medida es capaz de comparar los distintos elementos que aparecen en este proceso, proporcionándonos una medida adecuada para el proceso de agrupamiento. Durante el estudio de los diferentes elementos que deben ser comparados, hemos descubierto que las medidas de distancia tradicionales no se ajustan correctamente a la comparación de nombres propios. Por eso motivo proponemos *pn-measure*, una nueva medida de distancia ad-hoc específica para comparar nombres propios.

  - Un método de validación para evaluar la calidad del proceso de control de autoridades. Este método requiere de un experto que proporcione una base la cual permita comparar las autoridades y calcular una serie de medidas que nos permitan evaluar los resultados obtenidos.

- El método de validación requiere la intervención de un experto que indica en cada momento cuál es la solución correcta. Sin embargo, al considerar la naturaleza de dicha información, se descubrió que también podía ser incorporada en el proceso de agrupamiento, lo que ayuda a la mejora de los resultados obtenidos. Basándonos en esa idea, surgió el uso de agrupamiento semi-supervisado. En el capítulo 3 se han estudiado estas técnicas llegando a las siguientes conclusiones:

– Entre las posibilidades que ofrece el agrupamiento semi-supervisado, la mejor opción son las *restricciones a nivel de instancia.* Dicha metodología necesita que un experto proporcione información acerca de los pares de instancias que deben aparecer (o no) en el mismo grupo.

– Los enfoques semi-supervisados clásicos están basados en la idea de modificar la medida de distancia usada para comparar los elementos. Sin embargo, debido a la adecuación de la medida ad-hoc diseñada para este problema no resultaba deseable su modificación. Además, dado que en el Control de Autoridades Automático no se dispone del número de grupos en los que deben agruparse los datos, dato que es necesario en los algoritmos clásicos, se requería el uso de agrupamiento jerárquico.

– Basándonos en las ideas anteriores, se han propuesto dos nuevos algoritmos de agrupamiento jerárquico semi-supervisado:

  ∗ Crisp Hierarchical Semi-Supervised, *Crisp HSS* es un nuevo algoritmo de agrupamiento semi-supervisado centrado en el uso de información externa para encontrar la partición óptima en procesos de agrupamiento jerárquico.

  ∗ Considerando la naturaleza difusa inherente en el agrupamiento jerárquico y las características del algoritmo *Crisp HSS*, resultaba razonable su extensión a una versión difusa. Fuzzy Hierarchical Semi-Supervised, *Fuzzy HSS*, es un nuevo algoritmo de agrupamiento semi-supervisado que utiliza restricciones a nivel de instancia difusas. Estas restricciones proporcionan un grado de creencia acerca de que dos elementos se encuentran (o no) en el mismo grupo, lo que ha proporcionado flexibilidad al trabajo del experto.

  * Ambos algoritmos han sido probados experimentalmente mostrando un rendimiento similar. Por tanto, hemos descubierto que es posible usar la versión difusa en aquellos problemas en los que no es posible proporcionar información cierta.

  * Debido a las características observadas en la información difusa, hemos proporcionado además un mecanismo para determinar cuánta información externa debe ser proporcionada por el experto.

- Estos nuevos algoritmos, *Crisp HSS* y *Fuzzy HSS* han podido ser aplicados con éxito al Control de Autoridades Automatizado.

- Como una extensión del algoritmo *Crisp HSS*, se ha estudiado la generación automática de restricciones a nivel de instancia. Aplicando repetidamente un proceso de agrupamiento particional basado en K-medias y usando inicializaciones aleatorias, ha sido posible encontrar relaciones entre elementos que determinan restricciones a nivel de instancia.

- Esta extensión ha sido probada con éxito en problemas genéricos de agrupamiento de documentos, demostrando que las restricciones generadas automáticamente pueden ser utilizadas en aquellos problemas dónde el conocimiento experto no esté disponible.

Chapter 1

Introduction

This thesis approaches the "Use of Data Mining techniques for the Automatic Authority Control". During this Introduction, the Automatic Authority Control Problem will be described from a general point of view on Section 1.1. The motivation and objectives of this thesis are covered in Section 1.2. In Section 1.3 it is described how the Authority Control problem can be seen as a clustering problem. Section 1.4 describes some background and previous works and finally, in Section 1.5 the outline of this thesis is described.

1.1 The authority control problem

Authority Control is the process of recognizing different representations of the same concept in the collection of a library [86]. Library of Congress defines Authority Records as a tool used by librarians to establish forms of names (for persons, places, meetings, and organizations), titles, and subjects used on bibliographic records. Authority records enable librarians to provide uniform access to materials in library catalogs and a clear identification of authors and subject headings. For example, works about "movies", "motion pictures", "cinema", and "films" should all appear under the established subject heading "Motion pictures."

Authority control could be applied to several entities, being the names, venues and series the most common ones. However, on this thesis, the problem that will be approached is the *name authority control*. This modality focus on personal names and their different representations.

Traditionally, the authority control process was performed by librarians, as they usually are people in charge of a library. They normally are perfectly aware of the records kept in their libraries, so they could take the role of human experts. Considering that the records differ among libraries, the authority control process should be performed in a different way for each one and it could not be shared or reused. This creates a locality problem that has been tried to address by some initiatives who had been tried to internationalize and automate it, as the one held by LITA (Library & Information Technology Association) since 1979. [44]

The first steps towards automatic authority control were oriented to the standardization of the records, with the development of the MARC (Machine Readable Catalog) System [42]. This initiative was the first one providing an uniform way to store bibliographic records using a common language for libraries worldwide. That system included a tool for representing authority control results by the use of access points suggesting alternative names for specific authors. Although it represented a first step in that direction, MARC System does not provide automation, remaining authority control as a manual process performed by librarians.

Nowadays, libraries have expanded with the development of Digital Libraries. They are websites containing the same kind of records that could be found on a physical one, like books, audios, scientific publications, etc. On this thesis we will focus on the scientific publications area. A Digital Library of scientific publications is a website containing bibliographic information about publications. Some examples of these Digital Libraries are DBLP, CiteSeerX, or INSPEC, among others. They normally provide their users with the publication itself and some extra information about them like references or related publications. A reference or citation is a set of bibliographic fields that keeps information about the document constituting its bibliographic entity.

Performing a proper authority control process could be a tough task for every library, but it becomes specially harder when we are talking about a digital library on the web. There are several causes for that problem, as the demise of the figure of the librarian, in some cases, or the representations problems caused by noise, on those cases where their records had been automatically retrieved.

A good example of what we are talking about could be found on the search results of the surname Delgado over CiteSeerX. Between the results it is possible to find the publication Agricultural Growth Linkages in Sub-Saharan Africa1. The author field of this publication reads "by Peter Hazell, Anna A. Mckenna, Peter Gruhn, Baba Di Oum, Sene Gal, Wenche Barth Eide, Nor Way, Geoff Miller, Aus Tra Lia, Benno Ndulu, Tan Za Nia, Christopher L. Delgado, Christopher L. Delgado, Jane Hopkins, Jane Hopkins, Valerie A. Kelly, Valerie A. Kelly, Behjat Hojjati, Behjat Hojjati, Jayashree Sil, Jayashree Sil, Claude Courbois, Claude Courbois". Giving a close eye to this list, it is possible to see that it has a high number of duplicates. Furthermore, some entries do not seem to correspond to personal names, like Sene Gal, Nor Way, Aus Tra Lia, and Tan Za Nia which look like they are country names. Checking the original publication from which this information has been extracted, it is possible to see that its real authors are Christopher L. Delgado, Jane Hopkins, Valerie A. Kelly with Peter Hazell, Anna A. McKenna, Peter Gruhn, Behjat Hojjati, Jayashree Sil, and Claude Courbois. Comparing this list with the one from Cite-

SeerX, it is possible to find some names which do not correspond with any author on the list. A further check on the document shows us that these names are the Board of Trustees, and Senegal, Norway, Australia, and Tanzania are their countries of origin.

Mostly all digital libraries provide some sort of search facilities to ease the searching and finding process. Normally, that is the most common way to look for some information on them as they provide the specific publications that match a query made by the user. Commonly, digital libraries allow different kinds of searches, like searching by title or name. For our purpose, we will focus on the search by name, that returns publications written by an author specified by the user in the search query. However, obtaining all publications by a specific author could be a difficult task due to the lack of uniformity in the names, due to the lack of an authority control process. Some of these Digital Libraries are making big efforts to offer partial name disambiguation, but there is still a big amount of data where that information is not available or it is not correct. This problem becomes harder when users try to find information about a specific author. Automatic authority control could be very helpful in this problem for both the Digital Libraries and users. For the Digital Libraries, an automatic authority control process could help, or even avoid, the manual identification of the authorship in publications. For their users, it could provide more accurate searches by name.

Searching over an author name in a Web Digital Library could not be an easy task due to the so called Identity Uncertainty problem (the impossibility of distinguish between two authors sharing the same name), as well as the multiple representations available for it, as the use of abbreviations or not, the existence of a middle name or a second surname, language-specific characters, etc. This problem may be more severe if we use sources that contain noise, inconsistencies, or character-set related problems side-effects of the use of automatic gathering data systems. All those issues could result in mistakes and noise that could make the search by author process at some Digital Libraries very difficult.

Let us illustrate with an example the kind of problems that the users may deal with when searching for an author name. Suppose that we are interested in publications written by an author called James Smith, so we search in a Digital Library using the query *James Smith*. As a result, we get three entries: "Data Mining for educational purposes", "How to teach Data Mining", and "Electric Calibration of a PBC". If we know that *James Smith* works in Data Mining, it is likely that we realize that "Electric Calibration of a PBC" is not one of his works, but to detect that, we need some further knowledge that is not always available. Furthermore, we are interested in a paper called "Data Mining in practice" that we are sure that has been written by James Smith, but it is not shown in the previous results. This is because it has been published under J. Smith, and so, it has not been shown in the previous search as it is not under the same name.

The previously stated example shows the two biggest issues that make difficult to find all publications by an author in most Digital Libraries. The first one is the so called *Identity Uncertainty* problem. It can be defined as the impossibility of distinguishing the authority of a document without using extra information if there are two or more authors sharing the same name. The second one is the *Multi-representation* problem, which arises when there are different representations for an author's name. Different publication venues demand different ways of writing a person's name, so when results have been automatically extracted from those sources, the same author would be represented under different names. Besides these problems, it is also common to find ambiguous entries due to errors like misspellings, noise, character-set related problems, etc.

Finally, it should be noted that Authority Control can be seen as a specific case of *Entity Resolution*. Entity Resolution has been defined as the problem of identifying and joining duplicates on a database with different representations for the same entity. That is, to find those entries on a database (or databases) referring to the same thing. If we consider a Digital Library as a database of publications, Authority Control could be a specific case of Entity Resolution where authors with different representations of their names are identified. Under that idea we are going to approach the Automatic Name Authority Control problem from an Entity Resolution perspective.

1.2 Motivation

As we stated the Automatic Name Authority Control is a process that applied to a Digital Library could ease the "name search" process. Nowadays there are initiatives like ORCID [1] helping to identify the authorship in research publications. They propose the use of a unique researcher ID linking unambiguously each publication with its author. That process requires for all authors to manually mark their publications on every source where they are available. However, as this is a recent initiative, there is a big number of publications that have not been marked with their respective researcher IDs yet. The Automatic Authority Control could be of much help to this kind of initiatives, as it avoids the need to manually mark the publications.

It is the purpose of this thesis to use Data Mining to solve this problem. Among all the different types of techniques belonging to this topic, the one that seems most appropriate for it is *Clustering*. This technique, considered part of the unsupervised

[1] http://orcid.org/

learning methodologies, finds the groups underlying in a dataset, without the need of supervision provided by class labels. This techniques are very suitable for those problems where the number or the nature of the classes are not known, as with Authority Control. When performing Authority Control over search results, there is not a predefined number of Authorities. Even if some Digital Libraries, like DBLP, return their results grouped by author, that grouping usually has a lot of mixed or split authors, so that number could not be used. For this reason, *Clustering* is the Data Mining technique that could help us to automate this problem

.

During the process of approaching this problem from a clustering perspective, the validation problem arose. It meant that there was not an unsupervised way to find out if the results were or were not correct. For that reason, in a first approach, an expert has been included in the process who provided a baseline to define the quality of the clustering. Next, we considered that this information seemed too valuable to use it just for validation purposes, as it could be helpful in the grouping process itself. That lead to the idea of the use of *Semi-supervised clustering.*

Semi-supervised clustering are Data Mining techniques in the middle of supervised and unsupervised learning. They apply a small portion of external information to the clustering process with the intention of improving in some way its results. It is the intention of this thesis to study a new *Semi-supervised clustering* algorithm that could be used in the automatic Authority Control process, as well as other domains.

In short, this thesis has three basic objectives:

- The Automatic Authority Control from a clustering perspective.
- The design of a new Semi-supervised Clustering algorithm.

- The application of this new algorithm to the Automatic Authority Control problem.

How the first foundation was approached and lead to the second one is described on further detail in the next section.

1.3 Authority Control as a clustering process

Authority Control can basically be seen as the process of grouping records and labeling them with an authority mark. Focusing on the problem addressed in this thesis, *Name Authority Control* in digital libraries of scientific publications, it can be seen as the process of grouping scientific publications by author. It is our intention to address this problem from a searching perspective, i.e. with the intention of easing the search process and helping the user to find the desired information in a more clear way.

Talking about grouping is talk about clustering. According to [61], clustering can be defined as "a statistical classification technique for discovering whether the individuals of a population fall into different groups by making quantitative comparison of multiple characteristics". It means that clustering is the process of finding the underlying groups hidden in some data. In this specific case, it consists in finding the different publications of every author in a document corpora obtained from a digital library. From the definition, it is possible to deduce that the elements should be compared in some way, and these comparisons should be made considering multiple characteristics. Understanding properly these two concepts is the key for a good clustering process.

A quantitative *comparison* is, in this context, a measure. This measure provides a way to compare two instances (documents in our problem) assigning a value representing their level of similarity. Most clustering algorithms are based on these kind of measures and it is very important to choose the one that fits best the requirements of the problem. This lead us to the second element, the *characteristics*. These are elements inherent to the problems that represent the important aspects of the compared entities and allow to characterize them.

On chapter 2 we have addressed these issues by providing a formal model that allow an intermediate representation of the Authority Control problem. It is done by its definition as a specific case of entity resolution. Under this model two different proposals for *cases*, i.e. elements to include in the process have been made. Regarding distance measures, different proposals have been made as well as including an ad-hoc measure to compare personal names.

Using those definitions, a first proposal of a system for automatic authority control has been made based on unsupervised clustering. This system is able to take the output of a search over a Digital Library, and return those publications grouped by author name. As publications have several authors, they have been grouped according to the one that was queried in the search.

Clustering algorithms may be divided in two very big categories: *partitional* and *hierarchical*. Partitional algorithms divide the input data into a number of groups previously specified by the user. On the other hand, hierarchical clustering groups all instances into a tree structure where each branch represents a cluster. Under this model, there is no need to pre-specify the number of clusters that should be obtained, but it is possible to apply some technique to calculate it automatically. One thing about searches is that the user does not know how many documents will be returned as the result of its query, so there is not way to predict how many authors will be found. For that reason, the automatic authority control problem should be solved by means of hierarchical clustering.

To validate the output of that system, an expert was included in the process for gaving us a baseline of correct groups (authorities) that could be used for further comparisons with the output of the system. However, that information is very costly, as the expert should take some time studying the results as somehow provide a manual authority control.

Considering the characteristics of this expert information, it was possible to see that they established some relationships providing feedback about what elements should be placed in the same group, and what elements should not. Under those premises arose the idea of using that information as part of the clustering process to try to find an even better solution than the one from traditional clustering.

The process of introducing external information in the clustering process is called *Semi-supervised clustering*. Among all the semi-supervised clustering literature, there is a pretty interesting model called *Instance level constraints*[108]. They provide insight about pairs of instances in two ways: *must-link* and *cannot-link*. As can be deduced by the names, *must-link* constraints indicate pairs of instances that should be in the name cluster, whilst *cannot-link* indicate that those elements should not be together. The process of translating from the correct authorities to these constraints is trivial, as elements in the same cluster produce must-link constraints and elements in different clusters provide cannot-link constraints.

Traditionally, semi-supervised clustering used the external information to modify the clustering process in some way. However, experimental results have shown that our proposal of distance measure and cases provide a pretty good clustering tree, but the methodology to find its optimal number of clusters could be improved. Under this idea, we introduce a new semi-supervised clustering algorithm that introduces external information to find the optimal partition of the tree. Chapter 3 provides an extensive description of this algorithm.

On first instance, as the constraints defined from the expert information are crisp, the semi-supervised clustering algorithm made use of this kind information. However, due to the fuzzy nature of hierarchical clustering and the difficulties that the experts sometimes find to provide their information, a *fuzzy instance level constraints* model was proposed. It represents *must-link* and *cannot-link* constraints in a fuzzy way, allowing the expert to provide a degree of belief in the information that is providing without the need of being sure about it.

After the definition of these models, there is an extension that arises naturally from them. Even if the fuzzy information provides a pretty flexible model for the expert, it is still quite costly. For that reason, the automatic generation of instance level constraints have been studied and tested on Chapter 4.

1.4 Background

As stated on Section 1.2 This thesis has two main foundations: Automatic Authority Control and Semi-supervised Clustering. For this reason, the background of these topics is going to be studied separately. Section 1.4.1 contains a literature review of authority control and how this problem has been approached previously from a Data Mining perspective. On the other hand, Section 1.4.2 covers *Semi-supervised Clustering.*

1.4.1 Authority control

In 1979, LITA held the conference "Authority control: the key to tomorrow's catalog" from which two conclusions were drawn: the need of creating a global authority file shared by different institutions worldwide and the importance of an automatic system capable of keeping consistence between different records within a library [44].

Regarding automation, in 1987, Library of Congress took the first steps on it by editing the first edition of the specifications document for the MARC 21 system. MARC stands from Machine Readable Catalog and names a standard system to store bibliographic records using a common language for libraries worldwide. This system, in addition to unify the way in which the information is stored, first proposes a way to keep the authority control in the catalog using access points suggesting alternative names for an authority [42].

Authority files internationalization is still today a project under development, achieved only locally or regionally by institutions like Library of Congress at USA and Consortium of European Research Libraries (CERL) on Europe. Concerning the automation of the authority control process several approaches had been taken.

In 2000, Snyman et al [102] , made an interesting proposal of internationalization of the authors names by assigning them an unique identifier, working in the same way than ISSBN does for books. Actually, the ORCID initiative (Open Researcher and Contributor ID) is planning to develop an independent registry of unique identifiers for personal's names of individual researchers.

Trying to automate and improve the process of gathering bibliographic data, in 2005 the project MarcOnt, experimented with the use of ontologies to make them more complete and compatible with other web formats like BibTex or Dublin Core [68].

In the same year, Stefan Gradmann, from University of Hamburg, wrote about the possibility of using Semantic Web as a tool to model a bibliographic catalogue using web information. That work was motivated by the amount of poor structured information available at online catalogues [46].

A year before, in 2004, the first approaches to automatic authority control system were developed. One of the best examples of that was the OpenDBLP project [57]. On that work the author tried to solve problems raised from the lack of name and conference authority control. Their proposal is not fully automatic, but is the final user who, using a web interface, decides the changes that must be done.

Most of the automatic approaches to this problem that can be found in the literature suggest solutions based on Data Mining techniques. [55] propose a solution to this problem using two different supervised learning approaches: a Naïve Bayes classifier and a Support Vector Machine over a model built from three attributes: coauthor names, paper titles, and journal titles; as well as author names as labels for training the classifier. In [45], the authors also propose an unsupervised hierarchical learning approach, with the use of a K-way spectral clustering.

In [58], the authors propose an integrative framework for solving the name disambiguation problem in two steps. In the first step, a blocking method creates candidate classes of authors with similar names and in the second step, a clustering method, DBSCAN, groups papers by author. They use a distance metric between papers based on an online active selection support vector machine algorithm (LASVM).

A heuristic hierarchical model is proposed in [23], where the authors use the same three attributes as [55]: coauthor names, paper titles, and journal titles. Their method combines a distance function to compare similarity among the input records with heuristics that implies certain assumptions about citation records, like assuming that an author usually publishes with the same group of coauthors. Furthermore, they also apply the same model to build synthetic authorship records using a model based on probabilities [41].

The potential of exploiting relationships between entities has been considered in [12, 64], where the authors propose the use of a graph of relations between elements to share additional information that can help to identify the authorship of a publication. [104] propose an unified framework for name disambiguation in Digital Libraries using a probabilistic approach based on Hidden Markov Random Fields. For that, they build a graph based on relationships within publications based on parameters like venue, authorship, citations, etc.

The previously defined works approach the Authority Control problem for the whole library, although using a representative subset of the records to test the method. However, in [13], the authors propose to solve entity resolution from a query dataset using relational clustering techniques. Another approach from the Entity Resolution point of view can be found in [86], where the authors try to introduce some authority control using a manual preprocessing stage and developing a probabilistic model to decide whether two citations correspond to the same paper.

As can be seen in the previous proposals, the Authority control problem has been addressed from very different points of view. However, the solution offered has been typically restricted to the *Identity Uncertainty* problem. Among these solutions we can find a wide range of grouping methods, both supervised and unsupervised. Among the supervised proposals, some of them make use of Support Vector Machines (SVM) [91, 55], or SVM variations like LASVM [58]. Other proposed methods include: Naïve-Bayes [55], Random Forest [106], Latent Dirichlet Allocation [41] or

Markov Chains [104]. Unsupervised methods include WAD method [92] or Hierarchical Heuristic Based Method [23].

Besides Data Mining techniques, other methods propose the use of semantic information as [20], where the authors use a list of words related to the author and [70] that uses an Actor Ontology to create semantic lists of authorities.

Considering the literature to solve the Authority Control problem, there are still some unresolved problems. One the biggest lacks of current systems is that they cannot be applied for general purposes. It means that they need very specific attributes, like venue or citations, that are not available in many Digital Libraries or that are hard to obtain. Our proposal works with a very simple model that only uses attributes that can be found in any source: *title* and *authors*. Furthermore, additional non-standard attributes like *abstract* can be also included in the model, but they are not necessary for proper performance and can only be used when they are available. In addition to that, our method does not require any training or privileged information like the number of authors or publications.

We face the problem of automatic authority control using surname searches. It allows us to perform the Authority Control process on-demand. By using this approach, it is possible to obtain almost all publications for a specific author. Normally, the surname of an author does not change among publications. Some venues have specific rules about how to write first or middle names, but these do not apply to surnames. For that reason, searching for the surname will return almost all publications by that specific author, excluding those with typos. Using that way to obtain data, it is possible to find almost all the information needed to disambiguate authors without having to use the classic approaches of using the whole database. In that way, a controlled-size dataset is generated, avoiding the scalability problem that can arise with large datasets. Another advantage of combining our choice of attributes with this approach is that the need for building complex graphs of relationships among elements is avoided, allowing an easy and controlled way of comparing elements.

Our methodology to for automatic authority control in searching is based only on the information about publications that is available in most Digital Libraries. It solves at the same time both problems related with name ambiguity in searches: *Identity Uncertainty* and *Multi-representation.* To do that, we propose a methodology based on Data Mining techniques.

1.4.2 Semi-supervised clustering

Chapelle et al. [17] define semi-supervised clustering as: *"halfway between supervised and unsupervised learning. In addition to unlabeled data, the algorithm is provided with some supervision information –but not necessarily for all examples"*. This definition set us semi-supervised clustering as a technique that introduce external information in a clustering process. Its main difference with classification is the amount of external information required in the process, as generally, semi-supervised techniques only require supervision for a small amount of examples. There are some reviews about this specific topic like [5], [48] where the most important publications in the area are summarized.

There are two basic currents in semi-supervised clustering: using *partially labeled data* and using *instance level constraints.* The first approach is similar to supervised classification in the sense that both use some labeled examples to find a model that allows the proper classification of some data. However, semi-supervised methods need a considerably smaller amount of labeled examples than supervised ones. The first works found in this line are related with fuzzy clustering where some modifications to the FCM algorithm have been proposed [87]. After that first work, several improvements can be found also in the same line [11, 90, 71, 10, 16, 88, 89, 113].

In addition to this first fuzzy proposal and its variants, the use of partially labeled data has been extensively studied in the literature. One of the most popular proposals in that area is *Semi-supervised clustering by seeding* [6], where the authors propose the use of semi-supervision as seeds for the K-means algorithm. Some other relevant papers on this area include: [9, 53, 101, 103, 24].

The second approach, *Instance Level Constraints*, was first described by Wagstaff et al. [108]. This method introduces semi-supervision as some kind of constraint that indicates whether two pairs of instances should be placed on the same cluster. The first method developed following this model was a modification of the COBWEB algorithm that incorporated the constraints. The same authors also proposed a modification of the K-means algorithm [109] called COP-KMEANS, that basically modified the original algorithm by avoiding assigning points to clusters that violate any constraint. However, one of the potential drawbacks of that proposal is that it does not allow any constraint to be violated. Basu et al. on [7] proposed the PCKMeans method to solve that problem by allowing the user to specify a penalty for violating some of the constraints.

In addition to these first proposals, that could be considered the basis of the use of instance level constraints, it is possible to find several other approaches in the literature. Some traditional methods have their semi-supervised version, like Markov Random Fiels [66], Conditional Distributions [101], Spectral Clustering [110], Density Clustering [98], among others [69, 8]. Several studies have also been made regarding the feasibility of instance level constraints. Some of the most important ones could be found in [24, 107, 25].

Regarding similar domains of applications than the Automatic Authority Control problem, Semi-supervised clustering with instance level constraints have been applied to text clustering on several occasions: [84, 117, 59, 18, 116].

Most attention in semi-supervised clustering has been focused on partitional clustering. Some of the proposals on hierarchical clustering have been oriented to distance-learning [14]. Based on that idea, on [26], the authors make a complexity study and offer a proposal introducing a third kind of instance level constraints. Other proposals that introduce new kinds of constraints include [54]. All these works and some others like [81], [82], [3], [115] introduce semi-supervision in the process of creating the dendrogram. The approach proposed in this paper introduces semi-supervision differently, as we do not modify the dendrogram or the distance function. This paper starts with the premise that the dendrogram is already created and introduces semi-supervision in a further stage that is finding the optimal partition of the data when the number of groups is not known. As far as we know, this is the first proposal that uses semi-supervision in that way.

Even if most of the first approaches to semi-supervised clustering where related with fuzzy clustering, there are not so many examples of the use of fuzzy information in semi-supervision. Pedrycz [87] proposed to include some fuzzy-labeled patterns into fuzzy clustering procedure, idea that was also incorporated on some of its extensions [90, 15, 75, 114]. Other proposals based on *instance level constraints* define a function that establish the cost of violating the constraints for the fuzzy clustering algorithm [51, 49, 50, 43, 79, 74]. A different approach where the authors use the constraints to assign a initial value to seed the fuzzy clustering algorithm can be found in [112].

The fact that instance level constraints can have some level uncertainty is not an approach that has been considered often in the literature. For this reason we introduce the *fuzzy instance level constraints* on Chapter 3 as a model to give experts flexibility when providing their information.

1.5 Outline

This thesis is organized as follows:

- Chapter 2 is devoted to the Authority control problem from a Clustering perspective. This chapter describes a system to approach this problem automatically, as well as a formal model that provides a proper representation for automatic processing of the required data. An initial approach to the proposals in this chapter have been published in a JCR journal [38], as a work in the 10th International Conference on Intelligent Systems Design and Applications [31] and communicated in the XVI Congreso Español sobre Tecnologías y Lógica Fuzzy [32]. Additionally, part of the theoretical model described have been submitted to an international journal with impact factor [33].

- Chapter 3 described a semi-supervised clustering approach that could be applied to the Automatic Authority Control. This approach has two different parts: *Crisp HSS* which is a semi-supervised hierarchical clustering algorithm based on crisp information and its fuzzy counterpart *Fuzzy HSS*. Part of this chapter has been presented as a work in the IFSA World Congress and NAFIPS Annual Meeting, 2013 [34], the 15th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems [35], communicated in the XVII Congreso Español sobre Tecnologías y Lógica Fuzzy [36] and submitted to an international journal with impact factor [37].

- Chapter 4 proposes a methodology for the automatic generation of *instance level constraints* that could be used in semi-supervised clustering process. Parts of this chapter have been published in a international journa with impact factor [30].

Chapter 2

Authority Control from an unsupervised clustering perspective

In the previous chapter it was outlined how the Authority Control problem can be seen as a clustering problem. In this Chapter, all the elements that allow approaching the problem in such way will be described. In Section 2.1 there is a proposal of system architecture, that describes our proposal of getting from data in Digital Libraries in the Web to resolved *Authorities*. In addition to that, in Section 2.2 there is a full description of the theoretical model that provides a proper representation of the problem that can be used in a clustering process. Section 2.3 covers all the details of the clustering process, including the divergence measures, and a proposal for verifying the results. Finally, in Section 2.6 this approach is tested experimentally from the perspective of unsupervised clustering.

2.1 System Architecture

In this section, a system to help in the process of searching by name over a digital library is introduced. To start working with the system, the user must introduce an author name. This name must specifically be a surname corresponding with the

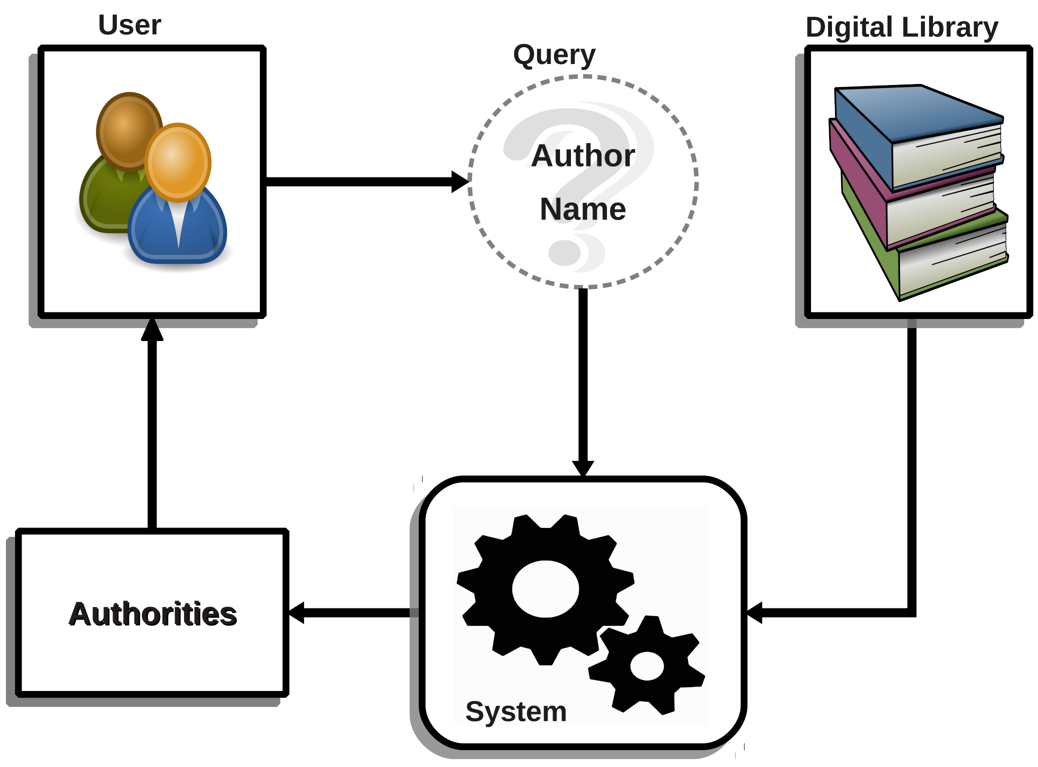


Figure 2.1: Authority Searching Process.

author that the user wants to check. It will be used to query the DL that will return a set of publications as search results. The system will group these results according to their authorship and they will be returned to the user in an easy way to read. This process has been summarized in Figure 2.1.

This system can be divided into two different modules:

- Data Preparing Module
- Data Mining Module

The *Data Preparing Module* is in charge of gathering the search results from the

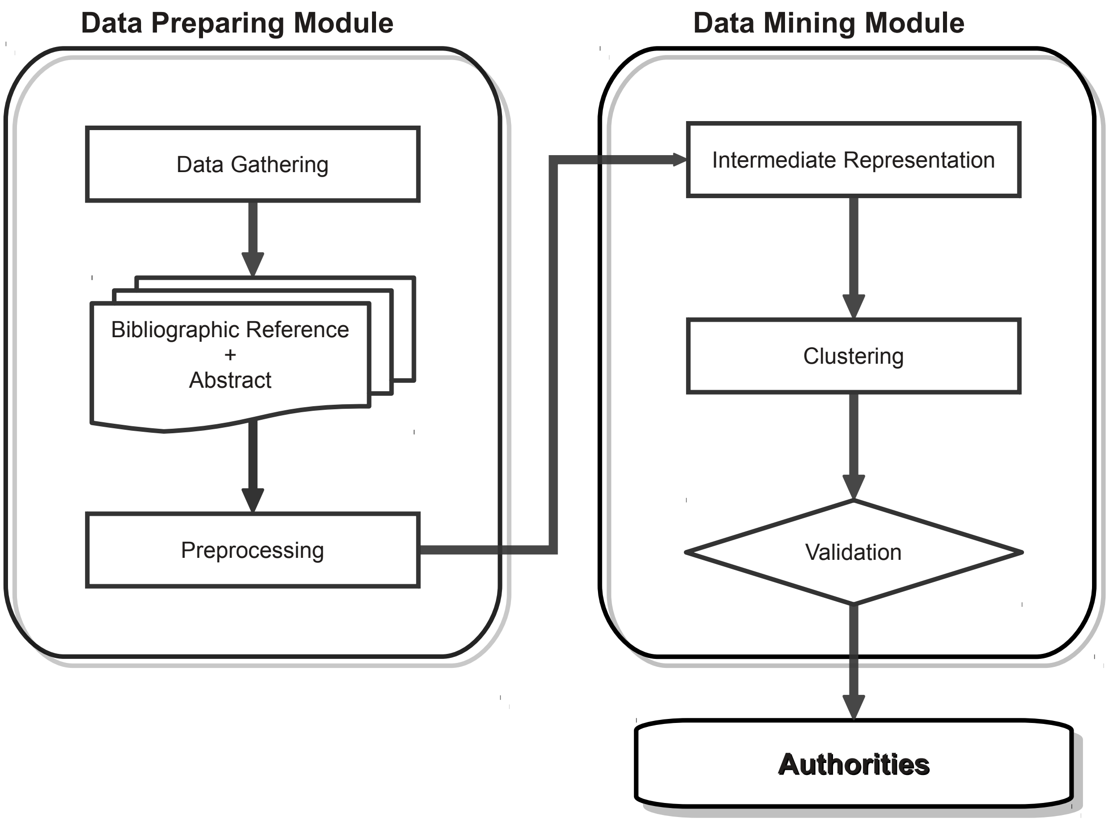


Figure 2.2: Description of the different modules of the system.

digital library and prepare them to be grouped into authorities in the *Data Mining Module*. System architecture is described in detail in Figure 2.2, including all subparts in each module. The following subsections will provide details of each of the modules composing this system.

2.1.1 Data Preparing Module

The first part of the system is the *Data Preparing Module* whose main function is to collect data from the DL and prepare it to be used in the *Data Mining Module*. However, the information that can be obtained from the web is textual, so it needs a representation to provide a proper data structure to allow an automatic processing.

It is possible to distinguish two different parts in the *Data Preparing Module*: *Data Gathering* process and *Preprocessing* stage. In the first part, a digital library is queried to collect its results. In the second part, the information gathered in the previous step is preprocessed to clean it and prepare it for the *Data Mining Module*.

Data Gathering

To query the digital library, the system asks requires an author *surname* from the user. Using this surname as a query, the system collects all search results from the source and extracts from them the information that will be necessary for our system. Specifically, this information includes: *bibliographic references* and *abstract*. Bibliographic references are structures keeping details of the document like title, publication year, author names, journal or proceeding name, etc. Their main advantage is that they keep information in a structured way allowing automatic processing. It is important to point out that, in some DLs, the abstract can be included in the bibliographic reference, so it is not necessary to extract it independently. Apart from that, in other DLs, abstract information is not available, so the system uses only the bibliographic reference.

Generally, search results are provided as HTML documents, but any textual format can be used. Once these documents are processed and the data have been gathered, the bibliographic reference is also processed to extract each one of its components, according to the type of document it represents. When all the information has been extracted, it is stored into an object-relational database. This database will keep the original records during the whole process.

The previously described process can be summarized into three steps: *search results processing*, *bibliographic reference processing* and *information storing.* After them, we have several details from publications, including title, authors, year, publisher, etc. However, our system only needs title and authors, as well as abstract, if available. We have restricted our system to work with these data because it is possible to find all these items in the bibliographic reference for any kind of publication in any source. The previously described information goes through a *Preprocessing* stage that will clean it and prepare it for the *Data Mining Module.*

Preprocessing

The *Preprocessing* stage has two purposes: to clean the information extracted in the previous module and to extract clean terms from title and abstract. Preprocessing is necessary in our system because we intend to work with noisy DLs. Noisy information may have problems of accuracy and completeness as well as missing values and/or codification errors. All these problems can be partially fixed by this process. For preprocessing purposes, we have distinguished between two types of data: *titles and abstracts* and *author names* that are preprocessed in two different ways.

Terms from title and abstract are extracted through series of filters that clean and standardize the words that may be found in a sentence or in a text. Some examples of these filters are: *Tokenizer* [100], *Number removal*, *StopWord removal* and *Porter Stemming* [95], etc. As a result of this preprocessing, two bags of terms are obtained: one for titles and one for abstracts. Further details of the process used for this specific solution can be seen in Section 2.6.1.

For author names, a basic filtering process has been used. This process removes extra-spaces and special characters including accents. After that, the resulting strings are transformed into an uppercase representation.

Once the preprocessing has been done, data is ready to go through a *Data Mining* process, which will be performed in the next module. It will be described in the next section.

2.1.2 Data Mining Module

The *Data Preparing Module*, was in charge of collecting data and preprocessing it in order to prepare it for the *Data Mining Module*. In this module publications are grouped by author in such a way that each group represents an authority.

The *Data Mining Module* has three different parts:

- **Intermediate representation:** Expresses data into an intermediate representation following a formal model that will be described on Section 2.2.1.
- **Clustering process:** Groups the data into authorities.
- **Validation:** Evaluates the authorities that are the output of the clustering process.

The previously stated elements will be described on detail in following sections.

2.2 Intermediate Representation

This Section formally proposes a general model of Intermediate Representation for a clustering problem that can be applied to many different domains. The Authority Control problem can be described under this model, as it can be seen on Section 2.2.2. Using that description, a representation for its elements is proposed allowing further processing. With this formal representation we are looking for a model that allow us to express the data in such way that it allows automatic processing through a clustering process.

2.2.1 Formal problem definition

The *Authority Control* problem can be considered from a wider general perspective. If an authority is seen as an entity represented by its different names, it is possible to define a general model describing a process to find these names.

Let us define:

- $E = \{e_1, \ldots, e_n\}$ an entity set.
- $H = \{h_1, \ldots, h_k\}$, a set of *names*, represented by strings.
- $C = \{c_1, \ldots, c_m\}$, a set of *cases*, i.e. data elements used to characterize an entity that are extracted from any textual information, so that $\forall c \in C \,|c = (h, f); \text{with}\, h \in H$, where h can be shared among various cases. f is a vector of *factors* extracted from texts. Example of factors can be *date, venue, document type, terms, etc.*

Using these definitions, the problem of finding the entity set E can be translated into finding the names corresponding to any partition of the space of cases as follows:

Let $\mathcal{P}(C) = \{C_1, \ldots, C_n\}$ be a partition of the space of cases so that for each $C_i \in \mathcal{P}(C)$, there is a $H_{C_i} = \{h | \exists c \in C_i; c = (h, f)\}$. Obviously, $\{H_{C_i}; i \in \{1, \ldots, n\}\}$ verifies that $\cup_{i=1}^{n} H_{C_i} = H$ and it could be $\exists i, j \in \{1, \ldots, n\} | H_{C_i} \cap H_{C_j} \neq \emptyset$. As a result, the set $\mathcal{R}(H) = \{H_{C_i}, i = 1, \ldots, n\}$ is a cover of the set H.

Partition $\mathcal{P}(C)$ is obtained using a clustering process. The divergence measure for this process should reflect the goal of grouping the cases c according to h, i.e. the divergence measure should allow grouping cases by name.

Each set H_{C_i} is identified with an entity of the set E. It is important to remark that in an ideal situation, $H_{C_i} \cap H_{C_j} = \emptyset$ because the names identifying an entity should not be related with any other entity. However, this hypothesis is not real at all, as the same name can identify different entities. Adding factors to the process could help to disambiguate among cases and the divergence measure used, should be appropriate to the names of the problem.

This formal model can be applied to any problem where it is possible to obtain data about an entity on the same format of C. In this problem, the model has been applied to finding authorities in bibliographic records, but some other examples of applications are: event or celebrity name recognition from news, company names from financial information, etc. Even if it is general enough to be fit several domains, for each new problem it would be necessary to redefine its elements according to the characteristics of the sought entity. Specially, it is very important to give a definition of $c = (h, f)$ where h is the new entity name (the companies or the celebrities in the previous examples) and f is defined according to the information available about that entity. The distance measures and the clustering algorithms should also be redefined according to the considered entity. Once the problem under consideration is modeled under these premises, the rest of the processes in this methodology could be applied to find the entities.

Applying the previous model to find all the different representations of the name of an author in a Digital Library is essentially an Authority Control process. For that purpose, it is possible to use publications as *cases*, allowing an easy identification of the elements of this formal model on the Authority Control problem:

- The set of entities E is the set of authors.
- The set C of cases is the set of all search results returned by a query. Each c_i is a publication returned by that search.

- $c = (h, f)$, where h is a name and f is a vector of factors. Specifically, h is the author name and factors, f, is a vector containing several elements obtained from the DL.

- $\mathcal{P}(C)$ is a partition to be obtained from the search results.

- $\mathcal{R}(H)$ can be obtained by extracting the names h from $\mathcal{P}(C)$.

On the next Section, the automatic authority control problem is described under this model.

2.2.2 Factor Selection

From the definition on Section 2.2.1, it is known that any C is in the format of $c = (h, f)$, where h is an author name and f is a vector of factors. Digital Libraries offer several elements that can be used for this purpose, like titles, authors, year of publication, venue, location, abstract, email, etc. Since this process is designed to fit most DLs, it is important to choose factors that are common to all of them. Specifically, after consulting all the information available in most Digital Libraries, two elements are common in all sources consulted: *titles* and *authors*. For that reason, these elements are used as factors of this problem. Furthermore, it is also possible to include *abstract*, if it is available. The generality of these elements is a very strong point of this system, as they can be found in mostly all DLs. In contrast, other terms candidates, like venue or author email, are not available in some of the sources.

Regarding the *authors*, it is possible to distinguish two different types: *author* and *coauthors*. *Author* is defined as the name that matches with the search query term and the *coauthors* are the remaining ones. If there is more than one author name in the same publication matching the search term, the first appearance is the

	author names	terms from title and abstract					coauthor names
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
c_i	JOHN DOE	$\dots$	w_3	w_1	w_3	$\dots$	{JAMES SMITH, K. JONES}
c_j	J. DOE	$\dots$	w_2	w_3	w_2	$\dots$	{KAREN JONES}
c_k	JOHN DOE	$\dots$	w_3	w_3	w_1	$\dots$	{MATH BROWN}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 2.1: Example of a data representation table

one selected as *author*.

So, with the previous selection of cases, two possibilities are proposed for the definition of c:

- **Hypothesis a:** *c = (author name, [{title, abstract}]).*
- **Hypothesis b:** *c = (author name, [{title, abstract}, coauthors]).*

Both hypotheses include the author name, on the basis that an author can be characterized by its name. Additionally, in *hypothesis a*, title and abstract have been used on the premise that authors normally write about a topic that can be outlined from them. *Hypothesis b* follows the same principles as *Hypothesis a*, but it also considers coauthors, because authors usually publish their articles in groups of the same people.

Once the elements have been identified, the real data obtained from DLs need a structure for automatic processing. Let us remember that this is textual data, so they need some kind of treatment before applying clustering processing. For that reason, an intermediate representation for C has been defined as a table where each row represents a case c_i and columns are author names, coauthor names, titles, and abstracts. An example of this representation can be seen on Table 2.1.

For representation purposes, instead of directly using title and abstract, they are preprocessed in the way of their *terms*. To represent these terms, the "vector space model" proposed by [100] is used. Under this model, a vector is built with all terms considered in the problem and weights are assigned for every case. To assign these weights for this specific problem, we propose the use of W defined in (2.1).

$$W(title, abstract, term) = \begin{cases} w_1 & \text{if } term \in \text{ title} \\ w_2 & \text{if } term \in \text{ abstract} \\ w_3 & \text{otherwise} \end{cases} \tag{2.1}$$

2.3 Clustering

The clustering process is in charge of grouping the publications, properly represented as described in previous sections. Clustering is a data driven process that requires some information about *how to compare the elements*, i.e, some divergence measures providing proper comparison of the data. In Section 2.4 several divergence measures for the different solutions proposed for this problem are studied.

Additionally, there are many techniques of clustering in the literature [61], each one with its specific peculiarities that makes it very suitable for a kind of problem. It is important to choose the one that better fits the requirements of the solution that it is being sought. In Section 2.3 the characteristics of the problem will be studied in order to try to find the best clustering algorithm for this problem.

Clustering Process

The partition $\mathcal{P}(C)$ of the input space C is obtained using clustering techniques. Clustering is an unsupervised learning technique that divides a set of elements X into k subsets $X = \{X_1, X_2, \ldots, X_k\}$ called clusters. All elements inside a cluster are similar according to some proximity measure and different from the other clusters.

For this problem, there is no prior information about how many groups, k, are in C. For that reason, only those techniques that do not use this information can be applied. Hierarchical clustering is a family of clustering algorithms that do not need information about the number of groups to be obtained. Instead of finding a direct partition of the set, they create a hierarchy of elements in a binary tree called dendrogram.

Any hierarchical clustering technique can be used on this methodology, as long as it does not require the use of a distance metric. For this purpose, classic algorithms have been chosen: *Complete Linkage*, *Single Linkage*, and *Group Average*, as they are very powerful techniques that fit the divergence requirements of the problem. The main drawback of using this kind of algorithms is scalability. However, as the information is extracted from surname searches and they do not return very big lists of publications, it is not an issue. Nevertheless, in those cases where scalability is really and issue and classic techniques cannot be applied, ROCK algorithm [52] can be used as an alternative, as it allows non-Euclidean distances. An experimental approach to choose the best clustering algorithm can be found on Section 2.6.4.

A partition can be extracted from the dendrogram by cutting it at a specific α-level. There are several methods that can be used to obtain the optimal α-cut of a dendrogram. The problem of finding optimal number of clusters on hierarchical clustering has been treated from several perspectives in the literature. One of the most popular methods is the so called "Elbow-criterion" [56]. It tests all the possible cuts of the dendrogram, validating its solution with the Sum of Squared Differences (SSD). Therefore, the SSD is plotted against the numbers of Clusters in the analysis and an optimal number of clusters is determined by identifying the “elbow” in the plot. However, this approach is very costly when the validation cannot be performed by using SSD, but an expert is needed for that.

One of the first and most extensive reviews of procedures for determining the optimal number of clusters in a data set was performed in 1985 by Milligan and Cooper [80]. mong the methods proposed in that paper, *Upper-Tail* method [83] is a simple method that has been applied in several domains. After that, other reviews like [56], [39] have focused on specific kinds of criteria like those for binary datasets or hypervolume. On [28], the authors make use of several fuzzy-set tools to propose four different objective functions that can give a ranking of optimal partitions of the dendrogram. Among the proposals made on that paper, one of the most interesting ones is the criterion that considers measures of cluster dispersion looking for the partition that minimizes a certain measure of intracluster distances and maximizes the intercluster distance.

2.4 Divergence Measures

A very important point in the clustering processes is the choice of the divergence measure, as it allows a proper comparison among elements.

From the previously defined hypotheses, it is possible to distinguish different kinds of textual information inside a case c, namely *terms* and *personal names*. Even if they are both strings, each one has its own semantic, so it seems reasonable to compare them using a measure that can take advantage of their meaning. For that reason, two different divergence measures are proposed: *term distance* and *personal name divergence*.

2.4.1 Term divergence measure

Terms, as described on Section 2.2.2, come from titles and abstracts. For *term divergence measure*, d_t, the use of the *Cosine distance* [100] is proposed, as it is a well known measure used to compare documents, especially for comparing vectors of terms. Vectors used for this distance follow the model described on Section 2.2.2

2.4.2 Personal name divergence measures background

There are several possible definitions for *personal name divergence.* The related literature [19, 21] recommends the use of classic string comparison measures like Levenshtein distance [72] and Jaro Winkler distance [111]. Specifically, they say that the characteristics of the strings to be compared must be considered in order to choose a proper measure. Previous works, [19] recommends Jaro-Winkler distance because it has been proved to work best with personal names.

This measure compares two strings $s1$ and $s2$ using a variant of Jaro distance [63] designed to compare personal names. It is defined by Expression 2.2 where m is the number of matching characters and t is the number of transpositions.

$$d_j(s1, s2) = \frac{1}{3}\left(\frac{m}{|s1|} + \frac{m}{|s2|} + \frac{m-t}{m}\right) \tag{2.2}$$

Two characters from $s1$ and $s2$ are a match if their position in the string is not further than the value obtained from expression 2.3. The number of transpositions is defined by the number of matching characters with different order divided by two.

$$\left\lfloor \frac{\max(|s1|, |s2|)}{2} \right\rfloor - 1. \tag{2.3}$$

However, even if classic string comparison measures can be used for comparing personal names, some of them have many problems. Specifically, if the names have a common part, i.e. they have the same first name or surname, these measures do not discriminate well among them. This fact has been tested experimentally on Section 2.6.3.

2.4.3 Author's personal name divergence (pn-measure)

Pn-measure is a new divergence measure designed to overcome the problems of comparing personal names with classic name comparison measures. It has been designed to fit the most important characteristics of personal names, making it very suitable for comparing name strings.

Names have a very specific and clearly established syntax. Personal names are usually composed by up to four words corresponding to first name, second name, first surname, and second surname. This number may vary depending on the origin of the person or personal signature preferences. Furthermore, it is also quite usual to use initials instead of the full name, as well as other modifications.

To design this divergence, the most common variations that occur in a personal name have been considered, as *Initials*, *Short names*, etc. Some of the properties of international names have been considered, like the existence of two surnames. However, those names that invert the order first name, surname order have not been considered for this problem, as normally publications ask those authors to write their names as: first name, surname. Using these rules, this measure has been designed algorithmically as a set of rules for comparing personal names.

The proximity between two personal names is obtained by means of the *PN-*

MEASURE defined in Function 1. It takes two strings, $name_1$, $name_2$ and an optional *common_term*. The first step is to check if the *common_term* is in both $name_1$, $name_2$. If it is part of both, it means that they have the same surname. According to that, each string is divided into *fstname* and *surname*. So, if they have the same surname, they are divided considering where it appears. Otherwise, they are divided according to their number of words (See rules in Function 1). In the specific case when $surname_1$ or $surname_2$ have the same value as the common term, they are considered as any other surname, because the common term is identified with its first match. After dividing both $name_i$ and $name_2$ into $fstname_1$, $surname_1$ and $fstname_2$, $surname_2$, they are compared following the criteria in (2.5). From this comparison, v_{fst} is obtained from checking the similarity of $fstname_1$ and $fstname_2$; and v_s is obtained from $surname_1$, $surname_2$. Finally, v_{fst} and v_s are compared using the criteria in (2.5). It returns the divergence between $name_1$ and $name_2$.

$$substring\_cmp(s_1, s_2) = \begin{cases} 1.0, & \text{if } s_1 = s_2 \\ 0.5, & \text{if } s_1 \text{ is initial of } s_2 \\ 0.5, & \text{if } s_2 \text{ is initial of } s_1 \\ 0.8, & \text{if } s_1 \text{ is substring of } s_2 \\ 0.8, & \text{if } s_2 \text{ is substring of } s_1 \\ 0, & \text{Otherwise} \end{cases} \tag{2.4}$$

$$comp(v_{fst}, v_s) = 1 - \begin{cases} v_{fst}, & \text{if } v_s = \emptyset \\ (v_{fst} + v_s)/2, & \text{if } v_{fst} \geq 0.5 \text{ and } v_s \geq 0.5 \\ 0, & \text{if } v_{fst} < 0.5 \text{ or } v_s < 0.5 \\ 0.8, & \text{if } surname_1 = \emptyset \text{ or } surname_2 = \emptyset \text{ and } v_{fst} = 1 \\ v_{fst}/2, & \text{if } surname_1 = \emptyset \text{ or } surname_2 = \emptyset \text{ and } v_{fst} < 1 \end{cases} \tag{2.5}$$

Algorithm 1 Name divergence measure

```
function PN-MEASURE($name_1$, $name_2$, $common\_term$) return $divrg$
    for $string$ in $\{name_1, name_2\}$ do
        if $common\_term \neq$ null and $common\_term \in string$ then
            Find the first appearance of $common\_term$ in $string$.
            $fstname_i$ = substring before the common term.
            $surname_i$ = substring after the common term or $\emptyset$ if there is none.
        else
            switch number of words in $string$ do
                case 1
                    $fstname_i$ = WORD1.
                    $surname_i = \emptyset$.
                case 2
                    $fstname_i$ = WORD1.
                    $surname_i$ = WORD2.
                case 3
                    $fstname_i$ = WORD1.
                    $surname_i$ = WORD2 WORD3.
                case 4
                    $fstname_i$ = WORD1 WORD2.
                    $surname_i$ = WORD3 WORD4.
                case 5
                    $fstname_i$ = WORD1 WORD2.
                    $surname_i$ = WORD3 WORD4 ...
        end if
    end for
    $v_{fst} = substring\_cmp(fstname_1, fstname_2)$ using (2.4).
    $v_s = substring\_cmp(surname_1, surname_2)$ using (2.4).
    $divrg = pn\text{-}measure(v_{fst}, v_s)$ using (2.5).
end function
```

It is important to remark that weights on (2.4) and (2.5) have been obtained experimentally [29]. The weights on (2.4) are designed to consider the level of similarity between the two parts of the names, so their values try to reflect the importance of their kind of match. For that reason, the most common types of match have been ranked and they have an scored assigned according to their importance. The higher score is assigned to the exact match, then substring (one name is part of the other) and initials. The values in (2.5) have been assigned to equalize the importance of the *first name* and the *surname*. If the author does not have a second surname, it is possible that $surname_1$ or $surname_2$ do not have any value (represented as $\emptyset$). When the surname is not available on both names, only the *first name* is considered and its final weight is the one assigned by (2.4). When only one of the surnames is missing, the score of v_{fst} is penalized, reducing it 20% in the case of exact first name match and 50% otherwise. These reductions are done under the premise that it is more likely to be the same person if the name is exactly the same than in any of the other considered cases. Similar rules apply when there are both v_{fst} and v_s. If one of the values is lower than 0.5 they are considered different, otherwise the final score is the average of both v_{fst} and v_s values. These weights can be modified in any other domains or for other kind of names. New rules can also be added as well based on the considered problem.

Pn-Measure properties

pn-measure is a pseudo-distance because it fulfills some of the distance conditions, although it does not satisfy the triangle inequality. Specifically:

a) $\textit{pn-measure}(name_1, name_2) \geq 0$, and $\textit{pn-measure}(name_1, name_2) = 0$ if and only if $name_1 = name_2$, as can be seen in the definition

b) Symmetric: $\textit{pn-measure}(name_1, name_2) = \textit{pn-measure}(name_2, name_1)$ from (2.4) and (2.5)

c) It does not satisfy the triangle inequality: $\textit{pn-measure}(name_x, name_z) \nleq \textit{pn-measure}(name_x, name_y) + \textit{pn-measure}(name_y, name_z)$ in the case that we have three names, one of them is substring of the other two.

Let us prove this last point with a counterexample. Given three names $name_1 =$ JOHN DOE, $name_2 =$ JO DOE and $name_3 =$ JOAN DOE:

- Remove the common term *DOE*.
- $fname_1 =$ JOHN, $fname_2 =$ JOAN and $fname_3 =$ JO
- $\textit{pn-measure}(name_1, name_3) = \textit{pn-measure}(JOHN, JOAN) = 1$
- $\textit{pn-measure}(name_1, name_2) = \textit{pn-measure}(JOHN, JO) = 0.2$
- $\textit{pn-measure}(name_2, name_3) = \textit{pn-measure}(JO, JOAN) = 0.2$
- So $\textit{pn-measure}(name_1, name_3) \nleq \textit{pn-measure}(name_1, name_2) + \textit{pn-measure}(name_2, name_3)$ because $1 \nleq 0.2 + 0.2$

The fact that it is not a distance, but a divergence measure, must be taken into consideration when choosing a clustering algorithm. The vast majority of the existing clustering methods require distances or euclidean distances, making them not suitable for our purposes. So, other kind of algorithms must be chosen for this methodology, which will be covered in Section 2.3.

pn-measure use example

The use of *pn-measure* can be illustrated with an example. Let $name_1 =$ JOHN DOE DOE and $name_2 =$ JANE DOE DOE, be two author names of publications obtained under the search *DOE* on a Digital Library. With these parameters, the use of this measure is as follows:

1. Given the common term *DOE*. Input strings: $name_1$ = JOHN DOE DOE; $name_2$ = JANE DOE DOE.

2. Find the common term: $name_1$ = JOHN <u>DOE</u> DOE; $name_2$ = JANE <u>DOE</u> DOE.

3. Assign the substring before the common term to $fstname_i$.
   $fstname_1$ = JOHN; $fstname_2$ = JANE;

4. Assign the substring after the common term to $surname_i$.
   $surname_1$ = DOE; $surname_2$ = DOE.

5. Compare both $fname_1$, $fname_2$ using (2.4).
   $v_{fst} = substring\_cmp(\text{JOHN, JANE}) = 0$

6. Compare both $surname_1$, $surname_2$ using (2.4).
   $v_s = substring\_cmp(\text{DOE, DOE}) = 1$

7. The similarity of $name_1$, $name_2$, is obtained by comparing v_f and v_s using (2.5).
   $divrg = comp(v_{fst}, v_s) = pn\text{-}measure(0, 1) = 1$

2.4.4 Divergence measure proposals for the different hypotheses

Hypothesis a: defined as c = *(author name, [{title, abstract}])*. A divergence measure suitable for this proposal should combine the previously defined *personal name measure* and *term distance*. Let us define a general model for it:

$$m_a(c_1, c_2) = \begin{cases} 1, & \text{if } m_h = 1; \\ (1-\lambda) \times m_h + \lambda \times d_t, & \text{otherwise.} \end{cases} \tag{2.6}$$

Where c_1 and c_2 are two cases, m_h is *personal name measure* which can have several definitions as stated on Section 2.4.2 and 2.4.3; d_t is a *term distance*, described on Section 2.4.1; and λ is a weighting parameter in [0-1] used to control the influence of the components in the measure. This weight must be determined experimentally considering the importance of each factor for the specific problem.

Hypothesis b : states that *c = (author name, [{title, abstract}, coauthors])*. Let us define m_b a measure based on three components: m_h *personal name measure*, d_t *term distance*, and m_{co} *coauthor measure*:

$$m_b(t_1, t_2) = \begin{cases} 1, & \text{if } m_h = 1; \\ (1 - \lambda_1 - \lambda_2) \times m_h + \lambda_1 \times d_t + \lambda_2 \times m_{co}, & \text{otherwise.} \end{cases} \tag{2.7}$$

Where λ_1, λ_2 are two weighting parameters in [0,1] used to control the influence of each component on the measure. To obtain the coauthor measure m_{co}, *personal name measure* has been used because coauthors are lists of names. Every item in one list of coauthors is compared to all the items in the other list using *pn-measure*. The score of the most similar elements within the two lists is the resemblance among coauthors. Formally, m_{co} can be defined from sim_{co}, that is defined as:

$$m_{co}(A_1, A_2) = \min(\textit{pn-measure}(i, j)) \forall i \in A_1, j \in A_2 \tag{2.8}$$

Where *pn-measure*(i, j) is *pn-measure*, and A_1, A_2 are two lists of coauthors. In the special case where there is no coauthor to compare, this measure return 0.

These hypotheses as well as all the possible definitions for their measures have been experimentally proved on Section 2.6.3.

2.5 Validation

Following the steps described on previous sections, a partition $\mathcal{P}(C)$ has been obtained by means of a clustering process. Each one of these clusters contains the publications by a specific author. By extracting the names from these clusters, it is possible to obtain all the representation of the name of the author in that Digital Library, i.e. the *authority*. However, as these authorities are not previously labeled, it is necessary to find a way to know if all names in a group correspond to the same person. As there is not a proper "correct solution" or ground truth, an expert is required for this task. This expert is a person with full knowledge about publications and access to more details about a specific publication. Using her/his knowledge, it is possible to check the results of this method. To measure how "good" is a solution, every publication inside a cluster is assigned to one of these three categories:

- **Correct:** Elements that belong to a group.
- **Incorrect:** Elements misplaced into a group.
- **Not grouped:** Elements that form a cluster by themselves, but should be inside another group.

To determine the elements that belong to each of these categories, two different approaches have been followed: *manual* and *semiautomatic*.

2.5.1 Manual validation approach

In the *Manual* approach, an expert validates the authorities by checking all publications within every group and between groups. The expert must decide whether the publications inside an authority are properly placed there by marking them as *Correct*, *Incorrect*, and *Not grouped* according to his/her expert knowledge. After that, the percentage of the total number of elements on each category is obtained. For performing this task, the experts may use any information available that could help them to determine authorship, like the paper itself, traditional search engines, personal websites, etc. It is important to note that this manual process has to be performed for every execution of the system, taking from hours to several days depending of the number of publications and authors in the dataset, so it was only used on preliminary stages of the research.

2.5.2 Semiautomatic validation approach

The *semiautomatic* approach eases the validation process by building a *Reference Set* considered the ground truth for a specific search that could be reused among executions. This approach has two parts. In the first part, the expert manually builds a Reference Set, *RS* for each surname search. These *RS*s are used to automatically validate the authorities. Let us define them formally as:

$$RS = \{r_1, \ldots, r_n\} \tag{2.9}$$

where r_i are its elements defined as:

$$r_i = (id, names, coauthors, amb) \tag{2.10}$$

where:

- *id* is a unique identifier.
- *names* are all the known forms or representations for this authority personal name.
- *coauthors* is a list of all coauthors known for this authority.

- *am* is an *ambiguous mark* used to mark when an authority is ambiguous.

A reference set entry is said to be ambiguous when it is possible to mix it with another one, that is, when the list *names* contains at least one element that can be found on the list *names* of another entry. Formally, let r_i, r_j be ambiguous if $names_i \bigcap names_j \neq \emptyset$. The concept of an ambiguous entry is very related with the *Identity Uncertainty* problem. Using an example for better understanding, two entries are ambiguous when both contain the form "J. Smith" for the name of an author.

To build the Reference Set, the clusters are given to the expert under the previous notation. The expert modifies them by moving all misplaced publications into their correct cluster. Using previously built clusters helps the expert to perform this task, as when there are hundreds or thousands of papers, it is easier to correct the mistakes in a given solution than built sets from scratch. The expert is also asked to mark as *ambiguous* the authorities (clusters) that share some form of a name. This ambiguous marking can be performed automatically using the correct solution from the expert.

Figure 2.3 describes with an example the process to be followed for marking each publication as *correct*, *incorrect*, or *not grouped.* The automatic validation method, uses the concept of *popular elements* to find out which groups of publications correspond with an *authority* for *RS*. *Popular elements* are then defined as those publications sharing the same author and it is the one appearing more times in this particular group. Therefore, for each authority in the reference set, the validation method proceeds in five steps:

- **Step 1:** *Find the preliminary set of authorities.* All authorities are checked and those that match the names in the entry set authority are added to the preliminary set. If there are some of them whose elements have already been marked as processed, they are excluded of the set.

Figure 2.3: Description of the validation process.

- **Step 2:** *If the entry of the reference set has been marked as ambiguous, check co-authors to disambiguate.* In the case that the reference set entry has been marked as ambiguous, it checks co-authors. To do so, it removes from the set of working authorities those that do not have any common co-author with the reference set entry.

- **Step 3:** *Consider forms not in the reference set entry.* From the preliminary set, find the *non-matching publications.* For every authority that contains at least a non-matching publication, check if the form of the non-matching publication is the *popular form.* If that happens, it means that this authority does not correspond to the Reference Set entry, so its publications are errors. For that reason, the rest of the publications are marked as *Incorrect* and the non-matching are skipped for latter processing. Otherwise, it means that the non-matching publications are errors and should be marked as *Incorrect.* The elements skipped at this time, will be marked in another iteration of the algorithm.

- **Step 4:** *Mark the correctly grouped publications.* Using the remaining publications in the authorities of the preliminary set; if there is only one form for all publications independently of their authority, mark the non marked elements as *Correct.* Otherwise, find the most popular element and mark its publications as *Correct.*

- **Step 5:** *Mark Not Grouped elements.* At this step, if there are unmarked publications, which have not been skipped in Step 3, mark them as *Not grouped.* This situation is reached only when the rest of the publications in the authorities have been previously marked as *Correct* or *Incorrect*, so the remaining elements should be in other group.

For this validation method, the expert must build a *Reference Set* for each search. For practical reasons, it is not possible to predict all searches that can be made over a DL. So, a new RS is built for any new query and it is reused in future searches. Once the reference sets are built, validation can be performed fully automatically, even if any other parameter of the method has changed.

The previously described process is able to obtain the amount of publications that have been correctly, incorrectly, or not grouped at all in an authority. By repeating it for all the authorities returned from our methodology, it is possible to validate them and to obtain the amount of elements in each of the three groups. This process will be used on Section 2.6 to validate our experiments.

2.6 Experimental Results

This section contains experimental tests for different aspects of the methodology. It starts providing a description of the data extraction and cleaning processes. Using that clean data, all factor and measure proposals made on previous sections are tested. As a result of that tests, a study of performance under digital libraries is conducted. Finally, the expert selection is tested.

2.6.1 Data extraction

This methodology works with data from most Digital Libraries. The preferred way to obtain the data is by means of automatically parsing the *bibliographic reference*, i.e. Bibtex, of each publication. However, if the source does not offer this characteristic, data can also be gathered manually. Title, authors, and abstract, if available, are extracted for each publication and kept for further processing.

Sometimes, Digital Libraries obtain their data from poorly supervised automatic capturing processes, meaning that it may have codification errors, accuracy, and completeness problems, missing values or different levels of noise. For all these reasons, it is difficult to perform an automatic processing of data without a prior cleaning process.

For preprocessing purposes, as stated on Section 2.1.1, two different types of data have been differentiated: *textual information* (titles and abstracts) and *author names*.

To extract clean terms from titles and abstracts, a series of well-known filters have been applied, to extract terms as well as clean and standardize them. These filters and the order in which they are applied are:

- **Tokenizer [100]:** This filter splits strings into tokens. A token can be defined as the substring between two spaces, the beginning of the string and a space or a space and the end of a string. This filter uses a string as input and returns a set of tokens. During this process, a deep clean is also performed, removing special characters, accents, and punctuation marks. Additionally, we have used a version of this filter that changes the token into an uppercase representation.
- **Number removal:** The aim of the number removal filter is to remove those strings whose characters are exclusively numbers. The reason to do this is that we are interested in words that can represent topics, and strings whose characters are only numbers are usually years, version numbers, etc.
- **StopWord removal:** A StopWord [77] is a word used to link words in sentences but which does not have a concrete meaning. Examples of StopWords are articles, determiners, prepositions, etc. They are very common words without meaning, and thus, it is necessary to remove them. There are several lists of stopwords available online. For this filter, the one for the Snowball project

[94] has been used.

- **Stemming:** Stemming is the process that reduces words to their roots, removing their endings. Porter's algorithm [95] is the most extended one, with several implementations in different programming languages, the fact that led us to use it.

All filters are applied to each publication's title and abstract. As a result, for each publication, there are two bags of terms, one for titles and one for abstracts.

For names data, a basic filtering process has been used that removes extra-spaces and special characters including accents. After that, the resulting strings are transformed into uppercase representations.

2.6.2 Datasets

To test this methodology, three different digital libraries have been selected: *CiteSeerX*, *DBLP*, and *INSPEC*. These three have been chosen as they provide a standardized bibliographic reference and each one presents a different level of noise. Additionally, it is important to remark that some of them offer partial name resolution. All papers gather have a different number of words in title and abstract as well a different number of authors/coauthors that goes from one to a few tens.

Every source has been queried with twelve different surnames from three different origins: Asian, English and Spanish. Chosen surnames correspond to the most popular ones according to Wikipedia. Specifically, Asian surnames are the most popular in China and Korea; the English surnames are the most popular in the UK and the Spanish are the most popular in Spain as well as others added by our own knowledge about them, as our team have acted in the role of experts for

		Number of instances		
Group	Surname	DBLP	INSPEC	CiteSeerX
Asian	Kim	4160	2174	500
	Lee	5102	1617	500
	Li	5089	788	500
	Wang	5628	1398	496
English	Brown	3075	200	481
	Johnson	3945	194	480
	Smith	5003	197	485
	Williams	2777	181	479
Spanish	Delgado	767	251	190
	García	3685	220	364
	Sánchez	3892	240	222
	Vila	260	280	108

Table 2.2: Dataset description

validation. Dataset characteristics can be seen on Table 2.2. From every search result, authors, title, and abstract have been obtained, except for DBLP, where abstract information is not available. The number of publications on each dataset is the number of publications returned by the source on a single search. Some sources, like CiteSeerX provide a maximum of 500 documents for each query.

2.6.3 Problem one: Comparison of the proposed different hypotheses and divergence measures.

On Section 2.2.2 two different hypotheses for the definition of c were made:

- *Hypothesis a:* $c = (h, f)$ where h is the author name and f are terms from title and abstract.
- *Hypothesis b:* $c = (h, f)$ where h is the author name, f contains terms from title and abstract and a list of coauthors.

These two hypotheses are tested experimentally to determine which one better fits the problem. Additionally, distance measures described on Section 2.4.2 are tested with both options, in order to determine the one that should be used on each case.

These experiments were performed using English and Spanish surnames from *CiteSeerX*, as it is quite representative of the kind of information found in a Digital Library. The parameters for the expressions W (2.1), m_a (2.6), and m_b (2.7) have been determined experimentally in preliminary tests [29]. Suggested values for them are $w_1 = 2$, $w_2 = 1$, $w_3 = 0$, as experiments show that the words from the title (w_1) are more informative than the ones from abstract (w_2) as well as those that are not in the document (w_3) should not have any impact on the score. Parameter λ has been set to $\lambda = 0.5$ to give the same importance to both parts of the equation λ_1 and λ_2 are set to $\lambda_1 = 0.2$, and $\lambda_2 = 0.3$ to give slightly more importance to names of authors and coauthors than to terms.

The hierarchical clustering algorithm used is *Complete Linkage*; however, this choice will be discussed later on Section 2.6.4. The obtained dendrogram has been cut using the *average of the intra-cluster and inter-cluster distance* criterion. Finally, validation has been performed using the semiautomatic method (see Section 2.5), obtaining the percentage of *Correct*, *Incorrect* (Inc.), and *Not Grouped* (N.G.) (as described on Section 2.5). Additionally, the *Purity*(2.11]) [78] and the *Inverse Purity* (2.12]) have also been calculated [1].

$$purity(C, \Theta) = \frac{1}{N} \max_j |c_k \cap \theta_j| \tag{2.11}$$

$$purity(\Theta, C) = \frac{1}{N} \max_j |\theta_k \cap c_j| \tag{2.12}$$

Where $C = c_1, \ldots, c_k$ is the partition obtained from the clustering process and $\Theta = \theta_1, \ldots, \theta_j$ are the classes according to dataset labels. N is the number of instances.

The first test has been performed for Hypothesis a, $c = (h, f)$ where h is the author name and f is a vector of terms extracted from title and abstract. Two different definitions of m_h are tested: Jaro Winkler distance and *pn-measure*. Table 2.3 shows the performance for each dataset. As it is possible to see, *pn-measure* clearly outperforms results obtained from Jaro-Winkler, with a correct rate close to 87% compared to the 45% of the latter. Table 2.4 shows a statistical comparison of both approaches using a one way ANOVA with level of significance 0.03. From this table, it follows that the differences between the two approaches are significant for both *Correct* and *Incorrect* elements, which is a clear outperform of *pn-measure* even if the differences in *Not Grouped* are not significant. This is because by using *Jaro Winkler* distance, the methodology is returning very few clusters grouping several authors on the same cluster.

Table 2.5 shows the results for the experiments using *Hypothesis b*: $c = (h, f)$ where h is the author name, f is a vector containing terms extracted from title and abstract and a list of coauthors. As can be seen in the table, the use of Jaro Winkler distance for m_h is not a good choice for this problem. That is related with the existence of a common substring in all author names (the search term). For that reason, Jaro Winkler distance returns very similar scores for any two strings. Using *pn-measure* overcomes this problem and allows the distinction between different names even if they have a common substring. For m_{co}, *coauthor measure*, it is possible to see that *pn-measure* returns better results than Jaro Winkler, but these differences are not as significant as with authors, because coauthors do not normally have common substrings. Table 2.6 shows an ANOVA test for this hypothesis, where

Algorithm	Surname	% Correct	% Inc.	% N. G.	Purity	I.Purity
(1) m_h = Jaro Winkler	Brown	36.97	63.03	0.00	0.36	0.97
	Delgado	33.33	58.20	8.47	0.42	0.87
	García	62.37	35.71	1.92	0.64	0.95
	Johnson	50.00	47.92	2.08	0.53	0.95
	Sánchez	43.69	55.41	0.90	0.45	0.97
	Smith	45.25	54.75	0.00	0.45	0.96
	Vila	52.78	43.52	3.70	0.56	0.89
	Williams	39.04	59.71	1.25	0.4	0.93
	Average	**45.43**	**52.28**	**2.29**	**0.48**	**0.94**
(2) m_h = pn-measure	Brown	86.98	10.50	2.52	0.9	0.96
	Delgado	82.01	5.82	12.17	0.94	0.88
	García	88.43	6.06	5.51	0.94	0.94
	Johnson	87.27	8.98	3.76	0.91	0.94
	Sánchez	85.59	11.71	2.70	0.88	0.95
	Smith	85.54	9.30	5.17	0.91	0.93
	Vila	90.74	2.78	6.48	0.97	0.94
	Williams	86.64	7.93	5.43	0.92	0.94
	Average	**86.65**	**7.88**	**5.47**	**0.92**	**0.94**

Table 2.3: Results for the different measure definitions in Hypothesis a

		Sum of Sqr.	df	Mean Square	F	Sig.
Correct	Between Grp.	6796.766	1	6796.766	142.443	0.000
	Within Grp.	668.020	14	47.716		
	Total	7464.786	15			
Incorrect	Between Grp.	7884.552	1	7884.552	169.735	0.000
	Within Grp.	650.330	14	46.452		
	Total	8534.882	15			
NotGrouped	Between Grp.	40.354	1	40.354	4.741	0.470
	Within Grp.	119.172	14	8.512		
	Total	159.526	15			

Table 2.4: ANOVA For Hypothesis a

Algorithm	Surname	% Correct	% Inc.	% N. G.	Purity	I.Purity
(3) m_h = Jaro Winkler m_{co} = Jaro Winkler	Brown	33.19	65.97	0.84	0.34	0.9
	Delgado	36.5	49.21	14.29	0.51	0.86
	García	59.07	39.56	1.37	0.60	0.94
	Johnson	45.62	51.25	3.13	0.49	0.92
	Sánchez	42.34	56.76	0.90	0.43	0.96
	Smith	44.42	55.58	0.00	0.44	0.92
	Vila	41.66	50.93	7.41	0.49	0.82
	Williams	41.34	58.66	0.00	0.41	0.88
	Average	**43.02**	**58.66**	**3.49**	**0.46**	**0.9**
(4) m_n = Jaro Winkler m_c = pn-measure	Brown	33.61	66.39	0.00	0.34	0.95
	Delgado	44.44	48.15	7.41	0.52	0.89
	García	57.42	39.01	3.57	0.61	0.95
	Johnson	48.34	50.83	0.83	0.49	0.93
	Sánchez	43.69	56.31	0.00	0.44	0.98
	Smith	40.50	59.50	0.00	0.41	0.96
	Vila	40.74	53.70	5.56	0.46	0.8
	Williams	40.92	57.62	1.46	0.42	0.93
	Average	**43.71**	**53.94**	**2.35**	**0.46**	**0.92**
(5) m_n = pn-measure, m_c = Jaro Winkler	Brown	86.74	10.52	2.74	0.89	0.97
	Delgado	85.72	4.76	9.52	0.95	0.89
	García	86.98	6.93	6.09	0.93	0.94
	Johnson	84.70	8.38	6.92	0.92	0.9
	Sánchez	86.04	10.36	3.60	0.9	0.95
	Smith	86.57	8.26	5.17	0.92	0.94
	Vila	87.04	4.63	8.33	0.95	0.9
	Williams	85.56	8.58	5.86	0.91	0.92
	Average	**86.17**	**7.8**	**6.03**	**0.92**	**0.93**
(6) m_n = pn-measure m_c = pn-measure	Brown	89.71	9.03	1.26	0.91	0.98
	Delgado	82.54	6.88	10.58	0.93	0.88
	García	88.65	6.09	5.26	0.94	0.94
	Johnson	88.10	7.72	4.18	0.92	0.94
	Sánchez	84.69	11.26	4.05	0.89	0.95
	Smith	84.10	8.88	7.02	0.91	0.93
	Vila	90.74	2.78	6.48	0.97	0.94
	Williams	87.34	7.17	5.49	0.93	0.94
	Average	**86.98**	**7.48**	**5.54**	**0.93**	**0.94**

Table 2.5: Results for the different measure definitions in Hypothesis b

		Sum of Sqr.	df	Mean Square	F	Sig.
Correct	Between Grp.	14943.987	3	4981.329	171.534	0.000
	Within Grp.	813.115	28	29.040		
	Total	15757.102	31			
Incorrect	Between Grp.	16984.476	3	5661.492	163.248	0.000
	Within Grp.	971.051	28	34.680		
	Total	17955.527	31			
NotGrouped	Between Grp.	71.637	3	23.879	2.099	0.123
	Within Grp.	318.495	28	11.375		
	Total	390.132	31			

Table 2.6: ANOVA For Hypothesis b

it is possible to see that the differences are significant for the tested measures.

If results from *Hypothesis a* (Table 2.3) are compared with results from *Hypothesis b* (Table 2.5), it is possible to see that both hypotheses have a very similar performance, with average differences of less than 1%. However, the best results are obtained using *Hypothesis b*, as it definitely helps to discriminate in extreme cases where not only the names are very similar, but also the words from title and abstract. A 2-sided Dunnet test has been performed to compare statistically all previous experiments. For simplicity, each experiment has been numbered from 1 to 6, as it can be seen on Tables 2.3 and 2.5.

The Dunnet test compares different approaches with a control group, in this case, approach 6, which seems to be the best one. Results at 0.03 significance level can be seen on Table 2.7. From this table, it is possible to see that the differences are significant between measures 1-6, 3-6, and 4-6, all of which correspond with the ones that used Jaro-Winkler distance. This fact allows us to reject this measure as an option for this method. Differences of approaches using *pn-measure* are not significant, independently of the hypothesis used. It means that each one of these approaches is as good as the others, allowing us to use all of them, depending of the

Dependent Variable	(I)	(J)	Mean Diff. (I-J)	Std. Error	Sig.	97% Conf. Interval Upper B.	Lower B.
Correct	1	6	-41.555∗	2.969	0.0	-49.945	.33.165
	2	6	-0.334	2.969	1.0	-8.724	8.056
	3	6	-43.966∗	2.969	0.0	-52.356	-35.576
	4	6	-43.276∗	2.969	0.0	-51.666	-34.886
	5	6	-0.815	2.969	0.99	-9.205	7.575
Incorrect	1	6	44.805∗	3.107	0.0	36.027	53.583
	2	6	0.408	3.107	1.0	-8.371	9.186
	3	6	46.014∗	3.107	0.0	37.236	54.792
	4	6	46.463∗	3.107	0.0	37.684	55.241
	5	6	0.326	3.107	1.0	-8.452	9.104
Not Grouped	1	6	-3.250	1.614	0.181	-7.811	1.311
	2	6	-0.074	1.614	1.0	-4.634	4.487
	3	6	-2.048	1.614	0.588	-6.608	2.513
	4	6	-3.186	1.614	0.195	-7.747	1.374
	5	6	0.4887	1.614	0.998	-4.071	5.049

Table 2.7: Dunnet test 2-sided. ∗ The mean difference is significant at the .03 level.

data.

Considering the results obtained in these experiments, it is possible to say that the best proposal for this problem is *Hypothesis b*, although *Hypothesis a* can also be used. For *personal name measure*, it is clear that the *pn-measure* is the best option for comparing names. So, for further experimentation, we propose the use of *Hypothesis b* with *pn-measure* in m_h and m_{co}.

2.6.4 Problem two: Finding the best clustering method.

The experiments in the previous section set up a case definition and a name comparison measure for this methodology. However, the clustering algorithm was selected based on previous approaches without further discussion. At this point, it is our intention to discuss this choice by testing different clustering algorithms to find the one best suiting this problem. The data and validation method for these experiments are the same than on Section 2.6.3.

Four different clustering algorithms are compared: *Complete Linkage*, *Single Linkage*, *Group Average*, and *ROCK* with different values for θ and using 1 as function f as authors recommend [52].

Results of these experiments can be seen on Table 2.8, where results from ROCK have been given on average for space reasons. It can be seen that there are no significant variations between *Complete Linkage* and *Group Average*, while *Single Linkage* and *ROCK* perform clearly worst. Considering the results, either Single Linkage and ROCK are not the best options for this method, as they are clearly outperformed by other options. However *Single Linkage* will be used for comparison in statistical analysis, as it performs better on average than any configuration for ROCK.

Tables 2.9 and 2.10 show the results of an ANOVA test with a post-hoc analysis using the Dunnet Method. The Dunnet test compares data divided into different categories using a control group. This analysis has been performed on three of the algorithms: *Complete Linkage* (C. L.), *Single Linkage* (S. L.), and *Group Average* (G.A.), using the latter as control group. Results show that there are no significant differences between Complete Linkage and Group Average and that both algorithms can be good candidates to use with our methodology. However, for further experiments, *Complete Linkage* is used, as it offers smoother results, with less differences within datasets.

2.6.5 Problem three: Digital Libraries comparison

This methodology can be used to compare some of the most popular Digital Libraries nowadays. This study has been restricted to the three mentioned on Section 2.6.2,

Algorithm	Surname	% Correct	% Inc.	% N. G.	Purity	I.Purity
Complete Linkage	Brown	89.27	9.68	1.05	0.91	0.98
	Delgado	82.01	6.88	11.11	0.93	0.88
	García	88.74	5.77	5.49	0.94	0.94
	Johnson	88.1	7.72	4.18	0.92	0.94
	Sánchez	84.67	11.26	4.05	0.89	0.95
	Smith	84.95	8.66	6.39	0.91	0.93
	Vila	89.81	2.78	7.41	0.97	0.94
	Williams	87.24	7.11	5.65	0.93	0.94
	Average	**86.98**	**7.48**	**5.54**	**0.93**	**0.94**
Single Linkage	Brown	49.68	50	0.32	0.5	1
	Delgado	51.85	47.09	1.06	0.53	0.99
	García	70.33	28.3	1.37	0.72	0.98
	Johnson	58.66	41.34	0	0.59	0.99
	Sánchez	68.92	30.63	0.45	0.69	0.99
	Smith	47.42	52.37	0.21	0.48	1
	Vila	75	19.44	5.56	0.81	0.95
	Williams	48.64	50.73	0.63	0.49	0.99
	Average	**58.79**	**39.99**	**1.22**	**0.6**	**0.99**
Group Average	Brown	89.26	9.67	1.05	0.9	0.98
	Delgado	87.83	2.12	10.05	0.98	0.89
	García	91.48	0	8.52	1	0.91
	Johnson	89.35	5.22	5.43	0.95	0.95
	Sánchez	67.57	0	32.43	1	0.68
	Smith	85.36	13.22	1.42	1	0.85
	Vila	88.89	5.56	5.55	0.94	0.94
	Williams	93.5	5.24	1.26	0.95	0.97
	Average	**86.65**	**5.12**	**8.23**	**0.97**	**0.9**
ROCK $\theta = 0.1$	**Average**	**45.31**	**8.87**	**45.82**	**0.91**	**0.54**
ROCK $\theta = 0.3$	**Average**	**44.74**	**9.12**	**46.14**	**0.91**	**0.54**
ROCK $\theta = 0.5$	**Average**	**45.49**	**7.9**	**46.61**	**0.92**	**0.53**
ROCK $\theta = 0.7$	**Average**	**52.93**	**0.68**	**46.39**	**0.99**	**0.54**

Table 2.8: Comparison of different clustering algorithms

		Sum of Sqr.	df	Mean Square	F	Sig.
Correct	Between Grp.	4165.279	2	2082.64	31.905	0.000
	Within Grp.	1370.788	21	65.276		
	Total	5536.0673	23			
Incorrect	Between Grp.	6072.660	2	3036.33	50.539	0.000
	Within Grp.	1261.661	21	60.08		
	Total	7334.321	23			
NotGrouped	Between Grp.	201.68	2	100.84	2.542	0.103
	Within Grp.	833.067	21	39.67		
	Total	1034.747	23			

Table 2.9: ANOVA For Different Clustering Methods

Dependent			Mean Diff.	Std.		97% Conf. Interval	
Variable	(I)	(J)	(I-J)	Error	Sig.	Upper B.	Lower B.
Correct	C.L.	G.A.	0.20625	4.03967	0.998	-10.3497	10.7622
	S.L.	G.A.	-27.8425^*	4.03967	0.000	-38.3985	-17.2865
Incorrect	C.L.	G.A.	2.35375	3.87553	0.771	-7.7733	12.4808
	S.L.	G.A.	34.85875^*	3.87553	0.000	24.7317	44.98587
NotGrouped	C.L.	G.A.	-2.5475	3.14920	0.638	-10.7766	5.6816
	S.L.	G.A.	-7.01375	3.14920	0.067	-15.2429	1.2154

Table 2.10: Dunnet test 2-sided. Using Group Average as control group. $*$ The mean difference is significant at the .03 level.

as they allow an easy automatic processing, offering a bibliographic reference that can be automatically parsed. These sources are *DBLP*, *CiteSeerX* and *INSPEC*. The groups of surnames used for these experiments are English and Spanish. Details about specific searches and dataset sizes were previously provided in Section 2.6.2.

Table 2.11 shows the results of applying this methodology to the same searches in different DLs. It can be seen that the methodology performs well independently of the source, reaching a correct average over 80%. Additionally, the same test has been performed including and excluding the abstract. This can be done by modifying parameters w_1, w_2, and w_3 when building the term vector (see Section 2.2.2). Results show that, even if including the abstract improves the results, it is possible to obtain a proper classification without it.

Table 2.12 shows the results of a Bonferroni test with $\alpha = 0.03$, for Correct elements. Abbreviations used are CTSX for CiteSeerX and INPS for INSPEC, as well as (a) stands for the use of terms from abstract and (na) means that the terms from the abstract has not been included. From this table, it is possible to see that the mean differences are not significant, so this methodology can be used for any source with or without the abstract.

2.6.6 Problem four: Surname Language comparison

Digital Libraries contain records from authors coming from all around the world, although most of their records are written in English. In this test, it is our intention to check the performance of this system regarding the different origins of the authors. To accomplish this task, three different groups of surnames have been taken: Asian, English and Spanish. For each group, four different surnames have been taken based on their popularity. These surnames have been searched over *CiteSeerX* including the abstract information. Specific details about the datasets can be found on Sec-

DBLP										
	Abstract					No Abstract				
Surname	Corr.	Inc.	N.G.	Pur.	I.Pur.	Corr.	Inc.	N. G.	Pur.	I.Pur.
Brown	–	–	–	–	–	89.3	9.37	1.33	0.91	0.98
Delgado	–	–	–	–	–	96.09	2.61	1.3	0.97	0.99
García	–	–	–	–	–	86.24	11.91	1.85	0.88	0.98
Johnson	–	–	–	–	–	89.72	9.47	0.81	0.91	0.98
Sánchez	–	–	–	–	–	83.99	14.19	1.82	0.86	0.97
Smith	–	–	–	–	–	90.01	8.57	1.42	0.91	0.99
Vila	–	–	–	–	–	97.69	1.54	0.77	0.98	0.99
Williams	–	–	–	–	–	89.2	9.87	0.94	0.9	0.99
Average	–	–	–	–	–	**90.28**	**8.44**	**1.28**	**0.92**	**0.98**
CiteSeerX										
	Abstract					No Abstract				
Surname	Corr.	Inc.	N.G.	Pur.	I.Pur.	Corr.	Inc.	N. G.	Pur.	I.Pur.
Brown	89.27	9.68	1.05	0.91	0.98	87.39	9.67	2.94	0.9	0.97
Delgado	82.01	6.88	11.11	0.93	0.88	51.85	0	48.15	1	0.52
García	88.74	5.77	5.49	0.94	0.94	88.71	6.61	4.68	0.93	0.95
Johnson	88.1	7.72	4.18	0.92	0.94	87.06	7.52	5.43	0.92	0.94
Sánchez	84.68	11.26	4.05	0.89	0.95	84.68	11.26	4.05	0.89	0.95
Smith	84.95	8.66	6.39	0.91	0.93	83.26	10.12	6.61	0.9	0.94
Vila	89.81	2.78	7.41	0.97	0.91	89.81	2.78	7.41	0.97	0.91
Williams	87.24	7.11	5.65	0.93	0.94	86.22	9.39	4.38	0.91	0.94
Average	**86.86**	**7.48**	**5.68**	**0.93**	**0.94**	**82.37**	**7.17**	**10.46**	**0.93**	**0.89**
INSPEC										
	Abstract					No Abstract				
Surname	Corr.	Inc.	N.G.	Pur.	I.Pur.	Corr.	Inc.	N. G.	Pur.	I.Pur.
Brown	92.23	0	7.77	1	0.92	91.44	0	8.56	1	0.91
Delgado	92.37	6.83	0.8	0.93	0.97	89.92	8.06	2.02	0.92	0.97
García	90	0	10	1	0.9	90.2	0	9.8	1	0.9
Johnson	89.64	0	10.36	1	0.89	85.05	9.28	5.67	0.91	0.94
Sánchez	86.67	11.25	2.08	0.89	0.96	85.71	11.76	2.52	0.88	0.95
Smith	89.29	0	10.71	1	0.89	81.96	12.89	5.15	0.87	0.92
Vila	75.72	0	24.28	1	0.75	75.72	0	24.27	1	0.75
Williams	83.05	0	16.95	1	0.83	83.05	0	16.95	1	0.83
Average	**87.37**	**2.26**	**10.37**	**0.91**	**0.89**	**85.38**	**5.25**	**9.37**	**0.89**	**0.9**

Table 2.11: Digital Libraries comparison

	Source	Mean Diff.	Std.		97% Conf. Interval	
(I)	(J)	(I-J)	Error	Sig.	Upper B.	Lower B.
DBLP	CTSX(a)	3.430	3.489	1.0	-7.699	14.559
	CTSX(na)	7.907	3.489	0.297	-3.222	19.037
	INSP(a)	2.909	3.489	1.0	-8.221	14.038
	INSP(na)	4.899	3.489	1.0	-6.231	16.028
CTSX(a)	DBLP	-3.430	3.489	1.0	-14.559	7.699
	CTSX(na)	4.478	3.489	1.0	-6.652	15.607
	INSP(a)	-0.521	3.489	1.0	-11.651	10.608
	INSP(na)	1.469	3.489	1.0	-9.661	12.598
CTSX(na)	DBLP	-7.907	3.489	0.297	-19.037	3.222
	CTSX(a)	-4.478	3.489	1.0	-15.607	6.652
	INSP(a)	-4.999	3.489	1.0	-16.128	6.131
	INSP(na)	-3.009	3.489	1.0	-14.138	8.121
INSP(a)	DBLP	-2.909	3.489	1.0	-14.038	8.221
	CTSX(a)	0.521	3.489	1.0	-10.608	11.651
	CTSX(na)	4.999	3.489	1.0	-6.131	16.128
	INSP(na)	1.990	3.489	1.0	-9.139	13.119
INSP(na)	DBLP	-4.899	3.489	1.0	-16.028	6.231
	CTSX(a)	-1.469	3.489	1.0	-12.598	9.661
	CTSX(na)	3.009	3.489	1.0	-8.121	14.138
	INSP(a)	-1.990	3.489	1.0	-13.119	9.139

Table 2.12: Bonferroni Test for Correct elements, at α significance level 0.03, comparing different digital libraries.

Language	Surname	Correct	Incorrect	Not Grouped	Purity	Inverse Purity
Asian	Kim	94.02	4.12	1.86	0.96	0.97
	Lee	94.55	4.65	0.81	0.95	0.98
	Li	90.54	9.05	0.4	0.91	0.99
	Wang	88.54	11.04	0.41	0.89	0.98
	Average	**91.91**	**7.22**	**0.87**	**0.93**	**0.98**
English	Brown	89.27	9.68	1.05	0.91	0.98
	Johnson	88.1	7.72	4.18	0.92	0.94
	Smith	84.95	8.66	6.39	0.91	0.93
	Williams	87.24	7.11	5.65	0.93	0.94
	Average	**87.39**	**8.29**	**4.32**	**0.92**	**0.95**
Spanish	Delgado	82.01	6.88	11.11	0.93	0.88
	García	88.74	5.77	5.49	0.94	0.94
	Sánchez	84.68	11.26	4.05	0.89	0.95
	Vila	89.81	2.78	7.41	0.97	0.91
	Average	**86.31**	**6.67**	**7.02**	**0.93**	**0.92**

Table 2.13: Surname language comparison

tion 2.6.2. Results from this test can be seen on Table 2.13 and a Bonferroni test comparing the amount of correct elements can be seen on Table 2.14.

The distance measure used in this methodology is based on three different components: *author names*, *terms* and *coauthors*. Considering that most documents in digital libraries are written in English, the *terms* should not be affected by the origin of the author. *Author names* and *coauthors* are compared using *pn-measure*. This measure, defined on Section 2.4.3, is specifically designed to compare personal

	Source	Mean Diff.	Std.		97% Conf. Interval	
(I)	(J)	(I-J)	Error	Sig.	Upper B.	Lower B.
Asian	English	-5.93	5.55	1.0	-25.97	14.11
	Spanish	-0.32	5.55	1.0	-20.36	19.71
English	Asian	5.93	5.55	1.0	-14.11	25.97
	Spanish	5.60	5.55	1.0	-14.43	25.64
Spanish	Asian	0.32	5.55	1.0	-19.71	20.36
	English	-5.60	5.55	1.0	-25.64	14.43

Table 2.14: Bonferroni Test for Correct elements, at α significance level 0.03, comparing the author surname language.

Expert	Correct	Incorrect	Not Grouped	Purity	Inverse Purity
1	93	9	6	0.91	0.97
2	93	9	6	0.91	0.97
3	97	3	7	0.97	0.94
4	83	14	11	0.87	0.95
5	93	6	9	0.94	0.98

Table 2.15: Validation results for the group of experts.

names independently of their language of origin. As it is shown on Table 2.13, the results are quite similar among languages, with a slight outperform of the Asian surnames. However, as it is possible to see on Table 2.14, the mean differences are not significant, so there is no statistical difference among languages.

2.6.7 Problem five: Expert Validation

An expert for this problem could be any person with knowledge about publications and access to a search engine to check if a specific publication belongs to the author under consideration. However, creating a Reference Set can be a very tedious task considering that some of the datasets have up to 5000 publications. To validate the reliability of the expert used in this process, a group of five experts have been called. These experts are researchers with full knowledge about scientific publications, citations, and access to several repositories where they could look up for the specific papers under consideration.

The same experiment, using *Hypothesis b* and *pn-measure* for the search of the surname Vila under CiteSeerX, has been given to the five experts. They have been asked to correct the output of that experiment grouping the elements in the three previous categories: *Correct*, *Incorrect*, *Not Grouped.* The results from the different experts can be seen on Table 2.15.

As it is possible to see on Table 2.15, most experts provide very similar results. This kind of data is objective enough for one expert to provide reliable results.

2.6.8 Discussion

Experiment performed on previous sections have shown that both factor hypotheses provide good results, with a slight outperform of *Hypothesis b*, as can be seen on Section 2.6.3. It means that the title and author information are informative enough to make a good characterization of the publications. However, including coauthoring information, as it is done on hypothesis b, can help to make a more clear distinction, especially in those extreme cases with very similar names and titles.

Regarding divergence measures, experiments have shown that classic name comparison measures are not a good choice when comparing authors with similar surnames, not even the ones recommended for comparing names [19]. As it has been determined on the experiments, using the *Jaro Winkler* measure returns authorities with very low purity and quite high inverse purity. This is because this distance does not discriminate among names, so the obtained authorities group more than one author in the same group. However, the *pn-measure* clearly outperforms in all tests where it was used, returning more accurate clusters.

Hierarchical clustering must be used for this problem, as there is no prior knowledge about how many authors can be returned in a search. This number is determined after the clustering process by a criterion to find the optimal α-cut of a dendrogram. From all the algorithms considered, the best option for this data is *Complete Linkage*. It is important to consider that one of the main problems of hierarchical clustering is scalability. *ROCK* is a hierarchical clustering algorithm that can be used under this method, as it does not require a distance metric. This algorithm has been tested on Table 2.8, performing worse than classic algorithms. From that result, it is possible to conclude that in those cases where scalability is a major factor, it can be used with the drawback of losing accuracy. However, for the problem considered, scalability is not really an issue as the size of the datasets is controlled by the search of a surname.

It is quite interesting to compare source inherent characteristics with the results of this methodology. *DBLP* is a source whose data has been previously name resolved, although their results have quite a lot of inconsistencies, mistakes, and poorly grouped names. This fact is reflected on the results, reaching the best results with peaks close to 98%, even if the abstract information is not available. *CiteSeerX* is the source considered with more inconsistencies and thus, the one with worst results. In this case, including the abstract can really make a difference as there are cases like *Delgado* where we can improve up to 82.01% from 51.85%. This is probably because titles for these authors are not very informative and they do not allow the extraction of the topic. *INSPEC* differs from the others on its wider scope of documents. Whilst CiteSeerX and DBLP are computer sciences oriented, INSPEC covers also other topics like physics or engineering. This reflects in the small amount of incorrect elements, that becomes even smaller if we include the abstract. The bigger amount of topics gives more dissimilarities among them, so it is probably causing the methodology to isolate some elements instead of grouping them properly.

Regarding the language, as all the considered papers are in English, the *terms distance* is not affected by the origin of the author, as can be seen on Section 2.6.6. Additionally, most publishers require a specific name-surname order in their publications, so those authors, like Asians whose names are originally written as surname-name, appear in the same order as the others. The three components of the distance help to disambiguate those cases where there are several authors with the same name. In extreme cases where some of the components are not discriminative enough, it is also possible to adjust the parameters λ_1 and λ_2 for a better fit.

2.7 Conclusions

During this chapter a first approach to the Automatic Authority Control problem based on unsupervised clustering has been made. This approach includes different options for *factors* used in the process as well as the divergence measure that should compare them. To choose among the presented options an extensive experimental study has been performed using data gathered from three different digital libraries. From the results of these experiments it is possible to conclude that from all options proposed for this methodology, it is recommended to use *author, coauthor, title, and abstract* as *factors*. The divergence measure should be *pn-measure* and the clustering algorithm *Complete Linkage*

Validation for these experiments have been performed in a semiautomatic way, meaning that an expert is needed in the process to provide a "correct answer" that would be compared against the output of the system. Despite this, the methodology underlying the system is completely unsupervised, meaning that it does not need an expert to obtain the authorities, just to evaluate their quality. Section 2.6.7 showed that, for this specific problem, finding a good expert is not very difficult. Regardless of those results, providing expert knowledge is still a very time consuming and laborious task.

During this chapter, this expert information in the shape of *Reference Sets* have been reused in different experiments. However, considering the time and effort required by the expert to provide it, getting it for a single use of this system could be considered by a potential user like a waste of resources.

Considering the nature of the expert information, it is possible to see that these so called "correct answers" are way more than a baseline to validate the all the elements in the output. They provide a very complete insight of what elements should be placed in the same clusters, and what it is even more important, which ones should not be together. That information could be very valuable if it is used inside the process and not just in the output, as has been done in this chapter. That led to the idea of introducing expert knowledge inside the clustering process to help it steering the solution into a better one. Under this idea, it could be possible to ask the expert for less information, but use it in a more efficient way by improving the clustering results.

Semisupervised clustering is a technique that introduces external information into the clustering process. By applying it to the process of Automatic Authority Control described in this chapter, it is possible to take advantage of that information to obtain a better clustering solution. This approach will be studied in depth in the following chapter.

Chapter 3

A new semi-supervised clustering approach

In the previous Chapter, the Automatic Authority Control problem has been approached from an unsupervised clustering perspective. Under that premise, we designed a system to approach this problem establishing the elements that could lead to a solution: a proper intermediate representation and a distance measure. Experimental results showed that the model could provide a good solution for the problem but there is room for improvement with the *Not Grouped* elements, as they are elements that should join other clusters, but appear isolated.

The validation of that system has been done using expert knowledge. However, that kind of information could be very useful when used in the clustering process as semi-supervised clustering. Under this idea, a new semi-supervised hierarchical clustering algorithm that could be applied to the Automatic Authority Control problem will be described. This algorithm is based on the *instance level constraints* approach [108], introducing two types of external information into the clustering process: *must-link* and *cannot-link* that could be derived from the expert information. This kind of constraints indicate whether two instances in a clustering process must be in the same cluster, *must-link* or in different clusters, *cannot-link*.

Most semi-supervised clustering algorithms using instance level constraints are based on the idea of modifying the distance measure to fit the constraints [8, 18, 25]. However, that model does not seem suitable for the Automatic Authority Control problem, because as the ad-hoc distance measure developed in Section 2.4 seems to characterize properly the problem. With that idea in mind, it is our intention to use semi-supervision in a different way providing users with the flexibility of selecting a distance measure of their choice. Additionally, the use of hierarchical clustering is desirable in those problems, like Automatic Authority Control, where the number of desired groups is not known in advance.

The previously stated requirements for the semi-supervised algorithm led us to the idea of applying it to find the optimal partition in a dendrogram. Finding this partition automatically is considered one of the biggest problems when using hierarchical clustering. The nature of the instance level constraint makes us think that they are a good model to approach this problem.

Under that idea, the Crisp Hierarchical Semi-Supervised algorithm,*Crisp HSS*, is proposed as a semi-supervised hierarchical clustering algorithm. However, there are some problems where the expert knowledge it is not easy to obtain in a crisp way. Sometimes the expert is not sure or does not have enough information to provide true instance level constraints. For that reason, we propose the use of fuzzy instance level constraints, with the Fuzzy Hierarchical Semi-Supervised algorithm, *Fuzzy HSS*. That model provides the expert with flexibility of giving a degree of belief in the information that she/he is providing.

This chapter is organized as follows: Section 3.1 provides a theoretical formulation of the semisupevised hierarchical clustering problem. In Section 3.2, we define the *Crisp HSS* algorithm. *Fuzzy HSS* algorithm is detailled Section 3.3. Both approaches are compared experimentally in Section 3.5 and some conclusions are provided in Section 3.7.

3.1 Problem formulation

Hierarchical clustering is a kind of clustering algorithm that, instead of creating a direct partition of the dataset, it creates a hierarchy of elements in a binary tree called dendrogram D. Each dendrogram is equivalent to an ultrametric distance matrix $\mathcal{U}$ and generates a fuzzy similarity relation among items [65]. So, it generates a fuzzy similarity matrix. Hierarchical clustering does not generate a partition of the input data but a set of nested partitions according to a level α. These α-levels are just the α-cuts from the previous fuzzy similarity matrix. Then, the problem of finding the partition of the dendrogram is equivalent to the problem of finding the best α-cut of the fuzzy matrix [28]. As this cut will be performed at a node with an associated level α, it will be called optimal α-cut. It is our intention to use instance level constraints to find the optimal cut of the dendrogram D via finding the optimal α-cut of the fuzzy matrix.

Let us define this problem as:

- A set of instances $X = \{x_1, \dots, x_n\}$ that it is known to have some underlying structure, so that it can be divided into an unknown number of disjoint groups.
- The output of most hierarchical clustering technique: a matrix $\mathcal{U}$ of ultrametric distance that, properly normalized, can be transformed into a matrix S representing a fuzzy similarity relation [28]. Thereby, for each node of the dendrogram D, there is an α value associated to it. These α are just the α-cuts

of the fuzzy similarity matrix $\mathcal{S}$. Branches from D correspond to the different partitions of X. These branches will be called *cuts*.

- Two sets of crisp instance level constraints *MUST-LINK* $\mathcal{ML}$ and *CANNOT-LINK* $\mathcal{CL}$ or fuzzy instance level constraints *Fuzzy MUST-LINK* $\mathcal{FML}$ and *Fuzzy CANNOT-LINK* $\mathcal{FCL}$ provided by an expert.

Traditionally, this partition has been obtained by unsupervised methods like the ones used on Chapter 2. Now, it is our goal to find that partition by using *instance level constraints* [108] to find this optimum α-cut. These constraints can be given by an expert who has some knowledge about the data to be clustered. However, it is not necessary to have information about all the data to be clustered, as this expert should not always have information about all entries in the dataset or that information may not be clear.

This problem will be addressed from two different perspectives: Crisp instance level constraints, with the *Crisp HSS* algorithm and Fuzzy instance level constraints with *Fuzzy HSS*.

3.2 Crisp HSS

To approach this problem from a crisp perspective, let us define the Crisp Hierarchical semi-supervised algorithm, *Crisp HSS* [34]. It provides a way of incorporating semi-supervision into hierarchical clustering by the use of crisp instance level constraints.

In the sense of Wagstaff and Cardie, [108], two types of instance level constraints are considered: *must-link* constraints and *cannot-link* constraints. Let us define them as:

- *MUST-LINK* $\mathcal{ML}$: Given x_i and x_j, if there is an $\mathcal{ML}(x_i, x_j)$, it means that x_i and x_j belong to the same cluster and must be grouped together.
- *CANNOT-LINK* $\mathcal{CL}$: Given t_i and t_j, if there is an $\mathcal{CL}(x_i, x_j)$, it means that x_i and x_j do not belong to the same cluster and cannot be grouped together.

It is possible to represent constraints among the different elements to be clustered using a matrix A. This matrix has three possible values:

$$A[x_i, x_j] = \begin{cases} 1 & \text{if there is a } \mathcal{ML}(x_i, x_j). \\ 0 & \text{if there is a } \mathcal{CL}(x_i, x_j). \\ -1 & \text{otherwise.} \end{cases} \tag{3.1}$$

The matrix A of external information can be built with the information offered by the expert. It is important to remark that there is no need for the expert to give information about all pairs of t. Considering that must-link constraints are transitive, it is necessary to obtain the transitive closure on A to keep consistency in the data.

In the same way, for each α-level in the dendrogram, it is also possible to create a matrix B from the matrix U representing the joined elements at each time. So, for each α-level, we will have a matrix B_α with the information of the elements at that specific α-level. Each $B_\alpha(x_i, x_j)$ will have two possible values:

- $B_\alpha(x_i, x_j) = 1$ if the elements x_i and x_j are in the same cluster
- $B_\alpha(x_i, x_j) = 0$ if the elements x_i and x_j are not in the same cluster

Matrix A can be compared with the different matrices B_α according to (3.2).

$$r(A, B_\alpha) = \frac{\sum_{i=1,j=1}^{n} r_A(A(x_i, x_j), B_\alpha(x_i, x_j))}{n_{constraints}} \tag{3.2}$$

Where n is the number of elements to be clustered, $n_{constraints}$ is the number of constraints offered by the expert and r_A can be calculated as (3.3).

$$r_A(A(x_i,x_j),B_\alpha(x_i,x_j)) = \begin{cases} u_m, & \text{if } A(x_i,x_j) = B_\alpha(x_i,x_j) \text{ and } ; (x_i,x_j) \in \mathcal{ML} \\ u_c, & \text{if } A(x_i,x_j) = B_\alpha(x_i,x_j) \text{ and } ; (x_i,x_j) \in \mathcal{CL} \\ u_{nm}, & \text{if } A(x_i,x_j) \neq B_\alpha(x_i,x_j) \text{ and } ; (x_i,x_j) \in \mathcal{ML} \\ u_{nc}, & \text{if } A(x_i,x_j) \neq B_\alpha(x_i,x_j) \text{ and } ; (x_i,x_j) \in \mathcal{CL} \\ 0, & \text{Otherwise.} \end{cases} \tag{3.3}$$

Weights u_m and u_c represent the importance that satisfying a constraint has in the problem and u_{nm}, u_{nc} represent the penalty of violate it.

Using these expressions, the optimum α-cut can be obtained by maximizing the expression (3.4)

$$\max_{\alpha \in B} r(A, B_\alpha) \tag{3.4}$$

The value α by which the matrix B_α that maximizes (3.4) has been obtained is the optimum α-cut of the dendrogram. By cutting the tree at that level, the optimum partition of the data will be obtained.

This process can be summarized algorithmically as Algorithm 2.

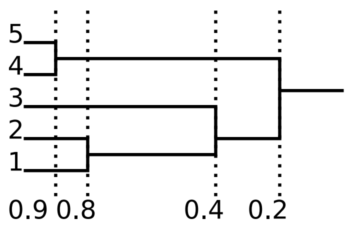
Use example

Let us illustrate this method with an example. Figure 3.1 shows an example of a dendrogram and its equivalent opposite of the ultrametric matrix returned by a clustering process over 5 elements. An expert with knowledge about the problem has given the following crisp constraints:

- $\mathcal{ML}(1,2)$

Algorithm 2 Crisp HSS

```
for each $\alpha \in D$ do
    Define the partition $B_\alpha$
    $r_\alpha = 0$
    for each $A[x_i, x_j] \in A$ where $x_i \neq x_j$ and $A[x_i, x_j] \neq -1$ do
        if $A(x_i, x_j) = B_\alpha[i, j]$ then
            if $(x_i, x_j) \in \mathcal{ML}$ then
                $r_\alpha + = u_m$
            else if $(x_i, x_j) \in \mathcal{CL}$ then
                $r_\alpha + = u_c$
            end if
        else if $A(x_i, x_j) \neq B_\alpha[i, j]$ then
            if $(x_i, x_j) \in \mathcal{ML}$ then
                $r_\alpha + = u_{nm}$
            else if $(x_i, x_j) \in \mathcal{CL}$ then
                $r_\alpha + = u_{nc}$
            end if
        end if
    end for
end for
Select the partition $\alpha$ for which $r_\alpha$ is maximum.
```



$$\begin{pmatrix} 1.0 & 0.8 & 0.4 & 0.2 & 0.2 \\ 0.8 & 1.0 & 0.4 & 0.2 & 0.2 \\ 0.4 & 0.4 & 1.0 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 1.0 & 0.9 \\ 0.2 & 0.2 & 0.2 & 0.9 & 1.0 \end{pmatrix}$$

Figure 3.1: Example of dendrogram and its equivalent opposite of the ultrametric matrix

- $\mathcal{ML}(4,5)$
- $\mathcal{CL}(2,3)$
- $\mathcal{CL}(1,5)$

From these constraints, we build the matrix A according to (3.1). The ultrametric matrix of Figure 3.1, provides four different α-cuts: $\alpha = \{0.9, 0.8, 0.4, 0.2\}$. The dotted lines in the dendrogram show the clusters formed at each level. Matrices B_α can be deduced from that Figure. Weight in formula (3.3) are set to $u_m = u_c = 1$ and $u_{nm} = u_{nc} = 0$, so we ignore those constraints that are violated.

So, for each α:

$$r(A, B_{0.9}) = \begin{cases} A[1,2] \neq B_{0.9}[1,2]; & 0 \\ A[4,5] = B_{0.9}[4,5]; & 1 \\ A[2,3] = B_{0.9}[2,3]; & 1 \\ A[1,5] = B_{0.9}[1,5]; & 1 \end{cases} = 3/4 = 0.75 \tag{3.5}$$

$$r(A, B_{0.8}) = \begin{cases} A[1,2] = B_{0.8}[1,2]; & 1 \\ A[4,5] = B_{0.8}[4,5]; & 1 \\ A[2,3] = B_{0.8}[2,3]; & 1 \\ A[1,5] = B_{0.8}[1,5]; & 1 \end{cases} = 4/4 = 1 \tag{3.6}$$

$$r(A, B_{0.4}) = \begin{cases} A[1,2] = B_{0.4}[1,2]; & 1 \\ A[4,5] = B_{0.4}[4,5]; & 1 \\ A[2,3] \neq B_{0.4}[2,3]; & 0 \\ A[1,5] = B_{0.4}[1,5]; & 1 \end{cases} = 3/4 = 0.75 \tag{3.7}$$

$$r(A, B_{0.2}) = \begin{cases} A[1,2] = B_{0.2}[1,2]; & 1 \\ A[4,5] = B_{0.2}[4,5]; & 1 \\ A[2,3] \neq B_{0.2}[2,3]; & 0 \\ A[1,5] \neq B_{0.2}[1,5]; & 0 \end{cases} = 2/4 = 0.5 \tag{3.8}$$

Now, using (3.4) we get:

$$\max_{\alpha \in [0,1]} r(A, B_\alpha) = \max\{0.75, 1, 0.75, 0.5\} = 1 \rightarrow \alpha = 0.8 \tag{3.9}$$

So, the optimum α-cut is found at $\alpha = 0.8$ obtaining three clusters.

3.3 Fuzzy HSS

The use of crisp information has some limitations, specifically, in those problems where the expert is not able to provide a certain answer. This can happen either because it is not possible to find an expert with deep knowledge of the field or because the classes or categories are not clearly defined. In all these situations, it could be possible to find someone who, instead of giving true information about whether two elements are or not in the same cluster, is able to provide some "degrees of belief" about them. Considering that fuzzy logic is a very good way to model uncertainty, instance level constraints can be provided on a fuzzy way as a value in the interval [0,1]. In an attempt to overcome the limitations of crisp constraints, in this Section, we introduce *Fuzzy HSS*, a Fuzzy Hierarchical Semi-supervised algorithm that uses fuzzy instance level constraints that allow the representation of uncertainty based on the ideas of *Crisp HSS*.

Classic crisp instance level constraints are expressed in an absence/presence way, meaning that the constraint can only have two values, one if there is a constraint of any kind between two elements or zero if there is none. It is our intention to introduce a mechanism to describe uncertainty in fuzzy instance level constraints. For that reason, the classic model is softened allowing degrees of belief for constraints between two elements. With these ideas, the previous definition can be redefined as:

- *FUZZY MUST-LINK, $\mathcal{FML}$*: Given x_i and x_j, if a $\mathcal{FML}(x_i, x_j) = \beta$ exists, then x_i and x_j belong to the same cluster with a degree $\beta \in [0, 1]$; $1 \leq i, j \leq n$.
- *FUZZY CANNOT-LINK, $\mathcal{FCL}$*: Given x_i and x_j, if a $\mathcal{FCL}(x_i, x_j)$ exists, then x_i and x_j are not in the same cluster with a degree $\gamma \in [0, 1]$; $1 \leq i, j \leq n$.

Using this model, an expert can now tell us that two elements are in the same cluster (*must-link*) with a degree of belief in the interval [0,1] according to her/his certainty on the provided information. The same idea can be applied to *cannot-link* constraints.

It is necessary to define a proper representation for the constraints, allowing the comparison with the dendrogram. As the dendrogram is equivalent to a matrix $\mathcal{U}$ of size $n \times n$, where n is the number of instances to be grouped, it seems reasonable to represent constraints using matrices. On the *Crisp HSS* approach, we use matrix A (3.1) to represent the constraints. Under that model, with just one matrix it is possible to define both *must-link* and *cannot-link* constraints. For this fuzzy approach, as each constraint has an associated degree, it is not possible to represent them using just one matrix, because there is no way to distinguish the kind of constraint to be expressed. For that matter, let us define two matrices A_m (for *must-link*) and A_c (for *cannot-link* as:

$$A_m[i, j] = \begin{cases} \beta, & \text{if } \exists \mathcal{FML}(x_i, x_j) \\ UND, & \text{if } \nexists \mathcal{FML}(x_i, x_j) \end{cases} \tag{3.10}$$

$$A_c[i,j] = \begin{cases} \gamma, & \text{if } \exists \mathcal{FCL}(x_i, x_j) \\ UND, & \text{if } \nexists \mathcal{FCL}(x_i, x_j) \end{cases} \tag{3.11}$$

Where β is the degree of belief of the fuzzy must-link constraint, γ is the degree of belief of the fuzzy cannot-link constraint, and *UND* is a parameter used to indicate that there are not any constraints between instances x_i and x_j. Generally, *UND* takes the value -1. Additionally, it is very important to consider that $\nexists (x_i, x_j) \in X | \mathcal{FML}(x_i, x_j) = \beta$ and $\mathcal{FCL}(x_i, x_j) = \gamma$. This restriction indicates that the constraints should not contradict themselves, meaning that there cannot be a *must-link* constraint and a *cannot-link* constraint for the same pair of elements.

As described on Section 3.2, let us remember that for each α-cut of the dendrogram it is also possible to create a matrix B from the matrix U representing the elements that have been clustered on each step. So, for each α-cut there is a matrix B_α with the elements joined at that specific α-level. Each $B_\alpha(x_i, x_j)$ will have two possible values:

- $B_\alpha(x_i, x_j) = 1$ if elements x_i and x_j are in the same group
- $B_\alpha(x_i, x_j) = 0$ if elements x_i and x_j are not in the same group

Matrices A_m and A_c can be compared with the different matrices A_α according to (3.12).

$$fr(A_m(x_i, x_j), A_c(x_i, x_j), B_\alpha(x_i, x_j)) = \begin{cases} v_m, & \text{if } A_m(x_i, x_j) \geq \alpha \text{ and } B_\alpha(x_i, x_j) = 1; \\ w_m, & \text{if } A_m(x_i, x_j) \geq \alpha \text{ and } B_\alpha(x_i, x_j) = 0; \\ v_c, & \text{if } A_c(x_i, x_j) \geq \alpha \text{ and } B_\alpha(x_i, x_j) = 0; \\ w_m, & \text{if } A_c(x_i, x_j) \geq \alpha \text{ and } B_\alpha(x_i, x_j) = 1; \\ 0, & \text{otherwise .} \end{cases} \tag{3.12}$$

Where α is the fuzzy value from the B_α, v_m and v_c are used to adjust the weight that reinforcing the constraint (the constraint matches the partition) has in the expression and w_m and w_c is the penalty of contradict the constraint.

Using the previously defined expressions, the optimum α-cut can be calculated by maximizing the expression in (3.13)

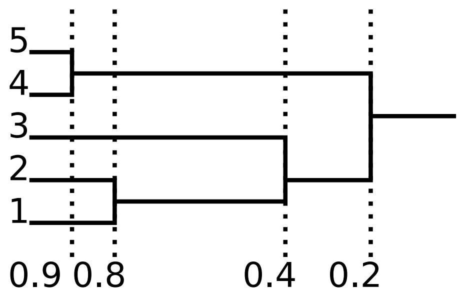
$$FR = \max_{\alpha \in [0,1]} \{ \sum_{i,j}^{n} fr(A_m(x_i, x_j), A_c(x_i, x_j), B_\alpha(x_i, x_j))) \} \tag{3.13}$$

The α from matrix B_α for which (3.13) have its maximum is the optimum α-cut of the dendrogram. Cutting the tree at that level will give us the optimum partition of the dataset.

All this process can be summarized algorithmically as Algorithm 3.

Algorithm 3 Fuzzy HSS

```
for each $\alpha \in D$ do
    Define the partition $B_\alpha$
    $fr_\alpha = 0$
    for each $\mathcal{FML}(x_i, x_j) \in A_m$ do
        if $A_m(x_i, x_j) \geq \alpha$ and $B_\alpha[i, j] = 1$ then
            $fr_\alpha += v_m$
        else if $A_m(x_i, x_j) \geq \alpha$ and $B_\alpha[i, j] = 0$ then
            $fr_\alpha += w_m$
        end if
    end for
    for Each $\mathcal{FCL}(x_i, x_j) \in A_c$ do
        if $A_c(x_i, x_j) \geq \alpha$ and $B_\alpha[i, j] = 1$ then
            $fr_\alpha += v_c$
        else if $A_c(x_i, x_j) \geq \alpha$ and $B_\alpha[i, j] = 0$ then
            $fr_\alpha += w_c$
        end if
    end for
end for
Select the partition $\alpha$ for which $tr_\alpha$ is maximum.
```



$$\begin{pmatrix} 1.0 & 0.8 & 0.4 & 0.2 & 0.2 \\ 0.8 & 1.0 & 0.4 & 0.2 & 0.2 \\ 0.4 & 0.4 & 1.0 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 1.0 & 0.9 \\ 0.2 & 0.2 & 0.2 & 0.9 & 1.0 \end{pmatrix}$$

Figure 3.2: Example of dendrogram and its equivalent opposite of the ultrametric matrix

Use example

Let us illustrate this method with an example. Figure 3.2 recalls the example dendrogram showed for *Crisp HSS* and its equivalent opposite of the ultrametric matrix returned by a clustering process over 5 elements. An expert with knowledge about the problem has given the following constraints:

- $A_m(1, 2) = 0.8$
- $A_m(4, 5) = 1.0$
- $A_c(2, 3) = 0.9$
- $A_c(1, 5) = 0.6$

From the ultrametric matrix we have four different α-cuts: $\alpha = \{0.9, 0.8, 0.4, 0.2\}$. For each one of them we are going to evaluate $fr(A_m, A_c, B_\alpha)$, with values of $v_m = v_c = 1$, $w_m = w_c = -0.5$.

So, for each α:

$$fr(A_m, A_c, B_{0.9}) = \begin{cases} A_m(1,2) = 0.8 < 0.9 \text{ and } B_{0.9}(1,2) = 0; & 0 \\ A_m(4,5) = 1.0 \geq 0.9 \text{ and } B_{0.9}(4,5) = 1; & 1 \\ A_c(2,3) = 0.9 \geq 0.9 \text{ and } B_{0.9}(2,3) = 0; & 1 \\ A_c(1,5) = 0.6 < 0.9 \text{ and } B_{0.9}(1,5) = 0; & 0 \end{cases} = 2 \tag{3.14}$$

$$fr(A_m, A_c, B_{0.8}) = \begin{cases} A_m(1,2) = 0.8 \geq 0.8 \text{ and } B_{0.8}(1,2) = 1; & 1 \\ A_m(4,5) = 1.0 \geq 0.8 \text{ and } B_{0.8}(4,5) = 1; & 1 \\ A_c(2,3) = 0.9 \geq 0.8 \text{ and } B_{0.8}(2,3) = 0; & 1 \\ A_c(1,5) = 0.6 < 0.8 \text{ and } B_{0.8}(1,5) = 0; & 0 \end{cases} = 3 \tag{3.15}$$

$$fr(A_m, A_c, B_{0.4}) = \begin{cases} A_m(1,2) = 0.8 \geq 0.4 \text{ and } B_{0.4}(1,2) = 1; & 1 \\ A_m(4,5) = 1.0 \geq 0.4 \text{ and } B_{0.4}(4,5) = 1; & 1 \\ A_c(2,3) = 0.9 \geq 0.4 \text{ and } B_{0.4}(2,3) = 1; & -0.5 \\ A_c(1,5) = 0.6 \geq 0.4 \text{ and } B_{0.4}(1,5) = 0; & 0 \end{cases} = 1.5 \tag{3.16}$$

$$fr(A_m, A_c, B_{0.2}) = \begin{cases} A_m(1,2) = 0.8 \geq 0.2 \text{ and } B_{0.2}(1,2) = 1; & 1 \\ A_m(4,5) = 1.0 \geq 0.2 \text{ and } B_{0.2}(4,5) = 1; & 1 \\ A_c(2,3) = 0.9 \geq 0.2 \text{ and } B_{0.2}(2,3) = 1; & -0.5 \\ A_c(1,5) = 0.6 \geq 0.2 \text{ and } B_{0.2}(1,5) = 1; & -0.5 \end{cases} = 1 \tag{3.17}$$

Now, using (3.13) we get:

$$FR = \max_{\alpha \in [0,1]} fr(A_m, A_c, B_\alpha) = \max\{2, 3, 1.5, 1\} = 3 \rightarrow \alpha = 0.8 \tag{3.18}$$

So, the optimum α-cut is found at $\alpha = 0.8$ obtaining three clusters.

3.4 Using fuzzy entropy to determine the optimal number of fuzzy constraints.

It is our intention to determine how much information is needed in the process and trying to find out a mechanism to indicate the expert how much information should be included in the algorithm. Typically, on semi-supervised processes, the results tend to stabilize after a certain number of constraints. However the amount of information needed in this process is still a question to be answered. We address this problem by proposing a method to determine how much information is needed by means of the fuzzy entropy concept.

The simplest approach to this problem could be to use as much information as the expert can provide. It is appropriate for those cases when the expert provide only a small amount of information. However, when the expert can provide a lot of constraints, this could not be the best approach, as they can increase computation time unnecessarily or even, in some cases, induce over-learning in the process. Some approaches to this problem in the crisp context include: [22, 47, 27, 99]

To solve this problem, we propose the use of *fuzzy entropy*. This concept, first described by De Luca and Termini [76], can be defined as a measure of the degree of fuzziness of a fuzzy set. That work established several properties that should be fulfilled by a measure of fuzzy entropy. Based on those properties, there are several proposals in the literature to measure fuzzy entropy of a fuzzy set X.

- *De Luca and Termini* [76], based on Shannon entropy:

$$D_l(f) \equiv H(f) + H(f') \tag{3.19}$$

  where H is the Shannon entropy [96] and $f'(x) \equiv 1 - f(x)$.

- *Quadratic Entropy*[40]

$$D_q = x(1-x) \tag{3.20}$$

- *Battacharyya Entropy*[40]:

$$D_b = \sqrt{x(1-x)} \tag{3.21}$$

- *Pal&Pal Entropy* [85]:

$$D_p(X) = \frac{1}{n(\sqrt{e}-1)} \sum_i [S_n(\mu_X(x_i)) - 1] \tag{3.22}$$

  with

$$S_n(\mu_X(x_i)) = \mu_X(x_i)e^{1-\mu_X(x_i)} + (1-\mu_X(x_i))e^{\mu_A(x_i)} \tag{3.23}$$

- *Rogas Entropy [97]*

$$D(X) = \frac{X \cap X^C}{X \cup X^C} \tag{3.24}$$

- *Simple entropy*, proposed by Knopfmacher [67] as the easiest function that fulfills the fuzzy entropy properties

$$D_t(f(x)) = \sum_{i=1}^{n} \frac{d_t(f(x_i))}{n}; \tag{3.25}$$

$$d_t(f(x_i)) = \begin{cases} f(x_i), & \text{for } f(x_i) \in [0, \frac{1}{2}] \\ (1 - f(x_i)), & \text{for } f(x_i) \in [\frac{1}{2}, 1]. \end{cases} \tag{3.26}$$

All these entropy measures fulfil the same properties [76], so they are considered equivalent. To determine which entropy measure should be used in this process, we have compared five different measures available in the literature: *Simple* [67], *Quadratic* [40], *Battacharyya*[40], *Pal & Pal*[85], and *Rogas*[97]. De Luca and Termini's [76] entropy has not been considered, as it uses logarithms whose properties for values like 0 or 1 are not the most appropriate for instance level constraints.

For this study, we have used five well-known datasets from the UCI repository [2]: *Glass*, *Iris*, *Wine*, *PenDigits* and *Vowel.* Specific details about each dataset will be given in Section 3.5. We have chosen these labeled datasets with the intention of using their class labels to simulate the role of an expert.

Instance level constraints have been generated according to the class labels of the datasets. For that purpose, a *fuzzy must-link* constraint has been created for every pair of instances belonging to the same class. The degree of belief on that constraint, β, has been assigned randomly. In the same way, for every pair of elements that do not belong to the same class, there is a *fuzzy cannot-link* constraint. Again, the degree of belief on that constraint γ has been assigned randomly. These random degrees of belief have been used to simulate an expert that is giving true information but with not so much knowledge about the problem. For us, this is the worst case scenario, as we consider that the information given by the expert is always correct.

For each $\mathcal{FML}$ and $\mathcal{FCL}$ of every dataset, we have taken a random small partition of the data and iteratively increased in small portions. As a result, we have

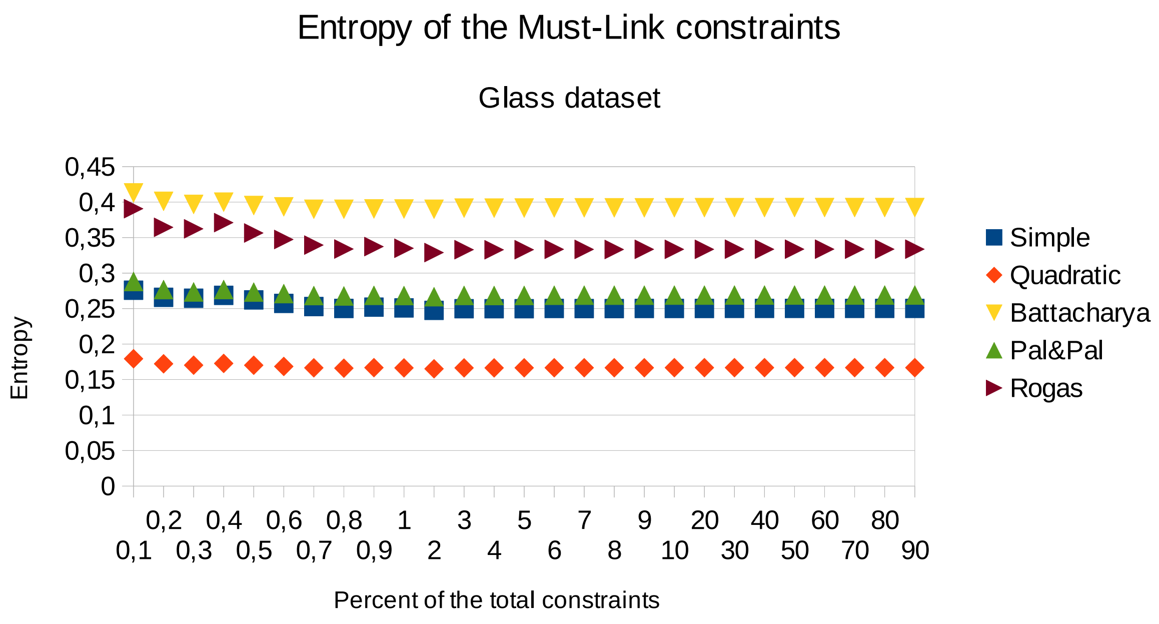
		Sum Sq	Df	F value	Pr (>F)
	Problem	0.005462	4	6.5534	3.604e-05
Must	Entropy	0.000862	4	1.0343	0.3887
Link	Problem:Entropy	0.001920	16	0.5760	0.9028
	Residuals	0.130235	625		
	Problem	0.003318	4	11.5396	4.805e-09
Cannot	Entropy	0.000096	4	0.3328	0.8560
Link	Problem:Entropy	0.000780	16	0.6778	0.8173
	Residuals	0.044927	625		

Table 3.1: Two way ANOVA with problem and entropy as factors for the must-link and cannot-link constraints entropy

several nested subsets that contain from 0.1% to 90% of the original constraints. We have measured the entropy of all of them in five different ways: *Simple*, *Quadratic*, *Battacharyya*, *Pal & Pal*, and *Rogas*.

Figures 3.3, 3.4, 3.5, 3.6 and 3.7 show these results. Considering that the constraints are randomly included on the subsets, we have used the average of five executions. The Wine dataset has fewer must-link constraints than the others, so we have used only those subsets that contain constraints. From the graphs, it is possible to see that for each problem and type of constraints, all curves in the graph have a very similar shape. It led us to the idea that all entropies behave similarly, so the one selected would not affect the results. A more formal study of this effect can be found on Table 3.1, where a two-way ANOVA test has been performed for each kind of constraints on the variations between the entropy of one subset and the next one. As entropies are similar on shape but start on different values, we consider that the best way to measure the "shape" of the curve is to obtain the differences between the entropy of one subset and the next one.

The *Problem* and *Entropy* measures have been used as factors on the ANOVA. As can be seen on Table 3.1, the interaction term, *Problem:Entropy*, is not statis-



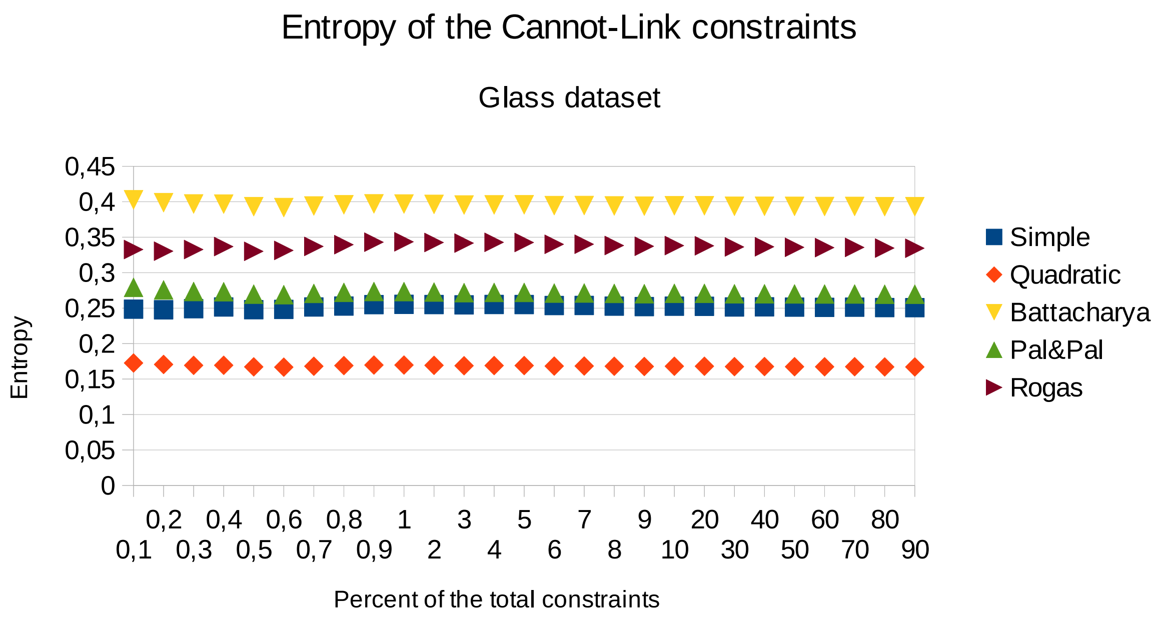
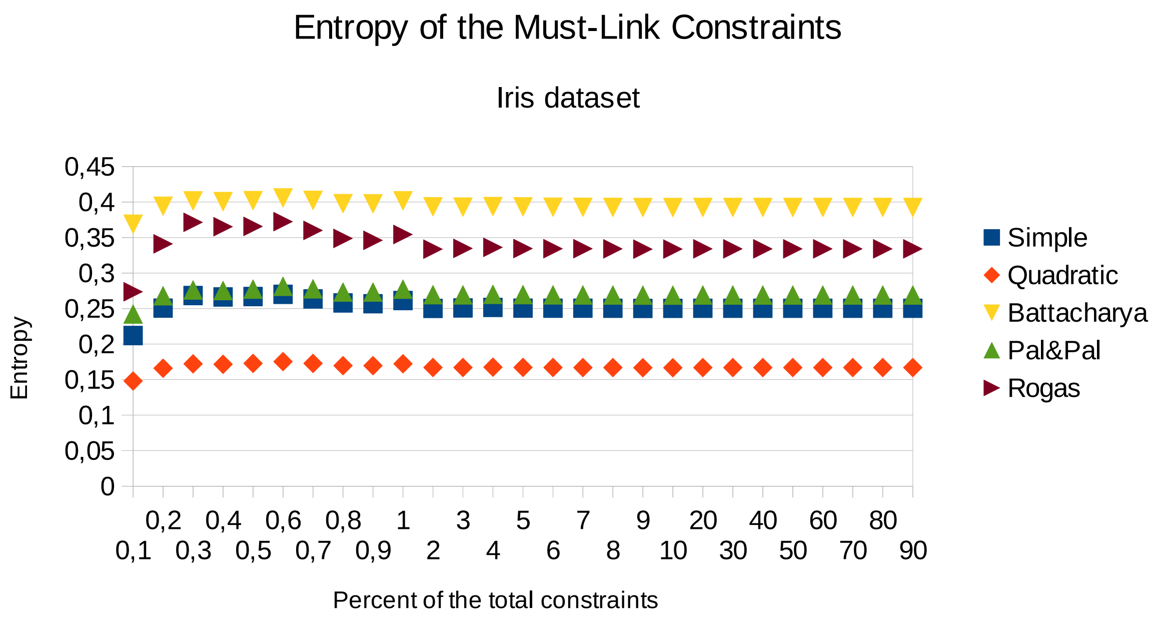


Figure 3.3: Results from measuring different entropies of the must-link/cannot-link with different subsets of constraints for the Glass dataset.



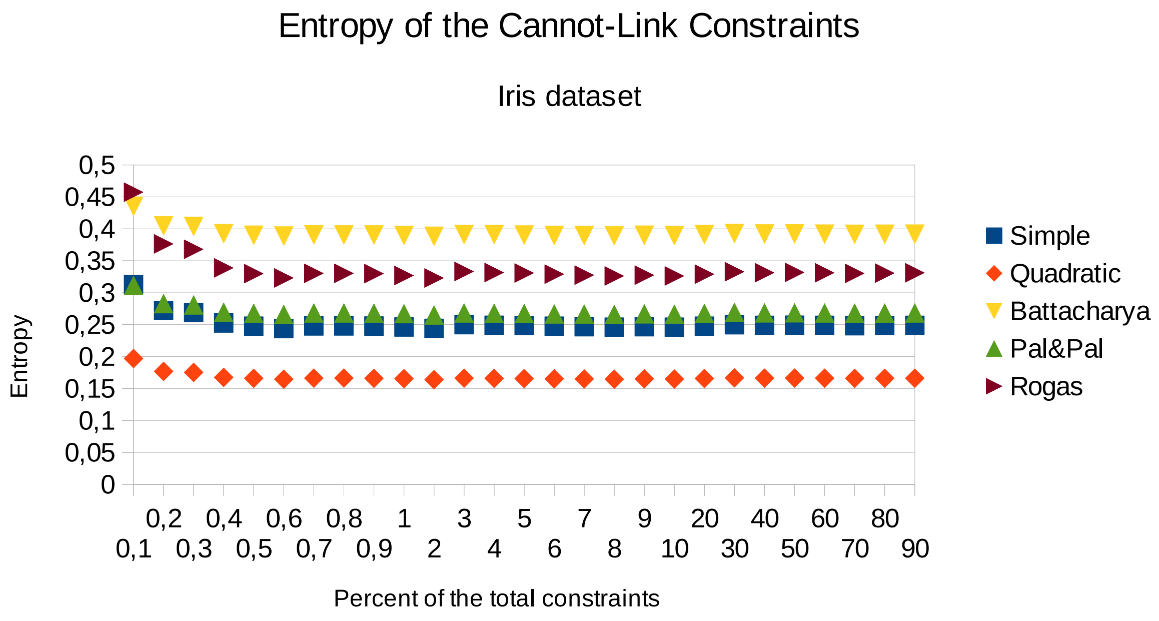
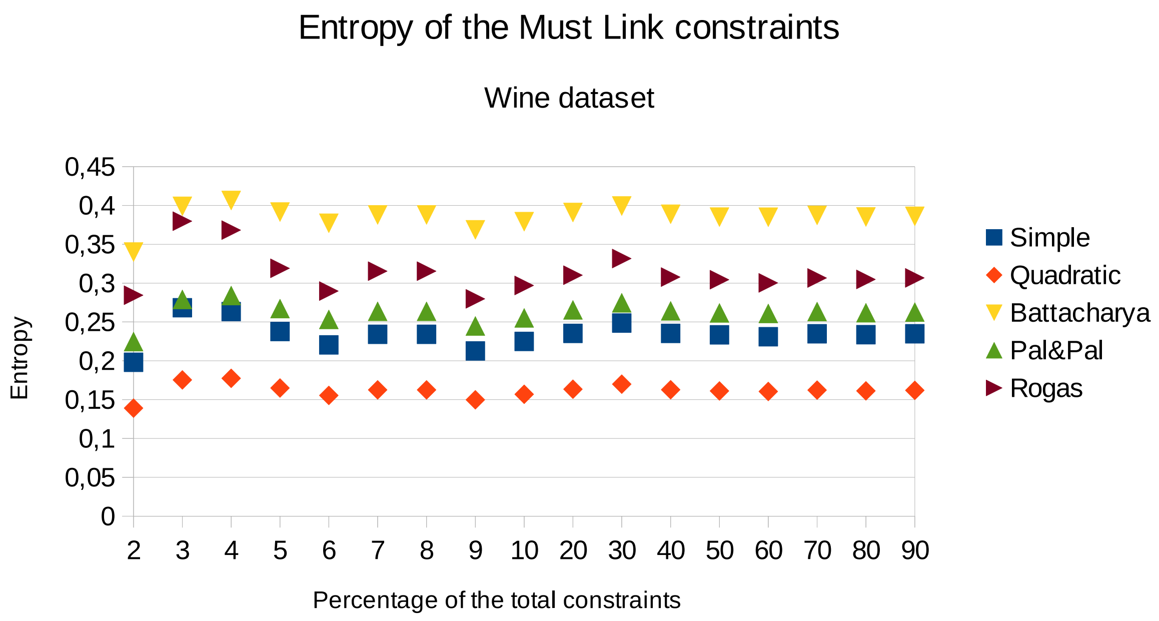


Figure 3.4: Results from measuring different entropies of the must-link/cannot-link with different subsets of constraints for the Iris dataset.



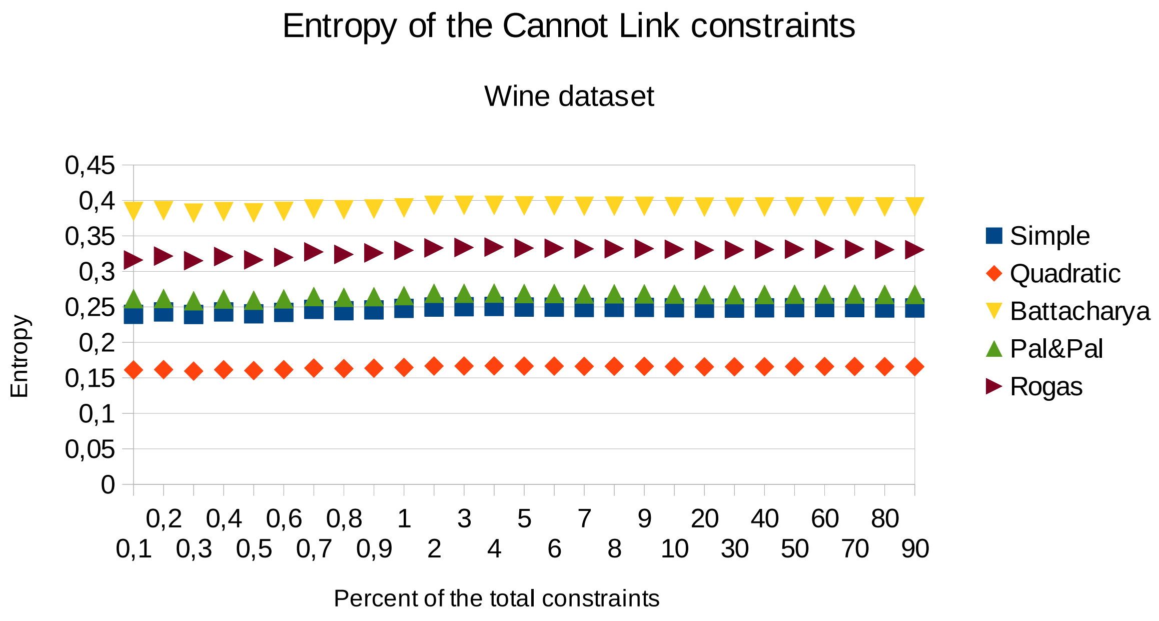
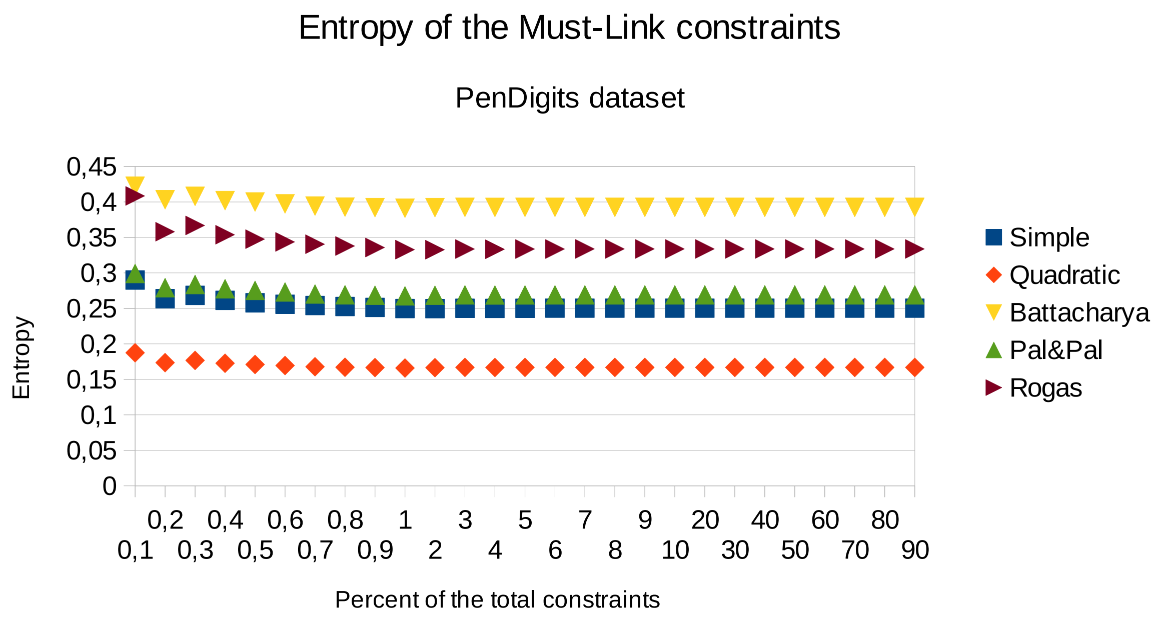


Figure 3.5: Results from measuring different entropies of the must-link/cannot-link with different subsets of constraints for the Wine dataset.



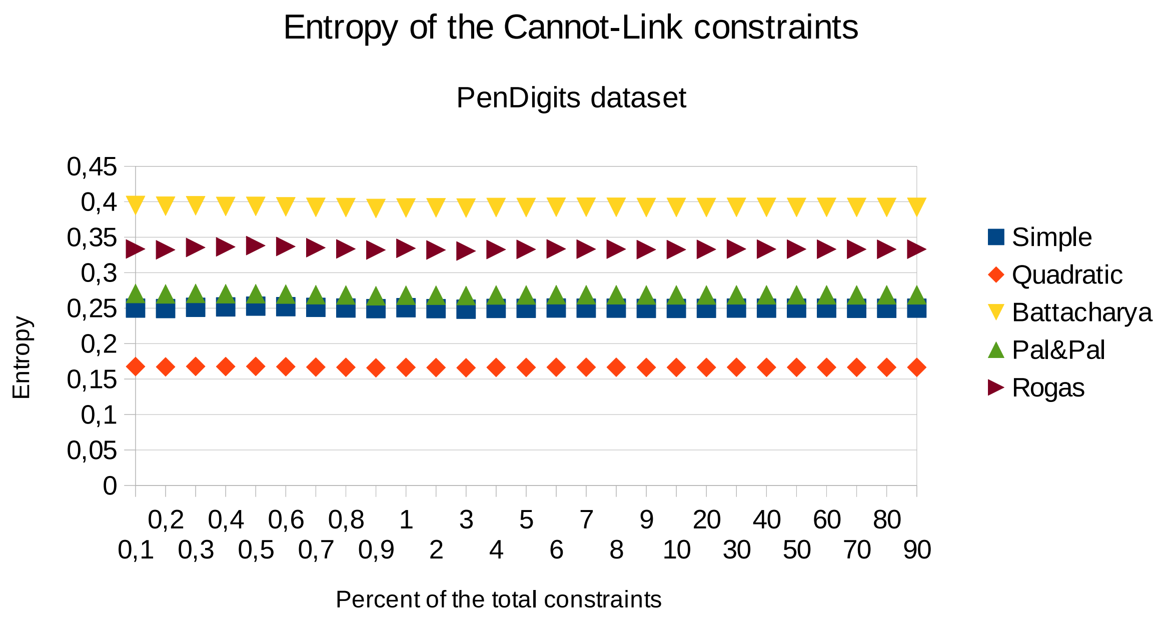
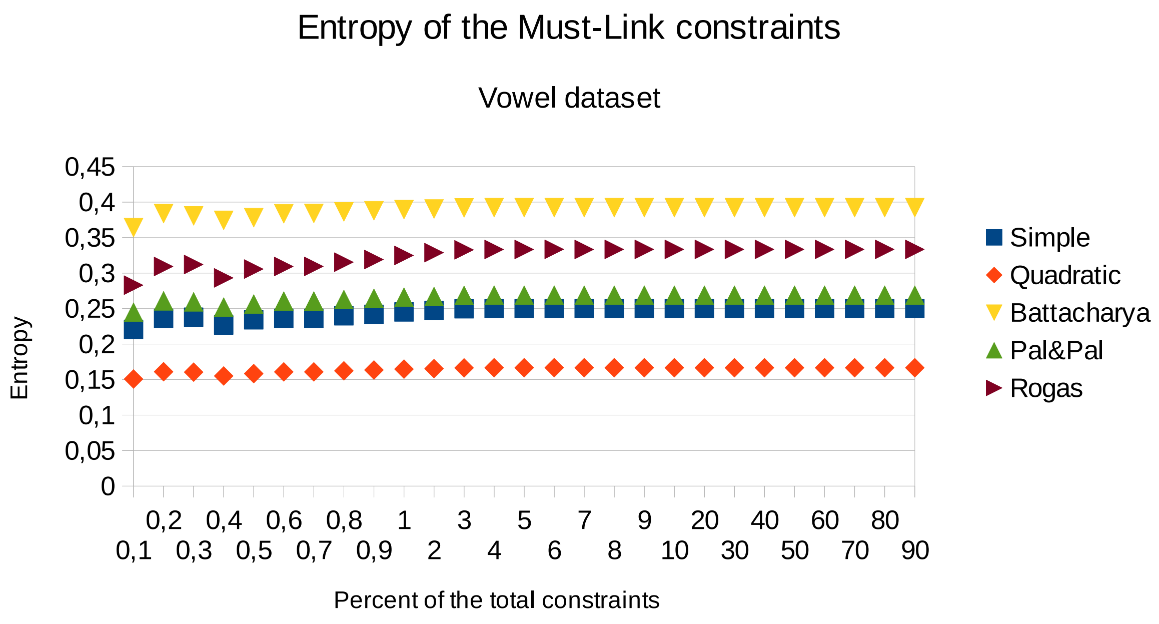


Figure 3.6: Results from measuring different entropies of the must-link/cannot-link with different subsets of constraints for the Pendigits dataset.



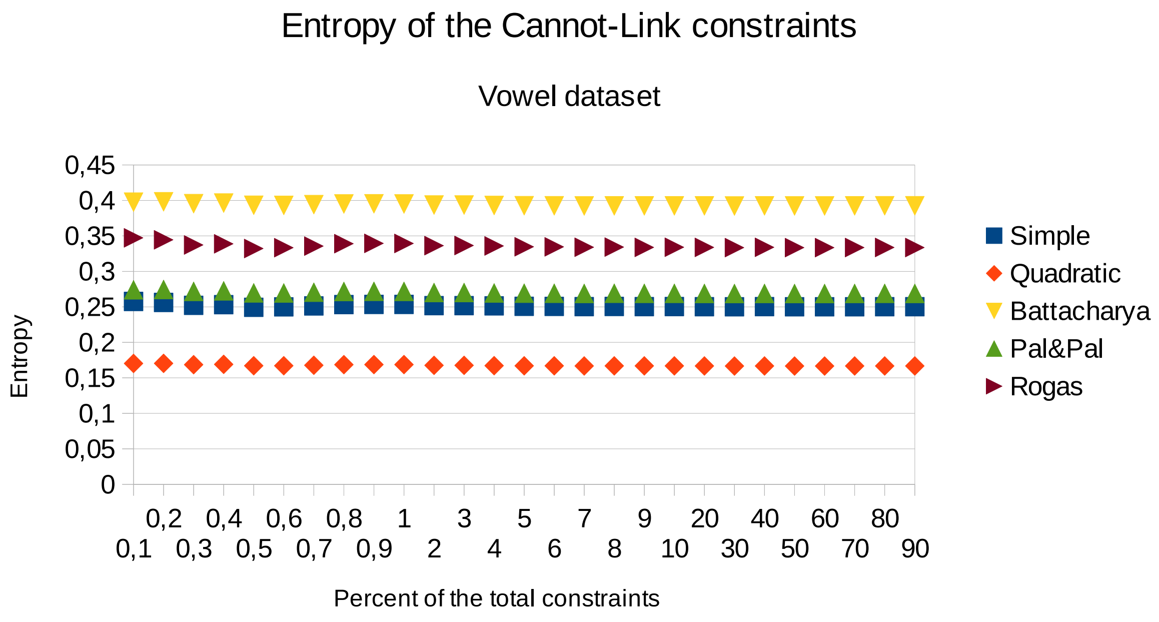


Figure 3.7: Results from measuring different entropies of the must-link/cannot-link with different subsets of constraints for the Vowel dataset.

tically significant, which means that the factors do not affect each other, and so, they must be considered separately. Factor *Problem* is significant for both kind of constraints, but factor *Entropy* is not in any case. It means that the null hypothesis cannot be rejected for the factor *Entropy*, i.e., all population means are equal. Considering that, we can say that all entropies have the same behavior. For that reason, selecting one or another should not change the results of the method. *Simple Entropy* [67] is suggested to be used, as it is very easy to calculate. Additionally, considering that the problem factor is statistically significant, we cannot say that the entropy is the same for every problem. This is confirmed on the graphs, for every problem, all entropies have the same shape, but they differ among problems. So, once we have decided which entropy we will use, it should be studied for the problem under consideration.

After the previous experiments where it has been tested that the entropy of fuzzy instance level constraints tends to stabilize after a certain amount of constraints, it is our intention to use it to determine how much information we need for this semi-supervised clustering method. To find that point of stability, all constraints provided by the expert are considered in the shape of matrices A_m and A_c. To determine the amount of external information, the next steps should be followed:

1. For each matrix:
2. Remove most constraints from matrix E, but a random small percentage of the original elements
3. Calculate entropy for the remaining elements
    (a) Add another random small percentage of the original elements to the matrix
    (b) Calculate entropy for the matrix

(c) If the entropy of the matrix differs from the previous one in less than a threshold θ, go to a). Otherwise, stop the process and this percentage is the proper amount of constraints for that matrix.

How to determine threshold θ depends of the constraints provided by the expert and must be determined experimentally. However, a reasonable value for this parameter for most problems would be between 0.001 and 0.01. After these steps, two reduced A_m and A_c matrices are obtained that can be applied in *Fuzzy HSS*.

3.5 Experimental comparison of Crisp HSS and Fuzzy HSS

Even if *Crisp HSS* and *Fuzzy HSS* methods have been designed with the requirements of the Automatic Authority Control problem in mind, these methods can be applied in a wider range of problems. To properly test and compare the performance of these two algorithms, they have been applied to several classic problems with different sizes and number of classes taken from the UCI repository [2]. The reason to choose these general problems instead of the Automatic Authority Control is the limitations of the ad-hoc pseudo-distance required for it. The most popular classic semi-supervised clustering algorithms, like *COPK-Means* [109] are based on the *k-means* algorithm, that cannot be applied to that problem.

The performance of the *Fuzzy HSS* and *Crisp HSS* methods is tested in comparison with other proposals: *COPKmeans* [109], *Upper-Tail* [83] and a traditional unsupervised algorithm [28] that we will be referring as *Unsupervised. COPKMeans* has been chosen as it applies semi-supervision in a traditional way. Additionally, *Fuzzy HSS* and *Crisp HSS* have been compared with two unsupervised methods to obtain the optimal partition of a dendrogram (*Upper-Tail* and *Unsupervised*).

For this study, we have used the same five datasets than on Section 3.4:

- *Glass*: 214 instances and 6 classes with very different sizes.
- *Iris*: 150 instances and 3 classes of the same size. Two of the classes are overlapped.
- *Wine*: 178 instances and 3 classes of different sizes.
- *Pen Digits*: 1000 instances, 10 classes. This dataset contains more instances than the previously considered ones, hence, more elements per class.
- *Vowel*: 11 classes, 1000 instances. As with PenDigits, this dataset is quite big and allow us to test our algorithms in bigger problems.

These datasets, as they all have different class distribution and sizes, allow us to test our algorithms in a wide range of problems. Additionally, as they are labeled, these class labels can be used to simulate the role of an expert. These labels have only been used to generate external information (constraints) and do not have any other influence in the clustering process.

As in Section 3.4, instance level constraints have been generated according to the class labels of the datasets. For that purpose, a *must-link* constraint has been created for every pair of instances belonging to the same class. To fuzzify them into *fuzzy must-link*, a random degree of belief β has been assigned to each pair. In the same way, for every pair of elements that do not belong to the same class, there is a *cannot-link* constraint. Again, to fuzzify them into *fuzzy cannot-link* the degree of belief on that constraint γ has been assigned randomly. These random degrees of belief have been used to simulate an expert that is giving true information but with not so much knowledge about the problem. For us, this is the worst case scenario, as we consider that the information given by the expert is always correct.

3.5.1 Experiment setup

The parameter setting for the different algorithms is as follows:

- *Complete Linkage* [61] with Euclidean distance is the hierarchical clustering algorithm applied on the *Fuzzy HSS*, *Crisp HSS*, *Upper Tail* and *Unsupervised* methods.
- Parameter K, representing the number of groups in *COPKmeans* is set to the number of classes on each dataset.
- Parameter k in *Upper Tail* method has been set to 1.25 according to [80]. This k should not been mistaken with the number of groups used in the *COP-Kmeans.*
- The number of constraints for *Fuzzy HSS*, *Crisp HSS* and *COP-Kmeans* methods has been determined by means of fuzzy entropy as described on Section 3.4. That methodology to determine the optimal number of constraints has been applied to the fuzzy constraints, and to better compare results, the resulting reduced A_m and A_c are converted into A for the crisp algorithm.
- Weights u_m, u_c, u_{nm}, u_{nc} for *Crisp HSS* in Expression 3.3 have been set to $u_m = u_c = 1$ and $u_{nm} = u_{nc} = -0.5$, with the intention of stressing the fulfilled constraints and penalize contradicted ones.
- Weights v_m, v_c, w_m, w_c for *Fuzzy HSS* in Expression 3.12 have been set to $v_m = v_c = 1$ and $w_m = w_c = -0.5$, with the same meaning as for *Crips HSS.*

3.5.2 Validation measures

To validate the performance of the clustering methods, three different indexes are obtained for each of the experiments:

- Purity [78]. It measures how the clustering solution fits the classes as (3.27):

$$purity(C,\Theta) = \frac{1}{N}\max_j |c_k \cap \theta_j| \tag{3.27}$$

  Where $C = c_1, \ldots, c_k$ is the partition obtained from the clustering process and $\Theta = \theta_1, \ldots, \theta_j$ are the classes according to dataset labels. N is the number of instances.

- Normalized Mutual Information [78], *NMI*. Defined as (3.28):

$$NMI(C,\Theta) = \frac{I(C,\Theta)}{[H(C) + H(\Theta)]/2} \tag{3.28}$$

  Where $I(C,\Theta) = \sum_k \sum_j = \frac{|c_k \cap \theta_j|}{N} \log \frac{N(c_k \cap \theta_j)}{|c_k||\theta_j|}$ and H is the entropy of the partition. C, Θ and N have the same meaning as in purity.

- Adjusted Rand Index, *aRand* [60]. Given a contingency table $[n_{ij}]$ where each entry n_{ij} denotes the number of objects in common between cluster C_i and Θ_j $n_{ij} = |C_i \cap \Theta_j|$. The adjusted Rand Index is defined as (3.29):

$$aRand = \frac{\sum_{ij}\binom{n_{ij}}{2} - [\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}]/\binom{n}{2}}{\frac{1}{2}[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}] - [\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}]/\binom{n}{2}} \tag{3.29}$$

  Where a_i and b_j are the sum of the rows and columns of the contingency table, respectively.

These measures provide a proper characterization of the clustering solution in comparison with the expected labels. They have been chosen for these experiments because of the labeled data, in contrast with the measures used in Chapter 3. Those measures were oriented to study the distribution of single elements in the solution and were more appropriated for unlabeled data and expert validation.

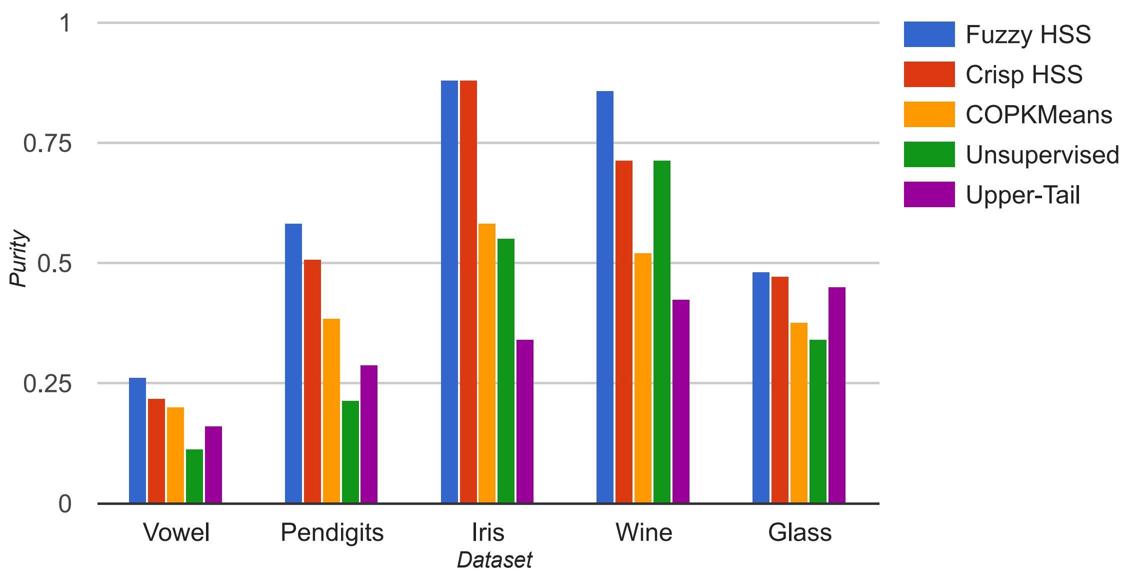


Figure 3.8: Purity of different methods to obtain the optimal α-cut of a dendrogram.

3.5.3 Results

Results from the experiments can be seen in Figures 3.8, 3.9 and 3.10. Considering the three indexes at the same time, it is possible to see that the hierarchical semi-supervised methods (*Fuzzy HSS* and *Crisp HSS*) outperform the other methods, specially for Purity (Figure 3.8) and Adjusted Rand Index (Figure 3.10). *COP-Kmeans* performs worse in all considered experiments and the performance of the *Unsupervised* and *Upper-Tail* is different for each dataset. Specifically, *Upper Tail* method returns higher values of NMI than *Fuzzy HSS* in some of the datasets. However, this outperform in that particular index is related with the very high number of clusters achieved by the *Upper Tail* method, returning from twice to ten times more groups than the expected number.

On Table 3.2, there is a one-way ANOVA for the Purity, NMI and aRand indexes using method as factor. As it is possible to read in the table, p-values are lower than the significance level 0.05. From this result, it can be deduced that the means of

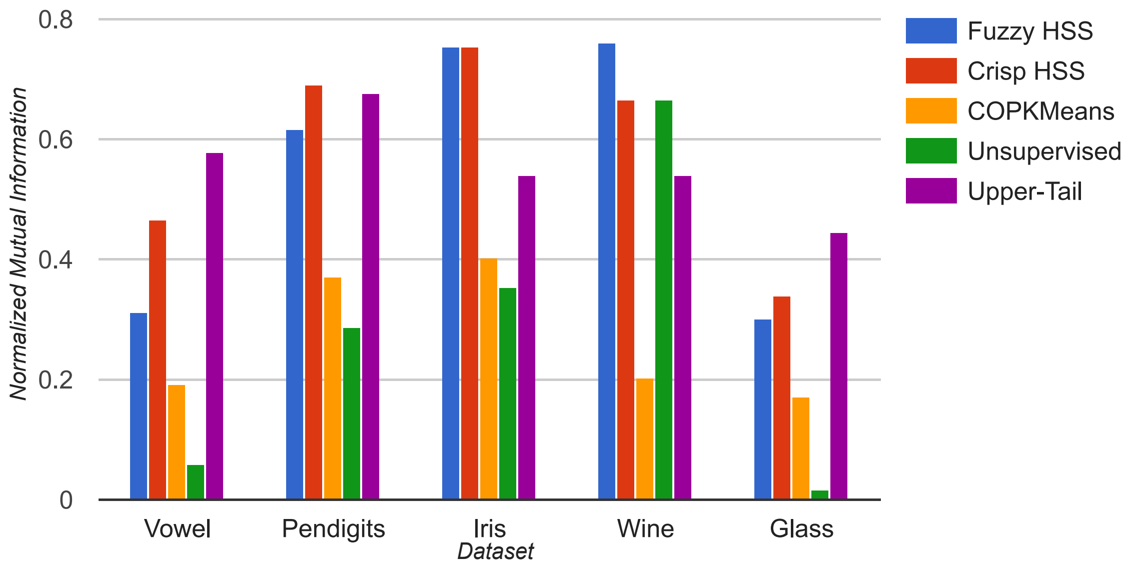


Figure 3.9: Normalized Mutual Information of different methods to obtain the optimal α-cut of a dendrogram.

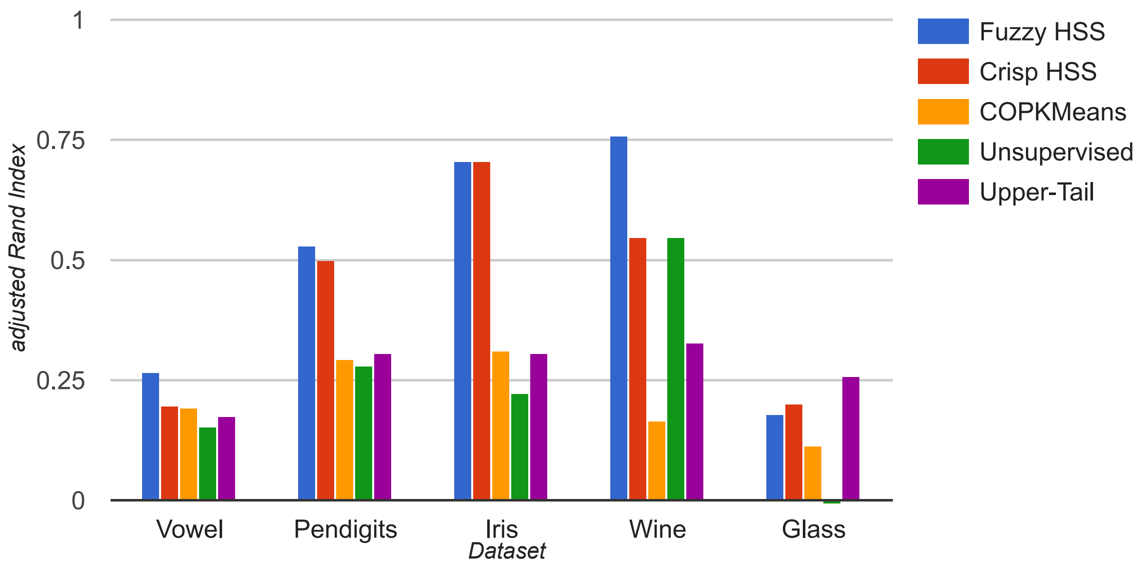


Figure 3.10: Adjusted Rand Index of different methods to obtain the optimal α-cut of a dendrogram.

Measure		Df	Sum Sq	Mean Sq	F val	Pr (>F)
Purity	Method	4	2.844	0.7109	18.01	5.6e-13
	Residuals	245	9.673	0.0395		
NMI	Method	4	4.906	1.226	37.22	<2e-16
	Residuals	245	8.074	0.033		
aRand	Method	4	3.063	0.7658	22.13	1.3e-15
	Residuals	245	8.478	0.0346		

Table 3.2: One way ANOVA with method as factor for Purity, Normalized Mutual Information and Adjusted Rand Index.

		Fuzzy HSS	Crisp HSS	COPKmeans	Upper Tail
Purity	Crisp HSS	0.1751	-	-	-
	COPKmeans	9.6e-05	0.0003	-	-
	Upper Tail	2.0e-11	3.9e-08	0.0459	-
	Unsupervised	3.2e-08	2.0e-05	0.4958	0.1866
		Fuzzy HSS	Crisp HSS	COPKmeans	Upper Tail
NMI	Crisp HSS	0.34	-	-	-
	COPKmeans	2.6e-13	4.8e-16	-	-
	Upper Tail	0.74	0.19	2.1e-12	-
	Unsupervised	1.2e-12	2.5e-15	0.80	9.5e-12
		Fuzzy HSS	Crisp HSS	COPKmeans	Upper Tail
aRand	Crisp HSS	0.12	-	-	-
	COPKmeans	3.2e-12	2.5e-08	-	-
	Upper Tail	3.7e-09	8.8e-06	0.22	-
	Unsupervised	1.3e-10	5.6e-07	0.54	0.55

Table 3.3: Pairwise t-test for the for Purity, Normalized Mutual Information and Adjusted Rand Index of the different methods.

the validation measures are different for each method. For a deeper study on these differences, we have performed a pairwise t-test that can be seen on Table 3.3. From that Table, it is possible to read that the differences on the means of the *Fuzzy HSS* and the *Crisp HSS* are not different for all the considered indexes, meaning that these two methods behave similarly. It means that even with the flexibility, and thus uncertainty, provided by the fuzzy model it is possible to use it with similar results than the crisp model. When comparing *Fuzzy HSS* with other methods, these differences depend on the considered index, arising similar conclusions than the ones from Figures 3.8, 3.9 and 3.10.

3.6 Crisp HSS and Fuzzy HSS in the Authority Control problem

On previous experiments we have tested the performance of *Crisp HSS* and *Fuzzy HSS* in a general environment. In the following, we have tested these techniques in the *Automatic Authority Control* problem. This study will be focused on two main aspects:

- The study of the performance of Crisp HSS and Fuzzy HSS in the Authority Control problem.
- The study of the performance according to the amount of constraints introduced in the process.

3.6.1 Experimental study of the performance of Semisupervised methods in the Authority Control problem.

To test the performance of the *Crisp HSS* and *Fuzzy HSS* in the Authority Control, all data used in the experiments in 2.6.5, corresponding to the searches of 8 different surnames in three different Digital Libraries: *CiteSeerX*, *DBLP* and *INSPEC*.

Details regarding the size of the datasets can be found in Table 2.2. The system's configuration used includes titles, authors and abstracts when available, using the same measures than in Section 2.6.5: *pn-measure* for author and coauthor names and cosine similarity for terms from title and abstracts.

Crisp Constraints for *Crisp HSS* have been provided by an expert. Their fuzzy version for *Fuzzy HSS* have been generated from the crisp ones by assigning a random degree of belief. The fuzzy entropy method described on Section 3.4 has been used to determine the amount of external information required by the fuzzy semi-supervised algorithm. The same amount of constraints has been applied to the *Crisp HSS*. Considering the random selection of the constraints for the entropy process, all executions of the semi-supervised methods have been performed 5 times showing the results on average.

Fuzzy HSS and Crisp HSS methods have been compared with the traditional unsupervised method described in [28] and *Upper-Tail* [83]. Method COPKmeans [109] used in Section 3.5 has not been included as it cannot be applied in this problem because it works only with euclidean distance, not allowing ad-hoc measures. For each dataset, five different performance measures have been considered: percent of Correct, Incorrect and Not Grouped elements, Purity and Inverse Purity. These measures, described in Section 2.6.3 and 2.5 have been considered the best to study this problem. Weights u_m, u_c, u_{nm}, u_{nc} for *Crisp HSS* have been set to $u_m = u_c = 1$ and $u_{nm} = u_{nc} = -0.5$, with the intention of stressing the fulfilled constraints and penalize contradicted ones. Weights v_m, v_c, w_m, w_c for *Fuzzy HSS* have been set to $v_m = v_c = 1$ and $w_m = w_c = -0.5$, with the same meaning as for *Crips HSS*.

Figures 3.11, 3.12, 3.13, 3.14 and 3.14 show the average performance of the different methods for the considered problems. Data has been grouped according to

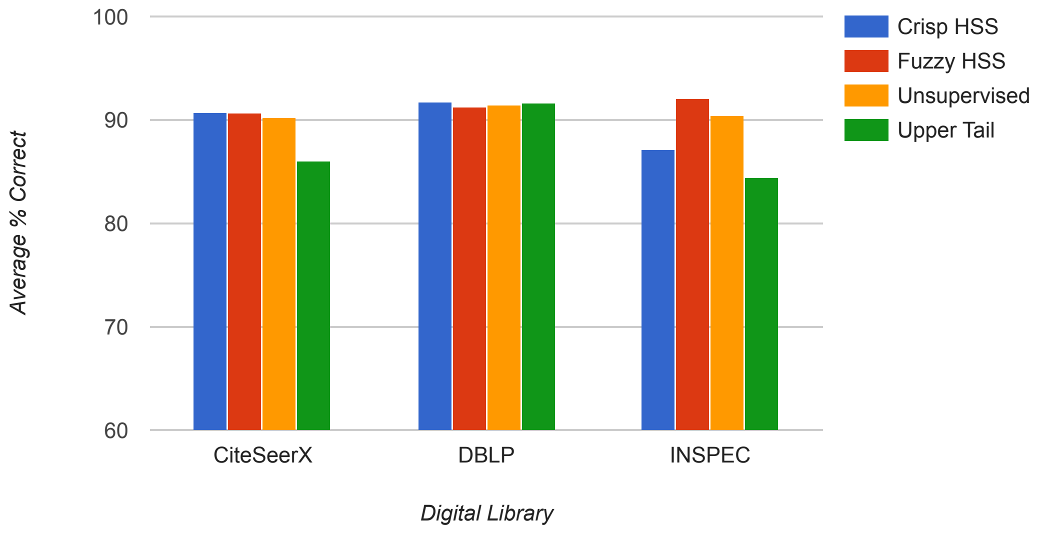


Figure 3.11: Comparison of different methods to cut the dendrogram. Correct Elements (results shown on average).

their origin to visualize properly how the methods behave in the different Digital Libraries considered. Tables 3.4, 3.5 and 3.6 show the results detailed by each considered surname.

Focusing on Figure 3.11 where we can see the percentage of correct elements, it is possible to see that *Fuzzy HSS*, *Crisp HSS* and the *Unsupervised* methods have a very similar performance, specially for data from CiteSeerX and DBLP. If we compare the results of these graphs with Tables 3.4, 3.5 and 3.6, it is possible to see that the semi-supervised methods perform best in almost all datasets, and in those cases where they do not, the outperforming is by less that 1% of correct elements. It is interesting to consider the performance of the *Fuzzy HSS* method in comparison with the *Crisp HSS* in the INSPEC datasets where the first clearly outperforms the latter. This behaviour is probably related with the nature of that data where the groups are smaller and the fuzzy constraints are more able to characterize it than

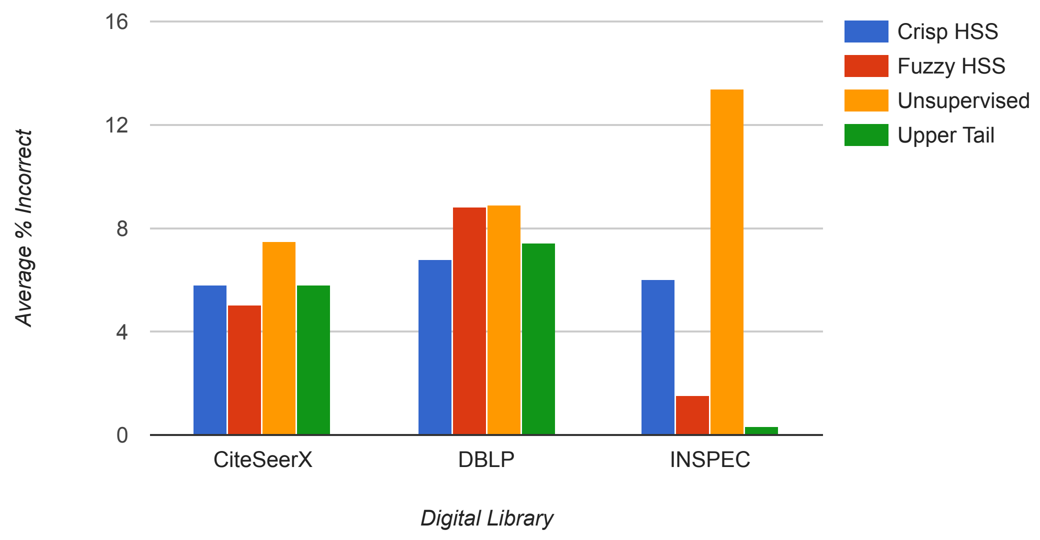


Figure 3.12: Comparison of different methods to cut the dendrogram. Incorrect Elements (results shown on average).

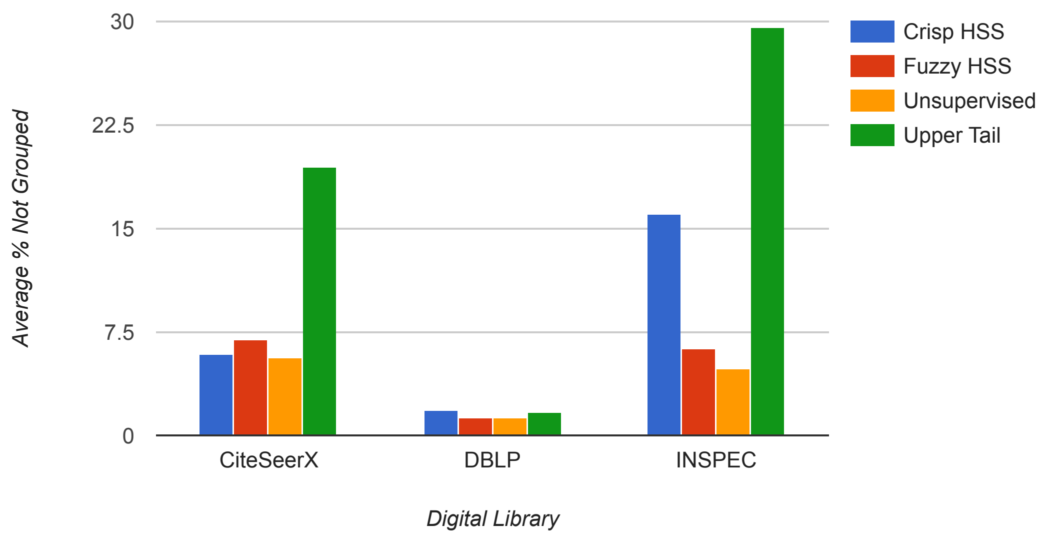


Figure 3.13: Comparison of different methods to cut the dendrogram. Not Grouped Elements (results shown on average).

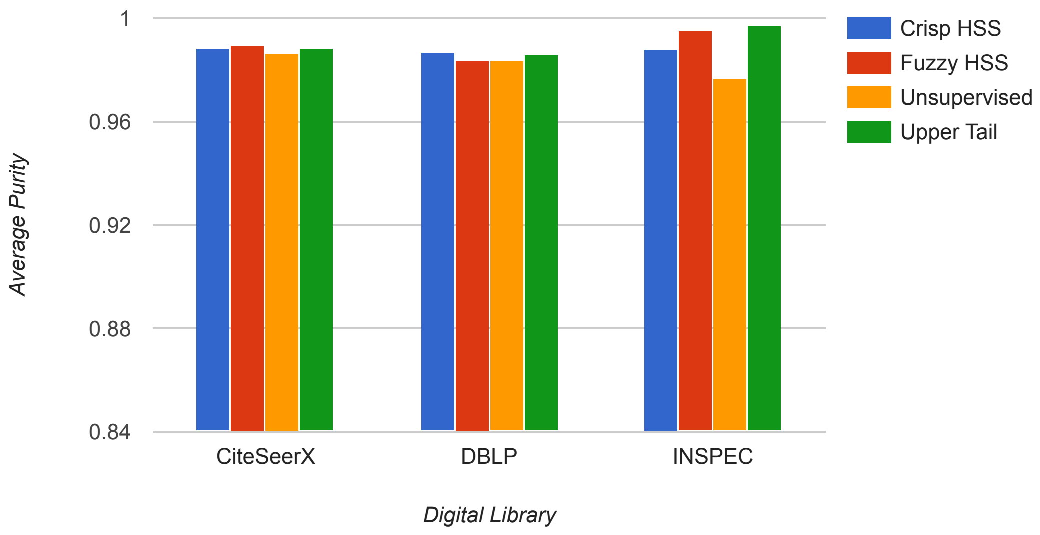


Figure 3.14: Comparison of different methods to cut the dendrogram. Purity (results shown on average).

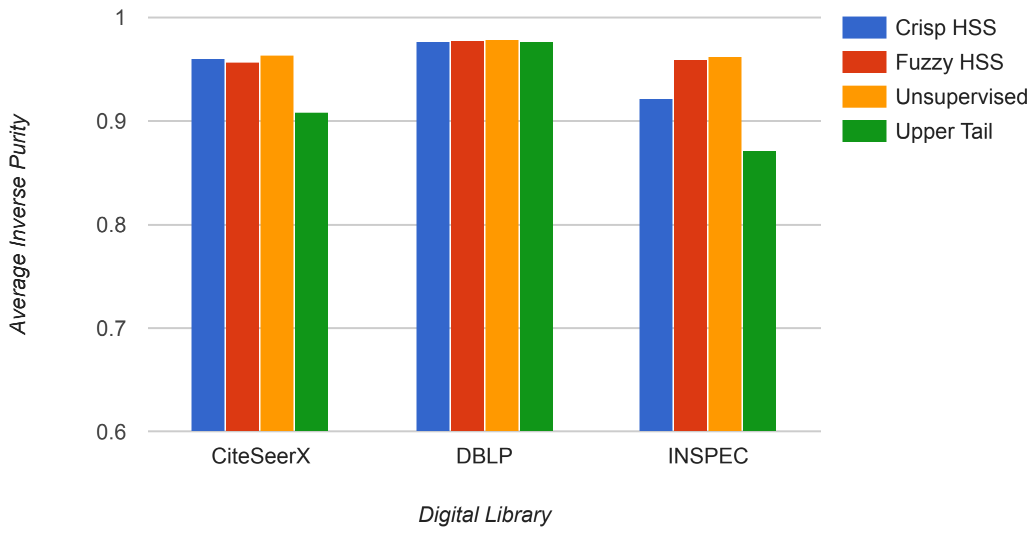


Figure 3.15: Comparison of different methods to cut the dendrogram. Inverse Purity (results shown on average).

their crisp counterpart.

It is interesting to see the results of the "negative" measures, *Incorrect* and *Not grouped*, in Figures 3.12 and 3.13, respectively. They show us that the *Crisp HSS* and *Fuzzy HSS* return less incorrect elements than the other two methods, except for *Upper Tail* in the INSPEC data. Not Grouped Elements are higher in the semi-supervised methods, meaning that they isolate some elements instead of grouping them incorrectly. We consider that kind of behaviour is desirable for this problem as the users should receive the authorities with as few mistakes as possible. On the other hand, *Upper Tail* method, green in the Figures, is clearly outperformed by the other methods.

The rest of the measures Purity (Fig. 3.14) and Inverse Purity (Fig. 3.15) reinforce the behaviour seen in the rest of the graphs. *Fuzzy HSS*, red in the Figures, is the method that provides a better general performance, specially for the Inspec datasets. Considering that Purity and Inverse Purity have very similar values in all problems, it means that the results are pretty consistent and fit the expected solutions pretty clearly.

To statistically test how different are the methods among themselves, we have performed an ANOVA test that can be seen on Table 3.7. From that table, it is possible to see that there are significant differences in two of the three measures: Correct elements, Not Grouped elements and Inverse Purity. This means that the performance of the methods is different according to the specific aspect that is being considered.

In Table 3.8 there is a Pairwise t-test comparing the different methods for each measure. As it is possible to see in the Table, the differences are not statistically

Surname	Method	Correct (%)	Incorrect (%)	N.G (%)	Purity	Inv Purity
Brown	Crisp	89.66	9.29	1.05	0.91	0.98
	Fuzzy	89.41	9.54	1.05	0.90	0.98
	Unsupervised	89.27	9.68	1.05	0.91	0.98
	Upper Tail	87.39	10.92	1.68	0.89	0.97
Delgado	Crisp	85.71	3.17	11.11	0.97	0.87
	Fuzzy	85.71	3.17	11.11	0.97	0.87
	Unsupervised	82.01	6.88	11.11	0.93	0.88
	Upper Tail	31.75	0.00	68.25	1.00	0.32
Garcia	Crisp	89.70	4.16	6.14	0.96	0.94
	Fuzzy	90.36	0.77	8.88	0.99	0.91
	Unsupervised	88.74	5.77	5.49	0.94	0.94
	Upper Tail	88.77	5.75	5.48	0.94	0.95
Johnson	Crisp	88.00	6.88	5.12	0.93	0.93
	Fuzzy	87.00	7.25	5.75	0.93	0.93
	Unsupervised	88.10	7.72	4.18	0.92	0.94
	Upper Tail	86.88	8.75	4.38	0.91	0.94
Sanchez	Crisp	89.64	4.77	5.59	0.95	0.94
	Fuzzy	89.55	4.41	6.04	0.96	0.93
	Unsupervised	84.68	11.26	4.05	0.89	0.95
	Upper Tail	43.69	0.00	56.31	1.00	0.44
Smith	Crisp	85.95	7.64	6.40	0.92	0.93
	Fuzzy	86.20	7.31	6.49	0.93	0.92
	Unsupervised	84.95	8.66	6.39	0.91	0.93
	Upper Tail	81.82	9.09	9.09	0.91	0.90
Vila	Crisp	89.26	5.19	5.56	0.95	0.94
	Fuzzy	87.96	2.41	9.63	0.98	0.90
	Unsupervised	89.81	2.78	7.41	0.97	0.91
	Upper Tail	88.89	4.63	6.48	0.95	0.93
Williams	Crisp	88.56	5.51	5.93	0.94	0.94
	Fuzzy	87.93	5.55	6.51	0.94	0.93
	Unsupervised	87.24	7.11	5.65	0.93	0.94
	Upper Tail	88.73	7.31	3.97	0.93	0.96

Table 3.4: Results of the different methods and measures for the CiteSeerX data.

Surname	Method	Correct (%)	Incorrect (%)	N.G (%)	Purity	Inv Purity
Brown	Crisp	90.69	6.01	3.30	0.94	0.96
	Fuzzy	88.34	9.86	1.80	0.90	0.97
	Unsupervised	89.3	9.37	1.33	0.91	0.98
	Upper Tail	88.29	9.95	1.76	0.90	0.97
Delgado	Crisp	95.85	2.84	1.30	0.97	0.99
	Fuzzy	95.98	2.71	1.30	0.97	0.99
	Unsupervised	96.09	2.61	1.30	0.97	0.98
	Upper Tail	95.83	2.87	1.30	0.97	0.99
Garcia	Crisp	93.09	6.04	0.87	0.94	0.99
	Fuzzy	87.78	11.84	0.38	0.88	0.99
	Unsupervised	86.24	11.91	1.85	0.88	0.98
	Upper Tail	91.91	7.33	0.76	0.93	0.99
Johnson	Crisp	89.75	7.53	2.71	0.92	0.96
	Fuzzy	87.33	10.95	1.72	0.89	0.97
	Unsupervised	89.72	9.47	0.81	0.91	0.98
	Upper Tail	87.71	10.62	1.67	0.89	0.97
Sanchez	Crisp	86.07	11.27	2.66	0.89	0.96
	Fuzzy	84.72	12.90	2.38	0.87	0.97
	Unsupervised	83.99	14.19	1.82	0.86	0.97
	Upper Tail	90.34	6.29	3.37	0.94	0.96
Smith	Crisp	89.24	8.52	2.24	0.91	0.96
	Fuzzy	88.76	9.68	1.56	0.90	0.97
	Unsupervised	90.01	8.57	1.42	0.91	0.99
	Upper Tail	88.29	10.23	1.48	0.90	0.97
Vila	Crisp	96.31	2.92	0.77	0.97	0.98
	Fuzzy	96.92	2.31	0.77	0.98	0.98
	Unsupervised	97.69	1.54	0.77	0.98	0.98
	Upper Tail	95.38	1.92	2.69	0.98	0.96
Williams	Crisp	89.41	9.59	1.00	0.90	0.98
	Fuzzy	88.59	10.53	0.88	0.89	0.99
	Unsupervised	89.20	9.87	0.94	0.90	0.98
	Upper Tail	89.02	10.19	0.79	0.90	0.99

Table 3.5: Results of the different methods and measures for the DBLP data.

Surname	Method	Correct (%)	Incorrect (%)	N.G (%)	Purity	Inv Purity
Brown	Crisp	93.50	0.00	6.50	1.00	0.94
	Fuzzy	91.70	2.90	5.40	0.97	0.94
	Unsupervised	92.23	0.00	7.77	1.00	0.92
	Upper Tail	76.50	0.00	23.50	1.00	0.77
Delgado	Crisp	74.98	0.00	25.02	1.00	0.75
	Fuzzy	96.81	0.40	2.79	1.00	0.97
	Unsupervised	92.37	6.83	0.80	0.93	0.97
	Upper Tail	43.03	0.00	56.97	1.00	0.43
Garcia	Crisp	88.94	0.00	11.06	1.00	0.88
	Fuzzy	94.84	1.47	3.69	0.99	0.95
	Unsupervised	90.00	0.00	10.00	1.00	0.90
	Upper Tail	73.73	0.00	26.27	1.00	0.73
Johnson	Crisp	72.16	0.00	27.84	1.00	0.72
	Fuzzy	92.06	2.27	5.67	0.98	0.94
	Unsupervised	89.64	0.00	10.36	1.00	0.89
	Upper Tail	69.59	0.00	30.41	1.00	0.70
Sanchez	Crisp	81.83	5.67	12.50	0.94	0.87
	Fuzzy	93.33	0.42	6.25	1.00	0.94
	Unsupervised	86.67	11.25	2.08	1.00	0.96
	Upper Tail	64.17	0.00	35.83	1.00	0.64
Smith	Crisp	84.18	0.00	15.82	1.00	0.84
	Fuzzy	91.73	1.63	6.63	0.98	0.93
	Unsupervised	89.29	0.00	10.71	1.00	0.89
	Upper Tail	81.12	0.00	18.88	1.00	0.81
Vila	Crisp	57.14	42.86	0.00	0.57	0.96
	Fuzzy	84.78	0.36	14.86	1.00	0.84
	Unsupervised	75.72	0.00	24.28	1.00	0.75
	Upper Tail	84.42	2.54	13.04	0.97	0.86
Williams	Crisp	70.72	0.00	29.28	1.00	0.71
	Fuzzy	91.60	2.87	5.52	0.97	0.93
	Unsupervised	83.05	0.00	16.95	1.00	0.83
	Upper Tail	68.51	0.00	31.49	1.00	0.69

Table 3.6: Results of the different methods and measures for the INSPEC data.

Measure		Df	Sum Sq	Mean Sq	F val	Pr (>F)
Correct	Method	3	1783.06	594.35	5.75	0.0012
	Residuals	92	9510.82	103.38		
Incorrect	Method	3	49.19	16.40	0.51	0.6792
	Residuals	92	2981.61	32.41		
Not Grouped	Method	3	2176.68	725.56	5.47	0.0017
	Residuals	92	12210.17	132.72		
Purity	Method	3	0.01	0.00	0.53	0.6630
	Residuals	92	0.30	0.00		
Inverse Purity	Method	3	0.21	0.07	5.56	0.0015
	Residuals	92	1.18	0.01		

Table 3.7: One way ANOVA for the different methods as factor for Correct, Incorrect, Not Grouped, Purity and Inverse Purity.

		Crisp HSS	Fuzzy HSS	Upper Tail
Correct	Fuzzy HSS	0.16	-	-
	Upper Tail	0.02	0.00	-
	Unsupervised	0.44	0.53	0.00
		Crisp HSS	Fuzzy HSS	Upper Tail
Incorrect	Fuzzy HSS	0.51	-	-
	Upper Tail	0.29	0.69	-
	Unsupervised	0.93	0.56	0.33
		Crisp HSS	Fuzzy HSS	Upper Tail
Not Grouped	Fuzzy HSS	0.36	-	-
	Upper Tail	0.01	0.00	-
	Unsupervised	0.53	0.78	0.00
		Crisp HSS	Fuzzy HSS	Upper Tail
Purity	Fuzzy HSS	0.50	-	-
	Upper Tail	0.28	0.68	
	Unsupervised	0.95	0.55	0.31
		Crisp HSS	Fuzzy HSS	Upper Tail
Inverse Purity	Fuzzy HSS	0.34	-	-
	Upper Tail	0.01	0.00	-
	Unsupervised	0.49	0.80	0.00

Table 3.8: Pairwise t-test for the different methods considered for this problem.

significant between the *Crisp HSS* and *Fuzzy HSS*, so both methods can be used. It is interesting to mention that the differences between the semisupervised methods and the traditional unsupervised are not significant. However, if we check in Tables 3.4, 3.5 and 3.6 the semi-supervised methods outperform the traditional unsupervised in most cases. Those cases where there is not an outperforming, the results are very similar, meaning that the *Unsupervised* method found the best cut possible. On the other hand, for those cases where there is still room for improvement, semi-supervised methods could improve the correct elements more than 10%, as happens with datasets Vila and Williams from Inspec.

From the results of this experiment it is possible to read that the semi-supervised methods *Fuzzy HSS* and *Crisp HSS* work properly for this problem. They are also able to find the best cut of the dendrogram, even in those problems where the Unsupervised methods are not able to find it.

3.6.2 Study of the influence of the amount of external information.

For this experiment, it is our intention to test how the performance of the Authority Control problem evolves according to the amount of external information introduced in the semi-supervised clustering algorithms, *Fuzzy HSS* and *Crisp HSS*. For that purpose both algorithms have been applied to the eight datasets from CiteSeerX corresponding to the Spanish and English surnames (see Section2.6.2). The external information has been added iteratively to the process, randomly selecting a very small initial set and repeating the process adding more constraints iteratively. Weights for both algorithms and other configurations of the system are the same than in previous section. Again, as the constraints are added to the process randomly, each experiment have been performed 5 times and the results are shown in average.

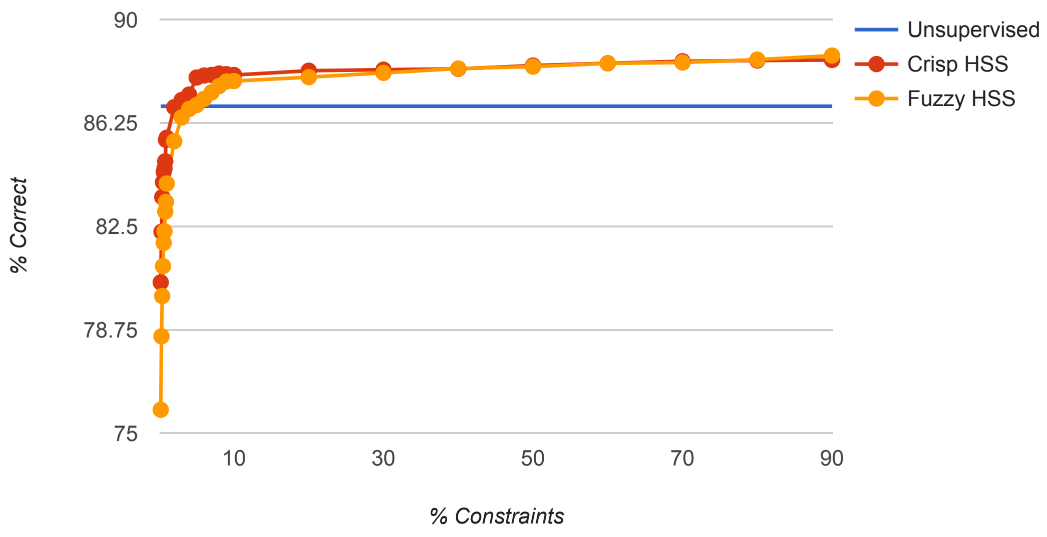


Figure 3.16: Evolution of the percentage of Correct Elements regarding the amount of external information.

Figures 3.16, 3.17, 3.18, 3.19 and 3.20 show the average evolution of results for the eight considered datasets with different amount of constraints under the different measures considered. The traditional Unsupervised method has been added as a baseline for comparison. As it is possible to see in Figure 3.16 for the Correct elements, with very limited external information, less than 5% of the total constraints the results are over 80% but do not outperform the Unsupervised method. By adding more information it is possible to improve these results, but at a certain point they stabilize and there is no further improvement. This behaviour is observable in both *Crisp HSS* and *Fuzzy HSS*.

The same behaviour is observable the rest of the Figures (Fig. 3.17, 3.18, 3.19 and 3.20) with very few constraints the performance is poor and it improves until a certain point where there is no further improvement in the results. When adding a

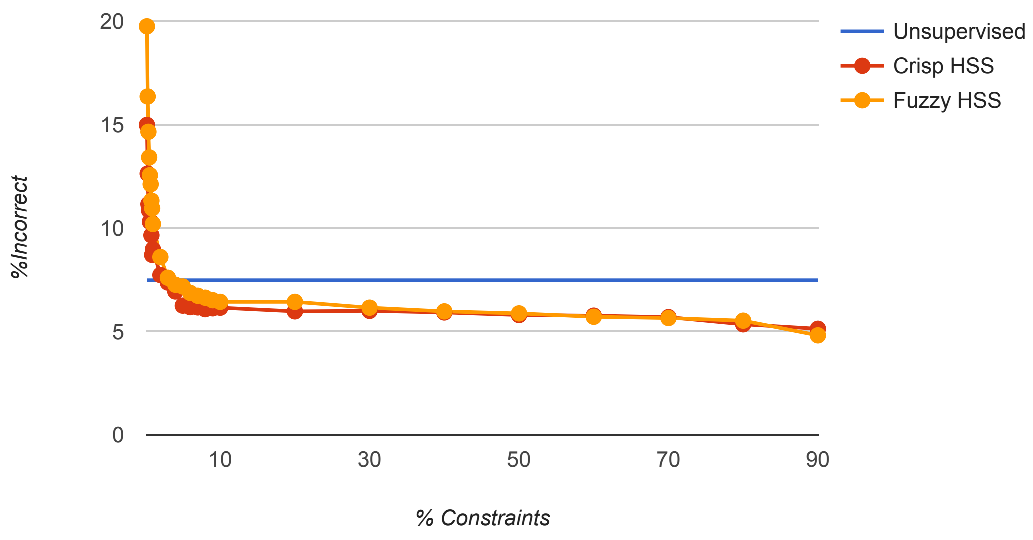


Figure 3.17: Evolution of the percentage of Incorrect Elements regarding the amount of external information.

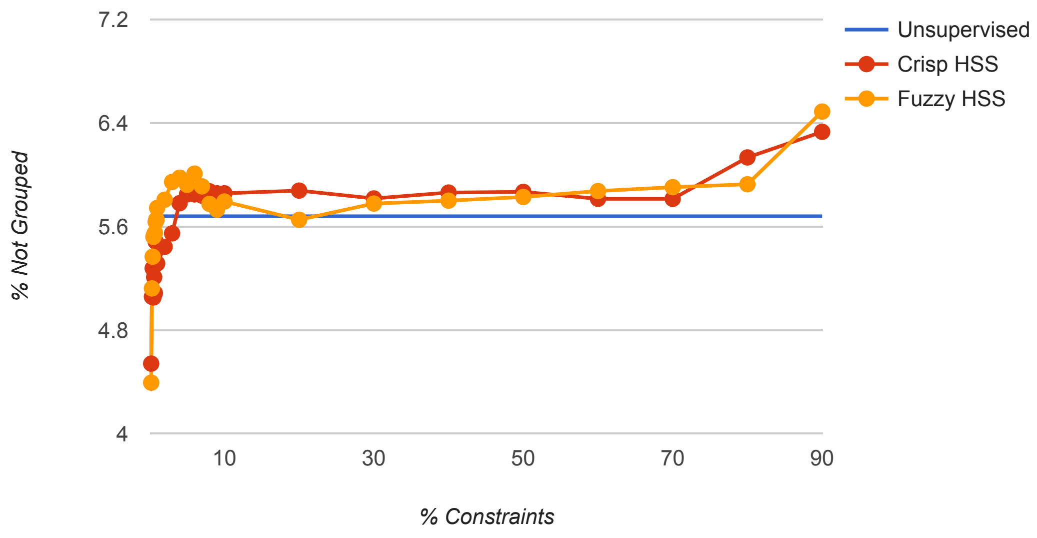


Figure 3.18: Evolution of the percentage of Not Grouped Elements regarding the amount of external information.

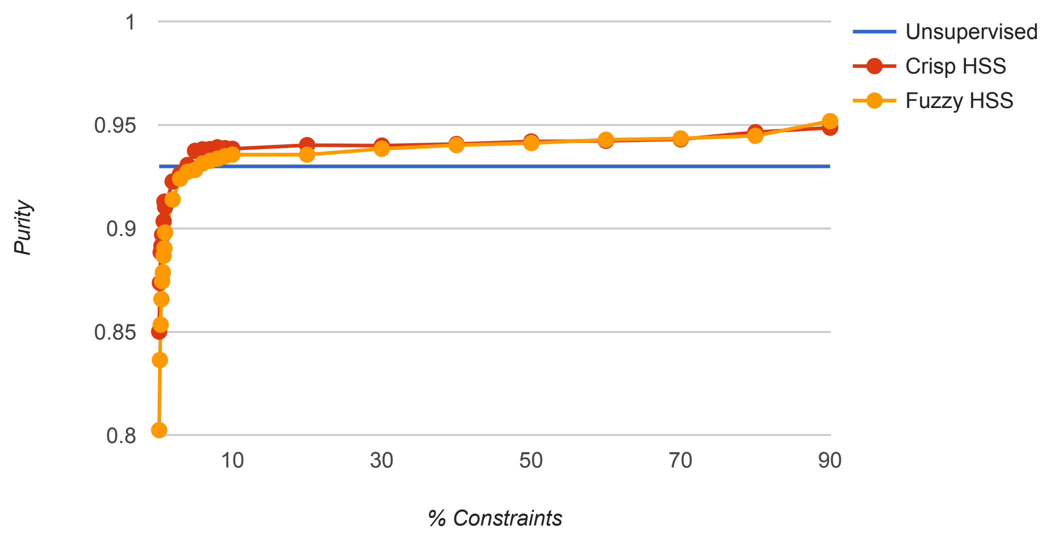


Figure 3.19: Evolution of the Purity regarding the amount of external information.

lot of external information, around 80% - 90%, the number of not grouped elements increases and the incorrect decreases. This means that some elements that previously where incorrectly placed now appear isolated. However, even if that behaviour is advisable, it is not recommended for semi-supervised algorithms to put that many external information in the process.

To statistically test the evolution of the constraints, an Anova test has been performed for the results of the process using 0.5%, 1%, 5%, 10%, 50% and 90% of the external information. Table 3.9 shows the results of this test for each considered measures for the *Crisp HSS* method. The same test for the *Fuzzy HSS* method can be seen on Table 3.10. At it is possible to see in both cases, the differences are significant for three of the considered measures: Correct, Incorrect and Purity. For Not grouped and Inverse purity we cannot say that there are any differences in the means, so their results are very similar.

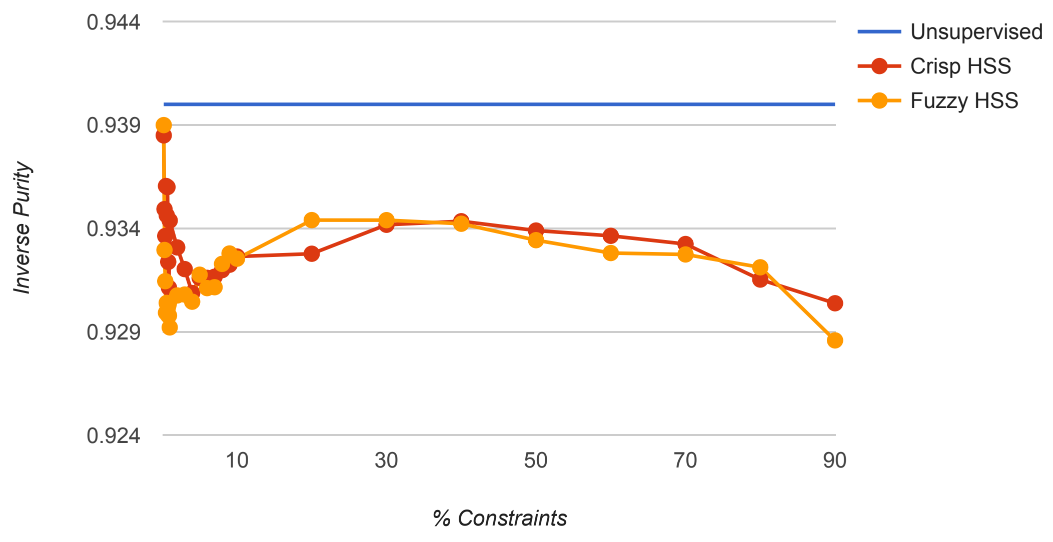


Figure 3.20: Evolution of the percentage of Inverse Purity regarding the amount of external information.

To perform a deeper study of these results, a pairwise t-test has been performed to check each of the levels against the remaining ones. Tables 3.11 and 3.12 show that information. From the tables it is possible to read that the means differences are not significant for those executions using 5% and more of constraints, in both algorithms. This test confirms what we have observed in Figures 3.16, 3.17, 3.18, 3.19 and 3.20, the results tend to stabilize and there is a point where futher improvement cannot be achieved.

3.6.3 Discussion

The application of *Crisp HSS* and *Fuzzy HSS* to the Automatic Authority Control problem has been studied in two ways. Fist, we studied the performance of these methods on the problem, and then, we considered interesting to see how the

Measure		Df	Sum Sq	Mean Sq	F val	Pr (>F)
Correct	Constraints	5	127.3	25.466	4.257	0.0032
	Residuals	42	251	5.982		
Incorrect	Constraints	5	197.3	39.46	5.59	0.000491
	Residuals	42	296.5	7.06		
Not Grouped	Constraints	5	8.3	1.669	0.216	0.954
	Residuals	42	324.1	7.717		
Purity	Constraints	5	0.01973	0.003946	5.594	0.000488
	Residuals	42	0.02963	0.000705		
Inverse Purity	Constraints	5	0.00017	0.0000333	0.042	0.999
	Residuals	42	0.03351	0.0007978		

Table 3.9: One way ANOVA for the Crisp HSS with different amounts of instance level constraints as factor for Correct, Incorrect, Not Grouped, Purity and Inverse Purity.

Measure		Df	Sum Sq	Mean Sq	F val	Pr (>F)
Correct	% Constraints	5	127.33	25.47	4.26	0.0032
	Residuals	42	251.25	5.98		
Incorrect	% Constraints	5	197.30	39.46	5.59	0.0005
	Residuals	42	296.49	7.06		
Not Grouped	% Constraints	5	8.35	1.67	0.22	0.9536
	Residuals	42	324.11	7.72		
Purity	% Constraints	5	0.02	0.00	5.59	0.0005
	Residuals	42	0.03	0.00		
Inverse Purity	% Constraints	5	0.00	0.00	0.04	0.9989
	Residuals	42	0.03	0.00		

Table 3.10: One way ANOVA for the Fuzzy HSS with different amounts of instance level constraints as factor for Correct, Incorrect, Not Grouped, Purity and Inverse Purity.

		0.5%	1%	5%	10%	50%
Correct	1%	0.19429	-	-	-	-
	5%	0.00340	0.08133	-	-	-
	10%	0.00277	0.06982	0.94092	-	-
	50%	0.00125	0.03811	0.72423	0.78037	-
	90 %	0.00078	0.02615	0.60618	0.65869	0.87032
		0.5%	1%	5%	10%	50%
Incorrect	1%	0.16528	-	-	-	-
	5%	0.00126	0.04702	-	-	-
	10%	0.00101	0.03985	0.94064	-	-
	50%	0.00046	0.02152	0.73429	0.79096	-
	90 %	9.9e-05	0.00610	0.40444	0.44719	0.61931
		0.5%	1%	5%	10%	50%
Not Grouped	1%	0.85	-	-	-	-
	5%	0.57	0.70	-	-	-
	10%	0.57	0.70	1.00	-	-
	50%	0.56	0.69	0.99	0.99	-
	90 %	0.36	0.47	0.73	0.73	0.74
		0.5%	1%	5%	10%	50%
Purity	1%	0.16512	-	-	-	-
	5%	0.00125	0.04693	-	-	-
	10%	0.00101	0.03978	0.94072	-	-
	50%	0.00046	0.02147	0.73430	0.79088	-
	90 %	9.9e-05	0.00608	0.40437	0.44706	0.61923
		0.5%	1%	5%	10%	50%
Inverse Purity	1%	0.91	-	-	-	-
	5%	0.75	0.85	-	-	-
	10%	0.81	0.90	0.94	-	-
	50%	0.88	0.97	0.87	0.93	-
	90 %	0.69	0.78	0.93	0.87	0.80

Table 3.11: Pairwise t-test for the Crisp HSS with different amounts of instance level constraints as factor for Correct, Incorrect, Not Grouped, Purity and Inverse Purity.

		0.5%	1%	5%	10%	50%
Correct	1%	0.19	-	-	-	-
	5%	0.00	0.08	-	-	-
	10%	0.00	0.07	0.94	-	-
	50%	0.00	0.04	0.72	0.78	-
	90%	0.00	0.03	0.61	0.66	0.87
		0.5%	1%	5%	10%	50%
Incorrect	1%	0.17	-	-	-	-
	5%	0.00	0.05	-	-	-
	10%	0.00	0.04	0.94	-	-
	50%	0.00	0.02	0.73	0.79	-
	90%	0.00	0.01	0.40	0.45	0.62
		0.5%	1%	5%	10%	50%
Not Grouped	1%	0.85	-	-	-	-
	5%	0.57	0.70	-	-	-
	10%	0.57	0.70	1.00	-	-
	50%	0.56	0.69	0.99	0.99	-
	90%	0.36	0.47	0.73	0.73	0.74
		0.5%	1%	5%	10%	50%
Purity	1%	0.17	-	-	-	-
	5%	0.00	0.05	-	-	-
	10%	0.00	0.04	0.94	-	-
	50%	0.00	0.02	0.73	0.79	-
	90%	0.00	0.01	0.40	0.45	0.62
		0.5%	1%	5%	10%	50%
Inverse Purity	1%	0.91	-	-	-	-
	5%	0.75	0.85	-	-	-
	10%	0.81	0.90	0.94	-	-
	50%	0.88	0.97	0.87	0.93	-
	90%	0.69	0.78	0.93	0.87	0.80

Table 3.12: Pairwise t-test for the Fuzzy HSS with different amounts of instance level constraints as factor for Correct, Incorrect, Not Grouped, Purity and Inverse Purity.

different amounts of external information affected in the performance of the method.

From the first experiment, it is interesting to conclude that these methods are able to find a better cut of the dendrogram than the Unsupervised methods. However, as the system described in Chapter 2 is able to obtain a good dendrogram, the outperforming of these methods is not as clear as with the general problems studied on Section 3.5. This lead us to the idea that *Crisp HSS* and *Fuzzy HSS* are a good alternative to those problems where the unsupervised methods do not return good solutions.

The study of the influence of the constraints confirms the idea that both *Crisp HSS* and *Fuzzy HSS* behave quite similarly. Additionally it is possible to see that even if the results tend to improve when the more information is introduced in the process, the improvement is limited after a certain amount of constraints. It is quite interesting to see what happens when there is less than 10% of external information. In those points, the improvements are quite big, leading us to the idea that it necessary to add, at least 5% or 10% of constraints to obtain a good behaviour.

3.7 Conclusions

The methods described in this chapter rely on semi-supervision to find the optimal α-cut of a dendrogram, so it needs external information in the shape of instance level constraints. They have been tested in a wide range of general problems and in the Automatic Authority Control problem, outperforming in both cases the classical methods.

The use of fuzzy information is more flexible than the crisp model used in *Crisp*

HSS. It allows *Fuzzy HSS* to be applied on those cases where it is not possible, or it is very difficult, to find an expert with certain knowledge about the problem. To properly compare these two methods in a general context, we have used classic labeled datasets. Nevertheless, in real-life problems like Authority Control, the labels are unknown and the role of the expert is crucial; therefore giving her/him as much flexibility as possible could be very beneficial for the process. Obviously, as this method requires an expert, it cannot be applied to those problems where she/he is not available. For those cases classic unsupervised clustering is a good option and the optimal partition of the dendrogram can be obtained through traditional methods like *Upper-Tail*. However, as it has been shown in Section 3.5, the use of semi-supervision improves the performance of the clustering. In addition to that, the use of fuzzy instance level constraints, not only is more flexible, but provide a mechanism to know how many information must be required to the expert.

The benefits of instance level constraints are not just related with a better performance of the algorithm, but they could also provide valuable information after the clustering process. These relationships can be considered as hints that help to the interpretation of the information inside the clusters in real-life problems where the classes are not available. Moreover, the use of a hierarchical clustering approach avoids giving the number of clusters, as with the COP-Kmeans algorithm. It makes this method quite suitable for those problems where this parameter is unknown.

This approach applies semi-supervision once the hierarchical clustering had been applied. This makes it very flexible, as the clustering method and the distance between elements must be chosen by the user. This means that the improvement that can be achieved with the semi-supervision is related to the quality of the dendrogram itself. If the hierarchical clustering algorithm used to obtain the dendrogram does not find a good grouping of the data, the semi-supervised method is going to find its best partition, but probably, it does not have a very good purity, as can be seen on Figure 3.8 with *Vowel*, *Pendigits*, and *Glass* experiments. On the other hand, when the unsupervised methods are able to find a pretty good solution, as with the *Automatic Authority Control* problem (Figures 3.11, 3.12, 3.13, 3.14 and 3.14) the semi-supervised clustering outperform traditional methods, but this outperform is more discrete than in other cases.

The outperforming of the *Crisp HSS* and *Fuzzy HSS* in comparison with the unsupervised method, as it can be seen on Figure 3.8 and Table 3.3 is related with the exploration of the dendrogram by the constraints. It tries to find the partition that better fits them, being also able to overcome the limitations of a non-fulfilled constraint. Let us remember that a contradicted constraint gives a negative feedback to the overall, but it does not invalidate the whole partition, as happens with the COP-Kmeans algorithm.

In conclusion, the use of semi-supervision could be a very powerful tool when used in hierarchical clustering. *Crisp HSS* and *Fuzzy HSS* algorithms are able to find the best partition of a dendrogram without modifying the distance measure suggested for the problem. They are able to outperform unsupervised solutions when adding a limited amount of external information.

Chapter 4

Automatic Constraints Generation for Semi-supervised Clustering

In Chapter 3, we showed up how the semi-supervision can be used to improve results in clustering processes. That semi-supervision came from experts with some kind of knowledge about the problems approached. In an attempt to ease the task of the expert, we proposed the use of fuzzy semi-supervision, allowing the expert to give impressions based on degrees of belief instead of certain knowledge. Nevertheless, there are still some problems where such experts are not available.

To try to provide a mechanism to apply semi-supervision to such problems, in this chapter we propose a method to automatically generate crisp *instance level constraints*.

4.1 Automatic Constraints Generation

The human expertise is preferable when supervision is recommended, although user-provided suggestions or hints are often expensive and time consuming to obtain.

However, in some approaches, human intervention can be replaced by automatically generated knowledge. The study of the data and their placement in the n-dimensional space evidences some structural relationship that can be a valid support for driving in the constraints attribution.

In this approach, constraints are generated by the study of the inherent nature of the data. It is done using a partitional clustering process that obtains a partition of the data according to some distance criteria. Specifically, our method uses k-means [62], a well-known flat clustering that finds a partition $P_K = \{S_1, \ldots, S_k\}$ for a given k by minimizing the within-cluster sum of squares, according to the following objective function:

$$J = \sum_{i=1}^{k} \sum_{j=1}^{n} \|x_{i,j} - \mu_i\|^2 \tag{4.1}$$

where $\|x_{i,j} - \mu_i\|$ is the distance between a data point $x_{i,j}$ and the centroid μ_i of the cluster S_i.

The partition P_K gives an approximate idea about how the input data is organized. Regardless the possible mistakes and inaccuracies that P_K could present, it is possible to use this information to generate constraints. Each $S_a \in P_K = \{x_{k_1}, \ldots, x_{k_t}\}$ is a partition containing similar elements. If we consider that all elements from that partition should be in the same group, must-link constraints are pretty straightforward.

Formally: for each pair of instances $(x_i,\ x_j)$ that are in the same cluster in the clustering-driven partitioning, there exists a *must-link* constraint $\mathcal{ML}(x_i, x_j), \forall(x_i, x_j) \in S_a | S_a \in P_K$.

Under this assertion, the set of must-link constraints, $\mathcal{ML}$, contains all pairs of elements that are in the same cluster (considering all clusters independently): $\mathcal{ML} = \cup_{i=1}^{k} \mathcal{ML}_i$, where $\mathcal{ML}_i$ is a set of must-link constraints from a partition S_i.

Similarly, *cannot-link* constraints are defined between pairs of instances $(x_i,\, x_j)$ that are in different clusters, i.e. $\forall (x_i, x_j); d_i \in S_a, x_j \in S_b | S_a, S_b \in P_K; S_a \neq S_b$. Under this definition, the set of cannot-link constraints, $\mathcal{CL}$,contains all pairs of elements that are in the different clusters (considering all clusters independently): $\mathcal{CL} = \cup_{\forall a,b:a\neq b} \mathcal{CL}_{a,b}$, where $\mathcal{CL}_{a,b}$ is a set of cannot-link constraints composed of the pair $(x_i, x_j) \in S_a \times S_b | a \neq b$.

This approach is sensible to the initial configuration of the *k-means* algorithm. However, this can be overtaken by exploiting the random component of the k-means initialization. The partition returned by k-means depends of an initial centroid, μ_i that normally is randomly generated on each execution of the algorithm. It means that every execution may provide slightly different partitions. Under that assumption, by executing the algorithm repeatedly, the original set of constraints is refined defining the constraints. In this sense, if two instances x_i and x_j are placed in the same cluster in all executions of the k-means clustering, then there is a *must-link*, $\mathcal{ML}(x_i, x_j)$, constraint between them. In the same way, if two instances x_i and x_j are never placed in the same cluster in all executions, then there is a *cannot-link* constraint $\mathcal{CL}(x_i, x_j)$. This process has been summarized in Algorithm 4.

Considering that we are keeping only that information coherent in all the executions of the k-means clustering algorithm, there are several pairs of instances without an associated constraint. This is because they are placed in the same cluster in some executions and in different clusters in others. Moreover, data structure can affect the performance of k-means clustering, which generally tends to produce

Algorithm 4 Constraints generation

```
Get $P_K$ an initial partition returned by k-means
for all $K_i \in P_K$ do
    if $(d_i, d_j) \in K_i$ then
        Add $(d_i, d_j)$ to $ML$
    else
        Add $(d_i, d_j)$ to $CL$
    end if
end for
for each k-means execution do //
    Get $P_a = \{S_1, \ldots, S_k\}$ the partition returned by k-means
    for all $ML(d_i, d_j) \in ML$ do
        if $d_i \in S_a$ and $d_j \in S_b$ with $S_a \neq S_b$ then
            Remove $(d_i, d_j)$ from $ML$
        end if
    end for
    for all $CL(d_i, d_j) \in CL$ do
        if $d_i, d_j \in S_a$ then
            Remove $(d_i, d_j)$ from $CL$
        end if
    end for
end for
```

clusters of relatively uniform size. In case of bad partitioning, some of the resulting clusters could not be used for the constraint generation, because their data relations are considered not good enough for the constraints (for example if they are too big, compared with the remaining clusters). In that case, the constraints are generated considering only the clusters that fit some criteria, discarding all the remaining information.

Constraints generated under this method can be used for any semi-supervised clustering algorithm based on crisp instance level constraints.

4.2 Experimental Results

In this Section is our intention to study the performance of automatically generated constraints. To do so, several questions that arise in relation with the automatic generation will be analyzed:

- How the k-means configuration affects the quality of the constraints;
- How the automatically generated constraints using k-means perform in comparison with other methods to generate constraints;
- How the semi-supervised approach performs in comparison with unsupervised methods.

It is interesting to point out that the automatic generation of constraints is based on the *k-means* algorithm, which in based in *Euclidean* distance, so this automatic generation method could not be suitable for those problems where this distance is not available. For this reason, the automatic constraints generation is tested for the *document clustering* process. The purpose to do this is to experiment with the approach in a wider environment where the task of an expert could be very hard. This problem can be seen as a generalization of the Automatic Authority Control. Instead of grouping scientific publications by author, we group other kind of documents by topic.

Two datasets have been used to validate this automatic constraints generation approach. They have gone through a preprocessing process, where some cleaning and dimensionality reduction tasks have been carried out, specifically tailored with respect to the nature of each dataset:

- *Web Snippets* dataset [93]. Contains 2280 short texts taken from Google, unevenly divided into eight categories. Each snippet contains from 6 to 20 words approximately. During the preprocessing step, it has been prepared by removing the first words of each snippet, as they seem to be part of the URL of the original document (information that it is not useful for categorization). Additionally, a stemming process has been applied and its stop words have been removed. A dimensionality reduction has been performed by removing those words with a correlation higher than 0.5 using Pearson's method. During the constraint generation process, when executing the k-means algorithm, these clusters with sizes bigger than 25% of the whole dataset size have been ignored. Binary Term Frequency has been used as weighting measure for the terms of this dataset.

- *Reuters-21578* collection [73]. It is a subset of the well known Reuters-21578 collection containing 2014 documents. Ten independent categories have been selected from the dataset: *trade*, *ship*, *wheat-grain*, *gold*, *sugar*, *money-fx*, *interest*, *crude*, *money-supply* and *coffee* with different sizes. This dataset has been preprocessed using a lowercase representation, removing numbers, punctuation and stop words. It has also gone through a stemming process.

Dimensionality reduction has been performed using the 500 terms with higher TF-IDF value. As with the previous dataset, during the constraints generation process, these clusters containing more that 25% of documents have not been considered. TF-IdF has been used as weighting measure for the terms of this dataset.

For each dataset, constraints have been generated by 30 executions of the k-means algorithm. *Crisp HSS*, as defined in Section 3.2 is the semi-supervised clustering algorithm applied. As stated on its definition, it can be used to any hierarchical clustering algorithm that provides a dendrogram. So, for these data, we have used hierarchical agglomerative clustering algorithm with the Ward's method. The parameters setting for the formula in 3.3, that calculates the best partition according to the score r_α, are set to consider only satisfied constraints, so $u_{nm} = u_{nc} = 0$. In addition to that, weights u_m, associated to the importance of the must-link constraint and u_c, associated to the cannot-link, take the value $u_m = 2$, $u_c = 1$ to reinforce the information provided for the must link in contrast with the cannot-link. The reason to do that is because the automatic generation method, by definition, provides less *must-link* than *cannot-link* constraints. Using this setup, we obtain a more balanced contribution by the two types of constraints.

To validate the goodness of the clustering process, two different measures have been used: F-measure [4] and Normalized Mutual Information (briefly, NMI) [78].

F-measure evaluates the quality of the clusters by comparing the relationship between the retrieved documents on each cluster and the relevant documents according to their given class labels. F-measure (4.4) is defined as the harmonic mean of Precision (4.2) and Recall (4.3).

$$Precision = \frac{|\text{relevant documents} \cap \text{retrieved documents}|}{|\text{retrieved documents}|} \tag{4.2}$$

$$Recall = \frac{|\text{relevant documents} \cap \text{retrieved documents}|}{|\text{relevant documents}|} \tag{4.3}$$

$$F - measure = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.4}$$

The Normalized Mutual Information (4.5) evaluates the elements on each cluster against class labels. It measures and normalizes the mutual information between random variables P_α (the optimal partition of the dendrogram) and G (the ground truth given by class labels).

$$NMI(P_\alpha, G) = \frac{2I(P_\alpha, G)}{H(P_\alpha) + H(G)} \tag{4.5}$$

where $I(P_\alpha, G)$ is the mutual information between the two random variables and H is the Shannon entropy of the variable.

Sizes of constraints sets $\mathcal{ML}$ and $\mathcal{CL}$ generated automatically by this method are different for each dataset. Considering that the number of constraints introduced on the process of finding P_α affects its performance, different random subsets are taken from $\mathcal{ML}$ and $\mathcal{CL}$ of proportional size to the original set.

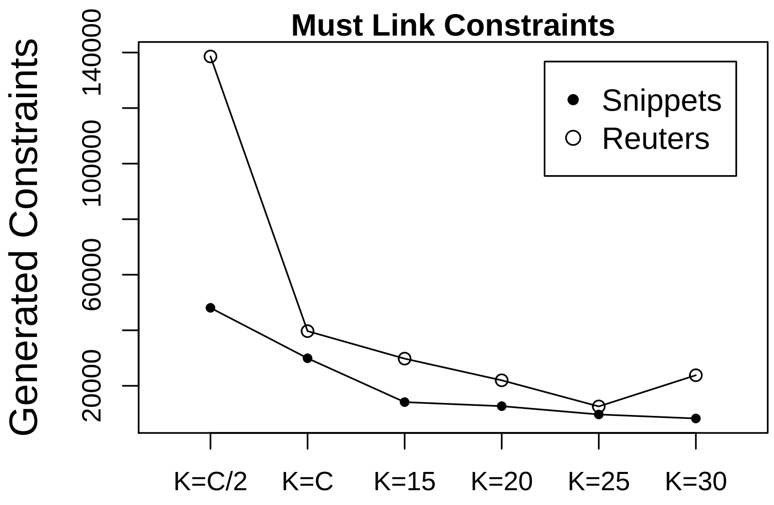
As in previous experimentation, in order to guarantee a complete analysis, different constraints sets of different sizes have been used, incrementally from 0.1% to 90% of each set. As these subsets of constraints are chosen randomly, each experiment has been executed 5 times using different random subsets of constraint. Thus, the values of F-measure and NMI on the next graphs, are the average of these 5 executions.

4.2.1 Effect of k-means configuration on the constraints

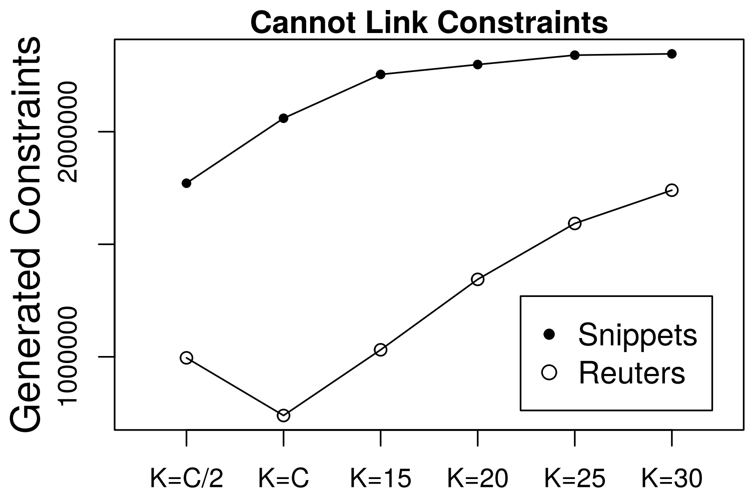
As stated, the constraint generation process is based on repeated executions of the k-means algorithm. This algorithm requires a parameter, k, representing the number of groups (clusters) in which the input data is split. The influence of k has been studied by generating several constraints sets with different values of k. Indeed, for each dataset, we have considered a wide range of k values that goes from half of expected groups according to class labels, to the effective expected groups (according to class labels), up to a maximum of $k = 30$ (see Figure 4.1). This range of values guarantees a complete view of the behaviour of this methodology; first considering a k smaller than the expected clusters sizes and then assessing how the increase of k impacts on the results.

Figure 4.1 shows the dependency between k (x axis) and the number of generated constraints (y axis). As it is possible to see in the Figure, k affects the size of the constraint sets. Let us notice that there are more *cannot-link* (Figure 4.1b) than *must-link* constraints (Figure 4.1a). This is due to the number of classes in the dataset: since a must-link constraint is defined between elements in the same class whilst a cannot-link is defined between elements in different classes. It results in more cannot-link constraints than must-link. This is because *cannot-link* relationships are defined between an element and all the elements within all other clusters, so, normally, there are more than, must-link than only exists between an element all others within its own cluster.

Figure 4.1a shows how the must-link constraints change as k increases. It is interesting to check how as must-link constraints decrease, the cannot-link increase (Figure 4.1b). This is related with the number of clusters and their sizes. A bigger value of k means smaller clusters, so less possibilities for the elements to must-link between each other, while there are more cluster (i.e., many subgroups of docu-

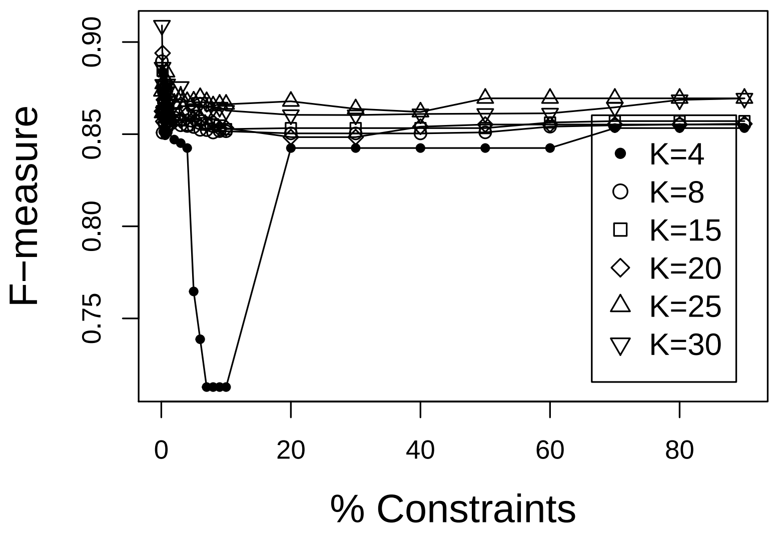


(a) Number of Must Link Constraints

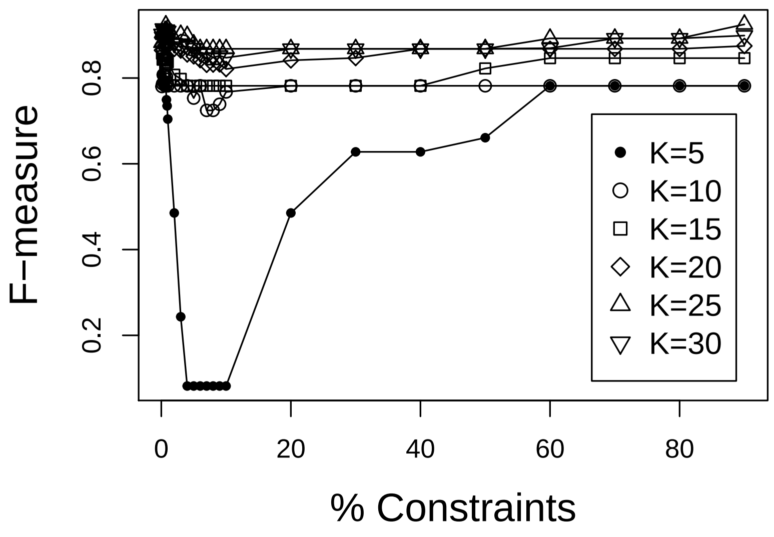


(b) Number of Cannot Link Constraints

Figure 4.1: Influence of the parameter k in the number of constraints.

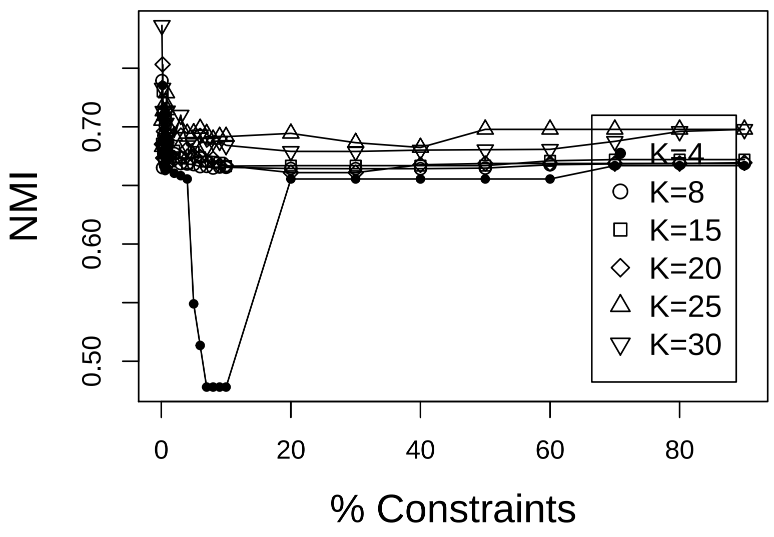


(a) Web snippets dataset

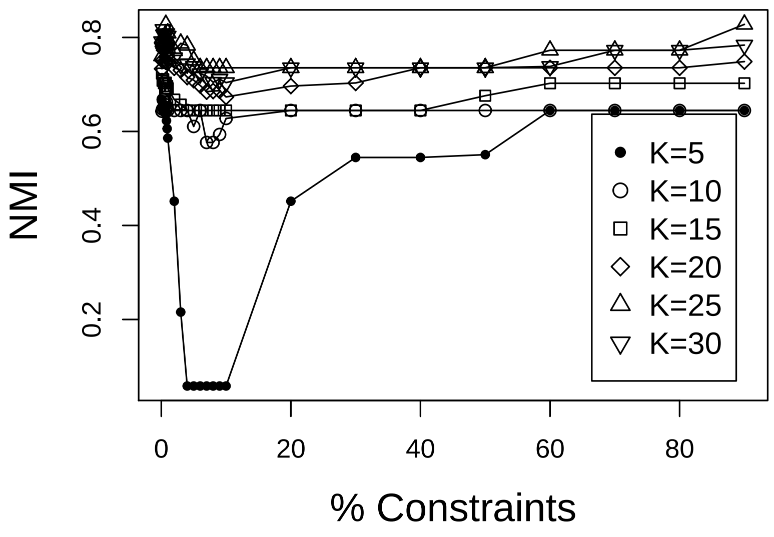


(b) Reuters dataset

Figure 4.2: F-measure with different K-mean configuration



(a) Web snippets dataset



(b) Reuters dataset

Figure 4.3: Normalized Mutual Information with different K-mean configuration

ments), so there are more possible combination for the cannot-link information. Similarly, when k is really small, there are more (must-links and cannot-links) constraints, but they are not of good quality, as information is mixed up. This can be seen in Figures 4.2 and 4.3, where the performance of the methodology for different values of k is shown.

Figure 4.2 shows the F-measure for all considered datasets. Let us notice that using a small k, the performance of this approach is poor, specially when the subset of constraints used is small. This can be seen more clearly for Reuters dataset (Figure 4.2b), where it needs up to 60% of the constraints to obtain an F-measure equal to 0.8. Figure 4.1a shows that the poor performance is related with the higher number of must-link constraints, coming from fewer clusters composed of mixed-data. This behaviour is also observable in Figure 4.3 where the Normalized Mutual Information is measured for all analyzed datasets.

As shown in Figure 4.1, the fixed number of cluster k in the k-means algorithm affects the number of constraints generated and then, the performance of our method when obtaining the partition of the document corpus (Figures 4.2 and 4.3). From the analysis of the figures, let us assert that the results tend to stabilize when the value of k is close to the number of classes or bigger. Specifically, in the Reuters dataset (Figures 4.2b and 4.3b), it is possible to observe a small improvement using K bigger than 15.

The resulting partition of a documents collection by the k-means algorithm often shows some small specific clusters and a big cluster with a lot of mixed data (with respect to the class labels). Particularly, by increasing k, the clusters get more specific and provide better constraints. Indeed, the number of generated must-link constraints, decreases as k increases because must-links are generated between instances inside the same cluster, so, having less elements per cluster provides less constraints. On the contrary, the cannot-link constraints increase, as more clusters means more cannot-link possibilities. This behaviour is clearly observable in the Reuters dataset (Figure 4.1), where, with a small k a lot of must-link constraints are generated; but increasing k those constraints tend to decrease. However, let us remark that the specific correspondence between the amount of must-link and cannot-link strictly depends on the dataset, its size and its underlying class distribution.

Since constraints are generated automatically, they may have inaccuracies. It means that some of the constraints could be not related with the actual structure of the data. These inaccuracies are mainly solved as k increases. Specifically, since the constraints are being generated with a more specific partition, the resulting must-link/cannot-link information should be of better quality, with a consequent improved performance. Figures 4.2 and 4.3 describe this behaviour. As k is getting bigger and the number of cannot-link constraint increases, the performance improves because the clustering algorithm tends to put documents together, so the cannot-link constraints help to split up those groups. At the same time, the contribution coming from must-link constraints is also important, because if there are only cannot-link constraints, the results will contain one or two documents per cluster, and that behaviour is not desirable. A good trade-off between the use of must-links and cannot-links justifies assigning a bigger weight to the u_m and u_{nm} parameters (in Equation 3.3), in order to compensate the outnumber of cannot-link constraints and their influence.

Figures 4.2 and 4.3 show also how the size of the sets of generated constraints introduced in the *Crisp HSS* algorithm affects the performance of the method. In general, with $k = 15$ and beyond (values that provide a good partitioning on these datasets), the results in terms of performance tend to stabilize: we can see that the differences in the results, when adding more constraints, are small. Adding more constraints suppose adding more information in the process, which increases the computational complexity. For that reason, and taking into account also the size the dataset, using around 30% of each must-link and cannot-link would be advisable.

4.2.2 Comparison with other kind of automatically generated constraints

Instance level constraints are traditionally provided by an human expert with some knowledge about the specific data. In [105] for instance, when class labels are available, the instance generation has been mapped to the provided classification (i.e., class labels have been used to get instance level constraints). For that purpose, a must link constraint $\mathcal{ML}(x_i, x_j)$ is defined between two instances x_i and x_j if they are labeled as to be in the same class. In the other hand, there is a cannot-link $\mathcal{CL}(x_i, x_j)$ between these elements that do not share the same label. Obviously, under this model, it is possible to reconstruct the original partition by using all the constraints generated under this model. As class labels have been provided by an expert, that model simulates the expert generated constraints.

Figures 4.4, 4.5, 4.6 and 4.7 show how automatically generated constraints perform in comparison with constraints generated using class labels. These figures compare automatic vs. label-based generated constraints both for the *best* (k=30) and the *worst* (k=4) number of clusters. Under the same amount of constraints, the partitioning coming from automatic constraints generation performs equally or better than the one obtained using constraints that come from class labels.

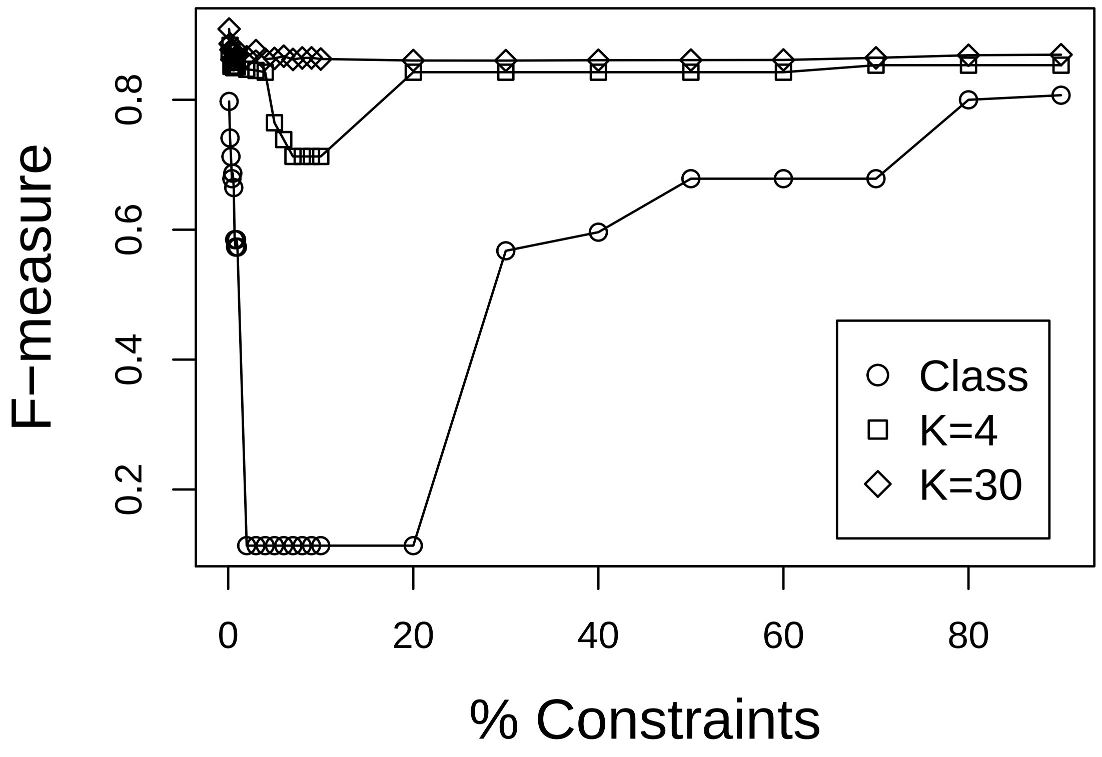


Figure 4.4: Comparison of the F-measure of class and k-means automatically generated constraints, Web snippets dataset

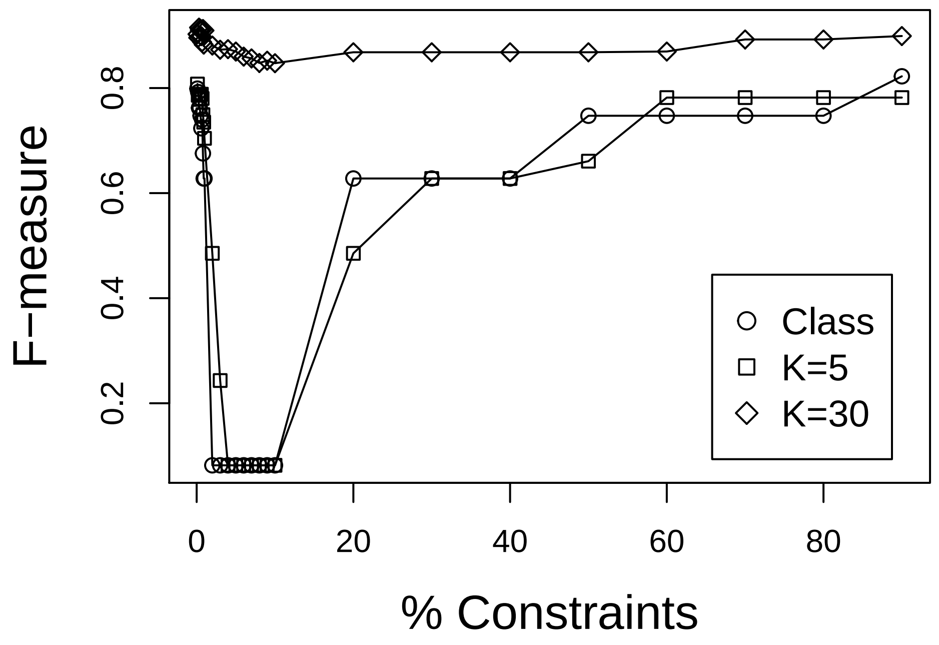


Figure 4.5: Comparison of the F-measure of class and k-means automatically generated constraints, Reuters dataset

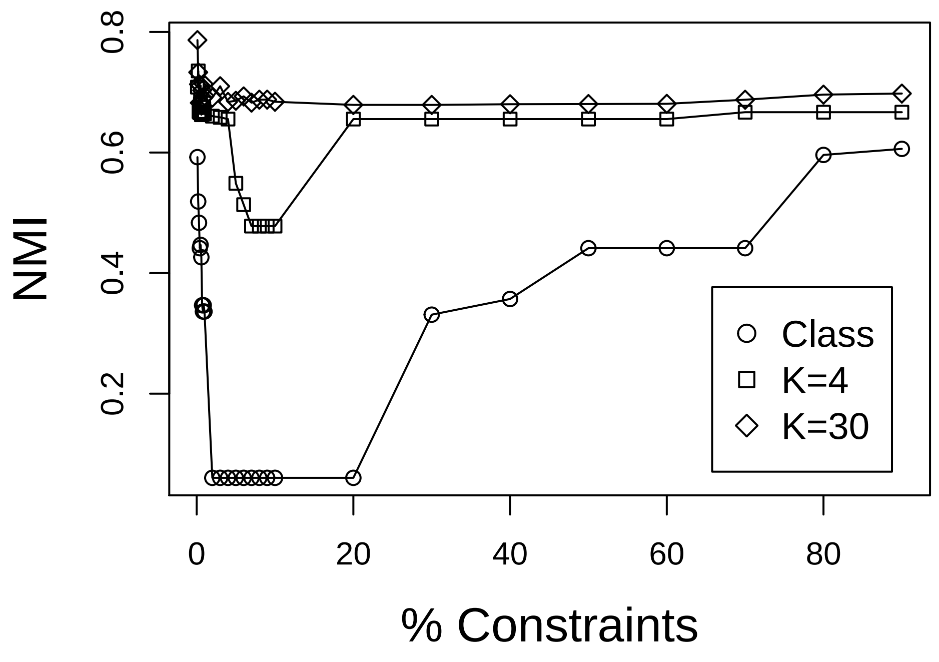


Figure 4.6: Comparison of the Normalized Mutual Information of class and k-means automatically generated constraints, Web snippets dataset.

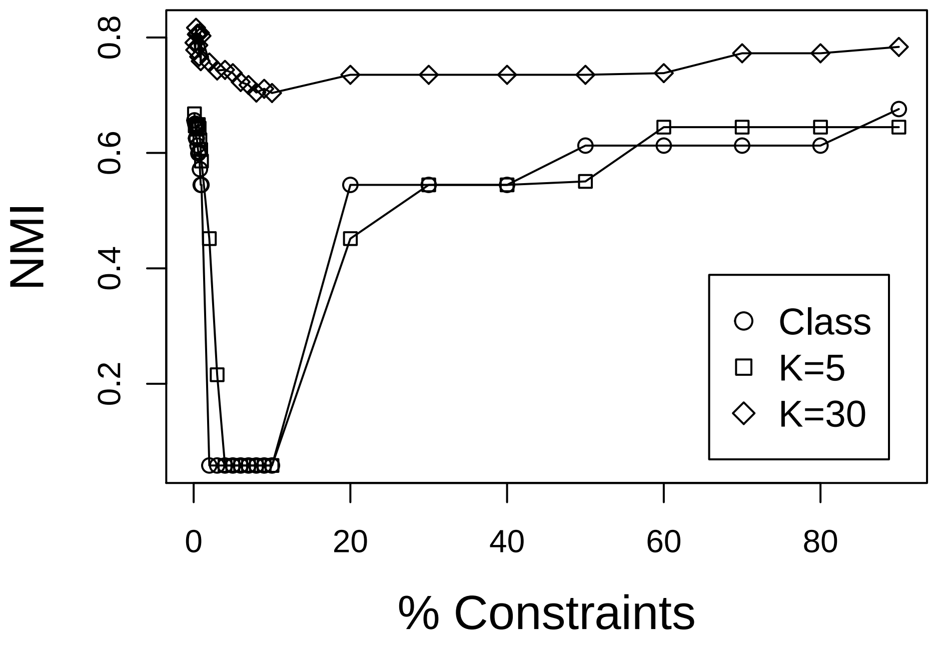


Figure 4.7: Comparison of the Normalized Mutual Information of class and k-means automatically generated constraints, Reuters dataset.

4.2.3 Comparison with unsupervised clustering

This method uses semi-supervision, through *Crisp HSS*, to calculate the optimal partition of a dataset by means of hierarchical clustering. The advantage of using hierarchical clustering is that the number of clusters is not fixed, so it is possible to get more specific clusters than using other methods in which this number of clusters is fixed. Figures 4.8, 4.9, 4.10 and 4.10 show how this method performs in comparison (in terms of F-measure and NMI) with some traditional unsupervised methods such as k-means and Ward's method.

This method (named *Auto* in the figures) has been evaluated with constraints generated with the best and the worst partitions for the two datasets (see Section 4.2.1), using 30% of the constraints.

Figures 4.8 and 4.9 show that the F-measure value from the semi-supervised method is higher than the F-measure value for the k-means and the Ward's method.

Let us remember that our semi-supervised approach makes use of hierarchical clustering and Ward's method to obtain the dendrogram. So, if we compare the partition obtained by cutting the dendrogram at the expected number of groups and our semi-supervised approach, let us observe that our method returns a more specific partition, since the clusters do not contain mixed data from different categories. Similar behaviour can be observed with the Normalized Mutual Information (Figures 4.10 and 4.10).

Moreover, this method outperforms the k-means algorithm, even if k-means was part of the constraints generation process. It means that k-means is not able to find by itself an accurate partition that represents the data, so the provided pairwise information are very useful to find the optimal cut of the dendrogram. As our semi-supervised clustering algorithm has some tolerance for not fulfilled constraints, it is able to overtake the possible mistakes that could be in the constraints by k-means, without affecting the performance, as shown in Figures 4.8, 4.9, 4.10 and 4.10.

Another important point is the role of the hierarchy-based partitioning in comparison with flat clustering, as they are able to provide a more specific cluster assignment. In particular, Figures 4.8, 4.9, 4.10 and 4.10 show how a specific partition outperforms those that come from obtaining the same number of partitions than the expected number of groups in the data. Our method does not modify the clustering algorithm itself, but it helps to find the better partition from the dendrogram D. It means that a partitioning that separates data into groups that "make sense" is already in the dendrogram, but it is not possible to obtain it but cutting the tree at the expected number of groups.

As an example, let us consider a dataset whose data have been split in several categories; where one of them is *sport*. Using the k-means algorithm to find the documents related with sports, would not get accurate results. It probably would return some documents related with some specific sport or there would be a lot of documents, with the information regarding sports probably mixed with some unrelated information. It depends on the "quality" of the document-term matrix: the selection of term-features (for instance, co-occurrences of terms) is crucial to get a good partitioning. Anyway, a good term-document matrix does not guarantee an accurate partitioning in flat clustering. It is possible that class labels are quite specific and that there is not enough vocabulary in the documents to identify them.

Instead, in hierarchical clustering, the partitioning generally reveals better cluster specialization, for instance, in the case of sports, related to football, tennis, basketball, etc. and these clusters normally do not contain mixed information. Indeed, thanks to the nature of the dendrogram, it would be possible to use its hierarchy of documents to provide a more in-depth insight of the terms of corpus that could lead to a more specific hierarchical classification.

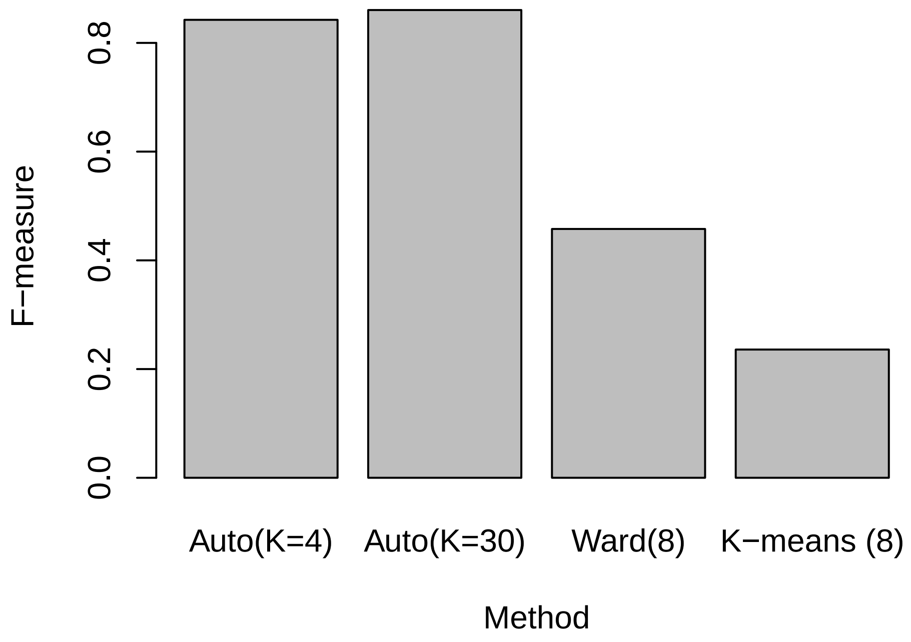


Figure 4.8: Comparison of the F-Measure of semi-supervised and unsupervised methods, Web snippets dataset.

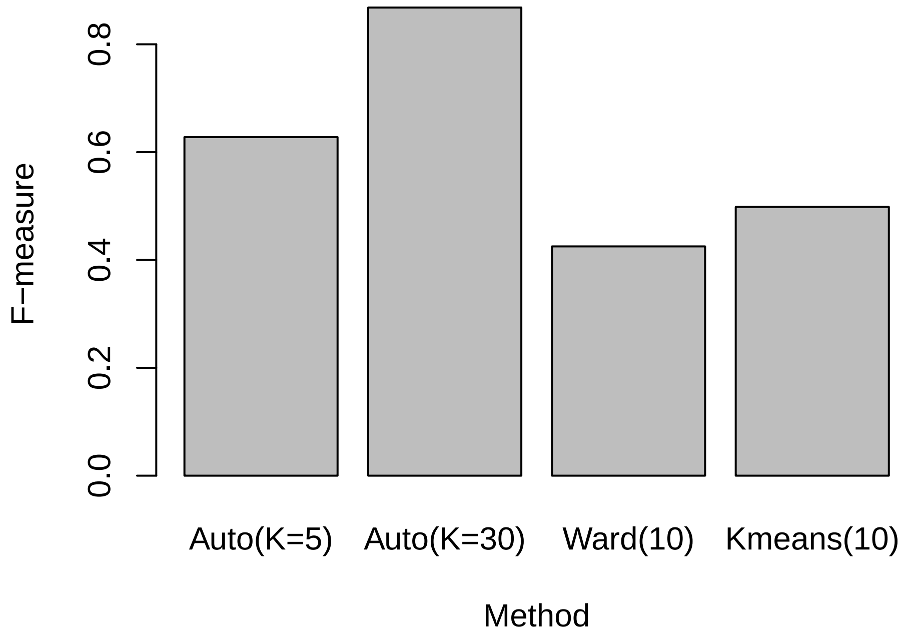


Figure 4.9: Comparison of the F-Measure of semi-supervised and unsupervised methods, Reuters dataset

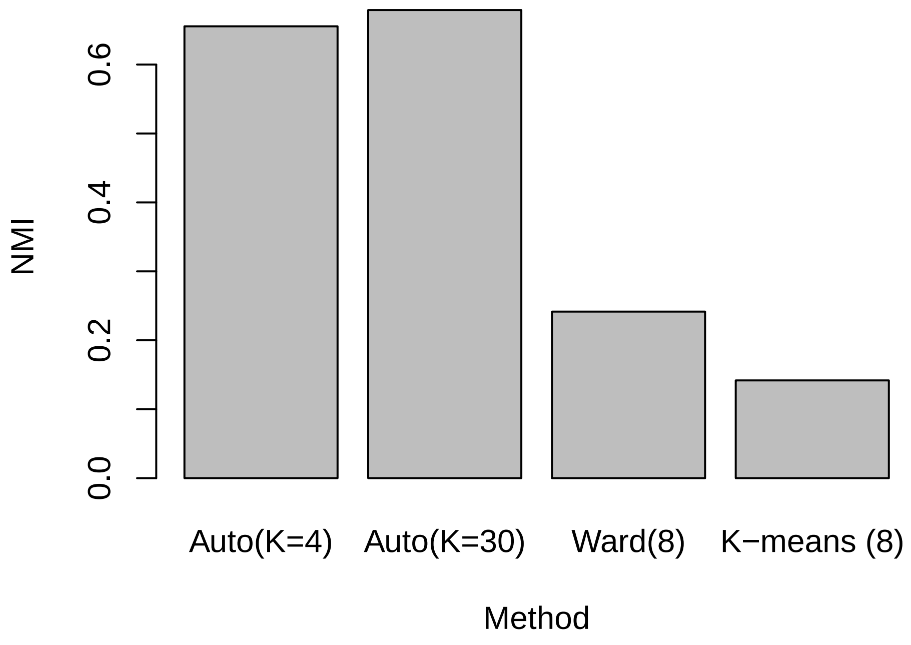


Figure 4.10: Comparison of the Normalized Mutual Information of semi-supervised and unsupervised method, Web snippets dataset

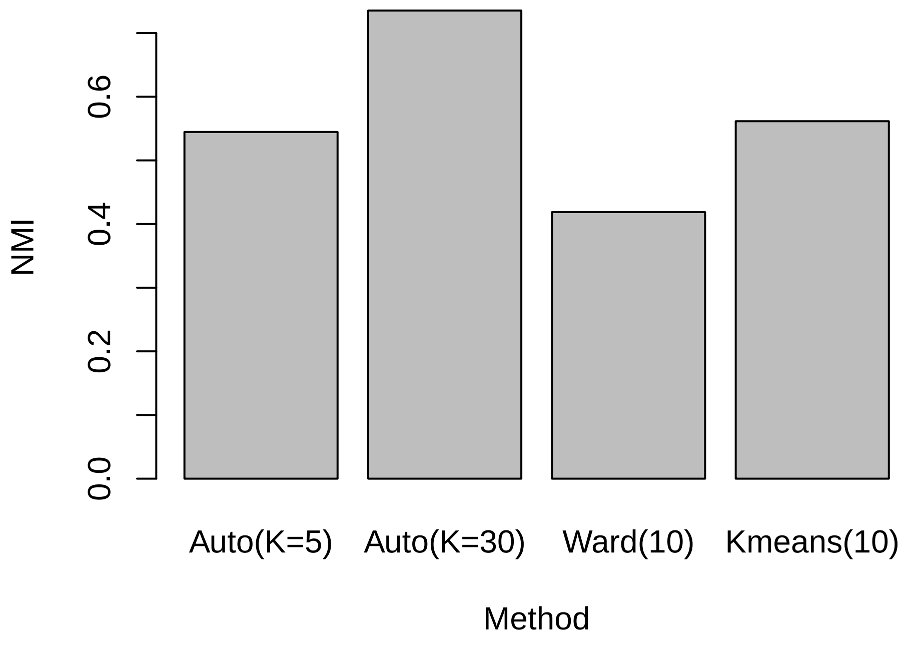


Figure 4.11: Comparison of the Normalized Mutual Information of semi-supervised and unsupervised method, Reuters dataset

4.3 Conclusions

In this chapter we have studied how the *instance level constraints* that can be applied to semi-supervised clustering can be automatically generated by studying the underlying nature of the data. The main advantage of this automatic semi-supervised approach is that it does not require some human expertise to provide external information.

By studying the nature of the data by means of the *k-means* algorithm it is possible to find relationships hidden in the data structure that provide very valuable information. As has been tested experimentally, this relationships may return better results than those obtained from class labels provided by an expert.

A very interesting point has arisen in this study, and it is the influence of the number of groups required from the *k-means* algorithm. As this parameter gets higher, more groups (smaller) are generated, so the constraints are more specific and has more mistakes. However, experiments have shown that there is an upper-limit in that parameter where there is no further improvement. So higher values of k are recommended.

In summary, for those datasets where the human expertise is not available, it is possible to use automatic generation of constraints by the study of the underlying nature of the data. Using this model it is possible to use the flexibility of some semi-supervised methods without requiring expert knowledge.

Chapter 5

Conclusions and Future Works

5.1 Conclusions

In this thesis, it has been our intention to approach the Authority Control problem from a Data Mining perspective. During the study and development of a methodology to solve this problem, several conclusions have arisen:

- Data Mining Methods, specifically, clustering techniques are a good tool to approach the Automatic Authority Control. Authority control can be viewed as the process of grouping records in a library by name, so the use of a technique to group elements is able to provide a good solution for the problem. In Chapter 2, we have proposed an automatic system to approach the problem from a clustering perspective. To do so, it has been necesary:

  – To provide a formal *intermediate representation* by considering the problem as a specific case of *Entity resolution*, understood as the problem of finding the different names of an entity in a text. Under this general formal model, an *authority* can be considered as an entity with different names, namely, the different representations of an author name.

  - Proposing a distance measure to compare the different elements of the problem. This measure is a combination of the different elements that are involved in the process. This point solves the problem of finding a suitable text distance measure needed in the clustering process. During the study of the different elements that should be compared to properly find the authorities, we discover that classic string comparison measures were not appropriate for names. For that reason, we have proposed *pn-measure* a new ad-hoc distance measure specific to compare personal names.

  - A validation method was proposed to find the quality of the authority control process. This method requires an expert who provides some ground truth about the obtained authorities allowing us to automatically calculate some indexes that characterize the results.

- The validation method proposed for the system requires human expertise. This means that there is a need for external information that indicate the correct solution. However, that information could be more useful than just the application to the evaluation of the results. Under that assumption, the use of semi-supervised clustering was the natural step to take. In chapter 3 we study these techniques getting to the following conclusions:

  - Among the possibilities offered by the semi-supervised clustering techniques, the use of *instance level constraints* is the best option for the kind of problems that we are trying to solve. This approach requires from the expert pairs of instances that should be placed together (or not) in the same cluster.

- Classical semi-supervised clustering approaches are based on the idea of modifying the distance measure used to compare elements. However, the distance measure selected for the problem seemed pretty reasonable and it would not be desirable to modify it. Additionally, we required the use of hierarchical clustering, as the number of groups is not known for the Automatic Authority Control problem.

- A new semi-supervised clustering algorithm is developed using the previous ideas:

  * Crisp Hierarchical Semi-Supervised algorithm, *Crisp HSS* is a new semi-supervised clustering algorithm that focuses on using external information in the process of finding the optimal partition in a process of hierarchical clustering.

  * Considering the inherent fuzziness of hierarchical clustering and the characteristics of the *Crisp HSS* algorithm, it seems reasonable to extend it to a fuzzy version. Fuzzy Hierarchical Semi-Supervised, *Fuzzy HSS*, is a fuzzy semi-supervised hierarchical clustering algorithm that proposes a new model of semi-supervision via fuzzy instance level constraints. These restrictions represent a degree of belief of two elements being (or not) in the same cluster, giving the expert more flexibility.

  * The performance of both algorithms have been tested showing similar results, meaning that it is possible to use the fuzzy version in those problems where the expert could not be sure of the provided information.

    * Due to the characteristics of the fuzzy information, we also provide a mechanism to determine how much information should be given by the expert. This is a very strong point of this algorithm as normally semi-supervised clustering uses as much information as it is available, but using this model that is not necessary.

- The new semi-supervised algorithms, *Crisp HSS* and *Fuzzy HSS* can be applied successfully to the Authority Control.

- As an extension of the Crisp HSS method, the automatic generation of constraints has been studied. By repeatedly applying the K-means algorithm for randomly initialized clusters, it is possible to find relationships between elements that define instance level constraints.

- This extension has been tested successfully in a general document environment showing that automatically generated constraints can be used in those problems where the the human expertise is not available.

5.2 Future works

The topics covered in this thesis can be extended in several ways.

The theoretical model introduced in Chapter 2 to approach the Automatic Authority Control as a generalization of the *Entity resolution* problem can be applied to several other areas. It could be a very interesting future work to use it for other kind of problems with textual data, like recognition of events or celebrity names in blogs or news, company names from financial information, among many others. Additionally the study of the Automatic Authority Control under other kind of entries different than names is a worthy possibility to be studied.

It is also compelling to study the application of the two semi-supervised hierarchical clustering algorithms proposed in Chapter 3, *Crisp HSS* and *Fuzzy HSS* to other kind of problems. They would be very suitable for problems like finding topics in Social Networks.

The study of the introduction of fuzzy instance level constraints in other kind of non-hierarchical clustering models also remains as a future work. They introduce a very interesting novel approach that could benefit other methodologies.

Finally, the automatic generation of the instance level constraints can be extended to generate fuzzy constraints. As the fuzzy constraints provide a mechanism to determine how much external information should be applied to the process, having a mechanism to generate them automatically should be a very interesting extension.

Bibliography

[1] E. Amigó, J. Gonzalo, J. Artiles, and F. Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12(4):461–486, July 2008.

[2] K. Bache and M. Lichman. *UCI Machine Learning Repository.* University of California, Irvine, School of Information and Computer Sciences, 2013.

[3] K. Bade and A. Nurnberger. Personalized Hierarchical Clustering. In *Web Intelligence, 2006. WI 2006. IEEE/WIC/ACM International Conference on*, pages 181–187, Dec. 2006.

[4] R. A. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[5] E. Bair. Semi-supervised clustering methods. *Wiley interdisciplinary reviews. Computational statistics*, 5(5):349–361, 2013.

[6] S. Basu, A. Banerjee, and R. J. Mooney. Semi-supervised Clustering by Seeding. In *Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, pages 27–34, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[7] S. Basu, A. Banjeree, E. Mooney, A. Banerjee, and R. J. Mooney. Active Semi-Supervision for Pairwise Constrained Clustering. In *In Proceedings of*

the 2004 SIAM International Conference on Data Mining (SDM-04, pages 333–344, 2004.

[8] S. Basu, M. Bilenko, and R. J. Mooney. A Probabilistic Framework for Semi-supervised Clustering. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 59–68, New York, NY, USA, 2004. ACM.

[9] S. Basu, I. Davidson, and K. Wagstaff. *Constrained Clustering: Advances in Algorithms, Theory, and Applications.* CRC Press, Aug. 2008.

[10] A. M. Bensaid and J. C. Bezdek. Semi-Supervised Point Prototype Clustering. *International Journal of Pattern Recognition and Artificial Intelligence*, 12(05):625–643, 1998.

[11] A. M. Bensaid, T. L. O. Hall, J. C. Bezdek, and L. P. Clarke. Partially Supervised Clustering for Image Segmentation. *Pattern Recognition*, pages 859–871, 1996.

[12] I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *ACM Trans. Knowl. Discov. Data*, 1(1), Mar. 2007.

[13] I. Bhattacharya, L. Getoor, and L. Licamele. Query-time entity resolution. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 529–534, Philadelphia, PA, USA, 2006. ACM.

[14] M. Bilenko, S. Basu, and R. J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 11–, Banff, Alberta, Canada, 2004. ACM.

[15] A. Bouchachia and W. Pedrycz. A Semi-supervised Clustering Algorithm for Data Exploration. In *Proceedings of the 10th International Fuzzy Systems Association World Congress Conference on Fuzzy Sets and Systems*, IFSA'03, pages 328–337, Berlin, Heidelberg, 2003. Springer-Verlag.

[16] A. Bouchachia and W. Pedrycz. Data Clustering with Partial Supervision. *Data Mining and Knowledge Discovery*, 12(1):47–78, 2006.

[17] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning.* MIT Press, Cambridge, MA, 2006.

[18] Y. Chen, L. Wang, and M. Dong. Semi-supervised Document Clustering with Simultaneous Text Representation and Categorization. In W. Buntine, M. Grobelnik, D. Mladenić, and J. Shawe-Taylor, editors, *Machine Learning and Knowledge Discovery in Databases*, number 5781 in Lecture Notes in Computer Science, pages 211–226. Springer Berlin Heidelberg, 2009.

[19] P. Christen. A Comparison of Personal Name Matching: Techniques and Practical Issues. In *Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops*, pages 290–294, Hong Kong, China, 2006. IEEE Computer Society.

[20] A. Chávez-Aragón, J. F. Ramirez Cruz, O. F. Reyes-Galaviz, H. Ayanegui-Santiago, and A. Portilla. An Algorithm to Tackle the Name Authority Control Problem Using Semantic Information. In *Proceedings of the 2009 Mexican International Conference on Computer Science*, ENC '09, pages 176–179, Mexico City, Mexico, 2009. IEEE Computer Society.

[21] W. W. Cohen, P. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IIWeb2003*, pages 73–78, Acapulco, Mexico, 2003.

[22] D. Cohn, R. Caruana, and A. Mccallum. Semi-supervised Clustering with User Feedback. Technical report, 2003.

[23] R. G. Cota, A. A. Ferreira, C. Nascimento, M. A. Gonçalves, and A. H. F. Laender. An unsupervised heuristic-based hierarchical method for name disambiguation in bibliographic citations. *Journal of the American Society for Information Science and Technology*, 61(9):1853–1870, 2010.

[24] I. Davidson and S. Ravi. Clustering with Constraints: Feasibility Issues and the k-Means Algorithm. 2005.

[25] I. Davidson and S. S. Ravi. The complexity of non-hierarchical clustering with instance and cluster level constraints. *Data Mining and Knowledge Discovery*, 14(1):25–61, Jan. 2007.

[26] I. Davidson and S. S. Ravi. Using instance-level constraints in agglomerative hierarchical clustering: theoretical and empirical results. *Data Min. Knowl. Discov.*, 18(2):257–282, 2009.

[27] I. Davidson, K. L. Wagstaff, and S. Basu. Measuring Constraint-Set Utility for Partitional Clustering Algorithms. In J. Fürnkranz, T. Scheffer, and M. Spiliopoulou, editors, *Knowledge Discovery in Databases: PKDD 2006*, number 4213 in Lecture Notes in Computer Science, pages 115–126. Springer Berlin Heidelberg, 2006.

[28] M. Delgado, A. F. Gomez-Skarmeta, and M. A. Vila. On the use of hierarchical clustering in fuzzy modeling. *International Journal of Approximate Reasoning*, 14(4):237 – 257, 1996.

[29] I. Diaz-Valenzuela. *Control de Autoridades automatico en bibliotecas Web*. Master Thesis, University of Granada, Granada, Sept. 2011.

[30] I. Diaz-Valenzuela, V. Loia, M. J. Martin-Bautista, S. Senatore, and M. A. Vila. Automatic constraints generation for semisupervised clustering: experiences with documents classification. *Soft Computing*, pages 1–11, Mar. 2015.

[31] I. Diaz-Valenzuela, M. J. Martin-Bautista, and M. A. Vila. An automatic data mining authority control system: A first approach. In *Intelligent Systems Design and Applications (ISDA), 2010 10th International Conference on*, pages 569 –574, Cairo, Egypt, Dec. 2010.

[32] I. Diaz-Valenzuela, M. J. Martin-Bautista, and M. A. Vila. Un sistema automático de Control de Autoridades para Bibliotecas Digitales. In *ESTYLF 2012, XVI CONGRESO ESPAÑOL SOBRE TECNOLOGÍAS Y LÓGICA FUZZY*, pages 450–455, Valladolid, España, Feb. 2012.

[33] I. Diaz-Valenzuela, M. J. Martin-Bautista, and M.-A. Vila. A proposal for Automatic Authority Control in Digital Libraries. *Information Processing and Magnament*, (Submitted), 2013.

[34] I. Diaz-Valenzuela, M. J. Martin-Bautista, and M. A. Vila. Using a semisupervised fuzzy clustering process for identity identification in digital libraries. In *IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS), 2013*, pages 831–836, 2013.

[35] I. Diaz-Valenzuela, M. J. Martin-Bautista, and M.-A. Vila. A fuzzy semisupervised clustering method: application to the classification of scientific publications. In *IPMU 2014 15th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, volume Acepted, Montpelier, France, 2014.

[36] I. Diaz-Valenzuela, M. J. Martin-Bautista, and M. A. Vila. Un método de Agrupamiento Semisupervisado difuso: Aplicación al ámbito de la clasifiación de publicaciones científicas. pages 511–516, Zaragoza, Spain, Feb. 2014.

[37] I. Diaz-Valenzuela, M. J. Martin-Bautista, and M. A. Vila. On the use of Fuzzy Constraints in Semisupervised Clustering. *IEEE Transactions on Fuzzy Systems*, ((submitted)), 2015.

[38] I. Diaz-Valenzuela, M. J. Martin-Bautista, M. A. Vila, and J. R. Campaña. An automatic system for identifying authorities in digital libraries. *Expert Systems with Applications*, 40(10):3994 – 4002, 2013.

[39] E. Dimitriadou, S. Dolničar, and A. Weingessel. An examination of indexes for determining the number of clusters in binary data sets. *Psychometrika*, 67(1):137–159, 2002.

[40] H. Emptoz. Nonprobabilistic entropies and indetermination measures in the setting of fuzzy sets theory. *Fuzzy Sets and Systems*, 5(3):307 – 317, 1981.

[41] A. A. Ferreira, M. A. Gonçalves, J. M. Almeida, A. H. F. Laender, and A. Veloso. A tool for generating synthetic authorship records for evaluating author name disambiguation methods. *Information Sciences*, 206(0):42 – 62, 2012.

[42] B. Furrie. *Understanding MARC Bibliographic: Machine-Readable Cataloging.* 2010. Published: Electronic Resource.

[43] C.-F. Gao and X.-J. Wu. A new semi-supervised clustering algorithm with pairwise constraints by competitive agglomeration. *Applied Soft Computing*, 11(8):5281 – 5291, 2011.

[44] M. W. Ghikas, editor. *Authority control: the key to tomorrow's catalog: proceedings of the 1979 Library and Information Technology Association Institutes.* Oryx Press, 1979.

[45] C. L. Giles, H. Zha, and H. Han. Name disambiguation in author citations using a K-way spectral clustering method. *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, 0:334–343, 2005.

[46] S. Gradmann. rdfs:frbr: Towards an Implementation Model for Library Catalogs Using Semantic Web Technology. *Cataloging & Classification Quarterly*, 39(3):63 – 76, 2005.

[47] D. Greene and P. Cunningham. Constraint Selection by Committee: An Ensemble Approach to Identifying Informative Constraints for Semi-supervised Clustering. In J. N. Kok, J. Koronacki, R. L. d. Mantaras, S. Matwin, D. Mladenič, and A. Skowron, editors, *Machine Learning: ECML 2007*, number 4701 in Lecture Notes in Computer Science, pages 140–151. Springer Berlin Heidelberg, 2007.

[48] N. Grira, M. Crucianu, and N. Boujemaa. Unsupervised and Semi-supervised Clustering: a Brief Survey. In *in 'A Review of Machine Learning Techniques for Processing Multimedia Content', Report of the MUSCLE European Network of Excellence (FP6*, 2004.

[49] N. Grira, M. Crucianu, and N. Boujemaa. Semi-supervised image database categorization using pairwise constraints. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 3, pages III–1228–31, Sept. 2005.

[50] N. Grira, M. Crucianu, and N. Boujemaa. Active semi-supervised fuzzy clustering. *Pattern Recognition*, 41(5):1834 – 1844, 2008.

[51] N. Grira, M. Crucianu, and Nozha Boujemaa. Semi-Supervised Fuzzy Clustering with Pairwise-Constrained Competitive Agglomeration. In *Fuzzy Systems, 2005. FUZZ '05. The 14th IEEE International Conference on*, pages 867–872, May 2005.

[52] S. Guha, R. Rastogi, and K. Shim. Rock: A robust clustering algorithm for categorical attributes. *Information Systems*, 25(5):345 – 366, 2000.

[53] A. Gómez-Skarmeta, A. Bensaid, and N. Tazi. Data mining for text categorization with semi-supervised agglomerative hierarchical clustering. *International Journal of Intelligent Systems*, 15(7):633–646, 2000.

[54] Y. Hamasuma, Y. Endo, and S. Miyamoto. On agglomerative hierarchical clustering using clusterwise tolerance based pairwise constraints. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 16(1):174–179, 2012.

[55] H. Han, L. Giles, H. Zha, C. Li, and K. Tsioutsiouliklis. Two supervised learning approaches for name disambiguation in author citations. In *Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '04, pages 296–305, Tuscon, AZ, USA, 2004. ACM.

[56] A. Hardy. An examination of procedures for determining the number of clusters in a data set. In *New approaches in classification and data analysis*, pages 178–185. Springer, 1994.

[57] Y. Hong, B.-W. On, and D. Lee. System Support for Name Authority Control Problem in Digital Libraries: OpenDBLP Approach. In R. Heery and L. Lyon, editors, *Research and Advanced Technology for Digital Libraries*, number 3232 in Lecture Notes in Computer Science, pages 134–144. Springer Berlin Heidelberg, 2004.

[58] J. Huang, S. Ertekin, and C. L. Giles. Efficient name disambiguation for large-scale databases. In *PKDD*, pages 536–544. Springer-Verlag, 2006.

[59] R. Huang, W. Lam, and Z. Zhang. Active Learning of Constraints for Semi-supervised Text Clustering. In *Proceedings of the 2007 SIAM International*

Conference on Data Mining, Proceedings, pages 113–124. Society for Industrial and Applied Mathematics, 2007.

[60] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, Dec. 1985.

[61] A. K. Jain. Data Clustering: 50 Years Beyond K-means. *Pattern Recogn. Lett.*, 31(8):651–666, June 2010.

[62] A. K. Jain and R. C. Dubes. *Algorithms for clustering data.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[63] M. A. Jaro. Probabilistic linkage of large public health data files. *Statistics in Medicine*, 14(5-7):491–498, 1995.

[64] D. V. Kalashnikov and S. Mehrotra. Domain-independent data cleaning via analysis of entity-relationship graph. *ACM Trans. Database Syst.*, 31(2):716–767, June 2006.

[65] A. Kaufmann. *Introduction to the Theory of Fuzzy Subsets.* Academic Pr, 1975.

[66] J. Kleinberg and E. Tardos. Approximation Algorithms for Classification Problems with Pairwise Relationships: Metric Labeling and Markov Random Fields. *J. ACM*, 49(5):616–639, Sept. 2002.

[67] J. Knopfmacher. On measures of fuzziness. *Journal of Mathematical Analysis and Applications*, 49(3):529–534, 1975.

[68] S. R. Kruk, M. Synak, and K. Zimmermann. MarcOnt: integration ontology for bibliographic description formats. In *Proceedings of the 2005 international conference on Dublin Core and metadata applications: vocabularies in practice*, pages 31:1–31:5, Madrid, Spain, 2005. Dublin Core Metadata Initiative.

[69] B. Kulis, S. Basu, I. Dhillon, and R. Mooney. Semi-supervised graph clustering: a kernel approach. *Machine Learning*, 74(1):1–22, Sept. 2008.

[70] J. Kurki and E. Hyvönen. Authority Control of People and Organizations on the Semantic Web. In *Proceedings of the International Conferences on Digital Libraries and the Semantic Web*, Trento, Italy, 2009.

[71] N. Labzour, A. Bensaid, and J. Bezdek. Improved semi-supervised point-prototype clustering algorithms. In *, The 1998 IEEE International Conference on Fuzzy Systems Proceedings, 1998. IEEE World Congress on Computational Intelligence*, volume 2, pages 1383–1387 vol.2, May 1998.

[72] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. In *USENIX Technical Conference*, pages 707 – 710, 1965.

[73] D. Lewis. Reuters-21578, 1987.

[74] K. Li and Y. Zhou. *An Improved Semi-supervised Fuzzy Clustering Algorithm.* 2013.

[75] H. Liu and S.-t. Huang. Evolutionary semi-supervised fuzzy clustering. *Pattern Recognition Letters*, 24(16):3105 – 3113, 2003.

[76] A. D. Luca and S. Termini. A definition of a nonprobabilistic entropy in the setting of fuzzy sets theory. *Information and Control*, 20(4):301 – 312, 1972.

[77] H. P. Luhn. A statistical approach to mechanized encoding and searching of literary information. *IBM J. Res. Dev.*, 1(4):309–317, Oct. 1957.

[78] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval.* Cambridge University Press, 2008.

[79] I. A. Maraziotis. A semi-supervised fuzzy clustering algorithm applied to gene expression data. *Pattern Recognition*, 45(1):637 – 648, 2012.

[80] G. Milligan and M. Cooper. An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 50(2):159–179, 1985.

[81] S. Miyamoto and A. Terami. Semi-supervised agglomerative hierarchical clustering algorithms with pairwise constraints. In *2010 IEEE International Conference on Fuzzy Systems (FUZZ)*, pages 1–6, July 2010.

[82] S. Miyamoto and A. Terami. Constrained agglomerative hierarchical clustering algorithms with penalties. In *Fuzzy Systems (FUZZ), 2011 IEEE International Conference on*, pages 422–427, June 2011.

[83] R. Mojena. Hierarchical grouping methods and stopping rules: an evaluation. *The Computer Journal*, 20(4):359–363, Jan. 1977.

[84] K. Nigam, A. K. Mccallum, S. Thrun, and T. Mitchell. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning*, 39(2-3):103–134, May 2000.

[85] N. R. Pal and S. K. Pal. Higher order fuzzy entropy and hybrid entropy of a set. *Information Sciences*, 61(3):211 – 231, 1992.

[86] H. Pasula, B. Marthi, B. Milch, S. Russell, and I. Shpitser. Identity Uncertainty and Citation Matching. In *In NIPS*. MIT Press, 2003.

[87] W. Pedrycz. Algorithms of fuzzy clustering with partial supervision. *Pattern Recognition Letters*, 3(1):13–20, Jan. 1985.

[88] W. Pedrycz, V. Loia, and S. Senatore. P-FCM: a proximity—based fuzzy clustering. *Fuzzy Sets and Systems*, 148(1):21–41, Nov. 2004.

[89] W. Pedrycz, V. Loia, and S. Senatore. Fuzzy Clustering With Viewpoints. *Fuzzy Systems, IEEE Transactions on*, 18(2):274–284, Apr. 2010.

[90] W. Pedrycz and J. Waletzky. Fuzzy clustering with partial supervision. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 27(5):787–795, Sept. 1997.

[91] H.-T. Peng, C.-Y. Lu, W. Hsu, and J.-M. Ho. Disambiguating authors in citations on the web and authorship correlations. *Expert Systems with Applications*, 39(12):10521 – 10532, 2012.

[92] D. A. Pereira, B. Ribeiro-Neto, N. Ziviani, A. H. Laender, M. A. Gonçalves, and A. A. Ferreira. Using web information for author name disambiguation. In *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '09, pages 49 – 58, Austin, TX, USA, 2009. ACM.

[93] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi. Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-scale Data Collections. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 91–100, New York, NY, USA, 2008. ACM.

[94] M. Porter. Snowball Project, 2001.

[95] M. F. Porter. An algorithm for suffix stripping. pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1980.

[96] D. Pötschke. Aczél, J. / Daróezy, Z., On Measures of Information and Their Characterizations, New York-San Francisco-London. 1975. Academic Press. XII, 234 S., $ 24.50 (Mathematics in Science and Engineering 115). *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 58(12):581–581, 1978.

[97] R. Rogas. *Neural Networks: A Systematic Introduction.* Springer-Verlag, Berlin, 1996.

[98] C. Ruiz, M. Spiliopoulou, and E. M. Ruiz. Density-based semi-supervised clustering. *Data Min. Knowl. Discov.*, 21(3):345–370, 2010.

[99] C. Ruiz, C. G. Vallejo, M. Spiliopoulou, and E. Menasalvas. Automated Constraint Selection for Semi-supervised Clustering Algorithm. In P. Meseguer, L. Mandow, and R. M. Gasca, editors, *Current Topics in Artificial Intelligence*, number 5988 in Lecture Notes in Computer Science, pages 151–160. Springer Berlin Heidelberg, 2010.

[100] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, Inc., New York, NY, USA, 1986.

[101] J. Sinkkonen and S. Kaski. Clustering based on conditional distributions in an auxiliary space. *Neural Computation*, 14(1):217–239, Jan. 2002.

[102] M. M. M. Snyman and M. J. van Rensburg. Revolutionizing Name Authority Control. In *Proceedings of the Fifth ACM Conference on Digital Libraries*, DL '00, pages 185–194, New York, NY, USA, 2000. ACM.

[103] Y. Song, S. Pan, S. Liu, F. Wei, M. X. Zhou, and W. Qian. Constrained Text Coclustering with Supervised and Unsupervised Constraints. *IEEE Transactions on Knowledge and Data Engineering*, 25(6):1227–1239, 2013.

[104] J. Tang, A. C. M. Fong, B. Wang, and J. Zhang. A Unified Probabilistic Framework for Name Disambiguation in Digital Library. *IEEE Trans. on Knowl. and Data Eng.*, 24(6):975–987, June 2012.

[105] W. Tang, H. Xiong, S. Zhong, and J. Wu. Enhancing Semi-supervised Clustering: A Feature Projection Perspective. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 707–716, New York, NY, USA, 2007. ACM.

[106] P. Treeratpituk and C. L. Giles. Disambiguating authors in academic publications using random forests. In *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '09, pages 39 – 48, Austin, TX, USA, 2009. ACM.

[107] K. Wagstaff, S. Basu, and I. Davidson. When Is Constrained Clustering Beneficial, and Why? In *Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA*. AAAI Press, 2006.

[108] K. Wagstaff and C. Cardie. Clustering with Instance-level Constraints. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 1103–1110, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[109] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained K-means Clustering with Background Knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 577–584, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[110] X. Wang, B. Qian, and I. Davidson. On constrained spectral clustering and its applications. *Data Mining and Knowledge Discovery*, 28(1):1–30, Sept. 2012.

[111] W. E. Winkler. The State of Record Linkage and Current Research Problems. Technical report, Statistical Research Division, U.S. Census Bureau, 1999.

[112] Y. Yan, L. Chen, and W.-C. Tjhi. Fuzzy semi-supervised co-clustering for text documents. *Fuzzy Sets and Systems*, 215:74–89, Mar. 2013.

[113] X. Yin, T. Shu, and Q. Huang. Semi-supervised fuzzy clustering with metric learning and entropy regularization. *Knowledge-Based Systems*, 35:304–311, Nov. 2012.

[114] D. Zhang, K. Tan, and S. Chen. Semi-supervised Kernel-Based Fuzzy C-Means. In N. R. Pal, N. Kasabov, R. K. Mudi, S. Pal, and S. K. Parui, editors, *Neural Information Processing*, number 3316 in Lecture Notes in Computer Science, pages 1229–1234. Springer Berlin Heidelberg, 2004.

[115] H. Zhao and Z. Qi. Hierarchical Agglomerative Clustering with Ordering Constraints. In *Proceedings of the 2010 Third International Conference on Knowledge Discovery and Data Mining*, WKDD '10, pages 195–199, Washington, DC, USA, 2010. IEEE Computer Society.

[116] J. Zhong, G. Dong, Y. Zhou, X. Li, L. Liu, and Q. Chen. A Semi-supervised Text Clustering Algorithm Based on Pairwise Constraints. *Journal of Information and Computational Science*, 8(6):951–960, 2011.

[117] S. Zhong. Semi-supervised model-based document clustering: A comparative study. *Machine Learning*, 65(1):3–29, Mar. 2006.