Expert Systems with Applications 42 (2015) 1436–1445

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

From data mining to knowledge mining: Application to intelligent agents

Amine Chemchem *, Habiba Drias *

USTHB-LRIA, BP 32 El Alia Bab Ezzouar, Algiers, Algeria

ARTICLE INFO

Article history:
Available online 10 September 2014

Keywords:
Knowledge mining
Induction rules
Classification
Clustering
Cognitive agent

ABSTRACT

The last decade, the computers world became a huge wave of data. Data mining tasks were invoked to tackle this problem in order to extract the interesting knowledge. The recent emergence of some data mining techniques provide also many interesting induction rules. So, it is judicious now to process these induction rules in order to extract some new strong patterns called meta-rules. This work explores this concept by proposing a new support for induction rules clustering and classification. The approach invokes k-means and k-nn algorithms to mine induction rules using new designed similarity measures and gravity center computation. The developed module have been implemented in the core of the cognitive agent, in order to speed up its reasoning. This new architecture called the Miner Intelligent Agent (MIA) is tested and evaluated on four public benchmarks that contain 25,000 rules, and finally it is compared to the classical one. As foreseeable, the MIA outperforms clearly the classical cognitive agent performances.

© 2014 Elsevier Ltd. All rights reserved.

1. Introduction

Nowadays, the induction rules have become inseparable pattern of the artificial intelligence thanks to their existence as the basis for many disciplines, such as the agent technology, data mining and knowledge discovery...This paper is about how to extend data mining techniques to induction rules in order to extract meta-rules. There are many data mining tasks for instance: clustering, classification, association rules mining, regression, prediction,...We are interested through this work in the first two tasks which are used on many applications (the image processing, the intrusion detection,...etc) and can be solved by different algorithms (k-means, HCA, fuzzy c-means...for clustering, KNN, SVM,ID3,...for classification). K-nn and K-means are in the top ten of data mining algorithms (Wu et al., 2008). The latter are extended to induction rules by introducing new version of similarity measure and gravity center computation. The algorithms called K-NN-IR and K-means-IR are developed and demonstrated on a public large scale benchmark including 25,000 induction rules. The whole idea behind this work is to improve the reasoning process by integrating the knowledge mining module in today's intelligent agent in order to speed up the reasoning engine process.

The rest of this paper is organized as follows: Next section shows a short history of data mining. Section 2 summarizes related works. In Section 3: induction rules representation are presented, followed by proposing mathematical preliminaries. In Section 5, the suggested algorithms are described and followed by the definition of a new architecture for intelligent agent. Then, experimental results are shown in Section 7 compared to the previously proposed algorithms. Finally we conclude by making some remarks and talking about future works.

2. Data mining overview

The generation of models from a large number of data is not a recent phenomenon. Egypt Pharaoh Amasis organizing the census of the population in the fifth century BC Rocchi (Rocchi, 2003). This is the seventeenth century we begin to analyze the data to find common characteristics. In 1662, John Graunt published his book "Natural and Political Observations Made upon the Bills of Mortality" in which he analyzed the mortality in London and trying to predict the appearances of the bubonic plague. In 1763, Thomas Bayes shows that we can determinate not only probabilities from observations derived from experience, but also the parameters for these probabilities. Legendre published in 1805 an essay on the least squares method for comparing a set of data with a mathematical model. From 1919 to 1925, Ronald Fisher develops the analysis of variance as a tool for its proposed medical statistical

* Corresponding authors.
E-mail addresses: aminechemchem@gmail.com (A. Chemchem), hdrias@usthb.dz (H. Drias).

http://dx.doi.org/10.1016/j.eswa.2014.08.024
0957-4174/© 2014 Elsevier Ltd. All rights reserved.

inference. The 1950s saw the advent of computer technology and computer calculation. Same methods and techniques are emerging such as segmentation, neural networks and genetic algorithms, and then in the 1960s, the decision tree, the method of mobile centers, these techniques allow researchers to exploit and discover models more accurate. The advent of the microcomputer stimulates research and statistical analyzes are more numerous and precise. The term "data mining" had a negative connotation in the early 1960s, expressing contempt for statisticians research approaches without correlation assumptions. it fell into oblivion, and Rakesh Agrawal employed again in the 80s when they were beginning research on databases with a volume of 1 Mb. The concept of data mining makes its appearance – according Pal (2007) – when the IJCAI[1] conferences took place in 1989. Then, in the 1990s, came the machine learning techniques such as SVM in 1998, complementing the tools of the data analysis. At the turn of the century, a company like Amazon uses these tools to offer our customers products that may interest. Actually, There are many tasks of data mining such as: Supervised and unsupervised classification, association rule mining, prediction and regression.

2.1. Supervised classification

Classification of a collection consists of dividing the items that make up the collection into categories or classes (Kotsiantis, 2007; Jain, Murty, & Flynn, 1999). In the context of data mining, classification is done using a model that is built on historical data. The goal of predictive classification is to accurately predict the target class for each record in new data, that is, data that is not in the historical data. A classification task begins with build data (also known as training data) for which the target values (or class assignments) are known. Different classification algorithms use different techniques for finding relations between the predictor attribute's values and the target attribute's values in the build data. K Nearest Neighbor (K-NN from short) is one of those algorithms that are very simple to understand, furthermore, it works incredibly well in practice, especially in the anomaly detection domain like Liao and Vemuri (2002), also for text categorization like in the work Guo, Wang, Bell, Bi, and Greer (2006). Also it is surprisingly versatile and its applications range from vision to proteins to computational geometry to graphs and so on. With KNN algorithm, we can obtain a satisfactory results, in addition, its basic principle is very simple, and easy to implement. It also might surprise many to know that K-NN is one of the top 10 data mining algorithms. K-NN is an non parametric learning algorithm, it is used when the data set does not obey a defined function as (gaussian mixtures, linearly separable etc). K-NN algorithm can be explained as follows, in the first time, training data that are already classified are considered, and then to classify the new data, we have to compute the similarities distance between this new data and all training data. After that the k nearest neighbors are extracted. In the end the new data is assigned to the most frequent class of these neighbors.

2.2. Clustering data technique

Clustering data mechanism consist to put the homogeneous data into the same group or class in order to dispatch the heterogeneous data into different groups. In the literature, it exists different manner to group the data, the two principals are: the hierarchical and the partitioning clustering. For the hierarchical clustering, the clusters are inside each others. This category of clustering is used when data can be separated in different levels. Also, CHA is the most known hierarchical algorithm, it starts by putting each instance in one cluster after that it computes the dissimilarities for all two instances to combine the clusters that have the lower distance. This process is repeated until we get one cluster (Steinbach, Ertöz, & Kumar, 2004; Han, Kamber, & Pei, 2006). In the contrary of the partitioning clustering, it consists to cluster the data separately. K-means is one of the simplest pure partitioning learning algorithms that solves the well known clustering problem (Han et al., 2006; MacQueen et al., 1967). The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed initially. The main idea is to define k gravity centers, one for each cluster. The centroids should be placed in a cunning way because the clustering result depends on their location in the clusters. In order to optimize the efficacy of the outcomes, it is judicious to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early grouping is done. At this point we need to recalculate k news centroids of the clusters resulting from the previous step, and iterates the process. The latter stops when no more changes of the clusters are observed, in other words when the centroids do not move any more.

3. Related works

Our interest in this study revolves around two main subjects: scalable cognitive agent and knowledge mining in general which involves induction rules mining. As for first subject, we found very few papers with ideas about the notion of scalable cognitive agent like Cao, Gorodetsky, and Mitkas (2009), and nothing about the paradigm that we would like to cover in this article. However, we notice that biologists and psychologists are showing interest in the study of scalable brain (Eliasmith, 2013). What can be said about the second topic is that the literature offers a large spectrum of detailed research on knowledge Mining. Mining knowledge including simple data and other patterns have been examined intensively over the last decade. In the following, we will talk about some knowledge mining.

Many works are about mining association rules in order to obtain meta rules whose purpose is to reduce the large number of discovered rules. The CLOSET algorithm was proposed in Strehl, Gupta, and Ghosh (1999) as a new efficient method for mining closed itemsets. CLOSET uses a novel frequent pattern tree (FP-tree) structure, which is a compressed representation of all the transactions in the database. Moreover, it uses a recursive divide-and-conquer and database projection approach to mine long patterns. Another solution for the reduction of the number is introduced by Hahsler and Chelluboina (2011) used an itemset-tid set search tree and pursued with the aim of generating a small non redundant rule set. To this goal, the authors first found minimal generator for closed itemsets, and then, they generated non redundant association rules using two closed itemsets. A new algorithm to group rules via hierarchical clustering has been developed in Berrado and Runger (2007) to visualize the large number of rules. The clustering of rules is done by defining a new distance called $d_{Jaccard}$ that represents the number of items of the two rules divided by the number of unique items. Saneifar, Bringay, Laurent, and Teisseire (2008) were interested in discovering sets of data. In their paper, they have developed a new similarity measure between two rules and extended k-means algorithm to cluster them. In literature some works about induction rules analysis have been proposed: In Poongothai and Sathiyabama (2012b), eh authors have developed a new algorithm to select the interesting induction rules from all the discovered rules in web mining

[1] International Joint Conference on Artificial Intelligence.

process. Another work Poongothai and Sathiyabama (2012a) was proposed in which a score function is used to evaluate such rules. Contrary to induction rules analyzing, there are not many works about induction rules mining. The latter consists in mining the induction rules in order to obtain meta knowledge. The first attempt was made in the previous works Drias, Aouichat, and Boutorh (2012), Chemchem, Drias, and Djenouri (2013) and Chemchem, Djenouri, and Drias (2013) where the concept of induction rules mining has been introduced for the first time. An interesting incremental clustering approach was proposed to address the problem however the design of both the distance between rules and the centroid is still not efficient enough. Recently, another previous work has been published (Chemchem et al., 2013) in order to cluster a big rule base using multilevel paradigm.

4. Induction rules representation

The web is featured by huge amounts of data bases from different fields. Applying data mining process to these data bases, many large knowledge bases too are extracted. It is very interested to deal this knowledge bases in order to discover a new hidden patterns that called "meta-knowledge". For this, a new concept of knowledge mining is necessary.

Before talking about knowledge mining, the knowledge representation must be presented first. Fig. 1 shows the various knowledge representation formalisms from the procedural form, which is rigid and well structured to the declarative form, which is on the contrary more open and free (Tuomi, 1999).

(1) The procedural approach: The procedural approach invokes the simplicity and the ease of understanding, represented by algorithms simulating real behaviors. In addition the procedural representation allows to treat problems with algorithmic style that is fully analyzable and understandable.
(2) The declarative approach: This approach is more flexible because it provides heuristic expressions using statements. The declarative representation allows to specify constraints and learn independently methods of use. The control structure is separated from the knowledge entered as rules on bulk data. The primary interest for this representation is modularity.

Furthermore, the procedural form is forced by a limited grammar for this, it is more interesting to deal the more declarative representation (natural language phrases). Nevertheless, natural language phrases grammar is very complicated, in other hand induction rules are the closest representation from natural language phrases, that is why this work is designed for induction rules mining.

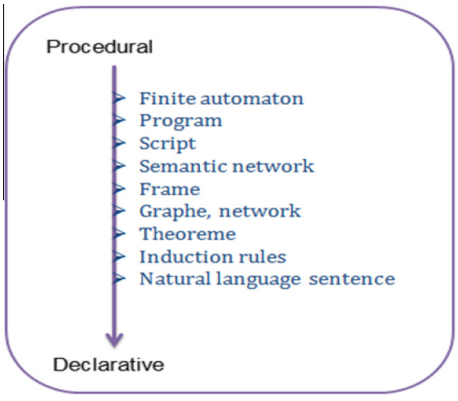

Fig. 1. Knowledge representation.

An induction rule is a Boolean formula of the form: $R : X \rightarrow Y$, where X and Y are sets of clauses. X is called the premise of the rule and Y its consequence (Grzymala-Busse, 1997).

The clause is a comparison between two elements as the form: a operator b; where $(a, b) \subset (A, V)/\ A$ is a set of variables, V is a set of values. (a, b) is an element of a Cartesian set A×V.

5. Mathematical preliminaries of induction rules mining

Defining the mathematical preliminaries of induction rules is a necessary step before the mining process. In this work we propose a new similarity measure and the gravity center computation formulas that will be used in their mining approaches.

5.1. Similarity measure

The similarity between two rules measures the degree of likeness between them. On the contrary, their dissimilarity denotes the degree of disparity. In Drias et al. (2012), the authors proposed the distance between two rules r1 and r2 as:

$D(r_1, r_2) = \frac{|C_1 \bigcup C_2| - |C_1 \bigcap C_2| + |V_1 \bigcup V_2| - |V_1 \bigcap V_2|}{|R_1 + R_2|}$.
where
C_i: represents the set clauses of R_i.
V_i: represents the set variables of R_i.

The distance D represents perfectly the similarity measure between two rules. Nevertheless, there is the supplementary operations such as $V_1 \bigcup V_2$ and $V_1 \bigcap V_2$, because the set V is included in the set C. Furthermore, $C_1 \bigcup C_2$ and $C_1 \bigcap C_2$ are already computed.

Intuitively, the rules are similar when they share a lot of identical components such as clauses. They are dissimilar if they are different from each other. For this purpose, We propose as a similarity measure noted *Dist_clauses* between two rules $R1$ and $R2$ which based only on the clauses sets the following formula:

$$Dist\_clauses(R1, R2) = Total\_clauses(R1, R2) - Shared\_clauses(R1, R2). \quad (1)$$

For example: Let consider two rules $R1$ and $R2$:
R1: IF (temperature = hot) AND (humidity = low) THEN (outlook = sunny),
R2: IF (outlook = sunny) AND (temperature = hot) AND (wind = light) THEN (play-tennis = no),
Clauses(R1) = {(temperature, hot), (humidity, low), (outlook, sunny)}.
Clauses(R2) = {(outlook, sunny), (temperature, hot), (wind, -light), (play_tennis, no)}.
Dist_clause(R1,R2) = Total_clause(R1,R2) − Shared_clause(R1, R2) = 5 − 2 = 3.

5.1.1. Analyze and demonstration

Let us consider the induction rules space noted IR that contain all possible rules.

To prove that Dist_clauses formula is a valid metric distance formula between two rules belonging to IR, the following proprieties should be demonstrated:

1. $\forall (r_1, r_2) \in IR^2, Dist\_clauses(r_1, r_2) \in IR$.
2. $\forall r \in IR, Dist\_clauses(r, r) = 0$.
3. $\forall (r_1, r_2) \in IR^2, Dist\_clauses(r_1, r_2) = Dist\_clausse(r_2, r_1)$.
4. $\forall (r_1, r_2, r_3) \in IR^3, Dist\_clauses(r_1, r_2) \leqslant Dist\_clausse(r_1, r_3) + Dist\_clauses(r_3, r_2)$.

Since, Clauses(r_i) is a set then:

$$Total\_clauses(r_1, r_2) = Clauses(r_1) \bigcup Clauses(r_2) \quad (1)$$

$$Shared\_clauses(r_1, r_2) = Clauses(r_1) \bigcap Clauses(r_2) \quad (2)$$

The following demonstration is based on Eq. (1) and Eq. (2).

Propriety 1. Dist_clause is decomposed on two part:

Total_clauses$(r_1, r_2) \in$ R and Shared_clauses$(r_1, r_2) \in$ R.

It is evident that Total_clauses − Shared_clauses ∈ R, so Dist_clauses ∈ R.

It implies that the first propriety is verified.

Propriety 2. Let be $r_1 \in$ IR:

$$\text{Dist\_clauses}(r_1, r_1) = Total\_clauses(r_1, r_1) - Shared\_clauses(r_1, r_1).$$

$$Total\_clauses(r_1, r_1) = clauses(r_1) \bigcup clauses(r_1).$$

$$\Rightarrow Total\_clauses(r_1, r_1) = clauses(r_1) \quad (3)$$

Furthermore;

$$\text{Shared\_clauses}(r_1, r_1) = clauses(r_1) \bigcap clauses(r_1).$$

$$\Rightarrow Shared\_clauses(r_1, r_1) = clauses(r_1) \quad (4)$$

By (3) and (4), it has:

Dist_clauses$(r_1, r_1) = 0$.

So, the second propriety is checked.

Propriety 3. Let us consider two rules $(r_1, r_2) \in IR^2$.

$$\text{Dist\_clauses}(r_1, r_2) = Total\_clauses(R1, R2) - Shared\_clauses(R1, R2).$$

$$\begin{aligned} \text{Total\_clauses}(r_1, r_1) &= clauses(r_1) \bigcup clauses(r_2) \\ &= clauses(r_2) \bigcup clauses(r_1). \\ &= \text{Total\_clauses}(r_2, r_1). \end{aligned}$$

$$\Rightarrow Total\_clauses(r_1, r_2) = Total\_clauses(r_2, r_1) \quad (5)$$

Furthermore;

$$\begin{aligned} \text{Shared\_clauses}(r_1, r_1) &= clauses(r_1) \bigcap clauses(r_1) \\ &= clauses(r_2) \bigcap clauses(r_1). \end{aligned}$$

$$\Rightarrow Shared\_clauses(r_1, r_2) = Shared\_clauses(r_1, r_2) \quad (6)$$

By (5) and (6), it has:

Dist_clauses$(r_1, r_2) = Dist_clauses(r_2, r_1)$.

⇒ the third propriety is verified.

Propriety 4. We pose:

$f(r_1, r_2) = Dist\_clauses(r_1, r_2)$. and $g(r_1, r_2, r_3) = Dist\_clauses(r_1, r_2) + Dist\_clauses(r_2, r_3)$.

If we can prove that: $Max(f(r_1, r_2)) \leqslant Min(g(r_1, r_2, r_3))$ then we conclude that:

$\forall (r_1, r_2, r_3) \in IR^3, Dist\_clauses(r_1, r_2) \leqslant Dist\_clausse(r_1, r_3) + Dist\_clauses(r_3, r_2)$.

In other term, f will be maximized and g will be minimized, if we find that f stay inferior to g, in this moment we can say that.

$\forall (r_1, r_2, r_3) \in IR^3, f(r_1, r_2) \leqslant g(r_1, r_2, r_3)$.

According to the number of clauses of r_1, and r_2, and r_3 respectively, we have six cases:

- Case 1: $clauses(r_2) \geqslant clauses(r_3) \geqslant clauses(r_1)$.
- Case 2: $clauses(r_2) \geqslant clauses(r_1) \geqslant clauses(r_3)$.
- Case 3: $clauses(r_3) \geqslant clauses(r_2) \geqslant clauses(r_1)$.
- Case 4: $clauses(r_3) \geqslant clauses(r_1) \geqslant clauses(r_2)$.
- Case 5: $clauses(r_1) \geqslant clauses(r_3) \geqslant clauses(r_2)$.
- Case 6: $clauses(r_1) \geqslant clauses(r_2) \geqslant clauses(r_3)$.

For each case, the following inequality should be demonstrated: $Max(f(r_1, r_2)) \leqslant Min(g(r_1, r_2, r_3))$.

Case 1: $clauses(r_2) \geqslant clauses(r_3) \geqslant clauses(r_1)$:

In one hand, Max(f) ⇒ Max(Total_clauses(r_1, r_2) and Min(Shared_clauses(r_1, r_2).

$$\text{So, it must be clauses}(r_1) \bigcap \text{clauses}(r_2) = \emptyset. \quad (7)$$

that implies:

$$\text{Total\_clauses}(r_1, r_2) = \text{clauses}(r_1) + \text{clauses}(r_2). \quad (8)$$

and

$$\text{Shared\_clauses}(r_1, r_2) = 0. \quad (9)$$

(8) and (9) give:

$$\text{Max}(f(r_1, r_2) = \text{clauses}(r_1) + \text{clauses}(r_2). \quad (10)$$

In other hand, Min(g) ⇒ Max(Shared_clauses(r_1, r_3), Max(Shared_clauses(r_3, r_2) and Min(Total_clauses(r_1, r_3), Min(Total_clauses(r_3, r_2)).

$$\text{Min(Total\_clauses}(r_1, r_3)) \Rightarrow \text{clauses}(r_1) \subset \text{clauses}(r_3). \quad (11)$$

So,

$$\text{Shared\_clauses}(r_1, r_3) = \text{clauses}(r_1). \quad (12)$$

To maximize Shared_clauses(r_3, r_2) and by (7), (11) it should be that: clauses $(r_3) \bigcap$ clauses$(r_2) =$ clauses$(r_3) \backslash$clauses(r_1).

That give us:

$$\text{Shared\_clauses}(r_3, r_2) = \text{clauses}(r_3) - \text{clauses}(r_1) \quad (13)$$

By hypothesis of case 1 it has:

$$\begin{aligned} \text{clauses}(r_3) > \text{clauses}(r_2) \Rightarrow \text{Min(Shared\_clauses}(r_3, r_2)) \\ = clauses(r_2). \end{aligned} \quad (14)$$

By (11)–(14) we obtain:

$$\text{Min(g)} = 2 * \text{clauses}(r_3) + \text{clauses}(r_2). \quad (15)$$

and

$$\text{clauses}(r_1) + \text{clauses}(r_2) \leqslant \text{clauses}(r_3) + \text{clauses}(r_2). \quad (16)$$

If we replace (16) in (15) and (10) we obtain:

Max(f) ⩽ Min(g). So,The first case is verified.

The other cases follow the same reasoning with a small modifications.

We have checked that the Dist_clauses respect the all proprieties of a standard distance. For consequent, we can say that Dist_clauses formula is a valid metric similarity measure between inductions rules.

5.2. Gravity center computation

The average of rules depends upon the similarity measure formula used. For example, for the previous measure, we were not interested in the average rule, but in the clauses of average rule. To cluster induction rules by k-means algorithm, we need a formula to compute a centroid. Conventional formulas used in the classical k-means cannot be applied here, since k-means manipulates data and not rules. For this reason, we propose three formulas based on propositional logic operators, to approximate the centroid of induction rules computation.

1. UOI (Union Over Intersection) Formula: Consists in calculating the union and the intersection of all clauses of *Ci* (a cluster of induction rules) separately, and then to subtract the intersection result from the union result.

Table 1
Table of distances.

Rules R_i	C_1	C_2	C_3	Min Distance
R_1	0	8	8	C_1
R_2	8	0	4	C_2
R_3	8	4	0	C_3
R_4	7	7	7	C_1

$clauses(center\ rule) = [clauses(r_1) \cup clauses(r_2) \cup \ldots \cup clauses(r_m)] - [clauses(r_1) \cap clauses(r_2) \cap \cdots \cap clauses(r_m)]$.

2. IUC (Intersection Union Center) Formula: In this case, we calculate the intersection of all clauses of *Ci* group rules, then we will do union between the obtained result and old gravity center of *Ci* as following:
   $clauses(center\ rule) = [clauses(r_1) \cap clauses(r_2) \cap \ldots \cap clauses(r_m)]\ rbrack \cup clauses(gi)$.
3. UIC (Union Intersection Center) Formula: We calculate the union of all clauses of *Ci* group rules then we will do intersection between the obtain result and oldster gravity center of *Ci* as following:
   $clauses(center\ rule) = [clauses(r1) \cup \ldots \cup clauses(rm)] \cap clauses(gi)$.

For example: Let consider the set clause rule_center of the previous iteration is $R1$, and three rules $R1, R2, R3$ as follows:

$R1$: if $x = 2$ and $y = 3$ and $z = 5$ then $t = 8$.
$R2$: if $x = 2$ and $y = 3$ and $z = 6$ then $t = 8$.
$R3$: if $y = 3$ and $z = 2$ then $t = 8$.
$clauses(R1) = \{(x,2),(y,3),(z,5),(t,8)\}$.
$clauses(R2) = \{(x,2),(y,3),(z,6),(t,8)\}$.
$clauses(R3) = \{(y,3),(z,2),(t,8)\}$.
$UOI(center\_rule) = [clauses(r_1) \cup clauses(r_2) \cup clauses(r_3)] - [clauses(r_1) \cap clauses(r_2) \cap clauses(r_3)] = \{(x,2),(z,5),(z,2),(z,6)\}$.
$IUC(center\_rule) = [clauses(r_1) \cap clauses(r_2) \cap clauses(r_3)] \cup clauses(gi) = \{(y,3),(t,8),(z,5),(x,2)\}$.
$UIC(center\_rule) = [clauses(r_1) \cup clauses(r_2) \cup clauses(r_3)] \cap clauses(gi) = \{(y,3),(t,8)\}$.

6. Induction rules mining tasks

From the preliminaries of induction rules presented in the previous section, the classical mining fields can be extended to deal with rules as follows:

6.1. Induction rules clustering

From the previous subsections, we can propose an adaptation of k-means algorithm to deal with induction rules noted K-means-IR as follows in Algorithm 1.

Algorithm 1. K-means-IR

Notation: G_i: gravity center of group i.
1-Choose k initials centers $C_1, C_2, \ldots, C_k$.
2- For each rule: affect it to group i that its gravity center is the nearest.
3- **If** (any group does not change) **then** stop and exit.
4- Calculate the new centers G_i for all C_i such as: G_i is the average of rules of the group C_i.
5- Go to 2.

For example: let consider this small induction rules set:

R1: If (temperature = hot) and (humidity = low) and (outlook = sunny) then (play_tennis = yes).

Table 2
Table of distances.

Rules R_i	C_1	C_2	C_3	Min Distance
R_1	3	8	8	C_1
R_2	11	0	4	C_2
R_3	11	4	0	C_3
R_4	4	7	7	C_1

R2: If (age = 29 years) and (income >= 3000) and (married = yes) then (children = 1).
R3: If (name = Casillas) and (age = 29 years) and (children = 1) then (team = Real_Mardrid).
R4: If (vehicle = classic) and (brake = true) then (danger_state = false).

By fixing the parameter k at 3 clusters the process of clustering is as follows:

Initially the 3 centroid are chosen randomly, for example: $C_1 = R_1, C_2 = R_2$, and $C_3 = R_3$ where C_i is the centroid of cluster i. and then we must to calculate the distances that separate each rules R_i to the centroid C_j as shown in Table 1:

After the affectation of each rules to the nearest centroid (as shown in the fifth column of Table 1), the new centroid are calculated:

$C_1 = \{(temperature, hot), (humidity = low), (outlook = sunny), (play\_tennis = yes), (vehicle = classic), (brake = true), (danger\_state = false)\}$.
$C_2 = R_2$.
$C_2 = R_2$.
$C_3 = R_3$.

The new centroid are different from the first ones, then we have to re-affect each rule to the nearest centroid as follows in Table 2:

From the fifth column of Table 2 we remark that the new centroid are the same of the last step, so the clustering process is stopped and the clusters are as follows:

$C1 = \{R_1, R_4\}$. $C2 = \{R_2\}$. $C3 = \{R_3\}$.

6.2. Induction rules classification

Similarly, The K-Nearest Neighbors algorithm known as K-NN is adapted to the supervised classification of induction rules noted K-NN-IR as explained in Algorithm 2.

Algorithm 2. K-NN-IR

1. **input:** a rule R and an integer k between 1 and the number of rules already classified.
2. **For** (each classified rule R_i) **do**
   Calculate the distance Dist(R, R_i).
   end for
3. - Select the k nearest neighbors of R;
4. **For** (each class) **do**
   Count the number of rules belonging to the k nearest neighbors of R
   end for
5. - Attribute to *R* the class that has the maximum count class.

For example: Let us consider the following small rules set already classified:

$R1 : If(temperature = hot)and(humidity = low)and(outlook = sunny)then(play\_tennis = yes) : class1$.

$R2 : If(temperature = hot)and(humidity = hight)and\ (outlook = sunny)then(play\_tennis = No) : class1$.

$R3 : If(age = 29years)and(income >= 3000)and(married = yes)\ then(children = 1) : class2$.

Table 3
Table of distances.

R_i	R_1	R_2	R_3	R_4	R_5	R_6
R_7	5	5	7	7	7	7

$R4 : If\,(age = 40years)and(income >= 3000)and\ \ (married = yes)\ then(children = 2) : class2.$

$R5 : If\,(name = Casillas)and(age = 29years)and(children = 1)then\ (team = Real\_Mardrid) : class3.$

$R6 : If\,(name = Rooney)and(age = 28years)and(children = 0)then\ (team = ManchesterUnited) : class3.$

And we must to classify a new Rule R7:

$R7 : If\,(outlook = sunny)and(wind = strong)then(plain\_travel = No).$

In the first step we must to calculate the distances that separate this new rule to each rule already classified, as shown in Table 3.

With fixing the parameter k to 3, the 3 nearest neighbors rules are chosen:

The 3 nearest neighbors $(R_7) = \{R_1, R_2, R_3\}$.

After the neighbors extraction step, the most frequent class for these neighbors is calculated:

$\{R_1$: class 1$\}$, $\{R_2$: class 1$\}$, $\{R_1$: class 3$\}$, we remark that the most frequent class is the class1, finally the new rule: R_7 is affected to the class 1.

7. The Miner Intelligent Agent

The cognitive agent is an autonomous software entity which develops its knowledge using knowledge discovery and engineering methods as well. It operates usually on using learning knowledge for solving problems. Given an order on different intelligence degrees, the cognitive agents can be classified into three categories:

- Cognitive agents based on expert systems. These agents develop their knowledge using an inference engine.
- Cognitive agents based on machine learning. These agents extract knowledge using machine learning techniques.
- Cognitive agents based on Data Mining. With this new trend, agents extract more efficiently knowledge because they rely on other more powerful methods of knowledge discovery (Shen, Hao, Yoon, & Norrie, 2006).

In this section we propose a new extension of intelligent agents called Miner Intelligent Agent or MIA for short. It is designed mainly by integrating the induction rules mining module in the agent architecture. This extension results consequently in a need to review the reasoning mechanism.

We define the MIA as a cognitive agent that has the ability to reason on a very large scale knowledge base. Therefore its design must rely not only on a knowledge based system but also on a scalability tool to manage the prohibitive base size.

This definition means that the difference between the ordinary cognitive agent (Wooldridge, Jennings, & Kinny, 2000) and the MIA resides in the capability to control extremely large amount of knowledge or not. As a consequence, to design a MIA, one approach could be to augment the ordinary cognitive agent architecture such as an expert system with a sophisticated tool to manage the scalability of the amount of knowledge. We can think in this case, in a first time about a strong clustering induction rules allowing the reduction of the tremendous knowledge volume to the agent scale. In the second time, and when the new fact comes from the agent environment, to classify them using the k-NN-IR algorithm in order to infer just the concerned ruled -induction rules belonging to the same class of the new fact- and not all the agent's rule base.

7.1. Architecture of the Miner Intelligent Agent

The general architecture of a MIA can be deduced from the definition and its analysis. Fig. 2 depicts the general framework of the miner intelligent architecture. It is composed by the following components:

- A huge knowledge base including induction rules and factual knowledge.
- An induction rules mining module to manage the scalability of the induction rules.
- A meta-knowledge base containing the clusters centers of the rule base computed by k-means-IR.
- An inference engine to allow the agent to reason on the meta-knowledge base.
- An interface allowing the agent to communicate with its environment.

- **The knowledge base** It contains all the knowledge perceived and developed by the agent. It contains induction rules of the form 'If condition Then action'. It also save facts interpreted from the environment or deduced by the agent through the inference engine.
- **The meta-knowledge base** It is structured as a set of reasonable size of knowledge bases relatively to the global amount of knowledge. In other terms, each cluster of the meta-knowledge

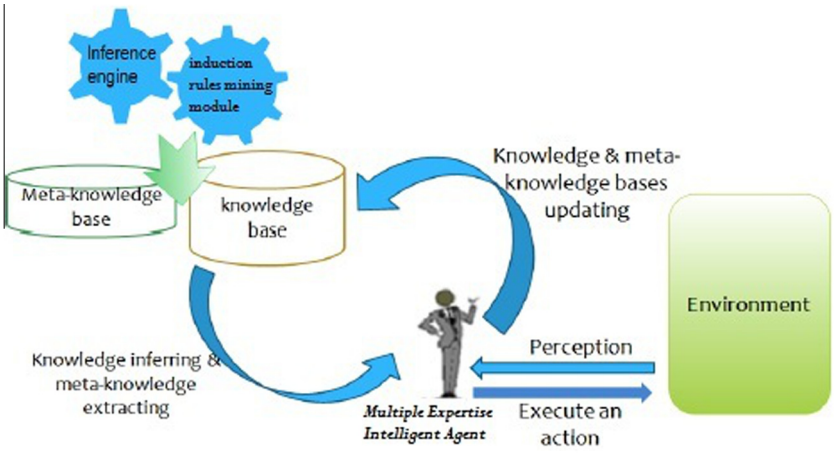

Fig. 2. The Miner Intelligent Agent architecture.

base constitutes a knowledge base and each meta-rules is the relationship between rules.

- **The Induction Rules Mining Module** The main role of this tool is to continuously cluster new knowledge, communicate the class that is supposed to be related to a given fact and finding the relationship between rules to accelerate the inference engine process. It is constituted by the two following components:
  - K-means-IR Algorithm: It clusters the rule base of the agent and communicates the results to the Meta-knowledge base.
  - K-NN-IR Algorithm: It consists to classify the new facts on the appropriate clusters on the meta-knowledge base.
- **The Inference Engine** The inference engine performs a forward or backward chaining on the rule base to ensure the reasoning process. Before starting an inference cycle, the agent has to localize according to the current fact the knowledge base of the whole meta-knowledge base on which it will operate. Of course the position of the concerned rules (the cluster of rules) is provided by the Induction Rules Mining Module. First it transforms the desired question to a rule form r without consequent part, then it classifies r in one of the clusters (finding by K-means-IR) of Meta-knowledge using K-NN-IR. Furthermore, it selects just the cluster concerned of the rules in order to infer them and finding a response to the new question.

For example: Let assume that a knowledge base of a Miner Intelligent Agent containing the following rules:

- R_1: IF (temperature = 20) And (outlook = sunny) Then (practice_sport = yes).
- R_2: IF (outlook = overcast) And (humidity = 80) Then (practice_sport = yes).
- R_3: IF (temperature = hot) And (wind = light) Then (practice_sport = no).
- R_4: IF (wind = light) And (outlook = rainy) Then (umbrella = yes).
- R_5: IF (engine = diesel) And (wheel = 4) Then (vehicle = yes).
- R_6: IF (wheel = 4) And (mark = audi) Then (vehicle = yes).

By applying k-means-IR on this small rule base, the meta_knowledge base could be as follows:

By K-means-IR algorithm, and (K = 3):

Cluster 1 = (R_1, R_2).

Cluster 2 = (R_3, R_4).

Cluster 3 = (R_5, R_6).

Now, we suppose that the MIA will infer how can we practice sport (yes or no), if the temperature is 30 and the humidity is 13, such as the new fact *ins* is represented as:

ins = if (temperature = 30) and (humidity = 13).

For answering to the question, the MIA uses the knowledge base and the meta-knowledge base to infer the class of this instance: First, it classes Ins using KNN-IR to the appropriate class. The result is the cluster 1. After that, it begins the inference on the closest rule to *ins* in the same cluster. Let we consider R_1 be this rule.

So, for inferring *ins* by the MIA, only R_1 and R_2 are used by the inference engine, on the contrary to the classical cognitive agent which will infer all the six rules of the rule base.

8. Evaluation and Experimentation

8.1. Data benchmark adaptation

In this part, we build our knowledge set from three public different benchmarks (Asuncion & Newman, 2007), as shown in Table 4. It includes in all 25,000 induction rules, all data sets are transformed to induction rule sets, as the following example:

If $(attribute_1 = value_1)$ and $(attribute_2 = value_2)$ and ...then $(attribute_n = value_n)$ where n is the last attribute of the data set.

The first one is originally a big data set known as: **Chess (King-Rook vs. King) Data Set**[2] It contains 28056 instances with 7 attributes.

The second benchmark is known as: **Abalone Data Set**,[3] it contains 4177 instances with 9 attributes.

The third data set is known as: **Car Evaluation Data Set**.[4] It contains 1728 instances, with 7 attributes.

Table 4
Benchmark construction.

Benchmark	Attributes	Data set size
Chess (King-Rook vs. King) Data Set	07	28056
Abalone Data Set	09	4177
Car Evaluation Data Set	07	1728

8.2. Evaluation pattern

Our benchmark as it is presented in the previous subsection(A), is built from three different benchmarks. To evaluate the clustering success rate, we fixed the parameter k of the clustering algorithms at 3. Then after the clustering step, the three obtained clusters are compared to the initial rule bases, if the clustering process is efficient, the respective clusters should be identical. The success rate is calculated using the following formula (Eq. (2)):

$$Success\_rate = \sum_{i=1}^{3} \frac{ncd_i}{npd_i}; ncd(KB_i) = max\left(\bigcap (KB_i, C_j)\right)/\forall j = 1 \ldots 3 \quad (2)$$

where:

ncri = number of correct rules from knowledge base i,

npri = number of pertinent rules from knowledge base i.

The number of correct rules of $KB_i = max$(number rules common(KB_i, C_j)) for all i and j in [1...3].

The number of pertinent rules of a $KB_i = The$ total number of its rules.

8.3. Induction rules clustering experimentations

Fig. 3 shows how the execution time augments with the increase of rules number for the k-means_IR using the three gravity center formulas (UCI, IUC, and UIC). According to this figure, UCI is slower than the two others. This is due to the fact that UCI uses three operators (Union Intersection and Over) between the rules and it requires a high complexity for computing the centroid. Furthermore, UIC is more efficient in CPU time than IUC. We explain this by the fact that union operator is faster than the intersection operator. In UIC, the union operator is applied on the rules of the same cluster after the intersection is performed only in the union rule and the previous centroid. However, in IUC the intersection is applied on all rules of the same cluster. Execution time of UIC does not exceed 2200 ms, when the number of rules is 25,000 rules, even though UCI execution time is 2600 ms when the number rules is 25,000.

Fig. 4 shows the success rate, computed by the Eq. 2, with the increase of rules number for the K-means_IR using the three centroid computation formulas. According to this figure we remark

[2] http://archive.ics.uci.edu/ml/datasets/Chess+%28King-Rook+vs.+King%29
[3] http://archive.ics.uci.edu/ml/datasets/Abalone
[4] http://archive.ics.uci.edu/ml/datasets/Car+Evaluation

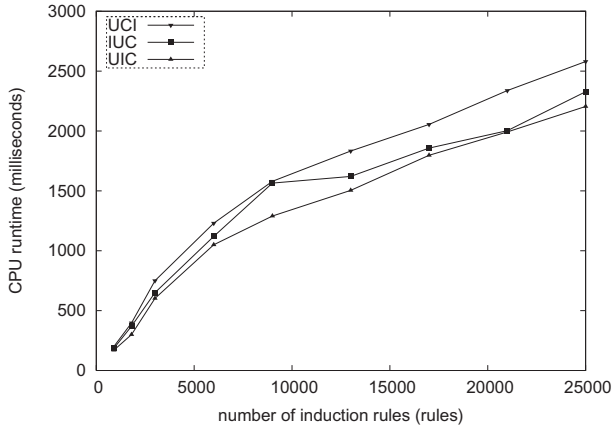

Fig. 3. CPU time for the three gravity center formulas.

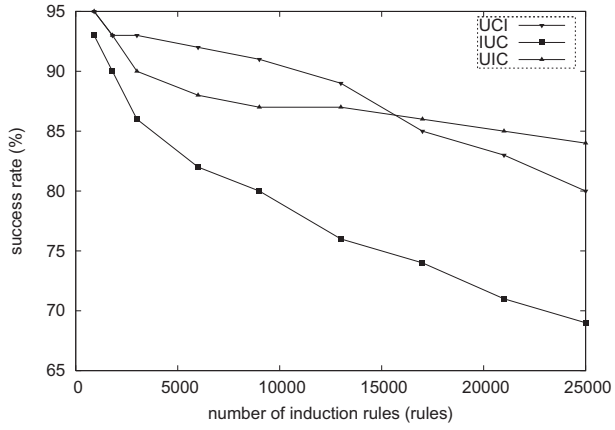

Fig. 4. Success rate for the k-means_IRC with the three gravity center formulas.

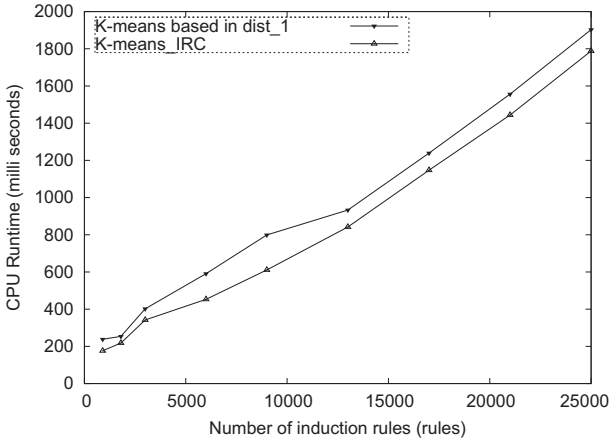

Fig. 5. CPU time for the two approaches.

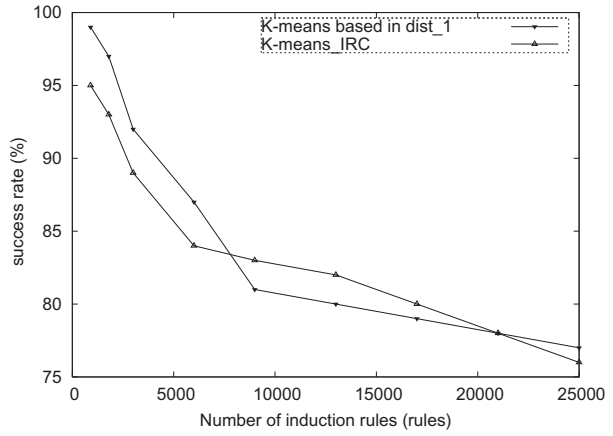

Fig. 6. Success rate for the two approaches.

that IUC is less efficient than the two other formulas, so when increasing the number of rules from 1000 to 25,000 rules, IUC success rate is reduced from 93%, to 69%, even though UIC is reduced just from 95%, to 84%.

From these results, it can be noted that the k-means IRC performance (execution time and success rate) depends on the chosen centroid formulas. So if we want to obtain the best performances on both criteria: success rate and CPU time, UIC should be applied, or UCI too, because it takes almost the same results of the first formula. But we have to ignore IUC because it is less efficient than the two other formulas on both the criteria. This is why we apply our algorithm k-means_IR using the UIC formula, in order to compar it with one previous algorithm based on similarity measure DIST1 presented previously. When comparing k-means_IR using the UIC formula with k-means based on DIST1, on the cpu run time criterion, the results in Fig. 5 are obtained.

Fig. 5 shows how the execution time augments with the increase of rules number for the k-means_IR using UCI formula, compared to the k-means based on DIST1 presented previously. According to this figure, k-means_IR is faster than the other algorithm. This is due to the fact that union operator is faster than the intersection operator. In UIC, the union operator is applied on the rules of the same cluster after the intersection is performed only in the union rule and the previous centroid. However, in dist1 the intersection is applied on all rules of the same cluster two times. Execution time of k-means_IR does not exceed 1800 ms, when the number of rules is 25,000 rules, even though K-means based on dist1 execution time is 1915 ms when the number rules is 25,000.

Fig. 6 shows the success rate, computed by the Eq. (2), with the increase of rules number for the K-means_IR using UIC centroid computation formula and the previously presented algorithm based on dist1. According to this figure we remark that there is not a big difference between the success rate of the two approaches, so when increasing the number of rules from 1000 to 25,000 rules, k-means based on dist1 success rate is reduced from 99%, to 77%, even though k-means_IR is reduced just from 95%, to 76%.

When fixing the number of rules at 10,000, and increasing the number of clusters, the success rate for the two algorithms are calculated using the F-measure formula explained in Eq. (3):

$$F\_measure_{\beta} = \frac{(\beta^2 + 1) \cdot P \cdot R}{\beta^2 \cdot P + R} \tag{3}$$

$$P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN} \tag{4}$$

where P is the precision rate and R is the recall rate. TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives and FN is the number of false negatives.

The F-measure can be used to balance the contribution of false negatives by weighting recall through a parameter $\beta \geqslant 0$. In our case β is set to 0, $F_0 = P$. In other words, recall has no impact on the F-measure when $\beta = 0$, because increasing β allocates an increasing amount of weight to recall in the final F-measure.

The obtained results are schematized in Fig. 7.

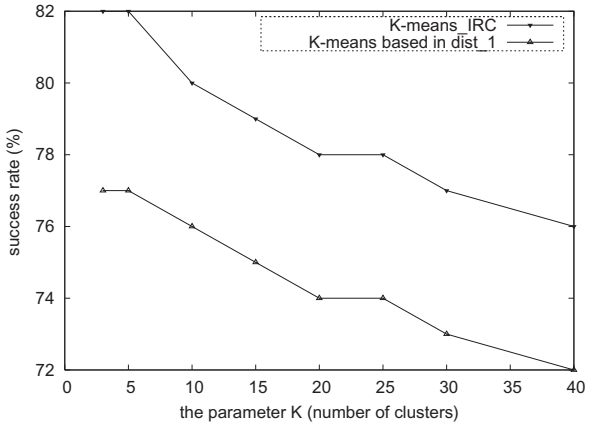

Fig. 7. F-measure comparison.

Fig. 7 shows the success rate, computed using Eq. (3), with the increase of the parameter K (number of clusters) for the K-means_IR using UIC centroid computation formula and the k-means algorithm based on dist1. According to this figure we remark that K-means_IR is more efficient than the first algorithm, so when increasing the number of clusters from 3 to 40clusters, k-means based on dist1 success rate is reduced from 77%, to 72%, even though k-means_IR is reduced just from 82%, to 76%.

8.4. Experimentation of KNN-IR

In this part, we give some experimentations of the KNN-IR algorithm based on the proposed distance formula using the benchmark described above. First, we labeled each rule base according to the domain type. Which each label represents the specified class. Then we divide the set of rules to two sets (training set and test set). The training set is considered as the input of KNN-IR. However, the test set is the set of rules that should be classified by KNN-IR. Moreover, the percentage of the correct rules can be computed as:

$$PCR = |CRC| \div |VS|.$$

where

PCR: represents the percentage of correct rules.

CRC: represents the rules classified correctly by applying KNN-IR.

VS: represents the test rules set.

Fig. 8 shows the number of correct rules changes according to the parameter K. In general, it exists a slight difference between the percentage proposal distance and dist1. Moreover dist1 and Dist_clauses converges to the extremum maximum (78% and 76.5%)

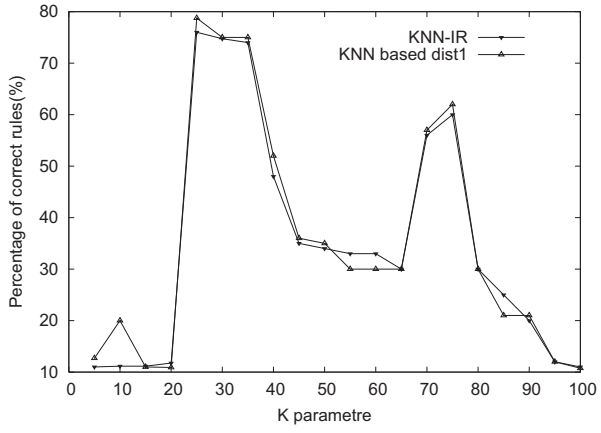

Fig. 8. Percentage of corrects rules according to the K parameter.

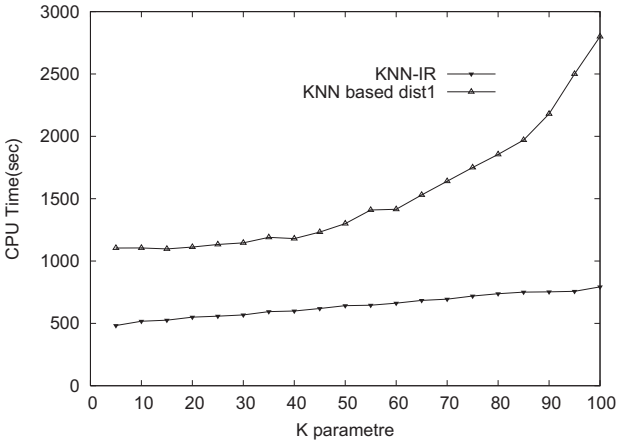

Fig. 9. CPU time according to the K parameter.

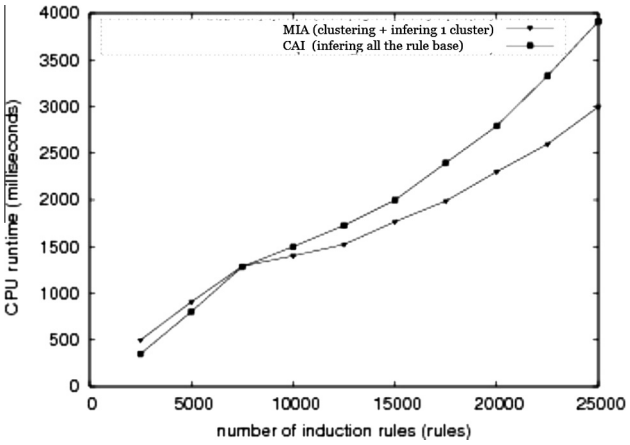

Fig. 10. CPU time comparison of the MIA to the classical agent.

respectively when it exists the number K is 25. After this result, we can say that the best value of K is 25 of the two distances. using the previous benchmark. Of course, by changing the benchmark, the *K* will be changed. Nevertheless, Fig. 9 prove the efficiency of our proposal formula. It outperforms dist1 w.r in CPU Time. By increasing the number of *K* from 5 to 100, the Dist_clauses's cpu time does not exceed 800 s, Or the dist1's cpu time is 1100 s when the number of K is 5 and exceeds 2700 s when the number of K is 100. Hence, we can say that Dist_clauses is more appropriate then dist1.

8.5. The comparison of the MIA to the classical agent

In this section, we compare the performance of the proposed architecture of multiple expertise intelligent agent to the classical intelligent agent. So, when the classical agent intelligent infers all the rule base at each arrival of a new fact, the multiple expertise agent intelligent clusters the rule base and infers just the cluster that contains the new fact. The obtained results are shown in Fig. 9.

The execution time for the two architectures of intelligent agent are compared in Fig. 10, when increasing the number of induction rules. We observe that the reasoning of classical agent is faster than the MIA while dealing with small rule base (less than 7500 rules), and we explain this by the fact that the MIA has an additional time of the clustering step. But while dealing with a large rule base, the MIA inferring is very fast in comparison to the classical agent inference. Moreover, when the classical agent deal with a number of induction rules varying between 2500 and 25,000, the execution time increases from 350 to 3912 ms whereas the

execution time of clustering and inference of the Miner Intelligent Agent increases just between 502 and 2997 ms.

When comparing the two architectures, we note that while dealing with small rule base; the classical agent architecture is faster than the proposed one. We explain that by the fact that the classic agent infers the small rule base immediately in order to extract the new rules. In the contrary to the MIA that starts with clustering the rule base, and then infer the cluster of rules that contain the new fact, that is why it takes more time than the simple agent while dealing with a small rule base.

But, while dealing with a big rule base (more than 7500), the MIA inference is faster than the classical inference engine, and this is due to the fact that the classical agent infers all the rule base. In the contrary to the MIA that select only the interesting rules using induction rules clustering, so it takes less time.

9. Conclusion

In this paper we presented a new paradigm of induction rules mining. Our work is focused on the study of the two fields: the clustering and the supervised classification. Also the algorithms Knn-IR and Kmeans-IR are proposed by incorporating new similarity measure and three gravity center computation formulas. We have tested and compared the proposed algorithms with other previous researches. The results are very satisfactory on both criteria: CPU run time and the success rate. Furthermore, we integrated the proposal support on the concept of intelligent agent. Hence the emergence of the new architecture of the intelligent agent: called (MIA) the Miner Intelligent Agent which allows to discover and infer the new knowledge very quickly.

As future works, we will try to implement a new platform for MIA called (Meta-Jad). The latter includes Meta-Jess to implement the new proposal motor engine.

This innovation will also provide the birth of a new agents more intelligent and faster than the classical ones, in order to reason on the huge knowledge bases of our real life. In addition, our new research will focus on the issue deducted from this work, like association rule mining between knowledge, and to create a new support of communication between these super intelligent agents which will communicate only the very important knowledge between them, instead of all basic data.

References

Asuncion, A., & Newman, D. (2007). Uci machine learning repository.

Berrado, A., & Runger, G. C. (2007). Using metarules to organize and group discovered association rules. *Data Mining and Knowledge Discovery, 14*(3), 409–431.

Cao, L., Gorodetsky, V., & Mitkas, P. A. (2009). Agent mining: The synergy of agents and data mining. *Intelligent Systems, IEEE, 24*(3), 64–72.

Chemchem, A., Djenouri, Y., & Drias, H. (2013). Incremental induction rules clustering. In *2013 8th International workshop on systems, signal processing and their applications (wosspa)* (pp. 492–497).

Chemchem, A., Drias, H., & Djenouri, Y. (2013). Multilevel clustering of induction rules for web meta-knowledge. In *Advances in information systems and technologies* (pp. 43–54). Springer.

Drias, H., Aouichat, A., & Boutorh, A. (2012). Towards incremental knowledge warehousing and mining. In *Distributed computing and artificial intelligence* (pp. 501–510). Springer.

Eliasmith, C. (2013). *How to build a brain: A neural architecture for biological cognition*. Oxford University Press.

Grzymala-Busse, J. W. (1997). A new version of the rule induction system lers. *Fundamenta Informaticae, 31*(1), 27–39.

Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2006). Using knn model for automatic text categorization. *Soft Computing, 10*(5), 423–430.

Hahsler, M., Chelluboina, S., 2011. Visualizing association rules in hierarchical groups. In *42nd Symposium on the interface: Statistical, machine learning, and visualization algorithms (interface 2011)*. The Interface Foundation of North America.

Han, J., Kamber, M., & Pei, J. (2006). *Data mining: Concepts and techniques*. Morgan Kaufman.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys (CSUR), 31*(3), 264–323.

Kotsiantis, S. B. (2007). Supervised machine learning: A review of classification techniques. *Informatica, 31*(3) (03505596).

Liao, Y., & Vemuri, V. R. (2002). Use of k-nearest neighbor classifier for intrusion detection. *Computers & Security, 21*(5), 439–448.

MacQueen, J., et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth berkeley symposium on mathematical statistics and probability* (Vol. 1, pp. 14).

Pal, N. (2007). *Advanced techniques in knowledge discovery and data mining*. Springer.

Poongothai, K., & Sathiyabama, S. (2012a). Efficient web usage miner using decisive induction rules. *Journal of Computer Science, 8*(6).

Poongothai, K., & Sathiyabama, S. (2012b). Integration of clustering and rule induction mining framework for evaluation of web usage knowledge discovery system. *Journal of Applied Sciences, 12*(14).

Rocchi, P. (2003). *The structural theory of probability: New ideas from computer science on the ancient problem of probability interpretation*. Springer.

Saneifar, H., Bringay, S., Laurent, A., & Teisseire, M. (2008). S 2 mp: Similarity measure for sequential patterns. In *Proceedings of the 7th australasian data mining conference* (Vol. 87, pp. 95–104).

Shen, W., Hao, Q., Yoon, H. J., & Norrie, D. H. (2006). Applications of agent-based systems in intelligent manufacturing: An updated review. *Advanced Engineering INFORMATICS, 20*(4), 415–431.

Steinbach, M., Ertöz, L., & Kumar, V. (2004). The challenges of clustering high dimensional data. In *New directions in statistical physics* (pp. 273–309). Springer.

Strehl, A., Gupta, G. K., & Ghosh, J. (1999). Distance based clustering of association rules. *Proceedings ANNIE 1999, 9*, 759–764.

Tuomi, I. (1999). Data is more than knowledge: Implications of the reversed knowledge hierarchy for knowledge management and organizational memory. In *Proceedings of the 32nd annual hawaii international conference on systems sciences, 1999. hicss-32* (pp. 12).

Wooldridge, M., Jennings, N. R., & Kinny, D. (2000). The gaia methodology for agent-oriented analysis and design. *Autonomous Agents and Multi-Agent Systems, 3*(3), 285–312.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., et al. (2008). Top 10 algorithms in data mining. *Knowledge and Information Systems, 14*(1), 1–37.