Available online at www.sciencedirect.com
ScienceDirect
Expert Systems with Applications 33 (2007) 215–220
Expert Systems with Applications
www.elsevier.com/locate/eswa

Large margin DragPushing strategy for centroid text categorization

Songbo Tan *

Software Department, Institute of Computing Technology, Chinese Academy of Sciences, P.O. Box 2704, Beijing, 100080, PR China
Graduate School of the Chinese Academy of Sciences, PR China

Abstract

Among all conventional methods for text categorization, centroid classifier is a simple and efficient method. However it often suffers from inductive bias (or model misfit) incurred by its assumption. DragPushing is a very simple and yet efficient method to address this so-called inductive bias problem. However, DragPushing employs only one criterion, i.e., training-set error, as its objective function that cannot guarantee the generalization capability. In this paper, we propose a generalized DragPushing strategy for centroid classifier, which we called as ''Large Margin DragPushing'' (LMDP). The experiments conducted on three benchmark evaluation collections show that LMDP achieved about one percent improvement over the performance of DragPushing and delivered top performance nearly as well as state-of-the-art SVM without incurring significant computational costs.
© 2006 Published by Elsevier Ltd.

Keywords: Text classification; Information retrieval; Machine learning

1. Introduction

With the exponential growth of text data from internet, text categorization has received more and more attention in information retrieval, machine learning and natural language processing community. Numerous machine learning algorithms have been introduced to deal with text classification, such as K-Nearest Neighbor (KNN) (Yang & Lin, 1999), centroid classifier (Han & Karypis, 2000), Naive Bayes (Lewis, 1998), support vector machines (SVM) (Joachims, 1998), Rocchio (Joachims, 1997), decision trees (Apte, Damerau, & Weiss, 1998), Winnow (Zhang, 2001), Perceptron (Zhang, 2001) and voting (Schapire & Singer, 2000).

Among all these methods, centroid classifier is a simple and effective method for text categorization. However it often suffers from inductive bias or model misfit (Liu, Yang, & Carbonell, 2002; Wu, Phang, Liu, & Li, 2002) incurred by its assumption. As we all know, centroid classifier makes a simple assumption that a given document should be assigned a particular class if the similarity of this document to the centroid of the class is the largest. However, this supposition is often violated (misfit) when there exists a document from class A sharing more similarity with the centroid of class B than that of class A. Consequently, the more serious the model misfit, the poorer the classification performance.

In order to address this issue effectively, numerous researchers have devoted their energy to refining the classifier model by proposing efficient strategies. One of the popular strategies is voting (Schapire & Singer, 2000). Wu et al. (2002) presented another novel approach to cope with this problem. However, these methods both suffer from their own shortcomings respectively. The computational costs of voting is quite significant and its application to text classification community is often severely limited; Meanwhile Wu's strategy may be sensitive to the imbalance of text corpora for it needs to split the training set.

In order to improve the performance of centroid classifier, an effective and yet efficient strategy, i.e., ''DragPushing'', was introduced to refine the centroid classifier model (Tan, Cheng, Ghanem, Wang, & Xu, 2005). The main idea behind this strategy is that it takes advantage

* Tel.: +86 10 88449181 713; fax: +86 10 88455011.
E-mail address: tansongbo@software.ict.ac.cn

0957-4174/$ - see front matter © 2006 Published by Elsevier Ltd.
doi:10.1016/j.eswa.2006.04.008

of training errors to successively refine classification model. For example, if one training example d labeled as class A is misclassified into class B, DragPushing "drags" the centroid of class A to example d, and "pushes" the centroid of class B against example d. After this operation, document d will be more likely to be correctly classified by the refined classifier.

However, DragPushing employs only one criterion, i.e., training-set error, as its objective function that cannot guarantee the generalization capability. In this paper, we further generalize DragPushing to Large Margin DragPushing (LMDP) which refines the centroid classifier model not only using training errors but also utilizing training margins. The goal of LMDP is to reduce the training errors as well as to enhance the generalization capability.

Extensive experiments conducted on three benchmark document corpora show that LMDP always performs better than DragPushing. Like DragPushing, LMDP is also much faster than many state-of-the-art approaches, e.g., SVM, while delivers performance nearly as well as SVM.

The rest of this paper is constructed as follows: Next section describes basic centroid classifier. DragPushing strategy is presented in Section 3. LMDP is introduced in the Section 4. Experimental results are given in Section 5. Finally Section 6 concludes this paper.

2. Centroid classifier

In our work, the documents are represented using vector space model (VSM). In this model, each document d is considered to be a vector in the term-space. For the sake of brevity, we denote summed centroid and normalized centroid of class C_i by $\overrightarrow{C_i^S}$ and $\overrightarrow{C_i^N}$ respectively.

First we compute summed centroid by summing document vectors in class C_i:

$$\overrightarrow{C_i^S} = \sum_{d \in C_i} \vec{d} \tag{1}$$

Next we evaluate normalized centroid by following formula:

$$\overrightarrow{C_i^N} = \frac{\overrightarrow{C_i^S}}{\left\| \overrightarrow{C_i^S} \right\|_2} \tag{2}$$

where $\|z\|_2$ denotes the 2-norm of vector z.

Afterwards we calculate the similarity between a document d and each centroid C_i using inner-product measure as follows:

$$sim(d, C_i) = \vec{d} \cdot \overrightarrow{C_i^N} \tag{3}$$

It is worth mentioning that the inner-product measure using formula (3) is equivalent to cosine measure since document vector $\vec{d}$ is computed using TFIDF formula (19) which is equal to counting TFIDF by following formula (4) and then normalizing it by 2-norm.

$$w_{tfidf}(t, \vec{d}) = tf(t, \vec{d}) \times \log(D/n_t + 0.01) \tag{4}$$

Finally, based on these similarities, we assign d the class label corresponding to the most similar centroid:

$$C = \arg\max_{C_i} \left(\vec{d} \cdot \overrightarrow{C_i^N} \right) \tag{5}$$

3. DragPushing strategy

Since the text collection may conflict the assumption of Centroid Classifier to some degree, inevitably the classification rules, i.e., class representatives or class centroids, induced by centroid classifier may contain some kinds of biases that result in degradation of classification accuracy on both training and test documents. An intuitive and straightforward solution to this problem is to make use of training errors to adjust class centroids so that the biases can be reduced gradually.

Initialization: to start, we need to load the training data and parameters including *MaxIteration* and *ErrorWeight*. Then for each category C_i we calculate one summed centroid $\overrightarrow{C_i^{S,0}}$ and one normalized centroid $\overrightarrow{C_i^{N,0}}$. Note that 0 denotes current iteration-step, i.e., the 0th iteration-step.

DragPushing: in one iteration, we need to categorize all training documents. If one document d labeled as class "A" is classified into class "B", DragPushing modifies the summed and normalized centroids of class "A" and class "B" by following formulas:

$$C_{A,l}^{S,0+1} = C_{A,l}^{S,0} + ErrorWeight \times d_l \;\; if \;\; d_l > 0 \tag{6}$$

$$C_{A,l}^{N,0+1} = \frac{C_{A,l}^{S,0}}{\left\| \overrightarrow{C_A^{S,0}} \right\|_2} \;\; if \;\; C_{A,l}^{S,0} > 0 \tag{7}$$

$$C_{B,l}^{S,0+1} = \left[C_{B,l}^{S,0} - ErrorWeight \times d_l \right]_+ \;\; if \;\; d_l > 0 \tag{8}$$

$$C_{B,l}^{N,0+1} = \frac{C_{B,l}^{S,0}}{\left\| \overrightarrow{C_B^{S,0}} \right\|_2} \;\; if \;\; C_{B,l}^{S,0} > 0 \tag{9}$$

where 0 denotes the 0th iteration-step and l stands for feature index of document vectors and centroid vectors. $[z]_+$ denotes the hinge function which equals the argument z if $z > 0$ and is zero otherwise, i.e., $[z]_+ = \max\{z, 0\}$. The reason for introduction of $[z]_+$ is that we found nonnegative centroids perform better than real centroids in our experience.

We call the former formulas (6,7) as "drag" formulas and the latter (8,9) as "push" formulas. Obviously after the executing of "drag" and "push" formulas, the centroid of class "A" will share more similarity with document d than that of class "B". As a result, the similarity between document d and the centroid of class "A" will be enlarged while the similarity between document d and the centroid of class "B" will be reduced.

Time Requirements: Assumed that there are D training documents, T test documents, W words in total, K classes