Explanation as a Defense of Recommendation

Aobo Yang[1], Nan Wang[1], Hongbo Deng[2], Hongning Wang[1]
[1]University of Virginia, Charlottesville, USA
[2]Alibaba Group, Hangzhou, China
{ay6gv,nw6a}@virginia.edu,dhb167148@alibaba-inc.com,hw5x@virginia.edu

ABSTRACT

Textual explanations have proved to help improve user satisfaction on machine-made recommendations. However, current mainstream solutions loosely connect the learning of explanation with the learning of recommendation: for example, they are often separately modeled as rating prediction and content generation tasks. In this work, we propose to strengthen their connection by enforcing the idea of sentiment alignment between a recommendation and its corresponding explanation. At training time, the two learning tasks are joined by a latent sentiment vector, which is encoded by the recommendation module and used to make word choices for explanation generation. At both training and inference time, the explanation module is required to generate explanation text that matches sentiment predicted by the recommendation module. Extensive experiments demonstrate our solution outperforms a rich set of baselines in both recommendation and explanation tasks, especially on the improved quality of its generated explanations. More importantly, our user studies confirm our generated explanations help users better recognize the differences between recommended items and understand why an item is recommended.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; **Sentiment analysis**; • **Computing methodologies** → **Natural language generation**; **Neural networks**; **Multi-task learning**.

KEYWORDS

Explainable Recommendation, Natural Language Generation, Sentiment Alignment

ACM Reference Format:
Aobo Yang[1], Nan Wang[1], Hongbo Deng[2], Hongning Wang[1]. 2021. Explanation as a Defense of Recommendation. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM '21), March 8–12, 2021, Virtual Event, Israel.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441726

1 INTRODUCTION

After extensive amount of research effort endeavored to advance the recommendation algorithms [1, 8, 16, 27, 29], solutions that explain the machine-made decisions have recently come under the spotlight [9, 40]. Numerous studies have shown that explanations help users make more accurate decisions [3], improve their acceptance of recommendations [9], and build up confidence towards the recommender system [31].

Textual explanations have been identified as a preferred medium for explaining the recommendations [40], e.g., "*This restaurant's decoration is unique and its sandwich is the best*". But due to the lack of explicit training data, most existing solutions appeal to user reviews as a proxy [4, 32, 34, 35, 37, 38, 41]: a good explanation should overlap with user-provided reviews. This is backed by extensive prior research in sentiment analysis [23] that there is a strong correlation between opinion ratings and associated review content. But the approximation also inadvertently shifts the objective of explanation learning to generating or even memorizing reviews, in a verbatim manner. It unfortunately drives the current practice in explainable recommendation to decoupling the learning of recommendation and explanation into two loosely linked sub-problems with their own objectives (e.g., rating prediction vs., content reconstruction) [20, 37, 41]. But we have to emphasize that the content generated with fairly fluent language is not sufficient to be qualified explanations, as a good explanation must elaborate why the recommendation is particular to the user. Ideally, based on the provided explanations, a user should reach the same conclusion as the system does about why an item is recommended, i.e., explanation as a defense of the recommendation.

To tie the loose ends in explainable recommendation, one needs to understand how users perceive and utilize the system-provided explanations. A recent user behavior study based on eye-tracking [5] finds that opinionated explanations at a detailed attribute-level stimulate users to compare across related recommendations, which in turn significantly increase users' product knowledge, preference certainty, and perceived recommendation transparency and quality. Motivated by this finding, we believe the sentiment delivered by the explanation text needs to reveal the details of how items are scored and ranked differently by the system. We formulate this as *sentiment alignment* between the explanation text and system's corresponding recommendation.

To demonstrate the importance of sentiment alignment, we compare example output from two explainable recommendation algorithms (one proposed in this work, and another from [20]) in Table 1. Both algorithms strongly recommended restaurant A over B, as suggested by the corresponding large margins in their recommendation scores. Note such scores are not presented to the users in practice; even presented, they do not carry any detail about why an item is preferred by the algorithm. With Algorithm 1's explanations, one can easily recognize restaurant A is recommended because of better quality in its food and service. But on the contrary, it is much harder to comprehend the recommendations based on the Algorithm 2's explanations, as the presented difference become subtle, though their readability is comparable to Algorithm 1's. Two major reasons cost misaligned explanations in the second algorithm: 1)

WSDM '21, March 8–12, 2021, Virtual Event, Israel

ACM ISBN 978-1-4503-8297-7/21/03…$15.00
https://doi.org/10.1145/3437963.3441726

Table 1: Case study on two explainable recommendation algorithms' output. Two restaurants are evaluated by the two algorithms, with corresponding recommendation scores and explanation output. We manually labeled attribute words in italic and sentiment words in bold.

Item	Algorithm 1 (Our proposed model)		Algorithm 2 (NRT [20])	
	Score	Explanation	Score	Explanation
A	4.2	the *sushi* is **good**, the *rolls* are **fresh** and the *service* is **excellent**.	4.1	their *prices* are **decent**, but the *portions* are **pretty small**.
B	2.1	it was a bit **loud** and the *service* was **slow**.	2.2	**great** *food*, **clean**, and **nice** *atmosphere*.

at training time, it only uses text reconstruction loss for explanation learning; 2) at inference time, the explanation is generated largely independently from the recommendation (as it only uses the predicted rating as an initial input for text generation). The failure to align sentiment conveyed in the explanation text with the recommendations not only cannot help users make informed decisions, but also makes them confused or even doubt about recommendations, which is totally against the purpose of explainable recommendation.

We propose to enforce sentiment alignment in *both* training and inference time for improved explainable recommendation. In particular, the learning of recommendation is modeled as a neural collaborative filtering problem [8], and the learning of explanation is modeled as a neural text generation problem [33]. We force the recommendation module to directly influence the learning of explanations by two means. First, we introduce two gated networks to our neural language model to fuse the intermediate output from the recommendation module to affect the word choice at every position of an explanation. Use examples shown in Table 1 again: given the currently generated content, the explanation module should properly choose the attribute words and corresponding sentiment modifiers (e.g., adjectives) to make their conveyed sentiment consistent with the recommendation module's prediction on this user-item pair. Second, a stand-alone sentiment regressor is added in between the two modules' output, such that its predicted sentiment score on the explanation text should be close to the given recommendation score. When discrepancy occurs, the explanation module is pushed to minimize the difference. At inference time, all our treatments for sentiment alignment are kept. But since the explanation module has been learnt, the sentiment score gap is minimized by solving a constrained decoding problem. Because the sentiment regressor can only be applied to a complete text sequence, we use the Monte Carlo Tree Search algorithm [14] for decoding with efficiency. Enforcing the alignment at inference time is vital, as it avoids the issue of decoupled output in existing explainable recommendation solutions.

We evaluate the proposed solution on both recommendation and explanation tasks, with particular focuses on the text quality, attribute personalization, and sentiment alignment of the generated explanations. The experiments are performed on Yelp and Ratebeer datasets in comparison with a rich set of popular baseline solutions. Empirical results show that our solution improves the performance on both tasks, with particularly improved explanation quality via its enhanced sentiment alignment. We also have our solution scrutinized under extensive user studies against competitive baselines. Positive user feedback suggests our explanations greatly help users gain a clearer understanding of the recommendations and make more accurate decisions.

2 RELATED WORK

User-provided reviews have been popularly used as a proxy of explanations in explainable recommendations [4, 18, 38]. One typical type of solutions directly extract representative text segments from existing reviews as explanations. For example, NARRE [4] uses attention to aggregate reviews to represent users and items for recommendation, in order to choose the most attentive reviews as explanations for each particular item. CARP [18] adopts the capsule network instead of attention for the same purpose. Wang et al. [38] extend the idea with reinforcement learning to extract the most relevant review text segments that match a given recommender system's rating prediction. However, such explanations are limited to an item's existing reviews, some of which may not even be qualified as explanations (e.g., describing a personal experience). Moreover, these models only focus on selecting reviews to identify the items' characteristics, instead of addressing the reasons for a particular recommendation provided by the system. The lack of relevance hurts users' trust on both system-provided explanations and recommendations, and thus undermines the value of explainable recommendation.

Another family of solutions learn to generate explanations from reviews. Many of them learn to predict informative elements retrieved from reviews as explanations [2, 7, 34, 37]. As a typical example, MTER [37] predicts items' attribute words and corresponding users' opinion words alone with its recommendations. Its explanations are generated by placing the predicted words into predefined templates, which however lack necessary expressiveness and diversity of nature language. Such robotic style explanations are usually considered less appreciated by users. To address this deficiency, neural language models have been applied to synthesize natural language explanations [19, 20, 22, 35]. For example, NRT [20] models explanation generation and item recommendation with a shared user-item embedding space, where its predicted recommendation rating is used as part of the initial state for corresponding explanation generation. MRG [35] integrates multiple modalities from user reviews, including ratings, text, and associated images, for explanation modeling, by treating them as parallel learning tasks. Neither the template-based or generation-based solutions paid enough attention to the sentiment alignment issue between recommendations and explanations. Although they jointly model recommendation and explanation (e.g., sharing embeddings), the objectives of training each module are still isolated. DualPC [32] realizes the importance of consistency between the two learning tasks, and introduces a duality regularization based on the joint probability of explanations and recommendations. However, the correlation imposed by duality does not have any explicit semantic meaning to the end users. In contrast, we require the output of models to be consistent in their carried sentiment, which is perceivable

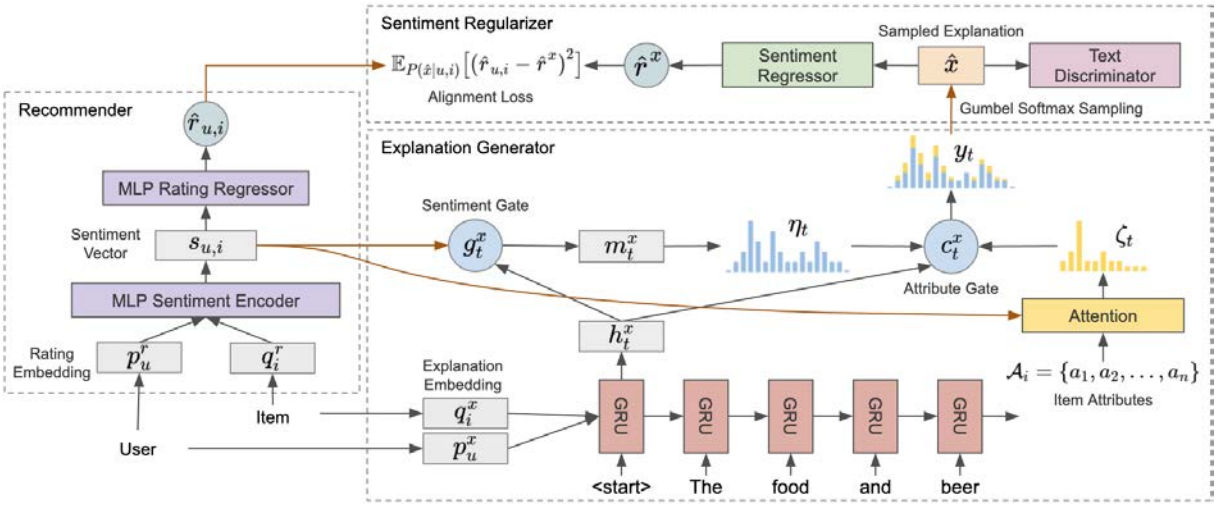

Figure 1: Model architecture of SAER. Sentiment alignment is explicitly enforced through three channels. First, SAER uses a shared sentiment vector to connect the recommender and explanation generator by the sentiment gate and attribute gate. Second, the sentiment regularizer samples generated explanations with Gumbel softmax and requires their carried sentiment (calculated by a pre-trained sentiment regressor) to match with the recommender's output score. Third, at inference time, constrained decoding is performed to ensure the alignment in the generated explanation. SAER also uses adversarial training to improve the explanations' readability in its sentiment regularizer.

by an end user. Moreover, due to the required duality, DualPC has to use an ad-hoc approximation to break the coupling between the two models' output at inference time, which unfortunately hurts the learnt consistence between the two models. Our solution treats explanation as a dependent of recommendation, and solves a constrained decoding problem to infer the most aligned explanation at testing time accordingly.

3 SENTIMENT ALIGNED EXPLAINABLE RECOMMENDATION

The problem of explainable recommendation can be formulated as follows: for a given pair of user u and item i, the model outputs a personalized recommendation based on its computed score $r_{u,i}$ and a word sequence $x_{u,i} = \{w_1, w_2, \ldots, w_n\}$ as its explanation. To learn such a model, we assume an existing training dataset, which includes a set of users $\mathcal{U}$, items $\mathcal{I}$, ratings $\mathcal{R}$, attributes $\mathcal{A}$, and explanation text $\mathcal{X}$, denoted as as $\{\mathcal{U}, \mathcal{I}, \mathcal{R}, \mathcal{A}, \mathcal{X}\}$. The attributes and explanations can be prepared from user-provided review corpora; and we will introduce the procedure we adopted for this purpose later in the experiment section. We also define a vocabulary set $\mathcal{V} = \{w_1, w_2, ..., w_{|\mathcal{V}|}\}$ for explanation generation. We define attributes as items' popular properties mentioned in the review text, and thus they are a subset of vocabulary $\mathcal{A} \subset \mathcal{V}$.

Our model architecture for addressing explainable recommendation is shown in Figure 1. It consists of three major components: 1) recommender, which takes a user and item pair (u, i) as input to predict a recommendation score $\hat{r}_{u,i}$, which measures the affinity between u and i; 2) explanation generator, which takes the (u, i) pair as input and generates a word sequence $\hat{x}_{u,i} = \{w_1, w_2, \ldots, w_n\}$ as the corresponding explanation; and 3) sentiment regularizer, which measures sentiment alignment between the generated explanation and recommendation. All three components closely interact with each other at *both* training and inference time for improved explanation generation, especially for enhanced sentiment alignment. We name our solution Sentiment Aligned Explainable Recommendation, or SAER in short. Next, we will zoom into each component to introduce its design principle and technical details.

3.1 Personalized Recommendation

As our focus in this work is not to design yet another recommendation algorithm, we adopted a latest neural collaborative filtering solution for the purpose [8]. Arguably any latent factor models that explicitly learn user and item representations [16, 37, 41] can be adopted. In this section, we will only cover the most important technical details of our recommender's design, and leave interested readers to its original paper for more details.

We stack two Multi-Layer Perceptron (MLP) networks to predict the recommendation score $\hat{r}_{u,i}$ for a given (u, i) pair. The first MLP encodes the (u, i) pair to a latent sentiment vector $\mathbf{s}_{u,i} \in \mathbb{R}^{d_s^r}$, and the second MLP maps the sentiment vector $\mathbf{s}_{u,i}$ into the numerical rating $\hat{r}_{u,i}$. We refer to the first MLP as *sentiment encoder* and the second one as *rating regressor*. Instead of using the predicted score $\hat{r}_{u,i}$ to influence explanation generation, we choose to inform the explanation generator by the encoded sentiment vector $\mathbf{s}_{u,i}$. We defer the details of this design to the next section.

In the recommendation module, we define the latent embedding matrices for users and items as $P^r \in \mathbb{R}^{d^r \times |\mathcal{U}|}$ and $Q^r \in \mathbb{R}^{d^r \times |\mathcal{I}|}$ respectively, where d^r is the dimension of the embedding vectors. The sentiment encoder concatenates the embedding vector p_u^r and q_i^r as its input and passes it through multiple layers with leaky ReLU activation to get the sentiment vector $\mathbf{s}_{u,i}$ encoded. Besides its use in the explanation generator, $\mathbf{s}_{u,i}$ is then mapped by the rating regressor through another set of multi-layer leaky ReLUs to get the final recommendation score $\hat{r}_{u,i}$.

In addition to the popularly used Minimal Squared Error (MSE) [4, 20] to train our recommender, we also introduce a pairwise hinge loss to improve the trained recommender's ranking performance. Specifically, for each user u, we collect a set of personalized item pairs $\mathcal{B}_u = \{(i,j)|r_{u,i} > r_{u,j}\}$, where i and j are two items rated by user u and one is preferred than another as observed in the training dataset. We did not use the popular BPR loss [28], because it tends to push ratings to extreme values, which is inconsistent with our sentiment regularizer's requirement to be explained later.

Based on the rating set $\mathcal{R}$ and personalized item pair set $\{\mathcal{B}_u\}_{u\in\mathcal{U}}$, the loss for recommender training is defined as:

$$L^r = \frac{1}{|\mathcal{R}|}\sum_{r_{u,i}\in\mathcal{R}}\left(\hat{r}_{u,i}-r_{u,i}\right)^2 + \sum_{u\in\mathcal{U}}\frac{\lambda_h}{|\mathcal{B}_u|}\sum_{(i,j)\in\mathcal{B}_u}\max\left(0,\beta-(\hat{r}_{u,i}-\hat{r}_{u,j})\right)$$

where $\beta > 0$ is a hyper-parameter to control the separation margin, i.e., it penalizes the model when the predicted difference between $\hat{r}_{u,i}$ and $\hat{r}_{u,j}$ is smaller than β, and λ_h is the coefficient to control the balance between MSE loss and pairwise hinge loss.

3.2 Explanation Generation

Motivated by the success of neural language generation, we appeal to a Recurrent Neural Network (RNN) model with Gated Recurrent Units (GRUs) [6] for explanation generation. To make the generation related to the user and item, we first map the input user u and item i to their embeddings p_u^x and q_i^x with the latent matrices $P^x \in \mathbb{R}^{d^x\times|\mathcal{U}|}$ and $Q^x \in \mathbb{R}^{d^x\times|I|}$ learnt by the explanation generator. We should note this set of embeddings are different from those used in the recommender (i.e., P^r and Q^r), as they should characterize different semantic aspects of users and items (ratings vs., text). We hence use superscript x to indicate variables and parameters related to explanation generator. To generate explanation text, the embeddings are concatenated and linearly converted into the initial RNN hidden state; and then the GRU generates hidden state $\mathbf{h}_t^x \in \mathbb{R}^{d_h^x}$ at position t with previous state $\mathbf{h}_{t-1}^x$ and input word w_t, and predicts the next word w_{t+1} recursively.

We initialize RNN with pretrained GloVe word embeddings $V \in \mathbb{R}^{d_v^x\times|\mathcal{V}|}$ [25].

Though similar model design has been used for explanation generation [20, 32], this straightforward application of RNN can hardly generate satisfactory explanations, where two issues are left open. First, a good explanation is expected to be personalized and specific about the recommendation; generic content, such as "*this is a nice restaurant*" can never be informative. It is important to explain the recommended item by the user's most concerned attributes [36]. Second, the sentiment carried in the explanation, especially on the mentioned attributes, should be explicit and consistent with the recommendation (as shown in our case study in Table 1). There is no guarantee that a simple RNN can satisfy both requirements.

We enhance our generator design with two gated sub-networks upon GRU to address the aforementioned issues. First, we design a sub-network, named attribute gate, to guide attribute word generation with respect to the input user-item pair and the predicted recommendation sentiment. The attribute gate is built based on a pointer network (or copy mechanism) [30, 39], which decides whether the current position should mention an attribute word and the corresponding distribution of attribute words based on the generation context. To make the choice of attribute word specific to the item, for each item i we build an attribute set with all attribute words that appear in i's associated training explanation text: $\mathcal{A}_i = \left\{a_k|a_k \in \{x_{u,i}|u \in \mathcal{U}\}\right\}$. To make the attribute choice depend on the already generated content, we attend on the concatenation of the current position's RNN hidden state $\mathbf{h}_t^x$ and sentiment vector $\mathbf{s}_{u,i}$ to compute the distribution of these attribute words,

$$\mathbf{z}_{t,k} = [\mathbf{h}_t^x, \mathbf{s}_{u,i}]^\top W_z^x \mathbf{v}_{a_k}, \forall k, a_k \in \mathcal{A}_i;\ \zeta_t = softmax(z_t), \quad (1)$$

where $W_z^x \in \mathbb{R}^{(d_h^x+d_s^r)\times d_v^x}$ and $\mathbf{v}_{a_k}$ is the word embedding of attribute a_k. $\mathbf{z}_{t,k}$ is computed for every a_k in $\mathcal{A}_i$, i.e., $\mathbf{z}_t = \{z_{t,1}, z_{t,2} ...z_{t,|\mathcal{A}_i|}\}$. ζ_t is the resulting attribute word distribution at position t. For better performance, an extra linear transformation can be applied to $\mathbf{h}_t^x$ to compress it into a lower dimension before computing attention, which helps avoid overfitting attentions to the text generation context but ignoring the sentiment context.

To decide if we need to generate an attribute word using Eq (1) at position t, we compute the copy probability with respect to the current context $\mathbf{h}_t^x$ by $c_t^x = \sigma(W_c^x\mathbf{h}_t^x + b_c^x)$, where $\sigma(\cdot)$ is the sigmoid function, $W_c^x \in \mathbb{R}^{d_h^x}$ and $b_c^x \in \mathbb{R}$. c_t^x allows us to mix the vocabulary distribution predicted by GRU and attribute word choice to get our final word distribution at position t.

Second, we design a sentiment gate to fuse the sentiment vector $\mathbf{s}_{u,i}$ to align sentiment in the generated explanation text. Our key insight is that not all words convey sentiment, we need to choose the right word at the right place to express consistent sentiment as needed by the recommender. Similar to our attribute gate design, we apply a soft gate to decide how each position is related to the intended sentiment. At position t, the sentiment gate calculates a ratio g_t^x with respect to the RNN's hidden state $\mathbf{h}_t^x$. The sentiment vector $\mathbf{s}_{u,i}$ is then weighted and merged with $\mathbf{h}_t^x$,

$$g_t^x = \sigma(W_g^x\mathbf{h}_t^x + b_g^x),\ \mathbf{m}_t^x = tanh\big(\mathbf{h}_t^x + g_t^x(W_m^x\mathbf{s}_{u,i} + b_m^x)\big) \quad (2)$$

where $W_g^x \in \mathbb{R}^{d_h^x}$ and $b_g^x \in \mathbb{R}$ produce a scalar g_t^x. $\mathbf{m}_t^x$ is the sentiment fused latent vector to predict the vocabulary distribution for position t. Because not all words are about sentiment, to better differentiate the positions where the intended sentiment needs to be expressed from the rest, we impose sparsity on the learned gate value g_t^x using L1 regularization at training time. In other words, the gate is open only when necessary.

We compute the final word distribution by consolidating the outputs of the two gated sub-networks (Eq (1) and (2)). First, the sentiment fused latent vector $\mathbf{m}_t^x$ is fed through a linear layer to calculate the vocabulary distribution $\eta_t = softmax(W_\mathbf{v}^x\mathbf{m}_t^x + b_\mathbf{v}^x)$, where $W_\mathbf{v}^x \in \mathbb{R}^{|\mathcal{V}|\times d_h^x}$ and $b_\mathbf{v}^x \in \mathbb{R}^{|\mathcal{V}|}$. Second, the vocabulary distribution η_t and attribute word distribution ζ_t are merged to obtain the final word distribution with respect to the copy probability c_t^x, i.e., $\mathbf{y}_t = (1-c_t^x)\eta_t + c_t^x\zeta_t$, where the value of w_k in ζ_t is 0 if w_k is not an attribute word.

The objective for explanation generation is to minimize the negative log-likelihood loss (NLL) on the training explanation set $\mathcal{X}$,

$$L^x = -\sum_{x\in\mathcal{X}}\sum_{w_t\in x}\log\mathbf{y}_t(w_t) + \lambda_g\sum_{x\in X}\sum_{w_t\in x}|g_t^x|$$

where $\mathbf{y}_t(w_t)$ is the resulting probability of word w_t and λ_g is the coefficient for the L1 regularization of the sentiment gate values.

3.3 Sentiment Alignment

Though our sentiment gate design (Eq (2)) introduces predicted sentiment from the recommender to the explanation generator, it is still insufficient to guarantee sentiment alignment, for three major reasons. First, word-based NLL training cannot maintain the whole sentence's sentiment. For example, as the number of sentiment words in an explanation is less than the number of non-sentiment words, the training is affected more by those non-sentiment words. This weakens its prediction quality on sentiment words. Second, the explanation generator might utilize the sentiment vector differently as the recommender does, so that the recommendation rating might diverge from the sentiment carried by the explanation. Third, the generation process at the inference stage works differently from the training stage [26]: at inference time, the previously decoded word is used as the input for the next word prediction, instead of the ground-truth word as at the training time. Hence, the learnt text pattern might not be fully exploited at the inference time.

We introduce the sentiment regularizer to close the loop between the recommender and explanation generator. It uses a stand-alone sentiment regressor to predict the sentiment rating $\hat{r}^x$ on the generated explanation text $\hat{x}_{u,i}$ for user-item pair (u, i), and requires the explanation generator to match the rating $\hat{r}_{u,i}$ from the recommender accordingly. We do not have any particular assumption about the sentiment regressor; and any state-of-the-art regression models can be leveraged [23]. In this work, we employed an MLP on top of a bidirectional RNN text encoder with inner attention for rating regression, and denote it as $f^R(x) \rightarrow r^x$. We pre-train this regressor based on ground-truth $\{\mathcal{R}, \mathcal{X}\}$ in the training set; and fix the learnt model thereafter.

To enforce sentiment alignment by the predicted ratings, we introduce a new loss to the training of our explanation generator,

$$L^a = \sum_{u \in \mathcal{U}, i \in \mathcal{I}} \mathbb{E}_{P(\hat{x}|u,i)} \left[(\hat{r}_{u,i} - f^R(\hat{x}))^2 \right] \tag{3}$$

where $P(\hat{x}|u, i)$ is the probability of generating $\hat{x}$ for the given u and i. We should note this loss is not necessarily restricted to the observed (u, i) pairs in the training set; instead, it could be any pairs of them, since both the recommender and explanation generator can generate output on any given (u, i) pair. It thus enables data augmentation for sentiment alignment.

However, because the word distribution is categorical, the generation of $\hat{x}$ is not differentiable. It makes direct optimization with respect to Eq (3) infeasible. As a result, we appeal to Gumbel softmax [11] to obtain approximated gradient of sampling from a categorical distribution. Briefly, Gumbel softmax reparameterizes the randomness in sampling by a gumbel distribution and simulates a relaxed one-hot vector with softmax. As we need a strict one-hot vector to represent each single word, we adopt the Straight-Through (ST) Gumbel softmax estimator [11]. For each (u, i) pair in Eq (3), we back-propagate the gradient from L^a to the explanation generator to improve the quality of sentiment alignment on the whole sequence.

Unfortunately, this new sentiment alignment loss might also attract the generation process to produce unreadable sequences, which however match the intended sentiment ratings. For example, the sentiment regressor may give a very positive rating to an unnatural sentence "*good good good good*", when the recommender also happens to predict a high rating for this item. Giving a higher weight to the NLL loss L^x in explanation learning cannot address this issue, as it cannot inform why a particular sequence should not be generated.

To improve the readability of our generated explanation, we introduce a text discriminator f^D, which learns to differentiate the authentic explanations from the generated ones, to guide the explanation generation as well. Our design allows any text classifier. In this work, we used an MLP binary classifier on top of a bidirectional RNN encoder for the purpose. We train the discriminator using cross-entropy loss with the ground-truth explanations x as positive and the generated explanations $\hat{x}$ as negative,

$$L^D = -\frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \log f^D(x) - \sum_{u \in \mathcal{U}, i \in \mathcal{I}} \mathbb{E}_{P(\hat{x}|u,i)} \left[\log(1 - f^D(\hat{x})) \right]$$

Correspondingly, another objective of explanation generation is to fool the discriminator, i.e., the adversarial loss,

$$L^c = - \sum_{u \in \mathcal{U}, i \in \mathcal{I}} \mathbb{E}_{P(\hat{x}|u,i)} \left[\log f^D(\hat{x}) \right]$$

This loss also requires sampled explanations $\hat{x}$ as the input, like the alignment loss defined in Eq (3). The same Gumbel softmax sampling technique is used for end-to-end training.

As we pointed out before, addressing the sentiment alignment issue in training alone is still insufficient, we introduce a constraint-driven decoding strategy to enhance the alignment at the inference stage as well. Similarly as in training, we use MSE to quantify the difference between the rating predicted from the explanation text and that from the recommender. But at the inference stage, since the explanation generator has been trained and fixed, the discrepancy can only be minimized by varying the generation of explanation text, e.g., trial and error.

Because the sentiment regressor can only be applied to a complete sequence, the search space is too large to enumerate by the generator. Hence, we treat generating explanation $\hat{x}$ at inference time as a sequence of decision making, where each action is to generate a word w_t at position t, given its already generated prefix as state. But we do not have feedback on the actions, until we complete $\hat{x}$; and the return for taking the series of actions can be measured by $Q(\hat{x}; \hat{r}_{u,i}) = [\hat{r}_{u,i} - f^R(\hat{x})]^2$. To find a policy that minimizes return (since we want to reduce the discrepancy), we need to estimate the value function under each state. This is a well studied problem in reinforcement learning, and it can be effectively addressed by Monte Carlo Tree Search (MCTS) [14]. Basically, we estimate the value function using our trained explanation generator for roll-out. When at position t for generating $\hat{x}_{u,i}$, we will sample n complete sequences for every action w using the current prefix $\{w_1, w_2, \ldots, w_{t-1}\}$, following the distribution specified by the explanation generator: $\hat{X}_{u,i,t}(w) = \left\{\hat{x}_k = MCTS_{u,i}(w_1, w_2, \ldots, w_{t-1}, w)\right\}_{k=1}^{n}$. Then the value of taking action w at position t can be estimated by,

$$Q(w_1, w_2, \ldots, w_{t-1}, w; \hat{r}_{u,i}) = \frac{1}{|\hat{X}_{u,i,t}(w)|} \sum_{\hat{x}_k \in \hat{X}_{u,i,t}(w)} Q(\hat{x}_k, \hat{r}_{u,i})$$

Based on the estimated values, we can take the action that minimizes the value. A recent study [10] suggests that top-k sampling oftentimes avoids bland and repetitive content compared to more commonly used greedy decoding strategies, such as top-1 or beam search. Therefore, we integrate our MCTS with top-k sampling,

i.e., at each decoding position t, we sample k most likely words according to word distribution $\mathbf{y_t}$ and then use MCTS to select the one that minimizes the estimated value under given state.

A vanilla implementation of MCTS is expected to be expensive and slow in our problem, as it needs to complete the sequence at each position from an RNN model for multiple times. Fortunately, our sentiment gate design provides a short path for efficient sampling: as sentiment is only carried by a small number of words, there is no need to conduct such expensive sampling procedure at every position. Instead, we only need to perform MCTS at positions where sentiment is expressed. Hence, we set a threshold on the sentiment gate's value to decide when to perform MCTS. When the gate's value is below the threshold, we will directly sample from the top-k words of the explanation generator's prediction.

3.4 End-to-End Model Training

Putting together the three components in our proposed explainable recommendation solution SAER, the overall objective of our model training is formulated as:

$$J = \min_{\Theta} \left(\lambda_r L^r + \lambda_x L^x + \lambda_a L^a + \lambda_c L^c + \lambda_n ||\Theta||^2 \right)$$

where Θ is the complete set of model parameters, and $\{\lambda_r, \lambda_x, \lambda_a, \lambda_c\}$ are the corresponding coefficients to control the relative importance of each component in model training. We also include an L2 regularization for the model parameters Θ, weighted by its coefficient λ_n. The parameters are then effectively estimated end-to-end with stochastic gradient optimizer of Adam [13].

However, due to our model's complex structure, it is challenging to fully unleash the optimizer's potential on its own. Therefore, we split the whole training process into five stages. First, estimate the sentiment regressor on $\{\mathcal{X}, \mathcal{R}\}$, as it does not depend on the other parts of our model. Second, pre-train the recommender on $\{\mathcal{U}, \mathcal{I}, \mathcal{R}\}$ till convergent. This step is essential to learn a good sentiment encoder whose output will be used to inform the explanation generator. Third, freeze the recommender and train the generator on $\{\mathcal{U}, \mathcal{I}, \mathcal{A}, \mathcal{X}\}$ with negative log-likelihood loss only. We found in our experiments that generation learning was more difficult than recommendation learning. First training the explanation generator separately can help align the training of both modules later. Fourth, after the separate training converges, start joint training of the recommender and explanation generator. This step allows the model to align the sentiment representation from both modules. At last, freeze the recommender, and turn on the sentiment regularizer to further improve the explanation generator. At this stage, the explanation discriminator and generator are trained in turn.

4 EXPERIMENTAL EVALUATION

We quantitatively evaluate our model's performance on personalized recommendation and explanation generation in two different domains: restaurant recommendation on Yelp reviews [1] and beer recommendation on Ratebeer reviews [21]. Our model is compared against a set of state-of-the-art baselines on both offline data and user studies, where encouraging improvements in both recommendation and explanation tasks are obtained.

[1] https://www.yelp.com/dataset

Table 2: Statistics of the processed datasets.

Dataset	# Users	# Items	# Reviews	# Attributes
Yelp	15,642	21,525	1,108,971	498
Ratebeer	3,895	6,993	1,073,762	333

4.1 Experiment Setup

4.1.1 Data Pre-Processing. As the attributes are not directly provided in these two review datasets, we use the Sentires toolkit [42] to extract attribute words from reviews and manually filter out inappropriate ones based on domain knowledge. Although reviews are directly treated as explanations in many previous studies [4, 38], a recent work [22] suggests a large portion of review content is only about subjective emotion and thus does not qualify as explanations, e.g., "*I love the food*". An informative explanation should depict the details of items, e.g., their attributes, to help users perceive the exact reason behind recommendations, e.g., "*the fish is fresh*". Therefore, we restrict ourselves to sentences containing attribute words as explanations in our experiments.

On top of the crafted explanations, we select 20,000 most frequent words and map others to unknown to build the vocabulary. Finally, as lots of users and items only have very few reviews in the datasets, we apply recursive filtering as in [37] to refine the datasets and alleviate this sparsity issue. The resulting statistics of the datasets are summarized in Table 2.

4.1.2 Baselines. To evaluate the personalized recommendation performance, we used the following recommendation baselines:

- **NMF**: Non-negative Matrix Factorization [17]. A widely used latent factor model, which decomposes the rating matrix into lower dimensional matrices with non-negative factors.
- **SVD**: Singular Value Decomposition [15]. It utilizes rating matrix as input for learning user and item representations.
- **NCF**: Neural Collaborative Filtering [8]. It is a modified matrix factorization solution which adopts neural networks to model the nonlinear vector operations.

We also include two explainable recommendation baselines that can output natural language sentences as explanations for comparing both the recommendation and explanation quality:

- **NARRE**: Neural Attentional Regression model with Review-level Explanations [4]. It learns the usefulness of the existing reviews through attention, and incorporates the review to enrich user and item representations for rating prediction. To fit in our evaluation, we select sentences from its most attentive reviews as explanations.
- **NRT**: Neural Rating and Tips Generation [20]. A multi-task learning solution for rating regression and content generation. It uses the predicted recommendation score to create initial states for content generation.

4.2 Quality of Personalized Recommendations

We evaluate the recommendation quality both in terms of rating prediction (by RMSE and MAE) and item ranking performance (by NDCG@{3,5,10} [12]). The results are shown in Table 3. SAER demonstrates better performance in all metrics on both datasets. In particular, thanks to the introduced hinge loss for pairwise ranking, SAER demonstrates improved ranking performance against all

Table 3: Evaluation of personalized recommendation in terms of rating prediction (RMSE, MAE) and item ranking (NDCG).

	Yelp					Ratebeer				
Model	RMSE	MAE	NDCG@3	NDCG@5	NDCG@10	RMSE	MAE	NDCG@3	NDCG@5	NDCG@10
NMF	1.1034	0.8164	0.3777	0.5067	0.7344	2.2228	1.6609	0.5143	0.6334	0.7766
SVD	1.0286	0.7975	0.3924	0.5246	0.7519	2.2942	1.6474	0.4952	0.6120	0.7593
NCF	1.0532	0.8251	0.3850	0.5150	0.7420	2.0857	1.5002	0.5421	0.6621	0.8004
NARRE	1.0275	0.8035	0.3918	0.5230	0.7509	2.0714	1.4975	0.5464	0.6641	0.8030
NRT	1.0254	0.8017	0.3947	0.5262	0.7540	2.0743	1.4922	0.5436	0.6620	0.8008
SAER	**1.0190**	**0.7948**	**0.3953**	**0.5278**	**0.7553**	**2.0628**	**1.4842**	**0.5468**	**0.6648**	**0.8034**

Table 4: BLEU scores of generated explanations.

Dataset	Model	BLEU-1	BLEU-2	BLEU-4
Yelp	NARRE	20.46	5.72	2.12
	NRT	26.25	8.84	2.97
	SAER (topk)	27.43	9.53	3.18
	SAER (reg + topk)	28.69	10.29	3.37
	SAER	**28.88**	**10.44**	**3.44**
Ratebeer	NARRE	29.78	9.47	3.27
	NRT	42.16	17.54	5.63
	SAER (topk)	43.92	19.60	6.56
	SAER (reg + topk)	45.69	21.09	7.02
	SAER	**46.01**	**21.60**	**7.32**

baselines, which only modeled recommendation as a rating prediction task. The performance difference among NCF, NRT and SAER is worth noting. Although their rating prediction modules all use MLP, NRT and SAER additionally leverage the content information for improved recommendation quality. Improvements from SAER against NARRE and NRT demonstrate that our sentiment vector and corresponding soft gate design better distill and exploit review data for joint learning. Again, as our focus in this work is not on improving recommendation quality, but more on explanation. Next, we will dive into our extensive evaluations about the generated explanation.

4.3 Quality of Generated Explanations

We evaluate the quality of our generated explanations from three perspectives: text quality, attribute personalization, and sentiment alignment. We introduce two variants of our model to better analyze the effects of our sentiment regularizer and constrained decoding strategy. 1) SAER (topk), it removes sentiment regularization and decodes by top-k sampling, such that sentiment alignment is only introduced by the soft gates, without the alignment loss, nor the constrained decoding; 2) SAER (reg + topk), it uses sentiment regularization (i.e., the alignment loss) and decodes by top-k sampling, such that sentiment alignment is only enforced at training time.

4.3.1 Quality of Generated Text. We measure the quality of generated explanation text with BLEU [24], and report the results in Table 4. The extraction-based NARRE performed clearly worse than other generation-based models. This is because the synthesized natural language explanations are not limited to the existing review content and is more flexible to customize for a particular user-item pair. NRT uses the predicted ratings in the initial state for content generation, in comparison to the sentiment vectors used in SAER. The performance gap between NRT and SAER (topk) suggests that our sentiment vectors are more expressive and the two soft gates

Table 5: Performance of attribute prediction in generated explanations.

	Yelp		Ratebeer	
Model	Precision	Recall	Precision	Recall
NARRE	0.1415	0.1906	0.2176	0.2245
NRT	0.1791	0.1997	0.3443	0.1720
SAER (topk)	0.2024	0.2297	0.3523	0.2554
SAER (reg + topk)	0.1992	0.2319	0.3549	0.2614
SAER	**0.2115**	**0.2391**	**0.3702**	**0.2677**

Table 6: Sentiment alignment evaluation of decoded explanations by RMSE. PD is the RMSE between explanation rating and predicted rating, and GT is the RMSE between explanation rating and ground-truth rating.

	Yelp		Ratebeer	
	PD	GT	PD	GT
NARRE	1.0932	1.4950	2.0996	2.9641
NRT	0.6676	1.2086	2.3302	3.1304
SAER (topk)	0.6908	1.2216	2.1727	3.0026
SAER (reg + topk)	0.6242	1.1849	1.6985	2.6769
SAER	**0.5505**	**1.1503**	**1.5911**	**2.6042**

can better guide explanation generation throughout the process, than only affecting RNN's initial state. The additional gain brought by the sentiment regularizer in SAER (reg + topk) and constrained decoding in SAER highlights the benefits of sentiment alignment in both training and inference time.

4.3.2 Attribute Personalization. An informative explanation should cover the users' most concerned aspects. We evaluate such performance in terms of attribute personalization. For each user-item pair, we evaluate precision and recall of attribute word in the algorithms' explanations against ground-truth explanations. The results in Table 5 show the improvement brought by our attribute gate, which is proved to be effective in predicting users' most concerned attributes. As the two baselines do not pay attention to items' attributes when generating the explanations, their quality in providing attribute-level explanations is much worse.

4.3.3 Sentiment Alignment Between Ratings and Explanations. Offline evaluation of sentiment alignment is not easy, since it should be evaluated by the end users who receive the recommendation and explanation. In addition to depending on user studies to evaluate this aspect (reported in the next section), we also use our pre-trained sentiment regressor for an approximated offline evaluation. For a generated explanation, we infer its carried sentiment by our sentiment regressor. We then compute the RMSE between the inferred

Table 7: Agreement rate between the model's predicted item ranking and the users perceived ranking based on the provided explanations.

	Model	gap>0.5	gap≤0.5
Agreement Rate	NRT	64.76%	52.62%
	SAER	**73.10%**	**61.90%**

rating from explanation and that predicted by the recommendation module (marked as PD). This measures sentiment difference between the recommendation and corresponding explanation. We also compare the inferred rating against the ground-truth rating (marked as GT) as a reference. The results are presented in Table 6. Without our sentiment regularizer, SAER (topk) can already significantly outperform the baselines on Yelp, which demonstrates the utility of our two gated network design for sentiment alignment. And the alignment loss and constrained decoding further push SAER's explanations closer to its recommendations. Compared to the ground-truth rating, sentiment carried by the explanation is closer to the recommender's prediction. We hypothesize that this can be caused by the difficulty to predict ground-truth rating: as reported in Table 3, the accuracy of the recommender's rating prediction is at around the same level.

5 USER STUDY

We conduct extensive user studies on Amazon Mechanical Turk to evaluate our explanations' utility to real users. We chose the restaurant recommendation task based on the Yelp dataset, as it is more familiar by general users.

We design two separate tasks. The first task focuses on evaluating if the generated explanations can help users make more informed decisions about the recommendations. In this task, we randomly pair items with different ratings predicted by a tested algorithm, and ask participants to read the corresponding explanations before choosing the item they perceived as the better one. We then evaluate the agreement rate between participants' choices and the algorithm's predictions. Specifically, without showing the actual predicted scores to participants, we present the corresponding explanations and require them to answer the following question:

> *"After reading the provided explanations, which restaurant would you like to visit? You are expected to judge the quality of the recommended restaurant based on the provided explanations, and then choose the one with better quality."*

In this experiment, we only adopted NRT as the baseline, because NARRE's explanations are item-based and thus not personalized for individual users.

To demonstrate the explanations' sentiment sensitivity towards recommendations, i.e., whether a user can correctly tell the difference between the two recommended items by reading the explanations, we group the results by the gap between the two items' predicted scores, and choose 0.5 as the threshold. We collected 420 responses for each model in each group, resulting 1,680 responses in total. The results are presented in Table 7. Both models' explanations are reasonably discriminative when the rating gap is larger. But it is more challenging to explain the difference when the recommendation scores are close. When the gap is smaller than 0.5, the agreement rate on NRT's results dropped to around 50%, which suggests users can barely perceive the differences by reading the explanations. In contrary, users can better tell the difference from SAER's explanations for making informed decisions.

Table 8: Up-vote rate of explanations' helpfulness.

	Model	Positive	Negative
Up-vote Rate	NARRE	23.33%	42.86%
	NRT	50.69%	26.98%
	GT	46.77%	**46.76%**
	SAER	**57.58%**	41.76%
Paired t-test	SAER v.s. NARRE	0	0.6786
	SAER v.s. NRT	0.0046	0
	SAER v.s. GT	0	0.9810

The second task studies whether the explanations can help users comprehend the reason of a recommended item. In particular, we ask the participants to compare explanations of the same recommended item but provided by different algorithms, and then select the most useful ones. We categorize the items as recommended (top ranked items) or not recommended (bottom ranked items) to study if the model can provide the correct explanations for both categories. For each item, we shuffle the explanations from different models for participants to select from. To help participants better judge the explanation quality, we also provide the restaurant's name and cuisine type. Specifically, we ask one of the following questions according to whether the item is recommended:

- **Positive recommendation**: *"Which of the following explanations help you the most to understand why you should pay attention to the recommended restaurant?"*
- **Negative recommendation**: *"Which of the following explanations help you the most to understand why our system believes the restaurant is NOT a good fit for you?"*

We choose NARRE, NRT and ground-truth explanations for comparison; and compare them by their received helpfulness votes.

We collected 904 responses for positive recommendations and 752 for negative. Table 8 reports the up-vote rates of the explanations from different models and the results of paired t-test. In positive recommendations, the generation-based methods, i.e., SAER and NRT, are preferred; and SAER significantly outperforms others. This reveals that the common and concise syntax and vocabulary of synthesized language are preferred in the explainable recommendation scenario, because users can more easily understand the explanations. On the negative recommendations, however, the results are mixed. SAER is still preferred over NRT, but worse than NARRE and ground-truth. The key reason is the inherent data bias: the Yelp dataset contains much more positive reviews than negative ones. Such imbalance makes SAER reluctant to generate negative explanations and less trained for negative content. Hence, its generated explanations cannot strongly justify the negative recommendations. But from a different perspective, this result also echoes the importance of aligned sentiment in explainable recommendation.

6 CONCLUSION AND FUTURE WORK

In this paper, we present a new explainable recommendation solution which synthesizes sentiment aligned neural language explanations to defend its recommendations. The alignment is obtained at the word-level by two customized soft gates, and at the sequence-level by a content-based sentiment regularizer, at both training and

inference time. Offline experiments and user studies demonstrate our model's advantages in both personalized recommendation and explanation tasks.

This work initiates the exploration of the critical role of sentiment in explainable recommendation. It leaves several valuable paths forward. Our sentiment regularizer design enables semi-supervised explainable recommendation via data augmentation. Considering the extreme sparsity of recommendation data, it can exploit the dominant amount of unobserved data for improved performance. Besides, recommendation eventually is a list-wise ranking problem; thus, it is vital to offer explanations that can reveal the relative order among the recommended items, i.e., a list-wise explanation.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful comments and suggestions. This work is partially supported by the National Science Foundation under grant SCH-1838615, IIS-1553568, and IIS-2007492, and by Alibaba Group through Alibaba Innovative Research Program.

REFERENCES

[1] Charu C Aggarwal et al. 2016. *Recommender systems*. Vol. 1. Springer.
[2] Qingyao Ai, Vahid Azizi, Xu Chen, and Yongfeng Zhang. 2018. Learning heterogeneous knowledge base embeddings for explainable recommendation. *Algorithms* 11, 9 (2018), 137.
[3] Mustafa Bilgic and Raymond J Mooney. 2005. Explaining recommendations: Satisfaction vs. promotion. In *Beyond Personalization Workshop, IUI*, Vol. 5.
[4] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural attentional rating regression with review-level explanations. In *Proceedings of the 2018 World Wide Web Conference*. 1583–1592.
[5] Li Chen, Dongning Yan, and Feng Wang. 2019. User Evaluations on Sentiment-based Recommendation Explanations. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 9, 4 (2019), 1–38.
[6] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).
[7] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. Trirank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 1661–1670.
[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. 173–182.
[9] Jonathan L Herlocker, Joseph A Konstan, and John Riedl. 2000. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*. ACM, 241–250.
[10] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751* (2019).
[11] Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* (2016).
[12] Kalervo Järvelin and Jaana Kekäläinen. 2017. IR evaluation methods for retrieving highly relevant documents. In *ACM SIGIR Forum*, Vol. 51. ACM New York, NY, USA, 243–250.
[13] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[14] Levente Kocsis and Csaba Szepesvári. 2006. Bandit based monte-carlo planning. In *European conference on machine learning*. Springer, 282–293.
[15] 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*. 426–434.
[16] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
[17] Daniel D Lee and H Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*. 556–562.
[18] Chenliang Li, Cong Quan, Li Peng, Yunwei Qi, Yuming Deng, and Libing Wu. 2019. A Capsule Network for Recommendation and Explaining What You Like and Dislike. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 275–284.
[19] Piji Li, Zihao Wang, Lidong Bing, and Wai Lam. 2019. Persona-Aware Tips Generation?. In *The World Wide Web Conference*. 1006–1016.
[20] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. 2017. Neural rating regression with abstractive tips generation for recommendation. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 345–354.
[21] Julian McAuley, Jure Leskovec, and Dan Jurafsky. 2012. Learning Attitudes and Attributes from Multi-Aspect Reviews. In *Proceedings of the 2012 IEEE 12th International Conference on Data Mining (ICDM '12)*. IEEE Computer Society, USA, 1020–1025.
[22] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 188–197.
[23] Bo Pang and Lillian Lee. 2007. Opinion Mining and Sentiment Analysis. *Found. Trends Inf. Retr.* 2, 1-2 (2007), 1–135.
[24] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 311–318.
[25] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.
[26] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732* (2015).
[27] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International Conference on Data Mining*. IEEE, 995–1000.
[28] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).
[29] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.
[30] Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1073–1083.
[31] Rashmi Sinha and Kirsten Swearingen. 2002. The role of transparency in recommender systems. In *CHI'02 extended abstracts on Human factors in computing systems*. ACM, 830–831.
[32] Peijie Sun, Le Wu, Kun Zhang, Yanjie Fu, Richang Hong, and Meng Wang. 2020. Dual Learning for Explainable Recommendation: Towards Unifying User Preference Prediction and Review Generation. In *Proceedings of The Web Conference 2020*. 837–847.
[33] Ilya Sutskever, James Martens, and Geoffrey Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*. 1017–1024.
[34] Yiyi Tao, Yiling Jia, Nan Wang, and Hongning Wang. 2019. The FacT: Taming Latent Factor Models for Explainability with Factorization Trees. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, 295–304.
[35] Quoc-Tuan Truong and Hady Lauw. 2019. Multimodal Review Generation for Recommender Systems. In *The World Wide Web Conference*. 1864–1874.
[36] Hongning Wang, Yue Lu, and Chengxiang Zhai. 2010. Latent aspect rating analysis on review text data: a rating regression approach. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. 783–792.
[37] Nan Wang, Hongning Wang, Yiling Jia, and Yue Yin. 2018. Explainable recommendation via multi-task learning in opinionated text data. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 165–174.
[38] Xiting Wang, Yiru Chen, Jie Yang, Le Wu, Zhengtao Wu, and Xing Xie. 2018. A reinforcement learning framework for explainable recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 587–596.
[39] Wenyuan Zeng, Wenjie Luo, Sanja Fidler, and Raquel Urtasun. 2016. Efficient summarization with read-again and copy mechanism. *arXiv preprint arXiv:1611.03382* (2016).
[40] Yongfeng Zhang and Xu Chen. 2020. Explainable recommendation: A survey and new perspectives. *Foundations and Trends® in Information Retrieval* 14, 1 (2020), 1–101.
[41] 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 *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. 83–92.
[42] Yongfeng Zhang, Haochen Zhang, Min Zhang, Yiqun Liu, and Shaoping Ma. 2014. Do users rate or review? Boost phrase-level sentiment labeling with review-level sentiment classification. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. 1027–1030.