An evolutionary learning spam filter system

Catalin Stoean[1], Ruxandra Gorunescu[2], Mike Preuss[3], D. 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. Accurate spam filters are of high necessity in present days as the high amount of commercial mail entering accounts has become a real threat to everyone, from causing personal computers to crash to costing big companies billions of dollars annually because of employees loss of productivity. Moreover, lately, spam also carries viruses along. Current paper presents an evolutionary model of a spam filter which can also be generalized to the more complex issue of text categorization. The model learns the rules that lie behind the classification of training e-mails into spam and non-spam and then uses them to label unseen, incoming mail. The evolutionary learning classifier system uses genetic chromodynamics to evolve the rule set. A comparison of its correctness of prediction depending on whether chromosomes representing rules are binary or real encoded is conducted. Experimental results showed that the binary encoding appeared to be less effective than the real one.
Keywords: spam, classification, rule discovery, evolutionary computation, genetic chromodynamics

1 Introduction

As everyone uses electronic mail, one frequently encounters a major problem: unsolicited, commercial e-mails, also called *spam*. According to a statement from an EU official who deals with spam ban enforcement, there are currently around 48% spam mails in the EU; forecasts say this number will rise above 50% during summer.
Spam e-mails represent big trouble for common Internet users and produce major loss for companies. There are laws against spam, but nobody knows how to enforce them. It is up to the Internet users to get rid of this problem. Thus, good filters have to be built for labelling e-mails, based on their content, as being spam or non-spam.
Very good filters that have been built so far generally use Naive Bayes engines, decision trees or pure machine learning techniques. In present paper, a learning classifier system for text categorization that uses techniques from the field of evolutionary computation is built. It is probably the first attempt to use this field in the problem of fighting spam.

Chromosomes are represented as binary and real in turn. Text categorization is treated in the particular situation when documents are represented by e-mails and there are only two categories, spam and non-spam. The evolutionary learning classifier system regards rules as in the famous Michigan approach to learning classifier systems but it replaces the credit assignment system with a multimodal optimization engine, called genetic chromodynamics. The two different chromosome representations are compared as to their effect on the percentage of correctly classified e-mails from the test set after applying the resulting rules of the evolutionary heuristics.
The paper is structured as follows. In subsection 2.1, a set of keywords for each of the two categories is extracted from e-mails in the training set. Subsection 2.2 presents the learning classifier system. It builds the rules that classified e-mails in the training set to either of the two categories. Subsection 2.3 discusses some general concepts of genetic chromodynamics. Section 3 describes the representation, initialisation of the population, fitness assignment, selection, variation operators and interpretation of resulting rules when chromosomes have real encoding. Section 4 presents the same components with respect to binary encoding. Section 5 compares the experimental results of the application of the obtained rules on the test set for each of the two representations. Thus, it can be seen how the choice of representation can influence the accuracy of the filter.

2 Proposed model

The model first extracts the most important keywords for each of the two categories, based on their weights in training e-mails, and then uses both keywords and weights in the evolutionary process of building the rules. The evolutionary heuristics uses first a real and then a binary encoding for chromosomes representing rules. It maintains their diversity, leading at convergence to multiple optima, through the principles of genetic chromodynamics (GC). Resulting rules are next used to label test e-mails and a comparison between the influence of the two proposed representations of chromosomes on the percentage of correctly classified cases is made.

2.1 Extraction of keywords

The keywords are extracted from the e-mails in the training set. For a category, they are taken to be words that appear often in e-mails from that category and rarely in e-mails from the opposite category. To accomplish that, weights for each word appearing in e-mails from the training set with respect to their categories (spam or non-spam) are computed.
First of all, some preprocessing needs to be done to all e-mails, be that they are from the training or test set; that means punctuation and HTML tags are removed and each remaining word is reduced to a word-stem.

Each e-mail will be further on represented as two vectors, one containing the word-stems remained after preprocessing, taken only one time, and the other one containing the number of occurrences of each word-stem in that e-mail, divided by the total number of word-stems the considered e-mail has ([8], [9]). This representation has to be obtained for all e-mails in the test collection.
Next, we have to compute how *important* the word-stems are for a category with respect to the other category; for each category, all word-stems in e-mails from the training set that have that category assigned are gathered in a large vector of word-stems. The weights of each word-stem from these e-mails are summed and thus, the second vector, containing weights for word-stems, is obtained.
Same representation has to be obtained for both categories, therefore the following vectors are constructed ([8], [9]):

$$WS = (ws_1, ..., ws_p) \text{ and } OS = (os_1, ..., os_p)$$

for *spam*, and

$$WH = (wh_1, ..., wh_q) \text{ and } OH = (oh_1, ..., oh_q)$$

for $non - spam$.
As the number of common word-stems of the two categories is high, their weights have to be penalized, as these word-stems are not very much specific to only one of the two categories ([9]):

$$os'_j = \frac{os_j}{1 + oh_k} \qquad oh'_k = \frac{oh_k}{1 + os_j} \tag{1}$$

where $ws_j = wh_k$, $j = 1, ..., p$ and $k = 1, ..., q$.
In present model, either if real or binary encoding is used, only the most important n word-stems for each of the two categories are considered: for each category, they are taken to be the n word-stems that have the highest values for their weights with respect to the weights of all word-stems in that category. Therefore, $2n$ word-stems are considered, n from spam and n from the non-spam category. These *keywords* will be used further on to represent e-mails in the training and test set [10].

2.2 Learning spam classifier system. GCSF model

Proposed model is a particular application of a new evolutionary approach to learning classifier systems.
So far, the evolutionary computation community has approached learning classifier systems from two different viewpoints. The first direction is represented by the Pittsburgh school. They developed an evolutionary system that considers a chromosome to represent an entire rule set. Rule sets are evolved by using a standard evolutionary heuristics and the best chromosome from all generations represents the solution of the classification problem ([6]).

The opposite direction is the work of the Michigan school. Here, each chromosome represents only one rule and the entire population represents the rule set. The problem with this approach lies in the fact that a standard evolutionary heuristics cannot evolve non-homogeneous chromosomes (rules). Thus, the Michigan approach suggested a mechanism that would assign positive credit to rules that cooperate and negative credit otherwise. This engine is called the credit assignment system ([6]).
The new approach takes the viewpoint of the Michigan school regarding representation, but replaces the credit assignment system by a simpler engine, offered by GC, which proved to be very efficient in detecting multiple optima ([3]).
This approach is used here in building a model of discovery of adaptive rules for spam detection. For a similar particular model, based on the same approach, which was used for rule discovery in the admission of elderly patients to long-term care see [4].
The choice of a proper representation of chromosomes proved to be of high importance for the effectiveness of the model. A real encoding ([10]) versus a binary encoding are studied further on.

2.3 Genetic Chromodynamics

GC ([3]) forms and maintains stable subpopulations that co-evolve and become better separated with each iteration and lead, at convergence, each one to an optimum. The heuristics begins with a large initial population which may be reduced at each generation. It uses a stepping-stone search mechanism, which means that each chromosome takes part in forming the new generation. The mechanism works in connection with a local interaction principle, meaning that its mate is found by applying a local selection scheme. If a second chromosome is found within its local range, they recombine and the competition for survival of the fittest is held between the resulting offspring and the first parent only. If there are no such chromosomes, then the current chromosome is mutated. A special operator is introduced, that merges very similar chromosomes into a single one, either the fittest or the mean.

3 GCSF model with real encoding for chromosomes ([10])

Let m denote the number of e-mails in the training set and let v be their set, $v = \{v_1, ..., v_m\}$. Furthermore, let D be the interval from 0 to the average weight of the word-stems in the training set.

GCSF real representation

Each chromosome c represents a rule; it is encoded as a list of real numbers of the following form:

$$(a_1, ..., a_{2n}) : b, \tag{2}$$

where $a_1, ..., a_n$ represent the weights of the spam keywords and $a_{n+1}, ..., a_{2n}$ the weights of the non-spam keywords. b represents the outcome and thus it is either 0 (spam) or 1 (non-spam). A chromosome is encoded thus in the same manner as all e-mails in the test collection, after keywords extraction.
A chromosome gives the threshold behind which an e-mail can be labeled as either spam or non-spam.

Initial population

The initial population represents the initial set of rules. Let us denote by s its initial size.
The values of the genes were obtained by uniformly generating random numbers in D. Taking the average of weights of the word-stems as the second extremity of the interval D appeared to be a better choice than the upper bound of the domain of weights, because there were very few maximum values; these would have made the system search also in those few spaces that contained high values and thus consume too much time just to figure out in the end that these values could not have become thresholds of decision.

Fitness assignment

Let $x = (a_1, ..., a_{2n}) : b_1$ and $y = (c_1, ..., c_{2n}) : b_2$, where $b_1, b_2 \in \{0, 1\}$ two entities of the form (2). Manhattan distance was chosen to compute the difference between x and y:

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

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

Let $c = (a_1, ..., a_{2n}) : b$ be the current chromosome.

Suppose there are u e-mails in the training set that are labeled with b. The following multi-objective problem has to be solved (P):

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

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

Problem (P) is solved through combining the objective functions f_1 and f_2 in a unique criterion function:

$$f_c(c) = f_1(c) + \frac{1}{f_2(c)} \tag{3}$$

One is led now to the minimization of the function in (3).

Mating selection operator

The mate for every chromosome is selected within its predefined local range, called *the mating region*. Proportional selection is used in this respect ([1], [2]).

Variation operators

Convex crossover is chosen for the first operator ([1], [2]).
As mutation is concerned, a gene oriented one is used. The gene to be mutated suffers a normal perturbation:

$a_i = a_i \pm ms \cdot N_i(0,1)$, where $i = 1, 2, ..., 2n$ and ms denotes the *mutation strength*.

Mutation does not apply to the gene representing the outcome.

Merging operator

Merging is an additional variation operator. All chromosomes that are alike behind a given threshold, called *the merging radius*, are taken into consideration. The best chromosome from the group with respect to its fitness value replaces all the others in the next population.
The merging operator does not take into account the outcome. Now, it is possible, in the early generations, for chromosomes that are very similar to have different outcomes. As the fitness evaluation takes into account the quality of the chromosome (rule) for the classification task, be that the above mentioned situation happens, only the chromosomes with the proper outcome for the given weights will survive.

Stop condition

The algorithm stops when, after a predefined number of iterations, denoted by t, no new offspring is accepted in the population.
The last population gives the set of rules optimal in number and each rule optimal for its corresponding outcome.

Parameter values

The values for the parameters involved are given in Table 1. These values were experimentally chosen so that GC principles obey.

Mating region	ms	Merging radius	t	s	n
0.6 * 2n	0.06	0.4 * 2n	100	100	15

Table 1. GCSF parameter values for real encoding

The mating region is considered in this way 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 strength 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 as it is 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.

Resulting rules. Interpretation

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:

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

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 in order to label that e-mail with non-spam. It is not always the case that every single keyword appears in an e-mail from the test set. This is the reason one cannot 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 than 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 also 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 each one of them is labeled as either spam or non-spam.

4 GCSF model with binary encoding for chromosomes

Let again m denote the number of e-mails in the training set and let v be their set, $v = \{v_1, ..., v_m\}$.
Here, e-mail representation also changes in the following manner. If a keyword appears in an e-mail, then that keyword is represented as 1; if not, its value is 0. Therefore, each e-mail from the test collection will be further on represented as a binary vector of 0 and 1 values.

GCSF binary representation

Each chromosome c represents a rule again; it is encoded as a list of binary digits of the following form:

$$(a_1, ..., a_{2n}) : b, \tag{4}$$

where a_i, $i = 1, 2, ..., n$, is 1 if that spam keyword is important for the current rule and 0 otherwise. The interpretation goes the same for the non-spam keywords, a_i, $i = n + 1, ..., 2n$. b represents the outcome again and thus it is either 0 (spam) or 1 (non-spam). A chromosome has this time, as well, the same structure as every e-mail in the collection after keywords extraction.
A chromosome gives the keywords that are important to a rule in order to label an e-mail as either spam or non-spam.

Initial population

The initial population represents the initial set of rules. Its size is again denoted by s.
The values of the genes were obtained by uniformly generating random digits in the $\{0, 1\}$ set.

Fitness assignment

The fitness evaluation of a chromosome c is regarded in the same manner as in the case of real encoding. Hamming distance was chosen to compute the difference between entities of the form (4).

Mating selection operator

The stepping-stone principle applies as in the real representation.

Variation operators

One point crossover and strong mutation are used ([1], [2]). Mutation does not apply to the gene representing the outcome.

Merging operator

If two chromosomes are similar behind the merging radius, they are replaced in the next generation by the fittest.

Stop condition

The algorithm stops as well when, after a predefined number of iterations, denoted again by t, no new offspring is accepted in the population.
The last population gives one more time the optimal set of rules, both in number and value.

Parameter values

The values for the parameters involved are given in Table 2. These values were experimentally chosen so that GC principles obey.

Mating region (= d)	Mutation probability	Merging radius	t	s	n
[2n/6]	2d/1000 - 0.001	[(2n - 1)/20] + 1	100	100	10

Table 2. GCSF parameter values for binary encoding

Some differences between the real encoding and the binary encoding arise at this point.
When difference between two chromosomes in the real encoding is computed, it is calculated as a Manhattan distance on a continuous interval; contrary, in the binary case, it is dealt with a discrete interval. That's why the mating regions in the two encodings have also to be different with respect to this reason. In the real encoding, the distance is computed relative to the difference between weights of every attribute, while the distance in the binary encoding is regarded so that the two chromosomes be not different from one another in more than one sixth of the binary values of attributes.
We extracted a maximum number of keywords for each category equal to 15 in both encodings. While in the real encoding, the correct percentage of classified cases grows as n approaches 15, reaching the maximum percent at 15, in the binary one, as n approaches 10, the filter becomes more accurate and more stable, reaching the fullest at 10, and then goes ineffective and unstable again - apparently, $n = 10$ offers a good equilibrium.

Resulting rules. Interpretation

In the final population, at least two chromosomes, one for each of the two categories, are obtained. For each e-mail in the test set, one will compute which type of rule can be better applied.

As there is usually more than one rule obtained for each category, the number of unmatched values with each rule corresponding to that category is computed for every e-mail in the test set. The minimum value of these numbers is retained. Therefore, for an e-mail, there are two minimums - one obtained with respect to the rules for spam and the other one with respect to the rules for non-spam.
An e-mail is said to be non-spam if the minimum value obtained for spam is higher than the other minimum, which means there are more unmatched values between the considered e-mail and the rules for spam. If the two minimum values are equal, the e-mail is considered to be good (non-spam) because one would rather receive spam into his/her account than lose good e-mails to spam.

5 Comparison on experimental results

The data on which the experiments were conducted comprises of 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.
The rules obtained through each encoding, at least two - one for each outcome - in each case, were applied to the test set and results after ten runs are shown in Table 3:

	Real encoding			Binary encoding		
	Non-spam	Spam	Overall	Non-spam	Spam	Overall
Mean	97.65	94.94	96.29	84.99	77.71	81.32
Standard deviation	0.19	0.70	0.32	0.0	0.0	0.0

Table 3. GCSF model: accuracy rates after ten runs

When real encoded, the rule set correctly predicted the category for 96.29% of all e-mails, and, differentiating, rightly categorizing 97.65% of non-spam and 94.94% of spam, giving a failure of 3.71% e-mails which were not placed anywhere.
The classification task was achieved very well as 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 his/her 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 change to 100% non-spam correctly classified and 94.94% spam correctly categorized. Therefore, this leads to an overall result of 97.47% correctly classified e-mails.
When binary encoded, the system correctly labeled only 81.32% of the test e-mails, 84.99% non-spam and 77.71% spam, respectively. There were no not placements because of the particular interpretation of the resulting rules.

These results indicate that real encoding, which obviously carries much more information than the binary one, although not as simple to work with, provides a far more accurate, stable and reliable categorization.

6 Conclusions and future work

Present paper proposes an evolutionary learning classifier system for spam detection. The model achieves an almost perfect classification result. Moreover, it can adapt to non-stationary conditions, that is, if spammers changed entirely the texts of spam e-mails, the filter would find other keywords, rules would be evolved again and automatically detect the commercial e-mails.

A comparison to what chromosome representation would be better suited for the model was performed. Real encoding proved to be more accurate for the given purpose.

In this respect, the model gave a high percentage of correct prediction of 100% of non-spam correctly classified and 94.94% spam correctly placed, achieving an overall result of 97.47% correctly classified e-mails.

A few comparisons to the other, non-evolutionary, models were conducted. One of the most often used methods in fighting spam is represented by the Naive Bayes technique ([5]): 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 [5], 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 [7], a boosting algorithm with decision stumps which uses only 536 e-mails taken from the same test collections from where we also draw our 1000 mails, correctly classified an overall percentage of 86.7% of mail messages. No information is given as to how many e-mails for each category, specifically, were correctly placed.

The results obtained in present paper were, no doubt, better than those obtained in [9], where the exact same test collection as here was used again and a pure machine learning engine performed the classification task. In [9], 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, a solution for a more accurate categorization might rely on a better HTML processing.

The impact of a size for n higher than 15 is to be studied in future work. Moreover, the study of the ideal size for n has to take into account not only accuracy but also the amount of computing time needed.

References

1. Dumitrescu, D., *Genetic Algorithms and Evolution Strategies*, Blue Publishing House, Cluj - Napoca, 2000
2. Dumitrescu, D., Lazzerini, B., Jain, L., C., Dumitrescu, A., *Evolutionary Computation*, CRC Press, Boca Raton, Florida, 2000
3. D. Dumitrescu, Genetic Chromodynamics, Studia Universitatis Babes Bolyai, Ser. Informatica, 39 - 50, 2000
4. 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.
5. P. Graham, A plan for spam, http://www.paulgraham.com/spam.html, August 2002.
6. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 2nd edition, Springer - Verlag, 1992
7. T. Nicholas, Using AdaBoost and Decision Stumps to Identify Spam E-mail, Stanford Natural Language Processing Group, CS224N/Ling237, 2003.
8. C. Stoean, Hierarchical Learning Text Categorization, ICCC 2004, May 27-29, 2004 Baile Felix Spa-Oradea, Romania, 2004, p. 383-387.
9. C. Stoean, A New Technique for Text Categorization. Application to Spam Filtering, Proceedings of Zilele Academice Clujene 2004 (to appear).
10. C. Stoean, R. Gorunescu, M. Preuss, D. Dumitrescu, Evolutionary Detection of Rules for Text Categorization. Application to Spam Filtering, Advances in Intelligent Systems and Technologies, Proceedings ECIT 2004, Third European Conference on Intelligent Systems and Technologies, Iasi, Romania, July 21 - 23, 2004 (to appear).