Advances in Intelligent Systems and Technologies
Proceedings ECIT2004 - Third European Conference on Intelligent Systems and Technologies
Iasi, Romania, July 21-23, 2004

Evolutionary Detection of Rules for Text Categorization. Application to Spam Filtering

Catalin Stoean[1], Ruxandra Gorunescu [2],

Mike Preuss[3], Dan Dumitrescu [4]

[1]*University of Craiova, Romania*
catalin.stoean@inf.ucv.ro
[2] *University of Craiova, Romania*
ruxandra.gorunescu@inf.ucv.ro
[3]*University of Dortmund, Germany*
mike.preuss@cs.uni-dortmund.de
[4]*"Babes-Bolyai" University of Cluj-Napoca, Romania*
ddumitr@cs.ubbcluj.ro

Abstract. *Proposed evolutionary model provides a powerful means of rule discovery in the field of text categorization. In present paper, the model focuses on the particular problem of preventing spam to enter our e-mail accounts. Spam blockers that have been provided by Internet companies so far are not very effective. An application built through the means of our model learns the approximate difference between spam and non-spam mail and labels incoming new mail efficiently. The rules that led to the membership of training mails to either spam or non-spam category are discovered, having been evolved through the principles of a special evolutionary metaheuristics, genetic chromodynamics. Resulting rules are used to establish the appropriate category for previously unseen mails.*

Keywords: *text categorization, spam, rule detection, evolutionary computation, genetic chromodynamics.*

1. Introduction

The Internet represents a source of large amounts of information; it is more and more difficult to find what one is looking for, even when using search engines. A good categorization of the information appears to be of great help for users that are looking for documents in a specific category. Moreover, as everyone is using electronic mail now, we all have one important problem: commercial e-mails, undesired e-mails, also called *SPAM*. How do we get rid of these e-mails?

Text categorization comes to help us in both organizing information and e-mail filtering. Having a set of predefined categories, text categorization establishes which of them can be a label for a first time seen document with respect to its content.

In present paper, the specific problem of categorizing e-mails is dealt with. There are two predefined categories, *spam* and *non-spam* e-mails (also called *ham*). Based on the content of one e-mail, its type, spam or non-spam, has to be decided.

Proposed model first extracts the most important keywords to the two considered categories from a set of training e-mails. Then, rules that refer to the considered keywords are built through the use of a special evolutionary metaheuristics, genetic chromodynamics. These rules are then applied to e-mails in the test set and their outcome is decided.

2. Proposed technique

The model comprises of three important stages. The first one deals with extracting the most important keywords specific to each of the two categories; in this respect, weights of words in e-mails from the training set are computed.

The second stage, the most important one, uses the field of evolutionary computation to produce the classification rules that were used in the decision making process.

Finally, the model predicts the type for e-mails in the test set and compares the obtained results with the outcomes e-mails had for real.

2.1. Keywords extraction

An e-mail is regarded as a bag of words; an important problem to be solved lies in the obtaining of a subset of these words that can *characterize* that e-mail and, implicitly, the category it has assigned.

In conclusion, special keywords have to be extracted for each one of the two categories; they are, of course, taken from the e-mails in the training set. For that, weights of every word will be computed for each category with respect to the weights of the considered words in the other category.

2.1.1. Preprocessing step

First of all, a preprocessing step is necessary. That means that each e-mail in the test collection will suffer some modifications:

- Punctuation is removed;
- *HTML* tags are taken away;
- No difference is made between small and capital letters;
- Finally, each remaining word is reduced to a word-stem.

This preprocessing step is absolutely necessary, but it is a very difficult task, since spam e-mails that contain *HTML* tags are very much *coded*. That means they are not written quite correctly: a lot of unnecessary tags are used, many tags split important keywords for spam only on the HTML code level, and not on the user level (where they appear normally) etc. In present model, the preprocessing that concerns the removal of *HTML* tags could be improved, as there are some *HTML* elements that escaped preprocessing.

The reduction of words to word-stems has the goal of representing in the same manner words that have the same meaning: *remove*, *removing*, *removal* or *removed*, for instance, all of them can be represented as *remov*. Word-stems will be called *terms* further on in this paper.

The preprocessing task has to be applied to all e-mails, be that they belong to either the training or the test set.

2.1.2. Representation of e-mails in the test collection

Each e-mail is represented as two vectors: one that contains all terms obtained after the preprocessing, terms taken only one time, and another vector that gives the number of times each term appears in the preprocessed e-mail ([5], [6]).

As the size of one e-mail does not influence its type (spam or non-spam), normalization has to be done ([5], [6]): each of the values in the vector that contains the terms occurrences will be divided by the number of all terms the e-mail has; the values of this vector will be called *weights* further on.

This representation has to be obtained for all e-mails, belonging either to the training or test set.

2.1.3. Representation of the two categories

For each of the two categories, a similar representation to the one e-mails have is obtained ([5], [6]). The vector of terms specific to one category includes all terms which appear in all e-mails in the training set that have the specified category assigned, terms taken only one time.

The other vector for the same category, the one that gives us the weights for terms in the first vector, is computed by summing, for each term, all weights of that term in each e-mail from the training set, e-mail that has that category assigned ([5], [6]).

In conclusion, each of the two categories will be represented as two vectors:

$$\mathbf{WS} = (ws_1, \ldots, ws_p) \quad \text{and} \quad \mathbf{OS} = (os_1, \ldots, os_p) \qquad (1)$$

for spam and

$$\mathbf{WH = (wh_1, \ldots, wh_q)} \quad \text{and} \quad \mathbf{OH = (os_1, \ldots, os_q)} \quad \mathbf{(2)}$$

for non-spam (*ham*) category.

Of course, the number of common terms of the two categories is high: these terms have to be penalized, as they are not very much specific to only one of the two categories. That is the reason why their weights are modified for each category with respect to the weights of the same terms in the other category ([6]):

$$os_j' = \frac{os_j}{1 + oh_k} \qquad oh_k' = \frac{oh_k}{1 + os_j} \qquad \mathbf{(3)}$$

where $ws_j = wh_k$, $j = 1,\ldots, p$ and $k = 1,\ldots, q$.

The most important n terms are now extracted from each of the two categories: for a category, they are taken to be the n terms that have the highest values for their weights with respect to the weights of all terms in the considered category.

In conclusion, $2n$ terms (keywords) are extracted, n from spam and n from the non-spam category. From this point on, e-mails in the training and test set will be represented by vectors referring only these $2n$ terms.

2.2. Defining the rules

Present model puts together two important heuristics of the field of evolutionary computation (EC): the representation of the rules is taken from the classical Michigan evolutionary learning classifier and the engine relies on the specific mechanisms of Genetic Chromodynamics (GC) ([1]); a similar model was used in [2] in modeling a multidisciplinary review panel for admission of elderly people to long-term care.

In the Michigan approach, each chromosome represents one individual rule and the entire set of chromosomes represents the set of all rules.

In the end of the heuristics, one is no longer interested in the fittest chromosome, as in standard evolutionary ones, but rather in a whole population of classification rules. Thus, the final population has to consist of multiple non-homogenous rules, each optimal for its outcome.

Present model offers a simpler alternative to the credit assignment system proposed by Holland in the Michigan approach in order to solve this multimodal optimization problem. And proposed replacement is GC, since it has many times proven to be a very efficient way in determining multiple optima points.

Therefore, the combination between the Michigan approach and the GC engine seems like a good match in the domain of evolutionary learning classifier systems ([2]).

2.2.1. Generalities of Genetic Chromodynamics

GC ([1]) forms and maintains stable subpopulations that co-evolve and become better separated with each iteration and lead, at convergence, each to an optimum. First, the population number is high and, as the algorithm runs, the number of chromosomes can be reduced with each generation.

GC uses a stepping-stone search mechanism in connection with a local interaction principle. By using the stepping-stone mechanism, each individual participates in the forming of the new generation: its mate is found by applying a local selection scheme; if a second chromosome is found within its local range (called the *mating region*), they recombine and the competition for survival of the fittest is held between the resulting offspring and the first parent only. If no chromosome can be found in the local range of the considered chromosome, it will be mutated.

A special operator is introduced that merges very similar chromosomes into one chromosome that can be considered either the fittest or the mean of the chromosomes to be merged.

2.2.2. Genetic Chromodynamics approach to the spam filtering problem. The GCSF model

As already mentioned before, n terms (keywords) are taken from each of the two categories, therefore there are $2n$ terms with their corresponding weights, in total.

For further reference, let m be the total number of e-mails in the training set and let v be their set, $v = \{v_1, \ldots, v_m\}$.

GCSF representation

As stated, each chromosome c represents a rule; it is represented as a list of real numbers of the following form:

$$(a_1, \ldots, a_{2n}) : b, \qquad \textbf{(4)}$$

where $a_1, \ldots, a_n$ represent the weights of the spam keywords and $a_{n+1}, \ldots, a_{2n}$ the weights of the ham keywords. b is either 0 (meaning *spam*) or 1 (meaning *ham*). A chromosome has the same structure an e-mail in the training set has.

A chromosome gives us the threshold behind which a preprocessed e-mail can be labeled as either spam or ham.

Initial population

The initial population represents the initial set of rules. Based on experimental results, the initial population size was considered equal to 100: by having taken a smaller size, a premature convergence was achieved and with a larger size, no significant improvement was reached.

The values of the genes were obtained by generating random numbers in the interval 0 to the average of weights the terms in the training set had; this interval will be further on denoted by *D*. The average appeared to be a better choice than the upper extremity of the domain because there were very few maximum values; these would have made the GC algorithm search also in spaces that contained high values and thus consume too much time just to conclude in the end that these values could not have become thresholds of decision.

Fitness assignment

First of all, the distance between two entities of the form (4) has to be set; having $x = (a_1, \ldots, a_{2n}) : b_1$ and $y = (c_1, \ldots, c_{2n}) : b_2$, where b_2, $b_2 \in \{0, 1\}$, the distance between x and y is the Manhattan distance:

$$d(x, y) = \sum_{i=1}^{2n} | a_i - c_i | \qquad \textbf{(5)}$$

The fitness evaluation for a given chromosome c minimizes the distance between the weights of that chromosome and the weights of the selected keywords from e-mails in the training set that have the same label as the chromosome c. At the same time, the distance between the weights of c and the weights of the keywords from e-mails in the training set that have the opposite label with respect to the chromosome c is maximized.

Suppose we apply the fitness function to a chromosome $c = (a_1, \ldots, a_{2n}) : b$. Be that there are u e-mails in the training set that are labeled with b, the following multi objective problem is obtained:

$$f_1 : D^{2n} \to R, \; f_1(c) = \frac{\sum_{i=1}^{u} d(c, v_i)}{u} \qquad \textbf{(6)} \text{ to be minimized}$$

$$f_2 : D^{2n} \to R, \; f_2(c) = \frac{\sum_{i=u+1}^{m} d(c, v_i)}{m - u} \qquad \textbf{(7)} \text{ to be maximized,}$$

The combination of the objective functions f_1 and f_2 in a unique criterion function is proposed:

$$f(c) = f_1(c) + \frac{1}{f_2(c)} \qquad \textbf{(8)}$$

The goal now is to minimize the function in (8).

Selection operator

As mentioned in section 2.2.1., the mate for every chromosome is selected within its local range. Proportional selection is used in this respect.

Variation operators

Traditional operators are used: convex crossover and gene oriented mutation.

An additional variation operator, merging, is considered. Each chromosome in the current population is taken into account; all chromosomes that have a degree of similarity with it higher than a given threshold (called the *merging radius*) are also considered. Only one chromosome from all these is kept in the next generation - the best one from the group with respect to its fitness value. All the others chromosomes are deleted from the population.

Stop condition

The algorithm stops when, after a predefined number of iterations, no new offspring is accepted in the population.

The last population contains the optimum set of rules, both in number and value.

Parameter values

The values for the parameters used by proposed model for experiments are presented in Table 1:

Mating region	**Mutation step size**	**Merging radius**	**Number of iterations without any change**
0.6 * (2*n*)	0.06	0.4 * (2*n*)	100

TABLE 1. GCSF parameter values

The mating region is considered so that the difference between the values of two chromosomes for each gene (keyword) be no higher than 10% of the maximum possible difference between them.

The mutation step size is in connection to the mating region, as it is compulsory that the offspring does not fall out of the range of its parent.

The merging radius is chosen so that the difference between the values of two chromosomes for each gene (keyword) be no higher than 6% of the maximum possible difference between them.

2.2.3. Interpretation of the obtained rules

The final population contains at least two chromosomes, one for each of the two categories. Therefore, at least two rules are finally obtained, one for each category.

If there is more than one rule for a certain category then, when applying the rules for an e-mail, at least one of them needs to be satisfied so that the e-mail be labeled with that category.

Consider a chromosome in the final population, with the outcome 1, representing thus a rule for non-spam e-mails. The chromosome has the following representation:

$$x = (a_1, a_2, \ldots, a_n, a_{n+1}, \ldots, a_{2n}) : 1$$

As the first n genes contain weights of the keywords for spam, only the last n weights are of interest. The rule gives us some minimum values for the weights of the keywords that have to be overtaken by the weights of the same keywords in an e-mail from the test set, so that to label that e-mail with ham.

It is not always the case that every single keyword appears in an e-mail from the test set. This is the reason one can not compare each value of the weights of the keywords in a rule with each of the values of the weights of these keywords in an e-mail from the test set. Alternatively, the following sum will be computed for the non-spam rule:

$$h_1 = \sum_{i=n+1}^{2n} a_i$$

All weights of the non-spam keywords that appear in an e-mail from the test set are also summed; if the obtained sum (denoted by h) is higher then h_1, then the e-mail is concluded to be a non-spam one. If there are more rules for non-spam e-mails, other sums h_2, h_3, …, h_j, are computed. In conclusion, for an e-mail to be labeled as non-spam, its sum h has to be higher than at least one of the h_j-s.

Same goes for spam labeling, but this time taking into account only the first n genes.

For all e-mails in the test set, both types of rules are applied and they are all labeled as either spam or ham.

2.3. Experimental results

The experiments were conducted on 1000 e-mails, 500 for each category: 668 were used for training (334 spam and 334 non-spam) and 332 (166 spam and 166 non-spam) for testing. Training and test sets are disjoint. The model used e-mails from the test collections available at http://spamassassin.org/publiccorpus.

By applying the obtained rules to the test set, for n equal to 15, 96.4% of all e-mails were correctly classified; as one would be interested in how many of these e-mails were from spam and how many were non-spam, the results for each category proved that:

- 97.6% of the good e-mails were correctly classified
- 95.2% of spam correctly classified
- 3.6% of the e-mails were not classified at all

The rules are very good since none of the e-mails in the test set were classified both as spam and non-spam.

The fact that the failed e-mails were not labeled in any way is a very good aspect, mainly because no good e-mail was labeled as spam.

Obviously, everyone would prefer the presence of spam e-mails in our account to the loss of good e-mails to spam: this is the reason it was decided to label the non-classified e-mails as non-spam. In this manner, none of the good e-mails would be lost and the percents would look like this:

- 100% of good e-mails correctly classified
- 95.2% correctly classified from spam.

Therefore, this offers an overall result of 97.6% correctly classified e-mails.

3. Conclusion and future work

Present paper treats the problem of text categorization and, as an application to it, it was chosen to build a spam filter. The model uses a machine learning engine so, if spammers changed entirely the texts of spam e-mails, the filter would find other keywords and rules that would automatically detect the commercial e-mails.

Comparisons to other models are presented further on.

One of the most often used methods in fighting spam is represented by the Naïve Bayes technique ([3]): impressive results were obtained by Graham in its *plan for spam* – for each word, probabilities to belong to each of the two categories are computed and then used to label a first time seen e-mail. The experimental results showed that the Bayes model obtained 100% for good e-mails and 95% for spam e-mails. Unfortunately, the results can not be directly confronted, as present model and Bayes model did not use the same test collection. In [3], the author's collection of spam and non-spam e-mails was used; this can prove to be a big advantage, since most of the good e-mails had some common patterns, while the test collection used by present model contains e-mails collected from many e-mail users.

In [4], a model which uses only 536 e-mails taken from the same test collections from where we also draw our 1000 mails, 86.7% of mail messages were correctly classified.

The results obtained in present paper were, no doubt, better then those obtained in [6], where the exact same test collection as here was used again. In [6], 98% of good e-mails and only 79% of spam e-mails were correctly classified.

As spam e-mails contain more HTML tags than non-spam e-mails and, at the same time, obviously more classification problems appear with the labeling of spam e-mails, we believe a solution for a more accurate categorization relies in a better HTML processing.

References

[1]. D. Dumitrescu, Genetic Chromodynamics, Studia Universitatis Babes Bolyai, Ser. Informatica, 39 - 50, 2000

[2]. R. Gorunescu, P. H. Millard, An Evolutionary Model of a Multidisciplinary Review Panel for Admission to Long-term Care, ICCC 2004, Baile – Felix, Oradea, 2004, p. 181 - 185.

[3]. P. Graham, A plan for spam, http://www.paulgraham.com/spam.html, August 2002.
[4]. T. Nicholas, Using AdaBoost and Decision Stumps to Identify Spam E-mail, Stanford Natural Language Processing Group, CS224N/Ling237, 2003.
[5]. C. Stoean, Hierarchical Learning Text Categorization, ICCC 2004, May 27-29, 2004 Baile Felix Spa-Oradea, Romania, 2004, p. 383-387.
[6]. C. Stoean, A New Technique for Text Categorization. Application to Spam Filtering, Proceedings of Zilele Academice Clujene 2004 (to appear).