DFC:
A Performant Dagging Approach of Classification Based on Formal Concept

Nida Meddouri, Normandie University, UNICAEN, ENSICAEN, CNRS - UMR GREYC, Caen, France
https://orcid.org/0000-0002-7815-630X
Hela Khoufi, Faculty of Computing and Information Technology, Khulais, Saudi Arabia
Mondher Maddouri, College of Business, Jeddah, Saudi Arabia

ABSTRACT

Knowledge discovery data (KDD) is a research theme evolving to exploit a large data set collected every day from various fields of computing applications. The underlying idea is to extract hidden knowledge from a data set. It includes several tasks that form a process, such as data mining. Classification and clustering are data mining techniques. Several approaches were proposed in classification such as induction of decision trees, Bayes net, support vector machine, and formal concept analysis (FCA). The choice of FCA could be explained by its ability to extract hidden knowledge. Recently, researchers have been interested in the ensemble methods (sequential/parallel) to combine a set of classifiers. The combination of classifiers is made by a vote technique. There has been little focus on FCA in the context of ensemble learning. This paper presents a new approach to building a single part of the lattice with best possible concepts. This approach is based on parallel ensemble learning. It improves the state-of-the-art methods based on FCA since it handles more voluminous data.

KEYWORDS
Classification Rule, Dagging, Data Mining, Ensemble Method, Formal Concept, Knowledge Discovery Data, Machine Learning

INTRODUCTION

In this paper, we are interested in classification. Classification is a two-phase process: a learning phase which organizes the information extracted from a set of objects (or data) and a classification phase which determines the label/class of new objects. Many supervised classification techniques are proposed such as Classification by Formal Concept Analysis, Decision Tree, Bayes Net, SVM, Neural Networks…

The Formal Concept Analysis is a formalization of the philosophical notion of concept, defined as a pair of extension and intention of the concept. The intention of a concept refers to necessary and sufficient attributes of the concept in question. The extension of a concept is the set of instances that have been learned this concept. Several classification methods are proposed since the semantic
richness is guaranteed by the Formal Concept Analysis (Poelmans et al., 2013). Unfortunately, this classification methods encountered some problems such as an exponential complexity (in the worst case), a high error rate and overfitting (Meddouri & Maddouri, 2009; Meddouri & Maddouri, 2010).

Several ensemble methods are used to improve the error rate of any single learner (Freund, 1995) (Freund & Shapire, 1996). These proposed methods are based on sequential learning (Boosting). All the data are considered in each learning step and the weights are assigned to learning instances. However, Kuncheva reported that sequential learning (Boosting) is not enough for efficient classifier such as Decision Tree (Kuncheva et al., 2002). In the area of supervised learning, other ensembles exist, and they are based on parallel learning. The difference between these two ensemble methods derives from how to select data for learning. They are distinguished by the data sampling techniques as Bootstrapping used to learn the classifiers from subsets. The particularity of learning from a Bootstrap is to combine hard learning instances to misleading instances in the training set (Unlike the sequential approach) (Breiman, 1996; Breiman, 1996b; Kuncheva, 2004).

There has been little research that focused on the classification based on Formal Concepts Analysis as part of the parallel learning. We propose to use and study the Formal Concepts Analysis in this context and compare it with respect to the sequential approach. The best-known method, which is based parallel learning is Dagging (Disjoint samples aggregating) creates a number of disjoint groups and stratified data from the original learning data set (Ting & Witten, 1997), each considered as a subset of learning. The weak learner is built on this learning sets. The predictions are then obtained by combining the classifiers outputs by majority vote (Ting & Witten, 1997; Davison & Sardy, 2006). This method has shown its importance in recent work. Then, we propose to use this technique in this work to study the classifier ensembles based on formal concepts, since a limited number of studies have focused on the formal concepts in the context of parallel learning.

First, we present the basics of Formal Concept Analysis and several classification methods based on lattice concept or sub-lattice of concepts. Second, we present ensemble methods based on sequential and parallel learning and we propose a new method exploiting the advantages of the Dagging to generate and combine in a parallel way a weak concept learner. Then, we present classifiers based on FCA and an amelioration of this classifier. A comparative and experimental study is presented to evaluate the performance of concept ensembles based on certain criteria such as the number, variety, and type of classifiers. Finally, the comparative study shows the importance of parallel learning compared to sequential learning.

**FORMAL CONCEPT ANALYSIS AND CLASSIFICATION**

The classification approach based on Formal Concept Analysis (FCA) is a symbolic approach allowing the extraction of connections, reasons, and rules according to the concepts discovered from data (Carpineto & Romano, 1996). A concept is a form of modeling an object with its attributes. One of the main strengths of the FCA is its visualization properties. FCA produces concept lattices that can be represented by a graph (mathematical structure) and it is seen as an area of research in which we evolve from one level to another, validating the characteristics associated with concepts. The semantic richness is guaranteed, and so, FCA facilitates the task of the classifier in knowledge discovery. Many learning methods based on Formal Concept Analysis are proposed, such as GRAND (Oosthuizen, 1988), LEGAL (Liquiere & Nguifo, 1990), GALOIS (Carpineto & Romano, 1993), RULEARNER (Sahami, 1995), CIBLE (Njewoua & Nguifo, 1999), CLNN&CLNB (Xie & al., 2002), IPR (Maddouri, 2004), NAVIGALA (Visani & al., 2011), CITREC (Douar & al., 2008), CBA lays (Gupta & al., 2005), HMCS-FCA-SC (Ferrandin & al., 2013), SPFC (Ikeda & Yamamotol., 2013), CLANN (Nguifo et al., 2008), FCA-BRG (Cintra et al., 2015) and RMCS (Kashnitsky & Ignatov, 2015).
Definition

A formal context is a triplet \( \langle O, \mathcal{P}, \mathcal{R} \rangle \), where \( O = \{ o_1, o_2, \ldots, o_n \} \) is a finite set of \( n \) instances, \( \mathcal{P} = \{ p_1, p_2, \ldots, p_m \} \) a finite set of \( m \) properties (binary attributes) and \( \mathcal{R} \) is a binary relation defined between \( O \) and \( \mathcal{P} \). \( \mathcal{R}(o_j, p_i) = 1 \) means that \( i^{th} \) instance \( o_i \) verifies the \( j^{th} \) property \( p_j \). In relation \( \mathcal{R} \) (Ganter & Wille, 2005). The context is habitually represented by a cross-table or a binary table as shown in Table 1.

Let \( A \subseteq O \) and \( B \subseteq \mathcal{P} \) be two finite sets. For both sets \( A \) and \( B \), operators \( \varphi(A) \) and \( \delta(B) \) are defined as (Ganter & Wille, 2005):

- \( \varphi(A) = \{ p \ \forall, o \in A \text{ and } (o, p) \in \mathcal{R} \} \)
- \( \delta(B) = \{ o \ \forall, p \in B \text{ and } (o, p) \in \mathcal{R} \} \)

Operator \( \varphi \) defines the properties shared by all elements of \( A \). Operator \( \delta \) defines instances which share the same properties included in set \( B \). Operators \( \varphi \) and \( \delta \) define a Galois connection between sets \( O \) and \( \mathcal{P} \) (Ganter & Wille, 2005). The closure operators are \( A'' = \varphi \circ \delta(A) \) and \( B'' = \delta \circ \varphi(B) \). Finally, the closed sets \( A \) and \( B \) are defined by \( A = \varphi \circ \delta(A) \) and \( B = \delta \circ \varphi(B) \).

A formal concept of the context \( \langle O, \mathcal{P}, \mathcal{R} \rangle \) is a pair \( (A, B) \) where \( A \subseteq O \), \( B \subseteq \mathcal{P} \), \( \varphi(A) = B \) and \( \delta(B) = A \). Sets \( A \) and \( B \) are called, respectively, the extent (domain) and intent (co-domain) of the formal concept (Table 2).

From the formal context \( \langle O, \mathcal{P}, \mathcal{R} \rangle \), we can extract all possible concepts organized as a complete lattice (called Galois lattice) (Ganter & Wille, 2005). We needed the following partial order relation ‘\( \ll \)’ is defined between two concepts, \( (A_1, B_1) \ll (A_2, B_2) \) if and only if \( A_1 \subseteq A_2 \) and \( B_2 \subseteq B_1 \).

### Table 1. Illustration of the formal context (Weather data under binary format)

| \( O \times \mathcal{P} \) | \( P_1 \) | \( P_2 \) | \( P_3 \) | \( P_4 \) | \( P_5 \) | \( P_6 \) | \( P_7 \) | \( P_8 \) | Play |
|---|---|---|---|---|---|---|---|---|---|
| \( o_1 \) | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | No |
| \( o_2 \) | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | No |
| \( o_3 \) | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | Yes |
| \( o_4 \) | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | Yes |
| \( o_5 \) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Yes |
| \( o_6 \) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Yes |
| \( o_7 \) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Yes |
| \( o_8 \) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Yes |
| \( o_9 \) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Yes |
| \( o_{10} \) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Yes |
| \( o_{11} \) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Yes |
| \( o_{12} \) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Yes |
| \( o_{13} \) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Yes |
| \( o_{14} \) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | No |
The concepts $A_1(B_1)$ and $A_2(B_2)$ are called nodes in the lattice. Hass diagram is the chart of Galois lattice (Ganter & Wille, 2005). Figure 1 represents the concept lattice (Galois lattice) of the context presented in Table 1.

**Classification**

The *Galois* lattice is a search space in which we evolve from a level to another, by validating the characteristics associated with the concepts (Meddouri & Maddouri, 2009). Classification based on FCA must determine the class of new objects by navigating on a lattice of formal concepts. Many classification systems exist in the literature using complete lattice or sub-lattice of concept (Fu et al., 2004).

**Complete Lattice-Based Methods**

Many classification systems use lattice concepts such as RULEARNER (Sahami, 1995), GALOIS (Carpineto & Romano, 1993) and NAVIGALA (Visani & al., 2011). The most common used

Table 2. Specification of binary attributes

| Attributes | Signification    |
|------------|------------------|
| $p_1$      | Outlook=sunny    |
| $p_2$      | Outlook=overcast |
| $p_3$      | Outlook=rainy    |
| $p_4$      | Temperature=hot  |
| $p_5$      | Temperature=mild |
| $p_6$      | Temperature=cool |
| $p_7$      | Humidity         |
| $p_8$      | Windy            |
method is GRAND (Oosthuizen, 1988). From a pseudo-lattice, the classes of which contributes to the construction of this lattice as attributes, GRAND induces the most general rules. To do this, it selects the intermediate nodes of the immediate attribute node representing the class. To illustrate the classification approach based on a complete lattice of formal concepts, we apply GRAND on the initial formal context of Table 1. We obtained the concepts of Figure 2.

From Figure 2, generating the classification rules are starting an attribute node that materializes ‘Play=Yes’. For example, GRAND select the concept number 12 (Figure 2). It looks for attribute nodes deriving this concept and gets attributes \( \{ p_6, p_7 \} \). The following classification rule will be generated:

IF Temperature=Cool AND Humidity THEN Play=Yes

The instances labeled by the attribute-node ‘Play=No’ do not form a concept. According to (Oosthuizen, 1988) (Nguifo & Njiwoua, 2005), negatively labeled instances cannot induce a classification rule, only the positively labeled instances induce classification rule. The instances labeled by the attribute-node ‘Play=No’ do not induce a concept. Therefore, instances labeled negatively are considered instances associated to attribute-node ‘Play=No’. The classification rules generated by GRAND from the formal context of Table 1 concern only the ‘Play=Yes’:

1. IF Outlook=Sunny AND Temperature=mild AND Humidity THEN Play=Yes
2. IF Outlook=Rainy AND Humidity AND Windy THEN Play=Yes
3. IF Outlook=Overcast AND Temperature=Cool AND Humidity THEN Play=Yes
4. IF Temperature=Cool AND Humidity THEN Play=Yes

Figure 2. Concepts generated by GRAND from the Table 1
5. IF Outlook=Overcast AND Temperature=Hot AND Windy THEN Play=Yes
6. IF Outlook=Sunny AND Temperature=Cool AND Humidity AND Windy THEN Play=Yes
7. IF Outlook=Rainy AND Temperature=Cool AND Humidity AND Windy THEN Play=Yes

A critical overview of algorithms based on the extraction of formal concepts shows that existing algorithms in literature failed in their objectives. Indeed, almost all these algorithms have been focused on the extraction of all concepts of concept lattice, which increases the costs of extraction of classification rules and makes their use almost impossible for large databases. There are three common limits for systems based on the concept lattice. The complexity (temporally and spatially) of lattice generation is exponential. The navigation in huge search space is hard (Fu et al., 2004). And the used data is binary. For these reasons, many researchers focused on sub-lattice-based classification.

Sub-Lattice-Based Methods

The major difference between lattice and sub-lattice based classification is the number of concepts generated. However, their limit is the possible loss of information in a condensed data representation or a partial reproduction of the complete lattice. Systems like LEGAL (Liquiere & Nguifo, 1990), CIBLE (Njiwoua & Nguifo, 1999), CLNN&CLNB (Xie & al., 2002) and IPR (Maddouri, 2004) are characterized by their ability to build a part of the concept lattice and inducing classification rules.

In IPR for example, the construction of a concept cover is based on heuristic algorithms which reduces the complexity of learning. The concepts are extracted one by one. Each pertinent concept is given by a local optimization of Shannon entropy function. Each pertinent concept with associated majority class, constructs a rule. As an illustrative example, we applied IPR on the preceding formal context (Table 1), we obtained the concepts of Figure 3.

From the context of Table 1, IPR induces the rules by calculating for each pertinent concept the majority class:

1. IF Humidity THEN Play=Yes.
2. IF Outlook=Rainy AND Humidity THEN Play=Yes.
3. IF Windy THEN Play=Yes.
4. IF Temperature=Cool AND Humidity THEN Play=Yes.
5. IF Outlook=Overcast AND Temperature=Mild THEN Play=Yes.
6. IF Outlook=Rainy AND Temperature=Mild THEN Play=Yes.

Figure 3. Concepts generated by IPR from the Table 1
7. IF Outlook=Sunny AND Temperature=Hot THEN Play=No.
8. IF Outlook=Sunny AND Temperature=Mild AND Humidity THEN Play=Yes.
9. IF Outlook=Overcast AND Temperature=Cool AND Humidity THEN Play=Yes.
10. IF Outlook=Sunny AND Temperature=Mild AND Windy THEN Play=No.
11. IF Outlook=Rainy AND Temperature=Mild AND Windy THEN Play=Yes.
12. IF Outlook=Overcast AND Temperature=Hot AND Windy THEN Play=Yes.
13. IF Outlook=Sunny AND Temperature=Cool AND Humidity AND Windy THEN Play=Yes.
14. IF Outlook=Rainy AND Temperature=Cool AND Humidity AND Windy THEN Play=Yes.
15. IF Outlook=Sunny AND Temperature=Hot AND Windy THEN Play=No.
16. IF Outlook=Rainy AND Temperature=Cool AND Humidity AND Windy THEN Play=Yes.

One limit of IPR is the induction of redundant concepts (including same instances) (Maddouri, 2004) to cover all the formal context. For example, \(o_5\) is covered by the concepts \(\{p_1, \{o_5, o_6, o_7, o_9, o_{10}, o_{11}, o_{13}\}\}\), \(\{p_6, p_7, \{o_5, o_6, o_7, o_9\}\}\) and \(\{p_5, p_6, p_7, p_8, \{o_5\}\}\).

Several methods offer a partial reproduction of the concept lattice i.e., a generation of text classification rules, but unfortunately this leads to inefficient methods. These methods, which deal only with binary attributes, also incur some difficulties as the exponential complexity (in the worst case) and the generation of redundant concepts. We notice that with those methods based on sub-lattice classification, the constructed concepts are chosen based on inappropriate criteria (i.e., the depth of the lattice, the covering of the context, etc.) (Meddouri & Maddouri, 2009).

**ENSEMBLE METHODS**

In machine learning, several researchers have focused on ensemble methods of classifiers.

These methods can improve the performance of individual and perceived ‘weak’ classifiers. Empirical studies have shown that the classifiers sets are often much more accurate than the individual classifiers (Bauer & Kohavi, 1999). Different theoretical explanations have been proposed showing the effectiveness of ensemble methods (Kleinberg, 2000).

Currently, there are two types of ensemble methods. The first type uses sequential learning to generate iteratively a set of classifiers, the same model from the same learning data which are weighted at each iteration. The second applies parallel learning of classifiers processing learning data independently. The predictions of the classifiers are then combined in an aggregation/vote or a stacking algorithm (Kleinberg, 2000).

In this paper, we present these two types of learning because of their importance in the literature and the lack of work that apply to the formal concepts, we propose two approaches. The first is based on the sequential learning i.e., that each classifier is built according to the performance of the previous classifier constructed. The second is based on the parallel learning.

**Boosting**

*Boosting* is an adaptive approach, which makes it possible to correctly classify an object that can be inappropriately classified by an ordinary classifier. The main idea of *Boosting* is to build many classifiers which complement each other, to build a more powerful classifier. At first, it selects a subset of instances from the learning data set (different subsets from the training data set in each iteration). Then, it builds a classifier using the selected instances. Next, it evaluates the classifier on the learning data set, and it starts again T times (T is the number of generated classifiers). *Adaboost (Adaptive Boosting)* is the most well-known method of *Boosting* for classifiers generation and combination.
For $\mathcal{K}$ class problem, let $Y = \{1, \ldots, \mathcal{K}\}$ be the class labels, with $y_i \in Y$ is the class label associated for each instance $o_i$ ($i = 1$ to $n$). To generate $T$ classifiers in AdaBoost, the distribution of the weight of $o_i$ is initially determined as:

$$D_0(i) = \frac{1}{n}$$

The weight of $o_i$ is:

$$w^t_{i,y} = D_0(i) / (\mathcal{K} - 1) \text{ for each } y \in Y - \{y_i\}$$

On each iteration $t$ from 1 to $T$, we define:

$$W^t_i = \sum_{y=y_i} w^t_{i,y}$$

and we set:

$$q^t_i(i, y) = \frac{w^t_{i,y}}{W^t_i} \text{ for each } y \neq y_i$$

The distribution of weights is calculated by:

$$D^t(i) = \frac{W^t_i}{\sum_{i=1}^{N} W^t_i}$$

Each generated nominal classifier $h_i$ provides an estimated probability $p^t_i(o_i, y_i)$ to the class $y_i$ from the entry $o_i$.

Three cases are presented:

- If $p^t_i(o_i, y_i) = 1$ and $p^t_i(o_i, y) = 0, \forall y \neq y_i$, $h_i$ has correctly predicted the class of $o_i$.
- If $p^t_i(o_i, y_i) = 0$ and $p^t_i(o_i, y) = 1, \forall y \neq y_i$, $h_i$ has an opposed prediction of the class of $o_i$.
- If $p^t_i(o_i, y_i) = p^t_i(o_i, y), \forall y \neq y_i$, the class of $o_i$ is selected randomly ($y$ or $y_i$).

The error rate of $h_i$ is calculated on the weighted training set. If an instance $o_i$ is correctly classified by $h_i$, then the weight of this instance is reduced, otherwise, is increased. The pseudo-loss of the classifier $h_i$ is defined as:

$$\epsilon_i = 0.5 \times \sum_{i=1}^{N} D^t(i) \left[1 - p^t_i(o_i, y_i) + \sum_{y \neq y_i} q^t_i(i, y) p^t_i(o_i, y)\right]$$
The weights are then updated according to $\beta_t$:

$$
\beta_t = \varepsilon_t / \left(1 - \varepsilon_t \right)
$$

The procedure is repeated $T$ times and the result of AdaBoost is determined by the combination of the generated classifier outputs:

$$
h_{fin}(o_i) = \arg\max_{y \in \mathcal{Y}} \sum_{t=1}^{T} \log \left(1 / \beta_t \right) \times p_t \left(o_i, y_i \right)
$$

The first variant of the AdaBoost algorithm is called Adaboost.M1 (Freund, 1995; Freund et al., 1998) that uses the previous process and stops it when the error rate of a classifier becomes over 0.5. The second variant is called AdaBoost.M2 (Freund & Shapire, 1996) which has the particularity of handling multi-class data and operating whatever the error rate is. In this study, we use AdaBoost.M2 since Adaboost.M1 has the limit to stop Boosting if the learning error exceeds 0.5.

According to (Warmuth et al., 2007), adaptive update of learning data in sequential learning (Boosting), increases the weight of those misclassified by the previous classifier and improves the performance of any learning algorithm (weak learner). However, the capacity of sequential learning has been challenged once highly noisy data are used. Noisy data will be ignored in parallel learning and possibly will spread equiprobable between Bootstraps or other subsets of training data resampled. Therefore, researchers have proposed techniques such as Bootstrapping (Breiman, 1996) and Disjoints Stratified (Ting & Witten, 1997) which consist of ignoring these noisy data or distributing on different sets of learning (Skurichina & Duin, 1998).

Additive Regression is an ensemble method that enhances the performance of a regression base classifier. Each iteration fits a model to the residuals left by the classifier on the previous iteration. Prediction is accomplished by adding the predictions of each classifier. Reducing the shrinkage (learning rate) parameter helps prevent overfitting and has a smoothing effect but increases the learning time (Friedman, 1999).

DECORATE is an ensemble method for building diverse ensembles of classifiers by using specially constructed artificial training instances. Comprehensive experiments have demonstrated that this technique is consistently more accurate than the base classifier, Bagging and Random Forests. DECORATE also obtains higher accuracy than Boosting on small training sets and achieves comparable performance on larger training sets (Melville & Mooney, 2003).

MultiBoosting is an extension to the highly successful AdaBoost technique for forming decision committees. MultiBoosting can be viewed as combining AdaBoost with Wagging. It is able to harness both AdaBoost’s high bias and variance reduction with Wagging’s superior variance reduction. Using C4.5 as the base learning algorithm, MultiBoosting is demonstrated to produce decision committees with lower error than either AdaBoost or Wagging significantly more often than the reverse over a large representative cross-section of UCI data sets. It offers the further advantage over AdaBoost of suiting parallel execution (Webb, 1999).

**Bagging**

In parallel approach, Bagging is based on Bootstraps. The particularity of these training sets is to reduce the impact of hard instances to learn (called outliers and misleaders) (Skurichina & Duin, 1998). Each classifier is trained on a set of $n'$ training instances $(n' < n)$, drawn randomly with replacement from the original training set of size $n$. Such a training set is called a Bootstrap replicate of the original set. Each Bootstrap replicate contains, on average, 63.2% of the original training set,
with many instances appearing several times. Predictions on the new instances are made by taking the majority vote of the ensemble.

(Breiman, 1996) reports that majority vote can turn good classifiers into almost optimal classifiers. Bagging is typically applied to learning algorithms that are unstable i.e. a small change in the training set leads to a noticeable change in the model produced (Melville & Mooney, 2005). Since each ensemble member is not exposed to the same set of instances, they are different from each other. By voting the predictions of each of these classifiers, Bagging seeks to reduce the error due to variance of the base classifier. Bagging of stable learners, such as Naive Bayes, does not reduce error (Melville & Mooney, 2005). (Kuncheva et al., 2002) report that parallel learning improves the performance of unstable classifier as neural networks and decision trees. They report that Bagging is not beneficial to improve the performance of a linear classifier on large data. It will be then advantageous to use these methods with unstable classifier (such as CNC).

In the literature of data sampling methods, stratified sampling has proved efficiency (Ting & Witten, 1997). Disjoint and stratified data sets are more representative of the original training database. Learning from stratified data samples allows to generate more efficient classifier than those generated from the weighted data in the case of sequential learning classifiers. Dagging has the particularity to learn parallel from stratified data sets. We also propose to exploit this variant of parallel learning from stratified data to generate classifiers based on nominal concepts.

Algorithm 1. Algorithm of Dagging

**Input:**
- \( T \): the number of classifiers to generate.
- Learning data \( \mathcal{O} \) of \( n \) instances: \( \mathcal{O} = \{(o_1, y_1), \ldots, (o_n, y_n)\} \) labeled \( y_i \in \{1, \ldots, C\} \).

**Output:** \( h_{vote} \) generated classifier.

**Begin**
- Divide the population into \( t \) strates;
- Establish the most complete list of each \( t \) constituting strates;
- For \( t \) from 1 to \( T \)
  - Calculate the percentage \( P_t \) instances of \( t^{th} \) strate with respect to \( \mathcal{O} \).
  - Choose simply and randomly instances from \( t^{th} \) strate to form \( \mathcal{O}^{th} \) respecting \( P_t \).
  - Learn weak classifier on \( \mathcal{O}^{th} \) to generate \( h_t \).
- \( h_{vote} = \arg \max_{y \in \{1, \ldots, C\}} \sum_{t=1}^{T} h_t(o, y) \);

**End**

Dagging (Kotsianti & Kanellopoulos, 2007; Anyfantis et al., 2007) is described with more details in Algorithm 1. The learning algorithm is executed \( T \) times on various disjoint and stratified sets of learning instances. For each set, we do not obtain new instances, but we are satisfied to have a similar distribution to the initial sample of learning instances. The samples are obtained by randomly drawing \( n' \) instances without replacement in the training sample \( \mathcal{O} \), with \( n' < n \). These samples respected the distribution of learning instances as classes.
PROPOSED METHOD BASED ON FORMAL CONCEPT ANALYSIS

We have recently proposed a classifier based on FCA, which has the particularity to reduce training time and complexity. The principle is to find the most relevant concept for inducing a classification rule.

Classifier Based on Formal Concepts Analysis: CNC

The learning algorithm considers the whole of training instances and use nominal attributes (not only binary attributes). We note $L$, the number of nominal attributes $AN$ with:

$$AN = \{ AN_i \mid i = \{1, \ldots, L\}, \exists o_i \in O, \exists p \in P, AN_i(o_i) = p\}$$

The pertinent nominal concept within the data set is extracted by selecting the nominal attribute which minimizes the measure of Informational Gain (IG) calculated from the learning context:

$$IG(AN, O) = E(O) - \sum_{j=1}^{Val.Att} \frac{S(v_j)}{N} E(v_j)$$

$IG$ of the nominal attribute $AN$ (represented by $Val.Att$ different values) is calculated from the entropy function: $E()$. $S()$ calculates the relevance of a value $v_j$ of the attribute $AN$ overall $O$. The variation of IG depends on $S(v_j)$, if we neglect the variation of $E(O)$. A data set that contains redundant/duplicate instances (as simple random samples), maximizes the value of $S(v_j)$ (if an instance containing $v_j$ is redundant) and minimize IG of the corresponding attribute. This paralyzed effect does not bring a lot of diversity in the values of IG for the same attribute. In a diverse set of data, value of $S(v_j)$ is minimized and IG the corresponding attribute is more important. This phenomenon helps to better enhance the attributes of a diverse set. The stratified random sampling ensures the proportional presence of all the various subgroups within the data set. Clearly, finding the best attribute that maximizes the $IG$ in a stratified set is more interesting than in another set. So, we are recommending learning CNC from stratified sets since it is based on the calculation of IG.

Once the nominal attribute is selected ($AN^*$), we extract associated instances to each value $v_j$ from this attribute.

**Proposition 1:** From a nominal context (multi-valued), the $\delta$ operator is set by:

$$\delta(AN^* = v_j) = \{ o \in O \mid AN^*(o) = v_j \}$$

Then, we look for the other attributes describing all the extracted instances (using the closure operator $\delta \circ \varphi(AN^* = v_j)$). For this, we give the following proposition:

**Proposition 2:** From a nominal context (multi-valued), the $\varphi$ operator is set by:

$$\varphi(B) = \{ v_j \mid \forall o, o \in B \text{ and } \exists AN_i \in AN \mid AN_i(o) = v_j \}$$
So we construct our pertinent concept associated with each value $v_j$ of the best attribute $AN^*$ \(\delta(AN^* = v_j), \delta \circ \varphi(AN^* = v_j)\). A weak classifier is obtained by seeking the majority class associated with the extent of the pertinent concept \(\delta(AN^* = v_j)\). It induces a classification rule. The condition part of the rule is made up by the conjunction of the attributes included in the intent: \(\delta \circ \varphi(AN^* = v_j)\). The conclusion part of the rule is made up by the majority class. After that, it uses the discovered rule to classify the learning data set $O$ and so our proposed learning algorithm of pertinent concept stops at this iteration.

Algorithm 2. Algorithm of Classifier Nominal Concept (CNC)

**Input:** Sequence of $n''$ instances $O = \{(o_1, y_1), \ldots, (o_{n''}, y_{n''})\}$ with labels $y_i \in \mathcal{K}$.

**Output:** $h_{CNC}$ a classifier rule.

**Begin**

From $O$, find the attribute having the best IG value $AN^*$.

From $AN^*$, find the nominal value having the important efficient $v$.

Calculate the closure associated to $v$ \(\{\delta(AN^* = v)\}, \delta \circ \varphi(AN^* = v)\}$;

Determine the majority class $y^*$ associated with $\delta(AN^* = v)$.

Induce the classification rule $h_{CNC}$;

Return $h_{CNC}$.

**End**

**Parallel Ensemble of CNC**

The sequential learning builds sequentially a set of classifiers from the same model and from adaptively weighted data. Each classifier is generated based on the reweighting of training data which is dependent upon the performance of the previous classifier. It was found that the addition of classifiers could affect paralyzing in the sense that it does not lead to an implicit improving in the performance of sequential learning, but rather to its degradation because of overfitting (Kuncheva, 2005; Buhlmann & Hothorn, 2010). This is due to the repetitive sampling of the training data from similar distributions learning data. Looking for the best attribute for CNC is from similar training data will be the same closing of this attribute and the same as the previous classifier. It will not improve performance for this type of classifier in the case of sequential learning.

In parallel learning, data is first divided into random and independent subsets called *Bootstraps*. The same classification algorithm is then applied to each of these *Bootstraps*. Finally, the aggregation of generated classifiers is made by a simple majority vote or a weighted vote to predict the final class.

The best-known method, which is based on this type of learning is called *Bagging* (Bootstrap Aggregating) (Breiman, 1996; Breiman, 1996b). Methods using parallel learning are distinguished by the data sampling techniques they use to learn the classifiers from specific subsets. The peculiarity of learning from a Bootstrap is to combine the hard instances to the misleading instances in the training set (unlike the sequential approach).

(Kuncheva et al., 2002) and (Breiman, 1996) report that parallel learning improves the performance of an unstable classifier such as Networks Neural or Decision Trees. However, this type of learning does not improve the performance of stable linear classifiers that generates. Kuncheva reported that learning parallel (Bagging) will not be beneficial to improve the performance of a
linear classifier on large data. So, it will be advantageous to use learning parallel with a formal and unstable classifier (such as CNC).

To study the performance CNC, we selected 24 samples of different data (Presented with more details in next section. The performance of CNC is obtained according to the principle of 10 cross-validation. In Table 3, we report the variance of the error rate for each data set.

These results show that CNC is an unstable classifier, considering that ensemble methods, which are based on the combination of several classifiers, are known to improve the performance of an individual weak and unstable learner like CNC.

**CNC From Disjoint and Stratified Learning Data**

Recently, a great number of studies in machine learning has been concerned with parallel learning of classifiers that allow the improvement of single learner performances (Breiman, 1996). Parallel learning is known to improve the performance of any learning algorithm and discovering weak classifiers (weak learner). We propose to exploit the advantages of parallel ensembles methods to improve the performance of CNC.

| Data Sets    | Error Rate | Variance |
|--------------|------------|----------|
| 1. Contact lenses | 5%         | 15.81    |
| 2. Weather    | 5%         | 15.81    |
| 3. Lymphography | 8.29%     | 9.82     |
| 4. Sonar    | 15.38%     | 9.82     |
| 5. Segment   | 7.01%      | 1.07     |
| 6. Heart statlog | 13.70%    | 3.05     |
| 7. Glass     | 31.77%     | 15.12    |
| 8. Diabetes | 10.17%     | 3.87     |
| 9. Iris      | 4.67%      | 3.22     |
| 10. Balance scale | 28.65%    | 4.59     |
| 11. Car       | 7.47%      | 8.01     |
| 12. Kr vs. kp | 33.95%     | 1.8      |
| 13. Waveform | 13.18%     | 1.3      |
| 14. Optdigits | 28.36%     | 1.55     |
| 15. Nursery  | 12.67%     | 4.47     |
| 16. Pendigits | 9.68%      | 0.84     |
| 17. Credit German | 4.60%     | 1.51     |
| 18. Japanese Vowels | 18.45%    | 1.56     |
| 19. Splice   | 33.10%     | 2.24     |
| 20. Spambase | 6.56%      | 0.69     |
| 21. CMC      | 34.49%     | 2.58     |
| 22. Solar flare | 0.19%    | 0.39     |
| 23. Page-blocks | 1.17%     | 0.45     |
| 24. Yeast    | 40.84%     | 3.08     |
Wagging (Weight aggregating) is another ensemble method considered as a variant of Bagging (Bauer & Kohavi, 1999; Webb, 2000). Instead of applying resampling to obtain random Bootstraps (case of Bagging), Wagging uses random weights for learning instances. In (Bauer & Kohavi, 1999), gaussian noise is used to vary the weight. However, this can produce near zero weight, which leads to ignorance or disposal of instances to which these weights are associated.

We also report DECORATE (Diverse Ensemble Creation by Oppositional Relabeling of Artificial Training Examples) (Melville & Mooney, 2005; Melville & Mooney, 2004). This method generates a set of different classifiers with better performance. This set is generated iteratively. Each classifier is built individually and then added to the set. The first classifier is generated from the learning instances. Following classifiers are generated from this learning instances combined to others artificially generated. Labels of artificial instances are chosen such that they are different from the predictions of the initial learning instances. This provides more diversity in the learning data set. To maintain good performance of this ensemble, the error of each generated classifier is calculated. If this error is greater than the ensemble, generated classifier is rejected. An improvement is made to DECORATE to enable it learning from data whose attributes are multivalued (Kotsiantis, 2008). DECORATE is dedicated to numeric data rather than the nominal data. Therefore, it cannot be used for classifiers based on Formal Concept Analysis.

Dagging (Disjoint samples aggregating) is a variant of Bagging that creates a few disjoint groups with stratified data (Davison & Sardy, 2006; Ting & Witten, 1997), each considered as a learning subset. The predictions are obtained from a majority vote for supervised classification problems (Ting & Witten, 1997). This method has shown its importance in recent works. We propose to use it and to study the behavior of our CNC classifier based on stratified sampling and analysis the difference compared to random sampling.

The principle of DNC (Dagging Nominal Classifier) is to take several disjoint and stratified samples \( \{O_1^n, \ldots, O_T^n\} \). Our learning algorithm CNC is then built on each of them to get a collection of classifiers \( \{h_1, \ldots, h_p\} \) that will be combined by majority voting (Ting & Witten, 1997). In the case of learning from stratified samples, the characteristic of this type of sample is that it allows the learning algorithm to generate classifiers from representative samples having the same composition as the initial training set.

COMPARATIVE STUDY

In this section, we compare the proposed method DNC (Dagging Nominal Concept) with existing ones based on Formal Concept Analysis: GRAND, IPR, CITREC and BNC. To compare the presented approaches, we consider their complexities. We also compare the DNC method with existing ones in literature: Bayes Net, Naive Bayes, SVM, 1-NN, Decision Stump, C4.5, Random Forest and Random Tree and we focus on their classification rates.

Comparison of Complexities

Concerning the complexities, the variable \( n \) means the number of instances and the variable \( m \) means the number of attributes. To calculate the complexity of DNC method, \( T \) is a variable which means the number of stratified data set. We examine the learning algorithm of pertinent concept and we estimate its complexity to \( O(n\log(n)+nm) \). The complexity of stratified sampling from a set is estimated to \( O(2n) \) (The complexity of listing all instances is \( O(n) \) and the complexity of the simple random sampling without replacement is estimated to \( O(n) \)). The complexity of closure operator is \( O(nm) \). Also, we examine the algorithm of Dagging and we estimate its complexity. The complexity of the Dagging iterations is \( O(T) \). The complexity of applying our weak classifier is \( O(2n+ nm) \). To conclude, the complexity of the DNC method is estimated to \( O(T(2n + nm))=O(2n+ nm) \). Compared to the complexities of other methods (Meddouri & Maddouri, 2009; Meddouri et Madduri,
2009; Meddouri & Maddouri, 2010), we remark that the DNC method has the least theoretical complexity (Table 4).

**Experimental Study**

To compare the presented approaches, we implement them under *Waikato Environment for Knowledge Analysis (WEKA)* (Hall et al., 2009) and we use the well-known data sets from *UCI Machine Learning Repository* (Asuncion & Newman, 2007). The chosen data sets were discretized with 2 discrétional filters under *WEKA*. The first filter used is an instance filter that converts a range of numeric and string attributes into nominal attributes. The second filter used is an instance filter that converts a range of nominal attributes into binary attributes (to evaluate methods based on *Formal Concept Analysis*).

These data sets are presented in Table 8 in 2 groups and depending on the data diversity to see if it affects the performance of *Dagging*. The 10 first data sets we will evaluate GRAND, RULEARNER, CITREC and IPR (the small data sets are reserved to these methods based on *Formal Concept Analysis* due to excessive consumption of memory resources). All other 24 data sets are used to evaluate some methods known in the literature as DNC. For each data set, we present respectively the number of instances, the number of numeric attributes (before discretization), the number of nominal attributes (after discretization), the number of classes and data diversity. The last column presents the ratio between the number of different vectors of instances (attributes) and the total number of vectors in each database (Table 5).

The performance of classifiers generated is evaluated in terms of error rates. To calculate these rates, the 10 Cross-validation method is used in *WEKA* the principle of which is to divide each base into 10 subsets. In turn, one subset is used for testing and the other ones for learning (Kohavi 1995).

In the next section, we will try to provide answers to the following questions: Does the classifier number influence the performance of *Dagging* of *CNC*? Is the *Dagging* of *CNC* more interesting than other classifiers? What is the best adaptive learning for *CNC*: sequential or parallel? What are the conditions under which *CNC* behaves better than other classifiers?

**Influence of the Number of Classifiers on DNC**

To study the performance of *Dagging* using *CNC*, we generated sets of 2 to 13 classifiers, and we reported their error rates in Table 9. Signs ‘-’ indicates that the method cannot process the *Weather* data set (14 instances), due to the small number of instances not enough to construct sufficient disjoint and stratified subset of data. From this table, we report that the performance of odd sets of classifiers

| Systems          | GRAND | ULEARNER | IPR  | CITREC | DNC   |
|------------------|-------|----------|------|--------|-------|
| Kind of lattice  | Lattice | Lattice | Cover | Sub-lattice | Sub-lattice |
| Data             | Binary | Binary | Binary | Binary | Nominal |
| Number of classes| Multi-class | Multi-class | Multi-class | Multi-class | Multi-class |
| Selection of concepts | Consistency | Support | Entropy | No-Inclusion+Supp | Infor. Gain |
| Combination of methods | No | No | No | Bayes/K-PPV | Dagging |
| Knowledge learned | Rules | Rules | Rules | Rules | Rules |
| Classification    | Vote | More weighted | More Weighted | Vote | Vote |
| Complexity        | $O\left(\frac{2^k k^4}{k=\min(n,m)}\right)$ | $O\left(\frac{2^k k^4}{k=\min(n,m)}\right)$ | $O\left(n^2 m^2 (m + n)\right)$ | $O\left(2^m n\right)$ | $O\left(mn\right)$ |
is better than the pair sets. This is due to the voting rule of the combination that works best with an odd number of classifiers (Kuncheva et al. 2003). Dagging with more than 8 classifiers behaves better. The performance of these classifier ensembles is not correlated with the diversity of training data. We note that with 10, 11 and 13 classifiers, Dagging using CNC produces the best performance on 7 data sets from 24. The average error (for all the bases) occurred with 11 classifiers (14.45%) which is lower than that obtained with 10 classifiers (14.72%) or 13 classifiers (14.59%). That is why in the next experiments, we will retain this value for the number of classifiers (Table 6).

### Comparison With ACF Based Classifiers Classification Methods

Table 7 presents the performance of different methods based on Formal Concept Analysis including DNC. Signs '-' indicate that the method cannot process the sample data, due to excessive consumption of memory resources. As shown in Table 7, DNC has the specific ability to reduce the error rates compared to methods based on Formal Concept Analysis (GRAND, RULERNER, IPR and CITREC).
### Table 6. DNC performance with different numbers of classifiers

| Data Sets         | 2        | 3        | 4        | 5        | 6        | 7        | 8        | 9        | 10       | 11       | 12       | 13       |
|-------------------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| 1. Contact lenses | 16.67    | 6.67     | 3.33     | 3.33     | 3.33     | 0        | 0        | 0        | 0        | 0        | 0        | 0        |
| 2. Weather        | 0        | 10       | 5        | 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0        | -        |
| 3. Lymphography   | 10.29    | 7.62     | 12.19    | 10.19    | 11.52    | 7.62     | 8.05     | 6.05     | 6.1      | 4.76     | 2.76     | 6        |
| 4. Sonar          | 14.95    | 15.83    | 21.64    | 18.71    | 12.55    | 20.69    | 16.83    | 12.55    | 13.98    | **10.07** | 11.52    | 10.14    |
| 5. Segment        | 7.01     | 6.97     | 7.01     | 6.36     | 6.23     | 6.32     | 5.93     | 5.5      | **4.37** | 6.54     | 7.1      | 5.58     |
| 6. Heart statlog  | 11.48    | 12.22    | 12.59    | 13.7     | 13.33    | 14.44    | 18.89    | 16.67    | 12.59    | 17.78    | 14.07    | **10.37**|
| 7. Glass          | 32.32    | 28.57    | 32.29    | 28.92    | 26.67    | 30.8     | 28.07    | 24.37    | 26.17    | 25.19    | **19.48**| 23.83    |
| 8. Pima           | 10.17    | 10.17    | 10.17    | 12.64    | 9.54     | 11.06    | 9        | 11.21    | 15.66    | 12.25    | 12.25    | -        |
| 9. Iris           | 2.67     | 3.33     | 2        | 2.67     | 1.33     | 1.33     | 1.33     | 1.33     | **0.67** | 1.33     | 2.67     | **0.67** |
| 10. Balance-scale | 24.96    | 23.22    | 25.93    | 23.53    | 22.55    | 23.68    | 21.28    | 22.73    | 23.21    | 24.17    | **20.96**| 23.7     |
| 11. Car           | 9.04     | 7.41     | 14.35    | 8.9      | 12.67    | 10.3     | 8.1      | **6.3**  | 11.11    | 6.31     | 11.34    | 7.69     |
| 12. Kr vs. kp     | 33.95    | 33.95    | 33.95    | 33.95    | 33.85    | 33.85    | 33.82    | 33.92    | 33.92    | 33.95    | 33.95    | **32.88**|
| 13. Waveform      | 13.18    | 12.7     | 12.58    | 12.56    | 12.74    | 12.3     | 12.14    | 12.86    | 13.32    | **11.44**| 12.02    | 11.52    |
| 14. Optdigits     | 28.35    | 28.29    | 28.26    | 28.31    | 28.35    | 28.22    | 33.08    | 28.42    | 28.86    | **27.38**| 28.26    | 28.47    |
| 15. Nursery       | 13.06    | 14.35    | 14.46    | 12.89    | 14.73    | 14.41    | 14.58    | 14.21    | 14.79    | **11.85**| 14.47    | 14.77    |
| 16. Pendigits     | 9.66     | 9.68     | 9.86     | 11.33    | 11.69    | 11.71    | **9.55** | 11.57    | 16.74    | 9.66     | 13.18    | 11.83    |
| 17. German credit | **4.6**  | 6.4      | 7.3      | 7.4      | 8.3      | 8.8      | 7.3      | 9.7      | 11       | 10.4     | 8.7      | 9.2      |
| 18. Japanese vowels | **18.45** | 16.72    | 17.55    | 16.6     | 15.4     | 15.53    | 14.24    | 13.84    | 17.71    | 17.44    | 17.5     | **12.76**|
| 19. Splice        | 33.1     | 33.1     | 32.82    | 32.13    | 33.1     | 31.32    | **28.4** | 31.57    | 29.5     | 33.1     | 29.97    | 28.53    |
| 20. Spambase      | 7.35     | 7.3      | 7.28     | 8.26     | 8.46     | 11.13    | 8.22     | 8.85     | 7.28     | 9.37     | **6.52** | 11.13    |
| 21. Cmc           | 34.02    | 33.34    | 32.65    | 29.53    | 31.44    | 30.62    | 32.12    | 30.01    | **29.33**| 29.6     | 31.44    | 31.77    |
| 22. Solar flare   | 0.37     | 0.09     | 0.09     | 0.09     | 0.19     | 0.19     | 0        | 0.09     | 0        | 0.09     | 0        | 0        |
| 23. Page blocks   | 1.17     | 1.17     | 1.17     | 1.17     | 1.17     | 1.3      | **1.13** | 1.17     | 1.13     | 1.17     | 1.15     | **1.13** |
| 24. Yeast         | 40.84    | 40.84    | 40.84    | 40.77    | 40.97    | 40.64    | 40.44    | 40.57    | 40.37    | **39.69**| 40.84    | 41.38    |

### Table 7. Error rates of classification methods based on Formal Concept Analysis

| Data sets         | GRAND    | RULERANER | IPR | CITREC | DNC |
|-------------------|----------|-----------|-----|--------|-----|
| 1. Contact lenses | 31.67    | 36.67     | 44  | 53.33  | 0   |
| 2. Weather        | 25       | 35        | 35  | 60     | 0   |
| 3. Lymphography   | -        | -         | -   | 34.57  | 4.76|
| 4. Sonar          | -        | -         | -   | 53.38  | 10.07|
| 5. Segment        | -        | -         | -   | 16.62  | 6.54|
| 6. Heart statlog  | -        | 45.56     | -   | 31.48  | 17.78|
| 7. Glass          | 56.99    | 66.45     | 66.32| -      | 25.19|
| 8. Diabets        | -        | -         | -   | 29.83  | 15.66|
| 9. Iris           | 66.67    | 39.33     | 50  | 4      | 1.33|
| 10. Balance scale | 65.08    | 32.76     | 35.83| 37.44  | 24.17|
It is very important to determine the number of concepts obtained by each method based on Formal Concept Analysis. To determine the number of concepts in the Galois lattice, we use the GALICIA software based on Godin algorithm which constructs the complete lattice of concepts (Godin et al. 1995; Valtchev et al. 2003). To determine the number of concepts generated by GRAND, RULERANER, CITREC, IPR and DNC, we have used the WEKA platform. We note that GRAND, RULERANER and IPR incur failures during their execution on Diabetes, Segments and Sonar. GALICIA also reported a failure when run on the Sonar data.

As shown in Table 8, the concept lattice contains a great number of concepts. CITREC and DNC induces a small part of the lattice. DNC reached the best error rates with a small number of concepts. This shows that the concepts generated by DNC are more pertinent than those generated by IPR and CITREC. DNC gives a small number of concepts in the least possible time compared to the other approaches.

Comparison With State-of-the-Art Classification Methods

Table 9 presents the performance of different classification methods from the literature (Bayes Net, Naive Bayes, SVM, IB1, Decision Stump, C4.5, Random Forest and Random Tree) on numeric data. We note that all this classification methods incur failures during their execution Japanese vowels since it cannot handle nominal class attribute.

Table 10 presents the performance of different classification methods cited previously, compared to our proposed method DNC on discretized data. We report that the used discretization filter data reduce the error rates of Naive Bayes and SVM. As shown in this table, DNC has the specific ability to reduce the error rates together with the best performance for 14 discretized data sets (Contact lenses, Weather, Lymphography, Sonar, Diabetes, Iris, Balance scale, Waveform, German credit, Japanese vowels, CMC, Solar flare, Page blocks and Yeast). DNC has the best performance for all the data sets (average of 14.45%), then SVM (average of 16.29%) compared to Random Forest (average of 17.37%).

Influence of the Classifier Type in Dagging

(Meddouri & Maddouri, 2010) found that the sequential learning is beneficial for classifiers having decision tree structure such as C4.5 and ID3. However, the CNC is among the worst classifiers. Our objective here is to study the behavior of the classifiers in parallel learning. The error rates of the ensembles of generated classifiers are reported in Table 11. These results show that CNC holds the best performance for 17 data sets from the 24 (average of 14.46%). SVM is better than Random

| Data sets     | Galois Lattice | GRAND | RULERANER | CITREC | IPR | DNC |
|---------------|----------------|-------|-----------|--------|-----|-----|
| 1. Contact lenses | 33             | 58    | 31        | 8      | 14  | 11  |
| 2. Weather    | 36             | 46    | 34        | 4      | 16  | 11  |
| 3. Lymphography | 6019          | -     | -         | 16     | 180 | 11  |
| 4. Sonar      | -              | -     | -         | 4      | -   | 11  |
| 5. Segment    | 3037           | -     | -         | 38     | -   | 5   |
| 6. Heart statlog | 3237          | -     | -         | 4      | 82  | 6   |
| 7. Glass      | 232            | 611   | 367       | -      | 49  | 8   |
| 8. Diabets    | 256            | -     | -         | 4      | -   | 5   |
| 9. Iris       | 11             | 161   | 151       | 7      | 8   | 5   |
| 10. Balance scale | 16            | 640   | 625       | 7      | 4   | 5   |
Table 9. Error rates of classification methods on numeric data

| Data sets          | Bayes Net | Naive Bayes | SVM | 1-NN | Decision Stump | J48 | Random Forest | Random Tree | DNC |
|--------------------|-----------|-------------|-----|------|----------------|-----|---------------|-------------|-----|
| 1. Contact lenses  | 28.33     | 28.33       | 28.33 | 36.67 | 28.33          | 18.33 | 28.33         | 28.33       | 0   |
| 2. Weather         | 40        | 40          | 30   | 50   | 70             | 45   | 30            | 40          | 0   |
| 3. Lymphography    | 14.29     | 16.95       | 13.57 | 19.1  | 24.52          | 23.05 | 19            | 25.05       | 4.76 |
| 4. Sonar           | 19.71     | 32.12       | 24.05 | 13.43 | 26.95          | 28.83 | 19.26         | 26.5        | 10.07 |
| 5. Segment         | 8.57      | 19.78       | 6.93  | 2.86  | 71.43          | 3.07  | 2.34          | 4.2         | 6.54 |
| 6. Heart statlog   | 18.89     | 16.3        | 15.93 | 24.81 | 27.41          | 23.33 | 21.85         | 23.7        | 17.78 |
| 7. Glass           | 29.39     | 51.41       | 43.87 | 29.5  | 55.09          | 33.25 | 27.14         | 29.96       | 25.19 |
| 8. Diabetes        | 25.64     | 23.69       | 22.66 | 29.83 | 28.13          | 26.17 | 26.16         | 31.89       | 15.66 |
| 9. Iris            | 7.33      | 4           | 4    | 4.67  | 33.33          | 4     | 4.67          | 8           | 1.33 |
| 10. Balance scale  | 27.7      | 9.61        | 12.32 | 20.97 | 44.94          | 23.35 | 19.52         | 22.71       | 24.17 |
| 11. Car            | 14.29     | 14.47       | 6.25  | 22.74 | 29.98          | 7.64  | 7.35          | 16.84       | 6.31 |
| 12. Kr vs. kp      | 12.08     | 12.11       | 4.57  | 10.04 | 33.95          | 0.56  | 1.19          | 3.72        | 33.95 |
| 13. Waveform       | 20.16     | 20          | 13.32 | 26.38 | 43.24          | 24.92 | 18.2          | 27.56       | 11.44 |
| 14. Optdigits      | 7.76      | 8.67        | 1.67  | 1.39  | 80.25          | 9.31  | 3.31          | 14.06       | 27.38 |
| 15. Nursery        | 9.67      | 9.68        | 6.92  | 21.28 | 33.75          | 2.95  | 1.77          | 5.37        | 11.85 |
| 16. Pendigits      | 12.1      | 14.25       | 2.04  | 0.64  | 79.62          | 3.44  | 1.18          | 4.28        | 9.66 |
| 17. German credit  | 24.5      | 24.6        | 24.9  | 28    | 30             | 29.5  | 27.5          | 32.9        | 10.4  |
| 18. Japanese vowels| -         | -           | -    | -     | -              | -     | -             | -           | 17.44 |
| 19. Splice         | 4.58      | 4.64        | 6.58  | 24.08 | 37.62          | 5.64  | 10.56         | 27.87       | 33.1  |
| 20. Spambase       | 10.19     | 20.71       | 9.59  | 9.22  | 21.95          | 7.02  | 5.17          | 9.06        | 9.37  |
| 21. CMC            | 48.95     | 49.21       | 51.21 | 55.74 | 57.3           | 47.87 | 49.15         | 53.36       | 29.6  |
| 22. Solar flare    | 2.91      | 2.34        | 0.56  | 0.75  | 0.47           | 0.47  | 0.56          | 0.65        | 0.09  |
| 23. Page blocks    | 6.49      | 9.15        | 7.07  | 4.13  | 6.87           | 3.12  | 2.78          | 3.82        | 1.17  |
| 24. Yeast          | 43.26     | 42.39       | 42.86 | 47.71 | 59.3           | 43.87 | 40.23         | 49.12       | 39.69 |
| **Average**        | 18.99     | 20.62       | 16.51 | 21.04 | 40.19          | 18.03 | 15.96         | 21.25       | 14.45 |
classification problem. Further work using the Bagging without a priori knowledge on diversity in ensembles of classifiers that generates (Breiman, 1996). In (Ting & Witten, 1997), the authors have shown, theoretically and experimentally, the importance and the reliability of Dagging (Meddouri et al. 2012) noticed that CNC is not good enough with sequential learning on data sets of different sizes. For this, we used 24 samples of data to generate 11 sets of nominal classifiers such as CNC in Boosting, Bagging and Dagging. Table 12 presents the results of all these methods, in terms of error rate and training time. We can see that Dagging is the best method producing low error rates for all generated ensembles.

All the error rates of Boosting are higher than those of Dagging. In addition, Boosting is 6 times slower than Dagging. We also report that Dagging is 2 times faster than Bagging.

In our experiments, the use of 11 sets of nominal classifiers based on Formal Concept Analysis, in Dagging, is more interesting than by Bagging and Boosting. The parallel learning of such classifiers is far better than the sequential learning.

| Data sets         | Bayes Net | Naive Bayes | SVM | 1-NN | Decision Stump | C4.5 | Random Forest | Random Tree | DNC |
|-------------------|-----------|-------------|-----|------|----------------|------|---------------|-------------|-----|
| 1. Contact lenses | 28.33     | 28.33       | 28.33 | 36.67 | 28.33          | 18.33 | 28.33         | 28.33       | 0   |
| 2. Weather        | 40        | 40          | 30 | 50 | 70 | 45 | 30 | 40 | 0   |
| 3. lymphography   | 15.67     | 16.33       | 12.9 | 20.38 | 24.52          | 21.67 | 18.33       | 24.33       | 4.76 |
| 4. sonar          | 14.38     | 14.38       | 14.33 | 20.14 | 26 | 20.19 | 20.62 | 24.52 | 10.07 |
| 5. Segment        | 8.05      | 8.48        | 4.2 | 6.02 | 71.43          | 4.68 | 3.64         | 7.66 | 6.54 |
| 6. Heart statlog  | 16.67     | 16.67       | 15.93 | 18.89 | 27.41          | 18.15 | 17.41 | 19.26 | 17.78 |
| 7. Glass          | 25.17     | 25.63       | 23.77 | 32.34 | 55.09          | 26.06 | 22.45       | 24.76 | 25.19 |
| 8. Diabetes       | 22.13     | 22.13       | 22.53 | 29.41 | 25.26          | 21.74 | 23.3         | 22.65 | 15.66 |
| 9. Iris           | 6         | 6           | 6 | 6.67 | 33.33          | 6 | 5.33 | 6 | 1.33 |
| 10. Balance-scale | 29.29     | 29.29       | 28 | 29.77 | 43.67          | 30.41 | 29.44 | 30.89 | 24.17 |
| 11. Car           | 14.29     | 14.47       | 6.25 | 22.74 | 29.98          | 7.64 | 7.35 | 16.84 | 6.31 |
| 12. kr vs kp      | 12.08     | 12.11       | 4.57 | 10.04 | 33.95          | 0.56 | 1.19         | 3.72 | 33.95 |
| 13. Waveform      | 19.22     | 19.26       | 14.04 | 27.42 | 49.26          | 23.52 | 19.78 | 32.36 | 11.44 |
| 14. Optdigits     | 7.62      | 7.69        | 2.76 | 6.23 | 80.53          | 22.06 | 8.49 | 31.14 | 27.38 |
| 15. Nursery       | 9.67      | 9.68        | 6.92 | 21.28 | 33.75          | 2.95 | 1.77         | 5.37 | 11.85 |
| 16. Pendigits     | 11.99     | 12.1        | 1.63 | 3.48 | 79.26          | 11.44 | 3.92 | 12.16 | 9.66 |
| 17. German credit | 24.4      | 24.2        | 24 | 30 | 29.79          | 27.9 | 31.7 | 10.4 | 17.44 |
| 18. Japanese vowels | 50.61     | 50.71       | 42.22 | 39.02 | 59.13          | 42.04 | 38.12 | 45.77 | 33.1 |
| 19. Splice        | 4.58      | 4.64        | 6.58 | 24.08 | 37.62          | 5.64 | 10.56 | 27.87 | 31.2 |
| 20. Spambase      | 9.72      | 9.8         | 5.67 | 8.02 | 22.67          | 7.24 | 5.96 | 10.52 | 9.37 |
| 21. CMC           | 47.18     | 47.18       | 46.98 | 52.69 | 57.3 | 45.76 | 47.79 | 49.01 | 29.6 |
| 22. Solar flare   | 2.91      | 2.34        | 0.56 | 0.75 | 0.47 | 0.47 | 0.56 | 0.65 | 0.09 |
| 23. Page block    | 6.21      | 6.43        | 2.91 | 3.45 | 6.38 | 3.18 | 2.8 | 3.47 | 1.17 |
| 24. Yeast         | 40.84     | 40.91       | 39.9 | 51.28 | 59.3 | 40.91 | 42.06 | 41.65 | 39.69 |
| Average           | 19.45     | 19.53       | 16.29 | 22.94 | 41.04          | 18.89 | 17.37 | 22.52 | 14.45 |
CONCLUSION

*Formal Concept Analysis* is an interesting formalism to study machine learning and classification methods. It allows a full construction of the concepts and the dependence relationships between concepts to build a lattice of Formal Concepts. We report common limits among all the supervised learning methods based on *Formal Concept Analysis*: absence of the adaptive aspect and the generation of concepts is either exhaustive or non-contextual.

In this paper, we propose a new classifier based on nominal concept and we showed that it is better than the other methods based *Formal Concept Analysis*. We select the nominal attribute which maximizes the *Informational Gain* from the nominal data. Pertinent concepts are calculated from closure operators associated to this nominal attribute. A classification rule is obtained by associating a majority class to the extension of the pertinent concept.

Since it is a weak classifier, we propose to improve its performance by using ensemble methods.
We have recently proposed a variant of Dagging to generate classifiers based on Formal Concept. Recent work has encouraged its use for linear classifiers. No studies have focused on the generation of classifiers by formal concept type Dagging. We recommend a parallel learning by Dagging for classifiers such Formal Concept and the sequential learning for the other classifier types. In parallel learning, a few classifiers are sufficient for obtaining better performance than the use of individual one. Data diversity can affect the whole.

More experiments are possible on larger data sets and with other ensemble methods such as Random Forests. Many improvements on the ensemble methods can be brought. DNC methods used majority vote for classifier combination. A variety of voting rules already exist. A study of these rules can be beneficial to improve the performance of CNC ensembles.
REFERENCES

Asuncion, A., & Newman, D. J. (2007). UCI Machine Learning Repository. http://www.ics.uci.edu/~mlearn/MLRepository.html

Bache, K., & Lichman, M. (2013). UCI Machine Learning Repository. University of California, School of Information and Computer Science. http://archive.ics.uci.edu/ml

Bauer, E., & Kohavi, R. (1999). An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. Machine Learning, 36(1/2), 105–139. doi:10.1023/A:1007515423169

Breiman, L. (1996). Bagging predictors. Machine Learning, 24(2), 123–140. doi:10.1007/BF00058655

Breiman, L. (2000). Bias, Variance, And Arcing Classifiers. Technical Report 460, Statistics Department, University of California.

Bühlmann, P., & Hothorn, T. (2009). Twin Boosting: Improved feature selection and prediction. Statistics and Computing, 20(2), 119–138. doi:10.1007/s11222-009-9148-5

Carpineto, C., & Romano, G. (1993). GALOIS : An order-theoretic approach to conceptual clustering. Machine Learning Proceedings, 1993, 33–40. doi:10.1016/B978-1-55860-307-3.50011-3

Carpineto, C., & Romano, G. (1996). A lattice conceptual clustering system and its application to browsing retrieval. Machine Learning, 24(2), 95–122. doi:10.1007/BF00058654

Cintra, M. E., Monard, M. C., & Camargo, H. A. (2015). FCA-based rule generator, a framework for the genetic generation of fuzzy classification systems using formal concept analysis. 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). doi:10.1109/FUZZ-IEEE.2015.7337950

Davison, A. & Sardy, S. (2006). Méthodes de rééchantillonnage pour l’estimation de variance. Academic Press.

Douar, B., Latiri, C. C., & Slimani, Y. (2008). Approche hybride de classification supervisée à base de treillis de galois: application à la reconnaissance de visages. Actes des 8mes Journées Francophones EGC, E-11, 309-320.

Ferrandin, M., Nievola, J., Enembreck, F., Scalabrin, E., Vieira Kredens, K. & Ávila, B.o. (2013). Hierarchical Classification Using FCA and the Cosine Similarity Function. . 10.13140/RG.2.1.2747.4085

Freund, Y. (1995). Boosting a Weak Learning Algorithm by Majority. Information and Computation, 121(2), 256–285. doi:10.1006/inco.1995.1136

Freund, Y., Iyer, R., Schapire, R. E., & Singer, Y., (2003). An efficient boosting algorithm for combining preferences. J. Mach. Learn. Res., 4, 933–969.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. In Proceedings of the Thirteenth International Conference on International Conference on Machine Learning (ICML’96). Morgan Kaufmann Publishers Inc..

Friedman, J. H. (2002). Stochastic gradient boosting. Computational Statistics & Data Analysis, 38(4), 367–378. doi:10.1016/S0167-9473(01)00065-2

Fu, H., Fu, H., Njwoua, P., & Nguifo, E. M. (2004). A Comparative Study of FCA-Based Supervised Classification Algorithms. Lecture Notes in Computer Science, 2961, 313–320. doi:10.1007/978-3-540-24651-0_26

Ganter, B., Stumme, G., & Wille, R. (Eds.). (2005). Lecture Notes in Computer Science Formal Concept Analysis. doi:10.1007/978-3-540-31881-1

Görgen, O., Missaoui, R., & Alaoui, H. (1995). Incremental concept formation algorithms based on galois (concept) lattices. Computational Intelligence, 11(2), 246–267. doi:10.1111/j.1467-8640.1995.tb0031.x

Gupta, A., Kumar, N., & Bhatnagar, V. (2005). Incremental Classification Rules Based on Association Rules Using Formal Concept Analysis. Lecture Notes in Computer Science, 3587, 11–20. doi:10.1007/11510888_2

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software. SIGKDD Explorations, 11(1), 10–18. doi:10.1145/1656274.1656278

Ikeda, M., & Yamamoto, A. (2013). Classification by selecting plausible formal concepts in a concept lattice. CEUR Workshop Proceedings, 977, 22–35.
Kashnitsky, Y., & Ignatov, D. (2014). Can FCA-based Recommender System Suggest a Proper Classifier? CEUR Workshop Proceedings, 1257.

Kleinberg, E. M. (2000). A Mathematically Rigorous Foundation for Supervised Learning. Lecture Notes in Computer Science, 1857, 67–76. doi:10.1007/3-540-45014-9_6

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In Proceedings of the 14th international joint conference on Artificial intelligence. Morgan Kaufmann Publishers Inc.

Kotsiantis, S. B., & Kanellopoulos, D. (2007). Combining Bagging, Boosting and Dagging for Classification Problems. Knowledge-Based Intelligent Information and Engineering Systems, 493–500. 10.1007/978-3-540-74827-4_62

Kotsiantis, S. B. (2008). Handling imbalanced data sets with a modification of Decorate algorithm. International Journal of Computer Applications in Technology, 33(2/3), 91. doi:10.1504/IJCAT.2008.021931

Kuncheva, L. I. (2005). Using diversity measures for generating error-correcting output codes in classifier ensembles. Pattern Recognition Letters, 26(1), 83–90. doi:10.1016/j.patrec.2004.08.019

Kuncheva, L. I. (2014). Combining Pattern Classifiers. 10.1002/9781118914564

Kuncheva, L. I., Skurichina, M., & Duin, R. P. W. (2002). An experimental study on diversity for bagging and boosting with linear classifiers. Information Fusion, 3(4), 245–258. doi:10.1016/S1566-2535(02)00093-3

Kuncheva, L. I., Whitaker, C. J., Shipp, C. A., & Duin, R. P. W. (2003). Limits on the majority vote accuracy in classifier fusion. Pattern Analysis & Applications, 6(1), 22–31. doi:10.1007/s10044-002-0173-7

Liquière M., Mephu Nguifo E., (1990). LEGAL: LEarning with GAlois Lattice. Actes des Journées Françaises sur l’Apprentissage (JFA), 93-113.

Maddouri, M. (2004). Towards a machine learning approach based on incremental concept formation. Intelligent Data Analysis, 8(3), 267–280. doi:10.3233/IDA-2004-8304

Meddouri, N., Khoufi, H., & Maddouri, M. S. (2012). Diversity Analysis on Boosting Nominal Concepts. Lecture Notes in Computer Science, 7301, 306–317. doi:10.1007/978-3-642-30217-6_26

Meddouri, N., & Maddouri, M. (2008). Classification methods based on formal concept analysis. In Proceedings of the 6th International Conference on Concept Lattices and Their Applications. Palacky University.

Meddouri, N., & Maddouri, M. (2009). Boosting Formal Concepts to Discover Classification Rules. Lecture Notes in Computer Science, 5579, 501–510. doi:10.1007/978-3-642-02568-6_51

Meddouri, N., & Maddouri, M. (2010). Adaptive Learning of Nominal Concepts for Supervised Classification. Lecture Notes in Computer Science, 6276, 121–130. doi:10.1007/978-3-642-15387-7_16

Melville, P., & Mooney, R. J. (2004). Diverse ensembles for active learning. Twenty-First International Conference on Machine Learning - ICML '04. doi:10.1145/1015330.1015385

Melville, P., & Mooney, R. J. (2005). Creating diversity in ensembles using artificial data. Information Fusion, 6(1), 99–111. doi:10.1016/j.inffus.2004.04.001

Mephu Nguifo, E., & Njiwoua, P. (2005). Treillis de concepts et classification supervisée. Techniques et Sciences Informatiques, 24(4), 449–488. doi:10.3166/tsi.24.449-488

Mephu Nguifo, E., Tsopzé, N., & Tindo, G. (2008). M-CLANN: Multi-class Concept Lattice-Based Artificial Neural Network for Supervised Classification. Lecture Notes in Computer Science, 5164, 812–821. doi:10.1007/978-3-540-87559-8_84

Njiwoua, P., & Mephu Nguifo, E. (1999). Améliorer l’apprentissage à partir d’instances grâce à l’induction de concepts: Le système CIBLe. Revue d’Intelligence Artificielle, 13(2), 413–440.

Oosthuizen, G. D. (1992). The use of a lattice in knowledge processing (Ph.D. Dissertation). University of Strathclyde.

Poelmans, J., Ignatov, D. I., Kuznetsov, S. O., & Dedene, G. (2013). Formal concept analysis in knowledge processing: A survey on applications. Expert Systems with Applications, 40(16), 6538–6560. doi:10.1016/j.eswa.2013.05.009

Sahami, M. (1995). Learning classification rules using lattices (Extended abstract). Machine Learning, ECML-95, 343–346. doi:10.1007/3-540-59286-5_83
Skurichina, M., & Duin, R. P. W. (1998). Bagging for linear classifiers. *Pattern Recognition, 31*(7), 909–930. doi:10.1016/S0031-3203(97)00110-6

Ting, K. M., & Witten, H. I. (1997). Stacking Bagged and Dagged Models. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML ’97)*. Morgan Kaufmann Publishers Inc.

Valtchev, P., Grosser, D., Roume, C., & Hacene, M. R. (2003). Galicia: an open platform for lattices. *Proceeding of the 11th International Conference on Conceptual Structures*, 241–254.

Visani, M., Bertet, K., & Ogier, J.-M. (2011). NAVIGALA: An original symbol classifier based on navigation through a galois lattice. *International Journal of Pattern Recognition and Artificial Intelligence, 25*(04), 449–473. doi:10.1142/S0218001411008634

Warmuth, M. K., Glocer, K., & Rätsch, G. (2007). Boosting algorithms for maximizing the soft margin. In *Proceedings of the 20th International Conference on Neural Information Processing Systems (NIPS’07)*. Curran Associates Inc.

Webb, G. (2004). MultiBoosting: A Technique for Combining Boosting and Wagging. *Machine Learning, 40*(2), 159–196. doi:10.1023/A:1007659514849

Xie, Z., Hsu, W., Liu, Z., & Lee, M. L. (2002). Concept lattice based composite classifiers for high predictability. *Journal of Experimental & Theoretical Artificial Intelligence, 14*(2-3), 143–156. doi:10.1080/09528130210164206

**ENDNOTES**

1. The data sets are selected from UCI Machine Learning Repository.
2. In the example of the formal context, the values of Play (or Class) are Yes (1 or ‘+’ in other contexts) for positive instances, and ‘No’ (2 or ‘-’ in other contexts) for negative instances. For example, in the context (Table 1), \{o_1, o_4, o_5, o_7, o_9, o_{10}, o_{11}, o_{12}, o_{13}\} are the instances of the Class ‘1’ or Play ‘Yes’ (positive), and \{o_1, o_6, o_9, o_{12}, o_{13}\} are the instances of the Class ‘2’ or Play ‘No’ (negative).
3. Available at https://www.cs.waikato.ac.nz/ml/Weka
4. weka. filters. supervised. attribute. Discretize-Rfirst-last
5. weka. filters. supervised. attribute. NominalToBinary
6. GALICIA is available at: http://www.iro.umontreal.ca/~galicia/
7. In this work, the SMO module of WEKA with a default parameter setting is used to perform classification via the SVM
8. In this work, the IB1 module of WEKA with default parameter settings is used to perform classification via the Nearest-neighbor classifier
9. In this work, the J48 module of WEKA with a default parameter setting is used to perform classification via the C4.5

**Nida Meddouri** obtained her doctorate in Computer Science at the University of Tunis El Manar in 2015. In 2019, she was hired as a teaching and research associate at the University of Caen Normandy and is currently a senior researcher at GREYC (CNRS - UMR 6072). She writes and presents numerous publications on Machine Learning based on Formal Concept Analysis and ensemble methods, and is the author of Learning Sets of Classifications Rules by Formal Concept Analysis (2015, Faculty of Mathematical, Physical and Natural Sciences of Tunis - University of Tunis El Manar).

**Mondher Maddouri** is a computer science engineer from El Manar University in 1994, PhD in Machine Learning from the same Tunisian university in 2000, and HDR in 2008. Actually, Full Professor at Carthage University (Tunisia) and acting in University of Jeddah (KSA). Also, invited Professor at ARTOIS University (France). Author of a hundred of publications on Machine Learning and Data Mining. Especially, the promotion of Machine Learning methods based on Formal Concept Analysis, ensemble learning and cloud computing. Interesting applications was designed for mining genomic data in sequence and 3D structures.