Spam Detection System Combining Cellular Automata and Naive Bayes Classifier

F. Barigou*, N. Barigou**, B. Atmani***
Computer Science Department, Faculty of Sciences, University of Oran
BP 1524, El M'Naouer, ES-SENIA, 31 000 Oran, Algeria
{*fatbarigou, **barigounaouel, ***atmani.baghdad}@gmail.com

Abstract. In this study, we focus on the problem of spam detection. Based on a cellular automaton approach and naïve Bayes technique which are built as individual classifiers we evaluate a novel method combining multiple classifiers diversified both by feature selection and different classifiers to determine whether we can more accurately detect Spam. This approach combines decisions from three cellular automata diversified by feature selection with that of naïve Bayes classifier. Experimental results show that the proposed combination increases the classification performance as measured on LingSpam dataset.

1 Introduction

Spam is rapidly becoming a major problem on the Internet. Some recent studies shows that about 80% of the e-mails sent daily are Spam [8]. The major problem concerning spam is that it is the receiver who is paying in terms of its time, bandwidth and disk space. To address this growing problem of spam, many solutions have emerged. Some of them are based on the header of the email such as black list, white list and DNS checking. Other solutions are based on the text content of the message such as filtering based on machine learning. Many techniques have been developed to classify e-mails –for good review the reader can look, e.g., [9]. In a previous study [4], we proposed CASD (a Cellular Automaton for Spam Detection) a new approach to spam detection, based on symbolic induction by cellular automata [3]. Experiments show a very high quality of prediction when using stemming and Information gain as a features selection function [5]. A performance improvement is also observed over NB and KNN proposed in [2] on Ling Spam corpora. In this paper, our aim is to further improve the spam detection by adopting a combination strategy of classifiers. One technique to create an ensemble of classifiers is to use different feature subsets for each individual classifier. We believe that by varying the feature subsets to train the classifiers we can improve the performance of filtering, since it is possible to incorporate diversity and produce classifiers that tend to have high variety in their predictions. In a set of experiment to prove this, the same learning algorithm of CASD is trained over three different subsets of features and combined by voting, with a naïve Bayes algorithm.

The remainder of this paper is organized as follows; in section 2, we give an overview of the different types of strategies for classifier combination and we follow with the related work in combining multiple classifiers for spam detection. Section 3, first introduces the Naïve Bayes classifier and the CASD based cellular automaton

and then moves to the proposed combination approach. Experimental results are presented in section 4. Conclusions are finally drawn in section 5.

2 Background

A general overview of classifier combination is given in section 2.1. Some background on the spam detection using classifier combination is given in section 2.2.

2.1 Combining Classifiers

An ensemble of classifiers combines the decisions of several classifiers in some way in an attempt to obtain better results than the individual members. Such systems are also known under the names multiple classifiers, committees or classifier fusion. Numerous studies have shown that combining classifiers yields better results than achievable with an individual classifier. A good overview of different ways of constructing ensembles as well as an explanation about why ensemble is able to outperform its single members is pointed in [11].

An ensemble of classifiers must be both diverse and accurate in order to improve accuracy, compared to a single classifier. Diversity guarantees that all the individual classifiers do not make the same errors. If the classifiers make identical errors, these errors will propagate to the whole ensemble and so no accuracy gain can be achieved in combining classifiers. In addition to diversity, accuracy of individual classifiers is important, since too many poor classifiers can overwhelm correct predictions of good classifiers [7, 15].

In order to make individual classifiers diverse, many ensemble methods use feature selection so that each classifier works with a specific feature set. To contribute to this research, we propose to employ multiple classifiers, each making predictions based on subsets of features.

2.2 Spam detection using multiple classifiers

In the context of spam filtering, a number of ensemble classification methods have been studied. Sakkis et al. [13] combined a Naïve Bayes (NB) and k-nearest neighbor (k-NN) classifiers by stacking method and found that the ensemble achieved better performance. Carreras and Marquez [6] used boosting decision trees with the AdaBoost algorithm. Compared with two learning algorithms, the induction decision trees (DT) and Naïve Bayes, Adaboost clearly outperformed the above two learning algorithms in terms of the F1 measure. Rios and Zha [12] applied random forests, an ensemble of decision trees, using a combination of text and meta data features. For low false positive spam rates, RF was shown to be overall comparable with support vector machines (SVM) in classification accuracy. Also, Koprincha et al. [10] studied the application of random forests to Spam filtering. The LingSpam and PU1 corpora with 10-fold cross-validation were used, selecting 256 features based on either information gain or the proposed term-frequency variance. Random forests produced the best overall results. Shih et al. [14] proposed an architecture for collaborative agents, in which algorithms running in different clients can interact for the

classification of messages. The individual methods considered include NB, Fisher's probability combination method, DT and neural networks. In the framework developed, the classification given by each method is linearly combined, with the weights of the classifiers that agree (disagree) with the overall result being increased (decreased). The authors argued that the proposed framework has important advantages, such as robustness to failure of single methods and easy implementation in a network.

3 Proposed Framework

In this research, we propose an ensemble of classifiers diversified by both manipulating input data and using two different classifiers Cellular automaton CASD [4] and Naïve bayes approach. These two classifiers are given in section 3.1 and 3.2 while the design of the proposed combination is discussed in section 3.3.

3.1 Naive Bayes Classifier

Naïve Bayes (NB) which has been widely used for spam filtering [1,2, 13] is a simple but highly effective classifier. It uses the training data to estimate the probability that an instance belongs to a particular class. NB requires little storage space during both the training and classification stages; the strict minimum is the memory needed to store the prior and conditional probabilities. In our experiments, each message is represented as a binary vector ($x_1, \ldots, x_m$), where $x_i = 1$ if a particular token X_i of the vocabulary is present, otherwise $x_i = 0$.

From Bayes' theorem, the probability that a message with vector $\vec{x} = (x_1, \ldots, x_m)$ belongs in category c (= spam or lefitimate) is: $P(c|\vec{x}) = \frac{P(c) \times P(\vec{x}|c)}{P(\vec{x})}$. NB classifies each e-mail in the category that maximizes the product $P(c) \times P(\vec{x}|c)$. The a priori probabilities p(c) are typically estimated by dividing the number of training e-mails of category *c* by the total number of training e-mails. And the probabilities $P(\vec{x}|c)$ are calculated as follows: $P(\vec{x}|c) = \prod_{i=1}^{i=m} P(xi|c) = \frac{Xc+1}{Nc+|vocabulary|}$ where Xc is the number of occurrences of token X in e-mails with label c, Nc is the total number of token occurrences in e-mails labeled *c* and |vocabulary| is the number of unique tokens across all e-mails.

3.2 CASD : a Cellular Automaton for Spam Detection

CASD is a classifier which is built on the cellular automaton CASI [3]. Besides its high classification accuracy, CASD also has advantages in terms of simplicity, classification speed, and storage space [5].

Cellular automaton CASI (Cellular Automaton for Symbolic Induction) is a cellular method of generation, representation and optimization of induction graphs generated from a set of learning examples. It produces conjunctive rules from a Boolean induction graph representation that can power a cellular inference engine. This Cellular-symbolic system is organized into cells where each cell is connected

only with its neighbors (subset of cells). All cells obey in parallel to the same rule called local transition function, which results in an overall transformation of the system. CASI uses a knowledge base in the form of two layers of finite automata. The first one, called CelFact, represents the facts base and the second one, called CelRule, represents the rule base. In each layer, the content of a cell determines whether and how it participates in each inference step; at every step, a cell can be active or passive, can take part in the inference or not. The states of cells are composed of three parts; EF, IF and SF, and ER, IR and SR which are the input, internal state and output parts of the CelFact cells, and of the CelRule cells, respectively. The neighborhood of cells is defined by two incidence matrices called R_E and R_S respectively. They represent the input respectively output relation of the facts and are used in forward chaining.

- The input relation, noted iR_Ej, is : *if (fact i $\in$ Premise of rule j) then iR_Ej =1 else iR_Ej = 0.*
- The output relation, noted iR_Sj, is : *if (fact i $\in$ Conclusion of rule j) then iR_Sj =1 else iR_Sj =0.*

The cellular automaton dynamics is implemented as a cycle of an inference engine made up of two local transitions functions *δfact* and *δrule*.

The transition function *δfact* which corresponds to the evaluation, selection and filtering phases *is defined as:* $(EF, IF, SF, ER, IR, SR) \xrightarrow{\delta_{fact}} (EF, IF, EF, ER + (R_E^T \times EF), IR, SR)$

The transition function *δrule* which corresponds to the execution phase is defined as: $(EF, IF, SF, ER, IR, SR) \xrightarrow{\delta_{rule}} (EF + (R_S \times ER), IF, SF, ER, IR, \overline{ER})$

3.2.1 Learning classifier system

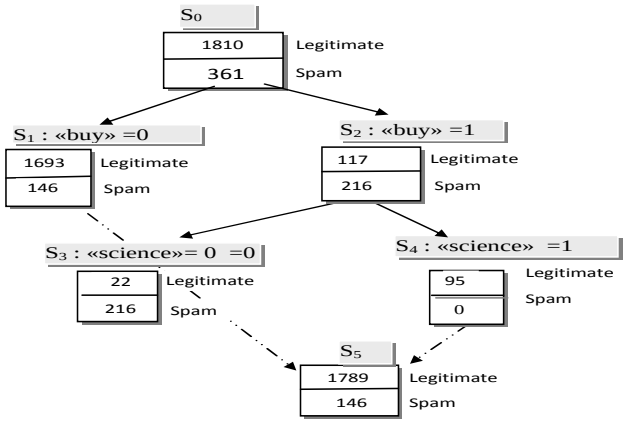

Fig.1. Example of an induction graph with only two terms.

During the learning phase, the Sipina method produces a graph. From this graph, a set of rules is inferred. They are in the form of "if condition1 and condition 2 and …condition n then conclusion". For example, in the graph of Figure 1, if we look to partition number 2 at node number 1 (S1) we have the rule "*if the term 'buy' is not*

present then the email is legitimate", because the majority of emails (1693) which do not contain this term are legitimate.

The set of rules generated from induction graph are modeled by the CASI automaton as follows:

- The set of all conditions and conclusions are represented by a Boolean facts base called *CelFact.*
- The set of rules is represented by a Boolean Rule-based called *CelRule*.
- An input matrix R_E which memorizes conditions of the rules.
- and finally, an output matrix R_S which memorizes conclusions of the rules.

Forward chaining will allow the model to move from initial configuration to the next configurations G_0 , G_1, ... G_n. The inference stops after stabilization with a final configuration. At this step the construction of cellular model is complete.

Table 1 presents the final configuration corresponding to the example of Figure 1. Three rules, represented by *CelRule* layer are deduced from the graph. The conditions and conclusions of these rules are stored in *CelFact* layer. The premises are the terms used in classification and the last two facts present the two classes. Note that no facts are established: EF = 0.

In the input matrix R_E (respectively output matrix R_S) are stored the premises (respectively the conclusions) of each rule. For example, the rule R2, has premises "buy = 1", "science=0" and a conclusion "class = spam".

Interaction between these two layers (CelFact and CelRule) is done by $\delta fact$ and $\delta rule$.

Table 1. Final Configuration: CelRule, CelFact, R_E, and R_S.

Rules	*ER*	*IR*	*SR*
R1	0	1	0
R2	0	1	0
R3	0	1	0
CelRule			

Facts	*EF*	*IF*	*SF*
buy=0	0	1	0
buy=1	0	1	0
science=0	0	1	0
science=1	0	1	0
S3:class=spam	0	1	0
S5:class=legitimate	0	1	0
CelFact			

R_E	*R1*	*R2*	*R3*
buy = 0	1	0	0
buy =1	0	1	1
science=0	0	1	0
science=1	0	0	1
S_3:class=spam	0	0	0
S_5:class=legitimate	0	0	0

R_S	*R1*	*R2*	*R3*
buy = 0	0	0	0
buy=1	0	0	0
science=0	0	0	0
science=1	0	0	0
S_3: class=spam	0	1	0
S_5:class=legitimate	1	0	1

3.2.2 Classification

We can use the model composed of CelFact, CelRule, R_E and R_S to classify new e-mails. The classification process is illustrated in Figure 2.

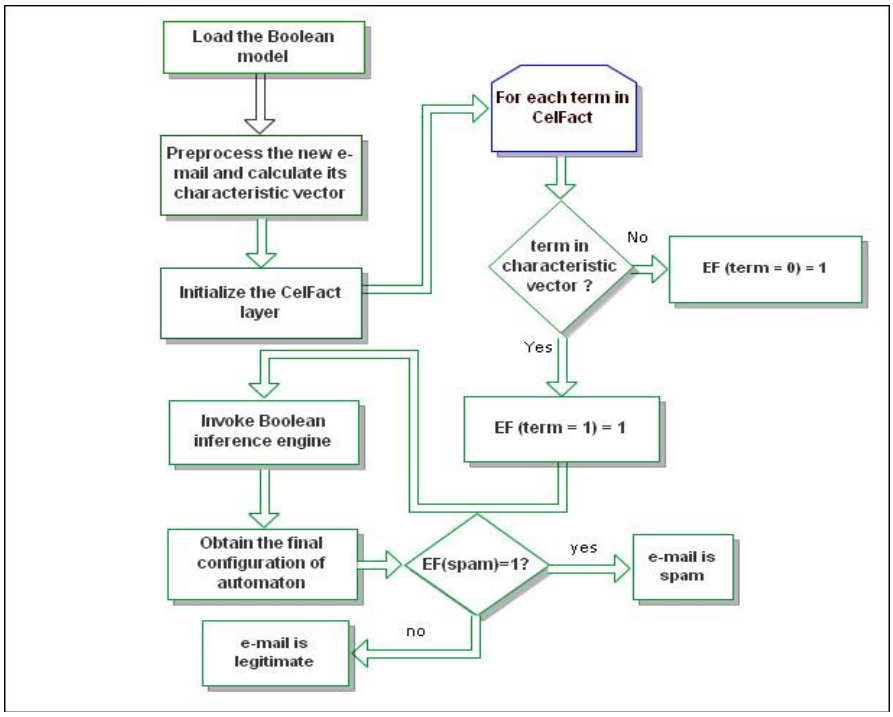

Fig.2. Classification Process

3.3 Proposed classifier combination

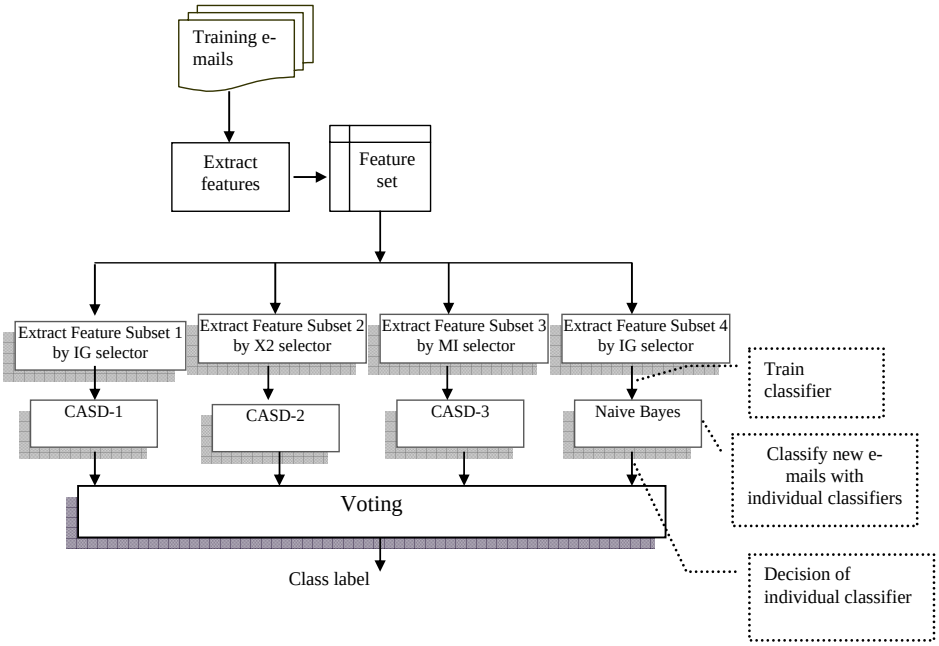

Fig.3. Architecture of the proposed ensemble classifiers for spam detection: 3CA-1NB (Three Cellular Automata combined with one Naïve Bayes).

Methods for creating ensembles [7, 15] focus on producing diversified base classifiers. Indeed, combination can be done by manipulating the training data, manipulating the input features, using different learning techniques to the same data. In this paper, we have chosen to consider combination by manipulating both features and using two different classifiers (CASD and NB).

The proposed approach termed 3CA-1NB (See Figure 3) combines three cellular automata classifiers (CASD), where each one is trained only with a feature subset. These subsets are generated with three different feature selection functions [17]: Information gain (IG), mutual information, and Chi-2 statistic respectively. We combine the decisions of these classifiers with that of Naïve bayes decision using voting[1] strategy. Our motivation for using this combining technique by varying feature selectors and using two different classifiers emerged from our preliminary results [4, 5] which indicate:

- The set of features selected by CASD during the learning phase depends on the selection function used to select features. For example, we observe that the features subset used by CASD after a selection based on information gain is generally

[1] E-mail is classified spam when at least two classifiers decide spam

different from that which was selected by the Chi-2 statistic or MI function. Therefore, we are guaranteed of having high feature set diversity.

- When using CASD, The quality of detection is better when features selection is done by MI or $\chi 2$ (we have precision=100%), while the coverage is very low in the case of a selection with $\chi 2$ (recall is very low) but very good with IG selector (see table 2 below). We want a classifier with high quality of detection and high coverage.
- Besides their simplicity, classification speed, CASD and NB also have advantages in terms of high classification accuracy.

4 Experimental study and results

We used the publicly available LingSpam corpora [2]. It comprises 2893 different e-mails, of which 2412 are legitimate e-mails obtained by downloading digests from the list and 481 are spam e-mails retrieved from one of the authors of the corpus [1, 13].

4.1 Linguistic preprocessing and feature selection

The first step in the process of constructing a classifier is the transformation of the e-mails into a format appropriate for the classification algorithms. We use an indexing module to:

(a) Tokenize texts and establish an initial list of terms;

(b) Eliminate stop words using a pre-defined stop list and;

(c) Perform stemming with a variant of the Porter[2] algorithm.

Prior experiments [5] have shown that stemming improves classification performance. In this paper we report results on stemmed data. Since the number of terms after this preprocessing phase is very high, and to reduce the computational cost and improves the classification performance, we must select those that best represent the emails and remove less informative and noisy ones. Based on a study of [17] indicating the most used feature selectors in text categorization, we have implemented three feature selectors: Information gain (IG), mutual information (MI) and $\chi 2$-statistic (CHI). The system calculates the chosen measure for all the terms, and then takes the first *k* terms corresponding to larger scores. In our experiments the threshold's parameter is set to k= 500. After feature selection process, each e-mail is represented by a vector that contains a weighting for every selected term. This weighting represents the importance of that term in that e-mail. In this paper, we deal with a binary weighting. The k^{th} document is represented by the characteristic vector $X_k = (a_{1k}, a_{2k}, \ldots a_{Mk})$. $(a_{ik}) = 1$ if the term "i" is present in document "k", 0 otherwise and M is the index size.

[2] http://tartarus.org/~martin/PorterStemmer/

4.2 Performance measures

To evaluate performance we calculated spam precision (SP), spam recall (SR), spam F1 measure (F1) and accuracy. (Shown in equations 1to 4). Let TN: the number of legitimate e-mails classified as legitimate (true negatives), TP: the number of spam emails classified as spam (true positives), FP: the number of legitimate e-mails classified as Spam (False Positives) and FN: the number of spam e-mails classified as legitimate (false negatives), then we have:

$$SP = \frac{TP}{TP+FP} \quad (1) \qquad SR = \frac{TP}{TP+FN} \quad (2)$$

$$F1 = \frac{2\times SP\times SR}{SP+SR} \quad (3) \quad A = \frac{TP+TN}{TP+FP+TN+FN} \quad (4)$$

Weighted accuracy (WA) was also calculated. More formally, WA is defined as follows:

$$WA = \frac{\lambda TN+TP}{\lambda(TN+FP)+TP+FN} \quad (5).$$

Three scenarios are evaluated and compared with previous work:

(a) λ=1; no cost considered;
(b) λ=9; semi-automatic scenario for moderately accurate filter, and
(c) λ=999 completely automatic scenario for a highly accurate filter.

The experiments were performed with a k-fold cross validation with k = 10. In this way, our dataset was split 10 times into 10 different sets of learning sets (90% of the total dataset) and testing sets (10% of the total data). We conduct the training-test procedure ten times and use the average of the ten performances as final result.

4.3 Results and discussion

To evaluate *3CA-1NB* and to show improvement over our previous work, we include the results of experiments on the LingSpam corpus with the CASD classifier using three subsets of features and NB classifier. In Table 2, we reproduce the best performing configuration. These configurations were used as members of the ensemble.

Table 2. Best configurations of NB, CASD and the corresponding performance.

Classifier	***Feature Selector***	***Feature Size***	***SP (%)***	***SR (%)***
NB	IG	500	99,00	82,10
CASD-1	IG	500	98,10	99,02
CASD-2	χ2	500	100	2,5
CASD-3	MI	500	100	44,30

Figure 4 illustrates the ensemble results obtained using the 3CA-1NB classifier alongside those cited above. The results indicate improved performance when classifying with 3CA-1NB. It is clear that the former outperforms individual classifiers in accuracy and F1-measure. We conclude that the proposed ensemble

approach gives better performance than the four base classifiers used separately. The ensemble approach exploits the differences in misclassification by individual classifier and improves the overall performance. We also compare 3CA-1NB with the ensemble approaches developed by [13]. Table 3 reports the best results that we have achieved with 3CA-1NB and which are actually better than the results of [13].

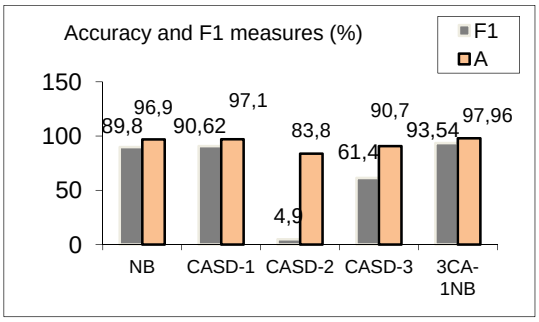

Fig.4. Performance of individual classifiers and *3CA-1NB* on Spam Filtering

Table 3. Performance of stacking and 3CA-1NB on spam filtering.

Classifier	**Performance Measures (%)**				**λ=9**	**λ=999**
	SP	***SR***	***SF1***	***A***	***WA***	***WA***
Stacking[13]	90,80	91,90	91,30	97,10	98,00	98,10
3CA-1NB	98,20	89,36	93,54	97,96	99,37	99,58

5 Conclusion

In this paper a new approach for creating a diversity ensemble of classifiers is proposed. This method uses feature subset selection to train and construct a diversified set of base classifiers. We combine the predictions from the different classifiers by a voting technique in order to increase the performance of spam detection.

The results of experiencing on LingSpam datasets show better performance of the proposed method. As a future perspective, we will investigate the effect of combining more types of classifiers, and also, exploring other combination techniques [11] to further increase accuracy.

References

1. Androutsopoulos, I., Koutsias, J (2000a), "An Evaluation of Naive Bayesian Networks.", In: Machine Learning in the New Information Age. Barcelona Spain (2000) 9-17
2. Androutsopoulos, I., Paliouras, G., Karkaletsis, V.,Sakkis, G., Spyropoulos, C.D., and Stamatopoulos,P. (2000b). "Learning to filter spam e-mail: a comparison of a naïve Bayes-

ian and a memory based approach". In Proc. of the Workshop on ML and Textual Information Access, PKDD 2000, France.
3. Atmani B., Beldjilali B. (2007). Knowledge Discovery in Database: Induction Graph and Cellular Automaton, Computing and Informatics Journal, 26, 171-197.
4. Barigou F., Atmani B., Beldjilali B.: Utilisation de la machine cellulaire pour la détection des courriels indésirables. EGC 2011: 321-322, Revue des Nouvelles Technologies de l'Information, RNTI-E-20.
5. Barigou N, Barigou F, Atmani B., "A Boolean model for spam detection", In: Proceedings of the International Conference on Communication, Computing and Control Applications, Tunisia (2011).
6. Carreras X., Marquez L., (2001), "Boosting Trees for Anti-Spam Email Filtering" in Proc. of RANLP-01, 4th International Conference on Recent Advances in Natural Language Processing.
7. Dietrich T.G., Ensemble methods in machine learning. In: Kittler J., Roli F. (eds), Proc. of 1st Int. Workshop on Multiple Classifier Systems, Springer Verlag LNCS 1857, 2000, 1-15
8. Flavio D. Garcia , Jaap-henk H. , Jeroen van N., "spam filter analysis" in 'Proc. of 19th IFIP International Information Security Conference, 2004
9. Guzella T. S., Caminhas W. M. 2009, "A review of machine learning approaches to spam filtering", Expert Systems with Applications, 36(7), 10206-10222.
10. Koprinska I., Poon J., Clarck J., Chan J. Learning to classify e-mail. Info. S. 177: 2167-2187, 2007.
11. Kuncheva L., Combining Pattern Classifiers, Methods and Algorithms, Wiley Inter Science, 2005.
12. Rios G., Zha H. Exploring support vector machines and random forests for spam detection, in: Proc. First International Conference on Email and Anti Spam (CEAS), 2004.
13. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. D. Spy-ropoulos, and P. Stamatopoulos. Stacking classifiers for anti-spam filtering of e-mail. Proceedings of 6th Conference on Empirical Methods in Natural Language Processing, 1:44-50, 2001.
14. Shih D. H., Chiang S., Lin I. B. Collaborative spam filtering with heterogeneous agents. Expert systems with applications, 34(4), 1555-1566, 2008.
15. Valentini G., Masuli F., Ensambles of Learning Machines. In: R.Tagliaferri, M. Marinaro (eds), Neural Nets WIRN Vietri-2002, Springer-Verlag LNCS, vol. 2486, 2002 , 3-19.
16. Zighed. "Graphe d'induction: Apprentissage et data mining". HERMES, 2000.
17. Yang Y., Pedersen J. O., "A comparative study on feature selection in text categorization", FISHER D. H., Ed., Proceedings of ICML-97, 14th International Conference on Machine Learning, Nashville, US, Morgan Kaufmann Publishers, 412–420,1997.