The Bag-of-Opinions Method for Review Rating Prediction from Sparse Text Patterns

Lizhen Qu
Max-Planck Institute for Informatics
lqu@mpii.mpg.de

Georgiana Ifrim
Bioinformatics Research Centre
ifrim@birc.au.dk

Gerhard Weikum
Max-Planck Institute for Informatics
weikum@mpii.mpg.de

Abstract

The problem addressed in this paper is to predict a user's numeric rating in a product review from the text of the review. Unigram and n-gram representations of text are common choices in opinion mining. However, unigrams cannot capture important expressions like "could have been better", which are essential for prediction models of ratings. N-grams of words, on the other hand, capture such phrases, but typically occur too sparsely in the training set and thus fail to yield robust predictors. This paper overcomes the limitations of these two models, by introducing a novel kind of bag-of-opinions representation, where an opinion, within a review, consists of three components: a root word, a set of modifier words from the same sentence, and one or more negation words. Each opinion is assigned a numeric score which is learned, by ridge regression, from a large, domain-independent corpus of reviews. For the actual test case of a domain-dependent review, the review's rating is predicted by aggregating the scores of all opinions in the review and combining it with a domain-dependent unigram model. The paper presents a constrained ridge regression algorithm for learning opinion scores. Experiments show that the bag-of-opinions method outperforms prior state-of-the-art techniques for review rating prediction.

1 Introduction

1.1 Motivation

Opinion mining and sentiment analysis has become a hot research area (Pang and Lee, 2008). There is ample work on analyzing the sentiments of online-review communities where users comment on products (movies, books, consumer electronics, etc.), implicitly expressing their *opinion polarities* (positive, negative, neutral), and also provide numeric *ratings* of products (Titov and McDonald, 2008b; Lerman et al., 2009; Hu and Liu, 2004; Titov and McDonald, 2008a; Pang and Lee, 2005; Popescu and Etzioni, 2005a). Although ratings are more informative than polarities, most prior work focused on classifying text fragments (phrases, sentences, entire reviews) by polarity. However, a product receiving mostly 5-star reviews exhibits better customer purchase behavior compared to a product with mostly 4-star reviews. In this paper we address the learning and prediction of numerical ratings from review texts, and we model this as a metric *regression* problem over an appropriately defined feature space.

Formally, the input is a set of rated documents (i.e., reviews), $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $\mathbf{x}_i$ is a sequence of word-level unigrams $(w_1, ..., w_l)$ and $y_i \in \mathbb{R}$ is a rating. The goal is to learn a function $f(\mathbf{x})$ that maps the word vector $\mathbf{x}$ into a numerical rating $\hat{y}$, which indicates both the polarity and strength of the opinions expressed in a document.

Numerical review rating prediction is harder than classifying by polarity. Consider the following example from Amazon book reviews:

The organization of the book is hard to follow and the chapter titles are not very helpful, so going back and trying to find information is quite *difficult.*

We note that there are many subjective words (*hard, helpful, difficult*) modified by opinion modifiers such as (*very, quite*) and negation words like (*not*). For rating prediction, considering opinion modifiers is crucial; *very helpful* is a much stronger sentiment than *helpful*. Negation words also need attention. As pointed out by Liu and Seneff (2009) we cannot simply reverse the polarity. For example, if we assign a higher positive score to *very helpful* than to *helpful*, simply reversing the sign of the scores would incorrectly suggest that *not helpful* is less negative than *not very helpful*.

The widely used unigram (bag-of-words) model (Pang and Lee, 2005; Snyder and Barzilay, 2007; Goldberg and Zhu, 2006; Ganu et al., 2009) cannot properly capture phrase patterns. Consider the following example: *not so helpful* vs. *not so bad*. In a unigram-based regression model each unigram gets a weight indicating its polarity and strength. High positive/negative weights are strongly positive/negative clues. It is reasonable to assign a positive weight to *helpful* and a negative weight to *bad*. The fundamental problem of unigrams arises when assigning a weight to *not*. If *not* had a strongly negative weight, the positive weight of *helpful* would be strongly reduced while the negative weight of *bad* would be amplified (by combining weights). This clearly fails to capture the true intentions of the opinion phrases. The same problem holds for *so*, which is an intensifier that should keep the same sign as the word it modifies. We refer to this limitation of the unigram model as **polarity incoherence**.

A promising way of overcoming this weakness is to include *n-grams*, generalizing the bag-of-words model into a bag-of-phrases model (Baccianella et al., 2009; Pang and Lee, 2008). However, regression models over the feature space of all n-grams (for either fixed maximal n or variable-length phrases) are computationally expensive in their training phase. Moreover and most importantly for our setting, including n-grams in the model results in a very high dimensional feature space: many features will then occur only very rarely in the training data. Therefore, it is difficult if not impossible to reliably learn n-gram weights from limited-size training sets. We refer to this problem as the **n-gram sparsity bottleneck**. In our experiments we investigate the effect of using bigrams and variable-length ngrams for improving review rating prediction.

1.2 Contribution

To overcome the above limitations of unigram and n-gram features, we have developed a novel kind of *bag-of-opinions* model, which exploits domain-independent corpora of opinions (e.g., all Amazon reviews), but is finally applied for learning predictors on domain-specific reviews (e.g., movies as rated in IMDB or Rottentomatoes). A document is represented as a bag of opinions each of which has three components: a root word, a set of modifier words and one or more negation words. In the phrase *not very helpful*, the opinion root is *helpful*, one (of potentially many) opinion modifier(s) is *very*, and a negation word is *not*. We enforce polarity coherence by the design of a learnable function that assigns a score to an opinion.

Our approach generalizes the cumulative linear offset model (CLO) presented in (Liu and Seneff, 2009). The CLO model makes several restrictive assumptions, most notably, that all opinion scores within one document are the same as the overall document rating. This assumption does not hold in practice, not even in reviews with extremely positive/negative ratings. For example, in a 5-star Amazon review the phrases *most impressive book* and *it helps explain* should receive different scores. Otherwise, the later transfer step to different domains would yield poor predictions. Due to this restriction, CLO works well on particular types of reviews that have pro/con entries listing characteristic major opinions about the object under review. For settings with individual reviews whose texts do not exhibit any specific structure, the CLO model faces its limitations.

In our bag-of-opinions method, we address the learning of opinion scores as a constrained ridge regression problem. We consider the opinion scores in a given review to be drawn from an unknown probability distribution (so they do not have to be the same within a document). We estimate the review rating based on a set of statistics (e.g., expectation, variance, etc.) derived from the scores of opinions in a document. Thus, our method has a sound statistical foundation and can be applied to arbitrary reviews with mixed opinion polarities and strengths. We avoid the n-gram sparsity problem by the limited-size structured feature space of *(root,modifiers,negators)* opinions.

We treat domain-independent and domain-dependent opinions differently in our system. In the first step we learn a bag-of-opinions model on a large dataset of online reviews to obtain scores for domain-independent opinions. Since the polarity of opinions is not bound to a topic, one can learn opinion scores from a pooled corpus of reviews for various categories, e.g., movies, books, etc., and then use these scored opinions for predicting the ratings of reviews belonging to a particular category. In order to also capture domain-dependent information (possibly complementary to the opinion lexicon used for learning domain-independent opinions), we combine the bag-of-opinions model with an unigram model trained on the domain-dependent corpus. Since domain-dependent training is typically limited, we model it using unigram models rather than bag-of-opinions. By combining the two models, even if an opinion does not occur in the domain-dependent training set but it occurs in a test review, we can still accurately predict the review rating based on the globally learned opinion score. In some sense our combined learning scheme is similar to smoothing in standard learning techniques, where the estimate based on a limited training set is smoothed using a large background corpus (Zhai and Lafferty, 2004).

In summary, the contributions of this paper are the following:

1. We introduce the bag-of-opinions model, for capturing the influence of n-grams, but in a structured way with root words, modifiers, and negators, to avoid the explosion of the feature space caused by explicit n-gram models.

2. We develop a constrained ridge regression method for learning scores of opinions from domain-independent corpora of rated reviews.

3. For transferring the regression model to newly given domain-dependent applications, we derive a set of statistics over opinion scores in documents and use these as features, together with standard unigrams, for predicting the rating of a review.

4. Our experiments with Amazon reviews from different categories (books, movies, music) show that the bag-of-opinions method outperforms prior state-of-the-art techniques.

2 Bag-of-Opinions Model

In this section we first introduce the bag-of-opinions model, followed by the method for learning (domain-independent) model parameters. Then we show how we annotate opinions and how we adapt the model to domain-dependent data.

2.1 Model Representation

We model each document as a bag-of-opinions $\{op_k\}_{k=1}^{K}$, where the number of opinions K varies among documents. Each opinion op_k consists of an opinion root w_r, $r \in S_R$, a set of opinion modifiers $\{w_m\}_{m=1}^{M}$, $m \in S_M$ and a set of negation words $\{w_z\}_{z=1}^{Z}$, $z \in S_Z$, where the sets S_R, S_M, S_Z are component index sets of opinion roots, opinion modifiers and negation words respectively. The union of these sets forms a global component index set $S \in \mathbb{N}^d$, where d is the dimension of the index space. The opinion root determines the prior polarity of the opinion. Modifiers intensify or weaken the strength of the prior polarity. Negation words strongly reduce or reverse the prior polarity. For each opinion, the set of negation words consists of at most a negation valence shifter like *not* (Kennedy and Inkpen, 2006) and its intensifiers like capitalization of the valence shifter. Each opinion component is associated with a score. We assemble the scores of opinion elements into an opinion-score by using a score function. For example, in the opinion *not very helpful*, the opinion root *helpful* determines the prior polarity positive say with a score 0.9, the modifier *very* intensifies the polarity say with a score 0.5. The prior polarity is further strongly reduced by the negation word *not* with e.g., a score -1.2. Then we sum up the scores to get a score of 0.2 for the opinion *not very helpful*.

Formally, we define the function $score(op)$ as a linear function of opinion components, which takes the form

$$\begin{aligned} score(op) &= sign(r)\beta_r x_r \\ &+ \sum_{m=1}^{M} sign(r)\beta_m x_m \\ &+ \sum_{z=1}^{Z} sign(r)\beta_z x_z \end{aligned} \tag{1}$$

where $\{x_z, x_m, x_r\}$ are binary variables denoting the presence or absence of negation words, modifiers and opinion root. $\{\beta_z, \beta_m, \beta_r\}$ are weights of each opinion elements. $sign(r) : w_r \to \{-1, 1\}$ is the opinion polarity function of the opinion root w_r. It assigns a value 1/-1 if an opinion root is positive/negative. Due to the semantics of opinion elements, we have constraints that $\beta_r \geq 0$ and $\beta_z \leq 0$. The sign of β_m is determined in the learning phase, since we have no prior knowledge whether it intensifies or weakens the prior polarity.

Since a document is modeled as a bag-of-opinions, we can simply consider the expectation of opinion scores as the document rating. If we assume the scores are uniformly distributed, the prediction function is then $f(\mathbf{x}) = \frac{1}{K}\sum_{k=1}^{K} score(op_k)$ which assigns the average of opinion scores to the document $\mathbf{x}$.

2.2 Learning Regression Parameters

We assume that we can identify the opinion roots and negation words from a subjectivity lexicon. In this work we use MPQA (Wilson et al., 2005). In addition, the lexicon provides the prior polarity of the opinion roots. In the training phase, we are given a set of documents with ratings $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$, and our goal is to find an optimal function f^* whose predictions $\{\hat{y}_i\}_{i=1}^{N}$ are as close as possibile to the original ratings $\{y_i\}_{i=1}^{N}$. Formally, we aim to minimize the following loss function:

$$L = \frac{1}{2N}\sum_{i=1}^{N}(f(\mathbf{x}_i) - y_i)^2 \tag{2}$$

where $f(\mathbf{x}_i)$ is modeled as the average score of opinions in review $\mathbf{x}_i$.

First, we rewrite $score(op)$ as the dot product $\langle \boldsymbol{\beta}, \mathbf{p}\rangle$ between a weight vector $\boldsymbol{\beta} = [\boldsymbol{\beta}_z, \boldsymbol{\beta}_m, \beta_r]$ and a feature vector $\mathbf{p} = [sign(r)\mathbf{x}_z, sign(r)\mathbf{x}_m, sign(r)x_r]$. In order to normalize the vectors, we rewrite the weight and feature vectors in the d dimensional vector space of all root words, modifiers and negation words. Then $\boldsymbol{\beta} = [.., \boldsymbol{\beta}_z, 0, .., \boldsymbol{\beta}_m, 0, .., \beta_r, 0..] \in R^d$ and $\mathbf{p} = [sign(r)\mathbf{x}_z, 0, .., sign(r)\mathbf{x}_m, 0, .., sign(r)x_r, ...] \in R^d$. The function $f(\mathbf{x_i})$ can then be written as the dot product $\langle \boldsymbol{\beta}, \mathbf{v_i}\rangle$, where $\mathbf{v_i} = \frac{1}{K_i}\sum_{k=1}^{K_i}\mathbf{p}_k$, with K_i the number of opinions in review $\mathbf{x_i}$. By using this feature representation, the learning problem is equivalent to:

$$\min_{\beta} \; L(\boldsymbol{\beta}) = \frac{1}{2N}\sum_{i=1}^{N}(\langle \boldsymbol{\beta}, \mathbf{v}_i\rangle + \beta_0 - y_i)^2$$

$s.t.$

$$\begin{aligned} \beta_z \leq 0 \quad z \in S_Z \\ \beta_r \geq 0 \quad r \in S_R \end{aligned} \tag{3}$$

where $\boldsymbol{\beta} \in R^d$, $\boldsymbol{\beta} = [\boldsymbol{\beta}_z, \boldsymbol{\beta}_m, \boldsymbol{\beta}_r]$. β_0 is the intercept of the regression function, which is estimated as the mean of the ratings in the training set. We define a new variable $\tilde{y}_i = y_i - \beta_0$.

In order to avoid overfitting, we add an $l2$ norm regularizer to the loss function with the parameter $\lambda > 0$.

$$LR(\boldsymbol{\beta}) = \frac{1}{2N}\sum_{i=1}^{N}(\langle \boldsymbol{\beta}, \mathbf{v}_i\rangle - \tilde{y}_i)^2 + \frac{\lambda}{2} \parallel \boldsymbol{\beta} \parallel_2^2$$

$s.t.$

$$\begin{aligned} \beta_z \leq 0 \quad z \in S_Z \\ \beta_r \geq 0 \quad r \in S_R \end{aligned} \tag{4}$$

We solve the above optimization problem by Algorithm 1 using coordinate descent. The procedure starts with $\boldsymbol{\beta}^0 = 0$, $\boldsymbol{\beta}^0 \in R^d$. Then it updates iteratively every coordinate of the vector $\boldsymbol{\beta}$ until convergence. Algorithm 1 updates every coordinate $\beta_j, j \in \{1, 2, ..., d\}$ of $\boldsymbol{\beta}$ by solving the following one-variable sub-problem:

$$min_{l_j \leq \beta_j \leq c_j} LR(\beta_1, ..., \beta_j, ..., \beta_d)$$

where l_j and c_j denote the lower and upper bounds of β_j. If $j \in S_Z$, $l_j = -\infty$ and $c_j = 0$. If $j \in S_R$, $l_j = 0$ and $c_j = \infty$. Otherwise both bounds are infinity.

According to (Luo and Tseng, 1992), the solution of this one-variable sub-problem is

$$\hat{\beta}_j = max\{l_j, min\{c_j, g_j\}\}$$

where

$$g_j = \frac{\frac{1}{N}\sum_{i=1}^{N} v_{ij}(\tilde{y}_i - \sum_{l \neq j} \beta_l v_l)}{\frac{1}{N}\sum_{i=1}^{N} v_{ij}^2 + \lambda}$$

Here g_j is the close form solution of standard ridge regression at coordinate j (for details see (Friedman et al., 2008)). We prove the convergence of Algorithm 1, by the following theorem using techniques in (Luo and Tseng, 1992).

Theorem 1 *A sequence of $\boldsymbol{\beta}$ generated by Algorithm 1 globally converges to an optimal solution $\beta^* \in \chi^*$ of problem (4), where χ^* is the set of optimal solutions.*

Proof: *Luo and Tseng (1992) show that coordinate descent for constrained quadratic functions in the following form converges to one of its global optimal solutions.*

$$\begin{array}{ll} min_{\boldsymbol{\beta}} & h(\boldsymbol{\beta}) = \langle \boldsymbol{\beta}, \mathbf{Q}\boldsymbol{\beta} \rangle / 2 + \langle \mathbf{q}, \boldsymbol{\beta} \rangle \\ s.t. & \mathbf{E}^T \boldsymbol{\beta} \geq \mathbf{b} \end{array}$$

where $\mathbf{Q}$ is a $d \times d$ symmetric positive-definite matrix, $\mathbf{E}$ is a $d \times d$ matrix having no zero column, $\mathbf{q}$ is a d-vector and $\mathbf{b}$ is a d-vector.

We rewrite LR in matrix form as

$$\begin{aligned} & \frac{1}{2N}(\tilde{\mathbf{y}} - \mathbf{V}\boldsymbol{\beta})^T(\tilde{\mathbf{y}} - \mathbf{V}\boldsymbol{\beta}) + \frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} \\ = & \frac{1}{2N}(\mathbf{V}\boldsymbol{\beta})^T(\mathbf{V}\boldsymbol{\beta}) + \frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} - \frac{1}{2N}((\mathbf{V}\boldsymbol{\beta})^T\tilde{\mathbf{y}} \\ & -\frac{1}{2N}\tilde{\mathbf{y}}^T(\mathbf{V}\boldsymbol{\beta})) + \frac{1}{2N}\tilde{\mathbf{y}}^T\tilde{\mathbf{y}} \\ = & \langle \boldsymbol{\beta}, \mathbf{Q}\boldsymbol{\beta} \rangle / 2 + \langle \mathbf{q}, \boldsymbol{\beta} \rangle + constant \end{aligned}$$

where

$$\mathbf{Q} = \mathbf{B}^T\mathbf{B}, \mathbf{B} = \begin{bmatrix} \sqrt{\frac{1}{N}}\mathbf{V} \\ \sqrt{\lambda}\mathbf{I}^{d \times d} \end{bmatrix}, \mathbf{q} = \frac{-1}{N}(\mathbf{V}^T\tilde{\mathbf{y}})$$

where $\mathbf{I}^{d\times d}$ is the identity matrix. Because $\lambda > 0$, all columns of $\mathbf{B}$ are linearly independent. As $\mathbf{Q} = \mathbf{B}^T\mathbf{B}$ and symmetric, $\mathbf{Q}$ is positive definite.

We define $\mathbf{E}$ as a $d \times d$ diagonal matrix with all entries on the main diagonal equal to 1 except $e_{ii} = -1, i \in S_Z$ and $\mathbf{b}$ is a d-vector with all entries equal to $-\infty$ except $b_i = 0$, for $i \in S_Z$ or $i \in S_R$.

Because the almost cyclic rule is applied to generate the sequence $\{\boldsymbol{\beta}^t\}$, the algorithm converges to a solution $\boldsymbol{\beta}^* \in \chi^*$.

Algorithm 1 Constrained Ridge Regression

```
1: Input: $\lambda$ and $\{\mathbf{v}_n, \tilde{y}_n\}_{n=1}^N$
2: Output: optimal $\boldsymbol{\beta}$
3: repeat
4:   for $j = 1, ..., d$ do
5:     $g_j = \frac{\frac{1}{N}\sum_{i=1}^{N} v_{ij}(\tilde{y}_i - \sum_{l \neq j}\beta_l v_l)}{\frac{1}{N}\sum_{i=1}^{N} v_{ij}^2 + \lambda}$
6:     $\hat{\beta}_j = \begin{cases} 0, & if\, j \in S_R\, and\, g_j < 0 \\ 0, & if\, j \in S_Z\, and\, g_j > 0 \\ g_j, & else \end{cases}$
7:   end for
8: until Convergence condition is satisfied
```

2.3 Annotating Opinions

The MPQA lexicon contains separate lexicons for subjectivity clues, intensifiers and valence shifters (Wilson et al., 2005), which are used for identifying opinion roots, modifiers and negation words. Opinion roots are identified as the positive and negative subjectivity clues in the subjectivity lexicon. In the same manner, intensifiers and valence shifters of the type {negation, shiftneg} are mapped to modifiers and negation words. Other modifier candidates are adverbs, conjunctions and modal verbs around opinion roots. We consider non-words modifiers as well, e.g., punctuations, capitalization and repetition of opinion roots. If the opinion root is a noun, adjectives are also included into modifier sets.

The automatic opinion annotation starts with locating the continous subjectivity clue sequence. Once we find such a sequence and at least one of the subjectivity clue is positive or negative, we search to the left up to 4 words for negation words and modifier candidates, and stop if encountering another opinion root. Similarly, we search to the right up to 3 unigrams for modifiers and stop if we find negation words or any other opinion roots. The prior polarity of the subjectivity sequence is determined by the polarity of the last subjectivity clue with either positive or negative polarity in the sequence. The other subjectivity clues in the same sequence are treated as modifiers.

2.4 Adaptation to Domain-Dependent Data

The adaptation of the learned (domain-independent) opinion scores to the target domain and the integration of domain-dependent unigrams is done in a second ridge-regression task. Note that this is a simpler problem than typical domain-adaptation, since we already know from the sentiment lexicon which are the domain-independent features. Additionally, its relatively easy to obtain a large mixed-domain corpus for reliable estimation of domain-independent opinion scores (e.g., use all Amazon product reviews). Furthermore, we need a domain-adaptation step since domain-dependent and domain-independent data have generally different rating distributions. The differences are mainly reflected in the intercept of the regression function (estimated as the mean of the ratings). This means that we need to scale the positive/negative mean of the opinion scores differently before using it for prediction on domain-dependent reviews. Moreover, other statistics further characterize the opinion score distribution. We use the variance of opinion scores to capture the reliability of the mean, multiplied by the negative sign of the mean to show how much it strengthens/weakens the estimation of the mean. The mean score of the dominant polarity (*major exp*) is also used to reduce the influence of outliers. Because positive and negative means should be scaled differently, we represent positive and negative values of the mean and *major exp* as 4 different features. Together with variance, they are the 5 statistics of the opinion score distribution. The second learning step on opinion score statistics and domain-dependent unigrams as features, re-weights the importance of domain-independent and domain-dependent information according to the target domain bias.

3 Experimental Setup

We performed experiments on three target domains of Amazon reviews: books, movies (DVDs), and music (CDs). For each domain, we use ca. 8000 Amazon reviews for evaluation; an additional set of ca. 4000 reviews are withheld for parameter tuning (regularization parameter, etc.). For learning weights for domain-independent opinions, we use a mixed-domain corpus of ca. 350,000 reviews from Amazon (electronics, books, dvds, etc.); this data is disjoint from the test sets and contains no reviews from the music domain. In order to learn unbiased scores, we select about the same number of positive and negative reviews (where reviews with more/less than 3 stars are regarded as positive/negative). The regularization parameters used for this corpus are tuned on withheld data with ca. 6000 thematically mixed reviews.[1].

We compare our method, subsequently referred to as *CRR-BoO* (Constrained Ridge Regression for Bag-of-Opinions), to a number of alternative state-of-the-art methods. These competitors are varied along two dimensions: 1) feature space, and 2) training set. Along the first dimension, we consider a) unigrams coined *uni*, b) unigrams and bigrams together, coined *uni+bi*, c) variable-length n-grams coined *n-gram*, d) the opinion model by (Liu and Seneff, 2009) coined *CLO* (cumulative linear offset model). As learning procedure, we use ridge regression for a), b), and d), and bounded cyclic regression, coined *BCR*, for c). Along the second - orthogonal - dimension, we consider 3 different training sets: i) domain-dependent training set coined *DD*, ii) the large mixed-domain training set coined *MD*, iii) domain-dependent training set and the large mixed-domain training set coined *DD+MD*. For the *DD+MD* training set, we apply our two stage approach for *CRR-BoO* and *CLO*, i.e., we use the mixed-domain corpus for learning the opinion scores in the first stage, and integrate unigrams from *DD* in a second domain-adaptation stage. We train the remaining feature models directly on the combination of the whole mixed-domain cor-

[1]All datasets are available from `http://www.mpi-inf.mpg.de/~lqu`

feature models		uni	uni+bi	n-gram	CLO	**CRR-BoO**
DD	book	1.004	0.961	0.997	1.469	0.942
	dvd	1.062	1.018	1.054	1.554	0.946
	music	0.686	0.672	0.683	0.870	0.638
MD	book	1.696	1.446	1.643	1.714	1.427
	dvd	1.919	1.703	1.858	1.890	1.565
	music	2.395	2.160	2.340	2.301	1.731
DD+MD	book	1.649	1.403	1.611	1.032	**0.884**
	dvd	1.592	1.389	1.533	1.086	**0.928**
	music	1.471	1.281	1.398	0.698	**0.627**

Table 1: Mean squared error for rating prediction methods on Amazon reviews.

pus and the training part of *DD*.

The CLO model is adapted as follows. Since bags-of-opinions generalize CLO, adjectives and adverbs are mapped to opinion roots and modifiers, respectively; negation words are treated the same as CLO. Subsequently we use our regression technique. As Amazon reviews do not contain pro and con entries, we learn from the entire review.

For BCR, we adapt the variable-length n-grams method of (Ifrim et al., 2008) to elastic-net-regression (Friedman et al., 2008) in order to obtain a fast regularized regression algorithm for variable-length n-grams. We search for significant n-grams by incremental expansion in backward direction (e.g., expand *bad* to *not bad*). BCR pursues a dense solution for unigrams and a sparse solution for n-grams. Further details on the BCR learning algorithm will be found on a subsequent technical report.

As for the regression techniques, we show only results with ridge regression (for all feature and training options except BCR). It outperformed ϵ-*support vector regression* (SVR) of libsvm (Chang and Lin, 2001), lasso (Tibshirani, 1996), and elastic net (Zou and Hastie, 2005) in our experiments.

4 Results and Discussion

Table 1 shows the mean square error (MSE) from each of the three domain-specific test sets. The error is defined as $MSE = \frac{1}{N}\sum_{i=1}^{N}(f(\mathbf{x}_i) - y_i)^2$. The right most two columns of the table show results for the full-fledge two-stage learning for our method and CLO, with domain-dependent weight learning and the domain adaptation step. The other models are trained directly on the given training sets. For the DD and DD+MD training sets, we use five-fold cross-validation on the domain-specific sets. For the MD training set, we take the domain-specific test sets as hold-out data for evaluation.

Table 1 clearly shows that our *CRR-BoO* method outperforms all alternative methods by a significant margin. Most noteworthy is the music domain, which is not covered by the mixed-domain corpus. As expected, unigrams only perform poorly, and adding bigrams leads only to marginal improvements. BCR pursues a dense solution for unigrams and a sparse solution for variable-length n-grams, but due to the sparsity of occurence of long n-grams, it filters out many interesting-but-infrequent ngrams and therefore performs worse than the dense solution of the *uni+bi* model. The CLO method of (Liu and Seneff, 2009) shows unexpectedly poor performance. Its main limitation is the assumption that opinion scores are identical within one document. This does not hold in documents with mixed opinion polarities. It also results in conflicts for opinion components that occur in both positive and negative documents. In contrast, *CRR-BoO* naturally captures the mixture of opinions as a bag of positive/negative scores. We only require that the mean of opinion scores equals the overall document rating.

The right most column of Table 1 shows that our method can be improved by learning opinion scores from the large mixed-domain corpus. How-

opinion	score
good	0.18
recommend	1.64
most difficult	-1.66
but it gets very good!	2.37
would highly recommend	2.73
would not recommend	-1.93

Table 2: Example opinions learned from the Amazon mixed-domain corpus.

ever, the high error rates of the models learned directly on the MD corpus show that direct training on the mixed-domain data can introduce a significant amount of noise into the prediction models. Although the noise can be reduced by learning from MD and DD together, the performance is still worse than when learning directly from the domain-dependent corpora. Additionally, when the domain is not covered by the mixed-domain corpus (e.g., music), the results are even worse. Thus, the two stages of our method (learning domain-independent opinion scores plus domain-adaptation) are decisive for a good performance, and the sentiment-lexicon-based BoO model leads to robust learning of domain-independent opinion scores.

Another useful property of BoO is its high interpretability. Table 2 shows example opinion scores learned from the mixed-domain corpus. We observe that the scores corelate well with our intuitive interpretation of opinions.

Our *CRR-BoO* method is highly scalable. Excluding the preprocessing steps (same for all methods), the learning of opinion component weights from the ca. 350,000 domain-independent reviews takes only 11 seconds.

5 Related Work

Rating prediction is modeled as an ordinal regression problem in (Pang and Lee, 2005; Goldberg and Zhu, 2006; Snyder and Barzilay, 2007). They simply use the bag-of-words model with regression algorithms, but as seen previously this cannot capture the expressive power of phrases. The resulting models are not highly interpretable. Baccianella et al. (2009) restrict the n-grams to the ones having certain POS patterns. However, the long n-grams matching the patterns still suffer from sparsity. The same seems to hold for sparse n-gram models (BCR in this paper) in the spirit of Ifrim et al. (2008). Although sparse n-gram models can explore arbitrarily large n-gram feature spaces, they can be of little help if the n-grams of interests occur sparsely in the datasets.

Since our approach can be regarded as learning a domain-independent sentiment lexicon, it is related to the area of automatically building domain-independent sentiment lexicons (Esuli and Sebastiani, 2006; Godbole et al., 2007; Kim and Hovy, 2004). However, this prior work focused mainly on the opinion polarity of opinion words, neglecting the opinion strength. Recently, the lexicon based approaches were extended to learn domain-dependent lexicons (Kanayama and Nasukawa, 2006; Qiu et al., 2009), but these approaches also neglect the aspect of opinion strength. Our method requires only the prior polarity of opinion roots and can thus be used on top of those methods for learning the scores of domain-dependent opinion components. The methods proposed in (Hu and Liu, 2004; Popescu and Etzioni, 2005b) can also be categorized into the lexicon based framework because their procedure starts with a set of seed words whose polarities are propagated to other opinion bearing words.

6 Conclusion and Future Work

In this paper we show that the bag-of-opinions (BoO) representation is better suited for capturing the expressive power of n-grams while at the same time overcoming their sparsity bottleneck. Although in this paper we use the BoO representation to model domain-independent opinions, we believe the same framework can be extended to domain-dependent opinions and other NLP applications which can benefit from modelling n-grams (given that the n-grams are decomposable in some way). Moreover, the learned model can be regarded as a domain-independent opinion lexicon with each entry in the lexicon having an associated score indicating its polarity and strength. This in turn has potential applications in sentiment summarization, opinionated information retrieval and opinion extraction.

References

Baccianella, S., A. Esuli, and F. Sebastiani. 2009. Multi-facet rating of product reviews. In *ECIR*. Springer.

Chang, C.C. and C.J. Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Esuli, A. and F. Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *LREC*, pages 417–422.

Friedman, J., T. Hastie, and R. Tibshirani. 2008. Regularization paths for generalized linear models via coordinate descent. Technical report, Technical Report, Available at http://www-stat. stanford. edu/jhf/ftp/glmnet. pdf.

Ganu, G., N. Elhadad, and A. Marian. 2009. Beyond the stars: Improving rating predictions using review text content. In 12^{th} *International Workshop on the Web and Databases*.

Godbole, Namrata, Manjunath Srinivasaiah, and Steven Skiena. 2007. Large-scale sentiment analysis for news and blogs. In *ICWSM*.

Goldberg, A. B. and X.J. Zhu. 2006. Seeing stars when there aren't many stars: Graph-based semi-supervised learning for sentiment categorization. In *HLT-NAACL 2006 Workshop on Textgraphs: Graph-based Algorithms for Natural Language Processing*.

Hu, M.Q. and B. Liu. 2004. Mining and summarizing customer reviews. In *CIKM*, pages 168–177. ACM New York,USA.

Ifrim, G., G. Bakir, and G. Weikum. 2008. Fast logistic regression for text categorization with variable-length n-grams. In *KDD*, pages 354–362, New York,USA. ACM.

Kanayama, H. and T. Nasukawa. 2006. Fully automatic lexicon expansion for domain-oriented sentiment analysis. In *EMNLP*, pages 355–363.

Kennedy, A. and D. Inkpen. 2006. Sentiment classification of movie reviews using contextual valence shifters. *Computational Intelligence*, 22(2):110–125.

Kim, S.M. and E. Hovy. 2004. Determining the sentiment of opinions. In *COLING*, pages 1367–1373.

Lerman, K., S. Blair-Goldensohn, and R. McDonald. 2009. Sentiment summarization: Evaluating and learning user preferences. In *EACL*, pages 514–522. ACL.

Liu, J.J. and S. Seneff. 2009. Review Sentiment Scoring via a Parse-and-Paraphrase Paradigm. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 161–169. ACL.

Luo, Z.Q. and Q. Tseng. 1992. On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72(1):7–35.

Pang, B. and L. Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL*, page 124. ACL.

Pang, B. and L. Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

Popescu, A.M. and O. Etzioni. 2005a. Extracting product features and opinions from reviews. In *HLT/EMNLP*, volume 5, pages 339–346. Springer.

Popescu, A.M. and O. Etzioni. 2005b. Extracting product features and opinions from reviews. In *Proceedings of HLT/EMNLP*, volume 5, pages 339–346. Springer.

Qiu, G., B. Liu, J.J. Bu, and C. Chen. 2009. Expanding Domain Sentiment Lexicon through Double Propagation. In *IJCAI*.

Snyder, B. and R. Barzilay. 2007. Multiple aspect ranking using the good grief algorithm. In *NAACL/HLT*, pages 300–307.

Tibshirani, R. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288.

Titov, I. and R. McDonald. 2008a. A joint model of text and aspect ratings for sentiment summarization. In *HLT/ACL*, pages 308–316.

Titov, I. and R. McDonald. 2008b. Modeling online reviews with multi-grain topic models. In *WWW*, pages 111–120. ACM.

Wilson, T., J. Wiebe, and P. Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *HLT/ACL*, pages 347–354.

Zhai, C. X. and J. Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems*, 22(2):179–214.

Zou, H. and T. Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B(Statistical Methodology)*, 67(2):301–320.