Cool Blog Classification from Positive and Unlabeled Examples

Kritsada Sriphaew[⋆], Hiroya Takamura, and Manabu Okumura

Precision and Intelligence Laboratory
Tokyo Institute of Technology
4259 Nagatsuta Midori-ku Yokohama 226-8503 Japan
kong@lr.pi.titech.ac.jp, {takamura,oku}@pi.titech.ac.jp
http://www.lr.pi.titech.ac.jp

Abstract. We address the problem of cool blog classification using only positive and unlabeled examples. We propose an algorithm, called PUB, that exploits the information of unlabeled data together with the positive examples to predict whether the unseen blogs are cool or not. The algorithm uses the weighting technique to assign a weight to each unlabeled example which is assumed to be negative in the training set, and the bagging technique to obtain several weak classifiers, each of which is learned on a small training set generated by randomly sampling some positive examples and some unlabeled examples, which are assumed to be negative. Each of the weak classifiers must achieve admissible performance measure evaluated based on the whole labeled positive examples or has the best performance measure within iteration limit. The majority voting function on all weak classifiers is employed to predict the class of a test instance. The experimental results show that PUB can correctly predict the classes of unseen blogs where this situation cannot be handled by the traditional learning from positive and negative examples. The results also show that PUB outperforms other algorithms for learning from positive and unlabeled examples in the task of cool blog classification.

Keywords: Cool blog, PU-learning, weighting examples, bagging.

1 Introduction

For text mining, blog is an interesting textual resource because of its unique characteristic and public availability. It is a recent form of online content that has become popular among the web users. The contents of each blog are frequently updated by the postings of blog entries in reverse chronological order. Nowadays, a large number of blogs are published over the blogosphere but only some of them are interesting from the viewpoint of the readers.

The problem to identify a set of blogs that can interest an individual reader from a huge amount of blogs is not trivial. In this work, we consider the interestingness of blogs in an aspect of their contents but not their design or appearance,

[⋆] The author is supported by Japan Soceity for the Promotion of Science (JSPS).

© Springer-Verlag Berlin Heidelberg 2009

named coolness. Finding the cool blogs with interesting contents can be useful, but a more challenging task is how to overcome the real-life scenario when the labeled data is rare and only positive examples are preferred.

The criteria to judge the coolness of blogs are fairly subjective based on the readers' intuition. For example, the readers whose interests are gourmet may prefer those blogs that recommend good restaurants more than ones that relate to the modern fashions. Some of the readers may have interest in both kinds of blogs. We can simply represent the cool blog classification as a basic binary classification where we suppose that the reader is interested in one specific group of blogs and seeks to distinguish between the interesting instances (i.e., positive examples) and the other instances (i.e., negative examples). In the real world case, positive examples can be collected from the blogs that have been visited many times by the user, but this is not applicable for the negative examples. Therefore, it often happens that we lack the labeled negative examples although there are a large number of unlabeled examples which we can employ to learn the classifier. This scenario is called the learning from positive and unlabeled examples (*PU-learning*). Moreover, the labeled positive examples may not cover all target concepts that the reader is interested in. This effects the difficulty of predicting unseen instances if only a small set of labeled examples is available. Especially to the labeled cool blogs, it is highly possible that we have less number of positive examples. Those labeled examples are only a portion of positive concepts that can interest the reader. There are other target and non-target concepts available in the unlabeled examples which can be used to learn the classification model.

To overcome these problems, we propose a new algorithm for PU-learning that employs two techniques for learning. First, the weighting technique is used for assigning a weight to each unlabeled example and all of unlabeled examples are assumed to be negative in the training set. With the training set that contains positive and assumed negative examples, the bagging technique is applied to obtain a smaller set of training used for learning the classifier by random sampling with replacement from the training set. After several iterations, a number of classifiers will be generated. Each of them must achieve admissible performance measure evaluated based on the whole labeled positive examples or has the best performance measure within iteration limit, but it does not overfit to the training positive examples. This enables the generated classifiers to predict the classes of unseen instances. By aggregating these classifiers, the variance error from various training sets generated by the samplings can be lessen.

The contributions of this work are as follows. We are the first to address the problem of cool blog classification in the scenario of PU-learning that is prevalent in the real world application. We propose an algorithm for PU-learning which needs only positive and unlabeled examples as the inputs without separated validation sets. Although it is difficult to predict the class of unseen instances when only a small number of labeled examples is available, the proposed algorithm can perform well in such case by employing the bagging technique to randomly sample the data for training.

2 Related Work

We divide this section into two parts. The first part discusses the problem of cool blog classification. The second part reviews existing approaches of PU-learning.

The problem of cool blog classification was first introduced in [1]. They introduced some assumptions on the topics of blogs to indicate the coolness. A set of features corresponding to their assumptions was proposed for encoding both positive and negative examples and they are learned by traditional SVM. However, their scenario is absolutely different from our work since we deal with a more practical scenario where only positive and unlabeled examples are available for learning. Another related research on blog data is the credibility assessment of blogs [2,3]. They proposed a set of factors which can be used to indicate the credibility of blogs. The objective of credibility assessment is totally different from cool blog classification in that they want to retrieve a set of highly credible blog entries with regard to the specified query while our objective is to identify cool blogs as a whole without prerequisite information. Although there are some studies on the interestingness of web contents [4,5] in the information retrieval area, they are not equivalent to this problem because their targets are to find interesting web pages based on the information of reader's profile.

For the problem of PU-learning, several algorithms have been published after the first work in [6]. As stated in [7], the algorithms for PU-learning can be divided into three families. The first family of algorithms is a two-step strategy where the heuristics to identify unlabeled examples that are likely to be negative are applied in the first step, and the standard learning method learns the identified examples and positive examples in the second step [8,9,10,11].

The second family of algorithms focuses on estimating statistical queries over positive and unlabeled examples in order to construct the model of negative examples [12]. This idea was also extended to the Co-Training scenario for which the data has two or more redundant views that are compatible but conditionally independent [13]. The shortcoming of this family of algorithms is the prerequisite knowledge of the prior probability of the positive examples which are usually not available in practice.

The third family of algorithms concerns with assigning weights to the examples, assuming unlabeled examples as negative and applying standard learning method. Lee and Liu [14] applied the logistic regression to learn the linear function from the weighted examples. The positive examples are assigned a specific weight while the unlabeled examples, that are assumed to be negative, are assigned another weight. Those two weights are derived from a validation set by optimizing the model to handle noise rates of greater than a half. Liu et al. [9] presented the biased SVM that allows noise for both positive and negative examples in the SVM formulation by giving two different weights for positive and negative errors, respectively, via regularization parameters. Zhang and Lee [15] gives a specific weight to the positive examples in the classification function. The classification function is a simple sign function of the difference between the weighted probability of being positive and the probability of being unlabeled for a given example, where a weight is estimated from a validation set and the probabilities are estimated based on tf-idf of the words in the training set.

The most recent successive approach was presented by Elkan and Noto in [16]. When the assumption that positive training examples are selected completely at random holds, they proved that a classifier trained on the positive and unlabeled examples predict probabilities that differ by only a constant factor from the true conditional probabilities of being positive. This assumption is coincident with a general situation when we prepare the data for cool blog classification. However, there are two major shortcomings of their approach. First, the high bias of the generated classifier that is trained from the whole positive and unlabeled examples may result to the overfit on the training set. Second, their method to duplicate the same unlabeled examples as both positive and negative will effect badly to the SVM.

Most of the existing algorithms for PU-learning have been evaluated by the modified versions of standard data sets, e.g., Reuters-21578 and 20Newsgroup. These modified datasets are generated by regarding some arbitrarily-selected examples as unlabeled. In contrast, our dataset for cool blog classification is not artificial in the sense that the unlabeled examples are not generated from the labeled ones. It may be the case that users would try to avoid annotating examples which are hard to make the decision on. In this work, we address the problem of PU-learning in this practical way using cool blog data and compare our approach with some existing PU-learning algorithms for performance study.

3 PUB: An Algorithm for PU-Learning Using Weighting and Bagging Techniques

In this work, we propose an algorithm that employs weighting and bagging technique for PU-learning, called PUB. The algorithm can be divided into two steps. In the first step, the weighting technique is used to assign individual weight to each unlabeled example. We follow the approach in [16]. In the second step, the bagging technique is applied to obtain several weak classifiers, each of which is learned on a small training set generated by randomly sampling some positive examples and some unlabeled examples, which are assumed to be negative. Each of the weak classifiers must achieve admissible performance measure evaluated based on the whole labeled positive examples or has the best performance measure within iteration limit. The majority voting function on all weak classifiers is employed to predict the class of a test instance.

3.1 Weighting Technique

Before giving an explanation about weighting technique, we first formally define the problem of cool blog classification using positive and unlabeled examples.

Let D denote a training set, $D = \{(x_n, y_n, s_n)\}$ where $n = 1, ..., N$. The x's are examples in the training set and y's are the class labels where $y_n = 0$ if x_n is a negative example and $y_n = 1$ if x_n is a positive example. At the beginning, y_n

is left unknown if x_n is an unlabeled example. The s's are other binary attributes of examples, where $s_n = 1$ if x_n is labeled and $s_n = 0$ if x_n is unlabeled. x, y and s can be viewed as random variables where there is some unknown probability distribution $p(x, y, s)$ over triples $< x, y, s >$. In our case, only positive examples are labeled. Therefore, no negative example is labeled, i.e., $p(s = 1|x, y = 0) = 0$.

Our goal is to learn a function $f(x)$ so that $f(x)$ becomes close to $p(y = 1|x)$ as much as possible, but we need an assumption about which positive examples are labeled. By following the same assumption as in [16], the labeled positive examples are chosen completely at random from all positive examples. Therefore, this assumption can be formally stated as $p(s = 1|x, y = 1) = p(s = 1|y = 1)$.

Suppose we want to learn a function $g(x)$ to classify whether a given document is labeled or not, such that $g(x) = p(s = 1|x)$. In [16], it is stated that we can obtain a goal function $f(x)$ from $g(x)$ by using the following lemma.

Lemma 1. *Suppose the "selected completely at random" assumption holds, then* $p(y = 1|x) = p(s = 1|x)/c$ *where* $c = p(s = 1|y = 1)$.

In other words, c is the constant probability that a positive example is labeled. In their work, c is estimated as $\frac{1}{n}\sum_{x \in P} g(x)$ where P is a set of labeled positive examples and n is the cardinality of P. This means we can calculate c from the probabilities of positive examples produced as output of the classifier $g(x)$. The proof can be found in [16].

In this scenario, it is obvious that $p(y = 1|x, s = 1) = 1$ for any labeled positive example, and $p(y = 1|x, s = 0)$ can be obtained from

$$p(y = 1|x, s = 0) = \frac{(1 - c)}{c} \frac{p(s = 1|x)}{1 - p(s = 1|x)}. \tag{1}$$

Here, $p(y = 1|x, s = 0)$ is the probability that an unlabeled example is positive, and we denote $w(x) = p(y = 1|x, s = 0)$. In other words, $1 - w(x)$ is the probability that an unlabeled example is negative. The original approach in [16] employs these probabilities by assigning different weights to examples for constructing a new training set. Each positive example is assigned with unit weight $p(y = 1|x, s = 1)$, while the unlabeled example x is duplicated; one copy is assumed to be positive with weight $w(x)$ and the other copy is assumed to be negative with weight $1 - w(x)$. This is the main shortcoming of the original approach since the duplication of examples with different labels effects to the failure of the generated classifier.

In this work, we propose to assign unit weight to the positive examples, and assume all unlabeled examples to be negative with their weights being $1 - w(x)$. These weighted examples are later used as a training set for the second step.

3.2 Bagging Technique

The original idea of bagging for classification is to generate m weak classifiers trained from different training sets that are sampled from the whole training set uniformly and with replacement [17]. Although it is possible to directly apply the original idea for PU-learning, there are three differences between the scenario of the original bagging technique and PU-learning. First, all examples in the training set must be labeled in the original scenario of bagging approach. Second, the training set must have both positive and negative examples. Third, there is no guarantee that each weak classifier trained from the sampled training set will achieve high performance on the target concepts, especially for the case that only positive examples are labeled.

We can simply solve the first and second problems by assuming all unlabeled examples as either positive or negative. However, we still need to solve the third problem simultaneously since our assumed classes of some unlabeled examples may be incorrect and effect to the poor performance of the generated classifiers. A modified version of bagging, BaggTaming [18], can be applied to solve the third problem by setting a condition that each classifier should achieve admissible performance measure when it is evaluated based on the target concepts. However, such work has different scenario to our problem since the nature of data used in their approach is different from PU-learning and we do not have labeled negative examples. Moreover, the third problem still remains if the original algorithm is applied because not all generated classifiers achieve admissible performance measure. Therefore, we proposed a new bagging technique to solve those three problems as shown by the pseudo code in Figure 1.

```
 1: Construct $D'$ from $D$ by assigning $y_n = 0$ if $x_n$ is an unlabeled example
 2: $\mathcal{W} = \emptyset$;
 3: while $|\mathcal{W}| < m$ do
 4:    $i = 1$
 5:    $maxperf = 0$
 6:    while $i \leq$ LIMIT do
 7:       Generate $D_i$ by random sampling with replacement from $D'$ and balancing
          the number of examples in each class
 8:       Learn $f_i(x) = p(y = 1|x)$ from $x \in D_i$
 9:       if $eval_{x' \in P}(f_i(x')) > maxperf$ then
10:          $f_{BEST} \leftarrow f_i(x)$
11:          $maxperf = eval_{x' \in P}(f_i(x'))$
12:       end if
13:       if $eval_{x' \in P}(f_i(x')) > \alpha$ then
14:          goto line 18
15:       end if
16:       $i \leftarrow i + 1$
17:    end while
18:    $\mathcal{W} = \mathcal{W} \cup f_{BEST}$
19: end while
20: $\hat{f}(x) = arg \max_{y \in \{-1,1\}} \sum_{f_j(x) \in \mathcal{W}} eval_{x' \in P}(f_j(x')) \times I(y == f_j(x))$
```

Fig. 1. Pseudo code of Bagging Technique in *PUB*

With the given training set D that contains positive and unlabeled examples, D' is constructed from D, where the positive examples are unchanged but the unlabeled examples are assumed to be negative. The algorithm has two repetition loops, where the outer loop (line 3) concerns with the construction of a set of weak classifiers W, while the inner loop (line 6) generates the candidate classifiers $f_i(x)$. Any candidate classifier that has the best performance (*maxperf*) in the inner loop (line 9-12) or achieves admissible performance measure α (line 13-15) is included to a set of weak classifiers $\mathcal{W}$ (line 18). In line 7, to generate a candidate classifier, we construct a new training set from D' by random sampling with replacement and balancing the number of examples in both classes. This is one difference of our algorithm to the original bagging approach. We propose to balance the number of examples since the number of negative examples (assumed from the unlabeled examples in line 1) is usually much larger than that of positive examples. This simple proposed solution can prevent skewed decision boundaries problem occurred in an unbalanced classification [19]. Furthermore, the approach of generating a new training set by random sampling with replacement can help solve the problem that the number of positive examples is small. This issue was rarely addressed by other PU-learning algorithms.

A function $eval_{x \in P}(f_i(x))$ (line 9) is the performance measure function of the classifier $f_i(x)$ evaluated based on $x \in P$. Since the scenario of PU-learning has only positive examples as target concept, we cannot use F-measure, which is commonly used in the classification. We turn to a similar performance measure proposed in [14], which can be calculated only from the positive examples. Given $precision = p(y = 1|f(x) = 1)$ and $recall = p(f(x) = 1|y = 1)$, the performance measure is calculated as

$$\frac{precision \times recall}{p(y = 1)} = \frac{recall^2}{p(f(x) = 1)}. \tag{2}$$

This measure behaves similarly to the F-measure in that it is large only when both *precision* and *recall* are large. Therefore, we use only recall and $p(f(x) = 1)$, which can be calculated from only positive examples as a performance measure.

The algorithm iteratively runs until we get m weak classifiers as conditioned in line 3. In the last step, majority voting function $\hat{f}(x)$ on all m weak classifiers is used to predict the class of a test instance, but each vote is weighted by the performance measure of each weak classifier as shown in line 20. The identity function $I[cond]$ produces 1 if the *cond* is true, otherwise 0. Note that a classifier $f_i(x)$ in this case will output 1 or -1.

Our algorithm employs the weighting technique in the first step to construct the training examples with their corresponding weights and applies the bagging technique to learn the classification function.

4 Experiments

To study the performance of the proposed algorithm, we conduct several experiments on the real data for cool blog classification in various settings. We also compare the performance of our algorithm with some other best performing algorithms for PU-learning.

4.1 Benchmark Corpus

We use the same corpus as presented in [1] for a task of cool blog classification. The data was provided by the blogWatcher Inc., a Japanese blog search company, for the purpose of research. There are 838,341 blog entries on distinct 60,736 blogs. Their issued dates are varied from Feb 2005 to Dec 2007. Manual labeling of cool blogs was done by an engineer in that company, who is a native Japanese speaker, without any given assumptions. The annotator selected the blogs for labeling in random, and judged the coolness of the whole blog instead of each individual blog entry. Finally, the corpus contains 270 positive blogs (with 9,827 blog entries), 270 negative blogs (with 3,836 blog entries) and the rest are left as unlabeled examples. We did not interview the annotator of his notion to label the cool blogs. This can help avoid the bias of our method to identify cool blogs. We encode each blog consisting of multiple blog entries as a simple bag-of-words vector.

Although our data is a collection of Japanese blogs, we did not use any language-specific techniques in the task of classification. In other words, the contribution of this work can be applied for identifying cool blogs in any languages if the concept of cool blog is similar to this work. Since the main objective of this work is to solve the problem of cool blog classification rather than the problem of PU-learning, the performance study of PUB against other PU-learning algorithms on standard corpus (e.g., Reuters-21578, 20Newsgroup) is out of scope.

4.2 Implementations and Parameter Settings

Our proposed algorithm is flexible for any classification method. In this work, we use SVM where we selected one implementation presented by M.W. Chang and H.T. Lin[1] that allows different weights (via regularization parameter C) for different examples. However, SVM is not a learning method that directly produces the correct probabilities. The output of SVM can be post-processed into calibrated probabilities by applying Platt scaling [20]. We apply Platt scaling only in the first step of an algorithm for weighting the examples. Through all experiments, the linear kernel function, which is commonly used for text classification, was employed with the optimized C.

Four parameters are needed to be defined in *PUB* algorithm, i.e., the number of weak classifiers (m), the admissible performance measure (α) of a weak classifier, the iteration limit of generating candidate classifiers (LIMIT) and the size of sampled training set ($|D_i|$). With the larger m, it tends to be that the more reliable result from the majority voting function will be obtained. However, the larger m also causes an algorithm to take more computational time. We can simply set m to be a specific number that has acceptable trade-off between

[1] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/#15

performance and computational time. In this work, we set the default m to be 30. For the α, we propose to get this threshold by learning a classifier from the whole positive examples and assuming all unlabeled examples to be negative. In our experiment, we found out that this value is close to the optimized value. For our case, the default α is 0.6. The parameter LIMIT performs in the same way as parameter m. If we set smaller LIMIT, most of the weak classifiers may not achieve α performance. In the preliminary experiment, we found that the number of iterations in the inner loop is less than ten if we set the value of α not too high. We set the default value of LIMIT to be 100. Since it is not straightforward to find the optimized value of $|D_i|$, we then investigate it by the experiment. By default, we set $|D_i|$ to be 50% of the number of positive examples.

4.3 Experimental Results

All experiments were conducted in three-fold cross validation on the labeled examples where two folds of positive examples and the whole unlabeled examples are used for training. The remaining positive examples with one fold of negative examples are used for testing. Note that the other two folds of negative examples are used for training only in the case of PN-learning (described below). We show the performance by averaging the results from those three validations.

Comparison with PN-learning: We conducted this experiment to compare the performance of PUB with the learning from positive and negative examples (PN-learning). The results are shown in Table 1. We used the default setting as shown in Section 4.2 for PUB. There are two alternative ways to exploit PN-learning. The first way is to provide actual negative examples for learning denoted by "PN-learning1". This is possible for this corpus since the annotator also labeled the negative examples but it is not practical in real world application. The second way is to assume all unlabeled examples to be negative denoted by"PN-learning2". The performance of PUB is surprisingly better than both cases of PN-learning. This is possible since the given negative examples for PN-learning may not be the good representatives for negative class, and this circumstance is also the same for the case of positive examples. Therefore, only one discriminative model may not be able to predict the classes of all instances that have the target concepts. We can say that PUB is a less-biased model in the sense that it combines several outputs from the discriminative models that are guaranteed to achieve admissible performance measure on the target concepts.

Table 1. Performance of PUB and PN-learnings

Method	Accuracy	Precision	Recall	F-measure
PN-learning1	0.787	0.819	0.741	0.778
PN-learning2	0.748	0.935	0.533	0.679
PUB	0.828	0.842	0.807	0.824

Table 2. Performance of PUB and existing PU-learning algorithms

Method	Accuracy	Precision	Recall	F-measure
PEBL	0.723	0.696	0.795	0.742
Biased SVM	0.748	0.935	0.533	0.679
W-LR	0.758	0.832	0.659	0.735
WUE	0.737	0.887	0.552	0.680
PUB	0.828	0.842	0.807	0.824

Table 3. Variation of Parameters for PUB

Parameter	Value	Accuracy	Precision	Recall	F-measure
m	5	0.813	0.850	0.759	0.802
	10	0.813	0.833	0.781	0.807
	30	0.828	0.842	0.807	0.824
	50	0.830	0.850	0.800	0.824
α	0.5	0.822	0.863	0.767	0.812
	0.6	0.828	0.842	0.807	0.824
	0.7	0.802	0.793	0.819	0.806
	0.8	0.776	0.762	0.837	0.798
$\vert D_i\vert$	45	0.815	0.813	0.819	0.816
	90	0.828	0.842	0.807	0.824
	180	0.798	0.862	0.711	0.779
	360	0.794	0.867	0.696	0.772
size of positive examples for training	20	0.769	0.727	0.859	0.788
	45	0.811	0.810	0.815	0.812
	90	0.817	0.830	0.796	0.813
	180	0.828	0.842	0.807	0.824

Comparison with existing PU-learning algorithms. We selected four best performing algorithms (mentioned in their papers) from the families of algorithms for PU-learning to study the relative performance of PUB against the existing algorithms. PEBL [11], Biased SVM [9], Weighted Logistic Regression (W-LR, for short) [14] and weighting unlabeled examples (WUE, for short) [16] are selected to investigate on this corpus. Since there is no source code available for such algorithms except W-LR[2], we carefully implement them following their papers. The optimized value of parameters in each algorithm is manually tuned and the best result of each algorithm is shown in Table 2. The results show that PUB is superior to the other PU-learning algorithms, and those previous PU-learning algorithms cannot achieve higher performance than the PN-learning.

Variations of PUB. We also varied the setting of parameters to show the best optimized values of our algorithm. The results are shown in Table 3. Here, we can achieve higher performance when we increase the value of m. However, with higher m, more computational time is taken while the performance becomes saturated

[2] http://www.comp.nus.edu.sg/~leews/publications/logistic.tar.gz

Table 4. Variation of method in PUB algorithm

Method	Accuracy	Precision	Recall	F-measure
original PUB	0.828	0.842	0.807	0.824
PUB without weighting technique	0.822	0.840	0.796	0.817

because of the identity function used for aggregating the classifiers. The optimized value of parameter α in this case is 0.6 since it provides the classifiers that can predict the target concept but not overfit to the training positive examples. However, this optimized value is close to the performance measure of the model that is trained from all positive examples and unlabeled examples assumed as negative, i.e., 0.53. Therefore, we can tune this parameter in this way. For D_i, our default setting for 50% of training positive examples performs the best. We also study the situation that a training set has an extremely small number of labeled examples. Although the performance of PUB algorithm becomes lower when the smaller number of positive examples is available, that performance is still better than the PN-learning approaches and other PU-learning algorithms.

We conducted the last experiment to study the significance of two techniques used in PUB algorithm by removing the weighting technique applied in the first step. The results in Table 4 show that the performance of this modified PUB becomes slightly lower than the original PUB. It means the proposed bagging technique plays an important role to improve the performance of PU-learning more than the weighting technique.

5 Conclusions and Future Work

This work addressed the problem of cool blog classification in the scenario of PU-learning which is prevalent to the real world application. We proposed an algorithm for PU-learning using weighting and bagging techniques. By investigation in several experimental settings, the proposed algorithm surprisingly showed that it can predict unseen cool blogs. This situation cannot be handled by the traditional PN-learning. The results show that the proposed algorithm is superior to other existing algorithms for PU-learning in the task of cool blog classification, and it can also deal with the case that has an extremely small number of labeled examples. For future work, we plan to extend our algorithm on other real world applications such as learning to annotate the NLP corpus using only a small set of labeled examples.

References

1. Sriphaew, K., Takamura, H., Okumura, M.: Cool blog identification using topic-based models. In: Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence, Sydney, Australia. IEEE, Los Alamitos (2008)
2. Rubin, V.L., Liddy, E.D.: Assessing credibility of weblogs. In: Proceedings of the AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs (CAAW) (2006)

3. Weerkamp, W., de Rijke, M.: Credibility improves topical blog post retrieval. In: Proceedings of ACL 2008: HLT, Columbus, Ohio, Association for Computational Linguistics, pp. 923–931 (2008)
4. Billsus, D., Pazzani, M.J.: Learning and revising user profiles: The identification of interesting web sites. Machine Learning 27, 313–331 (1997)
5. Pazzani, M.J., Muramatsu, J., Billsus, D.: Syskill & webert: Identifying interesting web sites. In: 13th National Conference on Artificial Intelligence, Portland, OR, US, vol. 1, pp. 54–61 (1996)
6. Denis, F.: Pac learning from positive statistical queries. In: Richter, M.M., Smith, C.H., Wiehagen, R., Zeugmann, T. (eds.) ALT 1998. LNCS, vol. 1501, pp. 112–126. Springer, Heidelberg (1998)
7. Zhang, B., Zuo, W.: Learning from positive and unlabeled examples: A survey. In: Proceedings of the International Symposiums on Information Processing, pp. 650–654 (2008)
8. Liu, B., Lee, W.S., Yu, P.S., Li, X.: Partially supervised classification of text documents. In: ICML, pp. 387–394 (2002)
9. Liu, B., Dai, Y., Li, X., Lee, W.S., Yu, P.S.: Building text classifiers using positive and unlabeled examples. In: ICDM 2003: Proceedings of the Third IEEE International Conference on Data Mining, p. 179. IEEE Computer Society Press, Washington (2003)
10. Li, X., Liu, B.: Learning to classify texts using positive and unlabeled data. In: IJCAI 2003: Proceedings of Eighteenth International Joint Conference on Artifical Intelligence, pp. 587–594 (2003)
11. Yu, H., Han, J., Chang, K.C.: Pebl: Web page classification without negative examples. IEEE Trans. on Knowl. and Data Eng. 16(1), 70–81 (2004)
12. Denis, F., Gilleron, R., Tommasi, M.: Text classification from positive and unlabeled examples. In: IPMU 2002, 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, pp. 1927–1934 (2002)
13. Denis, F., Gilleron, R., Laurent, A., Tommasi, M.: Text classification and co-training from positive and unlabeled examples. In: Proceedings of the ICML Workshop: the Continuum from Labeled Data to Unlabeled Data in Machine Learning and Data Mining, pp. 80–87 (2003)
14. Lee, W.S., Liu, B.: Learning with positive and unlabeled examples using weighted logistic regression. In: ICML 2003: Proceedings of the Twentieth International Conference on Machine Learning (2003)
15. Zhang, D., Lee, W.S.: A simple probabilistic approach to learning from positive and unlabeled examples. In: Proceedings of the 5th Annual UK Workshop on Computational Intelligence (2005)
16. Elkan, C., Noto, K.: Learning classifiers from only positive and unlabeled data. In: SIGKDD (2008)
17. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
18. Kamishima, T., Hamasaki, M., Akaho, S.: Baggtaming – learning from wild and tame data. In: Proceedings of the Workshop on Web 2.0 Mining at ECML PKDD 2008 (2008)
19. Morik, K., Brockhausen, P., Joachims, T.: Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In: ICML, pp. 268–277 (1999)
20. Platt, J.C.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: Smola, A., Bartlett, P., Scholkopf, B., Schuurmans, D. (eds.) Advances in Large Margin Classifiers, pp. 61–74 (1999)