Learning Blog Sentiment with Reduced Supervision

Prem Melville, Yan Liu, Wojciech Gryc, Richard Lawrence, Claudia Perlich
IBM T.J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598
{pmelvil,liuya,wgryc,ricklawr,perlich}@us.ibm.com

ABSTRACT

With the recent growth of Web 2.0 and its emphasis on information sharing and user collaboration, marketing organizations are understandably nervous about the impact of blogs and forums on the perception of product brands and corporate reputation. Automated monitoring of such discussions introduces a number of challenges of interest to the KDD community, including the capability to detect sentiment associated with specific entities like product brands or company names. From a classification perspective, sentiment prediction is particularly challenging because we require an approach that is applicable across a range of domains, but there are typically very few labeled sentiment examples in any single target domain (e.g. enterprise software). Given this objective, we explore approaches for sentiment prediction with reduced supervision including (a) cross-domain training, that utilizes training labels from related domains, and (b) a new framework, Precocious Naïve Bayes, that utilizes background knowledge to achieve improved accuracy with few labeled examples. We compare the relative performance of these approaches for characterizing sentiment in posts drawn from two very different domains: a specific enterprise-software brand, and recent discussions about US Presidential candidates. Our results show that Precocious Naïve Bayes out-performs learning using only training examples, as well as purely lexicon-based approaches. We further demonstrate that cross-domain learning, in which classifiers are trained against labeled examples from related domains, can provide reasonable accuracy in classifying sentiment in the specific enterprise-software brand.

General Terms

Sentiment Analysis, Text Categorization, Lexical Learning, Machine Learning, Background Knowledge, Naive Bayes

1. INTRODUCTION

Over the past decade, the Internet has created enormous opportunities for companies of all sizes to reach customers, advertise products, and transact business. In this well established business model, companies can largely control their own web-based reputation via the content appearing on their websites. Web 2.0, with its emphasis on sites that promote information sharing and user collaboration, is quickly altering this landscape. Increasingly, the focal point of discussion of all aspects of a company's product portfolio is moving from individual company websites to collaboration sites, blogs, and forums where essentially anyone can post comments that may influence perceptions and purchase behavior of a large number of potential buyers. This is of obvious concern to marketing organizations, not only because the spread of negative information can be difficult to control, but because it can be very difficult to even detect it given the huge growth in blogs, forums, and other Web 2.0 phenomena. This concern has given rise to the term "buzz", and several sites (e.g. [5]) and systems (e.g. [29]) have been introduced to monitor trends in the blog-based reputation of specific companies and product brands.

The automated analysis of blog-related buzz raises several interesting questions from a marketing perspective:

- Given a huge number (order 100 million) of blogs, how can we identify the subset of blogs and forums that are discussing not only a specific product, but higher-level concepts that are in some way relevant to this product?
- Having identified this subset of relevant blogs, how do we identify the most authoritative or influential bloggers in this space?
- How do we detect and characterize specific sentiment expressed about an entity (e.g. product) mentioned in a blog or forum?

Each of these marketing questions raises interesting technical issues of interest to the KDD community, which we are pursuing as a part of a broader agenda.

In this paper, we focus only on the problem of sentiment detection. Our examples are drawn from two different domains: analysis of blog posts relevant to enterprise-software products, and detecting sentiment in political blogs. In the context of blog analysis, the objective of sentiment detection is to determine the overall attitude expressed in a portion of text contained within a blog. The text can be either an entire blog post, or a snippet extracted in the proximity of the mention of a specific entity such as a company or product. Classification methods, based on combinations of text mining and natural language processing, can be developed and

used to score each such example in terms of the extent of the positive sentiment expressed. Such scores can be converted to labels such as *positive*, *neutral*, and *negative*.

We are interested in the particular challenge of ultimately building a system that provides insights to market intelligence across many different products and topics of interest. This implies in particular, that for each such domain, we have to reproduce the analysis and cannot assume to have any significant amount of labeled data. In the best case we may be able to hand-label a very small set of blogs. We are therefore focusing on effective sentiment analysis with little or no training data from the target domain.

Given this limitation on training data, we explore increasing levels of availability of relevant information, how to combine them, and their relative empirical benefit. Section 2 contains a brief overview of the relevant literature. In Section 3, we consider an initial model that does not really learn, since it uses no labeled data and only linguistic background knowledge in the form of a sentiment-related lexicon. At the next level, in Section 4, we use labeled training data from more or less similar domains (cross-domain learning) including software and movie reviews. Then, in Section 5, we consider that *some* manually collected labels may be available for the blog domain of interest and investigate if we can improve the model by incorporating lexicon-based background knowledge in a Bayesian framework. Section 6 describes experimental results comparing these approaches against several test data sets drawn from both technology-based and political blogs.

2. RELATED WORK

2.1 Sentiment Analysis of Blogs and Posts

Sentiment analysis of bloggers and their posts is becoming a popular area of research within information retrieval. At a high level, sentiment analysis can refer to a number of different approaches to classifying blog posts. Researchers have explored mood classification [20] [3] of posts labeled with emotional data by posts' authors. Similarly in the realm of politics, "sentiment" can refer to categorizing posts along the political spectrum [10].

Our focus is labeling posts as exhibiting positive or negative opinions on pre-specified topics – Lotus software or candidates in the 2008 Democratic Primaries. In the past, researchers focusing on such classification have broken tasks down into subtasks, building models based on sub-classifiers looking for spam, relevance, and sentiment [21]. A related approach involves looking at sentiment and active/passive writing styles [14]. It has also been observed that, depending on the subject, accuracies can vary a great deal, with technical domains being easier to classify than social and cultural ones [30].

Such a diversity of tasks is beyond the scope of this paper, as we focus on labeling a post as positive or negative, already having filtered out irrelevant and objective posts. We specifically focus on exploring how one can improve the classification of posts as positive or negative, and leave exploring other tasks related to this field for future work.

2.2 Knowledge/Dictionary-based approaches

There have been a number of efforts to use background knowledge inspired by computational linguistics in the form of dictionaries and lexicons to assign sentiments to documents. One important component is the development of domain-specific sentiment lexicons; since, sentiment-laden words in one domain may not necessarily carry the same interpretation in a another domain. Initial work including [8] develop a domain-specific lexicon manually. Semi-automated approaches ([15, 35, 17, 33]) rely only on a small initial domain-specific seed (either manually selected or derived from initial statistical analysis using the class labels) and use WordNet's synsets and hierarchies to acquire synonyms and antonyms as additional opinion words. Finally, Turney [31] proposed a close to fully automated approach to domain-specific opinion word selection using mutual information to measure the distance of corpus words to only two seed words: "excellent" or "poor". Ng et al. [22] observe that most semi-automated approaches yield unsatisfactory lexicons with either high coverage and low precision or low coverage and high precision.

Once a dictionary is established, the simplest approach to sentiment classification is to count the number of occurrences of positive and negative words and assign the document to the majority class of the sentiment words. A purely dictionary-based approach has a number of potential weakness: 1) The performance is very much dependent on the relevance of the dictionary; 2) context-dependent meaning of words cannot be assessed; 3) and, all positive and all negative words are implicitly assumed to carry equal amount of sentiment. Finally, we also observed, that while the resulting sentiment-ranking can be fairly good, the classification accuracy often suffers from an inappropriate selection of classification threshold. This may be directly related to the precision-coverage trade-off observed by Ng.

2.3 Machine learning approaches

Pang et al. [26] 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 with supervised learning, as we propose in Section 5.

Pang et al. also compare three learning algorithms — Naïve Bayes, Maximum Entropy and SVMs. For their domain, SVMs perform marginally better than the other approaches. 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. [25] 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. Using a Naïve Bayes classifier coupled with forward feature selection they were able to outperform SVMs and achieve almost 90% accuracy on their task. In Section 5 we also use an approach that extends Naïve Bayes classification. Judging from Durant and Smith's results, it may be possible to improve our performance using the same feature selection technique; however, that study is beyond the scope of this paper.

3. LEXICAL CLASSIFICATION: MODELING WITH NO LABELS

In the absence of any labeled data in a domain, one can build sentiment-classification models that rely solely on background knowledge, such as a lexicon defining the polarity of words. Given a lexicon of positive and negative terms, one straightforward approach to using this information is to measure the frequency of occurrence of these terms in each document. The probability that a test document belongs to the positive class can then be computed as:

$$P(+|D) = \frac{a}{a+b}$$

Where a and b are the number of occurrences of positive and negative terms in the document respectively. A document is then classified as positive if $P(+|D) > t$, where t is the classification threshold; otherwise, the document is classified as negative. In the absence of any prior information of the relative positivity and negativity of terms we use $t = 0.5$, i.e. we assume a document is positive if there are more positive terms than negative in the document. We refer to this classification approach as the Lexical Classifier, and use it as one of our baselines.

For this study, we used a lexicon provided by the IBM India Research Labs that was developed for other text mining applications [28]. 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 to account for domain-specific connotations.

4. CROSS-DOMAIN LEARNING MODELING WITH LABELS FROM RELATED DOMAINS

Cross-domain learning refers to a learning process that trains a classifier (or a regression model) with abundant data in domain A and tests it on domain B. It is essential for many web applications because the web content is constantly changing and it is difficult to get labeled data for each domain in a timely fashion. For example, in our task of blog sentiment prediction, even though we are able to label enough examples for one particular product, it is infeasible to get the labels for thousands of other products.

Cross-domain learning is closely related to transfer learning, concept drift and sample bias correction. Transfer learning corresponds to transferring knowledge learned from the old data to new situations or new tasks. Recent research on transfer learning mostly focuses on multi-task learning [6, 4, 2], which tackles the problem that the decision boundaries for task A and B are not in exactly the same places in the feature space, even when the feature spaces and input distributions are themselves identical. Concept drift in stream mining means that the statistical properties of the target variable, which the model is trying to predict, change over time. These changing properties might be the class prior $P(y)$, the sample distribution $P(x|y)$, the decision function $P(y|x)$ or maybe a combination of all. Sample bias correction is mostly concerned with distinct training distribution $P(x|\lambda)$ and testing distribution $P(x|\theta)$ with unknown parameters λ and θ.

In our task, the cross-domain learning, i.e. training on domain A and testing on domain B (without leveraging any label information from B), can be seen as a special case of dealing with concept drift except that there is no time-sensitive information in the data. In contrast, we do have abundant labeled data from several other domains and background knowledge to help the learning. Given a set of training data from M domains $\{(x_i^{(1)}, y_i^{(1)})\}$, $\{(x_i^{(2)}, y_i^{(2)})\}$, ..., $\{(x_i^{(M)}, y_i^{(M)})\}$, we can try four simple approaches:

1. Train one classifier $f^{(i)}$ on each domain and select the best one
2. Train one classifier f^{all} on a combined set from all domains
3. Build an ensemble classifier $f^e = \frac{1}{K}\sum_{k=1}^{K} f^{(k)}$
4. Use the background knowledge for variable selection and build a classifier only on the selected variables

Given the fact that we do not have any information from the testing set to leverage, the only important decision left is which base classifier to use. For cross-domain learning of blog sentiment, the major challenge is that some sentiment words used in domain A do not appear in domain B or vice versa. In previous work on concept drift, both generative models and discriminative models have been analyzed [32, 13]. For the generative model, we have posterior probability $P(y|x) \propto P(x|y)P(y)$. The error rate on the testing set relies on two factors: (1) shifting class distribution, i.e. the class prior $P(y)$ may change over different datasets; (2) shifting feature distribution: the density of each class $P(x|y)$ may change. From a practical point of view, the density estimation of the singleton words is challenging in generative models, such as Naïve Bayes. For the discriminative models, we can limit our discussion to a simple case where the conditional probability takes on a log-linear form, i.e. $\log P(y|x) \propto \sum_{k=1}^{K} w_k x_k$, where w_k is the feature weight that varies over time (or domains). Common priors can be easily applied to adjust the weight for singletons and solve the problem. Furthermore, it has been shown that discriminative classifiers that model $P(y|x)$ directly are not affected by sample selection bias[34]. Therefore we choose to use discriminative classifiers in our cross domain-learning (although there is no conclusive theoretical support).

5. PRECOCIOUS NAÏVE BAYES: MODELING WITH FEW LABELS

In this section we discuss how background knowledge, such as a lexicon, can be combined with few labeled examples to build a better sentiment classifier. For the purpose of this

paper we will focus on binary polarity classification of positive and negative sentiment. However, our methods can easily be extended to a multi-class setting, to incorporate other classes such as *neutral* or *objective*. In keeping with the text categorization literature we will use "document" interchangeably with "blog post".

Previous studies have shown that using only lexical information is not as effective for sentiment classification as building machine learning models from training examples [26]. However, completely ignoring the background knowledge provided by a lexicon, in lieu of training data, may not be optimal. As an alternative, we propose the Precocious Naïve Bayes classifier that provides a framework in which one can incorporate background lexical information, and refine this information for a specific domain using any available training examples.

Below, we present some basic concepts on multinomial Naïve Bayes text classification, required for the derivation of our framework. More details on event models for text classification and induction of Naïve Bayes classifiers can be found in [19].

The multinomial Naïve Bayes classifier commonly used for text categorization relies on three assumptions [23]: (1) documents are produced by a mixture model; (2) there is a one-to-one correspondence between each mixture component and class, and (3) each mixture component is a multinomial distribution of words, i.e., given a class, the words in a document are produced independently of each other.

Based on our generative model, the likelihood of a document is the sum of total probability over all (2 in our case) mixture components, i.e.,

$$P(D) = \sum_j P(D|c_j)P(c_j)$$

Where $P(c_j)$ is the probability of the class c_j, and $P(D|c_j)$ is the probability of the document given the class. We refer to the set of classes as $\mathcal{C}$, which for binary sentiment classification is $\{+, -\}$.

In computing the likelihood of a document, we make the naive assumption that the words (w_i) of a document are generated independently, so the probability of a document, D, being generated in a given class, c_j, is

$$P(D|c_j) = \prod_i P(w_i|c_j)$$

The Naïve Bayes classification rule uses Bayes' theorem to compute the class membership probabilities of each class,

$$P(c_j|D) = \frac{P(c_j)\prod_i P(w_i|c_j)}{P(D)} \quad (1)$$

and the class with the highest likelihood is predicted, i.e.,

$$argmax_{c_j} P(c_j)\prod_i P(w_i|c_j) \quad (2)$$

5.1 Initialization

Inducing a multinomial Naïve Bayes classifier involves estimating the model parameters $P(c_j)$ and $P(w_i|c_j)$. The idea behind Precocious Naïve Bayes is to initialize these parameters using the background knowledge provided by the lexicon. In the absence of any prior information about the class distribution we assume uniform class priors, so we will ignore the class priors $P(c_j)$ for the derivations below. We use the lexicon only to initialize the conditional probabilities of each word given the class. The exact values used to initialize these conditionals are derived below, based on a set of properties we would like these distributions to have. To aid our derivations, first we establish some important notation below.

Definitions:
$\mathcal{V}$ – the vocabulary, i.e., set of words in our domain
$\mathcal{P}$ – set of positive terms from the lexicon that exists in $\mathcal{V}$
$\mathcal{N}$ – set of negative terms from the lexicon that exists in $\mathcal{V}$
$\mathcal{U}$ – set of unknown terms, i.e. $\mathcal{V} - (\mathcal{N} \cup \mathcal{P})$
m – size of vocabulary, i.e. $|\mathcal{V}|$
p – number of positive terms, i.e. $|\mathcal{P}|$
n – number of negative terms, i.e. $|\mathcal{N}|$

Property 1: Since we do not know the relative polarity of terms in the dictionary, we assume all positive terms are equally likely to occur in a positive document, and the same is true for negative documents, i.e.,

$$\begin{aligned} P(w_i|+) &= P(w_j|+), \forall w_i, w_j \in \mathcal{P} \\ \text{and } P(w_i|-) &= P(w_j|-), \forall w_i, w_j \in \mathcal{N} \end{aligned} \quad (3)$$

We refer to the probability of any positive term appearing in a positive document simply as $P(w_+|+)$. Similarly, we refer to the probability of any negative term appearing in a negative document as $P(w_-|-)$.

Furthermore, in the absence of any knowledge about words that are not in our lexicon, we treat them equally in each class, i.e.,

$$\begin{aligned} P(w_i|+) &= P(w_j|+), \forall w_i, w_j \in \mathcal{U} \\ \text{and } P(w_i|-) &= P(w_j|-), \forall w_i, w_j \in \mathcal{U} \end{aligned} \quad (4)$$

We refer to these conditional probabilities of non-lexicon terms as $P(w_u|+)$ and $P(w_u|-)$.

Property 2: If a document D_i has α positive terms and β negative terms, and document D_j has β positive terms and α negative terms, we would like the D_i to be considered as likely to be a positive document, as D_j is likely to be a negative document. Using (1) this property of symmetry gives us the following requirement:

$$\begin{aligned} [P(w_+|+)]^\alpha [P(w_-|+)]^\beta &= [P(w_-|-)]^\alpha [P(w_+|-)]^\beta \\ \Rightarrow \left[\frac{P(w_+|+)}{P(w_-|-)}\right]^\alpha &= \left[\frac{P(w_+|-)}{P(w_-|+)}\right]^\beta \end{aligned}$$

For this to true for all values of α and β, we need

$$\begin{aligned} P(w_+|+) &= P(w_-|-) \\ \text{and } P(w_-|+) &= P(w_+|-) \end{aligned} \quad (5)$$

That is, the probability of a positive term appearing in a positive document is the same as that of a negative term appearing in a negative document; and the same is true for the conditional probabilities of terms occurring in documents with opposite polarity.

Property 3: Since a positive document is more likely to contain a positive term than a negative term, and vice versa,

we would like:

$$P(w_+|+) = r \times P(w_-|+)$$
$$\text{and } P(w_-|-) = r \times P(w_+|-)$$
$$\text{where } 0 < 1/r \leq 1 \quad (6)$$

We refer to the factor r as the *polarity level* — it is a measure of how much more likely it is for a positive term to occur in a positive document compared to a negative term.

Property 4: Since each component of our mixture model is a probability distribution, we have the following constraint on the conditional probabilities for each class, c_j:

$$\sum_i^m P(w_i|c_j) = 1 \quad (7)$$

Using the above properties as constraints, we can now derive the appropriate values to use for initializing our model.

From (7) it follows that

$$pP(w_+|+) + nP(w_-|+) + (m-p-n)P(w_u|+) = 1 \quad (8)$$

Which gives us the following inequality

$$pP(w_+|+) + nP(w_-|+) \leq 1$$
$$\Rightarrow pP(w_+|+) + n\frac{P(w_+|+)}{r} \leq 1$$

using (6).

Since $0 < 1/r \leq 1$, it follows that,

$$P(w_+|+) \leq \frac{1}{p+n}$$

Since we want to assign the maximum probability mass to the known terms in the lexicon, we set $P(w_+|+)$ to the maximum value possible, i.e.

$$P(w_+|+) = \frac{1}{p+n} \quad (9)$$

Now, it follows from (5) and (6) that

$$P(w_-|-) = \frac{1}{p+n}$$
$$P(w_+|-) = P(w_-|+) = \frac{1}{p+n} \times \frac{1}{r} \quad (10)$$

Now, solving for (8) (and analogously for the negative class), we get the following conditionals for the unknown terms:

$$P(w_u|+) = \frac{n(1-1/r)}{(p+n)(m-p-n)}$$
$$P(w_u|-) = \frac{p(1-1/r)}{(p+n)(m-p-n)} \quad (11)$$

Using (9), (10) and (11), we now have all the requisite parameters to initialize our Precocious Naïve Bayes model.

5.2 Induction

We refer to the conditional probabilities initialized using background knowledge, as described above, as $P_B(w_i|c_j)$. In the absence of labeled training data, these conditional distributions can be used to classify test documents using the classification rule (2), assuming uniform class priors.

Now, given a set of labeled documents we can begin to improve the model built on background knowledge by recomputing conditionals based on the background distribution and the training documents:

$$P(w_i|c_j) = \frac{t_{ij} + \omega P_B(w_i|c_j)}{\sum_i (t_{ij} + \omega P_B(w_i|c_j))} \quad (12)$$

Where ω is the *background-knowledge weight*, expressing our confidence in the lexicon — it can be thought of in terms of the number of training examples to which the background knowledge is equivalent. In the Naïve Bayes formulation, t_{ij} is the total number of occurrences of the word w_i in all documents belonging to class c_j. In practice, it is better to control for differing document sizes by normalizing each word vector representing each document, using the L2 norm. As such, we use these normalized word frequencies in our formulation.

The class prior probabilities are computed as the proportion of the total number of documents that belong to each class. As in traditional Naïve Bayes [24], we use Laplace smoothing to avoid assigning zero-probabilities to classes not represented in the training data:

$$P(c_j) = \frac{|D_j| + 1}{\sum_j |D_j| + |\mathcal{C}|}$$

Where D_j is the subset of documents in class j, and $|\mathcal{C}|$ is the total number of classes.

6. EXPERIMENTAL RESULTS

In this section we describe the data sets used in our study, present experimental results, and discuss the challenges of sentiment classification in the selected domains.

6.1 Data sets

As discussed earlier, the objective of this work is to automate the analysis of blog posts as they relate to product and/or brand names. For this purpose, we have created a set of labeled sentiment examples related to IBM Lotus software brand. While we are primarily interested in the sentiment of technology blogs, we have also applied our methodology to characterize sentiment in blogs discussing specific political candidates. For the transfer learning setting, we also require labeled examples from other domains. For this purpose, we have collected data from online reviews of Movies, Enterprise Software, and Personal Software. Table 1 provides a summary of the properties of the different datasets, which we discuss in more detail below. The size in Table 1 refers to the number of labeled examples (*positive* or *negative*). We manually labeled the Lotus and Political Candidate posts, while labels for the other datasets are inferred from user ratings.

Table 1: Data set properties

Domain	Name	Size	Positive Rate
Lotus Brand	*Lotus*	145	0.33
Political Candidates	*Political*	107	0.46
Movie Reviews	*Movie*	2000	0.50
Enterprise Software	*Epinions*	72	0.71
Personal Software	*Amazon*	11727	0.48

6.1.1 Blog domains

One non-trivial part of blog data collection for sentiment analysis is the extraction of the *relevant* text from the downloaded website. Blogs are by nature much more diverse in layout and structure than movie or product reviews. In addition, many blogs have significant numbers of comments (often from individuals other than the blogger) as well as explicit citations. Both comments and citation often exhibit the exact opposite sentiment from the main content. However, the automated distinction between core content, citations, and comments is very difficult. We are currently using 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: Our initial focus in this work was detecting sentiment around enterprise software, specifically IBM Lotus collaborative software. The Lotus data set 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. 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.

It is important to note that classification tends to be more accurate as a smaller number of individual bloggers is used because the diversity of writing styles and topics is much smaller than sample of posts. For example, a hypothetical blog with 25 posts on software bugs in Lotus is easier to classify than 25 negative posts on disparate topics like user interfaces, sales data, and other grievances.

Political Candidate blogs: Posts focusing on politics were taken from a continuously updated set of 11,788 political blogs which, at the time of writing, contained 317,566 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 labellers, 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 Political data set consisted of 49 positive and 58 negative posts.

6.1.2 Review domains

In addition to the commonly used movie review domain, we collected software reviews from two well-known sites: personal software from Amazon and enterprise software from Epinions. We discretized the ratings of the reviews and labeled reviews with four or more stars as positive, two or less stars as negative and discarded the neutral area of 3 stars.

Movies: We used the 2000 labeled movie reviews from [26], consisting of 1000 positive and 1000 negative reviews. Positive labels were assigned to ratings above 3.5 stars and negative to two and fewer stars.

Enterprise Software: The review domain that is closest to our main application domain, *Lotus*, is the enterprise software section from Epinions. We collected all available reviews from the last 5 years within the three enterprise software categories "Applications", "Databases" and "Server" at http://www.epinions.com/ensw. After removing the neutral reviews with 3 stars we obtain a set of 51 positive and 21 negative reviews.

Personal Software: Given the very limited number of enterprise software reviews at Epinions, we decided to collect additional software reviews for a large variety of personal software from Amazon. The only constraint we imposed was that the price of software had to be at least $50.

6.2 Results on learning with few labels

We compare our approach, Precocious Naïve Bayes to the following baselines: (1) the Lexical Classifier, (2) using only the background knowledge in Precocious Naïve Bayes– referred to as Only Background, and (3) only using the training data as in a Naïve Bayes model – referred to as Only Training. For the Naïve Bayes model using only training data, we compute the conditional probability estimates us-

ing Lidstone smoothing [18] with $\epsilon = 1 \times 10^{-6}$, i.e.,

$$P(w_i|c_j) = \frac{t_{ij} + \epsilon}{\sum_i t_{ij} + \epsilon|\mathcal{V}|} \quad (13)$$

This smoothing accounts for infrequent words that do not appear in the training set, but appear in the test set. Using $\epsilon < 1$ in Lidstone smoothing for word probabilities works better for text classification than using Laplace smoothing ($\epsilon = 1$) [1].

For Precocious Naïve Bayes, we set the *polarity level*, $r = 100$ — i.e., a positive word in the lexicon is consider 100 times more likely to appear in a positive document than a negative term. We also set the *background weight*, $\omega = 100$ — i.e., the background knowledge is considered to be equivalent to having 100 labeled examples. These parameters were set based on intuition; however, the relative performance of Precocious Naïve Bayes is not dramatically affected by small changes in these values.

We compare the different sentiment classification approaches on the *Lotus* and *Politics* datasets described in Section 6.1. We pre-processed the data by filtering out stop-words and stemming words using the Porter stemmer [27]. We also eliminated infrequent words that appeared in less that 3 blog posts. In order to evaluate the effect of varying amounts of training data, we generate learning curves averaged over multiple runs of 10-fold cross-validation. For *Lotus*, we used 10 runs of cross-validation; and for the more noisy *Politics* set we used 20 runs. The resulting learning curves are presented in Fig. 1, and the accuracies at the last point are summarized in Table 2.

Table 2: Comparing accuracy of different approaches to sentiment classification.

Model	*Lotus*	*Politics*
Lexical Classifier	68.23	55.20
Only Background	84.84	57.05
Only Training	88.40	59.24
Precocious Naïve Bayes	91.70	64.19

The results clearly demonstrate that combining background knowledge with training examples via a Precocious Naïve Bayes framework performs better than using each source of information in isolation. This improvement in accuracy produced by Precocious Naïve Bayes is statistically significant compared to all other methods according to paired t-tests ($p < 0.05$).

The learning curves show that, as expected, with very few training examples it is better to rely on only the background knowledge, than trying to estimate model parameters from a limited set of labels. However, with increasing amounts of labeled data, one can start building Only Training models that are better that Only Background. For our domains, with less than 60 labeled examples it is possible to train a Naïve Bayes model that is better that using only background knowledge. However, combining both background knowledge with training data as in Precocious Naïve Bayes is always a better alternative.

Using only the lexicon through the Lexical Classifier does not perform very well on our data sets. For *Politics* its accuracy is only a little better than the base rate of 54%, whereas for *Lotus* its accuracy is below the base rate of 77%, Interestingly, using only background knowledge in the Precocious Naïve Bayes framework (Only Background) performs better than the Lexical Classifier in both domains. This may be explained by the fact that the Precocious Naïve Bayes assumes that the documents are generated from a mixture of positive and negative sentiment multinomial distributions. The assumptions of this model appear to hold well for these data sets, which may account for the better performance of a Naïve Bayes classification that relies on conditionals initialized using the same lexicon as used by the Lexical Classifier. The difference between these approaches of using the lexicon is more dramatic for *Lotus* than for *Politics*. This seems consistent with the fact that the assumptions of our generative model are more strongly held for the *Lotus* domain as opposed to *Politics*, as discussed below.

The *Lotus* posts originate from a small set of blogs, and there is almost a one-to-one correspondence between blogs and sentiment, i.e., each blogger either loves or hates *Lotus*. This conforms better with the assumptions of our model compared to the *Politics* posts – that originate from many disparate sources that have positive and negative commentary about different electoral candidates. As a result, our methods perform better at classifying sentiment in the *Lotus* blogs compared to the *Politics* blogs.

6.3 Results on cross-domain training

In the cross-domain training experiments, we use Support Vector Machines with linear kernels as the base classifier, relying on the SVM^{light} package [16]. We focus our analysis on exploiting related domains for classifying *Lotus* blog-posts. We would expect to see qualitatively similar results for *Politics*.

Table 3 shows the accuracy of cross-validation (if training and test sets are the same) and cross-domain training (if training and test sets are different) on Movies, Epinions and Amazon respectively. From the results, we can see that (1) as expected the best accuracy is achieved when we train the classifier with examples coming from the same distribution (or domain) as the test set; (2) the Movies examples are the easiest to classify, followed by Amazon, and Epinions is the most difficult; (3) in terms of the effectiveness of cross-domain learning, the Movies set seems to be the most informative about general sentiment.

Table 3: Comparing accuracy of cross-domain training on Movies, Epinions and Amazon

Training Set	Test Set		
	Movies	*Epinions*	*Amazon*
Movies	81.5	65.9	60.1
Epinions	33.7	67.5	34.9
Amazon	67.2	60.9	73.8

Table 4 shows the accuracy on the target *Lotus* set using different cross-domain training methods. The cross-validation accuracy on the *Lotus* set is 93.8% (using SVM with 500 features selected via the Chi-squared scores). From the results, we can see that (1) among the three data sets from other domains, the Movies set achieves the highest accuracy; (2) a combined set of three domains is able to outperform each individual set and the ensemble approach

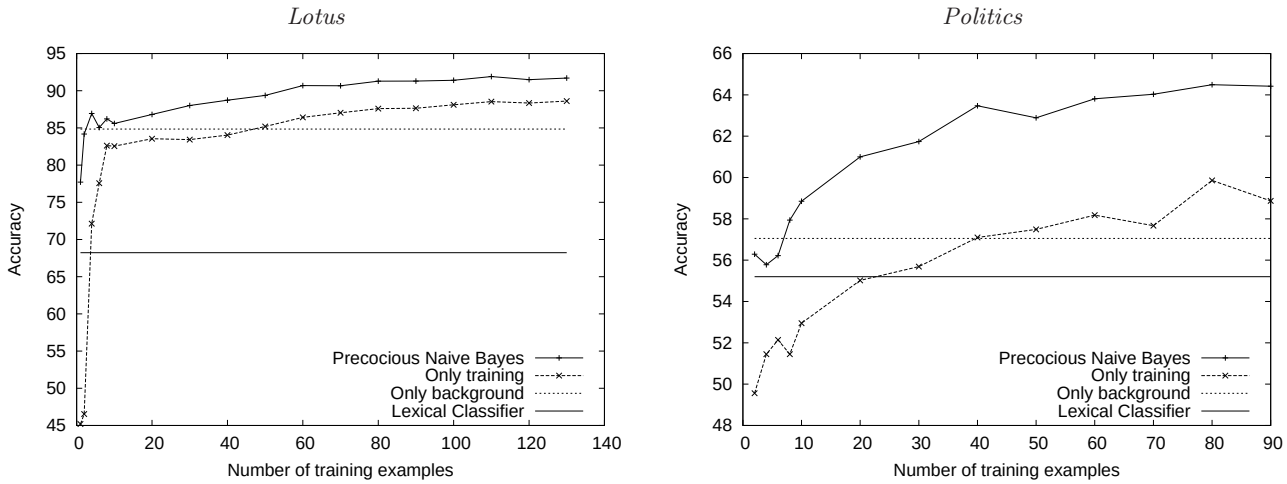

Figure 1: Comparing different approaches to sentiment classification.

performs better than training one single classifier with the combined set; (3) different feature selection criteria seem to affect the results significantly. We examine the top-ranked features based on chi-squared scores as shown in Table 5 and observe that the features selected with the *Movie* set and combined-set have the largest overlaps with the target *Lotus* set. In summary, we are able to achieve satisfactory results for blog sentiment prediction via cross-domain training although there is still room for improvement.

Table 4: Comparing accuracy of cross-domain learning with different training sets and feature selection methods on the Lotus set. Self-FS: feature selection based on the training set; Combined-FS: feature selection based on the combined set; Lexicon-FS; feature selection using the sentiment vocabulary.

Method	Training Set			
	Movies	*Epinions*	*Amazon*	Combined
Self - FS	79.3	50.3	76.6	86.9
Combined-FS	82.8	40.0	80.7	86.9
Lexicon-FS	85.5	33.0	78.6	84.3
Ensemble	N/A	N/A	N/A	87.8

Given the better results from the combined set (compared to individual domains), an interesting question to investigate is: where does the improvement come from? Table 4 seems to suggest that feature selection on a combined set helps to find informative sentiment words. Figure 2 plots the learning curve of different training sets on Lotus blogs. It can be seen that (1) given the same number of training examples, the combined set does not always yield the best performance; (2) the accuracy of the combined set saturates after 5000 or more examples are provided. Therefore the improvement seems to come from a better feature representation of the sentiment concept rather than more training examples, yet more thorough analysis is needed for any conclusions.

Table 5: Top ranked features in different data sets

Movies	bad, waste, worst, stupid, bore, perfect, portray, suppose, ridiculous, lame, great, awful, poor, outstanding
Amazon	easy , support, intuitive, great. try, waste, uninstall, custom, tech, money, buy, reinstal, fix, worst, error, bug, mess, call, excellent, best
Epinions	stalk, slmail, reinstal, card, webmail, unfortunate, query, winproxy, worm, penetrate, danger, proxy, pcanywhes
Combined set	easy, support, great, intuitive, waste, worst, uninstall, fix, money, reinstall, excel, error
Lotus	error, mess, busy, update, partner, great, help, late, fast, good

6.4 Challenges in Determining Sentiment of Blog Posts

Labeling blog posts as positive or negative is very complex, even for humans. While efforts were made to focus solely on the post content when categorizing posts, it is still the case that comments, citations, and quotes from other sources are included in the main body of a post. When a blog post's page becomes an area of discussion on a certain subject, labeling the entire page as positive or negative can be quite difficult. This is especially true for political blogs, where writers often make comparisons between multiple candidates, policies, or events. The issue of sentiment analysis is further complicated by the fact that bloggers often use jokes, anecdotes, and other cultural references to illustrate their opinions, making the labeling task unclear for people unfamiliar with the relevant facts or references. This makes sentiment classification extremely difficult for most algorithms.

In related work, it has been observed that sentiment classifiers tend to perform better on more technical subjects than social or creative ones. For example, Turney [31] sees accura-

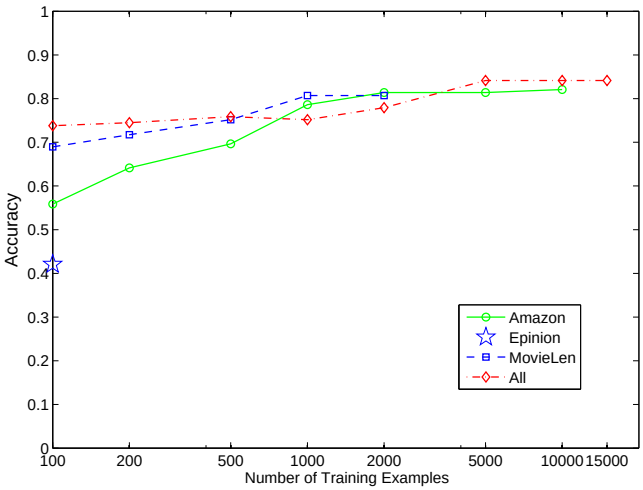

Figure 2: The learning curve of using different training set (Amazon, Epinion, Movies and a combined set of all) on the lotus blog test set

cies of 84% and 80% for reviews on automobiles and banks, respectively, and the same methods provide accuracies of only 66% for movies and 71% for travel destinations. Similarly, in our own results, political post classification tends to do far worse than labeling Lotus-focused posts. What is useful to note, however, is that our proposed methods still yield improved results on both domains.

7. FUTURE WORK

In this paper we explored two directions of dealing with little or no training data in a new application domain. In particular, Precocious Naïve Bayes looks at combining background knowledge with little training data from the application domain; while, cross-domain learning considers learning only from labeled data in related domains. One clear direction for future work is to explore a combination of these approaches, i.e., building better classifiers that exploit both background knowledge and alternative sources of labeled data. There has been a flurry of recent work in the area of transfer learning that could be applied to extend a background knowledge-based model to incorporate data from different domains. The fundamental challenge in such transfer learning is accounting for the training and test sets being from different distributions. Dai et al. [7] provide a solution for transfer learning in text classification using an EM-based Naïve Bayes classifier. Their solution first estimates the initial probabilities under a distribution $\mathcal{D}_l$ of the labeled data, and then uses the EM [9] algorithm to revise the model for the test distribution $\mathcal{D}_u$, using unlabeled instances from the test set. This Naïve Bayes Transfer classifier can easily replace the Naïve Bayes classifier that is at the base of Precocious Naïve Bayes, thus combining both background knowledge and transfer learning into a single framework.

8. CONCLUSION

Blog sentiment prediction is an important task in social network analysis and other Web 2.0 applications. Rooted in linguistics and psychology, sentiment prediction is a challenging task for machine learning, especially when our focus domain is constantly changing and there are very few or no labeled data in the target domain. In this paper, we explore two approaches to solving this problem — cross-domain training in the absence of target-domain data, and combining background knowledge with few training examples. We demonstrate that cross-domain learning, in which classifiers are trained against labeled reviews of movies, general enterprise software, and consumer software, can provide reasonable accuracy in classifying sentiment in a specific enterprise-software brand. Furthermore, we demonstrate that when provided with a few training examples in our target domain, we can combine background lexical information with supervised learning in a Precocious Naïve Bayes framework, to produce better results than using a lexicon or the training data in isolation. For future work, we are investigating the generality of Precocious Naïve Bayes and combining it with transfer learning.

9. ACKNOWLEDGMENTS

We would like to thank Vikas Sindhwani, Scott Spangler, and Ying Chen for insightful discussions.

10. REFERENCES

[1] R. Agrawal, R. J. B. Jr., and R. Srikant. Athena: Mining-based interactive management of text databases. In *Extending Database Technology*, pages 365–379, 2000.

[2] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

[3] K. Balog and M. de Rijke. Decomposing bloggers' moods. In *Proceedings of the WWW-2006 Workshop on the Weblogging Ecosystem*, 2006.

[4] J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

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

[6] R. Caruana. Multitask learning, 1997.

[7] W. Dai, G.-R. Xue, Q. Yang, and Y. Yu. Transferring naive bayes classifiers for text classification. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI'07)*, pages 540–545, 2007.

[8] 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.

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[10] K. Durant and M. Smith. Mining Sentiment Classification from Political Web Logs. In *Proceedings of the KDD Workshop on Web Mining and Web Usage Analysis*, 2006.

[11] K. T. Durant and M. D. Smith. *Lecture Notes in Computer Science: Advances in Web Mining and Web Usage Analysis*, volume Volume 4811/2007, chapter Predicting the Political Sentiment of Web Log Posts Using Supervised Machine Learning Techniques Coupled with Feature Selection, pages 187–206. Springer, 2007.

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

[13] G. Forman. Tackling concept drift by temporal inductive transfer. In *Proceedings of the 29th annual international conference on Research and development in information retrieval (SIGIR)*, pages 252–259, 2006.

[14] M. Genereux and R. Evans. Towards a validated model for affective classification of texts. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text*, pages 55–62, 2006.

[15] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the 10th international conference on Knowledge discovery and data mining (KDD)*, pages 168–177, 2004.

[16] T. Joachims. Making large-scale support vector machine learning practical. In A. S. B. Schölkopf, C. Burges, editor, *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA, 1998.

[17] S.-M. Kim and E. Hovy. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics (COLING)*, 2004.

[18] B. Liu. *Web Data Mining*. Springer, 2007.

[19] A. McCallum and K. Nigam. A comparison of event models for naive Bayes text classification. In *Proceedings of the AAAI-98 Workshop on Text Categorization*, pages 41–48, 1998.

[20] G. Mishne. Experiments with mood classification in blog posts. In *Proceedings of Style2005–1st Workshop on Stylistic Analysis of Text for Information Access at SIGIR*, 2005.

[21] G. Mishne. Multiple ranking strategies for opinion retrieval in blogs. In *Proceedings of the Fifteenth Text Retrieval Conference (TREC)*, 2006.

[22] 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 *Proceedings of the 21st International Conference on Computational Linguistics (COLIN)and 44th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2006.

[23] K. Nigam. *Using Unlabeled Data to Improve Text Classification*. PhD thesis, Carnegie Mellon University, 2001.

[24] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39:103–134, 2000.

[25] B. Pang and L. Lee. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (ACL)*, 2004.

[26] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2002.

[27] M. F. Porter. *An algorithm for suffix stripping*, pages 313–316. Morgan Kaufmann Publishers Inc., 1997.

[28] G. Ramakrishnan, A. Jadhav, A. Joshi, S. Chakrabarti, and P. Bhattacharyya. Question answering via bayesian inference on lexical relations. In *Proceedings of the ACL 2003 workshop on Multilingual summarization and question answering*, pages 1–10, 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. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 417–424, 2002.

[31] 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.

[32] H. Wang, W. Fan, P. Yun, and J. Han. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the 9th international conference on Knowledge discovery and data mining (KDD)*, 2003.

[33] H. Yu and V. Hatzivassiloglou. Towards answering opinion questions: separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 129–136, 2003.

[34] B. Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning (ICML)*, page 114, 2004.

[35] L. Zhuang, F. Jing, and X.-Y. Zhu. Movie review mining and summarization. In *Proceedings of the 15th ACM international conference on Information and knowledge management (CIKM)*, pages 43–50, 2006.