A novel supervised learning algorithm and its use for Spam Detection in Social Bookmarking Systems

Anestis Gkanogiannis and Theodore Kalamboukis

Department of Informatics
Athens University of Economics and Business, Athens, Greece
utumno@aueb.gr tzk@aueb.gr

Abstract. A novel fast and accurate supervised learning algorithm is proposed as a general text classification algorithm for linearly separated data. The strategy of the algorithm takes advantage of the training errors to successively refine an initial classifier. Experimental evaluation of the proposed algorithm on standard text collections, show that results compared favorably to those from state of the art algorithms such as SVMs. Experiments conducted on the datasets provided in the framework of the ECDL/PKDD 2008 Challenge for Spam Detection in Social Bookmarking Systems, demonstrate the effectiveness of the proposed algorithm.

1 Introduction

Text categorization is the process of making binary decisions about related or non-related documents to a given set of predefined thematic topics or categories. The task is an important component in many information management organizations. In our participation on the ECML/PKDD challenge 2008 we have not deviate from the classical supervised text classification paradigm.

Support vector machines (SVMs) [1] have shown the best performance for text classification tasks. They are accurate, robust, and quick when applied to test instances. Their only drawback is their training complexity and memory requirements. In what follows we present an algorithm, which overcomes both these problems. The strategy of the proposed algorithm takes advantage of the training errors (misclassified examples) to successively refine an initial classifier. This refinement takes the form of an iterative Rocchio-like relevance feedback-learning technique to adjust the centroid vectors of the categories, in order to maximize the performance of the classifier. Our learning algorithm runs on batch mode using all the training data at each iteration step.

Rocchio's relevance feedback technique [2] is a query modification process that has been extensively investigated in the literature and used in information retrieval. Relevance feedback improves the query with terms that are considered relevant to the information neek. This is done iteratively either manually or automatically by selecting a predefined set of the top retrieved documents as relevant (pseudo-relevance feedback). The aim is to find the optimum query, that is a query that maximizes the similarity with relevant documents while minimizing similarity with non-relevant documents. If $R(\bar{R})$ is the set of relevant (non-relevant) documents, then we wish to find, Q_{opt}, such that:

$$Q_{opt} = argmax_q \left(sim(q,R) - sim(q,\bar{R}) \right) \tag{1}$$

where $sim(q,R)$ denotes the similarity of the query to the set of relevant documents. Using as similarity measure the *cosine* formula, we get that:

$$\vec{Q}_{opt} = \frac{1}{|R|} \sum_{d_j \in R} \vec{d}_j - \frac{1}{|\bar{R}|} \sum_{d_j \in \bar{R}} \vec{d}_j = \vec{C}_R - \vec{C}_{\bar{R}} \tag{2}$$

Thus, the optimum query is a vector defined by the difference of the centroids of the relevant and non-relevant documents, as it is shown in Figure 1a. In other words Q_{opt} defines a hyperplane, $h : \vec{Q}_{opt} \cdot \vec{x} - \theta = 0$ that separates the sets, R and $\bar{R}$, for an appropriate value of θ.

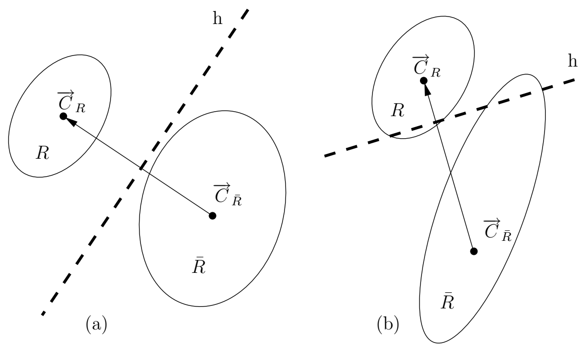

Fig. 1. Vector $C_R - C_{\bar{R}}$ is not always the optimum as it is the case in (1b)

This is not, however, always the case as it is shown in Figure 1b, where the query defined by equation (2) is not optimum although the sets R, $\bar{R}$ are linearly separable and as a consequense there is a hyperplane that separates them. Algorithmically the relevance feedback process has been implemented by updating the query iteratively according to the following equation:

$$\vec{Q}_{i+1} = \kappa \vec{Q}_i + \frac{\lambda}{|R|} \sum_{d_j \in R} \vec{d}_j - \frac{\mu}{|\bar{R}|} \sum_{d_j \in \bar{R}} \vec{d}_j \tag{3}$$

for a given initial query vector, $\vec{Q}_0$, where the constants κ, λ, μ are control parameters defined empirically. From (3) follows that the optimal query is, in general, a linear combination of the relevant and non-relevant documents. In the following we shall use the described Rocchio's feedback technique in constructing a classifier for a pair of linearly separable sets. The algorithm starts with an initial classifier, defined in (2), which is improved iteratively applying the relevance feedback technique on the misclassified examples (Figure 1b). In the next paragraph we describe the algorithm in more detail.

The rest of the paper is organized as follows. Section 2 provides a short description of the proposed algorithm. Section 3 present briefly the task of spam detection in social bookmarking systems and the preprocessing of the data. Section 4 presents the experimental results and finally in section 5 we conclude on the results and the advantages of the proposed algorithm.

2 The Learning Algorithm

We briefly describe here a modification of an algorithm [3], which has been tested, on several standard text collections describing a consistent behavior over all these collections with a performance comparable to SVMs, a state of the art classification algorithm. The algorithm constructs a common tangent hyperplane for the sets R,$\bar{R}$ such that these sets lie on opposite sides of the hyperplane.

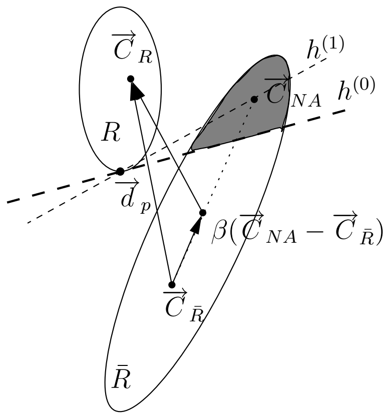
$\overrightarrow{C}_R$
$h^{(1)}$
$\overrightarrow{C}_{NA}$
$h^{(0)}$
R
$\overrightarrow{d}_p$
$\beta(\overrightarrow{C}_{NA} - \overrightarrow{C}_{\bar{R}})$
$\overrightarrow{C}_{\bar{R}}$
$\bar{R}$

Fig. 2. Rotation of hyperplane $h^{(0)}$ towards the misclassified examples

Initialization of the algorithm:

- Select initial vector $\overrightarrow{W}^{(0)} = \overrightarrow{C}_R^{(0)} - \overrightarrow{C}_{\bar{R}}^{(0)}$
- Calculate $s_j = \overrightarrow{W}^{(0)} \cdot \overrightarrow{d}_j$, $\forall\ d_j \in R \cup \bar{R}$
- Find s_p such that $s_p = min(\overrightarrow{W}^{(0)} \cdot \overrightarrow{d}_j)$, $\forall d_j \in R$

The hyperplane defined by $h^{(0)} : \overrightarrow{W}^{(0)} \cdot \overrightarrow{x} - \theta = 0$, with $\theta = s_p = \overrightarrow{W}^{(0)} \cdot \overrightarrow{d}_p$ by construction is vertical to the vector $\overrightarrow{W}^{(0)}$ and $\overrightarrow{d}_p$ lies on it (θ was defined at the value of $recall = 1$).

By construction all the relevant documents lie on the same side of $h^{(0)}$, the one pointed by its normal vector $\overrightarrow{W}^{(0)}$. If it happens all the non-relevant examples to lie on the other side of $h^{(0)}$, then $h^{(0)}$ is a separating hyperplane and the algorithm stops. However this is not generally the case, as it is shown in Figure 2, where the hyperplane $h^{(0)}$ defined by the above process, intersects the set of non-relevant documents and the examples in the gray area are misclassified. In this case we rotate the hyperplane $h^{(0)}$ towards the misclassified examples (Negative Accepted (NA) examples) until all the negative examples lie on the same side of the hyperplane.

Rotation of $h^{(0)}$ towards the misclassified examples: The rotation of the hyperplane is performed stepwise. At each step we determine the misclassified, NA, training examples by the current classifier, construct their centroid vector, $\overrightarrow{C}_{NA}$, and then rotate the hyperplane forcing it to pass through the points $\overrightarrow{d}_p$ and $\overrightarrow{C}_{NA}$. This is equivalent to the process of moving $\overrightarrow{C}_{\bar{R}}$ towards the misclassified examples $\overrightarrow{C}_{NA}$ by adding the vector $\beta(\overrightarrow{C}_{NA} - \overrightarrow{C}_{\bar{R}})$, i.e. $\overrightarrow{C}_{\bar{R}} \leftarrow \overrightarrow{C}_{\bar{R}} + \beta(\overrightarrow{C}_{NA} - \overrightarrow{C}_{\bar{R}})$, such that the plane defined by $\overrightarrow{W}^{(1)} = \overrightarrow{C}_R - (\overrightarrow{C}_{\bar{R}} + \beta(\overrightarrow{C}_{NA} - \overrightarrow{C}_{\bar{R}}))$ passes through the points $\overrightarrow{d}_p$ and $\overrightarrow{C}_{NA}$. With little algebra we estimate the value of the parameter β by:

$$\beta = \frac{\overrightarrow{W}^{(0)} \cdot (\overrightarrow{C}_{NA} - \overrightarrow{d}_p)}{(\overrightarrow{C}_{NA} - \overrightarrow{C}_{\bar{R}}) \cdot (\overrightarrow{C}_{NA} - \overrightarrow{d}_p)} \tag{4}$$

This rotation however may cause examples in the set R to be misclassified (PR, the set of Positive Rejected examples). Thus the process is repeated now on the set R by moving $\overrightarrow{C}_R$ towards the centroid of the misclassified examples $\overrightarrow{C}_{PR}$, i.e. $\overrightarrow{C}_R \leftarrow \overrightarrow{C}_R + \alpha(\overrightarrow{C}_{PR} - \overrightarrow{C}_R)$, such that the plane defined by $\overrightarrow{W}^{(2)} = (\overrightarrow{C}_R + \alpha(\overrightarrow{C}_{PR} - \overrightarrow{C}_R)) - \overrightarrow{C}_{\bar{R}}$, passes through the points $\overrightarrow{C}_{PR}$ and $\overrightarrow{C}_{NA}$. Similarly we find that the parameter α is determined by:

$$\alpha = -\frac{\overrightarrow{W}^{(1)} \cdot (\overrightarrow{C}_{PR} - \overrightarrow{C}_{NA})}{(\overrightarrow{C}_{PR} - \overrightarrow{C}_R) \cdot (\overrightarrow{C}_{PR} - \overrightarrow{C}_{NA})} \tag{5}$$

This alternation of the rotation towards either the NA or the PR continues until $|NA| = |PR| = 0$ or the number of iteration exceeds a predetermined value. This process converges to a common tangent hyperplane that leaves the sets R and $\bar{R}$ on opposite sides of the hyperplane.

3 Task Description

This year's challenge deals with two tasks about a new area called social bookmarking. Internet sites of this type offer to users the ability to share with each other material such as links to web pages, scientific publications and etc. The first task of this year's challenge deals with spam detection in such systems, whereas the second one deals with tag recommendations.

Spammers have already discovered that social bookmarking sites provide them with a large and continuously growing pool of potential customers. In fact it is much more attractive a spam post in a social bookmarking site than an annoying e-mail in someone's e-mail box. This task tries to identify such spam posts, using previously manually assigned as spam or not data from a well known social bookmarking site. The second task is about assisting the users when posting a new post, suggesting them with the appropriate tags that should accompany their post. The same non spam data used for the first task is used for training models for the second task. We have only submitted a solution for the first task, so in the rest of this paper we will refer to that task of the challenge.

The evaluation of the submissions was performed using the correct user labels as spammers or not and the participants ranked lists, with the Area Under the ROC (Receiver Operating Characteristics) Curve as an evaluation measure.

3.1 Data Description

The data used to train the model for the spam task, was taken by a well known social bookmarking site. Users of this site post their favorite links or publications, along with tags they assign to them. A snapshot was taken and all these posts where manually assigned as spam or not spam. More precisely each user of the site was labeled as spammer or not based on his posts. The final raw data where organized in 7 text files. 3 of them (tas, bookmark, bibtex) contain the data of spammers, 3 (tas_spam, bookmark_spam, bibtex_spam) of them contain the data of not spammers and the final file (user_spam) contains the true labels of the users as spammers or not.

Each post of a user may be a link to a web page (bookmark) or a link to a publication (bibtex). Files bookmark_spam and bookmark contain the bookmark posts of spammer and not spammer users, files bibtex_spam and bibtex contain the publication posts respectively and finally the files tas_spam and tas contain the posts of the users along with the assigned tags and references to the appropriate bookmark or bibtex entry.

In the first step of the preprocessing stage of the data we had to organize the raw text in a user basis. This means that for each user, spammer or not, we unified all of his posts, bibtex or bookmark. Each user was identified by a unique id and after this step for each user id we had a text fragment representing all of his posts, which means a text containing all the tags, all the bookmark posts and all the bibtex posts.

The second step of the preprocessing stage removes any unnecessary text, like common or stop words, performs stemming, using Porter's stemmer, and finally extracts the features of the dataset and produces the vectors of the users.

The final train dataset contains 31,715 user vectors (29,248 for spammers and 2,467 for not spammers). We found 244 spam and 84,298 not spam unique bibtex entries, 1,626,560 spam and 176,147 not spam unique bookmark entries.

The dimensionality of the vector space was 480,062, which means that the final datasets contains 480,062 unique features or words. Finally each vector was normalized to unity length.

4 Experimental Results

Using the user vectors defined from the training dataset, a model of our algorithm was trained. In order to evaluate the performance and due to the luck of any test dataset, the initial train set was randomly divided into two equal subsets. The first was used for training the model and the second for testing.

This training dataset contains 14,624 spam user vectors and 1,234 non spam user vectors. The evaluation dataset contains 14,624 spam user vectors and 1,233 non spam user vectors.

After training the model and applying it to the evaluation data, it correctly classified as spammers 14,403 users (TP, True Positives), correctly classified as not spammers 884 users (TN, True Negatives), incorrectly classified as spammers 349 users (FP, False Positives) and incorrectly classified as not spammers 221 users (FN, False Negatives). This means a F1 measure of 98.06%. Using the provided by the organizers of the challenge script for calculating the AUC value, a 96.20% AUC value was estimated. Figure 3 shows the ROC curve generated using the script we mentioned.

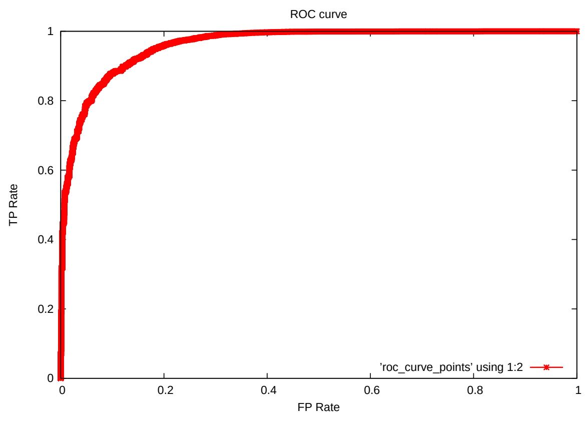

Fig. 3. ROC curve of the training set of the challenge

4.1 Test Dataset

The format of the test dataset was the same as the train dataset. After applying the same processing as in the train dataset case, the test dataset consisted of 7,205 user vectors.

Training the model with the full train dataset (31,715 user vectors), and applying it to the unlabelled test dataset, a list of the 7,205 users along with the determined by the model confidence values was submitted to the challenge.

By the end of the challenge, the true user labels where known and we where able to determine an AUC value of 97.96% and produced the ROC curve for the test dataset shown in the figure 4 below.

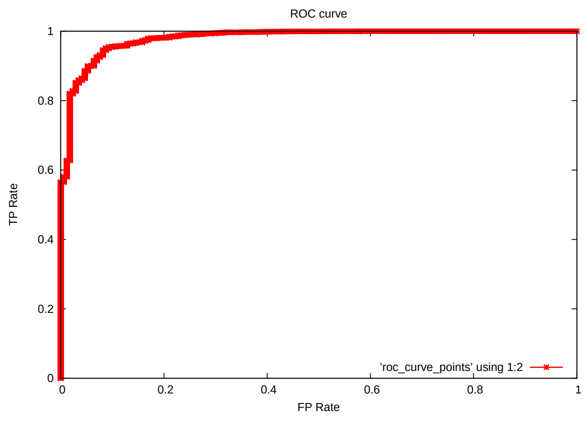

Fig. 4. The ROC curve of the testing set of the challenge

For a comparison of our method, we ran the challenge task with SVMs [4]. As it is reported in [5] the use of linear SVMs is both accurate and fast (to train and to use), so SVM's results were obtained using the SVMLight software package [6], with the default parameter values and a default linear kernel. Using the above mentioned script the AUC value was determined at 97.55%. We observe that our method outperforms SVMs both in accuracy and efficiency. Although the difference in performance is not significant, the training time of our algorithm took half the time of that of SVMs (both methods, where executed on the same machine). However the programming code of our algorithm in its current version is not optimized, and a new version is under development which will dramatically reduce the training time.

5 Concluding Remarks

In concluding, we have briefly described a novel algorithm for text categorization that is accurate and fast and we have demonstrated its performance in the ECML challenge of this year. The algorithm has been also tested on several standard text collections and showed a robust and comparable or even better performance compared with SVMs. As we have already mentioned the proposed algorithm overcomes both drawbacks of SVMs both in the training complexity and memory requirements. Definitely there are many details and extensions to the algorithm which are currently under investigation and will be published in due course.

References

1. Joachims, T.: Text categorization with support vector machines: learning with many relevant features (1998)
2. Rocchio, J.J.: Relevance feedback in information retrieval. In Salton, G., ed.: The SMART retrieval system: experiments in automatic document processing. Prentice-Hall, Englewood Cliffs, USA (1971) 313–323
3. Gkanogiannis, A., Kalampoukis, T.: An algorithm for text categorization. In: 31st ACM International Conference on Research and Development in Information Retrieval SIGIR-2008. (2008) 869–870
4. Cristianini, N., Shawe-Taylor, J.: An Introduction To Support Vector Machines (and other kernel-based learning methods). Cambridge University Press (2000)
5. Dumais, S., Platt, J., Heckerman, D., Sahami, M.: Inductive learning algorithms and representations for text categorization (1998)
6. Joachims, T.: Making large-scale support vector machine learning practical. In: Advances in Kernel Methods: Support Vector Machines. (1998)