Opinion Recommendation Using A Neural Model*

Zhongqing Wang[†,‡] and **Yue Zhang**[‡]
[†]Soochow University, Suzhou, China
[‡]Singapore University of Technology and Design, Singapore
wangzq.antony@gmail.com, yue_zhang@sutd.edu.sg,

Abstract

We present opinion recommendation, a novel task of jointly generating a review with a rating score that a certain user would give to a certain product which is unreviewed by the user, given existing reviews to the product by other users, and the reviews that the user has given to other products. A characteristic of opinion recommendation is the reliance of multiple data sources for multi-task joint learning. We use a single neural network to model users and products, generating customised product representations using a deep memory network, from which customised ratings and reviews are constructed jointly. Results show that our opinion recommendation system gives ratings that are closer to real user ratings on Yelp.com data compared with Yelp's own ratings. our methods give better results compared to several pipelines baselines.

1 Introduction

Offering a channel for customers to share opinions and give scores to products and services, review websites have become a highly influential information source that customers refer to for making purchase decisions. Popular examples include IMDB.com on the movie domain, Epinions.com on the product domain, and Yelp.com on the service domain. Figure 1 shows a screenshot of a restaurant review page on Yelp.com, which offers two main types of information. First, an overall rating score is given under the restaurant name; second, detailed user reviews are listed below the rating.

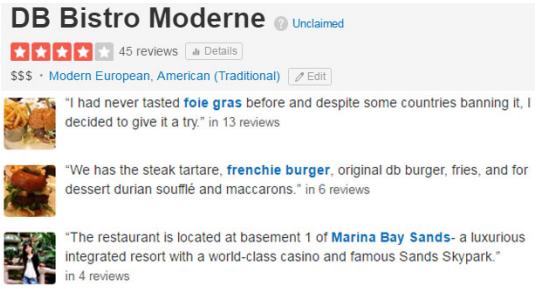

Figure 1: A restaurant review on Yelp.com.

Though offering useful overview and details about a product or service, such information has several limitations for a user who has *not* used the product or service. First, the overall rating is general and not necessarily agreeable to the taste of an individual customer. Being a simple reflection of all customer scores, it serves an average customer well, but can be rather inaccurate for individuals. For example, the authors themselves often find highly rated movies being tedious. Second, there can be hundreds of reviews for a product or service, which makes it infeasible for exhaustive reading. It would be useful to have a brief summary of all reviews, which ideally should be customized to the reader.

To address the limitations above, we propose a new task called **opinion recommendation**, which is to generate a *customized review score* of the product that the user is likely to give, as well as a *customized review* that the user would have written for the target product, if the user had reviewed the product. The proposed opinion recommendation task is closely related to several existing lines of work in NLP. The first is *sentiment analysis* (Hu and Liu, 2004; Pang and Lee, 2008) and *opinion summarization* (Nishikawa et al., 2010; Wang and Ling, 2016), which is to give a rating score or summary based on existing customer reviews. Our

*This work has been done when the first author worked at SUTD.

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1627–1638
Copenhagen, Denmark, September 7–11, 2017.

task is different in that we aim to generate user rating scores and review of a product unreviewed by the user. The second is *recommendation* (Su and Khoshgoftaar, 2009; Yang et al., 2014), which is to give a ranking score to a certain product or service based on the purchase history of the user and other customers who have purchased the target product. Our task is different in the source of input, which is *textual* customer reviews and ratings rather than *numerical* purchase history.

There are two types of inputs for our task, namely existing reviews of the target product, and the reviews of the user on other products, and two types of outputs, namely a customized rating score and a customized review. The ideal solution should consider the interaction between all given types of information, jointly predicting the two types of outputs. This poses significant challenges to statistical models, which require manually defined features to capture relevant patterns from training data. Deep learning is a relatively more feasible choice, offering viabilities of information fusion by fully connected hidden layers (Collobert et al., 2011; Henderson et al., 2013; Zhang and Weiss, 2016; Chen et al., 2016a). We leverage this advantage in building our model.

In particular, we use a sub RNN to model the semantic content of each review. A sub *product model* is used to consolidate existing reviews for the target product, and a *user model* is built by consolidating the reviews of the given user into a single vector form. To address potential sparsity of a user's history reviews, neighbor users are identified by collaborative filtering (Ding et al., 2006), and a vector representation is learned by using a neural *neighborhood model*. Finally, a deep memory network is utilized to find the association between the user and target product, jointly yielding the rating score and customised review. Experiments on a Yelp dataset show that the model outperforms several pipelined baselines. We make our source code publicly available under GPL at `https://github.com/wangzq870305/opinion_recommend`.

2 Related Work

Sentiment Analysis. Our task is related to document-level sentiment classification (Pang and Lee, 2008) for various neural network models have been used, including convolutional neural networks (Kim, 2014), recursive neural network (Socher et al., 2013) and recurrent neural network (Teng et al., 2016; Tai et al., 2015), Review rating prediction aims to predict the numeric rating of a given review. Pang and Lee (2005) pioneered this task by regarding it as a classification/regression problem. Most subsequent work focuses on designing effective textural features of reviews (Qu et al., 2010; Li et al., 2011; Wan, 2013).

User information has been widely investigated in sentiment analysis. Gao et al. (2013) developed user-specific features to capture user leniency, and Li et al. (2014) incorporated textual topic and user-word factors through topic modeling. For integrating user information into neural network models, Tang et al. (2015) predicted the rating score given a review by using both lexical semantic information and a user embedding model. Chen et al. (2016b) proposed a neural network to incorporate global user and product information for sentiment classification via an attention mechanism.

Different from the above research, which focuses on predicting the opinion on existing reviews, our task is to recommend the score that a user would give to a new product *without* knowing his review text. The difference originates from the objective. Previous research aims to predict opinions on reviewed products, while our task is to recommend opinion on new products, which the user has not reviewed.

Opinion Summarization. Our work also overlaps with to the area of opinion summarization, which constructs natural language summaries for multiple product reviews (Hu and Liu, 2004). Most previous work extracts opinion words and aspect terms. Typical approaches include association mining of frequent candidate aspects (Hu and Liu, 2004; Qiu et al., 2011), sequence labeling based methods (Jakob and Gurevych, 2010; Yang and Cardie, 2013), as well as topic modeling techniques (Lin and He, 2009). Recently, word embeddings and recurrent neural networks are also used to extract aspect terms (Irsoy and Cardie, 2014; Liu et al., 2015). While all the methods above are *extractive*, Ganesan et al. (2010) presented a graph-based summarization framework to generate concise abstractive summaries of highly redundant opinions, and Wang and Ling (2016) used an attention-based neural network model to absorb information from multiple text units and generate summaries of movie reviews. We also

perform abstractive summarization. However, different from the above research, which summarize existing reviews, we generate *customized* reviews for a unreviewed product.

Recommendation. has been solved on mainly purchase history. There are two main approaches, which are content-based and collaborative-filtering (CF) based (Adomavicius and Tuzhilin, 2005; Yang et al., 2014), respectively. Most existing social recommendation systems are CF-based, and can be further grouped into model-based CF and neighborhood-based CF (Kantor et al., 2011; Su and Khoshgoftaar, 2009). Matrix Factorization (MF) is one of the most popular models for CF. In recent MF-based social recommendation works, user-user social trust information is integrated with user-item feedback history (e.g., ratings, clicks, purchases) to improve the accuracy of traditional recommendation systems, which only factorize user-item feedback data (Ding et al., 2006; Koren, 2008; He et al., 2016).

There has been work integrating sentiment analysis and recommendation systems, which use recommendation strategies such as matrix factorization to improve the performance of sentiment analysis (Leung et al., 2006; Singh et al., 2011). These methods typically use ensemble learning (Singh et al., 2011) or probabilistic graph models (Wu and Ester, 2015). For example, Zhang et al. (2014) proposed a factor graph model to recommend opinion rating scores by using explicit product features as hidden variables. Different from the above research, we recommend user opinions.

Neural Network Models. Multi-task learning has been recognised as a strength of neural network models for natural language processing (Collobert et al., 2011; Henderson et al., 2013; Zhang and Weiss, 2016; Chen et al., 2016a), where hidden feature layers are shared between different tasks that have common basis. Our work can be regarded as an instance of such multi-tasks learning via shared parameters, which has been widely used in the research community recently.

Dynamic memory network models have been applied for NLP tasks such as question answering (Sukhbaatar et al., 2015; Kumar et al., 2016), language modeling (Tran et al., 2016) and machine translation (Wang et al., 2016). There are typically used to find abstract semantic representations of texts towards certain tasks, which are consistent with our main need, namely abstracting the representation of a product that is biased towards the taste of a certain user. We use a variation of the memory network model for obtaining user-specific review representation.

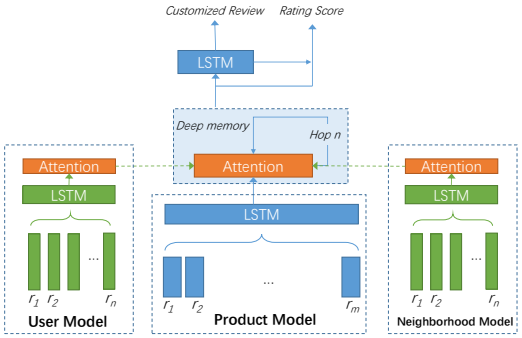

Figure 2: Overview of proposed model.

3 Model

Formally, the input to our model is a tuple $\langle R_T, R_U, R_N \rangle$, where $R_T = \{r_{T_1}, r_{T_2}, ..., r_{T_{n_t}}\}$ is the set of existing reviews of a target product, $R_U = \{r_{U_1}, r_{U_2}, ..., r_{U_{n_u}}\}$ is the set of user's history reviews, and $R_N = \{r_{N_1}, r_{N_2}, ..., r_{N_{n_n}}\}$ is the set of the user's neighborhood reviews. All the reviews are sorted with temporal order. The output is a pair $\langle Y_S, Y_R \rangle$, where Y_S is a real number between 0 and 5 representing the customized rating score of the target product, and Y_R is a customised review. A characteristic of our model is that Y_S and Y_R are generated on a product that the user has not reviewed.

For capturing both general and personalized information, we first build a *product model*, a *user model*, and a *neighborhood model*, respectively, and using a memory network model to integrate these three types of information, constructing a *customized product model*. Finally, we predict a customized rating score and a review collectively using neural stacking framework. The overall architecture of the model is shown in Figure 2.

3.1 Review Model

A review is the foundation of our model, based on which we derive representations of both a user and a target product. In particular, a user profile can be achieved by modeling all the reviews R_U of the user, and a target product profile can be obtained by using all existing reviews R_T of the product. We use the average of word embeddings to model a review. Formally, given a review $r = \{x_1, x_2, ..., x_m\}$, where m is the length of

the review, each word x_k is represented with a K-dimensional embedding e_k^w (Mikolov et al., 2013). We use the $\sum_k(e_k^w)/m$ for the representation of the review $e^d(r)$.

3.2 User Model

A standard LSTM (Hochreiter and Schmidhuber, 1997) is used to learn the hidden states of an user's reviews to build the user model. Denoting the recurrent function at step t as $\text{LSTM}(x_t, h_{t-1})$, we obtain a sequence of hidden state vectors $\{h_{U_1}, h_{U_2}, ..., h_{U_{n_u}}\}$ recurrently by feeding $\{e^d(r_{U_1}), e^d(r_{U_2}), ..., e^d(r_{U_{n_u}})\}$ as inputs, where $h_{U_i} = \text{LSTM}(e^d(r_{U_i}), h_{U_{i-1}})$. The initial state and all standard LSTM parameters are randomly initialized and tuned during training.

Not all reviews contribute equally to the representation of a user. We use the attention mechanism (Bahdanau et al., 2014; Yang et al., 2016) to extract the reviews that are relatively more important, aggregating the representation of reviews to form a vector. Taking the hidden state $\{h_{U_1}, ...h_{U_2}, ..., h_{U_{n_u}}\}$ of user model as input, the attention model outputs, a continuous vector $v_U \in \mathbb{R}^{d\times 1}$, which is computed as a weighted sum of each hidden state h_{U_i}, namely

$$v_U = \sum_i^{n_u} \alpha_i h_{U_i} \tag{1}$$

where n_u is the hidden variable size, $\alpha_i \in [0, 1]$ is the weight of h_{U_i}, and $\sum_i \alpha_i = 1$.

For each piece of hidden state h_{U_i}, the scoring function is calculated by

$$u_i = \tanh(W_U h_{U_i} + b_U) \tag{2}$$

$$\alpha_i = \frac{\exp(u_i)}{\sum_j \exp(u_j)} \tag{3}$$

where W_U and b_U are model parameters. The attention vector v_U is used to represent the user for the *User Model*.

3.3 Neighborhood Model

We use neighborhood reviews to improve the user model, since a user may not have sufficient reviews to construct a reliable model. Here a neighbor refers to a user that has similar tastes to the target user (Koren, 2008; Desrosiers and Karypis, 2011). The same as the user model, we construct the *neighborhood model* v_N using the neighborhood reviews $R_N = \{r_{N_1}, r_{N_2}, ..., r_{N_{n_n}}\}$ with an attention recurrent network.

A key issue in building the neighborhood model is how to find neighbors of a certain user. In this study, we use matrix factorization (Koren, 2008) to detect neighbors, which is a standard approach for recommendation (Ding et al., 2006; Li et al., 2009; He et al., 2016). In particular, users' rating scores of products are used to build a product-users matrix $M \in \mathbb{R}^{n_t \times n_u}$ with n_t products and n_u users. We approximate it using three factors, specifying soft membership of products and users (Ding et al., 2006) by finding:

$$\begin{aligned} &\min_{F,S,T} ||M - FST^T|| \\ &s.t. S \geq 0, F \geq 0, T \geq 0 \end{aligned} \tag{4}$$

where $F \in \mathbb{R}^{n_t \times K}$ represents the posterior probability of K topic clusters for each product; $S \in \mathbb{R}^{K\times K}$ encodes the distribution of each topic k; and $T \in \mathbb{R}^{K\times n_u}$ indicates the posterior probability of K topic clusters for each user.

As a result of matrix factorization, we directly obtain the probability of each user on each topic from the person-topic matrix T. To infer T, the optimization problem in Eq.4 can be solved using the following updating rule:

$$T_{jk} \leftarrow T_{jk} \frac{(M^T FS)_{jk}}{(TT^T M^T FS)_{jk}} \tag{5}$$

With the user-topic matrix T, we measure the implicit connection between two users using:

$$sim(i, j) = \sum_{k=1}^{k} T_{ik} T_{jk} \tag{6}$$

where $sim(i, j)$ measure the implicit connection degree between users i and j. If $sim(i, j)$ is higher than a threshold η, we consider user j as the neighbor of user i.

3.4 Product Model

Given the representations of existing reviews $\{e^d(r_{T_1}), e^d(r_{T_2}), ..., e^d(r_{T_{n_t}})\}$ of the product, we use LSTM to model their temporal orders, obtaining a sequence of hidden vectors $h_T = \{h_{T_1}, h_{T_2}, ..., h_{T_{n_t}}\}$ by recurrently feeding $\{e^d(r_{T_1}), e^d(r_{T_2}), ..., e^d(r_{T_{n_t}}\}$ as inputs. The hidden state vectors h_T are used to represent the product.

Customized Product Model. The product model represents salient information of existing reviews in their temporal order, yet do not reflect the taste of a particular user. We build the customised product model to integrate user information and product information (as reflected by the product model), resulting in a single vector that represents a customised product. From this vector we are able to synthesis both a customised review and a customised rating score. In particular, we use the user representation v_U and the neighbour representation v_N to transform the target product representation $h_T = \{h_{T_1}, h_{T_2}, ..., h_{T_{n_t}}\}$ into a customised product representation v_C, which is tailored to the taste of the user.

A naive model of yielding v_C could utilise the attention mechanism over h_t, deriving a weighted sum according to user information. On the other hand, dynamic memory networks have been shown highly useful for deriving abstract semantic information compared with simple attention, and hence we follow Sukhbaatar et al. (2015) and Xiong et al. (2016), building a variation of DMN to iteratively find increasingly abstract representations of h_t, by injecting v_U and v_N information.

The memory model consists of multiple dynamic computational layers (hops), each of which contains an attention layer and a linear layer. In the first computational layer (hop 1), we take the hidden variables h_{T_i} $(0 \leq i \leq n_t)$ of product model as input, adaptively selecting important evidences through one attention layer using v_U and v_N. The output of the attention layer gives a linear interpolation of h_T, and the result is considered as input to the next layer (hop 2). In the same way, we stack multiple hops and run the steps multiple times, so that more abstract representations of the target product can be derived.

The attention model outputs a continuous vector $v_C \in \mathbb{R}^{d \times 1}$, which is computed as a weighted sum of h_{T_i} $(0 \leq i \leq n_t)$, namely

$$v_C = \sum_{i}^{n_t} \beta_i h_{T_i} \tag{7}$$

where n_t is the hidden variable size, $\beta_i \in [0, 1]$ is the weight of h_{T_i}, and $\sum_i \beta_i = 1$. For each piece of hidden state h_{T_i}, we use a feed forward neural network to compute its semantic relatedness with the abstract representation v_C. The scoring function is calculated as follows at hop t:

$$\begin{aligned} u_i^t = \tanh(&W_T h_{T_i} + W_C v_C^{t-1} \\ &+ W_U v_U + W_N v_N + b) \end{aligned} \tag{8}$$

$$\beta_i^t = \frac{\exp(u_i^t)}{\sum_j \exp(u_j^t)} \tag{9}$$

The vector v_C is used to represent the customized product model. At the first hop, we define $V_C^0 = \sum_i h_{T_i}/n_t$.

The product model h_{T_i} $(0 \leq i \leq n_t)$ represents salient information of existing reviews in their temporal order, they do not reflect the taste of a particular user. We use the customised product model to integrate user information and product information (as reflected by the product model), resulting in a single vector that represents a customised product. From this vector we are able to synthesis both a customised review and a customised rating score.

3.5 Customized Review Generation

The goal of customized review generation is to generate a review Y_R from the customized product representation v_C, composed by a sequence of words $y_{R_1}, ..., y_{R_{n_r}}$. We use a standard LSTM decoder (Rush et al., 2015) to decompose the prediction of Y_R into a sequence of word-level predictions:

$$\begin{aligned} &\log P(Y_R|v_C) = \\ &\sum_j P(y_{R_j}|y_{R_1}, ..., y_{R_{j-1}}, v_C) \end{aligned} \tag{10}$$

where each word y_{R_j} is predicted conditional on the previously generated $y_{R_1}, ..., y_{R_{j-1}}$ and the customized product vector v_C. The probability is estimated by using standard word softmax:

$$\begin{aligned} &P(y_{R_j}|y_{R_1}, ..., y_{R_{j-1}}, v_C) = \\ &\qquad \text{softmax}(h_{R_j}) \end{aligned} \tag{11}$$

where h_{R_j} is the hidden state variable at timestamp j, which is modeled as $LSTM(u_{j-1}, h_{Rj})$. Here a LSTM is used to generate a new state h_{R_j} from the representation of the previous state $h_{R_{j-1}}$ and u_{j-1}. u_{j-1} is the concatenation of previously generated word $y_{R_{j-1}}$ and the input representation of customized model v_C.

3.6 Customized Rating Prediction

A straightforward approach to predicting the rating score of a product is to take the average of existing review scores. However, the drawback is that it cannot reflect the the variance in user tastes. In order to integrate user preferences into the rating, we instead take a user-based weighted average of existing rating scores, so that the scores of reviews that are closer to the user preference are given higher weights. However, existing ratings can be all different from a users personal rating, if the existing reviews do not come from the user's neighbours. We thus use the customized product vector v_c as a bias of the weighted average of existing rating scores.

Formally, given the rating scores $s_1, s_2, ..., s_n$ of existing reviews, and the the customized product representation v_C, we calculate:

$$Y_S = \sum_i^n \beta_i \cdot s_i + \mu \tanh(W_S v_C + b_S) \quad (12)$$

In the left term $\sum_i^n \beta_i \cdot s_i$, we use attention weights β_i in Eq.9 to measure the important of each rating score s_i. The right term $\tanh(W_S v_C + b_S)$ is a review-based shift, weighted by μ.

Since the result of customized review generation can be helpful for rating score prediction, we use neural stacking additionally feeding the last hidden state h_{R_n} of review generation model as input for Y_S prediction, resulting in

$$\begin{aligned} Y_S = &\sum_i^n \alpha_i \cdot s_i + \\ &+ \mu \tanh(W_S(v_C \oplus h_{R_n}) + b_S) \end{aligned} \quad (13)$$

where $\oplus$ denotes vector concatenation.

3.7 Training

For our task, there are two joint training objectives, for review scoring and review summarisation, respectively. For review scoring, the loss function is defined as:

$$L(\Theta) = \sum_{i=1}^{N} (Y_{S_i}^* - Y_{S_i})^2 + \frac{\lambda}{2}||\Theta||^2 \quad (14)$$

where $Y_{S_i}^*$ is the predicted rating score, Y_{S_i} is the rating score in the training data, Θ is the set of model parameters and λ is a parameter for L2 regularization.

	Amount
Business	15,584
Review	334,997
User	303,032

Table 1: Statistics of the dataset.

For customized review generation, loss is defined by maximizing the log probability of Eq.10 (Sutskever et al., 2014; Rush et al., 2015). The two loss functions for score and review prediction share the representation vectors under v_C, hence forming multi-task learning.

Standard back propagation is performed to optimize parameters, where gradients also propagate from the scoring objective to the review generation objective due to neural stacking (Eq.13). We apply online training, where model parameters are optimized by using AdaGrad (Duchi et al., 2011). Word embeddings are trained using the *Skip-gram* algorithm (Mikolov et al., 2013)[1].

4 Experiments

4.1 Experimental Settings

Our data are collected from the yelp academic dataset[2], provided by Yelp.com, a popular restaurant review website. The data set contains three types of objects: *business*, *user*, and *review*, where business objects contain basic information about local businesses (i.e. restaurants), review objects contain review texts and star rating, and user objects contain aggregate information about a single user across all of Yelp. Table 1 illustrates the general statistics of the dataset.

For evaluating our model, we choose 4,755 user-product pairs from the dataset. The user-product pairs are extracted by following criterions: for each selected user-product pair, the user should have written 10 reviews at least, and the product should contain 100 reviews at least. In addition, the gold-standard review that the user write for the corresponding product should contain 10 helpful hits at least. We did not try alternative data selection rules. We will give the detail in our draft.

For each pair, the existing reviews of the target service (restaurant) are used for the product model. The rating score given by each user to the target service is considered as the gold customized rating score, and the review of the target service given by

[1] https://code.google.com/p/word2vec/

[2] https://www.yelp.com/academic_dataset

each user is used as the gold-standard customized review for the user. The remaining reviews of each user are used for training the user model. We use 3,000 user-product pairs to train the model, 1,000 pairs as testing data, and remaining data for development.

We use the ROUGE-1.5.5 (Lin, 2004) toolkit for evaluating the performance of customized review generation, and report unigram overlap (ROUGE-1) as a means of assessing informativeness. Mean Square Error (MSE) (Wan, 2013; Tang et al., 2015) is used as the evaluation metric for measuring the performance of customized rating score prediction. MSE penalizes more severe errors more heavily.

4.2 Hyper-parameters

There are several important hyper-parameters in our models, and we tune their values using the development dataset. We set the regularization weight $\lambda = 10^{-8}$ and the initial learning rate to 0.01. We set the size of word vectors to 128, the size of hidden vectors in LSTM to 128. In order to avoid over-fitting, dropout (Hinton et al., 2012) is used for word embedding with a ratio of 0.2. The neighbor similarity threshold η is set to 0.25.

4.3 Development Experiments

4.3.1 Ablation Test

Effects of various configurations of our model, are shown on Table 2, where *Joint* is the full model of this paper, *-user* ablates the user model, *-neighbor* ablates the neighbor model, *-rating* is a single-task model that generates a review without the rating score, and *-generation* yields only the rating.

By comparing "Joint" and "-user,-neighbor", we can find that customized information have significant influence on both the rating and review generation results ($p - value < 0.01$ using t-test). In addition, comparison between "-Joint" and "-user", and between "-user" and "-user, -neighbor" shows that both the user information and the neighbour user information of the user are effective for improving the results. A users neighbours can indeed alleviate scarcity of user reviews.

Finally, comparison between "Joint" and "-generation", and between "Joint" and "-rating" shows that multi-task learning by parameter sharing is highly useful.

	Rating	Generation
Joint	0.904	0.267
-user	1.254	0.220
-neighbor	1.162	0.245
-user,-neighbor	1.342	0.205
-rating	-	0.254
-generation	1.042	-

Table 2: Feature ablation tests.

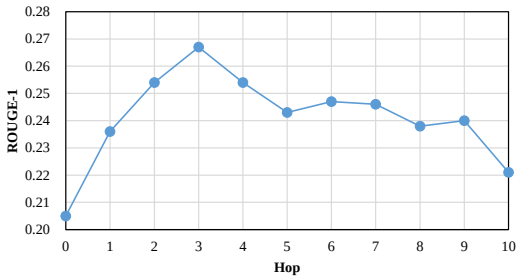

Figure 3: Influence of hops.

4.3.2 Influence of Hops

We show the influence of hops of memory network for customized review generation on Figure 3. When $hop = 0$, the model considers only the general product reviews ($-user, -neighbor$). When $hop \geq 1$, customized product information is leveraged. From the figure we can find that, when $hop = 3$, the performance is the best. It indicates that multiple hops can capture more abstract evidences from external memory to improve the performance. However, too many hops leads to over-fitting, thereby harms the performance. As a result, we choose 3 as the number of hops in our final test.

4.3.3 Influence of μ

We show the influence of the bias weight parameter μ for rating prediction in Figure 4. With μ being 0, the model uses the weighted sum of existing reviews to score the product. When μ is very large, the system tends to use only the customized product representation v_c to score the product, hence ignoring existing review scores, which are a useful source of information. Our results show that when μ is 1, the performance is optimal, thus indicating both existing review scores and review contents are equally useful.

4.4 Final Results

We show the final results for opinion recommendation, comparing our proposed model with the

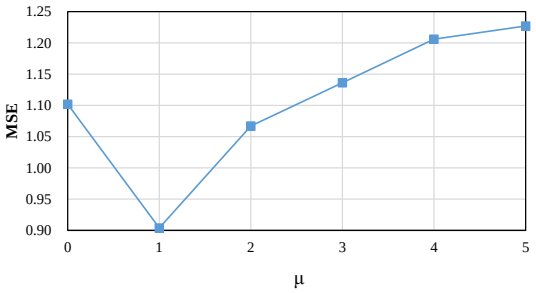

Figure 4: Influence of bias score.

	Rating	Generation
RS-Average-Yelp	1.280	-
RS-Linear	1.234	-
RS-Item	1.364	-
RS-MF	1.143	-
Sum-Opinosis	-	0.183
Sum-LSTM-Att	-	0.196
Joint	**1.023**	**0.250**

Table 3: Final results.

following state-of-the-art baseline systems:

- *RS-Average-Yelp* is the widely-adopted baseline (e.g., by Yelp.com), using the averaged review scores as the final score.
- *RS-Linear* estimates the rating score that a user would give by $s_{ui} = s_{all} + s_u + s_i$ (Ricci et al., 2011), where s_u and s_i are the the training deviations of rating score of the user u and the product i, respectively.
- *RS-Item* applies kNN to estimate the rating score (Sarwar et al., 2001). We choose the cosine similarity between v_c to measure the distance between product.
- *RS-MF* is a state-of-the-art recommendation model, which uses matrix factorisation to predict rating score (Ding et al., 2006; Li et al., 2009; He et al., 2016).
- *Sum-Opinosis* uses a graph-based framework to generate abstractive summarisation given redundant opinions (Ganesan et al., 2010).
- *Sum-LSTM-Att* is a state-of-the-art neural abstractive summariser, which uses an attentional neural model to consolidate information from multiple text sources, generating summaries using LSTM decoding (Rush et al., 2015; Wang and Ling, 2016).

Being non-opinion recommendation methods, all the baselines are single-task models, without considering rating and summarisation prediction jointly. The results are shown in Table 3. Our model (" Joint") significantly outperforms both "RS-Average-Yelp" and "RS-Linear" ($p - value < 0.01$ using t-test). Note that, our proposed rating recommendation for the user are significantly closer individual real user rating compared with Yelp's rating.

Our proposed model also significantly outperforms state-of-the-art recommendation systems (RS-Item and RS-MF) ($p - value < 0.01$ using t-test), indicating that textual information are a useful addition to the rating scores themselves for recommending a product.

Finally, comparison between our proposed model and state-of-the-art summarisation techniques (Sum-Opinosis and Sum-LSTM-Att) shows the advantage of leveraging user information to enhance customised review generation, and also the strength of joint learning.

4.5 Example Output

Table 4 shows example outputs of rating scores and reviews. *Ref.* is the rating score and review written by user her/himself, and *Base* is the baseline model, that generates the rating score by RS-MF, and review by Sum-LSTM-Att. From these examples, we can find that, both rating score and review which generated by the proposed Joint model is closer to the real user. In particular, in the first example, the baseline system correctly identifies the main both price and quality information, which the target user wrote in the review, yet the baseline model did not yield comments about the price based on reviews of other users. Associating reviews and ratings closely, the joint model gives a rating score that is much closer to the real user score compared to the score given by the recommendation model MF. In addition, we can also find some habits of certain users from their customized reviews, for example, Mexican food, cheap and clean restaurant.

5 Conclusion

We proposed a novel task called opinion recommendation, which is to generate the review and rating score that a certain user would give to an unreviewed product or service. In particular, a

	Rating	Review
Ref.	5.0	Amazing beer selection, enough food choices, and a much smaller bill than I was expecting ...
Base	4.0	Boulders is JAM, favorite neighborhood bar, have amazing food ...
Joint	4.6	Bar is cheap, food is good enough ...
Ref.	4.0	This is one of my favorite Mexican fast food restaurants. It's clean and cool in the summer...
Base	5.0	The restaurant is great, This Chipotle is a great location, Their medium salsa good ...
Joint	4.2	Mexican food my favorite, place is clean ...

Table 4: Example outputs.

deep memory network was utilized to find the association between the user and the product, jointly yielding the rating score and customised review. Results show that our methods are better results compared to several pipelines baselines using state-of-the-art sentiment rating and summarisation systems. Review scores given by the opinion recordation system are closer to real user review scores compared to the review scores which Yelp assigns to target products.

Acknowledgments

The corresponding author is Yue Zhang. We are grateful for the help of Xuanyi Li for his initial exploration. We thank our anonymous reviewers for their constructive comments, which helped to improve the paper. This work is supported by the Temasek Lab grant IGDST1403012 at Singapore University of Technology and Design, and supported by the National Natural Science Foundation of China (No.61402314).

References

Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Hongshen Chen, Yue Zhang, and Qun Liu. 2016a. Neural network for heterogeneous annotations. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 731–741.

Huimin Chen, Maosong Sun, Cunchao Tu, Yankai Lin, and Zhiyuan Liu. 2016b. Neural sentiment classification with user and product attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1650–1659.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Christian Desrosiers and George Karypis. 2011. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender Systems Handbook*, pages 107–144.

Chris Ding, Tao Li, Wei Peng, and Haesun Park. 2006. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 126–135. ACM.

John C. Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159.

Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd international conference on computational linguistics*, pages 340–348. Association for Computational Linguistics.

Wenliang Gao, Naoki Yoshinaga, Nobuhiro Kaji, and Masaru Kitsuregawa. 2013. Modeling user leniency and product popularity for sentiment classification. In *IJCNLP*, pages 1107–1111.

Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*, pages 549–558.

James Henderson, Paola Merlo, Ivan Titov, and Gabriele Musillo. 2013. Multilingual joint parsing of syntactic and semantic dependencies with a latent variable model. *Computational Linguistics*, 39(4):949–998.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 168–177.

Ozan Irsoy and Claire Cardie. 2014. Opinion mining with deep recurrent neural networks. In *EMNLP*, pages 720–728.

Niklas Jakob and Iryna Gurevych. 2010. Extracting opinion targets in a single and cross-domain setting with conditional random fields. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP 2010, 9-11 October 2010, MIT Stata Center, Massachusetts, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1035–1045.

Paul B Kantor, Lior Rokach, Francesco Ricci, and Bracha Shapira. 2011. Recommender systems handbook.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751.

Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 1378–1387.

Cane WK Leung, Stephen CF Chan, and Fu-lai Chung. 2006. Integrating collaborative filtering and sentiment analysis: A rating inference approach. In *Proceedings of the ECAI 2006 workshop on recommender systems*, pages 62–66.

Fangtao Li, Nathan Nan Liu, Hongwei Jin, Kai Zhao, Qiang Yang, and Xiaoyan Zhu. 2011. Incorporating reviewer and product information for review rating prediction. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pages 1820–1825.

Fangtao Li, Sheng Wang, Shenghua Liu, and Ming Zhang. 2014. Suit: A supervised user-item based topic model for sentiment analysis. In *AAAI*, pages 1636–1642.

Tao Li, Yi Zhang, and Vikas Sindhwani. 2009. A non-negative matrix tri-factorization approach to sentiment classification with lexical prior knowledge. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 244–252. Association for Computational Linguistics.

Chenghua Lin and Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*, pages 375–384.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8. Barcelona, Spain.

Pengfei Liu, Shafiq R. Joty, and Helen M. Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1433–1443.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119.

Hitoshi Nishikawa, Takaaki Hasegawa, Yoshihiro Matsuo, and Gen-ichiro Kikui. 2010. Optimizing informativeness and readability for sentiment summarization. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden, Short Papers*, pages 325–330.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, pages 115–124.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational Linguistics*, 37(1):9–27.

Lizhen Qu, Georgiana Ifrim, and Gerhard Weikum. 2010. The bag-of-opinions method for review rating prediction from sparse text patterns. In *COLING*

2010, 23rd International Conference on Computational Linguistics, Proceedings of the Conference, 23-27 August 2010, Beijing, China, pages 913–921.

Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors. 2011. *Recommender Systems Handbook*. Springer.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 379–389.

Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001*, pages 285–295.

Vivek Kumar Singh, Mousumi Mukherjee, and Ghanshyam Kumar Mehta. 2011. Combining collaborative filtering and sentiment classification for improved movie recommendations. In *Multidisciplinary Trends in Artificial Intelligence - 5th International Workshop, MIWAI 2011, Hyderabad, India, December 7-9, 2011. Proceedings*, pages 38–50.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*.

Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009:4.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2440–2448.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1556–1566.

Duyu Tang, Bing Qin, Ting Liu, and Yuekui Yang. 2015. User modeling with neural network for review rating prediction. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 1340–1346.

Zhiyang Teng, Duy-Tin Vo, and Yue Zhang. 2016. Context-sensitive lexicon features for neural sentiment analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1629–1638.

Ke M. Tran, Arianna Bisazza, and Christof Monz. 2016. Recurrent memory networks for language modeling. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 321–331.

Xiaojun Wan. 2013. Co-regression for cross-language review rating prediction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 2: Short Papers*, pages 526–531.

Lu Wang and Wang Ling. 2016. Neural network-based abstract generation for opinions and arguments. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 47–57.

Mingxuan Wang, Zhengdong Lu, Hang Li, and Qun Liu. 2016. Memory-enhanced decoder for neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 278–286.

Yao Wu and Martin Ester. 2015. FLAME: A probabilistic model combining aspect based opinion mining and collaborative filtering. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM 2015, Shanghai, China, February 2-6, 2015*, pages 199–208.

Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 2397–2406.

Bishan Yang and Claire Cardie. 2013. Joint inference for fine-grained opinion extraction. In *Proceedings of the 51st Annual Meeting of the Association*

for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers, pages 1640–1649.

Xiwang Yang, Yang Guo, Yong Liu, and Harald Steck. 2014. A survey of collaborative filtering based social recommender systems. *Computer Communications*, 41:1–10.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *NAACL 2016, 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, US*.

Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *The 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, Gold Coast , QLD, Australia - July 06 - 11, 2014*, pages 83–92.

Yuan Zhang and David Weiss. 2016. Stack-propagation: Improved representation learning for syntax. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.