Document-Word Co-Regularization for Semi-supervised Sentiment Analysis[*]

Vikas Sindhwani and Prem Melville
Business Analytics and Mathematical Sciences
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598
{vsindhw,pmelvil}@us.ibm.com

Abstract

The goal of sentiment prediction is to automatically identify whether a given piece of text expresses positive or negative opinion towards a topic of interest. One can pose sentiment prediction as a standard text categorization problem. However, gathering labeled data turns out to be a bottleneck in the process of building high quality text classifiers. Fortunately, background knowledge is often available in the form of prior information about the sentiment polarity of words in a lexicon. Moreover, in many applications abundant unlabeled data is also available. In this paper, we propose a novel semi-supervised sentiment prediction algorithm that utilizes lexical prior knowledge in conjunction with unlabeled examples. Our method is based on joint sentiment analysis of documents and words based on a bipartite graph representation of the data. We present an empirical study on a diverse collection of sentiment prediction problems which confirms that our semi-supervised lexical models significantly outperform purely supervised and competing semi-supervised techniques.

1 Introduction

In recent years there has been an explosion of user-generated content on the Internet in the form of weblogs (blogs), discussion forums and online review sites. This phenomenon presents many new opportunities and challenges to both producers and consumers alike. For producer, this user-generated content provides a rich source of implicit consumer feedback. Tracking the pulse of this ever-expanding blogosphere, enables companies to discern what consumers are saying about their products, which provides useful insight on how to improve or market products better. This has given rise to several sites (e.g. [2]) and systems (e.g. [29]) that monitor trends in the blog-based reputation of specific companies and products. For consumers, the plethora of information and opinions from diverse sources helps them tap into the wisdom of crowds, to aid in making more informed decisions. These decisions could range from which new digital camera to buy, which movie to watch, or even who to vote for in upcoming elections.

Though there is a vast quantity of information available, the consequent challenge is to be able to analyze the millions of blogs available, and to glean meaningful insights therein. One key component of this process is to be able to gauge the sentiment expressed in blogs around selected topics of interest. The emerging area of Sentiment Analysis [31, 21, 13] focuses on this task of automatically identifying whether a piece of text expresses a positive or negative opinion towards the subject matter. Detecting the sentiment expressed in documents turns out be an extremely difficult task, and the performance of sentiment classifiers can vary a great deal depending on the domain [30]. As such, one of the grand challenges of sentiment analysis is to build a robust system that provides insights across a growing list of different products and topics of interest. Such a system needs to be able to rapidly adapt to new domains with minimal supervision.

Most prior work in sentiment analysis use knowledge-based approaches, that classify the sentiment of texts based on lexicons defining the sentiment-polarity of words, and simple linguistic patterns. There have been some recent studies that take a machine learning approach [21, 11], and build text classifiers trained on documents that have been human-labeled as *positive* or *negative*. The knowledge-based approaches tend to be non-adaptive, while the learning approaches do not effectively exploit prior knowledge and require much effort through human annotation of documents. In this paper, we present a new machine learning approach that overcomes both drawbacks of previous learning approaches. Firstly, we incorporate prior knowledge of sentiment-laden terms directly into our model. Secondly, in order to adapt to new domains with minimal supervision, we also exploit the large amount of unlabeled

[*]Technical Report. A shorter version will appear in ICDM 2008.

data readily available. We present a unified framework in which lexical background information, unlabeled data and labeled training examples can be effectively combined. We demonstrate the generality of our approach, by presenting results on three, very different, domains — blogs discussing enterprise-software products, political blogs discussing US Presidential candidates, and online movie reviews.

We begin in Section 2 by discussing related work. Starting from Section 3, we outline a series of linear regularization models that incorporate increasing amounts of information. Section 4 presents our methods. Section 5 presents empirical results and Section 6 concludes this paper.

2 Related Work

In this section, we first review prior work in sentiment analysis. We then discuss related work in the incorporation of background knowledge and unlabeled data in learning.

2.1 Sentiment analysis

Most work in sentiment analysis has focused on identifying positive or negative sentiment in text passages online. These studies can be broadly classified into two categories: knowledge-based approaches and learning-based approaches. Knowledge-based approaches primarily use linguistic models or other forms of background knowledge to classify the sentiment of passages. A large focus of this area is the use and generation of dictionaries capturing the sentiment of words. These methods range from manual approaches of developing domain-dependent lexicons [7] to semi-automated approaches [13, 35, 17], and even an almost fully automated approach [30]. As observed by Ng et al. [19], most semi-automated approaches yield unsatisfactory lexicons, with either high coverage and low precision or vice versa.

More recently, Pang et al. [21] successfully applied a machine learning approach to classifying sentiment for movie reviews. They cast the problem as a text classification task, using a bag-of-words representation of each movie review. They demonstrate that a learning approach performs better than simply counting the positive and negative sentiment terms using a hand-crafted dictionary. However, they do not consider combining such background lexical information or unlabeled data with supervised learning, as we propose in this paper. Their results also suggest that using more sophisticated linguistic models, incorporating parts-of-speech and n-gram language models, do not improve over the simple unigram bag-of-words representation. In keeping with their findings, we also adopt a unigram text model.

Pang et al. [20] extend their work, by first classifying sentences as *subjective* versus *objective*, and then classifying only the *subjective* sentences based on sentiment polarity. They demonstrate that by focusing only on the subjective sentences in each review they were able to improve the overall sentiment classification accuracy. Such a two-staged approach could also improve our results; however, for this paper we focus only on our advances in the polarity prediction stage.

Durant and Smith [11] also apply a text categorization approach to classification of political alignment in blog posts. Although their data is similar to ours, their task of identifying *left* versus *right* political alignment is quite different from our goal of identifying positive and negative sentiment. They focus on improving classification through feature selection, and unlike our work, do not exploit alternative sources of information, such as lexicons and unlabeled data.

2.2 Use of background knowledge and unlabeled data in learning

Recently, there has been a growing interest in the use of background, prior or domain knowledge in supervised learning — including methods that use human-provided associations of features to particular classes. Most of this work has focused on using such prior class-bias of features to generate labeled examples that are then used for standard supervised learning. Schapire et al. [24] propose one such framework for boosting logistic regression, that uses hand-crafted rules generated from a list of relevant features to label *pseudo-examples*. They modify the boosting objective function to fit the training data, and the prior model based on these pseudo-examples.

Liu et al. [18] use background knowledge to generate labeled training examples from a large pool of unlabeled examples. In their work, they focus on the process of selecting features to be labeled by humans — using clustering, followed by entropy-based feature selection. Eventually, they use the Expectation-Maximization algorithm with Naïve Bayes to build classifiers trained on representative examples selected for each class and the remaining unlabeled data.

Provided with some features associated with each class, Wu and Srihari [32] assign labels to unlabeled documents, which are then used in conjunction with labeled examples to build a Weighted Margin Support Vector Machine. Unlike the above approaches, our methods directly couple linear model estimation with background knowledge incorporation in a single regularization-based optimization problem.

Dayanik et al. [8] explore several approaches to incorporating prior knowledge into Logistic Regression. In their study, human-annotated relevant features are given more ability to affect classification by assigning them a larger prior mode or variance than other features. Their approach of *mode-setting* is somewhat related to ours. However, they report that this method was unreliable; since, it occasionally produced the best, but usually produced the worst results compared to other approaches.

Finally, Druck et al. [10] incorporate prior knowledge through *labeled features*, which are used to directly constrain the model's predictions on unlabeled instances. Their Generalized Expectation criteria approach is applicable to any discriminative probabilistic model, and they demonstrate its utility specifically for multinomial logistic regression. Unlike their approach which uses only unlabeled instances, our method uses both labeled and unlabeled data in conjunction with labeled and unlabeled features.

3 Linear Sentiment Classification Models

In text classification, a document is typically represented as a bag-of-words feature vector $\boldsymbol{x} \in \mathcal{R}^D$. The entries in this vector usually specify frequencies, or weighted frequencies, for D words in a pre-specified vocabulary $\mathcal{V}$. Given such a representation, a linear classification model is specified by a weight vector $\mathbf{w} \in \mathcal{R}^D$ and a bias parameter b. The classification to positive $(+1)$ or negative (-1) class is obtained by evaluating the function $h(\boldsymbol{x}) = sign(\mathbf{w}^\top \boldsymbol{x} + b)$. For notational simplicity in the presentation below, let us re-define $\boldsymbol{x}$ as $(\boldsymbol{x}, 1)$ and $\mathbf{w}$ as $(\mathbf{w}, b)$ so that the classification rule may be written more compactly as $h(\boldsymbol{x}) = \mathbf{w}^\top \boldsymbol{x}$. We refer to the last component of $\boldsymbol{x}$ as the bias feature and the last component of $\mathbf{w}$ as the bias weight respectively. We next discuss ways to set the weight vector, $\mathbf{w}$, for sentiment classification given different kinds of information.

3.1 Unsupervised Lexical Classification

In the absence of any labeled data in a domain, one can build sentiment-classification models that rely solely on back- ground knowledge, such as a lexicon defining the polarity of words. Suppose we are given a manually constructed lexicon of positive and negative terms which we denote by $\mathcal{V}_+ \subset \mathcal{V}$ and $\mathcal{V}_- \subset \mathcal{V}$ respectively. One straightforward approach to using this information is to measure the relative frequency of occurrence of positive and negative terms in a document. The classification rule is then be given by,

$$h(\boldsymbol{x}) = sign(\sum_{i \in \mathcal{V}_+} x_i - \sum_{i \in \mathcal{V}_-} x_i) \tag{1}$$

This corresponds to the choice $w_i = +1$ for all $i \in \mathcal{V}_+$, $w_i = -1$ for all $i \in \mathcal{V}_-$, and $w_i = 0$ for all other terms. We denote the weight vector of such a lexical classifier by $\mathbf{w}_{lex}$. Note that the bias component is automatically set to 0, which is appropriate in the absence of any class distribution information.

For this study, we used a lexicon generated by the IBM India Research Labs that was developed for other text mining applications [22]. It contains 2,968 words that have been human-labeled as expressing positive or negative sentiment. In total, there are 1,267 positive and 1,701 negative unique terms after stemming. We eliminated terms that were ambiguous and dependent on context, such as dear and fine. It should be noted, that this list was constructed without a specific domain in mind; which is further motivation for using training examples and unlabeled data to learn domain-specific connotations.

3.2 Supervised Regularization Models

In a setting where l labeled documents, $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^l$, are available with $y_i \in \{+1, -1\}$, one may attempt to learn $\mathbf{w}$ by solving an optimization problem of the form,

$$\mathbf{w}^\star = \underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{l} \sum_{i=1}^{l} V(\mathbf{w}^T \boldsymbol{x}_i, y_i) + \frac{\gamma}{2} \|\mathbf{w}\|_2^2$$

where $V(\cdot,\cdot)$ is a loss function and γ is a real-valued regularization parameter. For various choices for the loss function V, this optimization problem spans a large family of learning algorithms. Popular choices include the hinge loss: $V(\mathbf{w}^\top \boldsymbol{x}, y) = \max\left[0, 1 - y\mathbf{w}^\top \boldsymbol{x}\right]$, logistic loss: $V(\mathbf{w}^\top \boldsymbol{x}, y) = \log\left[1 + \exp(-y\mathbf{w}^\top x)\right]$ and the squared loss: $V(\mathbf{w}^\top \boldsymbol{x}, y) = \frac{1}{2}(\mathbf{w}^\top \boldsymbol{x} - y)^2$, which respectively lead to the Support Vector Machine (SVM), Logistic Regression and the classical Regularized Least Squares (RLS) algorithms[1].

Our methods build on RLS due to its simplicity and excellent performance on classification tasks [23, 33]. In this case, the solution is given by the $D \times D$ linear system,

$$\left[\frac{1}{l}\mathbf{X}^\top \mathbf{X} + \gamma \mathbf{I}\right] \mathbf{w} = \frac{1}{l}\mathbf{X}^\top \mathbf{y} \tag{2}$$

where $\mathbf{X}$ is the $l \times D$ data matrix whose rows are document vectors, and $\mathbf{y}$ is the vector of labels. Since documents almost always only contain a very small fraction of words in the vocabulary, the data matrix $\mathbf{X}$ is highly sparse. Due to this fact, the above linear system can be very efficiently solved for large-scale problems (where both l and D are large) using sparse iterative techniques such as Conjugate Gradient.

4 Semi-supervised Lexical Classification

In this section, we begin by first incorporating lexical knowledge in supervised learners. In section 4.2 we extend this approach to also include unlabeled data.

[1]The bias weight is often excluded from the regularizer $\|\mathbf{w}\|^2$, though including it brings about simplifications without any performance consequences. See, e.g.,[16] for a discussion.

4.1 Incorporating Lexical Knowledge in Supervised Regularization Models

It is well-known that RLS may be interpreted as maximum aposteriori (MAP) estimation under a Gaussian likelihood model for errors $\left(y_i - \mathbf{w}^\top \boldsymbol{x}_i\right)$, and a zero-mean Gaussian prior for the weight parameters $\mathbf{w}$. A natural way to incorporate lexical prior knowledge is to assume a Gaussian prior for $\mathbf{w}$ with non-zero mean propotional to the lexical weight vector $\mathbf{w}_{lex}$. This immediately implies the following modified MAP estimation problem,

$$\underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{l} \sum_{i=1}^{l} V(\mathbf{w}^\top \boldsymbol{x}_i, y_i) + \frac{\gamma}{2} \|\mathbf{w} - \nu \mathbf{w}_{\text{lex}}\|_2^2 \quad (3)$$

where ν is a parameter. It can be easily seen that the solution is given by the following modified linear system,

$$\left[\frac{1}{l}\mathbf{X}^\top\mathbf{X} + \gamma\mathbf{I}\right]\mathbf{w} = \frac{1}{l}\mathbf{X}^\top\mathbf{y} + \gamma\nu\mathbf{w}_{\text{lex}} \quad (4)$$

We call this approach Lexical-RLS (LEX+RLS). The only difference above with respect to RLS (Eqn. 2) is the second term of the right-hand-side which incorporates the lexical weights. Note that Lexical RLS reduces to RLS when ν is set to 0, and defines the the lexical classifier (Eqn. 1) when there are no labeled examples, or when $\gamma \to \infty$.

4.2 Semi-supervised Learning in the Presence of Lexical Knowledge

Suppose now that in addition to lexical knowledge, we also have access to a large collection of unlabeled documents. Most semi-supervised classification algorithms implement the classical cluster assumption [3] which states the following: *if two documents are in the same cluster, they are likely to be of the same class*. Low-density techniques [14, 5, 4] implement this assumption by attempting to find separators that do not cut thru unlabeled data clusters. Similarly, Graph-based techniques [15, 34, 1] use unlabeled examples to find classifiers that give smooth predictions on data clusters.

In the presence of lexical knowledge we may further qualify the cluster assumption as follows: *if two documents are in the same cluster dominantly supported on positive (negative) sentiment words, they are likely to be positive (negative) sentiment documents*. In other words, the sentiment lexicon may be viewed as prior knowledge on the structure of the data clusters over which the cluster assumption ought to be enforced. Moreover, note that there is a clear duality between documents and words. Representing a word by its distribution vector over documents (i.e, a column of $\mathbf{X}$), one can formulate an equivalent cluster assumption for *words*: *if two words are in the same cluster dominantly supported on positive (negative) sentiment documents, they are likely to be positive (negative) sentiment words.* The sentiment polarity of documents determines the polarity of words, while the polarity of words determines the polarity of documents.

We now present a novel semi-supervised learning algorithm that simultaneously implements these dual document-word cluster assumptions while incorporating semi-supervision along both dimensions. We begin by introducing a bipartite graph representation of the data, previously utilized in the context of co-clustering [9]. We then formulate joint sentiment classification of documents and words in terms of transductive prediction on this graph whose nodes are viewed to be partially labeled. However, since this approach is strictly transductive and does not allow prediction on new completely unseen test documents, we formulate a new objective function that simultaneously projects the transductive solution to a linear model. We now outline these steps leading to the proposed algorithm.

Document-Word Bipartite Graph: In the semi-supervised setting, let $\mathbf{X}$ denote the $n \times D$ data matrix whose rows are the set of l labeled and $(n-l)$ unlabeled document vectors. Consider a bipartite graph, denoted by $\mathcal{G}$, with two sets of vertices: one corresponding to the n documents, and another corresponding to the D words in the vocabulary. Thus, $\mathcal{G}$ has $n + D$ vertices. An undirected edge (i, j) exists if the i^{th} document contains the j^{th} word. Since $\mathcal{G}$ is bipartite, there are no edges between words or between documents. An edge signifies an association between a document and a word. By putting positive weights on the edges, we can capture the strength of this association. We use $\mathbf{X}_{ij}$ as the edge weight which corresponds to frequency (or idf-weighted frequency) of term j in document i. Then, the $(n + D) \times (n + D)$ adjacency matrix of $\mathcal{G}$ can be easily seen to be given by,

$$\mathbf{A} = \begin{pmatrix} 0 & \mathbf{X} \\ \mathbf{X}^\top & 0 \end{pmatrix} \quad (5)$$

where the vertices of the graph are ordered by taking the n documents (same order as rows of $\mathbf{X}$) followed by the D words (same order as columns of $\mathbf{X}$).

Transductive Sentiment Prediction: Next, we view $\mathcal{G}$ as a partially labeled graph. Given sentiment labels for a few document and word vertices, consider the problem of completing the labeling of the rest of the vertices of the graph. Such prediction problems on graphs have been well-studied in the graph-based semi-supervised learning literature [15, 34, 1], but to the best of our knowledge they have never been applied to solve joint prediction problems on document and words. Our goal is to learn a real-valued sentiment-polarity score vector, $\boldsymbol{f}^d$, over document vertices and $\boldsymbol{f}^w$ over word vertices with the following properties: (a) If the i^{th} document is labeled, f_i^d should be close to the ± 1-valued label, (b) If the j^{th} word is labeled, f_j^w should be close to the ± 1-valued label and (c) If the association between the i^{th} document and the j^{th} word is strong, then f_i^d and f_j^w should be similar. It is important to note that the third property can be enforced also over unlabeled documents and unlabeled words. It turns out that the third property has close connections to the classical SVD applied to document-term matrices (see [9] for more details).

These three properties can be enforced through the terms of the objective function in the following minimization problem,

$$\underset{\boldsymbol{f}^d, \boldsymbol{f}^w}{\operatorname{argmin}} \frac{1}{l_d} \sum_{i=1}^{l_d} V(f_i^d, y_i^d) + \frac{1}{l_w} \sum_{i=1}^{l_w} V(f_i^w, y_i^w) + \mu \sum_{i=1}^{n} \sum_{j=1}^{D} \mathbf{X}_{ij} \left(f_i^d - f_j^w\right)^2 \quad (6)$$

where V as before is a loss function, l_d is the number of labeled documents, l_w is the number of labeled words, μ is a real-valued parameter. We also assume that the first l_d documents in $\mathbf{X}$ of the n total are the ones that are labeled. The third term can be shown to be a quadratic form involving the graph Laplacian matrix $\mathbf{L}$ [6] of $\mathcal{G}$, i.e.,

$$\sum_{j=1}^{D} \mathbf{X}_{ij} \left(f_i^d - f_j^w\right)^2 = \left(\boldsymbol{f}^{d\top} \boldsymbol{f}^{w\top}\right) \mathbf{L} \begin{pmatrix} \boldsymbol{f}^d \\ \boldsymbol{f}^w \end{pmatrix} \quad (7)$$

where the graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$ where $\mathbf{D}$ is the diagonal degree matrix associated with $\mathbf{A}$, i.e., $D_{rr} = \sum_s A_{rs}$. In particular, we use the associated *normalized* Graph Laplacian [6] in our formulations below, defined as $\tilde{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}$. The solution to Eqn. 6 can be obtained by solving a sparse linear system (see [1, 3] for details on Graph transduction in general).

Intuitive Interpretations: The transductive sentiment scores obtained by solving Eqn. 6 may be interpreted in different ways (see [3] for more discussion). The Random walk interpretation is as follows. Imagine starting from an *unlabeled* document i and walking to a word j in it with probablity $\frac{\mathbf{X}_{ij}}{\sum_j X_{ij}}$. Then, from j we walk to another document k with probability $\frac{\mathbf{X}_{kj}}{\sum_k X_{kj}}$. Continuing in this way bouncing between documents and words until a labeled node (document or word) is found, one can ask for the probability p of terminating the random walk at a positive sentiment document or word. The score given to the unlabeled document i is then $2p - 1$. Similarly, the random walk may be started from an unlabeled word to obtain a sentiment polarity score for that word. Another interpretation is the following: Consider $\mathcal{G}$ to be an electric network. Imagine connecting positive sentiment documents and words to a positive voltage source (+1V) and negative sentiment documents and words to a negative voltage source (-1V). Let X_{ij} be the conductance (inverse of resistance) between a document i and a word j. Then the sentiment score given to an *unlabeled* document or a word is the resulting voltage at that node in this electric network. Strictly speaking, these interpretations hold when the scores for labeled nodes are clamped at the labels while the third term in Eqn. 6 is minimized (this corresponds to the limiting solution of Eqn. 6 when $\mu \to 0$).

Smoothness Operators: The Laplacian matrix $\tilde{L}$ defines a large family of graph regularizers. To see this, note the following. $\tilde{L}$ can be easily shown to be a symmetric positive definite matrix. Let us denote its eigenvalues by $\lambda_1 \leq \ldots \leq \lambda_{n+D}$ and the associated complete orthonormal set of eigenvectors by $\phi_1 \ldots \phi_{n+D}$. Therefore, the spectral decomposition of the Laplacian is given as $\tilde{L} = \sum_{i=1}^{n+D} \lambda_i \phi_i \phi_i^\top$. These eigenvectors are ordered by their smoothness with respect to the graph $\mathcal{G}$ since it is easy to see that the quadratic form $\phi_i^\top \tilde{L} \phi = \lambda_i$ (recall from Eqn 7 that this quadratic form measures smoothness with respect to document-word associations). Since $\{\phi_i\}_{i=1}^{D}$ constitute a basis for $\mathcal{R}^{n+D}$, we can write $\left(\boldsymbol{f}^{d\top} \; \boldsymbol{f}^{w\top}\right) = \sum_{i=1}^{n+D} \alpha_i \phi_i$, and re-express the smoothness measure in Eqn 7 as,

$$\left(\boldsymbol{f}^{d\top} \boldsymbol{f}^{w\top}\right) \mathbf{L} \begin{pmatrix} \boldsymbol{f}^d \\ \boldsymbol{f}^w \end{pmatrix} = \sum_{i=1}^{n+D} \alpha_i^2 \lambda_i$$

While the l_2 norm of $\left(\boldsymbol{f}^{d\top} \; \boldsymbol{f}^{w\top}\right)$ is $\sum_{i=1}^{n+D} \alpha_i^2$, the smoothness measure above may be seen as a data-dependent norm that puts more weight on non-smooth high-frequency components (higher eigenvectors). We can construct other graph regularizers such this data dependent norm equals $\alpha_i^2 r(\lambda_i)$ for an increasing function $r(\cdot)$. Different choices of r enforce different amounts of smoothness generating a whole family of graph-based smoothness operators. See [28] for typical choices. In particular, we use $\mathbf{M} = \tilde{L}^p$ which corresponds to the choice $r(\lambda) = \lambda^p$ (note that the explicit eigendecomposition of $\tilde{L}$ is not required) where p parameterizes the amount of penalty put on high frequency components. In subsequent discussion, we use $\mathbf{M}$ to denote a generic graph regularizer derived from the Laplacian.

Out-of-Sample Prediction: Note that while $\boldsymbol{f}^d, \boldsymbol{f}^w$ provide sentiment polarity predictions for unlabeled documents and words, they do not provide a model that can be applied to unseen test data. To obtain a linear model, we propose a novel formulation that comprises of solving the following minimization problem,

$$\underset{\boldsymbol{f}^d, \boldsymbol{f}^w, \mathbf{w}}{\text{argmin}} \;\; \frac{\mu}{2(n+D)} \left({\boldsymbol{f}^d}^\top {\boldsymbol{f}^w}^\top \right) M \begin{pmatrix} \boldsymbol{f}^d \\ \boldsymbol{f}^w \end{pmatrix}$$
$$+ \frac{1}{l_d} \sum_{i=1}^{l} V(f_i^d, y_i^d) + \frac{1}{l_w} \sum_{i=1}^{l} V(f_i^w, y_i^w)$$
$$+ \frac{1}{2n} \sum_{i=1}^{n} \left(\mathbf{w}^\top \boldsymbol{x}_i - f_i^d \right)^2 + \frac{\gamma}{2} \|\mathbf{w} - \nu \boldsymbol{f}^w\|_2^2 \qquad (8)$$

The first four terms are inspired by the transductive formulation in Eqn. 6. The last two terms couple transductive learning with a linear model. In particular, through these terms we enforce the following: (a) the outputs produced by the linear model on documents, $\mathbf{w}^\top \boldsymbol{x}_i$, should be close to the transductive solution on document vertices, f_i^d, and (b) the weights of the linear model should be propotional to the sentiment polarity of words as learnt transductively through $\boldsymbol{f}^w$.

Proposed Algorithm: Let $\boldsymbol{y}^d$ denote the $n \times 1$ label vector for documents with entry 0 for unlabeled documents. Similarly, let $\boldsymbol{y}^w$ denote the $D \times 1$ label vector for words with entry 0 for unlabeled words (words not in the sentiment lexicon). Choosing V to be the squared loss, we obtain the solution to Eqn. 8 by solving the following linear system of size $(n + 2D) \times (n + 2D)$:

$$Q \begin{pmatrix} \boldsymbol{f}^d \\ \boldsymbol{f}^w \\ \mathbf{w} \end{pmatrix} = \begin{pmatrix} \frac{1}{l_d} \boldsymbol{y}^d \\ \frac{1}{l_w} \boldsymbol{y}^w \\ 0 \end{pmatrix} \qquad (9)$$

where $Q = T1 + T2 + T3 + T4$ given by

$$\begin{aligned} T1 &= \frac{\mu}{n+D} \begin{pmatrix} \mathbf{M} & 0 \\ 0 & 0 \end{pmatrix} \\ T2 &= \begin{pmatrix} \mathbf{I} & 0 & -\mathbf{X} \\ 0 & 0 & 0 \\ -\mathbf{X}^\top & 0 & \mathbf{X}^\top \mathbf{X} \end{pmatrix} \\ T3 &= \gamma \begin{pmatrix} 0 & 0 & 0 \\ 0 & \nu^2 \mathbf{I} & -\nu \mathbf{I} \\ 0 & -\nu \mathbf{I} & \mathbf{I} \end{pmatrix} \\ T4 &= diag(\frac{1}{l_d}[\boldsymbol{y}^d \neq 0], \frac{1}{l_w}[\boldsymbol{y}^w \neq 0], 0) \end{aligned} \qquad (10)$$

where the elements of $[\boldsymbol{y}^d \neq 0]$ equal 1 for indices corresponding to labeled documents and 0 otherwise. Above, we use $\mathbf{I}$ and 0 to denote identity and zero matrices of appropriate size. We solve the linear system in Eqn. 9 using the Conjugate Gradient (CG) method with a tolerance of $\epsilon = 0.0001$. Note that Q need not be explicitly computed and stored. Rather, since CG only accesses Q through matrix-vector multiplication of the form $\boldsymbol{v} = Q\boldsymbol{u}$, we compute this product efficiently on the fly using just the data matrix and the document-word label vectors. Since this is a core computation in our algorithm, we also give a short piece of Matlab code below to emphasize the ease with which matrix-vector products with Q can be computed.

```
function v = matvec(u,X,yd,yw,mu,gamma,nu,p)
 % Gives matrix vector product Q times a n+2D vector u
 % X is the nxD data matrix
 % yd is the nx1 document label vector (entry 0 for unlab)
 % yc is the Dx1 word label vctor (0 for unlab)
 % mu,gamma,nu are parameters defining Q
 % p defines M. i.e., M = L^p
 % where L is normalized laplacian
 [n,D]=size(X);
 dd = sum(X,2); % for normalizing laplacian
 dd(find(dd))=1./sqrt(dd(find(dd)));
 dc = sum(X)'; % for normalizing laplacian
 dc(find(dc))=1./sqrt(dc(find(dc)));
 fd = u(1:n);
 fw = u(n+1:n+D);
  w = u(n+D+1:n+2*D);
  g = [fd;fw]
  for i=1:p
     gd = dd.*g(1:n);
     gw = dc.*g(n+1:n+D);
      g = g - [dd;dc].*[X*gw; X'*gd];
  end
  v=mu*[g;zeros(D,1)]/(n+D);
  v=v + [fd - X*w; zeros(D,1); -X'*fd + X'*(X*w)]/2*n;
  v=v + gamma*[zeros(n,1); nu*(nu*fw-w); w - nu*fw];
  v=v + [(yd~=0)/sum(yd~=0);(yw~=0)/sum(yw~=0); zeros(D,1)].*u;
```

To obtain the exact solution, theoretically $n + D$ CG iterations are needed. However, very high quality approximate solutions are obtained extremely quickly (convergence depends on the practical rank of Q) in practice. We call our approach Semi-supervised Lexical Regularized Least Squares (SS+LEX+RLS) classification.

Advantages of the Proposed Algorithm: Unlike Transductive SVMs [14, 4] our algorithm is based on convex optimization and therefore does not suffer from local minima issues. Unlike, typical graph-based methods [15, 34, 1] which require an expensive construction of a nearest neighbor graph, our algorithm uses regularization operators defined on the biparite document-word graph. Thus, there is no expensive graph construction step. To the best of our knowledge, our algorithm is the first semi-supervised method that attempts to simultaneously implement cluster assumptions along both dimensions of the data matrix and exploit the duality of document-word associations in order to use lexical prior knowledge in the presence of unlabeled examples. Joint document-word analysis has previously been explored in the context of co-clustering in [9]. Our algorithm may be seen as providing two additional capabilities on top of the bipartite co-clustering approach: (a) the capibility to use semi-supervision for both document and words, and (b) the capability to produce out-of-sample predictions through a linear model. Finally, our algorithm is also closely related to a large class of multi-view learning algorithms, in particular the *co-regularization* approach of [27].

5 Empirical Study

5.1 Data sets

In order to test the generality of our approach we experimented on three qualitatively different domains — blogs discussing enterprise software, blogs about US Presidential candidates, and movie reviews. We describe each of these datasets in more detail below.

One non-trivial aspect of blog data collection for sentiment analysis is the extraction of the *relevant* text from the raw content. Blogs are inherently more diverse in layout and structure than movie or product reviews. For instance, many blogs have a significant number of comments from individuals other than the blogger, as well as explicit citations. These comments and citation often exhibit the exact opposite sentiment from the main content. However, automatically separating the core content from the citations and comments is quite difficult. In order to pre-process our blog data, we use the algorithm provided by [12] to extract text from parts of the Web-page where the ratio of HTML tags to words is above a minimal threshold. While this method works well in removing many of the key identifying characteristics of the blogger (e.g. blog title and blog rolls), it retains longer posts and their comments.

Lotus blogs: One of our motivations for mining sentiment of user-generated content is to automate the analysis of blog posts as they relate to products and brand names. Towards this goal, we created a data set targeted at detecting sentiment around enterprise software, specifically pertaining to the IBM Lotus brand. The Lotus data set we created, consists of posts from 14 individual blogs, 4 of which are actively posting negative content on the brand, with the rest tending to write more positive or neutral posts. In this data set, negative blog posts often complain about user interface challenges or software bugs. For example, a comment like “Could someone please tell me why Lotus notes takes 99% of my CPU usage?” could be seen as negative, while “DAMO demo provides a list of reason to go Lotus” is positive. The Lotus data was collected by downloading the latest posts from each blogger’s RSS feeds, or accessing the blog’s archives, if they exist. Each post was then read and labeled by hand as either positive, negative, neutral, or not relevant. For our analysis, only positive and negative posts were retained, creating a labeled set of 34 and 111 examples, respectively. In addition to the labeled set, we also created a unlabeled set by randomly sampled 2000 posts from a universe of 14,258 blogs that discuss issues relevant to Lotus software. Since some bloggers tend to exhibit consistently positive or negative sentiment, only the body of every post was used in the analysis, thus avoiding blog titles and recurring information like user names, which may ultimately lead to over-training of the models.

Political candidate blogs: For our second domain, we used data that we have been gathering from 16,742 political blogs, which contain over 500,000 posts. We focused our labeling effort on randomly selected posts containing the term “clinton” or “obama” in their URLs. Unlike the Lotus-focused posts, the political posts all come from diverse blogs. Furthermore, from the experience of human labelers, it appears that political sentiment is much more difficult to label than software reviews, as posts tend to be more emotional, discuss issues only implicitly related to candidates (e.g. economic or foreign policies), and may also use cultural and emotional references to pass judgment. A post was labeled as having positive or negative sentiment about a specific candidate (Barack Obama or Hillary Clinton) if it explicitly mentioned the candidate in positive or negative terms. Objective statements and quotations from newspapers and other sources were ignored. Similarly, if the blogger made implicit statements about a candidate (e.g. discussing racism or sexism in elections without specifically mentioning a candidate), that post would not be associated with sentiment. Essentially, only posts with clear opinions about a candidate were labeled and included in this analysis. For example, “I think Hillary Clinton is not doing good in this debate.” would be a labeled as negative for Clinton. On the other hand, “Obama names top fund-raisers, gives more details than Clinton.” would be seen as neutral because the reader cannot assign sentiment without making a personal value judgment on the statement. Throughout labeling, only positive and negative posts were retained – those labeled as neutral and not relevant were discarded. Similarly, if a post was seen as negative about one candidate and positive about another, then the post was also discarded. While discarding posts in such a manner is not an option if one were to deploy a classifier in a production environment, such a rigorous process of post selection was used to build a clean test set to evaluate our methods. The final labeled Political data set consisted of 49 positive and 58 negative posts. We created an additional set of 2000 unlabeled examples, that were sampled from all available posts from our political blogs. This unlabeled set contains 1000 posts containing the term “clinton” and 1000 containing “obama” in their URLs.

Movie reviews: Apart from the blog data that we collected, we also used the publicly available data set of movie reviews provided by Pang et al. [21]. This data set consists of 1000 positive and 1000 negative reviews from the Internet Movie Database. Positive labels were assigned to reviews that had a rating above 3.5 stars and negative labels were assigned to the rest.

Table 1 summarizes the properties of our labeled datasets; where, the *size* refers to the number of labeled examples (*positive* or *negative*), and the *positive rate* is the proportion of the examples that are positive. Both LOTUS and POLITICAL datasets also have an additional 2000 unlabeled examples, which are used in our semi-supervised setting. For MOVIES, we held-out labels of examples when unlabeled examples were required in our experiments.

Table 1. Summary of labeled data sets.

Domain	Size	Positive Rate
LOTUS	145	0.23
POLITICAL	107	0.46
MOVIES	2000	0.50

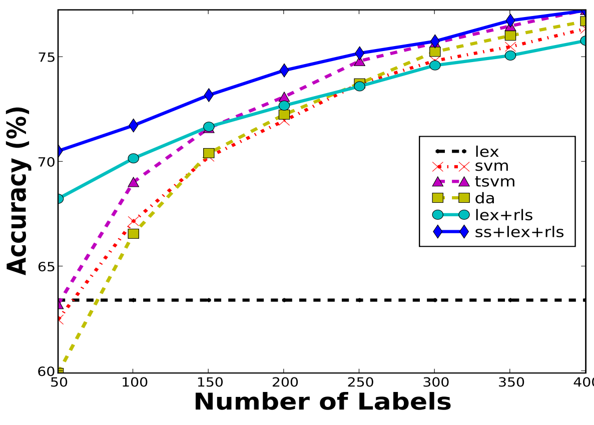
5.2 Results

We compare the approaches proposed in this paper: the lexical RLS (LEX+RLS) and the semi-supervised lexical RLS (SS+LEX+RLS), to the following: (a) unsupervised lexical classification (LEX) which gives a baseline, (b) Linear SVMs which are considered state-of-the-art for text classification, and (c) two implementations of the Transductive SVM [14, 26], one based on label switching (TSVM) and another based on deterministic annealing (DA) [26]. We carefully tune the regularization parameter for linear SVMs (in the range $\gamma = \frac{c}{l}$ where $c = \{0.001, 0.01, 0.1, 1, 10, 100\}$ and l is the number of labeled examples) to optimize test performance. Therefore, their performance reported here is meant to represent the best possible results one can hope to obtain with a state-of-the-art purely supervised learner. We report the best performance of TSVM and DA over the parameter settings used in [26]. Furthermore, TSVM and DA require an accurate estimate of positive class fraction. In practical semi-supervised settings, a noisy estimate of this fraction is obtained from the labeled data. In our experimental setting, we confer an advantage to TSVM and DA by setting the positive class fraction to the true value (see Table 1).

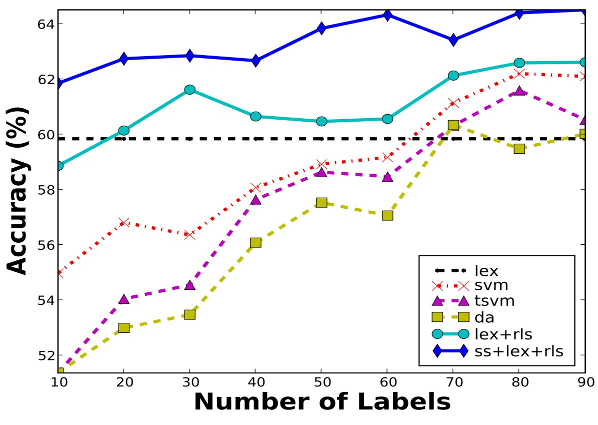
For SS+LEX+RLS, we need to set the following parameters: γ, μ, ν and p, the degree of the iterated graph Laplacian $M = \tilde{L}^p$. For all datasets, we used $p = 10$. We used $\gamma = 0.0001, \nu = 1.0, \mu = 10$ for MOVIES and $\gamma = 0.001, \nu = 0.1, \mu = 1$ for both POLITICAL and LOTUS. A careful optimization of these parameters may further improve the results presented here.

We generated learning curves averaged over 10 runs of 10-fold cross-validation. In the semi-supervised setting this experimental protocol needs more explanation. Let U be the set of truly unlabeled examples in the dataset, as we have in POLITICAL and LOTUS . Let L denote the labeled

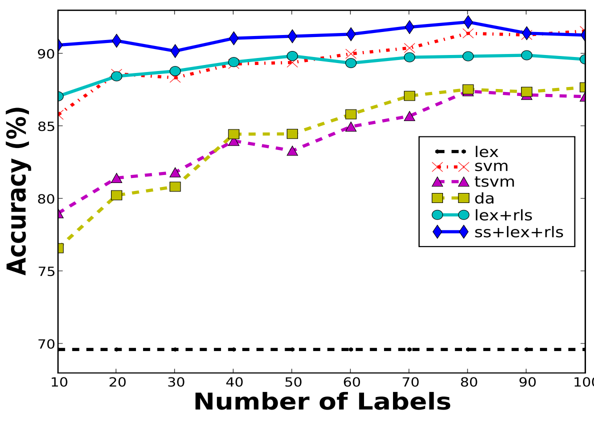
Figure 1. Learning Curves



(a) Performance on MOVIES



(b) Performance on POLITICAL



(c) Performance on LOTUS

set. In each of the 10 training-test splits in one run of 10-fold cross-validation, we partition L into L_{train} and L_{test} in the ratio 9:1. Next we take only a subset L_{lab} of L_{train} as labeled data and study the effect of gradually increasing the size of L_{lab}. Semi-supervised algorithms are provided $(L_{train} - L_{lab}) \cup U$ as the unlabeled set. Supervised algorithms only use L_{lab}. The linear models given by various algorithms are then evaluated on L_{test}. The resulting learning curves are shown in Figure 1.

It is clear from Figure 1 that by utilizing both lexical prior knowledge as well as unlabeled data SS+LEX+RLS significantly outperforms all competing alternatives on all datasets. As expected the smaller the labeled set, the larger the performance boost. On MOVIES, we see that with 50 labeled examples the SVM, TSVM and DA perform no better than the unsupervised lexical classifier. On the other hand, by simply combining these few labeled examples with lexical information, LEX+RLS already gives better performance. Finally, by further including unlabeled data SS+LEX+RLS gives by-far the best performance. Similar observations hold on LOTUS and POLITICAL. Surprisingly, on those datasets TSVM and DA turn out to perform worse than an SVM. Even if suboptimal local minima issues for TSVM and DA are kept aside, when labeled examples are extremely scarce the low-density separators found by these algorithms may not be sufficiently constrained. We conjecture that the blogosphere consists of clusters of bloggers focussing on similar sub-topics while the range of topics is very diverse (e.g., “iraq war” versus “health care“). This implies that without additional labeled data or prior knowledge such as what the lexicon provides, one may find good quality low-density decision boundaries that end up better separating topical sub-clusters as opposed to sentiment classes.

The unsupervised lexical classifier does not perform well, particularly on MOVIES and LOTUS. The underlying assumption of the Lexical Classifier is that a document is positive if there are more positive lexicon terms than negative terms in a document. Apart from the fact that the lexicon does not cover all terms that may appear in our vocabulary, it also does not capture domain-specific connotations of terms. The Lexical Classifier also fails to account for the degree of positive and negative sentiment associated with each term. We claim that semi-supervised learning can radically update our knowledge about the sentiment polarity of terms, beyond what can be captured by a limited labeled set. We can support this claim by examining the elements of our lexical background knowledge that have been altered by our semi-supervised model. Such insight can be easily gathered by comparing the sentiment polarity score $\boldsymbol{f}^w$ with the lexical labels $\mathbf{w}_{lex}$. Table 2 presents the top 20 lexicon terms for MOVIES sorted by $-w_{lex_i} f_i^w$ for a model trained with 400 labels and 1400 unlabeled examples; this set constitutes the terms that have changed most dramatically in sentiment. Also tabulated are the the corresponding weights learnt in the final linear model, the total number of documents in the entire dataset in which the words appears and the fraction p of those documents that have positive sentiment. This analysis gives us some insight into the domain-specificity of the sentimentality of certain terms, which is not possible to encode into a single general-purpose lexicon. For example, words such as revolution, capture and complex can be associated with positive experiences in descriptions of movies, though they may be generally considered negative in other contexts. The down-weighting of positive lexicon terms, such as talent for MOVIES is also consistent with the “thwarted expectation” narratives that Pang et al. [21] observed in this data. In Tables 3 and 4, we look at words not in the lexicon that recieve the highest positive and negative sentiment polarity scores as given by the magnitude of f_i^w. It is clear that the model correctly predicts positive sentiment around words like “shrek” and negative sentiment around words like “godzilla”. The sentiment labeling of such highly domain specific terms requires non-trivial subject matter expertise which emphasizes the need for models that can learn from partially specified lexicons.

Table 2. Stemmed lexical terms whose polarity is radically changed by our model on movies.

term	$\mathbf{w}_{lex}$	$\boldsymbol{f}^w$	$\mathbf{w}$	#docs	p
lone	-1	0.30,	0.31	81	0.70
origin	1	-0.29,	-0.11	524	0.49
basic	1	-0.27,	-0.12	280	0.46
show	-1	0.27,	0.03	807	0.52
revolut	-1	0.24,	0.27	17	0.82
pretti	1	-0.23,	0.13	395	0.42
know	1	-0.22,	0.02	911	0.50
reason	1	-0.22,	0.04	494	0.42
hatr	-1	0.22,	0.21	16	0.94
doubt	-1	0.22,	-0.02	184	0.59
captur	-1	0.21,	0.09	148	0.64
complet	1	-0.21,	-0.02	506	0.47
complex	-1	0.20,	0.08	146	0.70
talent	1	-0.20,	-0.08	367	0.41
upset	-1	0.19,	0.23	37	0.70
secur	1	-0.19,	-0.01	79	0.43
call	1	-0.19,	0.12	553	0.47
debat	-1	0.17,	0.22	31	0.84
critic	-1	0.16,	-0.06	263	0.57
plain	1	-0.15,	0.10	75	0.43

6 Conclusion

We have proposed a general semi-supervised learning algorithm based on joint regularization on documents and words. This model naturally incorporates lexical information as well as unlabeled data within standard regularized

Table 3. Stemmed terms not in the lexicon with highest positive sentiment polarities

term	$\boldsymbol{f}^w$	$\mathbf{w}$	#docs	p
shrek	1.99	1.18	5	1.00
donkei	1.36	1.05	5	1.00
life	1.33	0.14	827	0.60
mulan	1.29	0.69	14	0.93
homer	1.24	1.06	9	0.89
farquaad	1.21	0.80	5	1.00
fiona	1.19	0.86	10	0.80
truman	1.15	0.60	30	0.70
jacki	1.05	0.42	64	0.62
lithgow	0.97	0.96	9	1.00

Table 4. Stemmed terms not in the lexicon with highest negative sentiment polarities

term	$\boldsymbol{f}^w$	$\mathbf{w}$	#docs	p
horrend	-0.94	-1.07	15	0.27
suppos	-0.90	-0.51	322	0.32
prinz	-0.86	-0.77	20	0.15
ap	-0.86	-0.77	29	0.34
cop	-0.85	-0.46	173	0.40
batman	-0.84	-0.34	73	0.32
godzilla	-0.83	-0.48	30	0.33
werewolf	-0.82	-0.93	9	0.22
movi	-0.81	0.03	1652	0.49
thriller	-0.79	-0.39	230	0.42
freddi	-0.78	-0.76	31	0.23

least squares. We successfully applied this framework to the problem of sentiment prediction.

Our methods and applications can be immediately extended:

- *Labeled Features*: While we have demonstrated excellent empirical performance on sentiment classification tasks, our methods can be applied to a variety of classification problems where partial supervision may be available along both dimensions of the data matrix, in the form of row (i.e., labeled examples) and column labels (i.e., labeled features). To the best of our knowledge, our models are the first to allow dual semi-supervision in a regularization framework.

- *Other Loss functions*: While we have focussed on squared loss in this paper, our methods can be easily adapted to other loss functions, e.g., hinge loss and logistic loss, to implement SVM and logistic regression based models. We omit technical details due to lack of space.

- *Non-linear RKHS-based models*: We explored linear models which are appealing for high-dimensional text classification problems. However, our objective functions can also be minimized over a non-linear Reproducing Kernel Hilbert Space (RKHS) of functions. Details along this line are presented in [25].

- *Inter-document and Inter-word Linkages*: When building predictive models for web documents (e.g., blogs), the network structure induced by hyperlinks provides an independent source of information beyond what is already captured by the textual content. Under *assortivity* assumptions, i.e., that hyperlinks tend to connect documents with similar labels, it is natural to modify the document-word bipartite graph of Eqn. 5 to also include the web-graph.[2] In general one may also include a similarlity graph over words (e.g., with edges connecting synonyms, or related words from different languages in a multi-lingual corpora). Without any modifications to the algorithm, one may incorporate such network structures with the following definition,

$$\mathbf{A} = \begin{pmatrix} \mathcal{G}^g & \mathbf{X} \\ \mathbf{X}^\top & \mathcal{G}^w \end{pmatrix} \tag{11}$$

where $\mathcal{G}^d$ and $\mathcal{G}^w$ additionally encode document (examples) and word (features) similarity graphs. An alternative to combining document-document, word-word and document-word adjacency graphs is to directly linearly combine the associated graph Laplacians instead. Empirical studies along this direction are left for future work.

Acknowledgements

We thank Wojciech Gyrc, Sabrina Pei-yun Hsueh, Richard Lawrence, Yan Liu, Estepan Meliksetian and Claudia Perlich for their support and helpful conversations.

References

[1] M. Belkin, I. Matveeva, and P. Niyogi. Regularization and regression on large graphs. In *COLT*, 2004.

[2] Blogpulse: A service of nielsen buzzmetrics. http://www.blogpulse.com/.

[3] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, 2006.

[4] O. Chapelle, V. Sindhwani, and S. Keerthi. Branch and bound for semi-supervised support vector machines. In *NIPS*, volume 20, 2007.

[5] O. Chapelle and A. Zien. Semi-supervised classification by low density separation. In *AISTATS*, 2005.

[6] F. Chung, editor. *Spectral Graph Theory*. AMS, 1997.

[7] S. Das and M. Chen. Yahoo! for amazon: Extracting market sentiment from stock message boards. In *Proceedings of the 8th Asia Pacific Finance Association (APFA)*, 2001.

[2]In this case, one may construct undirected co-citation and/or bibliographic graphs from the initial web graph.

[8] A. Dayanik, D. D. Lewis, D. Madigan, V. Menkov, and A. Genkin. Constructing informative prior distributions from domain knowledge in text classification. In *SIGIR*, 2006.

[9] I. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *KDD*, 2001.

[10] G. Druck, G. Mann, and A. McCallum. Learning from labeled features using generalized expectation criteria. In *SIGIR*, 2008.

[11] K. T. Durant and M. D. Smith. *Lecture Notes in Computer Science: Advances in Web Mining and Web Usage Analysis*, volume 4811/2007, pages 187–206. Springer, 2007.

[12] Extracting the main content from a webpage. http://wshadow.com/blog/2008/01/25/extracting-the-main-content-from-a-webpage/.

[13] M. Hu and B. Liu. Mining and summarizing customer reviews. In *KDD*, pages 168–177, 2004.

[14] T. Joachims. Transductive inference for text classification using support vector machines. In *International Conference on Machine Learning*, 1999.

[15] T. Joachims. Transductive learning via spectral graph partitioning. In *International Conference on Machine Learning*, 2003.

[16] S. S. Keerthi and D. M. DeCoste. A modified finite Newton method for fast solution of large scale linear SVMs. *Journal of Machine Learning Research*, 6:341–361, 2005.

[17] S.-M. Kim and E. Hovy. Determining the sentiment of opinions. In *Proceedings of International Conference on Computational Linguistics*, 2004.

[18] B. Liu, X. Li, W. S. Lee, and P. Yu. Text classification by labeling words. In *AAAI*, 2004.

[19] V. Ng, S. Dasgupta, and S. M. N. Arifin. Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In *COLING & ACL*, 2006.

[20] B. Pang and L. Lee. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*, 2004.

[21] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *EMNLP*, 2002.

[22] G. Ramakrishnan, A. Jadhav, A. Joshi, S. Chakrabarti, and P. Bhattacharyya. Question answering via bayesian inference on lexical relations. In *ACL*, pages 1–10, 2003.

[23] R. Rifkin. Everything old is new again: a fresh look at historical approaches in machine learning. *Ph.D. Thesis, MIT*, 2002.

[24] R. E. Schapire, M. Rochery, M. G. Rahim, and N. Gupta. Incorporating prior knowledge into boosting. In *ICML*, 2002.

[25] V. Sindhwani, J. Hu, and A. Mojsilovic. Regularized coclustering with dual supervision. In *NIPS*, volume 21, 2008.

[26] V. Sindhwani and S. Keerthi. Large scale semi-supervised linear svms. In *SIGIR*, 2006.

[27] V. Sindhwani and D. Rosenberg. An rkhs for multi-view learning and manifold co-regularization. In *ICML*, 2008.

[28] A. Smola and I. Kondor. Kernels and regularization on graphs. In *COLT*, 2003.

[29] S. Spangler, Y. Chen, L. Proctor, A. Lelescu, A. Behal, B. He, T. Griffin, A. Liu, B. Wade, and T. Davis. COBRA-Mining Web for Corporate Brand and Reputation Analysis. *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, pages 11–17, 2007.

[30] P. Turney. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 417–424, 2002.

[31] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *HLT*, 2005.

[32] X. Wu and R. Srihari. Incorporating prior knowledge with weighted margin support vector machines. In *KDD*, 2004.

[33] P. Zhang and J. Peng. Svm vs regularized least squares classification. *ICPR*, 2004.

[34] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS*, volume 16, pages 321–328. MIT Press, 2004.

[35] L. Zhuang, F. Jing, and X.-Y. Zhu. Movie review mining and summarization. In *CIKM*, pages 43–50, 2006.