Text Classification from Positive and Unlabeled Documents Based on GA

Tao Peng, Fengling He, Wanli Zuo

College of Computer Science and Technology, Jilin University, Key Laboratory of Symbol
Computation and Knowledge Engineering of the Ministry of Education,
Changchun 130012, China
drtpeng@yahoo.com.cn

Abstract. Automatic text classification is one of the most important tools in Information Retrieval. As the traditional methods for text classification cannot find the best feature set, the GA is applied to the feature selection because it can get the global optimal solution. This paper presents a novel text classifier from positive and unlabeled documents based on GA. Firstly, we identify reliable negative documents by improved 1-DNF algorithm. Secondly, we build a set of classifiers by iteratively applying SVM algorithm on training example sets. Thirdly, we discuss an approach to evaluate the weighted vote of all classifiers generated in the iteration steps to construct the final classifier based on GA instead of choosing one of the classifiers as the final classifier. GA evolving process can discover the best combination of the weights. The experimental result on the Reuter data set shows that the performance is exciting.

1 Introduction

In general, text classification systems categorize documents into one (or several) of a set of pre-defined topics of interest. Text classification is of great practical importance today given the massive text available. With the rapid growth of information and the explosion of electronic text from the World Wide Web, one way of organizing this overwhelming amount of documents is to classify them into descriptive or topical taxonomies. Text categorization is used to automatically catalog news articles [1] and web pages [2], automatically learn the reading interests of users [3] [4]. In recent years, a number of statistical classification and machine learning techniques have been applied to text categorization, including regression models [5], nearest neighbor classifiers [6], decision tree, Bayesian classifiers [7], support vector machines [8], etc. One key difficulty with traditional approach is that they require a large, often prohibitive, number of labeled training examples to learn accurately. Labeling must often be done by a person, this is a painfully time-consuming process. Recently, researchers investigated the idea of using a small labeled set of every class and a large unlabeled set to help learning [9], [10], [11], [12], [13]. The PEBL algorithm achieves classification accuracy (with positive and unlabeled data) as high as that of traditional SVM (with positive and negative data) [14]. The PEBL algorithm uses the 1-DNF algorithm to identify the set of reliable negative documents and builds the set of classifiers by iteratively applying an SVM algorithm.

This paper describes a new text classification process that uses a genetic algorithm to evolve the weights of the metrics. We apply genetic algorithm to search out and identify the potential informative features combinations for classification and then use the F_1-Measure to determine the fitness in genetic algorithm. GAs are general-purpose search algorithms which use principles inspired by natural genetic populations to evolve solutions to problems [15], [16]. In our approach, not as usual, an individual is a combination of the real-coded metrics' weight, and it's more natural to represent the optimization problem in the continuous domain.

The rest of the paper is organized as follows. Section 2 describes the whole process of building the text classifier. Section 2.1 introduces how to improve 1-DNF algorithm and identify reliable negative examples. The process of building a set of classifiers by iteratively applying SVM algorithm on training examples set illustrated in Section 2.2. Section 2.3 describes evolving the weights with genetic algorithm. Section 3 reports the results of our experiments. Section 4 draws the conclusion.

2 Text Classification

The goal of text categorization is the classification of documents into a fixed number of predefined categories. Each document *d* can be in multiple, exactly one, or no category at all. Using machine learning, the objective is to learn classifiers from examples that do the category assignments automatically. Constructing our text classifier adopts three steps: Firstly, identify a set of reliable negative documents from the unlabeled set by using our improved 1-DNF algorithm (1-DNFII). Secondly, build a set of classifiers by iteratively applying the SVM algorithm, Thirdly, construct the final text classifier by using the weighted voting method based on GA.

2.1 Identifying Reliable Negative Documents

For identifying the reliable negative documents from the unlabeled examples set, we must identify the features of the negative documents. For example, if the frequency of a feature occurs in the positive examples set exceed 90%, whereas less than 10% in the unlabeled examples set, then this feature will be regarded as positive. Using this method, we can obtain a positive feature set *PF*. If a number of documents in the unlabeled examples set do not contain any feature in the positive feature set *PF*, these documents can be regarded as reliable negative examples. For describing expediently, we define the following notation: *P* represents the positive examples set, and *U* represents the unlabeled examples set, and NEG_0 represents the reliable negative documents set produced by our improved 1-DNF algorithm (1-DNFII), and NEG_i ($i \geq 1$) be the negative documents set produced by the *i*th iteration of the SVM algorithm, and *PON* represents the training examples set.

In 1-DNF algorithm [14], a positive feature is defined that occur in the positive set *P* more frequently than in the unlabeled set *U*. We found that this definition has an obvious shortcoming: it only considers the diversity of feature occurred frequency in *P* and *U*, and does not consider the absolute frequency of the feature in *P*. For example, the frequency of some feature is 0.2% in the positive data set and 0.1% in the unlabeled data set; this feature is obviously not positive feature. But if we use the 1-DNF algorithm to identify negative data, this feature will be regard as positive. The result is that the number of features in the *PF* must be much more. Nevertheless the number of documents in NEG_0 identified by 1-DNF algorithm is less. Sometimes, NEG_0 may be empty. From above discussion, we improved the 1-DNF algorithm (1-DNFII) by considering both the diversity of the feature frequency in *P* and *U* and the absolute frequency of the feature in *P*. In 1-DNFII, A feature is regarded as positive only when it satisfies the following conditions: Firstly, the frequency of the feature occurred in the positive data set is greater than the frequency of the feature occurred in the unlabeled data set. Secondly, the absolute frequency of the feature in the positive data set is greater than λ % (λ is a constant).

2.2 Building Text Classifiers with SVM

Unlike traditional approach that one specific classifier in the classifiers set generated during the iterative algorithm is designated as the final one, we make use of all of them to construct the final classifier based on GA voting method. Using 1-DNFII algorithm, we can obtain the more reliable negative documents set NEG_0. Now the training samples set is $PON = P \cup NEG_0$, and the unlabeled samples set is $U = U - NEG_0$. We use SVM algorithm learnt from the training data set *PON* to construct the initial classifier SVM_0, and use SVM_0 to classify *PP* and *U* (the classified negative documents is NEG_1). Then the training examples set is increased to $PON=P \cup NEG_1$, and unlabeled sample set is $U=U\text{-}NEG_1$. We use the training set *PON* to construct the second classifier SVM_1. SVM_1 is used to classify *PP* and *U* (the classified negative documents is NEG_2). Then the training example set is increased to $PON=P \cup NEG_2$, and the unlabeled set is $U=U\text{-}NEG_2$. This process iterates until no documents in *U* are classified as negative. Then we use GA evolving process to discover the best combination of the each individual classifier's weights to construct the final classifier. **Fig.1** describes the process of the constructing our classifier.

2.3 Evolve the weights with Genetic Algorithm

After building individual classifiers, we use the following function to construct the final text classifier.

$$FinalClassifier = \sum_{i=1}^{n} \omega_i Classifier_i \quad , \quad \sum_{i=1}^{n} \omega_i = 1 \qquad \textbf{(1)}$$

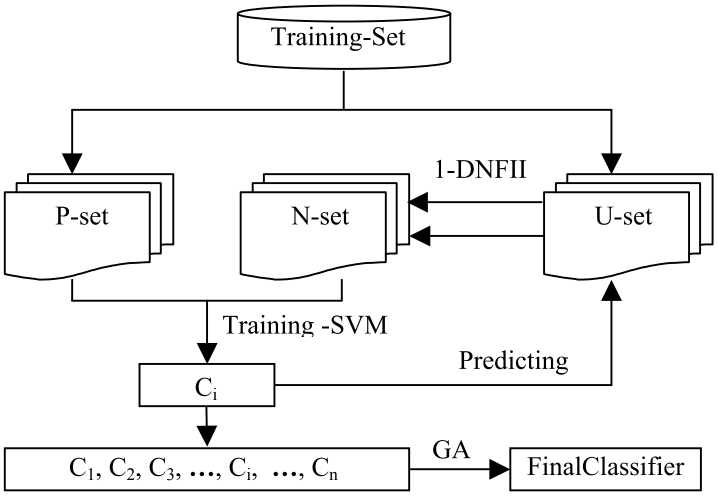

Fig. 1. Illustration of the procedure to build text classifiers from labeled and unlabeled examples based on GA. C_i represents the individual classifier produced by the *i*th iteration of the SVM algorithm

The weights of the function (1) are evolved with genetic algorithm. The individuals are real-coded, because that the representation of the solution could be very close to the natural formulation of our problem. Since the amount of the weights equals to 1, the weight ω_i is coded into the gene, c_i, and c_i is defined by

$$c_i = \sum_{j=1}^{i} \omega_j \quad , i=1,\ldots,n. \qquad (2)$$

Each individual in the population is the combination of $c_1, \ldots ,c_{n-1}$. Obviously, there must be a restriction of any individual, $x_i \geq x_{i+1}$, to ensure the individual could be decoded into the weights.

The standard fitness proportional model, which is also called roulette wheel selection, is used as the selection method to select the individuals for reproduction. The probability of an individual to be selected is $P_i = f_i \Big/ \sum_{j=1}^{n} f_j$, n is the population size [17]. Individuals are crossed using simple crossover method [18]. We can assume that $C_1 = (c_1^1,\ldots,c_n^1)$ and $C_2 = (c_1^2,\ldots,c_n^2)$ are two chromosomes selected for the crossover operator. The single crossing position $j \in \{1, \ldots ,n-1\}$ is randomly chosen and the two new chromosomes are built as

$$C_1' = (c_1^1, c_2^1,\ldots,c_i^1, c_{i+1}^2,\ldots,c_n^2) \quad C_2' = (c_1^2, c_2^2,\ldots,c_i^2, c_{i+1}^1,\ldots,c_n^1)$$

Of course, due to the restriction of the individual that is referred before, the genes of C_1' , C_2' must be sorted according to the sort ascending. If an individual is chosen for the mutation operator, one of the randomly chosen genes c_i will change to $c_i' \in (c_{i-1}, c_{i+1})$ which is a random value, and we assume that $c_0 = 0$, $c_n = 1$. Finally, all of the individuals, including new and old ones, are sorted by their fitness, and the best-fit individuals become the new population in the next generation. We set the probability of crossover to be 0.8, and the probability of mutation to be 0.05. After 50 generations, we finish the evolving process, choose the individual with the highest fitness of the population, and decode the genes to the weights as the result. **Fig.2** shows the original representation of chromosome suitable to huge-scale features.

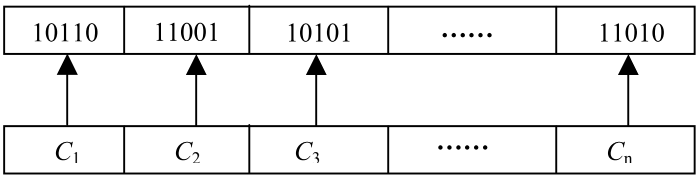

Fig. 2. The representation of chromosome.

3 The Experiments and Results

In the experiment, we used Reuters-21578, which has 21578 documents collected from the Reuters newswire, as our training sample set. Of the 135 categories in Reuters 21578, only the most populous 10 are used. In data pre-processing, we applied *stopword* removal and *tfc* feature selection, and removed the commoner morphological and inflexional endings from words using The Porter Stemming Algorithm. Each category is employed as the positive class, and the rest as the negative class. For each dataset, 30% of the documents are randomly selected as test documents, and the rest are used to create training sets as follows: γ percent of the documents from the positive class is first selected as the positive set *P*. The rest of the positive documents and negative documents are used as unlabeled set *U*. We range γ percent from 10%-50% to create a wide range of scenarios.

To evaluate our final classifier, we use the F_1-Measure, which is a commonly used performance measure for text classification. This measure combines precision and recall in the following way:

$$Precision = \frac{\#\,of\ correct\ positive\ predictions}{\#\,of\ \ positive\ predictions}\,. \quad \textbf{(3)}$$

$$Recall = \frac{\#\,of\ correct\ positive\ predictions}{\#\,of\ \ positive\ examples}\,. \quad \textbf{(4)}$$

$$F_1\ \text{-}\ Measure = \frac{2 * precision * recall}{precision + recall}\,. \quad \textbf{(5)}$$

We choose the best individual based on the metric of *fitness* function.

$$fitness = F_1\ \text{-}\ Measure\,. \quad \textbf{(6)}$$

In the experiments, we implemented the PEBL algorithm, one-class SVM (OCS) algorithm and our classifier based on GA voting (GAC) and compared their performance. **Fig. 3** shows the F_1-Measure value histogram of the above three classification methods.

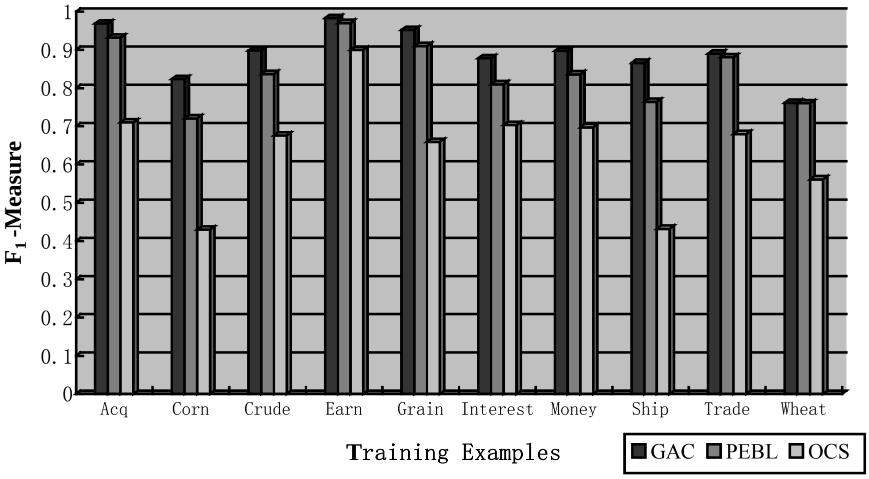

Fig. 3. The performance of the three classification methods. Where λ =0.1, which the performance of the final classifier is best (proved in another experiment).

4 Conclusions

This paper discussed a three-step strategy classification method from positive and unlabeled examples based on GA. We improved 1-DNF algorithm (1-DNFII) to increase the number of negative documents. The experiment shows that 1-DNFII has a better performance than 1-DNF algorithm. After building a set of classifiers by iteratively applying the SVM algorithm, we construct the final text classifier by using the weighted voting method based on GA (GAC). We also implemented the PEBL algorithm and one-class SVM (OCS) algorithm. Compared the three methods, the performance of the GAC and PEBL is greatly better than OCS. Especially, GAC is better than PEBL. But, our classification method (GAC) consumed a bit more resource of CPU and RAM than the other two.

Acknowledgment

This work is sponsored by the National Natural Science Foundation of China under grant number 60373099.

References

1. Lewis, D. D., Gale, W. A.: A sequential algorithm for training text classifiers. In SIGIR '94: Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (1994) 3–12
2. Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., & Slattery, S.: Learning to extract symbolic knowledge from the World Wide Web. In Proceedings of the Fifteenth National Conference on Artificial Intellligence (AAAI-98) (1994) 509–516
3. Pazzani, M. J., Muramatsu, J., Billsus, D.: Syskill & Webert: Identifying interesting Web sites. In Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96) (1996) 54–56
4. Lang, K.: Newsweeder: Learning to filter netnews. In Machine Learning: Proceedings of the Twelfth International Conference (ICML '95) (1995) 331–339
5. Y. Yang and J. P. Pedersen.: Feature selection in statistical learning of text categorization. In Proceedings of the Fourteenth International Conference on Machine Learning (1997) 412-420
6. E.S. Han, G. Karypis, and V. Kumar.: Text categorization using weight adjusted k-nearest neighbor classification, Computer Science Technical Report TR99-019 (1999)
7. D. Levis and M. Ringuette.: A comparison of two learning algorithms for text classification. In Third Annual Symposium on Document Analysis and Information Retrieval (1994) 81-93
8. C. Cortes and V. Vapnik.: Support vector networks, Machine learning (1995)volume 20, 273-297
9. Muslea, I., Minton, S., and Knoblock, C. A.: Active + semi-supervised learning = robust multiviewlearning. ICML-02 (2002)
10. F. Letouzey, F. Denis, and R. Gilleron.: Learning from positive and unlabeled examples. In Workshop on Algorithmic Learning Theory (ALT) (2000)
11. F. DeComite, F. Denis, and R. Gilleron.: Positive and unlabeled examples help learning. In Workshop on Algorithmic Learning Theory (ALT) (1999)
12. Xiaoli Li, Bing Liu, Learning to classify text using positive and unlabeled data. The International Joint Conference on Artifical Intelligence (IJCAI) (2003)
13. Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, Philip S. Yu, Building Text Classifiers Using Positive and Unlabeled Examples. Proceedings of the Third IEEE International Conference on Data Mining (ICDM) (2003) 179-187
14. Hwanjo Yu, Jiawei Han, Kevin Chen-Chuan Chang, PEBL: Positive example based learning for Web page classification using SVM. The international conference on Knowledge Discovery and Data mining (KDD) (2002)
15. Holland, J.H.: Adaptation in Natural and Artificial Systems. The University of Michigan Press (1975)
16. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. AddisonWesley, New York (1989)
17. Zhou C G, Liang Y C.: Computational Intelligence. Jilin university press, Changchun. China (2001)
18. F. Herrera, M. Lozano, J.L. Verdegay.: Tackling Real Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis. Artificial Intelligence Review, @1998 Kluwer Academic Publishers. Printed in the Netherlands (1998) 12: 265–319