Learning Classifiers without Negative Examples: A Reduction Approach

Dell Zhang
SCSIS
Birkbeck, University of London
London WC1E 7HX, UK
dell.z@ieee.org

Wee Sun Lee
Department of Computer Science
National University of Singapore
Singapore 117590
leews@comp.nus.edu.sg

Abstract

The problem of PU Learning, i.e., learning classifiers with positive and unlabelled examples (but not negative examples), is very important in information retrieval and data mining. We address this problem through a novel approach: reducing it to the problem of learning classifiers for some meaningful multivariate performance measures. In particular, we show how a powerful machine learning algorithm, Support Vector Machine, can be adapted to solve this problem. The effectiveness and efficiency of the proposed approach have been confirmed by our experiments on three real-world datasets.

1 Introduction

Standard machine learning techniques for building a binary classifier require a set of positive examples $\mathbf{x}$ with label $+1$ and a set of negative examples $\mathbf{x}$ with label -1. However, in practice it is often very difficult to get labelled negative examples. This is because people usually only keep the data that are of interest to them (i.e., positive examples), and it is unnatural to require people to label uninteresting data (i.e. negative examples). In such situations, what available in addition to positive examples is just a set of unlabelled examples. Can we still train classifiers effectively and efficiently without any negative examples? This problem is called *PU Learning* [9].

Definition 1. PU Learning. *Given an incomplete set of positive examples $P = \{\mathbf{x}_1, \ldots, \mathbf{x}_l\}$ that we are interested in, and a set of unlabelled examples $U = \{\mathbf{x}_{l+1}, \ldots, \mathbf{x}_{l+u}\}$ which contains both positive examples and negative examples, we would like to use P and U to train a classifier that can accurately classify positive and negative examples in U or in a separate test set.*

The problem of PU Learning occurs frequently in information retrieval and data mining applications [9]. For example, a researcher may have saved in her computer some journal articles on a specialised subtopic in bioinformatics (P), and she wants to find more materials on that subtopic from the PubMed Central digital library (U). For another example, a user searched the Web using a search engine and clicked on some returned links that he was interested in, then the search engine could improve its ranking of the search results by building a classifier based on the clicked links (P) and the other links (U).

We address this problem through a novel approach: reducing it to the problem of learning classifiers for some meaningful *multivariate performance measures*.

One of the most powerful machine learning techniques for classification is Support Vector Machine (SVM) [13] which has solid theoretical basis and broad practical success. For example, SVM in its simplest form, linear SVM, consistently provides state-of-the-art performance for text categorization tasks [16]. In this paper, we focus on adapting SVM algorithms for PU Learning, though our reduction approach to PU Learning is general.

The rest of this paper is organised as follows. We first propose our reduction approach to PU Learning (in Section 2), then present experimental evaluation (in Section 3), later discuss related work (in Section 4), and finally make conclusions (in Section 5).

2 Approach

2.1 Learning from P and N

In the ideal situation, we have in addition to P a set of negative examples N rather than a set of unlabelled examples ($\forall \mathbf{x}_i \in P : y_i = +1$ and $\forall \mathbf{x}_i \in N : y_i = -1$), so we can use both P and N to train a standard SVM classifier.

OP 1. SVM_{2C}

$$\min_{\mathbf{w}, \xi_i \geq 0} \qquad \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{n}\sum_{i=1}^{n} \xi_i$$

$$s.t. \quad \forall_{i=1}^{n}: y_i \mathbf{w}^T \mathbf{x}_i \geq 1 - \xi_i$$

SVM_{2C} attempts to find a hyperplane $h(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ that can separate positive examples and negative examples with a *large margin* $\frac{2}{||\mathbf{w}||}$ as well as small empirical *hinge loss* $\sum_{i=1}^{n} \xi_i$.

However, as explained earlier, a negative set N is often not available therefore SVM_{2C} is not applicable.

2.2 Learning from P Only

When we do not have negative examples, one possibility is to ignore U and use P only to train the so-called 'one-class' SVM classifier [13].

OP 2. SVM_{1c}

$$\begin{aligned} \min_{\mathbf{w},\xi_i \geq 0} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{l}\sum_{i=1}^{l}\xi_i \\ s.t. \quad & \forall_{i=1}^{l}: \mathbf{w}^T\mathbf{x}_i \geq 1 - \xi_i \end{aligned}$$

Note that our SVM_{1c} formulation here is different with the original one-class SVM [13] which relies on a parameter ν (as in ν-SVM [13]) but not C. The parameter ν has more intuitive sense than C: it directly controls the amount of margin errors and the number of support vectors. Nevertheless, the classification function of the original one-class SVM optimised for a given ν would be same as that of SVM_{1c} with a certain value of C. Current training methods for ν-SVMs (including the original one-class SVM) take at least quadratic time, whereas with our SVM_{1c} formulation we can achieve linear-time training [20].

Intuitively, SVM_{1c} is inferior for PU Learning because it ignores useful information that is present in the set of unlabelled examples U.

2.3 Learning from P and U

If we take the positive examples in U as noise, then we can consider U as a very *noisy* set of negative training examples. There are totally $n = l + u$ examples. Denote the *observed* label of an example $\mathbf{x}$ by $y \in \{-1, +1\}$, i.e., $\forall \mathbf{x}_i \in P: y_i = 1$ and $\forall \mathbf{x}_i \in U: y_i = -1$. Denote the *actual* label of an example $\mathbf{x}$ by $z \in \{-1, +1\}$ indicating its true relevancy. We know that $\forall \mathbf{x}_i \in P: z_i = y_i = 1$, but we have no idea about the *hidden* value of z_i for any $\mathbf{x}_i \in U$.

Assume that the examples in P are *randomly* sampled from the class of positive examples with a certain probability μ. The value of $\mu = \Pr[y = 1|z = 1]$ is an unknown constant. In other words, an actual positive example has probability μ to be observed in P and probability $1 - \mu$ to be left in U; while all actual irrelevant documents are put in U. We have the following relationships:

$$\begin{aligned} \Pr[y = +1] &= \Pr[z = +1]\mu; \\ \Pr[y = -1] &= \Pr[z = -1] + \Pr[z = +1](1 - \mu). \end{aligned}$$

Using P and U straightforwardly to train the standard SVM_{2C} would not work. Let's call the classification performance calculated over observed (noisy) labels y_i *observed* performance, and the classification performance calculated over actual labels z_i *actual* performance. SVM_{2C} minimises the observed error rate, but low observed error rate does not necessarily lead to low actual error rate [1]. For example, when there are 100 documents in U relevant to the given query P that consists of 10 documents, the actual optimal classifier h_1 would generate 100 observed errors, in contrast, the classifier h_2 which classifies all examples to be negative would generate 10 observed errors, consequently h_2 is favoured by SVM_{2C} over the actual optimal classifier h_1.

Our key insight is that we are able to train classifiers in the PU Learning setting, if we substitute some other multivariate performance measures for error rate.

Joachims has proposed a SVM formulation that directly minimises the loss function Δ corresponding to a multivariate performance measure [5].

OP 3. SVM^{perf}

$$\begin{aligned} \min_{\mathbf{w},\xi \geq 0} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\xi \\ s.t. \quad & \forall \bar{y}' \in \{+1, -1\}^n \setminus \bar{y}: \\ & \frac{1}{2n}\mathbf{w}^T\sum_{i=1}^{n}(y_i - y_i{}')\mathbf{x}_i \geq \frac{1}{2n}\Delta(\bar{y}', \bar{y}) - \xi \end{aligned}$$

Note that our SVM^{perf} formulation here has a slight difference with its original version [5]: a constant factor $\frac{1}{2n}$ is introduced to the constraints in order to better capture how C scales with training set size [6].

Balanced Accuracy

The *balanced accuracy* of a classifier is the arithmetic average of *sensitivity* and *specificity* [14]. It is also known as the Area Under the ROC Curve (AUC) for just one run [14].

The *actual* balanced accuracy is

$$B = \frac{\Pr[h(\mathbf{x}) = 1|z = 1] + \Pr[h(\mathbf{x}) = -1|z = -1]}{2}.$$

The *observed* balanced accuracy is

$$\widehat{B} = \frac{\Pr[h(\mathbf{x}) = 1|y = 1] + \Pr[h(\mathbf{x}) = -1|y = -1]}{2}.$$

Theorem 1. $\widehat{B} - \frac{1}{2} \propto B - \frac{1}{2}$.

Proof. It can be shown with some simple calculation that [1]

$$\begin{aligned} & (2\widehat{B} - 1)\Pr[y=1]\Pr[y=-1] \\ = \quad & (2B-1)\Pr[z=1]\Pr[z=-1]\mu \end{aligned}$$

Therefore, we have

$$\begin{aligned} \widehat{B} - \frac{1}{2} \quad &= \quad (B - \frac{1}{2})\mu \frac{\Pr[z=1]\Pr[z=-1]}{\Pr[y=1]\Pr[y=-1]} \\ &\propto \quad B - \frac{1}{2} \end{aligned}$$

□

This theorem implies that we can optimise the actual balanced accuracy B by optimising the observed balanced accuracy $\widehat{B}$.

Since $0 \leq \widehat{B} \leq 1$, we define the corresponding multivariate loss function as

$$\Delta_{ba}(\bar{h}(\bar{\mathbf{x}}), \bar{y}) = 1 - \widehat{B}.$$

Let SVM_{ba}^{perf} denote the SVM^{perf} with the loss function Δ_{ba}.

We are able to train SVM_{ba}^{perf} efficiently by transforming it to a specific case of SVM^{struct} — the structural SVM formulation which was first proposed for training SVMs to predict structural outputs [6]. For this purpose we have extended the original SVM^{struct} [6] to assign different weights λ_i to errors on different training examples $\mathbf{x}_i$.

OP 4. $\mathrm{SVM}_{ba}^{struct}$

$$\begin{aligned} \min_{\mathbf{w},\xi\geq 0} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\xi \\ s.t. \quad & \forall \bar{\eta} \in \{0,1\}^n \setminus \bar{0} : \\ & \frac{1}{n}\mathbf{w}^T \sum_{i=1}^{n} \eta_i y_i \mathbf{x}_i \geq \frac{1}{n}\sum_{i=1}^{n} \eta_i \lambda_i - \xi \\ & \lambda_i = 1/(4l) \text{ if } y_i > 0 \text{ and } \lambda_i = 1/(4u) \text{ if } y_i < 0 \end{aligned}$$

Theorem 2. SVM_{ba}^{perf} *is equivalent to* $\mathrm{SVM}_{ba}^{struct}$.

Proof. SVM_{ba}^{perf} and $\mathrm{SVM}_{ba}^{struct}$ have the same objective function to optimise, so we only need to show that they have an equivalent set of constraints.

For each $\bar{y}' \in \{+1,-1\}^n \setminus \bar{y}$, there is a unique corresponding $\bar{\eta} \in \{0,1\}^n \setminus \bar{0}$ through the following one-to-one map:

$$\eta_i = \begin{cases} 0 & \text{if } {y_i}' = y_i \\ 1 & \text{if } {y_i}' \neq y_i \end{cases}.$$

So $(y_i - {y_i}')/2 = \eta_i y_i$, and the left-hand-expression of each inequality constraint in SVM_{ba}^{perf} is same as that in $\mathrm{SVM}_{ba}^{struct}$. Now let's look at the right-hand-expression of each inequality constraint. Noticing $c = \sum_{y_i>0} \eta_i$ and $d = \sum_{y_i<0} \eta_i$, we can re-write $\sum_{i=1}^{n} \eta_i \lambda_i$ as

$$\begin{aligned} \sum_{y_i>0} \eta_i\lambda_i + \sum_{y_i<0} \eta_i\lambda_i \quad &= \quad \frac{1}{4l}\sum_{y_i>0} \eta_i + \frac{1}{4u}\sum_{y_i<0} \eta_i \\ &= \quad \frac{1}{2}\Delta_{ba}(\bar{h}(\bar{\mathbf{x}}), \bar{y}) \end{aligned}$$

we see that the right-hand-expression of each inequality constraint in SVM_{ba}^{perf} also turns out to be same as that in $\mathrm{SVM}_{ba}^{struct}$. Hence both optimisation problems would lead to the same solution $\mathbf{w}^*$. □

Theorem 3. SVM_{ba}^{perf} ($\mathrm{SVM}_{ba}^{struct}$) *can be trained in* linear *time w.r.t. the size of $P \cup U$, i.e., $l + u$.*

Proof. It has been shown that SVM^{struct} can be correctly trained by the *cutting-plane algorithm* in $O(sn)$ time where s is the average number of non-zero features and n is the number of training examples [5]. The same algorithm can be adapted for the training of $\mathrm{SVM}_{ba}^{struct}$ with little modification. In our case, there are $n = |P \cup U| = l + u$ training examples. □

Precision-Recall Product

In information retrieval applications, performance is more often evaluated in terms of *precision* and *recall* [12] rather than accuracy.

The *actual* precision and recall are

$$p = \Pr[z = 1 | h(\mathbf{x}) = 1] \text{ and } r = \Pr[h(\mathbf{x}) = 1 | z = 1]$$

respectively.

The *observed* precision and recall are

$$\hat{p} = \Pr[y = 1 | h(\mathbf{x}) = 1] \text{ and } \hat{r} = \Pr[h(\mathbf{x}) = 1 | y = 1]$$

respectively.

Generally speaking, we want both precision and recall to be high in a retrieval situation. The F_1 score, which addresses precision and recall equally, is probably the most popular multivariate performance measure in IR [12]. It is defined as the harmonic average of precision and recall,

$$F_1 = \frac{2}{\frac{1}{p} + \frac{1}{r}} = \frac{2pr}{p + r}.$$

Unfortunately in PU Learning, high observed F_1 does not guarantee high actual F_1.

We turn to optimise an alternative multivariate performance measure — pr, the product of precision and recall. Similar to the F_1 score, pr is high when both precision and recall are high, but low when either of them is low. In fact pr correlates with F_1 closely. It is easy to see that pr is a lower-bound of F_1 and $\sqrt{pr}$ is an upper-bound of F_1.

Theorem 4. $pr \le F_1 \le \sqrt{pr}$.

Proof. $F_1 \ge pr$ because $0 \le p + r \le 2$. $F_1 \le \sqrt{pr}$ because the harmonic average can never be greater than the geometric average. □

The relationship between the observed precision/recall and the actual precision/recall is given by the following two lemmas.

Lemma 1. $\hat{r} = r$.

Proof. This comes directly from the assumption that the examples in P is *randomly* sampled from the class of actual positive examples.

$$\begin{aligned}\hat{r} &= \Pr[h(\mathbf{x}) = 1|y = 1] \\ &= \frac{\Pr[h(\mathbf{x}) = 1, y = 1]}{\Pr[y = 1]} \\ &= \frac{\Pr[h(\mathbf{x}) = 1, z = 1]\mu}{\Pr[z = 1]\mu} \\ &= \Pr[h(\mathbf{x}) = 1|z = 1] \\ &= r.\end{aligned}$$

□

Lemma 2. $\hat{p} \propto p$.

Proof.

$$\begin{aligned}\hat{p} &= \Pr[y = 1|h(\mathbf{x}) = 1] \\ &= \frac{\Pr[h(\mathbf{x}) = 1|y = 1]\Pr[y = 1]}{\Pr[h(\mathbf{x}) = 1]} \\ &= \frac{\hat{r}\Pr[z = 1]\mu}{\Pr[h(\mathbf{x}) = 1]} = \frac{r\Pr[z = 1]\mu}{\Pr[h(\mathbf{x}) = 1]} \\ &= \frac{\Pr[z = 1, h(\mathbf{x}) = 1]}{\Pr[h(\mathbf{x}) = 1]}\mu \\ &= \Pr[z = 1|h(\mathbf{x}) = 1]\mu \\ &= p\mu \propto p.\end{aligned}$$

□

Theorem 5. $\widehat{pr} \propto pr$.

Proof. It is simply because $\hat{p} \propto p$ and $\hat{r} = r$. □

This theorem implies that we can optimise the actual precision-recall product pr by optimising the observed precision-recall product $\widehat{pr}$.

Since $0 \le \widehat{pr} \le 1$, we define the corresponding multivariate loss function as

$$\Delta_{pr}(\bar{h}(\bar{\mathbf{x}}), \bar{y}) = 1 - \widehat{pr}.$$

Let SVM_{pr}^{perf} denote the SVM^{perf} with the loss function Δ_{pr}.

Theorem 6. SVM_{pr}^{perf} *can be trained in* polynomial *time w.r.t. the size of* $P \cup U$*, i.e.,* $l + u$.

Proof. It has been shown that if the loss function Δ can be computed from the following *contingency table*, SVM^{perf} can be correctly trained by a *sparse-approximation algorithm* [15] in $O(n^2 t)$ time where n is the number of training examples and t is the number of different contingency tables [5].

	$y = +1$ ($\mathbf{x} \in P$)	$y = -1$ ($\mathbf{x} \in U$)
$h(\mathbf{x}) = +1$	a	b
$h(\mathbf{x}) = -1$	c	d

$\Delta_{pr}(\bar{h}(\bar{\mathbf{x}}), \bar{y})$ can be computed from the contingency table:

$$\Delta_{pr}(\bar{h}(\bar{\mathbf{x}}), \bar{y}) = 1 - \frac{a^2}{(a+b)(a+c)}.$$

Given P and U, the values in such a contingency table must satisfy the constraints $a, b, c, d \ge 0$, $a+c = l$ and $b+d = u$. Although there are $n! = (l+u)!$ different rankings, there are only $(l+1)(u+1) \in O(n^2)$ different contingency tables that are legitimate. Therefore SVM_{pr}^{perf} can be trained in at most $O(n^4)$ time. □

3 Experiments

3.1 Code

We have implemented the proposed SVM learning algorithms on the basis of Joachim's SVM^{perf}[1]. Our source code will be made available at the first author's homepage.

3.2 Data

We conduct our experiments on the following three real-world datasets which are pre-processed and publicly available[2].

- The **news20** dataset contains approximately 20,000 articles that were collected from 20 different newsgroups.
- The **siam-competition2007** dataset contains 28,596 aviation safety reports that were used in the SIAM Text Mining Competition 2007.
- The **mediamill-exp1** dataset contains 43,907 camera-shots from 85 hours of international news broadcast video data that were used in the MediaMill Challenge Problem for generic video indexing. Only the top 5 semantic concepts are used as categories in our experiments.

[1] http://svmlight.joachims.org/svm_perf.html

[2] http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets/

3.3 Setting

We construct binary classification tasks based on the default train/test split for each dataset. Given a category, the positive set P consists of the relevant examples before the split point, the negative set N consists of the irrelevant examples before the split point, while the unlabelled set U consists of all (relevant and irrelevant) examples after the split point. When the negative set is available, we can use both P and N to train SVM_{2c} which provides the ideal classification performance or the upper-bound for PU Learning. When the negative set is not available, we can use P only to train SVM_{1c}, or use both P and U to train our proposed SVM_{ba}^{perf} and SVM_{pr}^{perf}.

3.4 Results

We evaluate the classification effectiveness of SVMs on each dataset using the average accuracy, F_1 score and Area Under the ROC Curve (AUC), as shown in Table 1. We see that SVM_{ba}^{perf} and SVM_{pr}^{perf} work very well in the PU Learning setting on all the datasets: their classification performances are much higher than that of SVM_{1c}^{perf}, and are as good as that of SVM_{2c}^{perf} which makes of negative examples. This implies that our proposed classifiers SVM_{ba}^{perf} and SVM_{pr}^{perf} can be trained effectively no negative examples at all.

We run our experiments on a PC with Pentium 4 (3GHz) processor and 2GB memory, and report the average training time (in CPU seconds) of SVMs for PU Learning in Table 2. We see that SVM_{ba}^{perf} and SVM_{pr}^{perf} both can be trained efficiently. Moreover, SVM_{ba}^{perf} runs an order of magnitude faster than SVM_{pr}^{perf}: the linear time complexity of SVM_{ba}^{perf} makes it more scalable than SVM_{pr}^{perf}.

Table 2. The efficiency (training time) of SVMs for PU Learning.

dataset	SVM_{ba}^{perf}	SVM_{pr}^{perf}
news20	0.8905	7.1325
siam-competition2007	1.0409	43.0641
mediamill-exp1	3.6140	731.3120

4 Related Work

The problem of PU Learning has attracted much attention from information retrieval and data mining researchers in recent years [3, 2, 7, 11, 10, 8, 4, 18, 17, 19, 20]. Please refer to Liu's new book on Web data mining [9] for a comprehensive survey of this field.

Generally speaking, most existing approaches to PU Learning follow a two-step heuristic: (1) constructing a small reliable negative set $\widehat{N}$ by extracting some examples from U which look very unlike positive examples; (2) building a classifier based on P and $\widehat{N}$ iteratively. Our reduction approach to PU Learning is fundamentally different with them.

The work most related to ours is probably the Biased-SVM method which has shown excellent classification performance in comparison to other state-of-the-art PU Learning methods [10]. It also attempts to optimise a multivariate performance measure (proportional to pr), but through an indirect trail-and-error way: it tries a large number (typically hundreds) of SVMs each with a different cost parameter and then pick one from them according to the classification performance on a held-out validation set. So theoretically the classification performance of Biased-SVM could not be better than SVM_{pr}^{perf}. Our reduction approach to PU Learning has several advantages over Biased-SVM:

- it should be more effective because it directly optimises meaningful multivariate performance measures;
- it is hundreds of times more efficient because it only needs to train one classifier;
- a held-out validation set is no longer a prerequisite.

5 Conclusions

In this paper, we have proposed to solve the problem of PU Learning by reducing it to the problem of learning classifiers for some meaningful multivariate performance measures (namely balanced accuracy and precision-recall product). Specifically we have presented two variants of standard SVM, SVM_{pr}^{perf} and SVM_{pr}^{perf}, which can be used for effective and efficient PU Learning.

6 Acknowledgements

We thank the anonymous reviewers for their helpful comments.

References

[1] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT)*, pages 92–100, Madison, WI, 1998.

[2] F. Denis, R. Gilleron, and F. Letouzey. Learning from positive and unlabeled examples. *Theoretical Computer Science*, 348(1):70–83, 2005.

Table 1. The effectiveness of SVMs for PU Learning.

dataset	measure	SVM_{1c}	SVM_{ba}^{perf}	SVM_{pr}^{perf}	SVM_{2c}
20news	accuracy	0.7771	0.9795	0.9802	0.9805
	F_1	0.2751	0.7771	0.7887	0.7538
	ROC-AUC	0.9116	0.9877	0.9881	0.9904
siam-competition2007	accuracy	0.4072	0.9389	0.9267	0.9448
	F_1	0.1774	0.5499	0.5606	0.4251
	ROC-AUC	0.9009	0.9740	0.9736	0.9624
mediamill-exp1	accuracy	0.4517	0.7823	0.6640	0.7965
	F_1	0.5841	0.5950	0.6501	0.5846
	ROC-AUC	0.7732	0.8850	0.8833	0.8961

[3] F. Denis, R. Gilleron, and M. Tommasi. Text classification from positive and unlabeled examples. In *Proceedings of the 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)*, pages 1927–1934, Annecy, France, 2002.

[4] G. P. C. Fung, J. X. Yu, H. Lu, and P. S. Yu. Text classification without negative examples revisit. *IEEE Transaction of Knowledge and Data Engineering (TKDE)*, 18(1):6–20, 2006.

[5] T. Joachims. A support vector method for multivariate performance measures. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 377–384, Bonn, Germany, 2005.

[6] T. Joachims. Training linear SVMs in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 217–226, Philadelphia, PA, 2006.

[7] W. S. Lee and B. Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 448–455, Washington, DC, 2003.

[8] X. Li and B. Liu. Learning to classify texts using positive and unlabeled data. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 587–594, Acapulco, Mexico, 2003.

[9] B. Liu. *Web Data Mining: Exploring Hyperlinks, Contents and Usage Data*. Springer, 2006.

[10] B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu. Building text classifiers using positive and unlabeled examples. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM)*, pages 179–188, Melbourne, FL, 2003.

[11] B. Liu, W. S. Lee, P. S. Yu, and X. Li. Partially supervised classification of text documents. In *Proceedings of the 19th International Conference (ICML)*, pages 387–394, Sydney, Australia, 2002.

[12] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[13] B. Scholkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[14] M. Sokolova, N. Japkowicz, and S. Szpakowicz. Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation. In *Proceedings of the AAAI'06 workshop on Evaluation Methods for Machine Learning*, 2006.

[15] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6:1453–1484, Sep 2005.

[16] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 42–49, Berkeley, CA, 1999.

[17] H. Yu. Single-class classification with mapping convergence. *Machine Learning*, 75(1):49–69, 2005.

[18] H. Yu, J. Han, and K. C.-C. Chang. PEBL: Web page classification without negative examples. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 16(1):70–81, 2004.

[19] D. Zhang and W. S. Lee. A simple probabilistic approach to learning from positive and unlabeled examples. In *Proceedings of the 5th Annual UK Workshop on Computational Intelligence (UKCI)*, pages 83–87, London, UK, 2005.

[20] D. Zhang and W. S. Lee. Learning with support vector machines for query-by-multiple-examples. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 835–836, Singapore, 2008.