SVM Classifier Incorporating Feature Selection Using GA for Spam Detection

WANG Huai-bin [1] , YU Ying [2] , LIU Zhen [3]

[1]Dept. of computer science, Tianjin University of Technology,
300191, Tianjin, China
whb@tengda.net

[2] School of Computer Science and Technology, Harbin Institute of Technology,
150001, Harbin, China
yuying@insun.hit.edu.cn

[3] Nagasaki Institute of Applied Science, Japan
536 Aba-machi, Nagasaki 851-0193,Japan
liuzhen@cc.nias.ac.jp

Abstract. The use of **SVM** (Support Vector Machines) in detecting e-mail as spam or nonspam by incorporating feature selection using GA (Genetic Algorithm) is investigated. An GA approach is adopted to select features that are most favorable to **SVM** classifier, which is named as GA-SVM. Scaling factor is exploited to measure the relevant coefficients of feature to the classification task and is estimated by GA. Heavy-bias operator is introduced in GA to promote sparse in the scaling factors of features. So, feature selection is performed by eliminating irrelevant features whose scaling factor is zero. The experiment results on UCI Spam database show that comparing with original **SVM** classifier, the number of support vector decreases while better classification results are achieved based on GA-SVM.

Key words: Support Vector Machines **(**SVM); Genetic Algorithm (GA); Feature selection; Spam detection

1. Introduction

Spam is defined as junk e-mail message that is unwanted delivered by the internet mail service. There are various methods to classify e-mail [1,2,3] .Technical solutions to detect spam include filtering based on sender address or header content. The problem with filtering is that a valid message may be blocked sometimes. Rather, we distinguish whether an e-mail is spam according to features. The selection of features is flexible in deferent scenario, such as percentage of words in the e-mail that match specified word, percentage of words in the e-mail that match specified character, average length of uninterrupted sequences of capital letters etc.

In this paper, **SVM** is utilized to detect spam by incorporating feature selection using GA. As a new approach of pattern recognition, **SVM** [4] is based on Structural Risk Minimization (SRM). It is suitable to deal with magnitude features problems with a given finite amount of training data. Even so, there may be a large number of

available features to be redundant, noisy or unreliable in the training data to deteriorate the success of **SVM** strongly. So feature selection is necessary to improve the performance of **SVM** for more pleasing results.

The approach for feature selection can be divided into wrappers, filters and embedded methods essentially[5,6]. In this paper, feature selection and **SVM** classifier design are performed simultaneously in the framework of embedded methods, which is named as GA-SVM.

The rest of the paper is organized as follows. Section 2 reviews the **SVM** with feature scaling parameter and related algorithm on how feature relevance scalings are derived from the **SVM** methodology. In Section 3, details of the proposed algorithm are introduced. In section 4, numerical experiments on UCI real-world data show the results of our proposals. And Section 5 is the conclusion arisen from this work.

2. SVMs with Feature Scaling Parameter

Given l independent and identically distributed examples

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_l, y_l), \quad \mathbf{x}_i \in \mathbf{X} \subset R^n, \quad y_i \in Y = \{1, \cdots, k\} \tag{1}$$

The core desired solution of **SVM** is to find optimal hyperplane between two classes which can separate two different classes with the maximal distance, the standard **SVM** require the solution of the following convex Quadratic Programming (QP) optimization problem:

$$\begin{aligned} &\underset{w,b}{Minimize} \quad \Phi(w,b) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i \\ &subject\ to \quad y_i\left(w^T \varphi(x_i) + b\right) \geq 1 - \xi_i,\ \xi_i \geq 0,\ i = 1, 2, \cdots, l \end{aligned} \tag{2}$$

$C > 0$ is the penalty parameter to control the trade-off between maximizing the margin and minimizing the training error term ξ_i.

Scaling factor is a discrete real-valued weight that is associated with each feature to measure feature's importance. There are two prospective benefits of this method at least: Feature selection is performed by eliminating irrelevant features whose scaling is zero; an **SVM** classifier that has enhanced generalization ability can be learned concurrently. By introducing scaling factor, above QP problem is written as follow form:

$$\begin{aligned} &\underset{w,b}{Minimize} \quad \Phi(w,b) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i \\ &subject\ to \quad y_i\left(w^T (\Delta x_i) + b\right) \geq 1 - \xi_i,\ \xi_i \geq 0,\ i = 1, 2, \cdots, l \end{aligned} \tag{3}$$

in which

$$\Delta = diag\{\delta_1, \cdots, \delta_l, \cdots, \delta_n\},\ 0 \leq \delta_i \leq 1,\ i = 1, \cdots, n \tag{4}$$

Δ is n-dimension diagonal matrix of scaling factors.

J. Weston et al[7] find the optimum value of Δ that can minimize

$\frac{\text{Radius}}{\text{margin}} = R^2(\Delta)\mathbf{w}^2(\Delta)$ or some other differentiable criterion by gradient descent algorithm. In the paper of Olivier Chapelle et al [8], a similar iterative procedure is used in computing preprocessing scaling parameters. Alain Rakotomamonjy [9] has presented different criteria for feature selection algorithms. These criteria are derived from generalization error bounds of the **SVM** theory. Balaji Krishnapuram et al [10] adopt a Bayesian approach to learn both an optimal nonlinear classifier and a subset of predictor features simultaneously that are most relevant to the classification task. In Yves Grandvalet [11], the metric is automatically tuned by the minimization of the standard **SVM** empirical risk, where scale factors are added to the usual set of parameters defining the classifier.

3. Incorporating Feature Selection with SVM Classifier Design Using GA

3.1 Problem formulation

The basic intuition behind our approach to estimate the feature scaling and design classifier problems jointly is to extend the set of parameters to be learned. Consequently, the function similar to Ref. [10] is considered:

$$f(x,\theta,b) = sign\left[\sum_{i=1}^{m} \lambda_i y_i K_\theta(x_i, x_j) + b\right] \tag{5}$$

in which, some common types of $K_\theta(x_i, x_j)$ are listed as follow:

(1) The polynomial kernel:

$$K_\theta(x_i, x_j) = \left(1 + \sum_{l=1}^{n} \theta_l x_i^{(l)} x_j^{(l)}\right)^d \tag{6}$$

where *d* is the degree of polynomial

(2) The Radial Basis Function (RBF) kernel with parameter *g*:

$$K_\theta(x_i, x_j) = \exp\left(-g\sum_{l=1}^{n} \theta_l \left(x_i^{(l)} - x_j^{(l)}\right)^2\right) \tag{7}$$

(3) The sigmoid kernel (only for some values of coefficients τ_1, τ_2)

$$K_\theta(x_i, x_j) = \tanh\left(\tau_1 \sum_{l=1}^{n} \theta_l x_i^{(l)} x_j^{(l)} + \tau_2\right) \tag{8}$$

In above forms, $x_i, x_j \in R^n$ is *n*-dimensional vector, $x_i^{(l)}$ and $x_j^{(l)}$ are value of the *l*-th dimensional feature in x_i and x_j, and θ_l $(0 \le \theta_l \le 1)$ is used to measure relevance of feature *l*. Obviously, feature *l* is unrelated with this problem when $\theta_l = 0$.

Consequently, the main focus of the algorithm is how to estimate θ_l. Usually, search strategies to estimate θ_l can be classified into three categories: (1) Optimal search; (2) Heuristic search; (3) Random search.

3.2 GA based search strategies

GA is a category of randomized optimization procedures inspired by the biological mechanism of reproduction, which is utilized to search the optimal feature scaling that is most favorable to **SVM** classifier aiming at the feature selection and the improving of performance in our method. The feature scaling is automatically tuned by the maximization of fitness in GA. This is different from other search strategies of feature selection using **SVM** based criteria in the above Refs.. Ref. [10] need Laplacian prior to promote sparse in scaling. Refs. [9, 11] need to calculate the gradient of the criterion (or subject function) with regards to a scaling factor θ_l. However when the subject function is complex, the gradient is difficult to estimate, and the gradient descent algorithm can not always find the global optimal solution. Some weaknesses of above methods can be avoided because of needing no priors in the proposed GA.

A candidate solution to the problem is properly encoded as a string of symbols (e.g., binary) in GA. A set of such string structure the initial population randomly. There are three basic genetic operators used by GA to guide its search inside population: Selection, crossover, and mutation. Selection filters out solutions that perform poorly and choose high performance solutions to concentrate on or exploit probabilistically. Crossover and mutation generate new solutions for exploration. Using the feedback from the fitness function to evaluate and select better solutions, GA converging to a population of high-performance solutions eventually. Although GA does not guarantee a global optimum solution, it has the ability to search through very large search spaces and come to nearly optimal solutions fast. For more detail, refer to Srinivas, *et al.* [12]

3.3 Proposed GA-SVM Method for Estimate θ_l in SVM Classifier

3.3.1 Initial parameters

A simple encoding scheme is employed, in which the value of a bit binary string is determined by the value of scaling parameter θ_l. Each linked string (or called as citizen) thus represents a different subset of parameter $(\theta_1, \cdots, \theta_n)$.

In general, the initial population is generated randomly. In all of the experiments, we use a population size of 40 citizens and produce 100 generations. In most cases, the GA converges in less than 100 generations.

Selection strategy is cross generational in our methods. Assuming a population of size *N*, the offspring double the size of the population and we select the best *N* citizens from the combined parent–off spring population.

One-point crossover is used, in which the parent citizens are split at a common point chosen randomly and the resulting sub-citizens are swapped.

Traditional mutation operator is used which just flips a specific bit with a very low probability. The mutation probability is set as 0.04.

3.3.2 Fitness function

The goal of our algorithm is to achieve better performance by estimating scaling parameters in kernels. Therefore, the fitness evaluation contains three terms: (a) accuracy, (b) margin distance, (c) number of eliminated features (where scaling is zero), all of which correspond to a particular subset of $(\theta_1, \cdots, \theta_n)$. We use the fitness function to combine the three terms:

$$Fitness = \alpha_1 \times Accuracy + \alpha_2 \times \frac{number of\ eliminated\ feature}{number of\ original\ feature} + \alpha_3 \times Margin \quad (9)$$

Among three terms, accuracy is our major concern. It is estimated by a cross-validation data set which guides the GA search. Accuracy term ranges roughly from 0.00 to 1.00. The second term estimates the proportion of eliminated features, with assuming values ranging from 0.00 to 1.00. Margin distance suggests the general ability of classifier. Its value is problem dependent.

Based on the weights $\alpha_1, \alpha_2, \alpha_3$ that we have assigned to each term, we can select the term to dominate the fitness value mainly. If the accuracy term is decided to control the fitness value more, this implies that individuals with higher accuracy will outweigh individuals with lower accuracy, no matter how many features they contain. This allows driving toward higher classification accuracy than fewer features. In addition, the margin distance in the classification process is used to improve the general ability and provide additional guidance for the GA.

3.3.3 Zero-bias operator

In order to predigest complexity of classifier and reduce dimension of feature vector, zero-bias operator is induced to make θ_l near zero more frequently when searching. A binary string is set to 0 randomly with low probability of 0.06. This probability can be adjusted according to problem condition at hand.

3.3.4 Overall procedure

The overall algorithm, where $P(t)$ is the population of strings at generation t, n is the number of citizen in population, is given below:

Generation $t = 0$
Initialize $\theta_l\ (l = 1, \cdots, n), (0 \le \theta_l \le 1),$ form $P(t)$
Evaluate strings in $P(t)$
    While (termination condition is not satisfied) do
    Begin

(a) Selection, crossover, mutation and zero-bias operator from $P(t)$ according to fitness

(b) Recombine $P(t+1)$

(c) Evaluate $P(t+1)$ according to fitness

(d) $t = t+1$

End

Output optimal $\theta_l \ (l = 1, \cdots, n)$

4 Experiment and Analyze

To evaluate the performance of proposed algorithm to detect spam, experiments are conducted on Spam database in UCI[13].Standard **SVM** classifier and GA-SVM classifier are used respectively.

4.1 Experiment datasets

Spam database distinguishes whether an e-mail is spam or nonspam according to 57 attributes (57 continuous, 1 nominal class label). These attributes include percentage of words in the e-mail that match specified word (such as hp, free), percentage of words in the e-mail that match specified character (such as $), average length of uninterrupted sequences of capital letters etc.. There are 4601 instances, in which 1813 are spam. We select 2500 normal e-mails and 1601 spams to construct training set randomly. The remainder 288 normal e-mails and 212 spams are used as testing set.

4.2 Experiment results

In the experiment, we use standard **SVM** method to train optimal **SVM** classifiers firstly. Under the same kernel type, GA-SVM is utilized to finish feature selection and experiment is performed on feature subset obtained. The performance of these SVMs is measured by the precision of classification, the number of features and support vectors used finally.

The parameters in GA are set as: selection proportion is 0.5, P_c=0.4, P_m=0.04, P_z=0.06. Choosing the weights $\alpha_1, \alpha_2, \alpha_3$ in Eq.(9) for the fitness function is more subjective for user. In the scenario when the best performance is preferred to model cost, the weight α_1 associated with the accuracy term should be very high. Under other different situations when compactness of model is more important than other terms, a higher weight α_2 for the second term should be chosen. In our experiments, values of $\alpha_1, \alpha_2, \alpha_3$ are selected as follow:

$$\alpha_1 = 1, \ \alpha_2 = 0.5, \alpha_3 = 0.5$$

In the spam database, there are five features to be eliminated, such as word_freq_3d, word_freq_addresses, word_freq_857,word_freq_415 and

word_freq_table. Tab. 1 gives the results of **SVM** classifier and GA-SVM, in which Poly indicates polynomial kernel function.

Tab. 1 Result on UCI spam database

Database	Algorithm	Number of Feature	Number of SV	Precision (%)	
				Training	Testing
spam	SVM (Poly2,C=2.3229)	57	919	94.38	87.73
	GA-SVM (Poly2,C=2.5754)	52	905	94.43	87.89

As shown in Tab. 1, comparing with original **SVM**s, the number of features and support vectors decreased ,while better precision are achieved in spam database using GA-SVM. In experiment, we have noted that the training time of GA-SVM is longer than that of original **SVM** because of the search cost of GA.

5 Conclusion

A method to design **SVM** classifier and feature selection jointly using GA for spam detection is proposed in this paper. The relevant coefficients of various features to the classification task, measured by scaling factor, are estimated by GA. And GA approach exploits heavy-bias priors to promote sparse in the scaling factor of features, so feature selection is performed by eliminating irrelevant features whose scaling factor is zero. A **SVM** classifier that has enhanced generalization ability can be learned simultaneously.

Experimental comparisons using original **SVM** and GA-SVM demonstrate that GA-SVM can successfully achieve both of its objectives: better classification accuracy and automatic feature selection. It is also demonstrated that GA can provide a simple, general and powerful framework for tuning parameters in optimal problem, which directly improves detection performance and rate of **SVM**.

References

1. W. W. Cohen.Learning rules that classify e-mail. In Proc. 1996 AAAI Spring Symp. Inform. Access.
2. M. Sahami, S. Dumais, D. Heckerman, E. Horvitz. A Bayesian approach to filtering junk e-mail. In AAAI'98 Wkshp. Learning for Text Categorization, Madison, WI, July 27, 1998
3. Harris Drucker et al. Support Vector Machines for Spam Categorization. IEEE TRANSACTIONS ON NEURAL NETWORKS. 1999,10(5):1048-1054
4. Corinna Cortes, Vladimir Vapnik. Support -vector networks. Machine Learning.1995 (20):273-297.
5. Ron Kohavi, George H. John. Wrappers for feature subset selection. Artificial Intelligence. 97(1997) , 273-324.
6. Isabelle Guyon, Andre Elissee. An introduction to variable and feature selection. Journal of Machine Learning Research . 2003 (3):1157-1182

7. J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, V. Vapnik. Feature selection for support vector machines. Neural Information Processing Systems, Cambridge, MA, MIT Press, 2001
8. Olivier Chapelle, Vladimir Vapnik, Olivier Bousquet , Sayan Mukherjee. Choosing multiple parameters for support vector machines. Machine Learning. 2002 (46):131–159,
9. Alain Rakotomamonjy. Variable selection using SVM-based criteria. Journal of Machine Learning Research. 2003 (3):1357-1370
10. Balaji Krishnapuram, Alexander J. Hartemink, Lawrence Carin, Mario A.T. Figueiredo. A bayesian approach to joint feature selection and classifier design. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2004 (26) 9:1105-1111.
11. Yves Grandvalet, Stephane Canu. Adaptive scaling for feature selection in SVMs. NeuralInformation Processing Systems 15, 2002.
12. M. Srinivas, L. Patnaik. Genetic algorithms: a survey. IEEE Comput, 1994 (27) 6:17–26.
13. Blake, C. L., Merz, C. J. (1998). UCI repository of machine learning databases, URL: http://www.ics.uci.edu/~mlearn/MLRepository.html .