Incorporating Reviewer and Product Information for Review Rating Prediction

Fangtao Li[1], Nathan Liu[2], Hongwei Jin[1], Kai Zhao[3], Qiang Yang[2] and Xiaoyan Zhu[1]
[1]State Key Laboratory of Intelligent Technology and Systems
[1]Tsinghua National Laboratory for Information Science and Technology
[1]Dept. of Computer Science and Technology, Tsinghua University, Beijing, China
[2]Dept. of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong
[3]NEC Labs China, Beijing, China
{fangtao06,cs.jin.hongwei}@gmail.com; {nliu,qyang}@cse.ust.hk; zxy-dcs@tsinghua.edu.cn

Abstract

Traditional sentiment analysis mainly considers binary classifications of reviews, but in many real-world sentiment classification problems, non-binary review ratings are more useful. This is especially true when consumers wish to compare two products, both of which are not negative. Previous work has addressed this problem by extracting various features from the review text for learning a predictor. Since the same word may have different sentiment effects when used by different reviewers on different products, we argue that it is necessary to model such reviewer and product dependent effects in order to predict review ratings more accurately. In this paper, we propose a novel learning framework to incorporate reviewer and product information into the text based learner for rating prediction. The reviewer, product and text features are modeled as a three-dimension tensor. Tensor factorization techniques can then be employed to reduce the data sparsity problems. We perform extensive experiments to demonstrate the effectiveness of our model, which has a significant improvement compared to state of the art methods, especially for reviews with unpopular products and inactive reviewers.

1 Introduction

With the development of Web 2.0, more and more people are likely to express their opinions on the Web. They can post reviews at E-Commerce web sites, and express their opinions on any type of entities and events in forums, blogs and other discussion groups. The opinion information is very useful for users and customers alike, many of whom typically read product or service reviews before buying them. Businesses can also use the opinion information to design better strategies for production and marketing. Hence, in recent years, sentiment analysis and opinion mining have become a popular topic for machine learning and data mining.

One of the most important opinion-mining tasks is sentiment classification, whereby the opinion documents are categorized into two sentiment categories: positive and negative. In this paper, we focus on a finer-grained task, where we consider the object of our prediction can be among a finite range of integers. We call this task the rating-inference task; It determines an author's polarity evaluation within a multi-point scale (e.g. one to five "stars"). We explore solutions for this task in the context of product or service reviews, which are one of the most important opinion resources and widely used by costumers and companies. We observe that in many real-world scenarios, it is important to provide numerical ratings rather than binary decisions, especially when a customer compares several candidate products, all of them are positive in a binary classification, to make a purchase decision, since customers not only need to know whether a product is good or not, but also how good the product is. A recent study pointed out that many consumers are willing to pay at least 20% percent more for an "excellent" product (with 5-star rating) than a "good" product (with 4-star rating) [Pang and Lee, 2008]. Therefore, being able to infer ratings on reviews will serve the users' needs better.

Review-rating prediction has traditionally been model as a multi-class classification or regression task. Most of the previous solutions consider this problem as a feature engineering problem, and exploit various features from the review text, such words, patterns, syntactic structure and semantic topic to improve the performance [Qu *et al.*, 2010; Pang and Lee, 2005; Leung *et al.*, 2006]. In this paper, we argue that review rating is not only related to the reviews' text description, but also dependent on the authors (that is, reviewers) as well as the target products. Different users may have different sentiment expression preferences. For example, some users choose to use "good" to describe a just-so-so product, but others may use "good" to describe an excellent product. Beside the user bias, there is also a product bias. We may use different opinion words to review different products, or even use the same opinion word to express different sentiment polarities for different products; for example, the opinion word "long" can be express a "positive" feeling for cellphone's battery life, but may have a "negative" feeling for a camera's focus time. Therefore, it is important to consider the relationship between the review-authors, as well as that of the target products, for review rating prediction.

In this paper, we propose a novel solution for this review-rating problem. We incorporate the reviewer and product information for review rating prediction. The reviewer and product preferences in a text document are modeled with a tensor. Each dimension corresponds to reviewer, product or text feature. In the tensor, each element e_k^{ij} describes a sentiment effect of the feature k from a text review, which a reviewer i posted on the product j. Due to the problem of data sparsity and scale, it is impractical to directly compute every value in the tensor. In this paper, we employ a tensor factorization technique to map the reviewer, product and text feature jointly into the low-dimensional latent factor space. We perform extensive experiments to show that our method is indeed effective in solving the review-rating problem for large and sparse data.

2 Reviewer and Product-Aware Sentiment Analysis

In the following, we first briefly review the traditional content only models for sentiment analysis. We then explain our reviewer and product-aware sentiment-analysis model based on tensor factorization techniques.

2.1 Notations

For a typical online review site, we would have a set of N reviewers $\mathcal{A} = \{u_1, ..., u_N\}$ writing reviews on a set of M products $\mathcal{P} = \{p_1, ..., p_M\}$. Normally, a reviewer would have only reviewed a subset of the M products. Let $\mathcal{S} \subseteq \mathcal{A} \times \mathcal{P}$ denote the set of reviewer-product pairs for which the reviewer has written a review on the product. We represent the text content in each review written by reviewer i on product j using the content feature vector. In this paper, we don't focus on the feature engineering task. We want to propose a fundamental new model to incorporate the reviewer and product information for review rating prediction. Here, we only use the basic feature, bag of words, to represent the review. The review word vector is denoted as $\mathbf{x}_{ij} \in \mathbb{R}^K$, where K is the word vocabulary size. For each review $\mathbf{x}_{ij}$, there is an associated rating score r_{ij} which indicates the reviewer's sentiments towards the product. Our goal is to design a novel model to predict the rating r_{ij} for review x_{ij}, with consideration of the authored reviewer i and the target product j.

2.2 Content based Sentiment Analysis Model

Most existing sentiment analysis models only consider the review text when trying to determine the sentiment level expressed by a review. In this work, we simply consider the terms (i.e. words) in the review text as our feature set. Suppose there are K unique terms in the review collection, we can now represent each review as an K-dimensional vector $\mathbf{x}_{ij}$. Given a set of reviews $\{\mathbf{x}_{ij} | (i,j) \in \mathcal{S}\}$ and their associated ratings $\{r_{ij} | (i,j) \in \mathcal{S}\}$, we want to learn a function $f : \mathbb{R}^K \rightarrow \mathbb{R}$ that for a review's rating can be determined by $f(\mathbf{x}_{ij})$. A commonly used model $f(\cdot)$ is the linear regression model, where the function value is determined by linear combination of the input features. More formally, $f(\cdot)$ is parameterized by an K-dimensional vector $w \in \mathbb{R}^K$ and of the following form:

$$f(\mathbf{x}_{ij}) = \mathbf{w}^T \cdot \mathbf{x}_{ij} = \sum_k w_k \cdot x_{ijk} \tag{1}$$

where the parameter value w_k captures the effect of term k in determining the overall sentiment scale of a review. It is worth noting that under this model, the same word would have the same effect in reviews written by different reviewers on different products.

The best parameters $\mathbf{w}^*$ can be found by solving the following optimization problems:

$$\Omega(\mathbf{w}) = \sum_{(i,j) \in \mathcal{S}} \mathcal{L}(\mathbf{w}^T \mathbf{x}_{ij}, r_{ij}) + \alpha \cdot |\mathbf{w}|_F^2 \tag{2}$$

where $|\mathbf{w}|_F^2 = \sum_k w_k^2$ is a regularization term defined in terms of the Frobenius norm of the parameter vector $\mathbf{w}$ and plays the role of penalizing overly complex models in order to avoid fitting. $\mathcal{L}(\cdot)$ is a loss function that measures discrepancy between the predicted sentiment scale $\mathbf{w}^T \cdot \mathbf{x}_{ij}$ and the true scale r_{ij}. In this work, we focus on modeling sentiment scales expressed in the form of numeric ratings, for which a commonly used loss function is the least squares loss:

$$\mathcal{L}(x, y) = (x - y)^2 \tag{3}$$

which is computationally easy to handle due its smooth differentiability.

2.3 Incorporating Reviewer and Product Effects

As we have argued in the introduction, it is not reasonable to determine the sentiment scale of a review purely based on its textual content since different reviewers or different products may have consistent biases when being associated the same terms. For example, a sentiment word like "nice" may indicate different degrees of positive sentiment for a very picky user versus a normal user. It is therefore necessary to refine the content only model described in the previous section to accommodate reviewer and product specific effects. In this section, we propose to incorporate the reviewer and product effects when predicting sentiment scales by designing a predictor function $f : \mathbb{R}^K \times \mathcal{A} \times \mathcal{P} \rightarrow \mathbb{R}$ that utilizes not only the review content $\mathbf{x}_{ij}$ but also the reviewer identity i and product identity j.

In order to make the function $f(\cdot)$ reviewer and product dependent, we can make the parameter vector $\mathbf{w}_{ij}$ dependent on both i and j so that each word parameter w_{ijk} is able to capture the the specific effect of term k when it is in a review written by reviewer i on product j. We design our reviewer and product aware sentiment scale prediction function $f : \mathbb{R}^K \times \mathcal{A} \times \mathcal{P} \rightarrow \mathbb{R}$ as follows:

$$\begin{aligned} f(\mathbf{x}_{ij}, i, j) &= (\mathbf{w}^0 + \mathbf{w}_{ij})^T \cdot \mathbf{x}_{ij} \\ &= \sum_{k=1}^{K} \left(w_k^0 + w_{ijk} \right) \cdot x_{ijk} \end{aligned} \tag{4}$$

where we use a *base* parameter vector $\mathbf{w}^0$ to capture the reviewer and product independent effects of each term and introduces reviewer-product dependent *bias* parameter vector $\mathbf{w}_{ij}$ to add an offset to capture reviewer and product dependent term effects. Note that when the bias vectors $\mathbf{w}_{ij}$ are zero, we could recover the content only sentiment analysis model.

To learn the parameter vectors $\mathbf{w}_{ij}$ for all reviewer-product pairs, a naive approach is simply to build a separate regression model on each individual review $\mathbf{x}_{ij}$. However, this is totally impractical for two reasons. Firstly, it would explosively increase the number of free parameters to $K \times M \times N$. For popular review sites with large number of reviewers and products, such parametric complexity of the model becomes intractable. Secondly, this approach would inherently suffer from data sparsity since each reviewer can hardly write more than one review on the same product j, which makes it impossible to obtain sufficient training to learn $\mathbf{w}_{ij}$ reliably. To avoid these drawbacks, we propose a latent factor model, which can reduce the parametric complexity when modeling reviewer-product dependency in the parameter vector $\mathbf{w}_{ij}$. The latent-factor model can also effectively deal with the data sparsity issues brought forth by complex models, by jointly modeling the granular effects associated with all reviewer-product pairs via parameter sharing.

Note that the $\mathbf{w}_{ij}$ vectors can be naturally arranged as a 3 dimensional tensor $\mathbf{W} \in \mathbb{R}^{M \times N \times K}$ tensor where the first, second and third dimension correspond to reviewer, product and term respectively and each entry in the tensor correspond to a particular parameter w_{ijk}. As it is infeasible to model each w_{ijk} as a free parameter due to computational and data sparsity issues, we propose to decompose the $M \times N \times K$ dimensional tensor $\mathbf{W}$ using three low rank matrices $\mathbf{U} \in \mathbb{R}^{M \times D}$, $\mathbf{V} \in \mathbb{R}^{N \times D}$ and $\mathbf{P} \in \mathbb{R}^{K \times D}$: one for each of three dimensions of reviewers, products and terms. We refer to these matrices as latent factor matrices. Each row $\mathbf{u}_i, \mathbf{v}_j$ and p_k of these factor matrices correspond to the latent factors associated with each particular reviewer, product and term. Given the three factor matrices, the tensor $\mathbf{W}$ can be computed by multiplying three latent factor matrices together via tensor product:

$$\mathbf{W} = \mathbf{I} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{P} \tag{5}$$

where $\mathbf{I}$ denotes the $D \times D \times D$ identity tensor and $\times_k$ is the tensor product to multiply a matrix on the k-th dimension with a tensor. The number of parameters under this model is just $D \times (M + N + K)$, which is several orders of magnitude more compact than the full tensor model. Under this model, each parameter w_{ijk} can be determined as follows:

$$w_{ijk} =< \mathbf{u}_i, \mathbf{v}_j, \mathbf{p}_k >= \sum_{f=1}^{D} u_{if} \cdot v_{jf} \cdot p_{kf} \tag{6}$$

where $< \mathbf{u}_i, \mathbf{v}_j, \mathbf{p}_k >$ denote the tensor outer products between the reviewer, product and term factors. We can see that the tensor outer product operation determines the value w_{ijk} based on the pairwise interactions between the latent factors of all three entities: reviewer, product and term, which can naturally reflect how an reviewer uses a term when reviewing a product. Also we can see that the same reviewer factor $\mathbf{u}_i$ are shared when computing w_{ijk} values for different j and k combinations, which effectively captures the possible correlations between w_{ijk} values for the same reviewer. Similarly, the sharing of product and term factors when determining w_{ijk} for different (i, k) and (i, j) combinations are achieved in the same way.

With this tensor factorization model in place, we can reformulate the prediction function $f : \mathbb{R}^K \times \mathcal{A} \times \mathcal{P} \rightarrow \mathbb{R}$ as follows:

$$\begin{aligned} f(\mathbf{x}_{ij}, i, j) &= (\mathbf{w}^0 + \mathbf{w}_{ij})^T \cdot \mathbf{x}_{ij} \\ &= \sum_{k=1}^{K} \big(w_k^0 + < \mathbf{u}_i, \mathbf{v}_j, \mathbf{p}_k > \big) \cdot x_{ijk} \end{aligned} \tag{7}$$

where $< \mathbf{u}_i, \mathbf{v}_j, \mathbf{p}_k >$ is defined in Equation 6.

The model parameters $\mathbf{w}^0, \mathbf{U}, \mathbf{V}$ and $\mathbf{P}$ can be learnt by minimizing the following objective function:

$$\begin{aligned} \Omega(\mathbf{w}^0, \mathbf{U}, \mathbf{V}, \mathbf{P}) = & \sum\nolimits_{(i,j)\in\mathcal{S}} (r_{ij} - \hat{r}_{ij})^2 + \alpha \cdot |\mathbf{w}^0|_F^2 \\ & + \beta \cdot (|\mathbf{U}|_F^2 + |\mathbf{V}|_F^2 + |\mathbf{P}|_F^2) \end{aligned} \tag{8}$$

where $\hat{r}_{ij}$ denotes the value of $f(x_{ij}, i, j)$. Note that as $\beta \rightarrow +\infty$, we can force $\mathbf{U}, \mathbf{V}$ and $\mathbf{P}$ to zero and the resulted model would be equivalent to the content only sentiment analysis model, which can be considered as a special case of our proposed model.

Let $e_{ij} = \hat{r}_{ij} - r_{ij}$ denote the prediction error of the model. The partial derivatives of the objective function Ω with respect to the model parameters $\mathbf{w}^0$,$\mathbf{u}_i$,$\mathbf{v}_j$ and $\mathbf{p}_k$:

$$\frac{\partial\Omega}{\mathbf{w}^0} = \sum_{(i,j)\in\mathcal{S}} e_{ij} \cdot \mathbf{x}_{ij} \tag{9}$$

$$\frac{\partial\Omega}{\mathbf{u}_i} = \sum_{j\in\mathcal{P}_i} e_{ij} \cdot \big(\sum_k x_{ijk} \cdot \mathbf{p}_k \big) \otimes \mathbf{v}_j \tag{10}$$

$$\frac{\partial\Omega}{\mathbf{v}_j} = \sum_{i\in\mathcal{A}_j} e_{ij} \cdot \big(\sum_k x_{ijk} \cdot \mathbf{p}_k \big) \otimes \mathbf{u}_i \tag{11}$$

$$\frac{\partial\Omega}{\mathbf{p}_k} = \sum_{(i,j)\in\mathcal{S}} e_{ij} \cdot x_{ijk} \cdot \mathbf{u}_i \otimes \mathbf{v}_j \tag{12}$$

where $\otimes$ denotes the element-wise matrix multiplication operation. $\mathcal{P}_i$ denotes the set of products that have been reviewed by reviewer i whereas $\mathcal{A}_j$ denotes the set of reviewers who have reviewed product j. With these gradients, we can apply gradient descent algorithm to minimize the objective functions.

3 Experiments

3.1 Experiment Setting

Data Set

In this sub-section, we first describe the data sets used in our experiments. We employ two types of data sets. The first data set[1] is a collection of movie reviews crawled from Internet. A subset of this collection has been used in [Pang and Lee, 2005]. We follow the instructions to parse the collection. We acquire 15507 reviews with explicit stars. Following [Pang and Lee, 2005], all the review stars are mapped into a 1˜4 sentiment scales. The detailed description is shown in Table1

The second data set is collected by us. We have crawled a set of product reviews with their reviewers and products from

[1] http://www.cs.cornell.edu/people/pabo/movie-review-data/

Data Set	# reviewer	# product	# review	rating scale
Movie	458	4543	15501	1˜4
Epinions	5806	13269	220654	1˜5

Table 1: Data Set Description.

Epinions Web site[2]. We remove the anonymous and duplicate reviews, and finally get 220654 reviews. Each review in Epinions is rated from 1 to 5. The detailed description is also shown in Table1

For the supervised methods, we need to split the data set into training and test sets. In order to remove the uncertainty of the data split, a five-fold cross validation procedure is applied in our experiments, where four folds are used for training and one fold for test. Though we don't need to split data for unsupervised methods, we apply the unsupervised methods on the same test data set for the convenience of comparison.

Evaluation Metric

Since our task is a rating task, for a true 5 rating review, the prediction result of 3 is better than the result of 1. Therefore, we evaluate the results of different prediction methods using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE):

$$MAE = \frac{\sum_{(i)} |p_i - r_i|}{n} \tag{13}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{(i)} (p_i - r_i)^2} \tag{14}$$

where r_i is the true rating for the ith review, p_i is the predicted rating, n is the total number of reviews in our collection. The two metrics both measure how much our predicted rating deviates from the true rating. A smaller value indicates a more accurate prediction.

3.2 Baselines

We compare our methods with several baselines. We first design two heuristic methods: RANDOM is simply randomly assigning a rating score to a review; Majority is choosing the majority rating score in the training set to the reviews in test set.

We have introduced our basic model, linear regression (Reg), in the previous section. We also formulate the rating task as a multi-class classification problem. Each rating corresponds to a category. We use multi-class SVM as the classifier. SVM^{light} is used for rating prediction.

We also implement the state of the art method, PSP model [Pang and Lee, 2005], which employ the label similarity through metric labeling. Besides the initial label prediction function with the base learner, this method also assumes the similar reviews have similar ratings. The review similarity is computed by PSP, *positive-sentence percentage*, which denotes the percentage of positive sentences in all subjective sentences. The positive and subjective sentences are identified by Naive Bayes classifier trained with manually labeled sentence-level data set. We only can acquire the sentence level movie review data [Pang and Lee, 2005]. We will only provide the PSP results for the movie review data. We implement the PSP with two base learners, linear regression and SVM classification, respectively. These two advanced models are denoted as $Reg + PSP$ and $SVM + PSP$.

The last baseline is a state of the art matrix factorization (MF) based collaborative filtering model [Koren *et al.*, 2009], which relies on modeling the correlations between different users' rating behaviors in order to predict a user's rating based on the ratings of other similar minded users. The MF method is not able to utilize the review text and focus on modeling the review rating matrix only.

3.3 Experiment Results

Table 2 shows the rating prediction results in two data sets. From the table, we can see that even MF doesn't use any text features, it still achieves better results than Random and Majority in both two data sets. This shows the effectiveness of using the reviewer identity and product identity in our method. The text based methods, regression and classification, are more effective than the matrix factorization (MF) method, which shows the importance of the text features in the review rating task. The regression method Reg achieves consistently better results than the classification method SVM. This is because the rating scales are ordinal in this task, and the regression models this ordinal relation. The metric labeling methods, PSP, achieve best results among the baselines for movie review data set. The PSP methods incorporate the review neighbors into base learner to help the rating prediction task. This strategy improves both regression and SVM results. Our method, which integrates the reviewer and product information into text based predictor, achieves the best results for the two data sets. It achieves nearly 10% compared with the state of the art methods, Reg+PSP (from 0.635 to 0.571 on MAE) for movie review data set, and about 5% compared with regression method (from 0.660 to 0.63 on MAE) for the Epinions data set. This demonstrates the effectiveness of our tensor based model. In traditional regression and classification models, all the reviewers and products' reviews share the same weight vector for the features. In contrast, in our model, we learn a reviewer and product specific feature weight vector, which can provide more accurate results for each type of reviews.

From the table, we can see that the result for movie is better than the Epinions's result. One reason may be from the different rating scales. The number of rating scales is 5 for Epinions, while movie review data set only contains 4 rating scales. The second reason may be from the different matrix density, which we will discuss later. We also find that different methods play similar role in movie and Epinion data sets. MAE and RMSE also have the same trend to evaluate the results. Therefore, in the following experiments, we only provide the results on movie data set with MAE evaluation.

Product Popularity Measurement

In this section, we examine whether our methods work more effectively for reviews on popular products or unpopular products in more detail. Product popularity is determined by

[2] http://www.epinions.com

		Random	Majority	MF	Reg	Reg+PSP	SVM	SVM+PSP	Our
Epinions	MAE	1.612	1.051	0.962	0.660	—	0.683	—	**0.632**
	RMSE	1.986	1.632	1.406	0.937	—	1.004	—	**0.813**
Movie	MAE	1.085	0.801	0.762	0.646	0.635	0.689	0.674	**0.571**
	RMSE	1.396	1.063	0.957	0.899	0.886	0.975	0.932	**0.736**

Table 2: Results on rating prediction task.

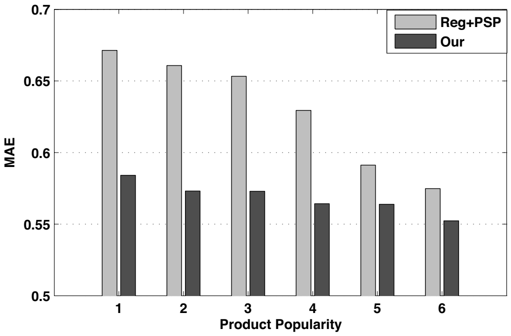

Figure 1: Evaluations on Different Product Popularity

the number of reviews for this product in training set. We then sort reviews into six groups depending on the popularity of the reviewed product, where the group 1 and 6 denotes reviews on least and most popular products, respectively. Detailed results are shown in Figure 1. We can see that while our proposed method consistently improves the performance for products with different popularity, the largest relative improvement is achieved on reviews written on less popular products. This is because the most popular products tend to have a lot of reviews and are over represented in the training data, whereas the less popular products are under represented. The regression model learns a unique global feature weight vector, the determination of which would be dominated by reviews on popular products. However, in our model, we learn product-specific feature weight vector via the flexible tensor factorization model, which can effectively capture product-specific behaviors even for those without large number of reviews. When measuring the performances for reviewer groups with different review quantities, we also observed similar results, which is not shown due to space limitation.

Matrix Density Measurement

We also study the influence of the author product review rating matrix density. The matrix density is computed by $\frac{Num_{review}}{Num_{author} * Num_{product}}$. We remove the reviews written by reviewers or products with fewer than n total reviews. With a small threshold n, the rating matrix would have a lower density and the data set would involve a larger number of unique authors and products and vice versa. The experiment results are show in Figure 2. We have two observations: first, the performance is improved as the density become bigger. This may be one of reasons that movie results are better than the Epinions results in Table 2, since the density for movie is bigger than the one for Epinions, 0.0075 v.s. 0.0029 from Tabel 1. The second observation is that although our model consistently achieves best results among different matrix densities, it achieves more relative improvement for the low density matrix when compared with the content only models. This is because when the matrix density is high, it mainly contains popular products and active reviewers, which have large amount of reviews for training. For more sparse rating matrices, there would be more inactive authors and unpopular products and the rating behaviors of the author and product population would naturally become more diverse. And therefore the performance is more likely to benefit from modeling author and product dependent effects.

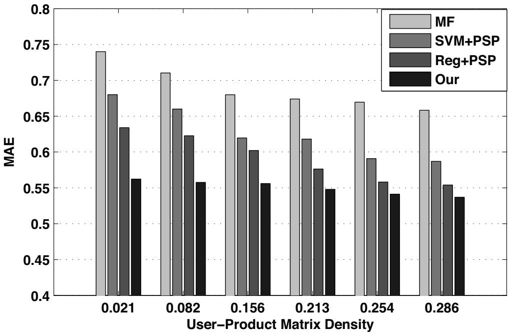

Figure 2: Evaluations on Different Matrix Density

Parameter Sensitivity

D is the number of latent factors. We perform experiments to study how D affects the performance of the prediction. Figure 3 shows the results. Each line is acquired with specific α and β. We can see that when D is greater than 3, the experiments have best results for the movie data set.

4 Related Work

Sentiment analysis and opinion mining have drawn a lot of attentions [Pang and Lee, 2008; Liu, 2010; Wu and Huberman, 2010]. Most previous work on sentiment classification has focused on the binary distinction of positive vs. negative (e.g. [Li *et al.*, 2010; Qiu *et al.*, 2009]). In this paper, we focus on a finer-grained review rating prediction task. Most existing studies [Pang and Lee, 2005; Leung *et al.*, 2006; Ganu *et al.*, 2009], focus on the review text for rating prediction. They derive syntactic and semantic features using natural language processing techniques, and use traditional

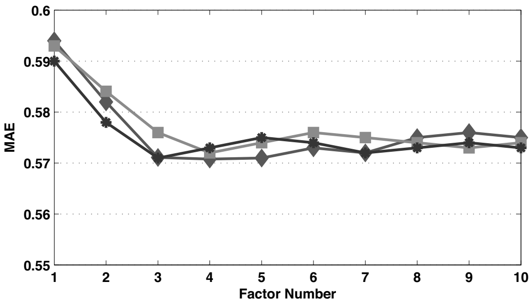

Figure 3: Evaluations on Different D

learning models, such as SVM, for rating prediction. However, in this paper, we propose a novel solution by providing a new learning model, which is reviewer and product dependent. This work is orthogonal to the existing text feature based studies.

Several previous studies find that the authorship affects performance in sentiment analysis [Pang and Lee, 2005; Seroussi *et al.*, 2010]. They train a separate classifier for each user, which is infeasible for learning an accurate classifier, when each user has only a small number of reviews. With our method, we employ a tensor factorization technique to discover the latent factors among different reviews, products and text features, which effectively reduce the sparsity and complexity problem.

Our work is also related with collaborative filtering (CF). CF techniques assume that the Users who have similar actions in the past tend to do similar things in the future. In this paper, we jointly model the reviewers, products and text features into the same latent factor space with the tensor factorization technique. Our problem, however, is different from content based collaborative filtering (CBCF) [Melville *et al.*, 2002], because content information is associated with only the user or the item whereas review text is associated with user-item pairs, which cannot be handled by existing CBCF models.

5 Conclusions

In this paper, we have proposed a novel learning framework to exploit the reviewer and product information for review rating prediction. We presented a tensor based method to represent the relationship among reviewers, products and text features. To reduce the data sparsity problem and the problem brought forth by the data complexity, we exploited tensor factorization to generate latent factors, whereby we can model the association among reviewers, products and text features. Our experiment results showed that it is effective to model reviewer and product information in the text based learner. We have achieved significant improvement as compared to several state-of-the-art methods, especially for the reviews with unpopular products and inactive reviewers.

In the future, we would like to consider more text features to improve our integrated mode, as well as consider other solutions under the tensor-based framework.

Acknowledgments

This work is supported by Canada's IDRC Research Chair in Information Technology program, Project Number: 104519-006, RGC/NSFC Joint Grant N_HKUST 624/09, the Chinese Natural Science Foundation No. 60973104, China 973 Project No. 2007CB311003 and China Core High-Tech Project No. 2011ZX01042-001-002. The first author also thanks the support of Google PhD Fellowship.

References

[Ganu *et al.*, 2009] Gayatree Ganu, Noemie Elhadad, and Amélie Marian. Beyond the stars: Improving rating predictions using review text content. In *WebDB*, 2009.

[Koren *et al.*, 2009] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42:30–37, August 2009.

[Leung *et al.*, 2006] CaneWing Leung, Stephen Chi Chan, and Fu Chung. Integrating collaborative filtering and sentiment analysis: A rating inference approach. In *ECAI Workshop*, pages 300–307, April 2006.

[Li *et al.*, 2010] Fangtao Li, Minlie Huang, and Xiaoyan Zhu. Sentiment analysis with global topics and local dependency. In *Proceedings of AAAI*, pages 1371–1376, Atlanta, Georgia, USA, 2010. AAAI.

[Liu, 2010] Bing Liu. Sentiment analysis and subjectivity. In N. Indurkhya and F. Damerau, editors, *Handbook of Natural Language Processing*. CRC Press, Boca Raton, FL, 2010. ISBN 978-1420085921.

[Melville *et al.*, 2002] Prem Melville, Raymod J. Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proceedings of AAAI*, pages 187–192, Menlo Park, CA, USA, 2002. AAAI.

[Pang and Lee, 2005] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*, pages 115–124, 2005.

[Pang and Lee, 2008] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2:1–135, January 2008.

[Qiu *et al.*, 2009] Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. Expanding domain sentiment lexicon through double propagation. In *Proceedings of IJCAI*, pages 1199–1204, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[Qu *et al.*, 2010] Lizhen Qu, Georgiana Ifrim, and Gerhard Weikum. The bag-of-opinions method for review rating prediction from sparse text patterns. In *COLING*, pages 913–921, Stroudsburg, PA, USA, 2010. ACL.

[Seroussi *et al.*, 2010] Yanir Seroussi, Ingrid Zukerman, and Fabian Bohnert. Collaborative inference of sentiments from texts. In *UMAP*, pages 195–206, 2010.

[Wu and Huberman, 2010] Fang Wu and Bernardo A. Huberman. Opinion formation under costly expression. *ACM Trans. Intell. Syst. Technol.*, 1:5:1–5:13, October 2010.