Comparative Explanations of Recommendations

Aobo Yang\textsuperscript{1}, Nan Wang\textsuperscript{1}, Renqin Cai\textsuperscript{1}, Hongbo Deng\textsuperscript{2}, Hongning Wang\textsuperscript{1}
\textsuperscript{1}University of Virginia, Charlottesville, USA
\textsuperscript{2}Alibaba Group, Hangzhou, China
\{ay6gy,nw6a,rc7ne\}@virginia.edu, dhb167148@alibaba-inc.com, hw5x@virginia.edu

Abstract

As recommendation is essentially a comparative (or ranking) process, a good explanation should illustrate to users why an item is believed to be better than another, i.e., comparative explanations about the recommended items. Ideally, after reading the explanations, a user should reach the same ranking of items as the system’s. Unfortunately, little research attention has yet been paid on such comparative explanations.

In this work, we develop an extract-and-refine architecture to explain the relative comparisons among a set of ranked items from a recommender system. For each recommended item, we first extract one sentence from its associated reviews that best suits the desired comparison against a set of reference items. Then this extracted sentence is further articulated with respect to the target user through a generative model to better explain why the item is recommended. We design a new explanation quality metric based on BLEU to guide the end-to-end training of the extraction and refinement components, which avoids generation of generic content. Extensive offline evaluations on two large recommendation benchmark datasets and serious user studies against an array of state-of-the-art explainable recommendation algorithms demonstrate the necessity of comparative explanations and the effectiveness of our solution.

CCS Concepts

- Information systems → Recommender systems;  
- Computing methodologies → Natural language generation.

Keywords

explainable recommendation, comparative explanation, text generation, extract-and-refine

ACM Reference Format:
Aobo Yang\textsuperscript{1}, Nan Wang\textsuperscript{1}, Renqin Cai\textsuperscript{1}, Hongbo Deng\textsuperscript{2}, Hongning Wang\textsuperscript{1}. 2022. Comparative Explanations of Recommendations. In Proceedings of the ACM Web Conference 2022 (WWW ’22), April 25–29, 2022, Virtual Event, Lyon, France. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3485447.3512031

1 Introduction

Modern recommender systems fundamentally shape our everyday life [1, 6, 14, 19, 28, 31, 43]. As a result, how to explain the algorithm-made recommendations becomes crucial in building users’ trust in the systems [47]. Previous research shows that explanations, which illustrate how the recommendations are generated [22, 30] or why the users should pay attention to the recommendations [33, 39, 46], can notably strengthen user engagement with the system and better assist them in making informed decisions [4, 15, 32].

When being presented with a list of recommendations, typically sorted in a descending order, a user needs to make a choice. In other words, the provided explanations should help users compare the recommended items. Figure 1 illustrates the necessity of comparative explanations. By reading the explanations for the hotels recommended in the figure, one can easily tell why the system ranks them in such an order. But if the system provided the explanation in the dashed box for Hotel C, it would confuse the users about the ranking, e.g., Hotel C becomes arguably comparable to top ranked Hotel A; but it was ranked at the bottom of the list. This unfortunately hurts users’ trust in all three recommended hotels.

Existing explainable recommendation solutions are not optimized to help users make such comparative decisions for two major reasons. First, the explanation of a recommended item is often independently generated without considering other items in the recommendation list. As shown in Figure 1, one single low-quality generation (the one in the dashed box) might hamper a user’s understanding over the entire list of recommendations. Second, the popularly adopted neural text generation techniques are known to be flawed of its generic content output [16, 41]. Particularly, techniques like maximum likelihood training and sequence greedy decoding lead to short and repetitive sentences composed of globally frequent words [42]. Such generic content cannot fulfill the

![Figure 1: An illustration about the necessity of comparative explanations. The recommended Hotel A, B, C are listed in a descending order, with the provided explanations to justify the ranking. But if we replace Hotel C’s explanation with the one in the dash box, users may no longer perceive the ranking of all three hotels.](image)
We do not have any assumptions about how the recommendation system's rating prediction. Xian et al. [45] extract attributes from reviews to explain a set of items based on users’ preferences. However, as such solutions are restricted to an item’s existing reviews, their effectiveness is subject to the availability and quality of existing content. For items with limited exposure, e.g., a new item, these solutions can hardly provide any informative explanations.

Generation-based solutions synthesize textual explanations that are not limited to existing reviews. One branch focuses on predicting important aspects of an item (such as item features) from its associated reviews as explanations [2, 5, 13, 35, 39]. For instance, MTER [39] and FacT [35] predict item features that are most important for a user to justify the recommendation. They rely on predefined text templates to deliver the predicted features. The other branch applies neural text generation techniques to synthesize natural language sentences. In particular, NRT [21] models item recommendation and explanation generation in a shared user and item embedding space. It uses its predicted recommendation ratings as part of the initial state for explanation generation. MRG [36] integrates multiple modalities from user reviews, including ratings, text, and associated images, for multi-task explanation modeling.

Our work is closely related to two recent studies, DualIPC [33] and SAER [46], which focus on strengthening the relation between recommendations and explanations. Specifically, DualIPC introduces duality regularization based on the joint probability of explanations and recommendations to improve the correlation between recommendations and generated explanations. SAER introduces the idea of sentiment alignment in explanation generation. However, both of them operate in a pointwise fashion, i.e., independent explanation generation across items. Our solution focuses on explaining the comparisons between items. We should also emphasize our solution is to explain the comparison among a set of recommended items, rather than to find comparable items [9, 25].

There are also solutions exploiting other types of information for explainable recommendation, such as item-item relation [8], knowledge graph [44] and social network [17]. But they are clearly beyond the scope of this work.

### 3 Comparative Explanation Generation

Item recommendation in essence is a ranking problem: estimate a recommendation score for each item under a given user and rank the items accordingly, such that the utility of the recommendations can be maximized [18, 29]. Instead of explaining how the recommendation scores are obtained, our work emphasizes on explaining how the comparisons between the ranked items are derived.

To learn the explanation model, we assume an existing corpus of item reviews from the intended application domain (e.g., hotel reviews). Each review is uniquely associated with a user u and an item c, and a user-provided rating $r_{uc}$ suggesting his/her opinion towards the item. We group the reviews associated with user u to construct his/her profile $\Omega_u = \{(x_1^u, r_{1u}), (x_2^u, r_{2u}), \ldots, (x_m^u, r_{mu})\}$,
where $x^u_j$ is the $j$-th review opinion text extracted from user $u$’s reviews and $r^u_c$ is the corresponding opinion rating. $r^u_c$ can be easily obtained when the detailed aspect ratings are available [38]; otherwise off-the-shelf sentiment analysis methods can be used for the purpose (interested users can refer to [39, 48] for more details). As regards cold-start for users without reviews, generic profiles can be used instead which sample reviews from similar users clustered by other non-review-related features, such as rating history. We create the item profile as $\Psi_c = \{x^c_j, x^c_j, \ldots, x^c_n\}$, where $x^c_j$ is the $j$-th review sentence extracted from item $c$’s existing reviews. Unlike the user profile, the item profile does not include ratings. This is because the ratings from different users are not directly comparable, as individuals understand or use the numerical ratings differently. Our solution is agnostic to the number of entries in user profile $\Omega_u$ and item profile $\Psi_c$ in each user and item.

We impose a generative process for a tuple $(x, r^u_c)$ from user $u$ about item $c$ conditioned on $\Psi_c$ and $\Omega_u$. We assume that when user $u$ is reviewing item $c$, he/she will first select an existing sentence from $\Psi_c$ that is mostly related to the aspect he/she wants to cover about the item. Intuitively, this can be understood as the user will first browse existing reviews of the item to understand how the other users evaluated this item. Then he/she will rewrite this selected sentence to reflect his/her intended opinion and own writing style. This can be considered as a set to sequence generation problem. For our purpose of explanation generation, we only concern the generation of opinionated text $x$. Hence, we take opinion rating $r^u_c$ as input, which leads us to the following formulation,

$$P(x|u, c, r^u_c) = \sum_{x_j \in \Psi_c} P_{ref}(x|x^c_j, r^u_c, \Omega_u) P_{ext}(x^c_j|r^u_c, \Omega_u) \tag{1}$$

where $P_{ext}(x^c_j|r^u_c, \Omega_u)$ specifies the probability that $x^c_j$ from item profile $\Psi_c$ will be selected by user $u$, and $P_{ref}(x|x^c_j, r^u_c, \Omega_u)$ specifies the probability that user $u$ will rewrite $x^c_j$ into $x$. We name the resulting model Comparative Explanator, or CompExp in short.

In Eq (1), $P_{ext}(x^c_j|r^u_c, \Omega_u)$ is essential to capture the comparative textual patterns embedded in user $u$’s historical opinionated text content. To understand this, we can simply rewrite its condition part: define $\Delta r^u_c = r^u_c - r^u_j$, we have $(r^u_c, \Omega_u) = (\{x^c_j, \Delta r^u_c\})_{j=1}^n$; hence, $P_{ext}(x^c_j|r^u_c, \Omega_u)$ characterizes whether the sentence $x^c_j$ about item $c$ is qualified to characterize the desired opinion difference conditioned on user $u$’s historical content $\Omega_u$ and target rating $r^u_c$. For example, a negative $\Delta r^u_c$ suggests the opinion conveyed in $x^c_j$ is expected to be less positive than that in $x^u_j$. On a similar note, $P_{ref}(x|x^c_j, r^u_c, \Omega_u)$ quantifies if $x$ is a good rewriting of $x^c_j$ to satisfy the desired opinion rating $r^u_c$ for item $c$ by user $u$.

One can parameterize $P_{ext}(x^c_j|r^u_c, \Omega_u)$ and $P_{ref}(x|x^c_j, r^u_c, \Omega_u)$ and estimate the corresponding parameters based on the maximum likelihood principle over observations in $\Omega_u$. However, data likelihood alone is insufficient to generate high-quality explanations, as we should also emphasize on fluency, brevity, and diversity of the generated explanations. To realize this generalized objective, assume a metric $\pi(x|u, c)$ that measures the quality of generated explanation $x$ for user $u$ about item $c$, the training objective of CompExp is set to maximize the expected quality of its generated explanations under $\pi(x|u, c)$,

$$J = \mathbb{E}_x - P(x|u, c, r^u_c) \pi(x|u, c) \tag{2}$$

In this work, we present a customized BLUE score specifically for the comparative explanation generation problem to penalize short and generic content.

Next, we dive into the detailed design of CompExp in Section 3.1, then present our metric $\pi(x|u, c)$ for parameter estimation in Section 3.2 and 3.3, and finally illustrate how to estimate each component in CompExp end-to-end in Section 3.4.

### 3.1 Extract-and-Refine Architecture

Our proposed model architecture for CompExp is shown in Figure 2, which in a nutshell is a fully connected hierarchical neural network. The explanations for a user item pair $(u, c)$ is generated via an extract-and-refine process, formally described in Eq (1). Comparing to existing pure generation-based explanation methods [21, 33, 46], one added benefit of our solution is to ensure faithfulness of the generated explanations: it avoids mentioning attributes that are not relevant to the target item. To address the limitations in directly using existing content, e.g., unaligned content style or sentiment polarity, the refinement step further rewrites the extracted sentence to make its content better fit for the purpose of comparative explanation, e.g., improve the quality defined by $\pi(x|u, c)$.

We refer to $P_{ext}(x^c_j|r^u_c, \Omega_u)$ as the extractor and $P_{ref}(x|x^c_j, r^u_c, \Omega_u)$ as the refiner. Next, we will zoom into each component to discuss its design principle and technical details.

**Figure 2:** The extract-and-refine model architecture for CompExp. The extractor extracts a candidate sentence from item $c$’s profile as a prototype for explanation generation; and the refiner rewrites this sentence to optimize the desired quality metric for comparative explanation.
3.1.1 Extractor. The extractor’s goal is to select a prototype sentence \( x_j^c \) from item \( c \)’s profile \( \Psi_c \) for a given opinion rating \( r_i^u \) that best satisfies the comparativeness suggested by the user profile \( \Omega_u \). We refer to \( x_j^c \in \Psi_c \) as an extraction candidate and \( \hat{x}_j^c \in \Omega_u \) as a reference. The extractor adopts a bidirectional GRU [10] as the universal text encoder to convert the extraction candidates and references into continuous embedding vectors. Since the pairwise comparison specified by \( \Delta r_i^u \) is a scalar, we use a one-hot vector to encode it when the ratings are discrete, otherwise we use a non-linear multi-layer perceptron (MLP) as the rating encoder.

Intuitively, in the one dimensional rating space, we can easily recover the intended sentence’s rating \( r_i^u \) from the rating of the reference sentence \( r_i^u \) and required rating difference \( \Delta r_i^u \). As an analogy, we consider the rating difference vector as the transform direction that suggests the ideal comparative explanation in the latent text space from a reference sentence \( x_j^c \), denoted as \( f(x_j^c, \Delta r_i^u) \rightarrow h_i \). As a result, \( h_i \) is the text embedding vector for the ideal comparative explanation. The extractor implements such a transformation using an MLP taking the concatenation of the text embedding and rating difference embedding vectors as input.

Given the desired comparative explanation \( h_i \), the extraction candidates can be evaluated by their similarities towards \( h_i \). This specifies a directional distribution \( Q(x; h_i) \) centered on \( h_i \) in the latent text embedding space. Since cosine is a commonly used similarity metric for text embeddings, we formulate \( Q(x; h_i) \) as a von Mises-Fisher distribution [12] over all the extraction candidates, \( Q(x; h_i) \propto f_{\text{MF}}(x; h_i, \kappa) = C_P(\kappa)e^{\kappa \cos(x, h_i)} \)

where \( f_{\text{MF}}(\cdot) \) is the probability density function, \( \kappa \) is the concentration parameter, and \( C_P(\kappa) \) is a normalization function about \( \kappa \). Because each reference sentence \( x_j^c \) will suggest a different directional distribution, we extend the von Mises-Fisher distribution to cover multiple centroids and define \( P_{\text{ext}}(x_j^c \mid r_i^u, \Omega_u) \) as follows,

\[
P_{\text{ext}}(x_j^c \mid r_i^u, \Omega_u) \propto \sum_{x_j^c \in \Omega_u} f_{\text{MF}}(x_j^c; f(x_j^c, \Delta r_i^u), \kappa) \tag{3}
\]

Intuitively, in Eq (3), each ideal embedding \( h_i \) suggests which extraction candidate better fits the comparativeness embedded in \( \Omega_u \). The summation over \( \Omega_u \) aggregates each reference sentence’s evaluation on candidate sentence \( x_j^c \). \( \kappa \) is kept as a hyper-parameter which shapes the extraction probability distribution: a larger \( \kappa \) value leads to a skewer distribution. We can use it to control the exploration of the extraction candidates during the policy gradient based model training, which will be introduced in Section 3.4.

3.1.2 Refiner. The objective of the refiner is to rewrite the extracted prototype to further improve the quality metric \( \pi(x \mid u, c) \). As we argued before, a better explanation should be more supportive to the pairwise comparison required by the user profile. Therefore, assuming the refiner successfully turns the prototype \( x_j^c \) into a better framed sentence \( \hat{x}_j^c \) about the item \( c \) for user \( u \), then when we give \( \hat{x}_j^c \) back to the extractor together with \( x_j^c \), the extractor should prefer the revised version over the original one. Otherwise, we should keep refining \( \hat{x}_j^c \) until the extractor believes it can no longer be improved. Hence, the refiner needs to find a direction such that \( P_{\text{ext}}(x_j^c \mid r_i^u, \Omega_u) < P_{\text{ext}}(\hat{x}_j^c \mid r_i^u, \Omega_u) \), which is exactly suggested by the gradient of \( P_{\text{ext}}(x_j^c \mid r_i^u, \Omega_u) \) with respect to \( x_j^c \), i.e., the fastest direction for \( x_j^c \) to increase the value of \( P_{\text{ext}}(x_j^c \mid r_i^u, \Omega_u) \).

As a result, our refiner simply pushes the text embedding vector of \( x_j^c \) along this gradient direction:

\[
z_j = \nabla_{x_j^c} P_{\text{ext}}(x_j^c \mid r_i^u, \Omega_u)
\]

\[
\propto \sum_{i} e^{\kappa \cos(x_j^c, h_i)} \left[ h_i \left| \frac{x_j^c \left| \Omega_u \right|}{|h_i|} - \cos(h_i, x_j^c) \right| |x_j^c|^{2} \right]
\]

Since the refinement step should only polish the extracted prototype instead of dramatically changing it, we normalize the gradient to a unit vector and restrict the step size to one in all cases, i.e., \( x_j^c = x_j^c + z_j |z_j| \). At last, we include a single-layer GRU with attention [23] as the text decoder to convert the refined text vector \( \hat{x}_j^c \) to the final explanation sentence \( x \).

Connecting these two modules together, CompExp generates explanations for a ranked list of recommended items one at a time. To understand why the generated explanations carry comparativeness, we can consider the user’s profile \( \Omega_u \) as an anchor. Because all the explanations are generated against this anchor, the comparisons among the explanations emerge.

3.2 Explanation Quality Metric

To train CompExp under Eq (2), we need to define the explanation quality metric \( \pi(x \mid u, c) \). There is no commonly agreed offline metric for explanation quality in the community yet. And obtaining real user feedback is not feasible for offline model training. Currently, most of explainable recommendation solutions [21, 33, 46] adopt metrics measuring the overlapping content between the generated explanations and user reviews, such as BLEU [26].

However, the BLEU metric, which is initially designed for machine translation, is problematic in explanation evaluation for at least two important reasons. First, it is biased towards shorter sentences. As a precision-based metric, BLEU overcomes the short-length issue by introducing the brevity penalty, which down-scales the precision when the generated length is smaller than its “best match length” [26]. The “best match length” design is reasonable in machine translation, because all reference sentences are valid translations covering the information contained in the source language, regardless of their length differences. However, when using review sentences as proxies of explanations, the reference sentences from one review can describe totally different aspects of the same item and vary significantly in length and information contained. Since short-length generation benefits precision (less prone to erroneous word choices), BLEU favors explanations exploiting the short references as the “best match”. As a result, it pushes the models to generate explanations that are generally much shorter than the average sentence length in a review, and hence fails to explain the item in details. Second, though precision-based, BLEU is incapable to differentiate the importance of different words in a reference sentence. Words are valued equally in machine translation, but their impact in explanations varies significantly to users. For example, in Figure 1, the feature and descriptive words such as “swimming pool” and “friendly” help users better understand the target item than a very frequent but generic word, like “hotel” and “good”. BLEU’s indiscrimination to words unavoidably favors the explanations with
more generic content due to their higher chance of appearance. We later demonstrate how the BLEU metric led to both short and generic explanations in our experiments.

To design a more appropriate metric to evaluate the explanation quality and better guide our model training, we propose IDF-BLEU, i.e., Inverse Document Frequency (IDF) enhanced BLEU. It introduces three changes on top of BLEU to balance the important factors in explanations: length, content overlapping, and content rarity.

First, to penalize an overly short generation, we replace the "best match length" in the brevity factor with the average length of words with high IDF, where $d$ is the average IDF per word $d_c = \sum_{g \in x} \text{IDF}(g) / l_x$ and $d_r$ is corresponding average value in references. Then combining them forms our IDF-BLEU,

\[
\text{IDF-BLEU} = \text{BP}_{len} \cdot \text{BP}_{IDF} \cdot \exp \left( \sum_{n=1}^{N} w_n \log p_n \right)
\]

where $w_n$ is BLEU’s parameter used as the weight of the n-gram precision $\pi(x|u, c)$ for CompExp training.

3.3 Hierarchical Rewards

CompExp is a fully connected neural network which can be trained end-to-end with the gradient derived from Eq (2). However, blind end-to-end training faces the risk that the model violates the purpose of our designed extract-and-refine procedure, as the model has a great degree of freedom to arbitrarily push the prototype $x_f^c$ in the continuous vector space to optimize Eq (2). For example, it could disregard the extracted prototype and generate totally irrelevant content to the target item $c$ in the refiner.

To enforce the extract-and-refine workflow, we introduce additional intrinsic reward $[37]$ for each layer respectively to regularize their behaviours. Specifically, as IDF-BLEU is used to measure the explanation quality in Eq (2), we directly use the extracted sentence’s IDF-BLEU to reward the extractor, i.e., introduce $\pi_{\text{ext}}(x_f^c|u, c) = \text{IDF-BLEU}(x_f^c)$. For the refiner, we discourage it in pushing the final generation too far away from the extracted one. Inspired by the clipped precision in Eq (4), we propose a clipped recall to measure how many words from the selected sentence $x_f^c$ are still covered in the refined sentence,

\[
a_n = \frac{\sum_{g^n \in x_f^c} \text{IDF}(g^n) \cdot \min(\text{Count}_{\text{clip}}(g^n), \text{Count}_s(g^n))}{\sum_{g^n \in x_f^c} \text{IDF}(g^n) \cdot \text{Count}_{\text{clip}}(g^n)}
\]

where $\text{Count}_{\text{clip}}(g^n)$ is the clipped count of n-gram $g^n$ towards the references like in BLEU, and $\text{Count}_s(g^n)$ is the count of $g^n$ in the refined explanation $x$. In other words, the denominator is the prototype’s overlap with the target references and the numerator is the overlap among the prototype, references, and the final explanation. We did not use classical recall definition because it would reward the refiner to retain the entire prototype. We only encourage the refiner to keep the n-grams that are actually presented in the references. We compute the refiner’s intrinsic reward by aggregating the clipped recall over different n-grams $\pi_{\text{ref}}(x_f^c) = \exp \left( \sum_{n=1}^{N} w_n \log a_n \right)$. We did not provide this reward to the extractor, because it biases the extractor to short and generic candidates which are easier for the refiner to cover.

With the hierarchical intrinsic rewards introduced for each component, we can optimize Eq (2) by policy gradient as

\[
\nabla_{\Theta} J = \left[ \lambda_1 \pi(x|u, c) + \lambda_2 \pi_{\text{ext}}(x_f^c) \right] \nabla_{\Theta} \log P_{\text{ref}}(x_f^c | r^{\mu}, \Omega_u) + \left[ \lambda_3 \pi(x|u, c) + \lambda_4 \pi_{\text{ext}}(x_f^c) \right] \nabla_{\Theta} \log P_{\text{ext}}(x_f^c | r^{\mu}, \Omega_u)
\]

where $\lambda_1$ to $\lambda_4$ are coefficients to adjust the importance of each reward, and $\Theta$ stands for the model parameters in CompExp.

3.4 Model Training

The whole model training process can be organized into two steps: pre-training and fine-tuning. The pre-training step aims to bootstrap the extractor and refiner independently. To prepare the extractor to recognize the comparative relationships among sentences, we treat every observed review sentence as the extraction target $\pi(x_f^c | u, c)$ and train the extractor to maximize its negative log-likelihood with regard to the corresponding user and item profiles.

It is important to pre-train the refiner as a generative language model, because it would be very inefficient to learn all the natural language model parameters only through the end-to-end training. However, we do not have any paired sentences to pre-train the refiner. We borrowed the method introduced in [12, 42] to manually craft such pairs. Specifically, for every sentence, we compute its cosine similarity against all other sentences in the same item profile in the latent embedding space, and select the most similar one to pair with. Then we use this dataset to pre-train the refiner with negative log-likelihood loss.

In the fine-tuning stage, we concatenate the pre-trained layers and conduct the end-to-end training with policy gradient. To make
We demonstrate empirically that CompExp can generate improved explanations compared to state-of-the-art explainable recommendation algorithms. We conduct experiments on two different recommendation scenarios: RateBeer reviews with single-ratings [24] and TripAdvisor reviews with multi-aspect ratings [38].

### 4.1 Experiment Setup

As our solution only focuses on explanation generation, it can be applied to any recommendation algorithm of choice. In our experiments, we directly use the ground-truth review ratings as the recommendation score to factor out any deviation or noise introduced by specific recommendation algorithms. For completeness, we also empirically studied the impact from input ratings if switched to a real recommendation algorithm’s predictions.

#### 4.1.1 Data Pre-Processing

In the RateBeer dataset, we segment each review into sentences, and label them with the overall ratings from their original reviews. In the TripAdvisor dataset, there are separate ratings for five aspects including service, room, location, value and cleanliness. Therefore, each TripAdvisor review is expected to be a mix of a user’s opinions on these different aspects about the item. We segment sentences in a TripAdvisor review to different aspects using the boot-strapping method from [38] and assign resulting sentences the corresponding aspect ratings. These two datasets evaluate CompExp under different scenarios: overall opinion vs., aspect-specific opinion. We also adopt the recursive filtering [39] to alleviate the data sparsity. The statistics of the processed datasets are summarized in Table 1.

#### 4.1.2 Baselines

We compared with three explainable recommendation baselines that generate natural language explanations, covering both extraction-based and generation-based solutions.

- **NARRE**: Neural Attentional Regression model with Review-level Explanations [7]. It is an extraction-based solution. It learns the usefulness of the existing reviews through attention and selects the most attentive reviews as the explanation.
- **NRT**: Neural Rating and Tips Generation [21]. It is a generation-based solution. It models rating regression and content generation as a multi-task learning problem with shared latent space. Content is generated from its neural language model component.
- **SAER**: Sentiment Aligned Explainable Recommendation [46]. This is another generation-based solution using multi-task learning to model rating regression and explanation generation. But it focuses specifically on the sentiment alignment between the predicted rating and generated explanation.

We include three variants of CompExp to better demonstrate the effect of each component in it:

- **CompExp-Ext**: the extractor of our solution. It directly uses the selected sentences as explanations without any refinement. This variant helps us study how the extractor works and also serves as a fair counterpart for the other extraction-based baseline.
- **CompExp-Pretrain**: our model with pre-training only, which is a simple concatenation of the separately trained extractor and refiner without joint training. We compare it with CompExp to show the importance of end-to-end policy gradient training.
- **CompExp-BLEU**: our model trained with BLEU instead of IDF-BLEU. We create this variant to demonstrate the flaws of using BLEU to evaluate the quality of generated explanations.

### 4.2 Quality of Generated Explanations

To comprehensively study the quality of generated explanations, we employ different types of performance metrics, including IDF-BLEU-{1, 2, 4}, BLEU-{1, 2, 4}, average sentence length, average IDF per word, rep/l and seq_rep_2, and feature precision & recall. Both rep/l and seq_rep_2 are proposed in [41] to evaluate content repetition and higher values mean the content is more repetitive. Features are items’ representative attributes that users usually care the most [39, 45, 46], e.g., “pool” in Figure 1. The precision and recall measure if features mentioned in the generated explanations also appear in the user’s ground-truth review. We also include ground-truth review sentences as a reference baseline (labeled as “Human”) to study the differences between human and algorithm generated content. The results are reported in Table 2.

We notice that Avg Length and IDF/word in CompExp-BLEU are much smaller than Human. This suggests simply optimizing BLEU led to much shorter and less informative content. This follows our discussion before: BLEU encourage a model to generate less words and abuse common words to achieve high n-gram precision. CompExp-BLEU’s low feature precision and recall also reflect its weakness in providing informative content. Therefore, the witnessed “advantages” of CompExp-BLEU in BLEU most likely come from shorter and more generic sentences, instead of really being closer to the ground-truth content.

#### 4.2.1 IDF-BLEU over BLEU

While CompExp-BLEU topped every BLEU category on both datasets, CompExp also led almost all IDF-BLEU categories. This shows the effectiveness of our model design and the importance of directly optimizing the target evaluation metrics. To understand whether IDF-BLEU is a better metric than BLEU in evaluating the generated explanations, we should consider how the “ground-truth” content from real users look like, e.g., their average length and IDF/word, which suggest how much information is usually contained in a user-written sentence. As we can clearly notice that Avg Length and IDF/word in CompExp-BLEU are much smaller than Human. This suggests simply optimizing BLEU led to much shorter and less informative content. This follows our discussion before: BLEU encourage a model to generate less words and abuse common words to achieve high n-gram precision. CompExp-BLEU’s low feature precision and recall also reflect its weakness in providing informative content. Therefore, the witnessed “advantages” of CompExp-BLEU in BLEU most likely come from shorter and more generic sentences, instead of really being closer to the ground-truth content.

#### 4.2.2 Advantages of CompExp

There is clear performance gap between the extraction-based solutions (NARRE, CompExp-Ext) and generation-based ones (NRT, SAER, CompExp). While generation-based solutions largely outperformed extraction-based ones in content overlapping with ground-truth (IDF-BLEU, BLEU, feature precision and recall), they were generally very different from human writings in terms of sentence length, use of rare words (IDF/word),

| Dataset     | # Users | # Items | # Reviews | Rating Range |
|-------------|---------|---------|-----------|--------------|
| RateBeer    | 6,566   | 19,876  | 2,236,278 | 0 - 20       |
| TripAdvisor | 4,954   | 4,493   | 287,879   | 1 - 5        |

The policy gradient training more resilient to variance and converge faster, it is important to have a baseline to update the model with reward advantages instead of using the rewards directly. We apply Monte Carlo sampling in both extractor and refiner to have multiple explanations, and use their mean rewards as the baseline.
4.2.3 Ablation Study. Though CompExp performed well as a whole, it is inspiring to study if each component works as expected in the extract-and-refine workflow. First, both CompExp-Ext and NARRRE are extraction-based with the same candidate pool, but CompExp-Ext showed obvious advantage under most categories of IDF-BLEU and BLEU. It suggests our extractor alone can act as a competent solution where generation-based models do not fit, e.g., real-time applications requiring minimum response time. The comparison between CompExp-Ext and CompExp-Pretrain demonstrates that the refiner is able to leverage the gradient direction to improve the prototypes, even when the prototypes are given by an extractor that has not been trained jointly with the refiner. At last, there are huge gains in all metrics between CompExp-Pretrain and CompExp in both datasets. It is obvious that our reward design is beneficial to both quality and diversity of the generated explanations.

### 4.2.4 Comparativeness

To verify if the generated explanations by CompExp capture the comparative ranking of items, we study its output’s sensitivity to the input recommendation ratings. As a starting point, the ground-truth explanation perfectly aligns with the recommendation ranking, which is derived from the ground-truth rating. If the generated explanation carries the same ranking of item, the generated content should be close to the ground-truth content. As a result, if we manipulate the input recommendation scores assigned, CompExp would generate explanations more helpful than the baselines’; and 3) can users better perceive the comparative ranking from our explanations than the baselines’. BLEU than BLEU; 2) do users find our comparative explanations feature precision and recall are competitive with SAER while leading the rest, though SAER enjoys additional advantage from predefined feature pool of each item as input. As a generation-based model, CompExp largely improved the average length, word rarity, and reduced repetition over NRT and SAER. The only exception is that CompExp-BLEU was less repetitive in RateBeer, but it is mainly because its explanations were very short in general.

### 5 User Study

We have three research questions to answer in user study: 1) does users’ judgement toward explanation quality aligns more with IDF-BLEU than BLEU; 2) do users find our comparative explanations more helpful than the baselines’; and 3) can users better perceive the comparative ranking from our explanations than the baselines’. To answer these three research questions, we design two user study tasks based on RateBeer dataset using Amazon Mechanical Turk.

---

**Table 2: Explanation quality evaluated under IDF-BLEU, BLEU, average sentence length, average IDF per word, rep/l, seq_rep_2, feature precision and recall on RateBeer and TripAdvisor datasets. Bold numbers are the best of the corresponding metrics with p-value < 0.05.**

| Model          | IDF-BLEU | BLEU | Avg Length | IDF/word | rep/l | seq_rep_2 | Feature precision | recall |
|----------------|---------|------|------------|----------|-------|-----------|-------------------|--------|
| **RateBeer**   |         |      |            |          |       |           |                   |        |
| Human          | /       | /    | /          | /        | /     | /         | /                 | /      |
| NARRRE         | 17.00   | 5.18 | 1.29       | 30.22    | 9.90  | 3.58      | 11.13             | 2.45   |
| NRT            | 30.38   | 16.30| 5.80       | 48.22    | 25.28 | 10.03     | 10.43             | 2.09   |
| SAER           | 31.79   | 16.02| 5.71       | 49.08    | 26.87 | 10.59     | 10.71             | 2.19   |
| CompExp-Ext    | 24.86   | 11.72| 2.99       | 38.54    | 18.74 | 5.98      | 12.10             | 2.36   |
| CompExp-Pretrain| 27.59  | 13.44| 4.19      | 44.93    | 21.53 | 7.95      | 10.55             | 2.07   |
| CompExp-BLEU   | 23.20   | 14.55| 4.70       | 53.45    | 32.42 | 11.62     | 7.09              | 1.83   |
| CompExp        | **32.36**| **19.55**| **6.95**| 49.14    | 29.63 | 11.41     | **10.52**         | **2.16**|
| **TripAdvisor**|         |      |            |          |       |           |                   |        |
| Human          | /       | /    | /          | /        | /     | /         | /                 | /      |
| NARRRE         | 11.97   | 3.43 | 1.59       | 20.45    | 6.23  | 3.38      | 13.17             | 2.41   |
| NRT            | 16.19   | 7.50 | **2.48**   | 50.62    | 13.07 | 5.11      | 10.22             | 1.81   |
| SAER           | 16.37   | 7.65 | 2.35       | 31.20    | 13.51 | 4.94      | 10.08             | 1.71   |
| CompExp-Ext    | 13.52   | 4.25 | 1.14       | 22.12    | 7.30  | 2.66      | 14.70             | 2.39   |
| CompExp-Pretrain| 14.50  | 6.11 | 1.99      | 27.14    | 11.12 | 4.32      | 10.79             | 1.92   |
| CompExp-BLEU   | 17.04   | 7.39 | 2.04       | **32.67**| **14.66**| **5.53**   | 10.77             | 1.76   |
| CompExp        | **21.35**| **8.01**| **2.16**| 31.70    | 12.23 | 4.16      | **13.35**         | **2.12**|

and content repetition (rep/l, seq_res_2). The extraction-based solutions use content provided by human, but they are limited to the existing content. The generation-based solutions customize content for each recommendation, but suffer from common flaws of generative models, e.g., short, dull, and repetitive. Among all the models, CompExp achieved the best balance among all metrics. It significantly exceeded all baselines in terms of IDF-BLEU-{1,2} and its BLEU was only behind CompExp-BLEU. Its feature precision and recall are competitive with SAER while leading the rest, though SAER enjoys additional advantage from predefined feature pool of each item as input. As a generation-based model, CompExp largely improved the average length, word rarity, and reduced repetition over NRT and SAER. The only exception is that CompExp-BLEU was less repetitive in RateBeer, but it is mainly because its explanations were very short in general.

4.2.5 Predicted Ratings. Motivated by the findings in Figure 3a, we further study how CompExp is influenced by a real recommendation algorithm’s predicted ratings. We employed the neural collaborative filtering [14] and used its predicted ratings in CompExp’s training and testing. The result is plotted in Figure 3b. Compared with previous randomly perturbed ratings, the predicted ratings bring very limited changes to the explanations. This confirms our experiment results based on ground-truth ratings can fairly represent CompExp’s performance in real-world usage scenarios.
In this experiment, we collected 660 user responses.

The first task studies the first two research questions together. Specifically, we shuffle explanations from different models about the same recommended item and ask the participants to compare them, and then select the most helpful ones. To help participants evaluate the explanation quality, we include the original user review as the item description, towards which they can judge if the explanation are accurate or informative. For each recommended item, we ask participants to answer the following question after reading its description and candidate explanations:

"Which of the following explanations best describe the characteristics of the given beer and help you the most to understand why you should pay attention to the recommendation?"

In this experiment, we collected 660 user responses.

The results are presented in Table 3 and 4. In Table 3, we used Cohen’s kappa coefficient to compare IDF-BLEU and BLEU’s agreement with users’ responses. For each test case, we pair explanations that the participants chose as helpful with the rest to form a set of explanation pairs. Then we use IDF-BLEU-[1,2,4] and BLEU-[1,2,4] to identify the helpful one in each pair. The kappa coefficient shows that IDF-BLEU aligns significantly better with users’ judgment in all three subcategories under paired t-test. Table 4 shows the helpfulness vote on each model and the paired t-test results of CompExp against other baselines. The helpfulness vote on CompExp is significantly higher than others, which suggests strong user preference over its generated explanations.

The second task addresses the last research question, i.e., if a user is able to perceive the ranking of recommended items from the explanations. In this task, we randomly paired items of different ratings and asked participants to identify which item is better by reading the provided explanations. We then evaluated the agreement rate between participants’ choices and the actual ranking. In particular, given the explanations of a model, the participants were required to answer the following question:

"After reading the explanations for recommended items, which item would you like to choose? You are expected to judge the quality of the items based on the provided explanations."

We chose SAER and NRT as baselines. Besides, we also include the ground-truth sentences from the actual user reviews as a reference. We collected 200 responses for each model.

Table 5 reports the agreement rates between the actual ranking and the ranking perceived by the participants. CompExp’s agreement rate is slightly higher than NRT and SAER, but it is far below the Ground-Truth. The Ground-Truth’s high agreement rate quantitatively confirms that the original user provided review sentences are highly comparative. This observation supports our choice of training the comparative explanation generation from paired user review sentences. And it also suggests there is still a performance gap in comparativeness for learning-based solutions to bridge. An improved objective for optimization, e.g., include quantified pairwise comparativeness, might be a promising direction.

6 Conclusion and Future Work

In this paper, we studied the problem of comparative explanation generation in explainable recommendation. The objective of our generated explanations is to help users understand the comparative item rankings provided in a recommender system. We develop a neural extract-and-refine architecture to generate such comparative explanations, with customized metrics to penalize generic and useless content in the generated explanations. Both offline evaluations and user studies demonstrated the effectiveness of our solution.

This work starts a bright new direction in explainable recommendation. Our current solution only focuses on explanation generation, by assuming a perfect recommendation algorithm (i.e., we directly used the ground-truth opinion ratings in our experiments). If important to improve our model by co-design with a real recommendation algorithm, whose recommendation score is expected to be noise and erroneous. In addition, we still heavily depend on existing review content to guide explanation generation. It will be more meaningful to introduce actual user feedback in this process, i.e., interactive optimization of explanation generation.

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.
The beer is a hazy yellow-orange color. It has a roasted malt and light smoke flavor, with a sweet aroma containing toasted malt, caramel, and alcohol notes. It pours a hazy golden color with a small white head.

The finish is dry and ashy. It is not bad, if one is looking for a refreshing, light wheat beer. It has a hazy golden color with a small white head.

The beer is a hazy yellow color. It has a roasted malt and light smoke flavor, with a sweet aroma containing toasted malt, caramel, and alcohol notes. It has a hazy golden body with a small white head.

The beer is a hazy yellow-orange color. It has a roasted malt and light smoke flavor, with a sweet aroma containing toasted malt, caramel, and alcohol notes. It has a hazy golden body with a small white head.

The beer is a hazy yellow-orange color. It has a roasted malt and light smoke flavor, with a sweet aroma containing toasted malt, caramel, and alcohol notes. It has a hazy golden body with a small white head.
users provided historical content. Sentences extracted by CompExp-Ext are much closer to the ground-truth than NARRE's. Comparing CompExp-Ext and CompExp shows the effectiveness our rewriting module in improving the explanation quality, especially in writing style and wording. For example, in Sample 1, the extracted explanation correctly covers the attribute “aroma” and “caramel”, but its sentence structure is different from the ground-truth’s. The refined explanation keeps the two correct attributes and improves the sentence structure. In Sample 2, while the extractor picks a sentence almost the same as the ground-truth, the refiner further changes the word “golden” to “yellow”, which better reflects the user’s preference in wording. However, both samples also suggest our refiner can be further improved in personalized feature revision. For example, in Sample 2, the end explanation inherits “hazy” from the extracted prototype while the item looks “clear” instead for the target user. Same for “malt” and “alcohol” vs., “cherry” and “raisins” in Sample 1. Obviously, these features are subjective and the target user hold a different opinion from the author of the extracted sentence. It would be a promising future direction to better personalize subjective features while still maintain the relevance and faithfulness to other objective facts given by the extraction.