Using Tri-Training and Support Vector Machines for addressing the ECML-PKDD 2006 Discovery Challenge

Dimitrios Mavroeidis, Konstantinos Chaidos, Stefanos Pirillos, Dimosthenis Christopoulos and Michalis Vazirgiannis

Department of Informatics, Athens University of Economics and Business, Greece

Abstract. In this paper we present and analyze the methodological approach we have used for addressing the ECML - PKDD Discovery Challenge 2006. The Challenge was concerned with the identification of individual user's spam emails based on a centrally collected training set. The task descriptions of the discovery challenge indicated that we should deviate from the classical supervised classification paradigm and attempt to utilize semi-supervised and transductive approaches. The format of the training data (bag-of-words providing only word IDs), did not allow either for the use of Natural Language Processing (NLP) approaches, or for the use of standard spam-recognition strategies. The submitted model, which achieved 5^{th} place on Task A of the challenge, was derived by Tri-Training, a recent development in Semi-supervised algorithms research. Given a standard classifier, Tri-Training initially uses bagging to produce three diverse training datasets-classifiers, which are used for classifying the unlabeled data and incorporating them into the training set in a theoretically sound way. The classifier we have used within Tri-Training was Support Vector Machines (SVM) and more precisely the Sequential Minimal Optimization (SMO) implementation of WEKA. Moreover, we have used feature normalization and logistic regression models to produce continuous outputs. Apart from a detailed description and a discussion of the submitted model, this paper contains an extensive empirical evaluation of two popular semi-supervised classification algorithms: Transductive Support Vector Machines (TSVM) and Tri-Training.

1 Introduction

The ECML-PKDD discovery challenge 2006, was concerned with the construction of a spam recognition filter, based on previously classified emails. The organizers considered the spam training set to be collected centrally, by server based spam filters that can construct a labeled spam/non-spam training set using publicly available sources and spam traps. Although, the centralized collection of the labeled training set presents several advantages, it is probable that the distribution of the server - collected training set is different than the distribution of the emails received by individual users. This raises the need of deviating from the classical supervised classification paradigm where the goal of classification algorithm is to minimize the expected error over instances taken from the same distribution as the training set, and utilize semi-supervised [1] or transductive [2] approaches that take into account the distribution of the test set (individual user's inboxes), where predictions should be made.

Semi-supervised algorithms take into account both labeled and unlabeled instances and attempt to find a balance between generalization (using the labeled examples) and adaptation (using the unlabeled examples) to construct a model that generalizes well on the whole space of labeled and unlabeled data. A slightly different research paradigm, that is related to semi-supervised learning is presented by transductive learning. The transductive paradigm considers a set of labeled training data and a set of unlabeled test data, and the goal is to perform predictions only on the test data (and not on the whole space of training and test data). Research on semi-supervised algorithms has been receiving increasing attention, and several algorithms have been proposed (i.e. Transductive Support Vector Machines (TSVMs) [3], Tri-Training [4], Spectral Graph Transduction [5]). The Discovery Challenge presents an excellent opportunity for evaluating these algorithms empirically and for exploring possible strategies for tuning their parameters effectively.

In the context of Task A of the Discovery Challenge we have conducted extensive experiments using TSVMs and Tri-Training. The submitted model that yielded the best result on the tuning data and achieved 5^{th} place in Task A of the contest, was derived by the Tri-Training algorithm. Given a classifier and a set of labeled and unlabeled data, Tri-Training initially constructs three diverse datasets-classifiers using bagging [6]. Subsequently, it uses an incremental procedure, where in each round, an unlabeled example is added in the training set of a classifier if the other two classifiers agree on the class label and certain theoretical criteria are met. The classifier used in the context of Tri-Training was the SVM [2] and more precisely the SMO [7] implementation of WEKA [8]. The SVM was parameterized by a linear kernel with complexity parameter $C = 0.015$. Moreover, we have used feature normalization and logistic regression models in order to produce continuous output.

The rest of the paper is organized as follows. Section 2 provides a short description of the Discovery Challenge. Section 3 presents the Data preprocessing strategies we have experimented with. Section 4 analyzes the model evaluation approaches we have used. Section 5 describes the learning algorithms used and presents the experimental results. Section 6 discusses the results and contains the concluding remarks.

2 Discovery Challenge Description

2.1 Task Description

The discovery challenge consisted of two tasks, Task A and Task B. Task A was concerned with the case where the size of the centrally collected training data was larger than the individual users' inboxes. More precisely, the centrally collected training set contained 4000 emails, and the three individual users inboxes, where predictions should be made, contained 2500 emails each. Task B was concerned with the case where the size of the centrally collected training data was small in comparison to the size of the inboxes of the individual users. In task B, the centrally collected training data contained 100 emailes, while predictions should be made on 15 user inboxes, containing 400 emails each. In order to tune the parameters of the algorithms, tuning data was provided for both tasks. Since we have submitted a solution only for task A of the challenge, in the rest of the document unless otherwise stated, we will refer to Task A of the challenge.

The evaluation of the submissions was performed using the correct class labels for the individual user's inboxes (where the predictions were made), with the Area Under the Receiver Operating Characteristics (ROC) Curve (AUC) as an evaluation measure. The ROC curve was originally introduced in the signal processing community for addressing the problem signal detection, and it has been utilized in various contexts (i.e. model selection [9], introduction of algorithms that optimize the AUC measure [10]) by the machine learning research community.

2.2 Data Description

The methodological approaches that could be utilized in the Discovery Challenge, were determined by the form of the data provided. The discovery challenge datasets were delivered in the bag-of-words representation, where the words were represented by numeric IDs. This prevented the contestants from using any Natural Language Processing (NLP) techniques that could enhance the performance of the learning algorithms. However, taking into account the fact the NLP techniques are receiving increasing attention from the machine learning community (i.e. Word Sense Disambiguation for Text Classification [11]), it would have been interesting if more information were provided, that allowed the use of NLP techniques.

Moreover, the email representations excluded the use of any traditional spam filter methods. Such methods are DNS Black-hole Lists (DNSBLs), which reject the email that come from certain IPs (e.g. dynamic and dial-up IP addresses), keyword-based filtering (e.g. block the emails that contain certain phrases), checksum-based filtering, which takes advantage of the fact that usually the spam emails that are sent by an individual user are almost identical, and several others. Providing such additional information would pose the challenge of deriving decision functions that combine spam recognition rules and the statistical models constructed by the learning algorithms. Although, this is the task that real world spam filters must achieve, this would probably be out of the scope of the ECML-PKDD conference.

3 Data preprocessing

3.1 Feature Selection

Although feature selection has been shown to improve algorithms' performance in several learning tasks and application areas, experimental results have suggested that text classification algorithms should not be expected to benefit from aggressive feature selection [12]. This is because most words (with the exception of stop-words and common terms, issues that can be dealt using stop word lists and term weighting) offer important information for the correct classification of text data. Whatsoever we have investigated possible benefits from using some popular feature selection algorithms.

The feature selection measures we have considered are the Bi-Normal Separation (BNS) metric and the Information Gain (IG). BNS is defined as: $F^{-1}(P(word|+)) -$

$F^{-1}(P(word|-)))$ where F^{-1} is the inverse of the cumulative probability function of the Normal Distribution. For a theoretical analysis of the BNS metric in the context of ROC analysis the interested reader can refer to [13]. The IG is defined as the difference in entropy caused by the existence of a feature. The IG has been used widely in the context of feature selection and machine learning algorithms.

We have conducted experiments using BNS and IG with SVMs and Naive Bayes classifiers on the training and the tuning datasets. The algorithms performed always better with all the features. We have also attempted to use BNS and IG for features selection, prior to using the Tri-Training algorithm (which yielded the best results on the tuning data). In this case also the algorithm performed superiorly when all the features were retained.

3.2 Feature Normalization

In the context of text classification, it has been suggested by many authors (i.e. [14],[15]) that feature normalization can significantly boost the performance of learning algorithms and especially Support Vector Machines. This can be easily understood if we consider that in the unnormalized case, the similarities between the emails will be affected by the size of the emails (longer emails will contain terms with higher frequency of occurrence).

In order to verify the appropriateness of normalization empirically we have experimented using normalized and unnormalized features with Support Vector Machine in the training and tuning data. The experiments have showed that normalization improves the classification results and is thus appropriate in the context of the challenge.

4 Model Evaluation

In order to investigate the possible strategies for model evaluation, we will firstly recall some details concerning the datasets that were provided by the Discovery Challenge. The main training and test data consisted of a labeled training set (*TrainData*) and three individual user's inboxes (*TestDataA*,*TestDataB*,*TestDataC*), where the predictions should be made. For tuning the parameters of the algorithms the organizers provided additionally, a labeled training set (*TuneTrainData* and an individual user's inbox where the labels were provided as well (*TuneTestData*).

Using these datasets it is straight forward to evaluate the performance of the supervised learning algoriths using k-fold cross validation on the *TrainData* and the *TuneTrainData* datasets. However, since we are interested in performing predictions on the individual users inboxes, we should investigate possible model evaluation strategies that involve the test data. A straight forward approach would be to simply construct the models on *TuneTrainData* and then use the whole *TuneTestData* to estimate the models' performance. However, in order not to favor models that overfit the *TuneTestData*, we have evaluated the algorithms using cross validation on the combination of the training and the test set. More precisely, we have divided both the *TuneTrainData* and the *TuneTestData* data in k folds. Then, the k fold cross validation result is derived as the average AUC score of the k runs, where in each run we train the model on *TuneTrainData*-$\{$ Fold i of the *TuneTrainData*$\}$ and then measure the AUC of the model on the *TuneTestData*-$\{$ Fold i of the *TuneTestData* $\}$.

5 Learning Algorithms

5.1 Supervised

As we have mentioned in the introductory section, the Discovery Challenge was concerned with the construction of "personalized" spam filters for individual users based on a centrally constructed labeled training set. This implied that semi-supervised and transductive algorithms were appropriate. However since such algorithms are not guaranteed to achieve better performance than standard supervised inductive classifiers, we have initially conducted experiments using two popular classifiers, Naive Bayes (with a kernel density estimator, to address the problem of numeric features) and SVM. For breverity, we do not present here the details of these algorithms. The interested reader can find detailed descriptions in any standard Data Mining textbook (i.e. [16]).

Concerning SVM, we have experimented using a linear kernel, feature normalization and logistic regression models for producing continuous outputs. It has to be mentioned that the use of logistic regression derives proper probabilistic output for the algorithm, however it does not affect the AUC performance (compared to using decision function values). We have explored the effect of using various values for the complexity parameter C. The values we have used were powers of the 2, ranging from 2^{-6} until 2^1. The results of the 10 fold cross validation on the *TrainData* and the *TuneTrainData*, did not vary significantly for the different values of C and the mean AUC was consistently above 0.98. The mean AUC scores of the 10 fold cross validation of the combined *TuneTrainData* and *TuneTestData* (as described in section 4), are reported on Figure 1.

Concerning the Naive Bayes, we have used Kernel Density estimator in order to address the problem of numeric features. The 10 fold cross validation results on the *TrainData* produced and average AUC score of 0.76, and similarly 0.80 on *TuneTrainData*. The average AUC derived by the 10 fold cross validation on the combined *TuneTrainData* and *TuneTestData*, was 0.37. This result signified that the distribution of the individual user's inbox was significantly different than the distribution of the centrally collected training data, and that standard Naive Bayes is highly inappropriate in the context of this challenge.

5.2 Transductive and Semi-Supervised

In our experimental evaluation of transductive and semi-supervised approaches, we have concentrated on two algorithms, TSVMs and Tri-Training. The TSVMs present the transductive version of the popular SVM classifier. The main intuition behind TSVM, is that instead of searching for the separating hyperplane that maximizes the margin between the two classes (as in SVM), it searches for the hyperplane that maximizes the margin between both training (labeled) and test (unlabeled) data.

Concerning TSVM, we have used the SVM-Light implementation available on the web site of T. Joachims (*http://www.cs.cornell.edu/People/tj/*). We have experimented

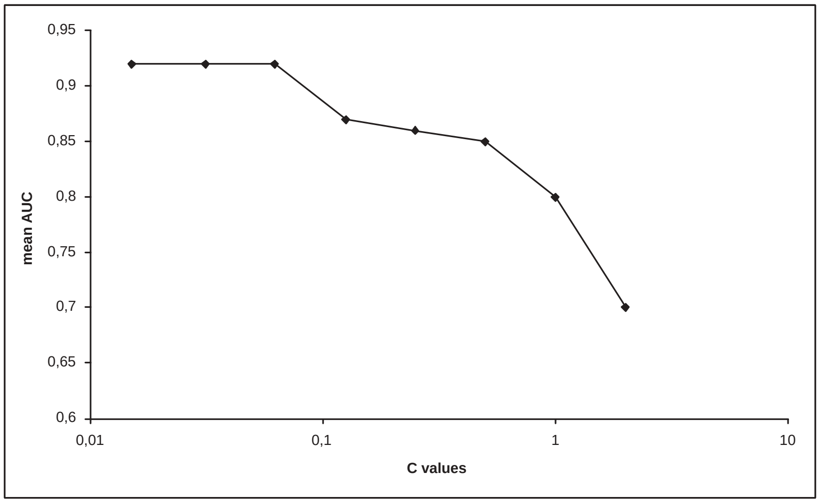

Fig. 1. SVM performance

using both Linear (inner product) and RBF Kernels. Concerning the Linear kernel we have used values that are powers of 2, ranging from 2^{-7} to 2^{12}. The 10 fold cross-validation results (in the fashion described in Section 4) for the various values of C are reported in Figure 2. Concerning the RBF kernel, we have experimented using values of C ranging from 2^5 to 2^{12} and γ values ranging from 2^{-4} to 2^{12}. The average AUC scores for these parameters of the RBF kernel were very low (consistently under 0.55). It has to be noted that in the TSVM experiments we had not normalized the feature space. This is because in the time we have deduced that normalization was appropriate, there was not adequate time for performing the normalized TSVM experiments.

We have also considered the Tri-Training algorithm, which has produced the best result on the cross validated test data. Tri-Training uses as input a supervised learning algorithm and a set of labeled and unlabeled instances. Subsequently, it uses bagging in order to produce three diverse training sets-classifiers. The main Tri-Training algorithm is based on an incremental procedure, where at each step, an instance is added to the training set of a classifier, if the other two classifiers agree on its label, and certain theoretical criteria are met. The criteria used, provide theoretical guarantees that the expected error of the classifier will be reduced when the new labeled example is added. For breverity we do not reproduce here all the details of the Tri-training algorithms, in [4], the interested reader can find the theoretical and empirical evidence for the appropriateness of the Tri-Training algorithm for semi-supervised learning tasks.

In the experimental evaluation we have used the Tri-Training implementation that is available on the web site of Ming Li, (*http://lamda.nju.edu.cn/lim/*), the second author of [4]. Prior to using Tri-Training, we have normalized the feature space. The classifier used for Tri-Training was an SVM and more precisely the SMO [7] implementation of WEKA [8]. Moreover, we have used logistic regression models for producing continuous outputs. It has to be noted that the use of logistic regression derives proper

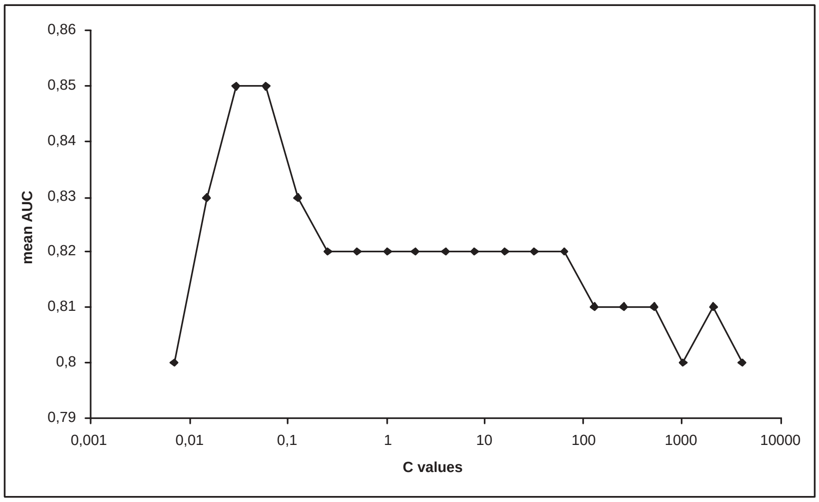

Fig. 2. TSVM performance

probabilistic output for the algorithm, however it does not affect the AUC performance (compared to using decision function values). In the experiments we have used a Linear Kernel (inner product) with various values of C. The values of C were again powers of 2, ranging from 2^{-7} to 2^{3}. In Figure 3, we report the average AUC scores derived from 10-fold cross validation (in the fashion described in Section 4). The best AUC score: 0.96 was achieved with the value $C = 0.015$.

6 Discussion - Conclusions

Based on the experimental results, an interesting observation that can be made concerns the kernel function, we have used in the submitted model. The Linear Kernel (inner product), yielded the best results on the tuning data among the algorithms we have experimented with and achieved 5^{th} place on Task A of the discovery challenge. Linear kernels are known to suffer from underfitting and it is general appreciated that they are not expressive enough for modeling complex real world data. Our experiments serve as an indication, that with the appropriate choice of C, linear kernels can be successfully applied in real world text classification problems.

Moreover, our experimental results have verified that Normalization can significantly improve classification performance of learning algorithms and that feature selection may not always be appropriate. Although these observations are widely known and have been discussed in various research papers (i.e. in [14, 15, 12]), our experimental results provide additional empirical verification.

Concerning the experimental results of the Tri-Training algorithm, it can be observed that the average AUC is more sensitive with respect to the C parameter than when using SVM and TSVM. An argument that can be used for explaining the sensitivity of Tri-Training, is that since in Tri-Training the results of the classifier are used

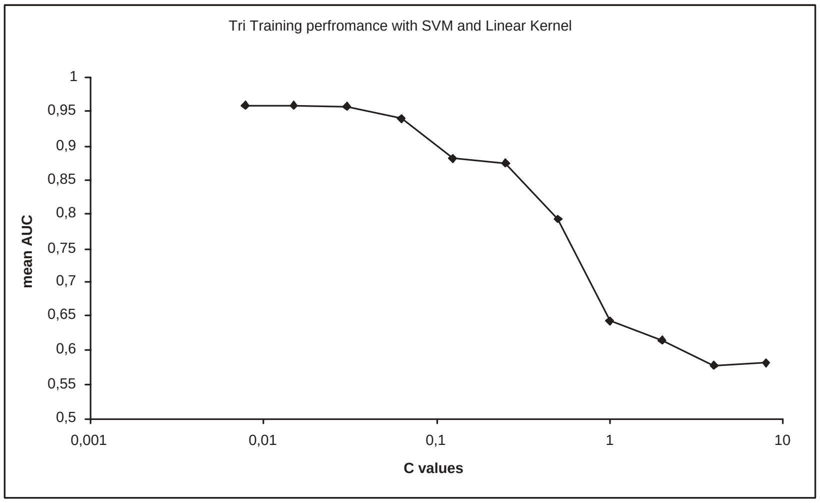

Fig. 3. Tri-Training performance

for adding unlabeled instance into the training set, small changes in the performance of the classifier may result in the addition of noise in the training set. Thus, a small reduction in the performance of the classifier may result in a much larger reduction of the performance of the Tri-Training algorithm. This signifies the importance of parameter tuning for semi-supervised algorithms that work by using a classifier to add the unlabeled instance into the training set (i.e. self-training).

In conclusion, we consider that the Discovery Challenge organized within the ECML - PKDD 2006, provided an excellent opportunity for the empirical evaluation of semi supervised algorithms. The experimental results can be useful both for theoretical and applied research, for understanding the properties of semi-supervised algorithms and identifying situations under which they should be expected to perform well.

References

1. Chapelle, O., Schölkopf, B., Zien, A., eds.: Semi-Supervised Learning. MIT Press, Cambridge (2006)
2. Vapnik, V.: Statistical Learning Theory. Wiley (1998)
3. Joachims, T.: Transductive inference for text classification using support vector machines. In: ICML. (1999) 200–209
4. Zhou, Z.H., Li, M.: Tri-training: Exploiting unlabeled data using three classifiers. IEEE Trans. Knowl. Data Eng. **17**(11) (2005) 1529–1541
5. Joachims, T.: Transductive learning via spectral graph partitioning. In: ICML. (2003) 290–297
6. Breiman, L.: Bagging predictors. Machine Learning **24**(2) (1996) 123–140
7. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In Scholkopf, B., Burges, C., Smola, A., eds.: Advances in Kernel Methods - Support Vector Learning, MIT Press (1998)

8. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann (2005)
9. Provost, F.J., Fawcett, T.: Robust classification for imprecise environments. Machine Learning **42**(3) (2001) 203–231
10. Ferri, C., Flach, P.A., Hernández-Orallo, J.: Improving the auc of probabilistic estimation trees. In: ECML. (2003) 121–132
11. Mavroeidis, D., Tsatsaronis, G., Vazirgiannis, M., Theobald, M., Weikum, G.: Word sense disambiguation for exploiting hierarchical thesauri in text classification. In: PKDD. (2005) 181–192
12. Forman, G.: An extensive empirical study of feature selection metrics for text classification. Journal of Machine Learning Research **3** (2003) 1289–1305
13. Hanley, J.: The robustness of the binormal assumptions used in fitting roc curves. Medical Decision Making **8** (1998) 197–203
14. Herbrich, R., Graepel, T.: A pac-bayesian margin bound for linear classifiers: Why svms work. In: NIPS. (2000) 224–230
15. A.B.A. Graf, A.S., Borer, S.: Classification in a normalized feature space using support vector machines. IEEE Transactions on Neural Networks **14** (2003) 597–605
16. Hand, D.J., Mannila, H., Smyth, P.: Principles of Data Mining. MIT Press (2001)