International Journal of Computational Intelligence Systems, Vol. 9, Supplement 1 (2016), 69-80

A View on Fuzzy Systems for Big Data: Progress and Opportunities

Alberto Fernández [1] Cristobal José Carmona [2] María José del Jesus [1] Francisco Herrera [3 4]

[1] *Department of Computer Science, University of Jaén, Jaén, 23071, Spain.*

E-mail: {alberto.fernandez,mjjesus}@ujaen.es

[2] *Department of Civil Engineering, University of Burgos Burgos, 09006, Spain*

E-mail: cjcarmona@ubu.es

[3] *Department of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology), University of Granada Granada, 18071, Spain*

[4] *Faculty of Computing and Information Technology, King Abdulaziz University Jeddah, Saudi Arabia*

E-mail: herrera@decsai.ugr.es

Received 8 February 2016

Accepted 12 March 2016

Abstract

Currently, we are witnessing a growing trend in the study and application of problems in the framework of Big Data. This is mainly due to the great advantages which come from the knowledge extraction from a high volume of information. For this reason, we observe a migration of the standard Data Mining systems towards a new functional paradigm that allows at working with Big Data. By means of the MapReduce model and its different extensions, scalability can be successfully addressed, while maintaining a good fault tolerance during the execution of the algorithms. Among the different approaches used in Data Mining, those models based on fuzzy systems stand out for many applications. Among their advantages, we must stress the use of a representation close to the natural language. Additionally, they use an inference model that allows a good adaptation to different scenarios, especially those with a given degree of uncertainty. Despite the success of this type of systems, their migration to the Big Data environment in the different learning areas is at a preliminary stage yet. In this paper, we will carry out an overview of the main existing proposals on the topic, analyzing the design of these models. Additionally, we will discuss those problems related to the data distribution and parallelization of the current algorithms, and also its relationship with the fuzzy representation of the information. Finally, we will provide our view on the expectations for the future in this framework according to the design of those methods based on fuzzy sets, as well as the open challenges on the topic.

Keywords: Big Data, Fuzzy Rule Based Classification Systems, Clustering, MapReduce, Hadoop, Spark, Flink

1. Introduction

In the last years, we are generating and storing a larger volume of data than that of the whole history[38,70]. Being able of discovering the information and knowledge "hidden" into these data, can allow a significant step forward for dozens of fields of application. Citing some interesting examples, we may refer to e-Sciences[30], Social Computing [65], and large-scale e-commerce[43,48], among others.

This new typology of problems in which both the Volume, Velocity, and Variety of the data has increased significantly, is what is known as Big Data[20,75]. In turn, the smart management of these data to extract rich and valuable information, is defined as Data Science[49,52]. This speciality includes several fields such as statistics, machine learning, data mining, artificial intelligence, and visualisation, amongst others. Hence, Big Data and Data Science are two terms with a high synergy between them[64].

New paradigms for the storage and parallel processing of the Big Data has been designed. The baseline of all functional models to work with Big Data is the MapReduce environment[14], being Hadoop its most relevant open source implementation [40]. The workflow of this paradigm implies that any algorithm must be divided into two main stages, Map and Reduce. The first one is devoted to split the data for processing, whereas the second collects and aggregates the results. By means of the codification of these simple functions, the computation is distributed automatically, in a transparent way to the user, also having a robust fault tolerance mechanism[20]. Classical methodologies need to be redesigned and adapted in order to be applied in this field[10,70].

Despite its good properties, the MapReduce programming model has also some drawbacks that are crucial for data scientists in data mining. We refer to the issue about the inefficient processing of iterative tasks, and the slow performance when combining multiple data sources[41,20]. For these reasons, some alternatives to the standard Hadoop-MapReduce framework have arisen. Specifically, we must stress Spark[71,72] and Flink[2], both of which are Apache projects.

Among all possible paradigms to address Data Mining problems, we focus our attention on those based on fuzzy sets and fuzzy logic[35]. This representation for the modeling of solutions, allows a simplicity and flexibility which makes them suitable for addressing complex problems. This can be achieved as they are able to cope with vague, imprecise or uncertain concepts that human beings use in their usual reasoning[51], as well as their interpretability based on the linguistic variables[22].

In the context of Big Data problems, the former properties make them a valuable tool for developing robust solutions, handling the variety and veracity inherent to the available data. Additionally, the use of fuzzy labels and their overlapping, allows a good coverage of the problem space. This last fact is especially significant for the MapReduce processing scheme. Specifically, we are referring to the case when the division of the original data into different chunks may cause a bias for the presence of small disjuncts[67], i.e. those disjuncts in the learned classifier that cover few training examples.

In this paper, our objective is to provide an analysis of fuzzy modeling for different tasks in Big Data problems, i.e. classification, clustering, subgroup discovery, and so on. Being a novel area of research, there are still few works on the topic. Therefore, it is important to understand the nature of Big Data and how this problem is currently addressed. In this sense, some criteria for the correct development of solutions under the umbrella of fuzzy sets must be established. This way, we can achieve the highest advantage in extending current approaches. In summary, we aim at defining the direction for the design of powerful algorithms based on fuzzy systems, and how the information extracted with these models can be useful for the experts.

In order to accomplish these goals, the remainder of the manuscript is organized as follows. Section 2 introduces the topic of Big Data, including the MapReduce standard procedure, as well as some recent extensions to work with iterative algorithms. Next, Section 3 presents the current state-of-the-art on fuzzy modeling for Big Data, describing how these methods are designed, and the inner problems associated to this framework. Next, in Section 4 we discuss the prospects on the topic, such as how to

develop robust methods to take the most advantage from this scenario. Finally, Section 5 summarizes and concludes the work.

2. Big data: Definition, MapReduce programming model, and New Technologies

In this section, we will first introduce some concepts on Big Data (Section 2.1). Then, we present an overview of the MapReduce programming model that supports scalability in data processing (Section 2.2). Finally, we will describe two novel programming frameworks developed as an alternative to the standard MapReduce, under the premise of solving the shortcomings of this model in certain scenarios (Section 2.3).

2.1. Introduction to Big Data

There is no commonly accepted definition for the Big Data term, but at least the key features involved in this topic are clear. In this context, data must be huge in Volume, high in Velocity, and diverse in Variety[10,20,34]. All these facts are known as the 3Vs model of Big Data, which are nowadays extended including up to 9V's, adding terms like Veracity, Value, Viability, and Visualization, amongst others[75].

The most significant issue related with Big Data analytics refers to those advantages, and also challenges, derived from collecting and processing vast amounts of data[38,47]. The final goal is to improve the productivity (in business) or obtain new scientific breakthroughs (in scientific disciplines). This can be achieved by the analysis of these large datasets for the development of custom models and algorithms in order to extract useful knowledge from it[70].

This situation has lead to the rise of the topic of Data Science[49,52]. As stated above, it can be defined as the process carried out to analyze and get insights from Big Data problems[64]. This implies a convergence of several fields into this framework, namely machine learning, predictive analysis, and statistics, among others.

To reach the desired level of scalability, new technology solutions have been developed. The whole architecture has been rebuild at the storage, processing, and database levels. The aim is to allow a data distribution among different computers, supporting parallel access from different disks to increase the speed ratio. The main difference for standard "grid-computing" approaches, is being able to provide a robust fault tolerance, as well as an efficient data combination from multiple sources.

Among all platforms solutions for Big Data purposes, we must stress the Apache's Hadoop open-source software [a][40]. It comprises a number of software tools, from the Hadoop Distributed File System (HDFS) to support the parallel data storage of the large files, HBase, a key-value store to provide online access using HDFS for its underlying storage component, and YARN (Yet Another Resource Negotiator), a cluster resource management system, which allows any distributed program to run on data in a Hadoop cluster[68].

2.2. MapReduce programming model

There are several tools that have been designed to address data processing in Big Data problems. Among them, the most popular one is the *MapReduce* distributed processing system[14,15]. Initially designed as a proprietary software from Google, *Hadoop* has implemented its own open source counterpart[40].

The MapReduce programming model simplify the massive parallel processing of large datasets by providing a design pattern that instructs algorithms to be expressed into two primary functions, which must be designed by users: Map and Reduce. In the first phase, "Map" is used for per-record computation, i.e. the input data is processed by this function to produce some intermediate results. Afterwards, these intermediate results will be fed to a second phase in a "Reduce" function, which aggregates the latter output and applies a given function for obtaining the final results.

The MapReduce model is defined with respect to an essential data structure known as <key,value>

[a] `http:http://hadoop.apache.org`

pair. The processed data, the intermediate and final results work in terms of <key,value> pairs. In this way, the *Map* and *Reduce* are defined as follows:

- **Map function**: the master node takes the input, divides it into several sub-problems and transfers them to the worker nodes. Next, each worker node processes its sub-problem and generates a result that is transmitted back to the master node. In terms of <key,value> pairs, the *Map* function receives a <key,value> pair as input and emits a set of intermediate <key,value> pairs as output. Then, these intermediate <key,value> pairs are automatically shuffled and ordered according to the intermediate key and will be the input to the *Reduce* function.
- **Reduce function**: the master node collects the answers of worker nodes and combines them in some way to form the final output of the method. Again, in terms of <key,value> pairs, the *Reduce* function obtains the intermediate <key,value> pairs produced in the previous phase and generates the corresponding <key,value> pair as the final output of the algorithm.

Fig. 1 illustrates a typical MapReduce program with its *Map* and *Reduce* steps. The terms k and v refer to the original key and value pair respectively; k' and v' are the intermediate <key,value> pair that is created after the *Map* step; and v'' is the final result of the algorithm.

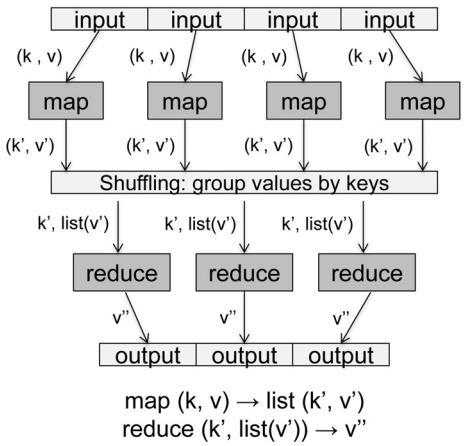

Fig. 1. The MapReduce programming model.

2.3. *Novel technological approaches for iterative processing*

The MapReduce framework has been established as a powerful solution in order to address Big Data Problems. However, its functional programming model is often restrictive under several scenarios. In particular, among all possible drawbacks of MapReduce, the greatest critic reported is the implementation of *iterative jobs*[42], due to the launching overhead when reloading from disk every time a job is executed.

For this reason, alternative solutions must be taken into account when developing machine learning algorithms, due to their implementation structure. Two novel programming models, Spark and Flink[59] have made available to achieve the highest performance in the scenario of Big Data. These are described in the remaining of this section.

2.3.1. *Spark*

Spark[71,72b] has been developed to overcome data reuse across multiple computations. It supports iterative applications, while retaining the scalability and fault tolerance of MapReduce, supporting in-memory processes. It is implemented in the functional programming language Scala and has further programming interfaces in Java, Python and the declarative query language SparkSQL.

Spark rests upon two main concepts: resilient distributed datasets (RDD), which hold the data objects in memory, and transformations or actions that are performed on the datasets in parallel.

RDDs are a collection of distributed items, with read-only-mode and fault-tolerant. As stated above, these data structures allow users making intermediate results to persist in memory, as well as controlling their partitioning to optimize data placement, and also manipulating them using a rich set of operators.

These operators are based on coarse-grained transformations. Specifically, there are 20 transformations and 12 actions usable. The difference between the two kinds of operators is that transformations are functions to generate and manipulate RDDs

[b] `http://spark.apache.org/`

out of input files or other RDDs, whereas actions can only be used on RDDs to produce a result set in the driver program or write data to the file system[71].

In order to manage data objects within the algorithms developed under Spark, two API are available in current version 1.6.0. On the first hand, DataFrame API, which is clearly the most stable option and therefore it offers the best performance. On the other hand, an API simply named as Dataset may be used also. However, being the most recent add to this framework, it does not yet leverage the additional information it has, and it can be slower than RDDs. It is expected to be fully developed and improved for Spark version 2.0.

It is worth to point out that Spark implements a Machine Learning library known as MLlib[c], included within the MLBase platform [39]. MLlib currently supports common types of machine learning problem settings, as well as associated tests and data generators. It includes binary classification, regression, clustering and collaborative filtering, as well as an underlying gradient descent optimization primitive. Finally, Mahout[50], maybe the most well-known Machine Learning library for Big Data, has recently announced its migration to Spark, due to the good properties of this framework for the execution of this type of algorithms. Neither of them support any fuzzy modeling algorithm yet.

2.3.2. *Flink*

The Flink framework[d] has its origins in the Stratosphere project [2,32]. The main idea behind this model is the optimization of execution plans, which is based on basic technologies from relational database systems. It is written in Java, but it also supports Scala and Python languages. It avoids the use of a declarative query language with aims at simplifying the development of applications for those users with less programming experience.

The Apache Flink framework currently provides 17 functions for data transformation, which are known as PACT operators, which are quite similar to that of the Spark framework. The main difference is based on two unique iteration operators, namely bulkIteration and deltaIteration. The former operator employ the whole dataset in every run, whereas the latter divides the data in a workset and solution set. One of the advantages of these iterators is that fewer data has to be computed and sent between nodes [19]. In accordance to the former, these are better suited for an efficient implementation of both machine learning and data mining algorithms, supporting also data stream processing.

As stated at the beginning, one of the most interesting features of the Flink framework is known as the PACT optimizer. It creates several semantically equivalent execution plans by reordering the PACT operators based on code analysis and data access conflicts[2,32]. The physical optimization chooses efficient strategies for data transport and operator execution on specific nodes. A cost-based strategy selects the most efficient plan based on criteria like network or storage I/O costs.

Finally, the Flink ecosystem also provides a Machine Learning library, FlinkML[f], which includes several data preprocessing, supervised learning, and recommender systems. It is still at a preliminary stage, but it is expected to add new algorithms in a short future.

3. Fuzzy Modeling in Big Data: Progress and Design

This section is devoted to present the current features related to fuzzy approaches in the scenario of Big Data. To this aim, we will first present those methodologies that have been already developed in this area of research (Section 3.1). Then, we will introduce several guidelines for the implementation of fuzzy models into a MapReduce framework (Section 3.2). Finally, we will point out some problems associated to the MapReduce execution style, and how the properties of fuzzy systems make them well suited to overcome them (Section 3.3).

[c] http://spark.apache.org/docs/latest/mllib-guide.html
[d] http://flink.apache.org/
[f] https://ci.apache.org/projects/flink/flink-docs-release-0.10/libs/ml/

3.1. What has been done?

The Big Data problem is a recent field of study, and therefore there are just few works which address this topic from the perspective of fuzzy modeling. Several proposals are mistakenly developed using the "Big Data buzzword," trying to take advantage of this hot topic. However, we must only focus on those procedures that are actually scalable by means of the proper mechanisms such as the MapReduce programming framework. Specifically, all approaches reviewed here are based on the Hadoop MapReduce implementation.

The highest effort has been carried out for clustering algorithms, especially for an scalable fuzzy c-means approach [5], but differences among these contributions are scarce. For example, in Ref. 46 the author study a MapReduce implementation using two jobs, one for the calculation of the centroids, and another for the membership matrix. The results in terms of purity shown by this model were comparable to other hard and fuzzy clustering techniques. Furthermore, a scalability analysis was conducted to demonstrate the performance of the parallel implementation with increasing number of computing nodes used. Another approach was based on an agglomerative implementation, using a novel initial center selection method for increasing the accuracy and convergence speed[74]. In Ref. 25, authors use Mahout [50] to study the performance of the fuzzy c-means implementation. Finally, in Ref. 28 the fuzzy clustering parallel implementation is applied to the organization of text documents.

In the context of classification tasks, the first Fuzzy Rule Based Classification System (FRBCS) adapted to the MapReduce scheme, named as Chi-FRBCS-BigData, was proposed in Ref. 56. As its name suggests, this method was based on the Chi et al.'s approach[11], which adapts its working procedure to follow a MapReduce scheme. The baseline fuzzy learning model had the advantage of providing fuzzy rules with same structure, and therefore to be independently created from a subset of examples. The Chi-FRBCS-BigData approach was based on two different MapReduce processes: the first one is devoted to the building of the fuzzy Knowledge Base from a Big Data training set; whereas the second procedure is aimed to carry the classification of the examples. Following the MapReduce paradigm, both schemes distribute the computations along several processing units that manage different chunks of information using Map functions. Then, the obtained results are simply aggregated within the Reduce functions.

An extension of this implementation was developed in Ref. 44. In this work, a cost-sensitive approach was derived from the original Chi-FRBCS-BigData approach in order to address classification with imbalanced datasets[45]. In this case, the computation of the rule weight was modified in order to take into account the cost of each class according to its representation in the training data. Considering the limitations of standard fuzzy learning approaches for high dimensionality and large number of instances, both approaches enable a good scalability, in terms of both execution time and accuracy, to millions of examples and half a hundred of attributes.

In Ref. 18 we may find the first MapReduce-based fuzzy rule based associative classifier. Standard associative rule mining algorithms are known to be of time complexity and with high memory constraints. This new model first extracts the frequent items; then, it generates the most significant classification association rules out of them by exploiting a parallel version of the well-known FP-Growth algorithm; finally, it prunes noisy and redundant rules by applying a parallel dataset coverage technique. In contrast with several state-of-the-art distributed learning algorithms, this proposed model achieves good results in terms of accuracy, model complexity and computation time.

In Ref. 54 a first approach for subgroup discovery tasks[7,8,29] was presented for the Big Data environment. In this contribution, a modification of the NMEEFSD algorithm[6] was implemented through the MapReduce paradigm in order to analyse huge quantity of information in a fast and efficient way. This preliminary version has been extended in Ref. 53. It modifies the design of the working MapReduce procedure in order to improve the influence of global information in the extraction process, taking advantage of the opportunities

that Spark provides to iterative jobs. The benefits of the NMEEFBD algorithm (NMEEFSD for Big Data) are contrasted by means of a great reduction of the total runtime, while maintaining the quality of subgroups.

3.2. How can it be designed?

In order to develop novel fuzzy models into the Big Data scenario, a thorough implementation of the Map and Reduce functions must be provided. The final goal is taking the highest advantage of the parallelization for both reducing the learning time costs and maintaining the overall accuracy. We must stress that this adaptation is not trivial, and requires some effort when determining how to proceed in a divide and conquer method from the original workflow.

First, we focus on a standard learning algorithm for fuzzy models with linguistic labels, although the guidelines given here can be easily extended to any other representation. There are two basic approaches that can be followed using the basic MapReduce programming model:

1. In the Map phase, "key-value" pairs are associated to the labels of the antecedent of the rules. In this case, the antecedent is used for joining rules in the Reduce phase.

2. "Key-value" pairs are linked to the whole rule antecedent for each Map process. Therefore, the Reduce stage is devoted to build an ensemble of all learned rules.

Second, we must take into account more sophisticated learning schemes that could require an iterative or a directed acyclic graph model. These cases implies one step forward in the design process for being able to provide an efficient scalable approach.

A clear example for the former is the iterative rule learning approach, implicit in several evolutionary fuzzy systems[13,12,21,24] . This scenario also includes boosting of fuzzy rules [16,58]. In both schemes, a single rule is created step by step and examples are then weighted or removed according to the coverage of the current rule base at that point.

The latter refers to those fuzzy rule learning algorithms that include a number of basic blocks in their workflow procedure. Specifically, we are addressing this scenario when the algorithm comprises different stages[1,31,36], or when rules are built using several partitions or models and then later aggregated.

The hitch in these cases is that MapReduce has a significant performance penalty when executing iterative jobs[20], as it reloads the input data from the disk every time, regardless of how much it has changed from the previous iterations. In accordance with the former, both the Spark and Flink programming frameworks must be considered as most productive tools. They allow more efficient implementations regarding their memory management and ad hoc operators to address both iterative and acyclic graph workflow jobs.

3.3. Associated problems of the MapReduce paradigm for classification

For every MapReduce procedure, original training data is divided into several chunks, each one with a disjunct subset of the examples. This issue implies several problems during the learning stage, that must be carefully addressed:

- A potential presence of small disjuncts[45,67] in the data associated to each Map as a consequence of the data division process. Small disjuncts, also known as "rare cases", are those disjuncts in the learned classifier that cover few training examples. Very specific rules must be created for these instances, which are discarded most of the times in favour of most general ones. On the other hand, if these rules are maintained in the Rule Base, the size of the former will be increased and the interpretability of the final model will be diminished.
- The skewed class distribution in classification problems[45], i.e. when one or several classes contains a lower number of examples than the ones of the other classes. The hitch in this case is that dividing the data intro smaller groups in the Map phase will cause minority class examples to be further underrepresented, and the output system will be biased.

- The lack of data in the training partitions[33,66], that causes a low density in the problem domain. It becomes very hard for the learning algorithm to obtain a model that is able to perform a good generalization when there is not enough data that represents the boundaries of the problem. As in the previous case, the division of the data into subsets for feeding the Map processes can accentuate this problem, especially in the case of those concepts with less representation in the training set. Additionally, the lack of data in the training data may also cause the introduction of small disjuncts.
- Finally, the locality of the data from which each "sub-model" is learned may has as consequence the generation of non-optimal systems. In such a case, when these models have opposite outputs at the classification stage, a proper combination of the former must be carried out to avoid erroneous results.

4. Discussion for Fuzzy Modeling in a Big Data Scenario

We must emphasize the necessity of working in the scenario of Big Data. Clearly, having a larger amount of data can allow researchers and corporations to extract better and more useful knowledge from the applications they are working with. By achieving higher quality results from data, implies a stronger support for taking future decisions on this context.

On the other hand, the significance of the use of fuzzy systems is straightforward. Their ability to provide a better representation of the problem space by means of fuzzy labels, makes them a very interesting approach when both the volume and variety of the dataset increases. Additionally, experts can benefit from the interpretability associated to linguistic labels.

New research in the Big Data scenario for fuzzy modeling must be focused on re-designing the state-of-the-art algorithms, as well as novel approaches for recent work scenarios. By doing so, a complete library of methods would be available for researchers and experts to take the full advantage of this novel work area. Additionally, this can be used as basis for the development of novel and more powerful fuzzy algorithms. However, the transition from standard learning approaches towards the MapReduce parallel model is not straightforward, and requires a thorough analysis of the designer.

In order to do so, we must focus our attention on a careful definition of the mechanisms of information combination. In this sense, the "Reduce" phase must be analyzed in detail. We must take into account the handicap of the division in the computation for different Maps, and to study the dependency of the models that have been learned in each one of the former. Determining an optimal combination of these models is a fundamental decision when applying a MapReduce scheme in Big Data Analytics. A smart combination of the systems learned within each Map must be carried out seeking for a good synergy, thus avoiding the local bias from the subsets they were learned with.

In particular, in Section 3.2 we referred to a two-fold type of models within the Map processes. Specifically, we can manage single rules, or a complete rule base depending on the case. In the first case, we should apply a fusion of the rules in the Reduce stage, whereas in the latter we will build an ensemble classifier[69]. However the performance in terms of accuracy and testing time of each solution will depend on the characteristics of the problem to solve, as well as on the requirements of the application. Additionally, when we are using approximate fuzzy models, the difficulties associated to a proper definition of the "key-value" pairs grows substantially if we aim at fusing or unifying the fuzzy rules into a single set.

We have also stressed the problem of the imbalanced distribution and the small disjuncts that emphasizes as the number of maps increases for a better scalability. We have referred to a first cost-sensitive model to address the class imbalance[44], but there is still much work to be carried out in this scenario. In this sense, studying the granularity level in the fuzzy labels can be a good choice in order to address the previous issues in a more sophisticated way.

Furthermore, and as stated in Section 3.2, both the Spark and Flink programming frameworks must be considered as productive Big Data technolo-

gies for the development of fuzzy learning algorithms, due to their good management of iterative and acyclic batch processing.

Finally, there are plenty of new and complex paradigms that are growing in importance in the last years that must be also taken into account. We may refer to multi-instance learning[17,62,63], multi-label [9,26,73], monotonic classification[27,37], semi-supervised learning [61], multi-view learning[60], social mining[4], and opinion mining [3,55], to enumerate the most relevant ones. Another source of research is the management of data streams [23,57] , where the time constraints and the need for an adaptive learning approach, imply a higher degree of difficulty. Regarding this last scenario, the use of Flink can allow an easier implementation as well as a robust environment for the efficient execution of the fuzzy algorithms.

5. Concluding Remarks

The features of fuzzy methods, i.e. their good management of uncertainty and noisy data, make them a valuable tool for Big Data tasks. However, their current development is still at a early stage, as there are just some few works on classification, association rule mining, subgroup discovery, and clustering. However, the initial results obtained implies the advantage of their use in this context. They allow at managing big datasets without damaging the classification accuracy and providing fast response times (increasing with the number of Maps). Additionally, the information in terms of interpretable systems can be very useful for experts. Fuzzy models provide a wide amount of advantages when applied to Big Data problems.

We described some of the main design principles for fuzzy approaches following the MapReduce functional programming model, being the standard in this framework. We have gone one step further, and we have carried out a brief discussion on the challenges on the topic. Among these prospects, we have emphasized the necessity of migrating the coding schemes towards two of the newest frameworks of big data analytics, i.e. Spark and Flink.

As future work on the topic, we have stressed several challenges to be taken into account, including the management of the Reduce stage, to address the imbalanced and small disjuncts problem related to the data division, as the application of fuzzy sets into different Big Data Science areas.

Acknowledgment

This work have been partially supported by the Spanish Ministry of Science and Technology under project TIN2014-57251-P; the Andalusian Research Plan P11-TIC-7765; and both the University of Jaén and Caja Rural Provincial de Jaén under project UJA2014/06/15.

References

1. J. Alcalá-Fdez, R. Alcalá, and F. Herrera. A fuzzy association rule-based classification model for high-dimensional problems with genetic rule selection and lateral tuning. *IEEE Transactions on Fuzzy Systems*, 19(5):857–872, 2011.
2. Alexander Alexandrov, Rico Bergmann, Stephan Ewen, Johann-Christoph Freytag, Fabian Hueske, Arvid Heise, Odej Kao, Marcus Leich, Ulf Leser, Volker Markl, Felix Naumann, Mathias Peters, Astrid Rheinländer, Matthias J. Sax, Sebastian Schelter, Mareike Höger, Kostas Tzoumas, and Daniel Warneke. The stratosphere platform for big data analytics. *International Journal on Very Large Databases*, 23(6):939–964, 2014.
3. Jorge A. Balazs and Juan D. Velasquez. Opinion mining and information fusion: A survey. *Information Fusion*, 27:95 – 110, 2016.
4. Gema Bello-Orgaz, Jason J. Jung, and David Camacho. Social big data: Recent achievements and new challenges. *Information Fusion*, 28:45 – 59, 2016.
5. James C. Bezdek. Fuzzy c-means cluster analysis. *Scholarpedia*, 6(7):2057, 2011.
6. C. J. Carmona, P. González, M. J. del Jesus, and F. Herrera. NMEEF-SD: Non-dominated Multiobjective Evolutionary algorithm for Extracting Fuzzy rules in Subgroup Discovery. *IEEE Transactions on Fuzzy Systems*, 18(5):958–970, 2010.
7. C.J. Carmona, P. González, M.J. del Jesus, and F. Herrera. Overview on evolutionary subgroup discovery: analysis of the suitability and potential of the search performed by evolutionary algorithms. *WIREs Data Mining and Knowledge Discovery*, 4(2):87–103, 2014.

8. Cristóbal J. Carmona, V. Ruiz-Rodado, María José del Jesús, A. Weber, M. Grootveld, Pedro González 0001, and D. Elizondo. A fuzzy genetic programming-based algorithm for subgroup discovery and the application to one problem of pathogenesis of acute sore throat conditions in humans. *Inormation. Sciences*, 298:180–197, 2015.
9. F. Charte, A.J. Rivera, M.J. del Jesus, and F. Herrera. Li-mlc: A label inference methodology for addressing high dimensionality in the label space for multi-label classification. *Neural Networks and Learning Systems, IEEE Transactions on*, 25(10):1842–1854, 2014.
10. C.L. Philip Chen and Chun-Yang Zhang. Data-intensive applications, challenges, techniques and technologies: A survey on big data. *Information Sciences*, 275:314–347, 2014.
11. Z. Chi, H. Yan, and T. Pham. *Fuzzy algorithms with applications to image processing and pattern recognition*. World Scientific, 1996.
12. O. Cordón, M.J. del Jesus, F. Herrera, and M. Lozano. Mogul: A methodology to obtain genetic fuzzy rule-based systems under the iterative rule learning approach. *International Journal of Intelligent Systems*, 14(11):1123–1153, 1999.
13. Oscar Cordon. A historical review of evolutionary learning methods for mamdani-type fuzzy rule-based systems: Designing interpretable genetic fuzzy systems. *International Journal of Approximate Reasoning*, 52(6):894–913, 2011.
14. Jeffrey Dean and Sanjay Ghemawat. MapReduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.
15. Jeffrey Dean and Sanjay Ghemawat. MapReduce: A flexible data processing tool. *Communications of the ACM*, 53(1):72–77, 2010.
16. María José del Jesús, Frank Hoffmann, Luis Junco Navascués, and Luciano Sánchez. Induction of fuzzy-rule-based classifiers with evolutionary boosting algorithms. *IEEE Transactions on Fuzzy Systems*, 12(3):296–308, 2004.
17. T. Dietterich, R. Lathrop, and T. Lozano-Perez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997.
18. Pietro Ducange, Francesco Marcelloni, and Armando Segatori. A mapreduce-based fuzzy associative classifier for big data. In Adnan Yazici, Nikhil R. Pal, Uzay Kaymak, Trevor Martin, Hisao Ishibuchi, Chin-Teng Lin, João M. C. Sousa, and Bülent Tütmez, editors, *FUZZ-IEEE*, pages 1–8. IEEE, 2015.
19. Stephan Ewen, Kostas Tzoumas, Moritz Kaufmann, and Volker Markl. Spinning fast iterative data flows. *PVLDB*, 5(11):1268–1279, 2012.
20. A. Fernández, S. Río, V. López, A. Bawakid, M.J. del Jesus, J.M. Benítez, and F. Herrera. Big data with cloud computing: An insight on the computing environment, mapreduce and programming framework. *WIREs Data Mining and Knowledge Discovery*, 4(5):380–409, 2014.
21. Alberto Fernandez, Victoria Lopez, Maria Jose del Jesus, and Francisco Herrera. Revisiting evolutionary fuzzy systems: Taxonomy, applications, new trends and challenges. *Knowlegde Based Systems*, 80:109–121, 2015.
22. Maria Jose Gacto, Rafael Alcalá, and Francisco Herrera. Interpretability of linguistic fuzzy rule-based systems: An overview of interpretability measures. *Information Sciences*, 181(20):4340–4360, 2011.
23. J. Gama and M. Gaber (Eds). *Learning from Data Streams – Processing techniques in Sensor Networks*. Springer, 2007.
24. David García, Antonio González, and Raúl Pérez. Overview of the slave learning algorithm: A review of its evolution and prospects. *International Journal of Intelligent Systems*, 7(6):1194–1221, 2014.
25. D. Garg and K. Trivedi. Fuzzy k-mean clustering in mapreduce on cloud based hadoop. In *2014 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*, pages 1607–1610, 2014.
26. Eva Gibaja and Sebastián Ventura. A tutorial on multi-label learning. *ACM Computing Surveys*, 47(3):52:1–52:38, 2015.
27. Sergio González, Francisco Herrera, and Salvador García. Monotonic random forest with an ensemble pruning mechanism based on the degree of monotonicity. *New Generation Computing*, 33(4):367–388, 2015.
28. Sumit Goswami and Mayank Singh Shishodia. A fuzzy based approach to text mining and document clustering. In *2013 International Conference on Computational and Information Sciences (ICCIS)*, 2013.
29. F. Herrera, C. J. Carmona, P. González, and M. J. del Jesus. An overview on Subgroup Discovery: Foundations and Applications. *Knowledge and Information Systems*, 29(3):495–525, 2011.
30. T Hey and A. E. Trefethen. The UK E-science core programme and the grid. *Future Generation Computer Systems*, 18(8):1017–1031, 2002.
31. J. Hühn and E. Hüllermeier. Furia: an algorithm for unordered fuzzy rule induction. *Data Mining and Knowledge Discovery*, 19(3):293–319, 2009.
32. Fabian Hueske, Mathias Peters, Matthias Sax, Astrid Rheinländer, Rico Bergmann, Aljoscha Krettek, and Kostas Tzoumas. Opening the black boxes in data flow optimization. *PVLDB*, 5:1256–1267, 2012.
33. T. Jo and N. Japkowicz. Class imbalances versus small disjuncts. *SIGKDD Explorations*, 6(1):40–49, 2004.
34. Karthik Kambatla, Giorgos Kollias, Vipin Kumar, and

Ananth Grama. Trends in big data analytics. *J. Parallel Distrib. Comput.*, 74(7):2561–2573, 2014.
35. G. Klir and B. Yuan. *Fuzzy sets and fuzzy logic: theory and applications*. Prentice-Hall, 1995.
36. Marcin Korytkowski, Leszek Rutkowski, and Rafal Scherer. Fast image classification by boosting fuzzy classifiers. *Information Sciences*, 327:175–182, 2016.
37. Wojciech Kotlowski and Roman Slowinski. On nonparametric ordinal classification with monotonicity constraints. *IEEE Transactions on Knowledge and Data Engineering*, 25(11):2576–2589, 2013.
38. T. Kraska. Finding the needle in the big data systems haystack. *IEEE Internet Comput.*, 17(1):84–86, 2013.
39. T. Kraska, A. Talwalkar, J.Duchi, R. Griffith, M. Franklin, and M.I. Jordan. Mlbase: A distributed machine learning system. In *Conference on Innovative Data Systems Research*, pages 1–7, 2013.
40. Chuck Lam. *Hadoop in action*. Manning, 1st edition, 2011.
41. K.-H. Lee, Y.-J. Lee, H. Choi, Y. D. Chung, and B. Moon. Parallel data processing with mapreduce: a survey. *SIGMOD Record*, 40(4):11–20, 2012.
42. Jimmy Lin. Mapreduce is good enough? *Big Data*, 1(1):BD28–BD37, 2013.
43. G. Linden, B. Smith, and J York. Amazon.com recommendations. item-to-item collaborative filtering. *IEEE Internet Comput.*, 7(1):76–80, 2003.
44. Victoria López, Sara del Río, José Manuel Benítez, and Francisco Herrera. Cost-sensitive linguistic fuzzy rule based classification systems under the mapreduce framework for imbalanced big data. *Fuzzy Sets and Systems*, 258:5–38, 2015.
45. Victoria Lopez, Alberto Fernandez, Salvador Garcia, Vasile Palade, and Francisco Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250(20):113–141, 2013.
46. Simone A. Ludwig. Mapreduce-based fuzzy c-means clustering algorithm: implementation and scalability. *Int. J. Machine Learning & Cybernetics*, 6(6):923–934, 2015.
47. V. Marx. The big challenges of big data. *Nature*, 498(7453):255–260, 2013.
48. E. W. T. Ngai and F. K. T. Wat. Literature review and classification of electronic commerce research. *Information & Management*, 39(5):415–429, 2002.
49. Cathy O'Neil and Rachel Schutt. *Doing Data Science*. O'Reilly Media, 2013.
50. Sean Owen, Robin Anil, Ted Dunning, and Ellen Friedman. *Mahout in Action*. Manning Publications Co., 2011.
51. W. Pedrycz and F. Gomide. *An Introduction to Fuzzy sets: Analysis and Design*. Prentice-Hall, 1998.
52. Foster Provost and Tom Fawcett. *Data Science for Business. What you need to know about data mining and data-analytic thinking*. O'Reilly Media, 1st edition, 2013.
53. F. Pulgar-Rubio, C. J. Carmona, M. D. Pérez-Godoy, A. J. Rivera-Rivas, P. González, and M. J. del Jesus. NMEEFBD: A MapReduce Solution for Subgroup Discovery in Big Data enviroments. *Knowledge-Based Systems*, Submited, 2016.
54. F. Pulgar-Rubio, C. J. Carmona, A. J. Rivera-Rivas, P. González, and M. J. del Jesus. Una primera aproximación al descubrimiento de subgrupos bajo el paradigma MapReduce. In *1er Workshop en Big Data y Análisis de Datos Escalable*, pages 991–1000, 2015.
55. Kumar Ravi and Vadlamani Ravi. A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. *Knowledge-Based Systems*, 89:14 – 46, 2015.
56. S. Río, V. López, J.M. Benítez, and F. Herrera. A mapreduce approach to address big data classification problems based on the fusion of linguistic fuzzy rules. *International Journal of Computational Intelligence Systems*, 8(3):422–437, 2015.
57. Andreu Sancho-Asensio, Albert Orriols-Puig, and Jorge Casillas. Evolving association streams. *Information Sciences*, 334-335:250 – 272, 2016.
58. Luciano Sánchez and José Otero. Boosting fuzzy rules in classification problems under single-winner inference. *International Journal of Intelligent Systems*, 22(9):1021–1034, 2007.
59. Norman Spangenberg, Martin Roth, and Bogdan Franczyk. Evaluating new approaches of big data analytics frameworks. In Witold Abramowicz, editor, *BIS*, volume 208 of *Lecture Notes in Business Information Processing*, pages 28–37. Springer, 2015.
60. Shiliang Sun. A survey of multi-view machine learning. *Neural Computing and Applications*, 23(7):2031–2038, 2013.
61. Isaac Triguero, Salvador García, and Francisco Herrera. Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. *Knowledge and Information Systems*, 42(2):245–284, 2013.
62. S. Vluymans, D. Sanchez-Tarrago, Y. Saeys, C. Cornelis, and F. Herrera. Fuzzy multi-instance classifiers. *IEEE Transactions on Fuzzy Systems*, in press, doi: 10.1109/TFUZZ.2016.2516582, 2016.
63. Sarah Vluymans, Danel Sanchez-Tarrago, Yvan Saeys, Chris Cornelis, and Francisco Herrera. Fuzzy rough classifiers for class imbalanced multi-instance data. *Pattern Recognition*, in press, doi: 10.1016/j.patcog.2015.12.002, 2016.
64. M.A. Waller and S.E. Fawcett. Data science, predictive analytics, and big data: a revolution that will transform supply chain design and management. *Journal of Business Logistics*, 34(2):77–84, 2013.
65. F.-Y. Wang, K. M. Carley, D. Zeng, and W. Mao.

Social computing: From social informatics to social intelligence. *IEEE Intelligent Systems*, 22(2):79–83, 2007.
66. M. Wasikowski and X.-W. Chen. Combating the small sample class imbalance problem using feature selection. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1388–1400, 2010.
67. Gary M. Weiss. The impact of small disjuncts on classifier learning. In Robert Stahlbock, Sven F. Crone, and Stefan Lessmann, editors, *Data Mining*, volume 8 of *Annals of Information Systems*, pages 193–226. Springer, 2010.
68. Tom White. *Hadoop: The Definitive Guide*. O'Reilly Media, 2nd edition, 2012.
69. Michael Wozniak, Manuel Graña, and Emilio Corchado. A survey of multiple classifier systems as hybrid systems. *Information Fusion*, 16:3 – 17, 2014.
70. Xindong Wu, Xingquan Zhu, Gong-Qing Wu, and Wei Ding. Data mining with big data. *IEEE Transactions on Knowledge Data Engineering*, 26(1):97–107, 2014.
71. Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J. Franklin, Scott Shenker, and Ion Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, pages 1–14, 2012.
72. Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. Spark: Cluster computing with working sets. In Erich M. Nahum and Dongyan Xu, editors, *HotCloud 2010*, pages 1–7. USENIX Association, 2010.
73. Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837, 2014.
74. Ruixin Zhang and Yinglin Wang. An enhanced agglomerative fuzzy k-means clustering method with mapreduce implementation on hadoop platform. In *2014 International Conference on Progress in Informatics and Computing (PIC)*, pages 509–513, 2014.
75. Paul C. Zikopoulos, Chris Eaton, Drik deRoos, Thomas Deutsch, and George Lapis. *Understanding Big Data - Analytics for Enterprise Class Hadoop and Streaming Data*. McGraw-Hill Osborne Media, 2011.