Learning from Labeled and Unlabeled Documents: A Comparative Study on Semi-Supervised Text Classification

Carsten Lanquillon

DaimlerChrysler AG
Research and Technology 3
D-89013 Ulm, Germany
carsten.lanquillon@daimlerchrysler.com

Abstract. Supervised learning algorithms usually require large amounts of training data to learn reasonably accurate classifiers. Yet, for many text classification tasks, providing labeled training documents is expensive, while unlabeled documents are readily available in large quantities. Learning from both, labeled and unlabeled documents, in a semi-supervised framework is a promising approach to reduce the need for labeled training documents. This paper compares three commonly applied text classifiers in the light of semi-supervised learning, namely a linear support vector machine, a similarity-based tfidf and a Naïve Bayes classifier. Results on a real-world text datasets show that these learners may substantially benefit from using a large amount of unlabeled documents in addition to some labeled documents.

1 Introduction

With the enormous growth of on-line information available through the World Wide Web, corporate intranets, electronic news feeds, and other sources, the problem of automatically classifying text documents into predefined categories is an important issue in many information organization and management tasks.

These classification problems can be solved by applying supervised learning algorithms to learn classifiers from labeled training examples that predict the class label of new, previously unseen documents. In order to learn reasonably accurate classifiers, we must be provided with enough labeled training examples. This commonly requires a person to read many documents and to decide on the class label to be given to each of these documents. This is a tedious and time consuming process. Thus, for complex learning tasks, providing sufficiently large sets of labeled training examples becomes prohibitive. By contrast, unlabeled documents are often readily available in large quantities. Therefore, a promising idea is to utilize unlabeled documents in addition to labeled documents when learning a classifier. We refer to learning from both, labeled and unlabeled data, as semi-supervised learning. See [6,9] for a discussion of related work.

In this paper, we empirically compare three commonly applied text learning algorithms, namely a linear support vector machine, a similarity-based tfidf and

D.A. Zighed, J. Komorowski, and J. Żytkow (Eds.): PKDD 2000, LNAI 1910, pp. 490–497, 2000.

a Naïve Bayes classifier in a semi-supervised framework [5,6,9]. Classification accuracy is reported on an independent test set. Further, we investigate the effect on semi-supervised learning when varying the number of features.

The remainder of this paper is organized as follows. Section 2 gives a brief introduction to text classification and some traditional learning algorithms. In Section 3, a general framework for learning from labeled and unlabeled documents is described. Initial experimental results that support the idea of semi-supervised learning are reported in Section 4. Section 5 concludes this paper.

2 Text Classification

The task of text classification is to automatically classify documents into a predefined number of classes. Each document can be in multiple, exactly one, or no class. In the experiments presented in Section 4, the task is to assign each document to exactly one class. When using supervised learning algorithms in this particular setting, a classifier can try to represent each class simultaneously. Alternatively, each class can be treated as a separate binary classification problem where each binary problem answers the question of whether or not a document should be assigned to the corresponding class [5].

2.1 Document Representation

In information retrieval, documents are often represented as feature vectors, and a subset of all distinct words or word stems occurring in the given documents are used as features. Words that frequently occur in many documents (*stop words* such as "and", "or" etc.) or words that occur only in very few documents may be removed. Further, measures such as *information gain* can be used for feature selection [14]. Each feature is given a weight which depends on the learning algorithm at hand. This leads to an attribute-value representation of text. Possible weights are, for example, binary indicators for the presence or absence of features, plain feature counts—*term frequency (tf)*—or more sophisticated weighting schemes, such as multiplying each term frequency with the *inverted document frequency (idf)* [11]. Finally, each feature vector may be normalized to unit length to abstract from different document lengths.

2.2 Learning Algorithms

A variety of text learning algorithms has been studied and compared in the literature, for example see [1,2,13]. We focus on three widely applied classifiers.

One of the learning algorithms we apply is the multinomial Naïve Bayes classifier (NB) which uses plain term frequency as feature weights [9]. The idea of the Naïve Bayes classifier is to use the joint probabilities of words (features) and classes to estimate the probabilities of the classes given a document. A document is then assigned to the most probable class. It is computationally very efficient because of the simplifying assumptions that the feature weights are conditionally

independent given the class label. Although this independence assumption is usually violated in practice, this method often yields good performance [2,14].

Further, we use a similarity-based method based on *tfidf* weights which we denote as *single prototype classifier (SPC)*. It is a variant of Rocchio's method for relevance feedback [10] applied to text classification and is also described as the *Find Similar* algorithm in [1]. The classifier models each class with exactly one prototype computed as the average (centroid) of all available training documents belonging to that class. We use a scheme for setting feature weights which is denoted as *ltc* in the Smart system. A document is assigned to the class of the prototype to which it has the largest cosine similarity.

Support vector machines (SVMs) are successfully applied to text classification problems [1,3,13]. SVMs can only solve binary classification problems. More complex classification tasks must be composed of binary decisions. We apply a SVM in its basic form, which is a hyperplane that separates examples of the two classes with maximum margin. The class label of a new document is determined based on which side of the hyperplane it is.

3 Semi-Supervised Learning

This section discusses the use of unlabeled in addition to labeled documents when learning a text classifier. Having given an argument for the use of unlabeled documents, we describe a general framework for semi-supervised learning.

3.1 Why Does Using Unlabeled Data Help?

As pointed out in [5,9], it is well known in information retrieval that words in natural language occur in strong co-occurrence patterns [12]. While some words are likely to co-occur in one document, others are not. When using unlabeled documents we can exploit information about word co-occurrences that is not accessible from the labeled documents alone. Albeit independent from the class labels, this information can help to enhance classification accuracy.

3.2 General Framework

We use a general framework for learning from labeled and unlabeled documents that can be instantiated with any learning algorithm [6]. It is a generalization of the approach proposed in [9]. The class labels of unlabeled documents are treated as missing values, and an EM-like scheme is used to alternately predict the missing class labels and build a new classifier based on both, the labeled and unlabeled documents together with the predicted pseudo class labels as described below. Table 1 gives an outline of this general framework.

Given a set of training documents D, for some subset of the documents $d_i \in D^l$ we know the true class labels, and for the rest of the documents $d_i \in D^u$, the class labels are unknown. Thus we have a disjoint partitioning of our training documents into a labeled set and an unlabeled set of documents $D = D^l \cup D^u$.

Table 1. Framework for semi-supervised learning.

- **Inputs:** Sets D^l and D^u of labeled and unlabeled documents.
- Build initial classifier, H, based only on the labeled documents, D^l.
- Loop while class memberships of unlabeled documents, U^u, change:
  - **(E-step)** Use current classifier, H, to score unlabeled documents.
  - Transform scores of unlabeled documents into class memberships, U^u.
  - **(M-step)** Re-build H based on D^l and D^u, with labels obtained from U^u.
- **Output:** Classifier, H, for predicting class labels of unseen documents.

The task is to build a classifier based on the training documents, D, for predicting the class label of new, previously unseen documents.

First, an initial classifier, H, is build based only on the labeled documents, D^l. Then the algorithm iterates the following three steps until the class memberships given to the unlabeled documents, D^u, by the current classifier, H, do not change from one iteration to the next. Corresponding to the **E-step**, the current classifier, H, is used to classify each unlabeled documents. The classifier may respond with any type of classification scores whose interpretation need not be probabilistic. In order to abstract from the classifier's response, in the next step we transform these scores into class memberships, yielding a class membership matrix, $U^u \in [0,1]^{c \times |D^u|}$, where c is the number of classes. The sum of class memberships of a document over all classes is assumed to be one. Possible transformations are, for instance, normalizing the scores to unity which yields soft memberships, or using hard memberships, e.g. setting the largest score to one and all other scores to zero. The choice of the transformation function depends on the classifier at hand such that the classifier knows how to make use of the class membership matrix, U^u. Notice that using hard memberships always allows us to use any traditional supervised learning algorithm. Now, provided with the class membership matrix, U^u, a new classifier, H, can be build from both, the labeled and unlabeled documents. This corresponds to the **M-step**. The final classifier, H, can then be used to predict the class labels of new, previously unseen examples.

3.3 Instantiations

In order to apply this algorithmic framework, the underlying classification algorithm, and the function for transforming classification scores must be specified. In the following, we instantiate the framework with the three text learning algorithms described in Section 2 to obtain semi-supervised text learners.

When using a Naïve Bayes classifier and leaving the resulting probabilistic classification scores unchanged, we end up with the algorithm given in [9]. This instantiation, which we refer to as *ssNB*, has a strong probabilistic framework and is guaranteed to converge to a local minimum [9].

Next, we will apply the single prototype classifier in combination with a transformation of classification scores into hard class memberships, yielding the

ssSPC classifier. Initial experiments showed that using soft class memberships for this classifier does not yield reasonable results. Notice that this instantiation turns out to be a variation of the well known *hard k-means* clustering algorithm [7]. The difference is that the memberships of the labeled documents remain fixed during the clustering iterations. The ssSPC algorithm is guaranteed to converge to a local minimum after a finite number of iterations [6].

Finally, we use a linear SVM with hard class memberships in the semi-supervised framework to yield the *ssSVM* classifier. Obviously, using soft class memberships for a conventional SVM does not make sense because an example is always on exactly one side of the separating hyperplane (unless it is on that hyperplane). Since we do not have any guarantee of convergence, we specify a maximum number of iterations for the EM scheme.

4 Experimental Results

This section gives empirical evidence that combining labeled and unlabeled documents with certain text classifiers using the algorithmic framework in Table 1 can improve traditional text classifiers. Due to space limitation, experimental results are reported on only on the 20 Newsgroups dataset, which is publicly available at *http://www.cs.cmu.edu/∼textlearning*. We use a modified version of the *Rainbow* system and the to run our experiments [8]. In addition, the SVM^{light} implementation is used to learn SVMs [4]. We follow the description in [9] to setup the experiments with the semi-supervised classifiers described Section 3, with the corresponding traditional supervised learners, and with a transductive linear support vector machine [5].

4.1 Dataset and Protocol

The 20 Newsgroups dataset consists of 20017 articles divided almost evenly among 20 different UseNet discussion groups [2]. The task is to classify an article into the one of the twenty newsgroups to which it was posted. When tokenizing the documents, UseNet headers are skipped, and tokens are formed from contiguous alphabetic characters. We do not apply stemming, but remove common stop words. The remaining words are used as vocabulary. We create a test set of 4000 documents and an unlabeled set of 10000 documents. Labeled training sets are formed by partitioning the remaining documents into non-overlapping sets. All sets are created with equal number of documents per class. Where applicable, up to ten trials with disjunct labeled training sets are run for each experiment. Results are reported as averages over these trials.

4.2 Results

Figure 1 shows the effect of using unlabeled documents in addition to labeled documents when learning a classifier with the different classifiers. The learning curve shows the classification accuracies of the traditional and semi-supervised

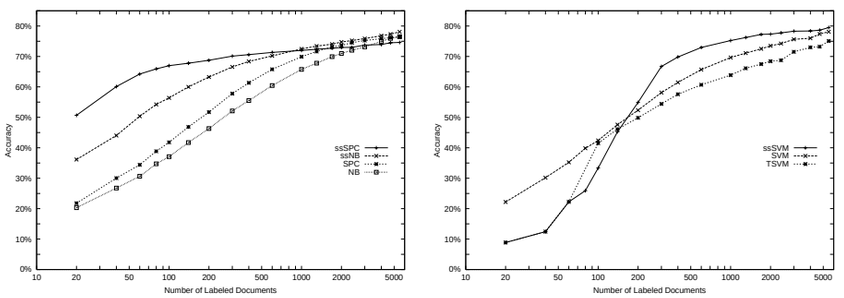

Fig. 1. Classification accuracy of the partially supervised learning framework (ss) using the single prototype classifier (SPC), the Naïve Bayes classifier (NB) and the linear support vector machine (SVM) compared to the traditional classifiers and a transductive linear support vector machine (TSVM) on the 20 Newsgroups dataset.

single prototype classifer (SPC) and Naïve Bayes classifier (NB) (left) and the traditional, semi-supervised and transductive linear SVM (right) when varying the number of labeled documents. The horizontal axes indicate the number of labeled training documents on a log scale. Notice, for instance, that 20 training documents correspond to one training document per class. The vertical axes indicate the average classification accuracy on the test sets.

Traditional SPC and SVM achieve approximately the same results. Only with many labeled training documents, the SVM slightly outperforms the SPC by about two points. The traditional NB is generally worse than SPC and SVM. The learning curves of the plain supervised learners illustrate that learning reasonably accurate classifiers requires a large amount of labeled training documents.

The semi-supervised SPC and NB perform substantially better than the traditional variants when the amount of labeled training documents is small. For instance, with 100 training examples, ssSPC achieves 67% accuracy on the test set while the traditional SPC reaches only about 42%. Thus, the classification error is reduced by about 43%. As one would expect, the more labeled documents are available, the smaller the performance increase. For large numbers of labeled documents, accuracy even slightly degrades when using unlabeled documents with ssSPC.

The results of the semi-supervised SVM are different. When only a small amount of labeled documents is available, adding unlabeled documents drastically hurts performance as compared to plain SVM. With more labeled data, however, ssSVM also starts to outperform its traditional variant. Compared to SPC and NB, learning a SVM involves a much more complex search. We thus hypothesize that the SVM requires more labeled documents in order to benefit from the unlabeled documents. Further note that using the SVM in the semi-supervised setting requires much more computing time than the other learning algorithms due to the more complex search. The transductive SVM cannot compete at all. It starts as bad as ssSVM and never gets better than plain SVM. Note, howe-

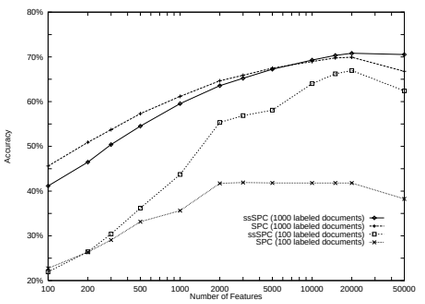

Fig. 2. Classification accuracy of traditional and semi-supervised SPC when varying the number of features for two amounts of labeled training examples.

ver, that the transductive SVM is actually designed to enhance performance on the unlabeled data used during training rather than on an independent test set. And in fact, further experiments show that TSVM outperforms SVM on the unlabeled dataset in the presence of many labeled training examples.

In Section 3 we argue that classification accuracy can be increased by exploiting co-occurrence patterns in unlabeled documents. The performance should therefore crucially depend on the number of features. Figure 2 shows the accuracies of the traditional and semi-supervised SPC for 100 and 1000 labeled training documents when varying the number of features from 100 to more than 50000 feature, which comes close to using all features. For 1000 labeled training documents, using unlabeled documents does not help a lot building the classifier. And this does not depend on the number of features. The respective curves are very similar. Things are quite different when using only 100 labeled documents. Here we have a rather large performance gain. Except for the decrease towards the end, using more feature generally yields a larger performance increase. When using too few features, we do not get any help from the unlabeled documents.

5 Conclusions and Future Work

This paper compared three commonly applied text classifiers in the light of semi-supervised learning with labeled and unlabeled documents, which is an important issue when hand-labeling documents is expensive but unlabeled documents are readily available in large quantities.

Empirical results on a real-world text classification task show that the single prototype classifier and the Naïve Bayes classifier can benefit substantially when incorporating unlabeled documents with some labeled documents into the learning process. We also see some improvement in classification accuracy when using support vector machines. However, the results are not as consistent as those for the single prototype and the Naïve Bayes classifiers. The support vector machines seem to require more labeled documents in order to benefit from a large set of unlabeled documents. Note that adding unlabeled documents to a

larger number of labeled training documents even hurts classification accuracy in some cases. Future work will focus on preventing the unlabeled documents from degrading performance. An interesting approach is to introduce a weight to adjust the contribution of unlabeled documents [9].

Further experiments with a semi-supervised k-nearest neighbor rule show no improvement at all. This is because there is no proper learning process. Consequently, this algorithm does not generalize and it cannot make use of additional co-occurrence patterns inherent to unlabeled documents. Note, that this supports our argument for using unlabeled documents given in Section 3.

An important finding is that in the semi-supervised setting the feature set should not be reduced too much since additional features may contain valuable information even though cross-validation on the labeled training data would suggest using less features. Note that the calculation of many common feature selection measures depend on the class labels. For unlabeled documents, however, this label is missing. Therefore, an interesting research issue is to use feature selection methods that are especially developed for clustering tasks.

References

1. S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representation for text categorization. In *Proceedings of the Seventh International Conference on Information and Knowledge Management*, 1998.
2. T. Joachims. A probabilistic analysis of the Rocchio algorithm with tfidf for text categorization. In *Proceedings of ICML'97*, 1997.
3. T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of ECML'98*. Springer Verlag, 1998.
4. T. Joachims. *Advances in Kernel Methods – Support Vector Learning*, chapter Making large-Scale SVM Learning Practical. MIT-Press, 1999.
5. T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of ICML'99*, 1999.
6. C. Lanquillon. Partially supervised text classification: Combining labeled and unlabeled documents using an EM-like scheme. In *accepted at ECML2000*, 2000.
7. J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.
8. A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. `http://www.cs.cmu.edu/~mccallum/bow`, 1996.
9. K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 2000. To appear.
10. J. J. Rocchio Jr. Relevance feedback in information retrieval. In *The SMART Retrieval System*. Prentice-Hall, 1971.
11. G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, 1989.
12. C. J. van Rijsbergen. A theoretical basis for the use of co-occurrence data in information retrieval. *Journal of Documentation*, 33(2):106–119, 1977.
13. Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceeding of the ACM SIGIR Conference*, pages 42–49, 1999.
14. Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of ICML'97*, pages 412–420, 1997.