Adversarial Generation of Target Review for Rating Prediction

Huilin Yu[1], Tieyun Qian[1(✉)], Yile Liang[1], and Bing Liu[2]

[1] School of Computer Science, Wuhan University, Hubei, China
{huilin_yu,qty,liangyile}@whu.edu.cn
[2] Department of Computer Science,
University of Illinois at Chicago, Chicago, IL, USA
liub@uic.edu

Abstract. Recent years have witnessed a growing trend of utilizing reviews to improve the performance and interpretability of recommender systems. Almost all existing methods learn the latent representations from the user's and the item's historical reviews, and then combine these two representations for rating prediction. The fatal limitation in these methods is that they are unable to utilize the most predictive review of the target user for the target item since such a review is not available at test time.

In this paper, we propose a novel recommendation model, called GTR, which can *generate the unseen target review with adversarial training for rating prediction*. To this end, we develop a unified framework to combine *the rating tailored generative adversarial nets* (RTGAN) for synthetic review generation and *the neural latent factor module* (NLFM) using the generated target review along with historical reviews for rating prediction. Extensive experiments on four real-world datasets demonstrate that our model achieves the state-of-the-art performance in both rating prediction and review generation tasks.

Keywords: Recommender systems · Review aware recommendation · Generative adversarial network

1 Introduction

A user's rating indicates his/her attitude towards an purchased item. Rating prediction aims to predict the user's ratings on unrated items which may reflect his/her potential interests on these items. Collaborative filtering (CF) approaches, which mainly depend on historical ratings, have aroused great research interests and become the dominant method in recommender systems. As a typical CF technique, matrix factorization (MF) learns the latent features of users and items by decomposing the user-item rating matrix, and then uses these two feature vectors to predict the rating that the user would assign to the item.

MF is the most widely used technique for rating prediction. However, MF based methods suffer from the data sparsity problem and the predicted rating lacks the interpretability on why the user gives high or low scores. To tackle these issues, textual reviews have become a key complementary data source to enhance the performance and interpretation of the rating prediction task [1, 8, 20, 32]. In particular, due to the power of non-linear combination of different types of information, impressive progress has been made by applying deep neural networks to this problem [3, 4, 6, 18, 26, 33].

The pioneering work by Zheng et al. [33] proposed a DeepCoNN model to represent both users and items in a joint manner using all the reviews of users and items. As proven in [3], *the target review*, which is written by the target user for the target item, provides much of the predictive value for rating prediction. The performance of the DeepCoNN model [33] drops severely when the target reviews are omitted. Indeed, the target review usually contains the target user's preference on the target item's attributes or properties and is closely related to the rating score. However, the target review will not be available at test time in real-world recommendation settings. The hereafter studies along this line do not access the target reviews in the validation and test set at any time to simulate a real world scenario. Clearly, the inherent limitation in these methods is that they are unable to utilize the most predictive target review.

In light of this, we propose a novel framework, namely GTR, to generate the target review for rating prediction. Our model has two distinguishing characteristics. Firstly, we *generate the target review with rating tailored generative adversarial nets (RTGAN)* which incorporates the rating into its objective function in addition to the user's and the item's historical reviews. Secondly, we *develop a neural latent factor module (NLFM) to accurately predict the rating score* by learning from the generated target review which encodes the user's specific preference on the item. In such a way, the target review naturally provides guidance for the rating prediction task beyond the above mentioned review-aware deep recommendation approaches [3, 4, 6, 18, 26]. Meanwhile, the rating drives the RTGAN module to produce a target review conveying consistent sentiment with the rating score.

We are aware of a few existing studies for generating reviews [5, 19, 28] or abstractive tips [15]. However, our GTR model is fundamentally different from the NRT [15], MT [19], and CAML [5] models, in the sense that all these approaches do not directly utilize the target review for rating prediction. Although the neural memory (NM) model proposed by Wang and Zhang [28] also integrates the target review in their prediction step, we distinguish our model with NM in both the review generation and rating prediction modules. We present a conditional GAN architecture for review generation, whereas NM [28] uses the sequence-to-sequence (seq2seq) [24] generative model. More importantly, we design a novel neural latent factor model to stress the target review to make good use of its predictive ability, while NM simply feeds the target review as the input of rating prediction in the last layer.

We have realized the proposed GTR model in both the rating prediction and review generation tasks. Empirical evaluation on four real world datasets proves that our proposed GTR model ahieves the state-of-the-art performance on both tasks.

2 Related Work

We summarize the research progress in review-aware rating prediction, categorized by the traditional methods and deep learning based methods. We omit the classic CF based methods which do not use text reviews.

2.1 Traditional Methods

When integrating review texts, the traditional methods can be roughly classified into three categories. The first one is to extract useful textual information such as topics or aspects from review texts and learn latent factors from ratings, and then link the textual information and latent factors together using linear [2,20,25,31] or Bayesian combination [17,29]. The second one is by extending the latent factor model [7,11,21,22,32] to encode the textual influence. The third one is to modify graphic models to include latent factors from ratings [1,8,27].

2.2 Deep Learning Based Methods

The first type of deep learning based methods only uses historical reviews without generating the target review. These approaches differ mainly in how they combine reviews with ratings. For example, NARRE [4] jointly learns hidden latent features for users and items using two parallel neural networks with the attention mechanism [4]. TARMF [18] adopts a neural network for mutual learning between reviews and ratings, where the features from reviews are optimized by an attention-based GRU network. A^{3}NCF [6] extracts features from reviews using topic models and fuses them with the embeddings from ratings, and it then captures a user's attention on the item with an attention network. MPCN [26] presents a pointer-based co-attention mechanism which can extracts multiple interactions between user and item reviews.

The second type of deep learning based methods generates the target review, but not all of them exploits the predictive ability of the target review. As we have illustrated this issue in the introduction section, here we discuss these methods on how they generate target reviews. NRT [15] is mainly for the purpose of enhancing explainability by generating tips based on a standard generative model with the GRU architecture. NM [28] adopts the seq2seq modeling [24] technique for review generation. Meanwhile, MT [19] uses an adversarial training process which helps overcome the problem of exposure bias in seq2seq models.

Our proposed GTR model falls into the second type of deep learning based methods. Similar to MT [19] in this type, our model also employs GAN for review generation. However, our model incorporates rating as one of the conditions in both the generator and discriminator, whereas MT relies purely on reviews. More importantly, MT adopts a traditional MF method for rating prediction, which does not take the target review into consideration. In contrast, our GTR can fully utilize the target review with a carefully designed neural latent factor model.

3 Problem Definition

This section presents the problem definition and notations. Let $\mathcal{U}$ be a user set and $\mathcal{I}$ be an item set, and $\mathcal{D}$ be a review set on the items in $\mathcal{I}$ written by a set of users in $\mathcal{U}$. Each review d_{ui} written by user u on item i has an accompanying rating r_{ui} indicating u's overall satisfaction towards i. We refer to all historical reviews written by the user, i.e., except that on item i, as the *user's historical review document* d_u. Similarly, the set of historical reviews on item i, except the one written by u, is referred to as the *item's historical review document* d_i. Each training instance is denoted as a sextuple $(u, i, d_{ui}, r_{ui}, d_u, d_i)$. The goal is to predict a rating $\hat{r}_{ui}$ and learn a synthetic target review s_{ui} for each item i that u does not interact with.

For ease of presentation, we summarize the notations in Table 1.

Table 1. Notations used in this paper

Variable	Interpretation
$\mathcal{U}$	User set
$\mathcal{I}$	Item set
$\mathcal{R}$	Rating set
$\mathcal{D}$	Review set
$u \in \mathcal{U}$	A user $u \in \mathcal{U}$
$i \in \mathcal{I}$	An item $i \in \mathcal{I}$
$r_{ui} \in \mathcal{R}$	User u's rating on item i
$d_{ui} \in \mathcal{D}$	User u's review on item i
$d_u \subset \mathcal{D}$	User u's all reviews except d_{ui}
$d_i \subset \mathcal{D}$	Item i's all reviews except d_{ui}
$\hat{r}_{ui}$	User u's predicted rating on item i
s_{ui}	User u's generated review on item i

4 Our Proposed Model

In this section, we introduce our proposed model. We begin with the overall architecture and then go to the details of two modules.

4.1 Model Overview

Our model consists of two modules. One is the rating tailored GAN (RTGAN), which takes the rating as an important condition in the generator and the discriminator of GAN for review generation. The other is the neural latent factor module (NLFM) that leverages the generated target review along with the historical reviews for ration prediction using a neural network. The overall architecture of our model is shown in Fig. 1.

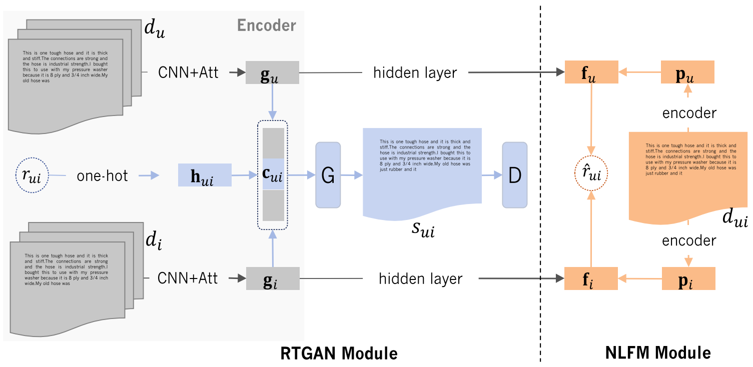

Fig. 1. The architecture of our GTR model

4.2 Rating Tailored GAN (RTGAN) Module

We have two basic assumptions for generating the synthetic target review s_{ui}. Firstly, s_{ui} should reflect the user u's preferences and the item i's features. Secondly, the sentiment expressed in s_{ui} should be consistent with the rating score r_{ui}. Following these assumptions, we design our rating tailored GAN (RTGAN) module conditioned on three types of information: 1) the user's historical review document d_u to capture u's preferences, 2) the item's historical review document d_i to represent i's features, and 3) the rating r_{ui} of the user u to the item i to serve as a constraint. During training, we learn a generator G using three types of condition information to produce a synthetic review, and a discriminator D to distinguish it with the real one.

4.2.1 Condition Information Encoder

We first introduce the condition information encoder (the left grey part in Fig. 1). It maps three types of condition information into user's general preference embedding $\mathbf{g}_u$, item's feature embedding $\mathbf{g}_i$, and the rating embedding $\mathbf{h}_{ui}$.

We take the process of mapping user's review document d_u to his/her preference embedding $\mathbf{g}_u$ as an example. Each word in d_u is randomly initialized as a d dimensional vector, and each review in d_u is transformed into a matrix with the fixed length T (padded with 0 if necessary). Since the text processing is not the focus of this study, we take the same TextCNN [4] approach to encode each review in d_u. Essentially, TextCNN can be summarized as a CNN structure followed by an attention mechanism. The convolution layer consists of m neurons. Each neuron is associated with a filter $\mathbf{K} \in \mathbb{R}^{t \times d}$ which produces features by applying convolution operator on word vectors. Let $\mathbf{V}_{ul}$ be the embedding matrix corresponding to the lth review in d_u, the j^{th} neuron in CNN produces its feature as:

$$\mathbf{z}_j = \sigma(\mathbf{V}_{ul} * \mathbf{K}_j + b_j), \tag{1}$$

where $*$ is convolution operator, b_j is bias term and σ is a nonlinear RELU activation function. We then apply a max-pooling operation to obtain the output feature $\mathbf{o}_j$ corresponding to this neuron. By concatenating the output from all m neurons, the convolution layer can produce the embedding $\mathbf{o}_{ul}$ of the review d_{ul} as:

$$\mathbf{o}_{ul} = [\mathbf{o}_1, \mathbf{o}_2, \mathbf{o}_3, ..., \mathbf{o}_m], \tag{2}$$

After getting the embedding for each review in d_u, the attention mechanism is adopted to get the weights for these reviews. The attention a_{ul} for review d_{ul} is defined as:

$$a^*_{ul} = \mathbf{h}_a^T ReLU(\mathbf{W}_O \mathbf{o}_{ul} + \mathbf{W}_i \mathbf{i}_{ul} + b_1) + b_2, \tag{3}$$

where $\mathbf{h}_a \in \mathbb{R}^t$, $\mathbf{W}_O \in \mathbb{R}^{t \times k_1}$, $\mathbf{W}_i \in \mathbb{R}^{t \times k_2}$, $b_1 \in \mathbb{R}^t$, $b_2 \in \mathbb{R}^1$ are model parameters, $\mathbf{i}_{ul} \in \mathbb{R}^K$ is the embedding of the item which the user write this review for.

A softmax function is used to normalize the above a^*_{ul} to get the final attention a_{ul}. The user's u general preference embedding $\mathbf{g}_u$ is then calculated as the attention weighted sum of all reviews $d_{ul} \in d_u$, i.e.,

$$\mathbf{g}_u = \sum\nolimits_{l=1,...|d_u|} a_{ul}\mathbf{o}_{ul} \tag{4}$$

The process of mapping item's review document d_i to its feature embedding $\mathbf{g}_i$ is all the same. Hence we have:

$$\mathbf{g}_i = \sum\nolimits_{l=1,...|d_i|} a_{il}\mathbf{o}_{il} \tag{5}$$

The mapping from the original rating r_{ui} to an one-hot embedding h_{ui} is straight-forward. We simply discretize the rating r_{ui} into a m-dimension vector ($m = 5$ in our case). If the value falls into an interval, the corresponding dimension is set to 1 and other dimensions are set to 0. For example, a rating $r_{ui} = 3.78$ will be mapped into a $\mathbf{h}_{ui}$ as $(0, 0, 0, 1, 0)^T$. Note that the rating r_{ui} is known only in training. During validation or test, we will use a basic rating from NLFM module instead. The detail will be given later.

4.2.2 RTGAN for Target Review Generation

A good number of generative methods have been proposed for text generation in recent years, such as seq2seq [24] based models, SeqGAN [30], and

RankGAN [16]. Since the reviews are usually long (with average length > 40), we adopt the state-of-the-art LeakGAN [9] model to generate reviews in this paper, and extend it by incorporating three types of condition information into both the generator and the discriminator.

Conditional Generator. Starting from the random state, LeakGAN generates texts via the adversarial generation of synthetic texts against real texts. This implies that, if simply adopting LeakGAN in our model, the generated reviews are only ensured to be written in a human-like style. However, we need to generate the target review that is written by a specific user for a specific item.

In order to provide additional information for guiding the target review generation, we incorporate LeakGAN with the conditional GAN by taking three types of information as the condition of the generator in LeakGAN. We call the combination of these three types of information as a condition vector $\mathbf{c}_{ui}$, and define it as:

$$\mathbf{c}_{ui} = \mathbf{g}_u \oplus \mathbf{g}_i \oplus (\mathbf{W}_r * \mathbf{h}_{ui}), \tag{6}$$

where $\mathbf{W}_r$ is a mapping matrix to transform the sparse $\mathbf{h}_{ui}$ to a dense vector.

Similar to many text generation methods [9,19], we employ a decoder GRU to iteratively generate a review word by word. Different from these methods, the decoder layer in our RTGAN module is conditioned on $\mathbf{c}_{ui}$, which is the combination of three types of information. By doing so, our generator produces a synthetic target review that reflects not only the user u's preferences but also the item i's features. Moreover, the sentiment contained in the synthetic review is also forced to match the rating score.

To ensure that the condition information is maintained during the generation process, the condition vector $\mathbf{c}_{ui}$ is concatenated with the word vector before it is fed into the decoder GRU at each time step. Suppose $\mathbf{x}_t$ is the embedding for the current word being processed at time step t, the concatenated vector $\mathbf{x}_t^{'} = \mathbf{c}_{ui} \oplus \mathbf{x}_t$ is input into the decoder GRU to get the hidden state $\mathbf{h}_t$. And then, the hidden state $\mathbf{h}_t$ is multiplied by an output projection matrix and passed through a softmax over all the words in the vocabulary to obtain the probability of each word in the current context. Finally, the output word y_t at time t is sampled from the multi-nominal distribution through a softmax layer.

The difference between the generator in our RTGAN module and that in LeakGAN is that, our generator is conditioned on the additional information as discussed above. For learning, we follow the generator training method in LeakGAN [9] by adopting a hierarchical architecture to effectively generate long texts.

Briefly, the hierarchical generator G consists of a high-level MANAGER module and a low-level WORKER module. At each step, the MANAGER receives a leaked feature vector $\mathbf{f}_t$ (which is the last layer in discriminator D), and uses $\mathbf{f}_t$ to form the guiding goal vector $\mathbf{g}_t$ for the WORKER module. Compared to the scalar classification probability of D, the leaked feature vector $\mathbf{f}_t$ is a much more informative guiding signal for G, since it tells what the position of currently-generated word is in the extracted feature space.

The loss for the MANAGER module is defined as:

$$L_{G_M} = -\sum\nolimits_{t=1}^{T} Q(\mathbf{f}_t, \mathbf{g}_t) * d_{cos}(\mathbf{f}_{t+c} - \mathbf{f}_t, \mathbf{g}_t), \tag{7}$$

where $Q(\mathbf{f}_t, \mathbf{g}_t)$ is the expected reward (the classification probability output by D) under the current policy, and d_{cos} represents the cosine similarity between the change of leaked feature representation of discriminator after c-step transition (from $\mathbf{f}_t$ to $\mathbf{f}_{t+c}$) and the goal vector $\mathbf{g}_t$, and T is the maximum sequence length we set for review. The loss function aims to force the goal vector to match the transition in the feature space while achieving high reward. Meanwhile, the loss for the WORKER module is defined as:

$$L_{G_W} = -\sum\nolimits_{t=1}^{T} r_t^I \cdot p(y_t | s_{t-1}, \mathbf{c}_{ui}), \tag{8}$$

where $p(y_t | s_{t-1}, \mathbf{c}_{ui})$ denotes the conditional generative probability of the next token y_t given a sequence $s_{t-1} = [y_0, y_1, ..., y_{t-1}]$ and the condition vector $\mathbf{c}_{ui}$ in WORKER module. r_t^I is the intrinsic reward defined as:

$$r_t^I = \frac{1}{c} \sum\nolimits_{i=1}^{T} d_{cos}(\mathbf{f}_t - \mathbf{f}_{t-i}, \mathbf{g}_{t-i}) \tag{9}$$

The objective in G is to minimize L_{G_M} and L_{G_W} in two modules, which are alternatively trained while fixing the other.

Conditional Discriminator. The discriminator learns to distinguish the ground-truth review d_{ui} from the synthetic one s_{ui}. We adopt the same CNN structure in the generator to process review texts, and we can get the embedding $\mathbf{d}_{ui}$ for d_{ui} and $\mathbf{s}_{ui}$ for s_{ui}, respectively. Different from the discriminator that only distinguishes between the real and the synthetic one, our discriminator needs to determine whether the review is related to the user and the item, and whether the review is written by the user for this item. Therefore, we take the condition information $\mathbf{c}_{ui}$ into account in the discrimination as well. The loss for the discriminator D is defined as:

$$L_D = -(log(D(\mathbf{d}_{ui} | \mathbf{c}_{ui})) + log(1 - D(\mathbf{s}_{ui} | \mathbf{c}_{ui}))), \tag{10}$$

where $D()$ is the probability function computed by applying a softmax layer to the concatenation of $\mathbf{d}_{ui}/\mathbf{s}_{ui}$ and $\mathbf{c}_{ui}$. The objective in D is to maximize the probability of classifying the ground-truth review as positive, and to minimize the probability of classifying the synthetic one as authentic.

The training of G and D in RTGAN module is an adversarial process. The goal of generator is to produce the most indistinguishable synthetic reviews to fool the discriminator, while the discriminator aims to distinguish synthetic and ground-truth reviews as much as possible. Hence we iteratively train G and D to reach an equilibrium.

4.3 Neural Latent Factor Model (NLFM) Module

Inspired by the neural latent factor models in [4,10], we propose our NLFM module by extending these neural models in the following ways. Firstly, we represent *general latent factors* of user and item merely based on historical reviews without ratings. Secondly, we extend to exploit the *special latent factors* which encode the user's preference on the item in the target review.

Specifically, the embeddings of user preferences and item features, i.e., $\mathbf{g_u}$ and $\mathbf{g_i}$, are passed from the RTGAN module, and then we map them with a hidden layer to get the general latent factors of user and item. To obtain the special latent factors, we transform the target review d_{ui} (s_{ui} when testing) through a CNN structure and a hidden layer as follows:

$$\mathbf{p}_u = tanh(\mathbf{W}_{su} * CNN(d_{ui}) + b_{su}), \tag{11}$$

$$\mathbf{p}_i = tanh(\mathbf{W}_{si} * CNN(d_{ui}) + b_{si}), \tag{12}$$

where $CNN()$ is a convolutional neural network that maps the target review d_{ui} into a feature vector, and $\mathbf{W}_{su}$, $\mathbf{W}_{si}$ are the projection matrices and b_{su}, b_{si} are biases.

Combining the general and special latent factors together, we can obtain the user's and item's overall representations:

$$\mathbf{f}_u = tanh(\mathbf{W}_{gu} * \mathbf{g}_u) + tanh(\mathbf{W}_{pu} * \mathbf{p}_u), \tag{13}$$

$$\mathbf{f}_i = tanh(\mathbf{W}_{gi} * \mathbf{g}_i) + tanh(\mathbf{W}_{pi} * \mathbf{p}_i), \tag{14}$$

where $\mathbf{W}_{gu}$, $\mathbf{W}_{pu}$, $\mathbf{W}_{gi}$, $\mathbf{W}_{pi}$ are weight matrices.

We then pass these two overall representations $\mathbf{f}_u$ and $\mathbf{f}_i$ to a prediction layer to get a real-valued rating $\hat{r}_{ui}$:

$$\hat{r}_{ui} = \mathbf{f}_u^T \mathbf{f}_i + b_u + b_i + b, \tag{15}$$

where b_u, b_i, and b denotes the user bias, item bias and global bias, respectively. Clearly, our predicted rating $\hat{r}_{ui}$ encodes the general user interests and item features as well as the user's specific interest on this item.

Since rating prediction is actually a regression problem, a commonly used squared loss is adopted as the objective function for our NLFM module:

$$L_r = \sum\nolimits_{u,i \in \mathcal{U},\mathcal{I}} (\hat{r}_{ui} - r_{ui})^2, \tag{16}$$

where U, I denotes the user and item set respectively, and r_{ui} is the ground-truth rating assigned by u on i.

4.4 Training and Prediction

We iteratively train the RTGAN and NLFM modules. Since these two modules share the parameters in the historical reviews encoder layer, the parameters will be iteratively updated.

At the time of validation and testing, we first get a basic rating using the user's and item's embeddings saved in NLFM after training. We then input this basic rating as a condition to RTGAN to generate the synthetic target review. Finally, the generated review is fed into NLFM to get the final rating score. Note that though we add the RTGAN module in order to generate and utilize the synthetic review, the rating prediction task in our GTR model can be performed offline like MF methods.

5 Experiments

5.1 Experimental Setup

Datasets We conduct experiments on two publicly accessible data sources: Amazon product review[1] and Yelp 2017[2]. We use three of product categories in Amazon: *Patio, Lawn and Garden*, *Automotive*, and *Grocery and Gourmet Food*. We take the 5-core version for experiments following the previous studies [4,6,26]. In this version, each user or item has at least 5 interactions. For all datasets, we extract the textual reviews as well as the numerical ratings to conduct experiments. The basic statistics of the datasets are shown in Table 2.

Table 2. Statistics of the datasets

Datasets	Users	Items	Ratings	Sparsity
Garden	1686	962	13272	0.9918
Automotive	2928	1834	20473	0.9962
Grocery	14679	8711	151254	0.9988
Yelp2017	29406	39643	1239518	0.9990

Evaluation Metrics. For rating prediction, we employ MAE [15] and MSE [19,26,28] as evaluation metrics. For review generation, we report the results in terms of negative log-likelyhood (NLL) [9,30] and ROUGE-1 [15,28]. All these metrics are widely used in text generation and recommendation systems.

Compared Methods. We compare our GTR model with the following state-of-the-art methods.

SentiRec [12] first encodes each review into a fixed-size vector using CNN and then generates recommendations using vector-encoded reviews.

MPCN [26] exploits review-level co-attention mechanism to determine the most informative reviews and gets the representations of users and items.

A^{3}NCF [6] designs a new topic model to extract user preferences and item characteristics from review texts and then feeds them into a neural network for rating prediction.

[1] https://jmcauley.ucsd.edu/data/amazon/html.
[2] www.yelp.com/datasetchallenge/.

ALFM [7] develops an aspect-aware latent factor model where a new topic model in integrated to model user preferences and item features from different aspects.

NARRE [4] processes each review using CNN and adopts attention mechanism to build the recommendation model and select useful reviews simultaneously.

TARMF [18] adopts attention-based RNN to extract textual features and maximizes the similarity between latent factors and textual features.

MT [19] jointly learns to perform rating prediction and recommendation explanation by combining MF for rating prediction and SeqGan [30] for review generation.

NRT [15] uses MF and generation networks to combine ratings, reviews, and tips for rating prediction and abstractive tips generation.

NM [28] uses a single neural network to model users and products, and generates customized product representations using a deep memory network, from which customized ratings and reviews are constructed jointly.

CAML [5] uses an encoder-selector-decoder architecture to model the cross knowledge transferred for both the recommendation task and the explanation task using a multi-task framework.

In addition to the above baselines, we propose two variants for MT and our GTR models. Specifically, **MT-lg** replaces SeqGan [30] in the review generation module of MT [19] with LeakGan [9] in our model to exclude the potential influence caused by using different generation models. **GTR-r** removes the rating condition from the generation module in our GTR model to investigate the effects of our rating tailored GAN.

We do not compare our model with other methods like DeepCoNN [33] and TransNet [3] using reviews for rating prediction, neither with the traditional methods like NMF [14], FM [23], and NeuMF [10] which do not use reviews. These methods have been shown to be weaker than the baselines [7,18,26] used in our experiments, thus we only show improvements over the baselines.

Parameter Settings. Each dataset is divided into 80%/10%/10% splits for training, validation, and testing, respectively. We train the model on the training set and tune the hyper-parameters on the validation set. The ground-truth reviews in the training set are used for training the model. Note that those in validation or testing sets are never accessed. Instead, only the generated target reviews are used for validation or testing.

The parameters of all baselines are the same as those in the corresponding original papers. For our GTR model, we set dimensionality to 32 for all embeddings of users, items, and word latent factors. In review generation, the maximum review length T is set to 40 words, and other parameters such as the kernel size of CNN are the same as those in LeakGAN. We use Adam [13] for optimization. We set learning rate = 0.002, minibatch size = 64, and dropout ratio = 0.5 for all the datasets.

Table 3. Rating prediction performance in terms of MAE and MSE. The best results are in bold and the second best ones (except those in our GTR-r variant) are underlined. - and * denote significant difference according to paired t-test between our model and each baseline for $p < 0.05$ and $p < 0.01$, respectively.

	Garden		Automotive		Grocery		Yelp	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
SentiRec	0.833^*	1.067^*	0.637^*	0.824^*	0.742^*	1.014^*	0.926^*	1.371^*
MPCN	0.852^*	1.166^*	$\underline{0.576}^*$	0.815^*	0.821^*	1.904^*	0.902^*	1.286^*
A^{3}NCF	0.793^*	1.035^*	0.696^*	0.823^*	0.777^*	1.020^*	0.846^*	1.137^*
ALFM	0.749^*	$\underline{0.984}^*$	0.631^*	$\underline{0.772}^*$	0.746^*	1.001^*	0.828^*	1.096^*
NARRE	0.772^*	0.990^*	0.621^*	0.781^*	0.743^*	0.997^*	0.819^*	1.105^*
TARMF	0.832^*	1.103^*	0.730^*	0.868^*	0.775^*	1.073^*	0.849^*	1.196^*
MT	0.848^*	1.112^*	0.747^*	0.879^*	0.769^*	1.015^*	0.852^*	1.191^*
MT-lg	0.799^*	1.074^*	0.701^*	0.851^*	0.762^*	1.005^*	0.855^*	1.148^*
NM	0.810^-	1.181^-	0.602^-	0.829^-	0.724^*	1.020^*	0.819^*	1.116^*
NRT	0.874^*	1.109^*	0.769^*	0.814^*	0.868^*	1.174^*	0.912^*	1.127^*
CAML	$\underline{0.742}$	1.023^*	0.625^*	0.775^*	**0.704**	**0.979**	$\underline{0.815}^-$	$\underline{1.089}^*$
GTR-r	0.750	0.972	0.602	0.767	0.737	0.994	0.821	1.091
GTR	**0.743**	**0.955**	**0.566**	**0.754**	$\underline{0.706}$	$\underline{0.981}$	**0.808**	**1.073**

5.2 Rating Prediction

The results of all methods for rating prediction are presented in Table 3. (1) The upper six rows from SentiRec to TARMF are *the first type* of review-aware rating prediction methods which do not generate target reviews. (2) The middle five rows from MT to CAML are *the second type* which generates target reviews/tips. (3) The last two rows are our GTR model and its variant. From Table 3, we have the following important observations.

Firstly, our GTR model statistically significantly outperforms all baselines in terms of MAE and MSE metrics on three of the four datasets. The baselines' performances fluctuate among different datasets. MPCN, ALFM, and CAML once becomes the second best in some cases. This shows that it is hard to get the consistently better performance for one method due to the characteristics of the different datasets. In contrast, our model achieves the best performance on Garden, Automotive, and Yelp datasets. CAML is the best on Grocery. However, the difference between our model and CAML on this dataset is not significant. All these results clearly demonstrate the effectiveness of our model.

Secondly, among six methods in the first type, ALFM and NARRE are generally better than other methods. Both these methods differentiate the importance of each review or each aspect. This infers that a fine-grained analysis on the reviews has great impacts on the related rating prediction task. Among five methods in the second type, CAML benefits a lot from the joint training of two tasks under the multi-task framework. Moreover, NM performs better than

MT and NRT which only generate but do not integrate target reviews for rating prediction. Both these clearly show the predictive ability of target reviews. Our GTR model's superior performance over NM can be due to our carefully designed NLFM module, which makes the best use of the target review. The other reason is that the quality of our generated reviews is higher than that of NM with the help of rating tailored adversarial learning.

Thirdly, MT-lg is better than the original MT, suggesting the importance of generative model. On the other hand, GRT-r performs worse than GTR, showing that rating condition plays a critical role in generating reviews consistent with rating scores. However, the enhanced MT-lg is still worse than our simplified version GRT-r. This indicates that our NLFM module performs much better the matrix factorization model in MT. NRT is designed for abstractive tips generation, which results in its inferior performance.

5.3 Review Generation

This section evaluates the performance of our GTR model on review generation by comparing it with the second-type baselines. The results are presented in Table 4. NLL measures the negative log likelihood of the test data under the generated language model, and ROUGE-1 counts the number of overlapping unigrams between each pair of the generated review and the ground truth one. For NLL, the smaller the better, whereas the larger the better for ROUGE-1. The best results are in bold and the second best ones (except those in our GTR-r variant) are underlined.

Table 4. Review generation performance in terms of NLL and ROUGE-1 (R-1)

	Garden		Auto.		Grocery		Yelp	
	NLL	R-1	NLL	R-1	NLL	R-1	NLL	R-1
MT	5.74	3.22	4.01	2.95	4.28	5.20	5.19	5.22
MT-lg	5.61	3.25	3.96	2.94	4.30	5.01	5.14	5.31
NM	5.63	3.33	4.06	2.98	4.81	4.40	5.84	6.25
NRT	6.24	0.52	4.34	1.72	4.57	7.51	5.43	6.21
CAML	**5.45**	**4.96**	**3.28**	**3.33**	<u>3.51</u>	<u>7.57</u>	**4.84**	**7.38**
GTR-r	5.68	3.42	3.99	2.74	4.34	4.54	5.08	5.74
GTR	<u>5.51</u>	<u>3.49</u>	<u>3.86</u>	<u>3.01</u>	**3.25**	**7.73**	<u>4.99</u>	<u>6.43</u>

From Table 4, it is clear that our GTR model can generate the best or second best reviews in terms of NLL and ROUGE-1 metrics on all datasets. Moreover, GTR-r's results are not as good as GTR. This, once again, demonstrates that our strategy of taking rating as the condition in GAN helps generate high-quality reviews. Among the baselines, CAML can generate good reviews with the help of supervision from the rating subtask under the multi-task learning framework.

We also find that both NRT and NM perform relatively poorly. The reason might be that they only adopt the maximum likelihood estimation to generate reviews without exploiting the adversarial network. On the other hand, MT-lg is better than MT, indicating that LeakGAN performs better than SeqGAN.

5.4 Case Study

In order to capture more details, we provide several examples in Table 5 to analyze the relevance between the generated synthetic reviews/ratings and the real ones.

Table 5. Examples of the predicted ratings and the generated reviews (Ref. denotes the ground-truth review and rating)

	Rating	Review
Ref	5.0	last very long time stainless steel very good quality i not buy another sure use alot
MT	4.25	good want use like another days earth hubby metal very activity...
MT-lg	4.47	think nice product use buy but still want again skin cool cold rarely does like...
NRT	4.17	shaped nice seldom introduced so sneak transplanting still momentum ...
NM	4.33	activity absolutely very well down won cool quality skin sheath ...
CAML	4.84	bought happy grill test propane ignition roast mind what built ...
GTR-r	4.68	good product use still some lot not very operate only so middle ...
GTR	4.85	worked very well very easy use still from some quality rain not sure good value few days ...

As can be seen, our GTR model gets the highest rating score, i.e., 4.85, which is very close to the real score. Furthermore, our generated review is suitable to express the strong positive sentiment reflected by the full credit, and it is most similar to the real review. We also need to point out that, the words in the latter half of our generated review are not very accurate. This also happens to other generated reviews. The reason is that some unrelated words are padded into the short reviews when training the model to reach the fixed length. Consequently, the network is unable to generate accurate words for the latter part of the sentence.

5.5 Parameter Analysis

In this subsection, we investigate the effects of two parameters, i.e., the number of latent factors and the max length of reviews. We first examine the effect of the latent factor size in Fig. 2(a) and 2(b). We can see that, with the increase number of latent factors, the performance could be enhanced since more latent factors bring better representation capability. However, too many latent factors may cause over-fitting and result in the decrease of performance.

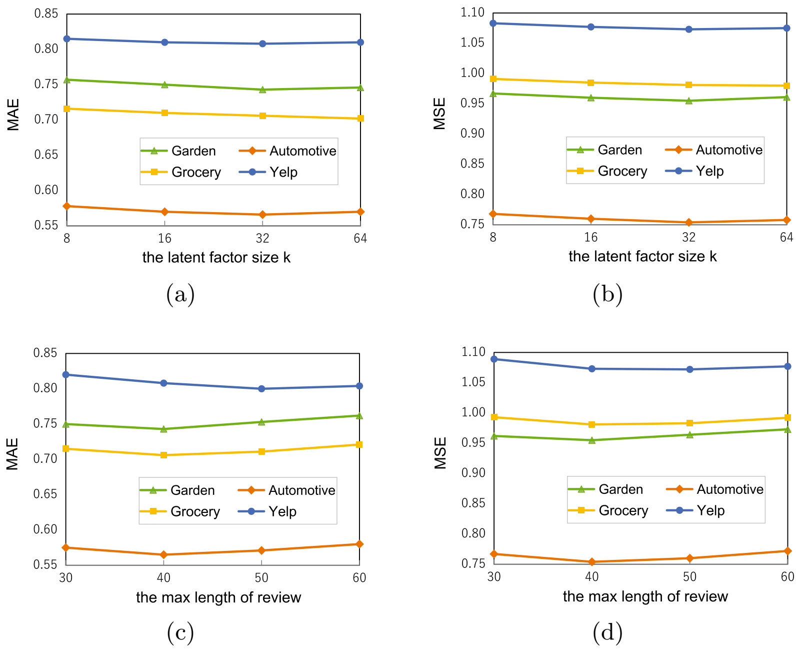

Fig. 2. Performance of different size of latent factor and max length of review.

We then study the effects of the max length of reviews in Fig. 2(c) and 2(d). When the review length is small, the part of texts that exceeds the specified length need to be truncated when preprocessing, which will result in a information loss. In this case, the smaller the specified length, the more information is missing, and thus the performance will decrease. When the review length increases, the reviews which is shorter than the threshold need to be padded. The irrelevant words padded would bring noises to the model, which will harm the performance of the model.

6 Conclusion

In this paper, we presented a novel GTR model to leverage the predictive ability of target reviews. We developed a unified framework to generate target reviews using a rating tailored GAN and to do rating prediction with a neural latent factor model which well exploits the generated target review besides historical reviews. We conducted extensive experiments on four real world datasets. Results demonstrate our model achieves the state-of-the-art performance in both rating prediction and review generation tasks.

As for future work, one possible direction is to generate target reviews with variable length. The second is to enhance the interaction between two modules under the multi-task framework. The third is to develop new approach instead of extending LeakGAN for review generation, which might be explored as a separate problem rather than a component in our rating prediction task.

Acknowledgments. The work described in this paper has been supported in part by the NSFC projects (61572376, 91646206), and the 111 project(B07037).

References

1. Bao, Y., Fang, H., Zhang, J.: Topicmf: simultaneously exploiting ratings and reviews for recommendation. In: AAAI, pp. 2–8 (2014)
2. Bauman, K., Liu, B., Tuzhilin, A.: Aspect based recommendations: recommending items with the most valuable aspects based on user reviews. In: KDD, pp. 717–725 (2017)
3. Catherine, R., Cohen, W.: Transnets: learning to transform for recommendation. In: RecSys, pp. 288–296 (2017)
4. Chen, C., Zhang, M., Liu, Y., Ma, S.: Neural attentional rating regression with review-level explanations. In: WWW, pp. 1583–1592 (2018)
5. Chen, Z., et al.: Co-attentive multi-task learning for explainable recommendation. In: IJCAI (2019)
6. Cheng, Z., Ding, Y., He, X., Zhu, L., Song, X., Kankanhalli, M.: A^3ncf: an adaptive aspect attention model for rating prediction. In: IJCAI, pp. 3748–3754 (2018)
7. Cheng, Z., Ding, Y., Zhu, L., Kankanhalli, M.: Aspect-aware latent factor model: rating prediction with ratings and reviews. In: WWW, pp. 639–648 (2018)
8. Diao, Q., Qiu, M., Wu, C.Y., Smola, A.J., Jiang, J., Wang, C.: Jointly modeling aspects, ratings and sentiments for movie recommendation (jmars). In: KDD, pp. 193–202 (2014)
9. Guo, J., Lu, S., Cai, H., Zhang, W., Yu, Y., Wang, J.: Long text generation via adversarial training with leaked information. In: AAAI (2018)
10. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: WWW, pp. 173–182 (2017)
11. Hu, L., Sun, A., Liu, Y.: Your neighbors affect your ratings: on geographical neighborhood influence to rating prediction. In: SIGIR, pp. 345–354 (2014)
12. Hyun, D., Park, C., Yang, M.C., , J.T., Yu, HSong, I., Lee.: Review sentiment-guided scalable deep recommender system. In: SIGIR, pp. 965–968 (2018)
13. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. In: Computer Science (2014)
14. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. In: NIPS, pp. 556-562 (2001)
15. Li, P., Wang, Z., Ren, Z., Bing, L., Lam, W.: Neural rating regression with abstractive tips generation for recommendation. In: SIGIR, pp. 345–354 (2017)

16. Lin, K., Li, D., He, X., Zhang, Z. and Sun, M.T.: Adversarial ranking for language generation. In: NIPS (2017)
17. Ling, G., Lyu, M.R., King, I.: Ratings meet reviews, a combined approach to recommend. In: RecSys, pp. 105–112 (2014)
18. Lu, Y., Dong, R., Smyth, B.: Coevolutionary recommendation model: mutual learning between ratings and reviews. In: WWW, pp. 773–782 (2018)
19. Lu, Y., Dong, R., Smyth, B.: Why i like it: multi-task learning for recommendation and explanation. In: RecSys, pp. 4–12 (2018)
20. McAuley, J., Leskovec, J.: Hidden factors and hidden topics: understanding rating dimensions with review text. In: RecSys, pp. 165–172 (2013)
21. Pappas, N., Popescu-Belis, A.: Sentiment analysis of user comments for one-class collaborative filtering over ted talks. In: SIGIR, pp. 773–776 (2013)
22. Š. Pero, Horáth, T.: Opinion-driven matrix factorization for rating prediction. In: UMAP, pp. 1–13 (2013)
23. Rendle, S.: Factorization machines. In: ICDM, pp. 995–1000 (2010)
24. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NIPS, pp. 3104–3112 (2014)
25. Tan, Y., Zhang, M., Liu, Y., Ma, S.: Rating-boosted latent topics: understanding users and items with ratings and reviews. In: IJCAI, pp. 2640–2646 (2016)
26. Tay, Y., Luu, A.T., Hui, S.C.: Multi-pointer co-attention networks for recommendation. In: KDD, pp. 2309–2318 (2018)
27. Wang, C., Blei, D.M.: Collaborative topic modeling for recommending scientific articles. In: KDD, pp. 448–456 (2011)
28. Wang, Z., Zhang, Y.: Opinion recommendation using a neural model. In: EMNLP, pp. 1626–1637 (2017)
29. Xu, Y., Lam, W., Lin, T.: Collaborative filtering incorporating review text and co-clusters of hidden user communities and item groups. In: CIKM, pp. 1661–1670 (2014)
30. Yu, L.T., Zhang, W.N., Wang, J., Yu, Y.: Seqgan: sequence generative adversarial nets with policy gradient. In: AAAI (2016)
31. Zhang, W., Wang, J.: Integrating topic and latent factors for scalable personalized review-based rating prediction. TKDE **28**(11), 3013–3027 (2016)
32. Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y., Ma, S.: Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In: SIGIR, pp. 83–92 (2014)
33. Zheng, L., Noroozi, V., Yu, P.S.: Joint deep modeling of users and items using reviews for recommendation. In: WSDM, pp. 425–434 (2017)