Leveraging Personalized Sentiment Lexicons for Sentiment Analysis

Dominic Seyler, Jiaming Shen, Jinfeng Xiao, Yiren Wang and ChengXiang Zhai
{dseyler2,js2,jxiao13,yiren,czhai}@illinois.edu
University of Illinois at Urbana-Champaign

ABSTRACT
We propose a novel personalized approach for the sentiment analysis task. The approach is based on the intuition that the same sentiment words can carry different sentiment weights for different users. For each user, we learn a language model over a sentiment lexicon to capture her writing style. We further correlate this user-specific language model with the user’s historical ratings of reviews. Additionally, we discuss how two standard CNN and CNN+LSTM models can be improved by adding these user-based features. Our evaluation on the Yelp dataset shows that the proposed new personalized sentiment analysis features are effective.

KEYWORDS
sentiment analysis, sentiment lexicons, personalization

1 INTRODUCTION
Sentiment analysis has seen lots of attention in research due to its large benefit to downstream applications, such as recommender systems [15, 16], social media analysis [4] and e-commerce websites [24]. In sentiment analysis, the system has to predict the sentiment of an input text. This is usually done by either predicting a coarse sentiment label, or a probability score for each sentiment dimension. User reviews, such as Yelp reviews, are an indirect statement of sentiment and can therefore be used as a proxy for the sentiment analysis paradigm. More concretely, in the case of rated user reviews the model receives the textual review and has to predict the user’s review score.

Since user ratings are highly subjective, it is natural to incorporate personalization into the model, which enables us to learn a user-specific predictive function. Along with text, the model takes user profiles as input and returns multi-level sentiment scores. The user profile includes documents previously written by the same author and can be utilized to infer user traits related to her writing style. As a result, a personalized sentiment model can help improve recommendations or search results for a user.

Based on the intuition that sentiment words carry different sentiment weight for different users, we leverage an external sentiment lexicon for personalization. Here, the model is personalized by learning a user-specific unigram language model over the most prominent words in a sentiment lexicon. Similar to Gao et al. [11], we try to estimate the general rating behavior of a user, which has been shown to be a strong signal for predicting future ratings. However, we capture a different set of statistical rating-based features and use them in combination with a personalized language model, which is not present in [11]. For example, a certain person might use the word “good” very generously, therefore it does not carry much value in predicting the sentiment. However, a more conservative person might use “good” only sporadically, thus, it has great predictive value for high-scoring reviews. We find that
the information of sentiment words can be enriched with signals derived from each user’s rating habit. We therefore exploit rating history in addition to sentiment lexicons.

We propose and design general personalized sentiment lexicon features and study how to effectively incorporate them into a learning model. Our model learns personalized sentiment lexicons and rating patterns as complementing features to common text-based features. Since neural network models have proven effective for sentiment analysis [7, 10, 12, 22, 25, 27], we show that our features boost performance in two neural models. However, how to integrate these features into a neural network is non-trivial. We accomplish this by leveraging the intuition from wide and deep learning in Cheng et al. [8], and concatenate review representations with our personalization features. When these personalization components are added to our models, they outperform other personalized baselines that use user history. Thus, our contributions are as follows:

1. We propose novel personalized sentiment features derived from a user-specific unigram language model over an external sentiment lexicon.
2. We show that the personalized sentiment features improve a deep learning framework, when integrated in addition to a user’s rating history.
3. We compare our model to other (deep) personalized baselines and find that our models perform up to 1.3% better.
4. We create a dataset with user history, based on the Yelp 2018 dataset.

2 RELATED WORK
The incorporation of sentiment lexicons for the sentiment analysis task is common [6, 13, 17, 23]. For example, Teng et al. [25] uses the context of sentiment words in a neural model to make sentiment predictions. However, none of the current methods explicitly models a user’s use of sentiment words. Personalization is usually done implicitly by grouping user reviews and trying to learn some hidden user representation [10, 12, 27]. For example, Tang et al. [22] represents users and products as separate feature vectors, where similar users and products are closer in the embedding space. Chen et al. [7] combines user and product information as attentions over these different levels for personalization. In contrast, our models capture user specificity by explicitly modeling the user’s writing style of expressing sentiment using a personalized sentiment lexicon.

3 OUR APPROACH
As in previous work [7, 22], we map the problem of sentiment analysis to predicting the score of user reviews on Yelp. The prediction task is as follows: given an input text (i.e., the review) and a user profile (i.e., a list of previous reviews by the same user), the model has to predict the review score. In what follows, we discuss our proposed user-specific features and our methods of representing the textual input. Figure 1 illustrates our model architectures.

3.1 User-Specific Features
Sentiment Language Model Features (lm): Sentiment words are important triggers for review rating classification. We use a unigram language model on two constructed sentiment lexicons (positive and negative) to capture user-specific language features. For sentiment lexicon construction, we build a sentiment language model derived from the SentiWordNet dictionary [5]. The candidate list is filtered with a sentiment score threshold: \( \text{score}^\text{pos}(\text{word}) > \epsilon \) or \( \text{score}^\text{neg}(\text{word}) > \epsilon \) (we empirically choose \( \epsilon = 0.3 \)), and a threshold for absolute difference between the positive and negative score: \( |\text{score}^\text{pos}(\text{word}) - \text{score}^\text{neg}(\text{word})| > \delta \text{diff} \) (we empirically choose \( \delta \text{diff} = 0.5 \)). We learn a user-specific language model with the vocabulary based on the sentiment lexicon using maximum-likelihood estimation. Notice that the vocabulary size is usually large (~4,000 in our case). We therefore use truncated-SVD to compress the user-specific language model into a low-dimensional space. Finally, each user \( u \) is represented by a dense vector \( s(u) \in \mathbb{R}^k \) where \( k \) is the number of largest singular values selected in the truncated SVD (\( k = 100 \) in our model).

Rating Features (rating): The user’s rating history is another useful signal for sentiment analysis [11]. Therefore, we obtain additional user-specific features based on her rating statistics, including maximum, minimum, mean score, standard deviation and the histogram of the user’s rating history. We denote the user-specific feature vector as \( r(u) \).

3.2 Text Representations
In this paper, we focus on studying the impact of adding personalized features to neural networks, though those features can also be incorporated into other machine learning models. Our model uses a hierarchical neural network model to encode the review text into a low dimensional vector space. At the word level, we use word embeddings for word representation. We use two different architectures for the review representations:

CNN: Similar to Kim [18], our CNN model obtains the review document representation directly from the word level \( (w_i) \) representation:

\[
d(N) = \text{CNN}(w_1, w_2, ..., w_T).
\]

CNN-LSTM: A hierarchical model, where at sentence level we adopt a CNN to embed each sentence in the review into a dense vector. The sentence representations are then passed into a LSTM layer to generate document level representations:

\[
s(k) = \text{CNN}(w_{k1}, w_{k2}, ..., w_{kt}), d(N) = \text{LSTM}(s(1), ..., s(N)).
\]

For each sentence vector \( s(i) \), the LSTM model will output a vector \( o(i) \). We use max pooling\(^1\) over these \( N \) output vectors for constructing the final document representation:

\[
d(N) = \max\{o(1), ..., o(N)\}.
\]

Inspired by Cheng et al. [8], the text representation, user sentiment language features and user rating features are concatenated and fed into a fully connected layer, which is passed into a softmax layer for classification. The personalized models with both review representations and user-specific features are denoted as P-CNN and P-CNN-LSTM.

\(^1\) We test mean pooling, direct sum of output vectors \( s(i) \), and the attention mechanism from Yang et al. [28]. Max pooling gives the best performance.
4 EVALUATION

4.1 Dataset
Currently available datasets on sentiment analysis (e.g., [9, 22]) contain a random sample of users and are split into training, development and testing. There are no guarantees that users in the training dataset will also be included in development and testing. In order to study the effect of personalization, the underlying dataset has to have user history available\(^2\). We therefore create our own dataset that abides by the previously mentioned characteristics.

We use the 2018 version of the Yelp dataset [1], which includes reviews about businesses, as well as meta-data about users and businesses. We find that in the dataset, the level of user activity varies greatly. While most users are moderately active, there are some users with over a thousand reviews. To make sure our training and evaluation are not dominated by these very active users, we restrict our analysis to users with 20-200 reviews.

Table 1 shows the characteristics our dataset (Yelp18-U), where the following two conditions are satisfied: (1) Ensure that all users in the development and test datasets have been observed in the training dataset. (2) The total number of reviews in the training set should be close to 50k. Using this process we sample 1,950 users with 50,018 reviews during 2011-15. Those reviews were used as training data and a development and test dataset are formed by taking reviews written by those users in 2016 and 2017, respectively.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Reviews</th>
<th>#Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>89,150</td>
<td>1,950</td>
</tr>
<tr>
<td>Train. (2011-2015)</td>
<td>50,018</td>
<td>1,950</td>
</tr>
<tr>
<td>Dev. (2016)</td>
<td>21,120</td>
<td>1,950</td>
</tr>
<tr>
<td>Test (2017)</td>
<td>18,012</td>
<td>1,950</td>
</tr>
</tbody>
</table>

Table 1: Statistics of the Yelp18-U dataset.

4.2 Experimental Setup
In our experiments we answer the following research questions: (1) we measure the effectiveness of our personalization features by performing an ablation study. (2) we compare our model’s performance with other personalized methods:

- **Semantic Representations for Users and Products (UPNN) [22]**: We use the source code provided by the authors [2] to run their models on our dataset and use the provided settings for hyper-parameters and GloVe’s [21] 200-dimensional word embeddings.
- **Neural Sentiment Classification (NSC) [7]**: We use the source code provided by the authors [3]. We use the standard settings for hyper-parameters and the pre-trained word embeddings that are provided with the source. There are three versions: NSC is the basic implementation, NSC+LA uses local semantic attention and NSC+UPA leverages user product attention.

We evaluate all models using classification accuracy, which is a hard mapping from the predicted label to the sentiment class and reflects the practical performance of a system. For our models, we apply grid search to find the optimal parameter settings that maximize accuracy on the development set, which are: dropout=0.5, kernel-number=200, learning-rate=1e-3 and batch-size=32. We train

\(^{2}\)This setup avoids the cold-start problem, however the focus of our work is on improving personalization, which requires historical user information.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN</td>
<td>66.37%</td>
</tr>
<tr>
<td>P-CNN</td>
<td>67.15% (+1.1%)</td>
</tr>
<tr>
<td>- rating feature (lm-only)</td>
<td>66.77% (+0.6%)</td>
</tr>
<tr>
<td>- lm feature (rating-only)</td>
<td>67.09% (+1.1%)</td>
</tr>
<tr>
<td>CNN-LSTM</td>
<td>67.26% (+1.7%)</td>
</tr>
<tr>
<td>- rating feature (lm-only)</td>
<td>66.04% (-0.2%)</td>
</tr>
<tr>
<td>- lm feature (rating-only)</td>
<td>67.04% (+1.3%)</td>
</tr>
</tbody>
</table>

Table 2: Feature ablation with improvement using personalization in parenthesis.

4.3 Feature Ablation Analysis
We investigate the effectiveness of our proposed personalized features by studying the performance in isolation and combined. Table 2 shows the results for our models on the Yelp18-U dataset. The models CNN and CNN-LSTM have no personalization component and perform worse than the personalized models. Surprisingly, CNN can outperform CNN-LSTM, even though it does not explicitly obtain a sentence representation, but the personalized CNN-LSTM model outperforms the personalized CNN model. The improvement of the lm features alone is smaller than the rating features (for P-CNN-LSTM it actually hurts performance). However, when lm is added to the rating features it is still able to improve overall, as they boost performance for both P-CNN and P-CNN-LSTM models. To this end, we conclude that both personalized features are effective and the highest additive benefit can be achieved when combined.

4.4 Comparison to Baselines
In Table 3, we compare our models to the baselines. We find that our CNN and CNN-LSTM models achieve comparable results with the NSC model, even though the NSC model has a more complex architecture (i.e., hierarchical LSTM). Adding user features improves performance for the baseline NSC+LA/UPA models and our P-CNN(-LSTM) models. Both of our personalized models outperform the best personalized baseline model. In addition, the P-CNN-LSTM model can slightly better leverage the user features (1.3% improvement), compared to the P-CNN model (1.1% improvement). Even though our models make use of a user’s history, it does not entail that these models cannot handle unseen users. In a real-word setting, the model’s performance would fall within the boundaries of the personalized and non-personalized models, depending on how many users with or without history are observed.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN</td>
<td>66.37%</td>
</tr>
<tr>
<td>P-CNN</td>
<td>67.15% (+1.1%)</td>
</tr>
<tr>
<td>- rating feature (lm-only)</td>
<td>66.77% (+0.6%)</td>
</tr>
<tr>
<td>- lm feature (rating-only)</td>
<td>67.09% (+1.1%)</td>
</tr>
<tr>
<td>CNN-LSTM</td>
<td>67.26% (+1.7%)</td>
</tr>
<tr>
<td>- rating feature (lm-only)</td>
<td>66.04% (-0.2%)</td>
</tr>
<tr>
<td>- lm feature (rating-only)</td>
<td>67.04% (+1.3%)</td>
</tr>
</tbody>
</table>

Table 3: Comparison to baselines.

4.5 Case Study
We perform a case study and investigate individual examples in our data that have a large difference between the predicted and the actual review score. We find that most examples fall within one of the following categories:

- word2vec [20] word embeddings on the Yelp dataset. User-specific features are estimated on the training set only. Thus, as long as the user is seen in the training data, the method can leverage the personalized features directly. If a user is previously unseen, their personalized feature values would be zero.
We presented a novel personalized approach for sentiment analysis. Customers of a business state that their satisfaction seed words in an embedding space, trained on informal text. One promising direction is to look at the nearest neighbors of sentiment dictionaries and the informal language used in social media. Overall performance and beat other personalized baselines.

### Table 3: Improvement over best non-personalized baseline (NSC) for CNN and CNN-LSTM and best personalized baselines (NSC+UPA) for P-CNN and P-CNN-LSTM.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPNN (Tang et al. [22])</td>
<td>59.11%</td>
</tr>
<tr>
<td>NSC (Chen et al. [7])</td>
<td>66.06%</td>
</tr>
<tr>
<td>NSC+LA (Chen et al. [7])</td>
<td>66.17%</td>
</tr>
<tr>
<td>NSC+UPA (Chen et al. [7])</td>
<td>66.42%</td>
</tr>
<tr>
<td>CNN</td>
<td>66.37% (+0.5%)</td>
</tr>
<tr>
<td>P-CNN</td>
<td>67.14% (+1.1%)</td>
</tr>
<tr>
<td>CNN-LSTM</td>
<td>66.15% (+0.1%)</td>
</tr>
<tr>
<td>P-CNN-LSTM</td>
<td>67.26% (+1.3%)</td>
</tr>
</tbody>
</table>

Misinterpreted sarcasm. Reviews that use positive sentiment words in an ironic way can mislead classification, e.g., “wow! the carne asada taco tastes like cheap steak.”

Times are changing. Customers of a business state that their service used to be great, but now it’s bad, e.g., “Everything started out awesome with UNK UNK then things changed [...]”

Disaster avoided. Reviewers spend most of the review complaining about something that went wrong but give a good score: “15 minutes for a cold sandwich when no one is in front of you is too long! [...] Thank you Kim for making things right!”

In these extreme cases, we find that there is often a part of the review that sounds extremely negative or positive while the review as a whole has the opposite rating.

### 5 CONCLUSION AND FUTURE WORK

We presented a novel personalized approach for sentiment analysis that captures a user’s specific use of sentiment words in a language model and correlates them with her rating behavior. We showed that incorporating personalized sentiment lexicons can improve overall performance and beat other personalized baselines.

For future work, we will explore more rigorous ways to derive key topic words that are explained as a whole has the opposite rating.

### ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1801652.

### REFERENCES