

Text Mining Techniques for Leveraging Positively Labeled Data

Lana Yeganova*, Donald C. Comeau, Won Kim, W. John Wilbur
National Center for Biotechnology Information, NLM, NIH, Bethesda, MD 20894 USA
{yeganova, comeau, wonkim, wilbur}@mail.nih.gov

* Corresponding author. Tel.: +1 301 402 0776

Abstract

Suppose we have a large collection of documents most of which are unlabeled. Suppose further that we have a small subset of these documents which represent a particular class of documents we are interested in, i.e. these are labeled as positive examples. We may have reason to believe that there are more of these positive class documents in our large unlabeled collection. What data mining techniques could help us find these unlabeled positive examples? Here we examine machine learning strategies designed to solve this problem. We find that a proper choice of machine learning method as well as training strategies can give substantial improvement in retrieving, from the large collection, data enriched with positive examples. We illustrate the principles with a real example consisting of multiword UMLS phrases among a much larger collection of phrases from Medline.

1 Introduction

Given a large collection of documents, a few of which are labeled as interesting, our task is to identify unlabeled documents that are also interesting. Since the labeled data represents the data we are interested in, we will refer to it as the positive class and to the remainder of the data as the negative class. We use the term negative class, however, documents in the negative class are not necessarily negative, they are simply unlabeled and the negative class may contain documents relevant to the topic of interest. Our goal is to retrieve these unknown relevant documents.

A naïve approach to this problem would simply take the positive examples as the positive class and the rest of the collection as the negative class and apply machine learning to learn the difference and rank the negative class based on the resulting scores. It is reasonable to expect that the top of this ranking would be enriched for the positive class.

But an appropriate choice of methods can improve over the naïve approach.

One issue of importance would be choosing the most appropriate machine learning method. Our problem can be viewed from two different perspectives: the problem of learning from imbalanced data as well as the problem of recommender systems. In terms of learning from imbalanced data, our positive class is significantly smaller than the negative, which is the remainder of the collection. Therefore we are learning from imbalanced data. Our problem is also a recommender problem in that based on a few examples found of interest to a customer we seek similar positive examples amongst a large collection of unknown status. Our bias is to use some form of wide margin classifier for our problem as such classifiers have given good performance for both the imbalanced data problem and the recommender problem (Zhang and Iyengar 2002; Abkani, Kwek et al. 2004; Lewis, Yang et al. 2004).

Imbalanced data sets arise very frequently in text classification problems. The issue with imbalanced learning is that the large prevalence of negative documents dominates the decision process and harms classification performance. Several approaches have been proposed to deal with the problem including sampling methods and cost-sensitive learning methods and are described in (Chawla, Bowyer et al. 2002; Maloof 2003; Weiss, McCarthy et al. 2007). These studies have shown that there is no clear advantage of one approach versus another. Elkan (2001) points out that cost-sensitive methods and sampling methods are related in the sense that altering the class distribution of training data is equivalent to altering misclassification cost. Based on these studies we examine cost-sensitive learning in which the cost on the positive set is increased, as a useful approach to consider when using an SVM.

In order to show how cost-sensitive learning for an SVM is formulated, we write the standard equations for an SVM following (Zhang 2004).

Given training data $\{(x_i, y_i)\}$ where y_i is 1 or -1 depending on whether the data point x_i is classified as positive or negative, an SVM seeks that vector w_i which minimizes

$$\sum_i h(y_i(\bar{x}_i \cdot \bar{w} + \theta)) + \frac{\lambda}{2} \|\bar{w}\|^2 \quad (1)$$

where the loss function is defined by

$$h(z) = \begin{cases} 1 - z, & z \leq 1 \\ 0, & z > 1. \end{cases} \quad (2)$$

The cost-sensitive version modifies (1) to become

$$r_+ \cdot \sum_{i \in C_+} h(y_i(\bar{x}_i \cdot \bar{w} + \theta)) + r_- \cdot \sum_{i \in C_-} h(y_i(\bar{x}_i \cdot \bar{w} + \theta)) + \frac{\lambda}{2} \|\bar{w}\|^2 \quad (3)$$

and now we can choose r_+ and r_- to magnify the losses appropriately. Generally we take r_- to be 1, and r_+ to be some factor larger than 1. We refer to this formulation as CS-SVM. Generally, the same algorithms used to minimize (1) can be used to minimize (3).

Recommender systems use historical data on user preferences, purchases and other available data to predict items of interest to a user. Zhang and Iyengar (2002) propose a wide margin classifier with a quadratic loss function as very effective for this purpose (see appendix). It is used in (1) and requires no adjustment in cost between positive and negative examples. It is proposed as a better method than varying costs because it does not require searching for the optimal cost relationship between positive and negative examples. We will use for our wide margin classifier the modified Huber loss function (Zhang 2004). The modified Huber loss function is quadratic where this is important and has the form

$$h(z) = \begin{cases} -4 \cdot z, & z \leq -1 \\ (1 - z)^2, & -1 < z \leq 1 \\ 0, & z > 1. \end{cases} \quad (4)$$

We also use it in (1). We refer to this approach as the Huber method (Zhang 2004) as opposed to SVM. We compare it with SVM and CS-SVM. We used our own implementations for SVM, CS-SVM, and Huber that use gradient descent to optimize the objective function.

The methods we develop are related to semi-supervised learning approaches (Blum and Mitchell 1998; Nigam, McCallum et al. 1999) and active learning (Roy and McCallum 2001; Tong and Koller 2001). Our method differs from active learning in that active learning seeks those unlabeled examples for which labels prove most informative in improving the classifier. Typically these examples are the most uncertain. Some semi-supervised learning approaches start with labeled examples and iteratively seek unlabeled examples closest to already labeled data and impute the known label to the nearby unlabeled examples. Our goal is simply to retrieve plausible members for the positive class with as high a precision as possible. Our method has value even in cases where human review of retrieved examples is necessary. The imbalanced nature of the data and the presence of positives in the negative class make this a challenging problem.

In Section 2 we discuss additional strategies proposed in this work, describe the data used and design of experiments, and provide the evaluation measure used. In Section 3 we present our results, in Sections 4 and 5 we discuss our approach and draw conclusions.

2 Methods

2.1 Cross Training

Let D represent our set of documents, and C_+ those documents that are known positives in D . Generally C_+ would be a small fraction of D and for the purposes of learning we assume that $C_- = D \setminus C_+$.

We are interested in the case when some of the negatively labeled documents actually belong to the positive class. We will apply machine learning to learn the difference between the documents in the class C_+ and documents in the class C_- and use the weights obtained by training to score the documents in the negative class C_- . The highest scoring documents in set C_- are candidate mislabeled documents. However, there may be a problem with this approach, because the classifier is based on partially mislabeled data. Candidate

misclassified documents are part of the C_- class. In the process of training, the algorithm purposely learns to score them low. This effect can be magnified by any overtraining that takes place. It will also be promoted by a large number of features, which makes it more likely that any positive point in the negative class is in some aspect different from any member of C_+ .

Another way to set up the learning is by excluding documents from directly participating in the training used to score them. We first divide the negative set into disjoint pieces

$$C_- = Z_1 \cup Z_2$$

Then train documents in C_+ versus documents in Z_1 to rank documents in Z_2 and train documents in C_+ versus documents in Z_2 to rank documents in Z_1 . We refer to this method as cross training (CT). We will apply this approach and show that it confers benefit in ranking the false negatives in C_- .

2.2 Data Sources and Preparation

The databases we studied are *MeSH25*, *Reuters*, *20NewsGroups*, and *MedPhrase*.

MeSH25. We selected 25 MeSH[®] terms with occurrences covering a wide frequency range: from 1,000 to 100,000 articles. A detailed explanation of MeSH can be found at <http://www.nlm.nih.gov/mesh/>.

For a given MeSH term m , we treat the records assigned that MeSH term m as positive. The remaining MEDLINE[®] records do not have m assigned as a MeSH and are treated as negative. Any given MeSH term generally appears in a small minority of the approximately 20 million MEDLINE documents making the data highly imbalanced for all MeSH terms.

Reuters. The data set consists of 21,578 Reuters newswire articles in 135 overlapping topic categories. We experimented on the 23 most populated classes.

For each of these 23 classes, the articles in the class of interest are positive, and the rest of 21,578 articles are negative. The most populous positive class contains 3,987 records, and the least populous class contains 112 records.

20NewsGroups. The dataset is a collection of messages from twenty different newsgroups with about one thousand messages in each newsgroup. We used each newsgroup as the positive class and pooled the remaining nineteen newsgroups as the negative class.

Text in the *MeSH25* and *Reuters* databases has been preprocessed as follows: all alphabetic characters were lowercased, non-alphanumeric characters replaced by blanks, and no stemming was done. Features in the *MeSH25* dataset are all single nonstop terms and all pairs of adjacent nonstop terms that are not separated by punctuation. Features in the *Reuters* database are single nonstop terms only. Features in the *20NewsGroups* are extracted using the Rainbow toolbox (McCallum 1996).

MedPhrase. We process MEDLINE to extract all multiword UMLS[®] (<http://www.nlm.nih.gov/research/umls/>) phrases that are present in MEDLINE. From the resulting set of strings, we drop the strings that contain punctuation marks or stop words. The remaining strings are normalized (lowercased, redundant white space is removed) and duplicates are removed. We denote the resulting set of 315,679 phrases by $U_{phrases}$.

For each phrase in $U_{phrases}$, we randomly sample, as available, up to 5 MEDLINE sentences containing it. We denote the resulting set of 728,197 MEDLINE sentences by $S_{phrases}$. From $S_{phrases}$ we extract all contiguous multiword expressions that are not present in $U_{phrases}$. We call them n-grams, where $n > 1$. N-grams containing punctuation marks and stop words are removed and remaining n-grams are normalized and duplicates are dropped. The result is 8,765,444 n-grams that we refer to as M_{ngram} . We believe that M_{ngram} contains many high quality biological phrases. We use $U_{phrases}$, a known set of high quality biomedical phrases, as the positive class, and M_{ngram} as the negative class.

In order to apply machine learning we need to define features for each n-gram. Given an n-gram grm that is composed of n words, $grm = w_1 w_2 \cdots w_n$, we extract a set of 11 numbers

$\{f_i\}_{i=1}^{11}$ associated with the n-gram grm . These are as follows:

f_1 : number of occurrences of grm throughout Medline;

f_2 : -(number of occurrences of $w_2\dots w_n$ not following w_1 in documents that contain grm)/ f_1 ;

f_3 : -(number of occurrences of $w_1\dots w_{n-1}$ not preceding w_n in documents that contain grm)/ f_1 ;

f_4 : number of occurrences of (n+1)-grams of the form $xw_1\dots w_n$ throughout Medline;

f_5 : number of occurrences of (n+1)-grams of the form $w_1\dots w_n x$ throughout Medline;

$$f_6: \log \left(\frac{p(w_1 | w_2)(1 - p(w_1 | \neg w_2))}{(1 - p(w_1 | w_2))p(w_1 | \neg w_2)} \right)$$

f_7 : mutual information between w_1 and w_2 ;

$$f_8: \log \left(\frac{p(w_{n-1} | w_n)(1 - p(w_{n-1} | \neg w_n))}{(1 - p(w_{n-1} | w_n))p(w_{n-1} | \neg w_n)} \right)$$

f_9 : mutual information between w_{n-1} and w_n ;

f_{10} : -(number of different multiword expressions beginning with w_1 in Medline);

f_{11} : -(number of different multiword expressions ending with w_n in Medline).

We discretize the numeric values of the $\{f_i\}_{i=1}^{11}$ into categorical values.

In addition to these features, for every n-gram grm , we include the part of speech tags predicted by the MedPost tagger (Smith, Rindflesch et al. 2004). To obtain the tags for a given n-gram grm we randomly select a sentence from $S_{phrases}$ containing grm , tag the sentence, and consider the tags $t_0 t_1 t_2 \dots t_{n-1} t_n t_{n+1}$ where t_0 is the tag of the word preceding word w_1 in n-gram grm , t_1 is the tag of word w_1 in n-gram grm , and so on. We construct the features

$$\begin{cases} \text{if } n > 2: \{(t_0, 1), (t_1, 2), (t_n, 3), (t_{n+1}, 4), t_2, \dots, t_{n-1}\} \\ \text{otherwise: } \{(t_0, 1), (t_1, 2), (t_n, 3), (t_{n+1}, 4)\}. \end{cases}$$

These features emphasize the left and right ends of the n-gram and include parts-of-speech in the middle without marking their position. The resulting features are included with $\{f_i\}_{i=1}^{11}$ to represent the n-gram.

2.3 Experimental Design

A standard way to measure the success of a classifier is to evaluate its performance on a collection of documents that have been previously classified as positive or negative. This is usually accomplished by randomly dividing up the data into training and test portions which are separate. The classifier is then trained on the training portion, and is tested on test portion. This can be done in a cross-validation scheme or by randomly re-sampling train and test portions repeatedly.

We are interested in studying the case where only some of the positive documents are labeled. We simulate that situation by taking a portion of the positive data and including it in the negative training set. We refer to that subset of positive documents as *tracer data* (Tr). The tracer data is then effectively mislabeled as negative. By introducing such an artificial supplement to the negative training set we are not only certain that the negative set contains mislabeled positive examples, but we know exactly which ones they are. Our goal is to automatically identify these mislabeled documents in the negative set and knowing their true labels will allow us to measure how successful we are. Our measurements will be carried out on the negative class and for this purpose it is convenient to write the negative class as composed of true negatives and tracer data (false negatives)

$$C'_- = C_- \cup Tr.$$

When we have trained a classifier, we evaluate performance by ranking C'_- and measuring how well tracer data is moved to the top ranks. The challenge is that Tr appears in the negative class and will interact with the training in some way.

2.4 Evaluation

We evaluate performance using Mean Average Precision (MAP) (Baeza-Yates and Ribeiro-Neto 1999). The mean average precision is the mean value of the average precisions computed for all topics in each of the datasets in our study. *Average precision* is the average of the precisions at each rank that contains a true positive document.

3 Results

3.1 MeSH25, Reuters, and 20NewsGroups

We begin by presenting results for the MeSH25 dataset. Table 1 shows the comparison between Huber and SVM methods. It also compares the performance of the classifiers with different levels of tracer data in the negative set. We set aside 50% of C_+ to be used as tracer data and used the remaining 50% of C_+ as the positive set for training. We describe three experiments where we have different levels of tracer data in the negative set at training time. These sets are C_- , $C_- \cup Tr_{20}$, and $C_- \cup Tr_{50}$ representing no tracer data, 20% of C_+ as tracer data and 50% of C_+ as tracer data, respectively. The test set $C_- \cup Tr_{20}$ is the same for all of these experiments. Results indicate that on average Huber outperforms SVM on these highly

imbalanced datasets. We also observe that performance of both methods deteriorates with increasing levels of tracer data.

Table 2 shows the performance of Huber and SVM methods on negative training sets with tracer data $C_- \cup Tr_{20}$ and $C_- \cup Tr_{50}$ as in Table 1, but with cross training. As mentioned in the Methods section, we first divide each negative training set into two disjoint pieces Z_1 and Z_2 . We then train documents in the positive training set versus documents in Z_1 to score documents in Z_2 and train documents in the positive training set versus documents in Z_2 to score documents in Z_1 . We then merge Z_1 and Z_2 as scored sets and report measurements on the combined ranked set of documents. Comparing with Table 1, we see a significant improvement in the MAP when using cross training.

Table 1: MAP scores trained with three levels of tracer data introduced to the negative training set.

No Cross Training MeSH Terms	No Tracer Data		Tr ₂₀ in training		Tr ₅₀ in training	
	Huber	SVM	Huber	SVM	Huber	SVM
celiac disease	0.694	0.677	0.466	0.484	0.472	0.373
lactose intolerance	0.632	0.635	0.263	0.234	0.266	0.223
myasthenia gravis	0.779	0.752	0.632	0.602	0.562	0.502
carotid stenosis	0.466	0.419	0.270	0.245	0.262	0.186
diabetes mellitus	0.181	0.181	0.160	0.129	0.155	0.102
rats, wistar	0.241	0.201	0.217	0.168	0.217	0.081
myocardial infarction	0.617	0.575	0.580	0.537	0.567	0.487
blood platelets	0.509	0.498	0.453	0.427	0.425	0.342
serotonin	0.514	0.523	0.462	0.432	0.441	0.332
state medicine	0.158	0.164	0.146	0.134	0.150	0.092
urinary bladder	0.366	0.379	0.312	0.285	0.285	0.219
drosophila melanogaster	0.553	0.503	0.383	0.377	0.375	0.288
tryptophan	0.487	0.480	0.410	0.376	0.402	0.328
laparotomy	0.186	0.173	0.138	0.101	0.136	0.066
crowns	0.520	0.497	0.380	0.365	0.376	0.305
streptococcus mutans	0.795	0.738	0.306	0.362	0.218	0.306
infectious mononucleosis	0.622	0.614	0.489	0.476	0.487	0.376
blood banks	0.283	0.266	0.170	0.153	0.168	0.115
humeral fractures	0.526	0.495	0.315	0.307	0.289	0.193
tuberculosis, lymph node	0.385	0.397	0.270	0.239	0.214	0.159
mentors	0.416	0.420	0.268	0.215	0.257	0.137
tooth discoloration	0.499	0.499	0.248	0.215	0.199	0.151
pentazocine	0.710	0.716	0.351	0.264	0.380	0.272
hepatitis e	0.858	0.862	0.288	0.393	0.194	0.271
genes, p16	0.278	0.313	0.041	0.067	0.072	0.058
Avg	0.491	0.479	0.321	0.303	0.303	0.238

Table 2: MAP scores for Huber and SVM trained with two levels of tracer data introduced to the negative training set using cross training technique.

2-fold Cross Training MeSH Terms	Tr ₂₀ in training		Tr ₅₀ in training	
	Huber	SVM	Huber	SVM
celiac disease	0.550	0.552	0.534	0.521
lactose intolerance	0.415	0.426	0.382	0.393
myasthenia gravis	0.652	0.643	0.623	0.631
carotid stenosis	0.262	0.269	0.241	0.241
diabetes mellitus	0.148	0.147	0.144	0.122
rats, wistar	0.212	0.186	0.209	0.175
myocardial infarction	0.565	0.556	0.553	0.544
blood platelets	0.432	0.435	0.408	0.426
serotonin	0.435	0.447	0.417	0.437
state medicine	0.135	0.136	0.133	0.132
urinary bladder	0.295	0.305	0.278	0.280
drosophila melanogaster	0.426	0.411	0.383	0.404
tryptophan	0.405	0.399	0.390	0.391
laparotomy	0.141	0.128	0.136	0.126
crowns	0.375	0.376	0.355	0.353
streptococcus mutans	0.477	0.517	0.448	0.445
infectious mononucleosis	0.519	0.514	0.496	0.491
blood banks	0.174	0.169	0.168	0.157
humeral fractures	0.335	0.335	0.278	0.293
tuberculosis, lymph node	0.270	0.259	0.262	0.244
mentors	0.284	0.278	0.275	0.265
tooth discoloration	0.207	0.225	0.209	0.194
pentazocine	0.474	0.515	0.495	0.475
hepatitis e	0.474	0.499	0.482	0.478
genes, p16	0.102	0.101	0.083	0.093
Avg	0.350	0.353	0.335	0.332

We performed similar experiments with the *Reuters* and *20NewsGroups* datasets, where 20% and 50% of the good set is used as tracer data. We report MAP scores for these datasets in Tables 3 and 4.

3.2 Identifying high quality biomedical phrases in the MEDLINE Database

We illustrate our findings with a real example of detecting high quality biomedical phrases among M_{ngram} , a large collection of multiword expressions from Medline. We believe that M_{ngram} contains many high quality biomedical phrases. These examples are the counterpart of the mislabeled positive examples (tracer data) in the previous tests.

Table 3: MAP scores for Huber and SVM trained with 20% and 50% tracer data introduced to the negative training set for *Reuters* dataset.

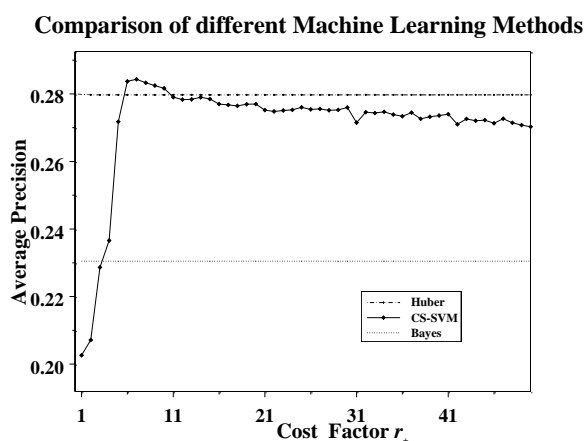
<i>Reuters</i>	Tr ₂₀ in training		Tr ₅₀ in training	
	Huber	SVM	Huber	SVM
No CT	0.478	0.451	0.429	0.403
2-Fold CT	0.662	0.654	0.565	0.555

Table 4: MAP scores for Huber and SVM trained with 20% and 50% tracer data introduced to the negative training set for *20NewsGroups* dataset.

<i>20NewsGroups</i>	Tr ₂₀ in training		Tr ₅₀ in training	
	Huber	SVM	Huber	SVM
No CT	0.492	0.436	0.405	0.350
2-Fold CT	0.588	0.595	0.502	0.512

To identify these examples, we learn the difference between the phrases in $U_{phrases}$ and M_{ngram} . Based on the training we rank the n-grams in M_{ngram} . We expect the n-grams that cannot be separated from UMLS phrases are high quality biomedical phrases. In our experiments, we perform 3-fold cross validation for training and testing. This insures we obtain any possible benefit from cross training. The results shown in figure 1 are MAP values for these 3 folds.

Figure 1. Huber, CS-SVM, and naïve Bayes classifiers applied to the *MedPhrase* dataset.



We trained naïve Bayes, Huber, and CS-SVM with a range of different cost factors. The results are presented in Figure 1. We observe that the Huber classifier performs better than naïve Bayes. CS-SVM with the cost factor of 1 (standard SVM) is quite ineffective. As we increase the cost factor, the performance of CS-SVM improves until it is comparable to Huber. We believe that the quality of ranking is better when the separation of $U_{phrases}$ from M_{ngram} is better.

Because we have no tracer data we have no direct way to evaluate the ranking of M_{ngram} . However, we selected a random set of 100 n-grams from M_{ngram} , which score as high as top-scoring 10% of phrases in $U_{phrases}$. Two reviewers manually reviewed that list and identified that 99 of these 100 n-grams were high quality biomedical phrases. Examples are: *aminoshikimate pathway*,

berberis aristata, *dna hybridization*, *subcellular distribution*, *acetylacetoin synthase*, etc. One false-positive example in that list was *congestive heart*.

4 Discussion

We observed that the Huber classifier performs better than SVM on imbalanced data with no cross training (see appendix). The improvement of Huber over SVM becomes more marked as the percentage of tracer data in the negative training set is increased. However, the results also show that cross training, using either SVM or Huber (which are essentially equivalent), is better than using Huber without cross training. This is demonstrated in our experiments using the tracer data. The results are consistent over the range of different data sets. We expect cross training to have benefit in actual applications.

Where does cost-sensitive learning fit into this picture? We tested cost-sensitive learning on all of our corpora using the tracer data. We observed small and inconsistent improvements (data not shown). The optimal cost factor varied markedly between cases in the same corpus. We could not conclude this was a useful approach and instead saw better results simply using Huber. This conclusion is consistent with (Zhang and Iyengar 2002) which recommend using a quadratic loss function. It is also consistent with results reported in (Lewis, Yang et al. 2004) where CS-SVM is compared with SVM on multiple imbalanced text classification problems and no benefit is seen using CS-SVM. Others have reported a benefit with CS-SVM (Abkani, Kwek et al. 2004; Eitrich and Lang 2005). However, their datasets involve relatively few features and we believe this is an important aspect where cost-sensitive learning has proven effective. We hypothesize that this is the case because with few features the positive data is more likely to be duplicated in the negative set. In our case, the *MedPhrase* dataset involves relatively few features (410) and indeed we see a dramatic improvement of CS-SVM over SVM.

One approach to dealing with imbalanced data is the artificial generation of positive examples as seen with the SMOTE algorithm (Chawla, Bowyer et al. 2002). We did not try this method and do not know if this approach would be beneficial for

textual data or data with many features. This is an area for possible future research.

Effective methods for leveraging positively labeled data have several potential applications:

- Given a set of documents discussing a particular gene, one may be interested in finding other documents that talk about the same gene but use an alternate form of the gene name.
- Given a set of documents that are indexed with a particular MeSH term, one may want to find new documents that are candidates for being indexed with the same MeSH term.
- Given a set of papers that describe a particular disease, one may be interested in other diseases that exhibit a similar set of symptoms.
- One may identify incorrectly tagged web pages.

These methods can address both removing incorrect labels and adding correct ones.

5 Conclusions

Given a large set of documents and a small set of positively labeled examples, we study how best to use this information in finding additional positive examples. We examine the SVM and Huber classifiers and conclude that the Huber classifier provides an advantage over the SVM classifier on such imbalanced data. We introduce a technique which we term cross training. When this technique is applied we find that the SVM and Huber classifiers are essentially equivalent and superior to applying either method without cross training. We confirm this on three different corpora. We also analyze an example where cost-sensitive learning is effective. We hypothesize that with datasets having few features, cost-sensitive learning can be beneficial and comparable to using the Huber classifier.

Appendix: Why Huber Loss Function works better for problems with Unbalanced Class Distributions.

The drawback of the standard SVM for the problem with an unbalanced class distribution results from the shape of $h(z)$ in (2). Consider the initial condition at $w = 0$ and also imagine that there is a lot more C_- training data than C_+ training data. In

this case, by choosing $\theta = -1$, we can achieve the minimum value of the loss function in (1) for the initial condition $w = 0$. Under these conditions, all C_- points yield $z = 1$ and $h(z) = 0$ and all C_+ points yield $z = -1$ and $h(z) = 2$. The change of the loss function $\Delta h(z)$ in (2) with a change Δw is given by

$$\Delta h(z) = \frac{dh(z)}{dz} \nabla_w z \cdot \Delta w = -y_i x_i \cdot \Delta w \quad (5).$$

In order to reduce the loss at a C_+ data point (x_i, y_i) , we must choose Δw such that $x_i \cdot \Delta w > 0$. But we assume that there are significantly more C_- class data points than C_+ and many such points x' are mislabeled and close to x_i such that $x' \cdot \Delta w > 0$. Then $h(z)$ is likely to be increased by $x' \cdot \Delta w (> 0)$ for these mislabeled points. Clearly, if there are significantly more C_- class data than those of C_+ class and the C_- set contains a lot of mislabeled points, it may be difficult to find Δw that can result in a net effect of decreasing the right hand side of (2). The above analysis shows why the standard support vector machine formulation in (2) is vulnerable to an unbalanced and noisy training data set. The problem is clearly caused by the fact that the SVM loss function $h(z)$ in (2) has a constant slope for $z \leq 1$. In order to alleviate this problem, Zhang and Iyengar (2002) proposed the loss function $h^2(z)$ which is a smooth non-increasing function with slope 0 at $z = 1$. This allows the loss to decrease while the positive points move a small distance away from the bulk of the negative points and take mislabeled points with them. The same argument applies to the Huber loss function defined in (4).

Acknowledgments

This research was supported by the Intramural Research Program of the NIH, National Library of Medicine.

References

- Abkani, R., S. Kwek, et al. (2004). Applying Support Vector Machines to Imbalanced Datasets. ECML.
- Baeza-Yates, R. and B. Ribeiro-Neto (1999). Modern Information Retrieval. New York, ACM Press.
- Blum, A. and T. Mitchell (1998). "Combining Labeled and Unlabeled Data with Co-Training." COLT: Proceedings of the Workshop on Computational Learning Theory: 92-100.
- Chawla, N. V., K. W. Bowyer, et al. (2002). "SMOTE: Synthetic Minority Over-sampling Technique." Journal of Artificial Intelligence Research **16**: 321-357.
- Eitrich, T. and B. Lang (2005). "Efficient optimization of support vector machine learning parameters for unbalanced datasets." Journal of Computational and Applied Mathematics **196**(2): 425-436.
- Elkan, C. (2001). The Foundations of Cost Sensitive Learning. Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence.
- Lewis, D. D., Y. Yang, et al. (2004). "RCV1: A New Benchmark Collection for Text Categorization Research." Journal of Machine Learning Research **5**: 361-397.
- Maloo, M. A. (2003). Learning when data sets are imbalanced and when costs are unequal and unknown. ICML 2003, Workshop on Imbalanced Data Sets.
- McCallum, A. K. (1996). "Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering." <http://www.cs.cmu.edu/~mccallum/bow/>.
- Nigam, K., A. K. McCallum, et al. (1999). "Text Classification from Labeled and Unlabeled Documents using EM." Machine Learning: 1-34.
- Roy, N. and A. McCallum (2001). Toward Optimal Active Learning through Sampling Estimation of Error Reduction. Eighteenth International Conference on Machine Learning.
- Smith, L., T. Rindfleisch, et al. (2004). "MedPost: A part of speech tagger for biomedical text." Bioinformatics **20**: 2320-2321.
- Tong, S. and D. Koller (2001). "Support vector machine active learning with applications to text classification." Journal of Machine Learning Research **2**: 45-66.
- Weiss, G., K. McCarthy, et al. (2007). Cost-Sensitive Learning vs. Sampling: Which is Best for Handling Unbalanced Classes with Unequal Error Costs? Proceedings of the 2007 International Conference on Data Mining.
- Zhang, T. (2004). Solving large scale linear prediction problems using stochastic gradient descent algorithms. Twenty-first International Conference on Machine Learning, Omnipress.
- Zhang, T. and V. S. Iyengar (2002). "Recommender Systems Using Linear Classifiers." Journal of Machine Learning Research **2**: 313-334.