Interacting with Explanations through Critiquing

Diego Antognini[1] , **Claudiu Musat**[2] and **Boi Faltings**[1]
[1]École Polytechnique Fédérale de Lausanne, Switzerland
[2]Swisscom, Switzerland
firstname.lastname@{epfl.ch, swisscom.com}

Abstract

Using personalized explanations to support recommendations has been shown to increase trust and perceived quality. However, to actually obtain better recommendations, there needs to be a means for users to modify the recommendation criteria by interacting with the explanation. We present a novel explanation technique using aspect markers that learns to generate personalized explanations of recommendations from review texts, and we show that human users significantly prefer these explanations over those produced by state-of-the-art techniques.

Our work's most important innovation is that it allows users to react to a recommendation by critiquing the textual explanation: removing (symmetrically adding) certain aspects they dislike or that are no longer relevant (symmetrically that are of interest). The system updates its user model and the resulting recommendations according to the critique. This is based on a novel unsupervised critiquing method for single- and multi-step critiquing with textual explanations. Empirical results show that our system achieves good performance in adapting to the preferences expressed in multi-step critiquing and generates consistent explanations.

1 Introduction

Explanations of recommendations aremake beneficial. Modern recommender systems accurately capture users' preferences and achieve high performance. But, their performance comes at the cost of increased complexity, which makes them seem like black boxes to users. This may result in distrust or rejection of the recommendations [Tintarev and Masthoff, 2015].

There is thus value in providing *textual explanations* of the recommendations, especially on e-commerce websites, because such explanations enable users to understand why a particular item has been suggested and hence to make better decisions [Kunkel *et al.*, 2018]. Furthermore, explanations increase overall system transparency [Tintarev and Masthoff, 2015] and trustworthiness [Zhang and Curley, 2018].

However, not all explanations are equivalent. [Kunkel *et al.*, 2019] showed that highly personalized justifications using

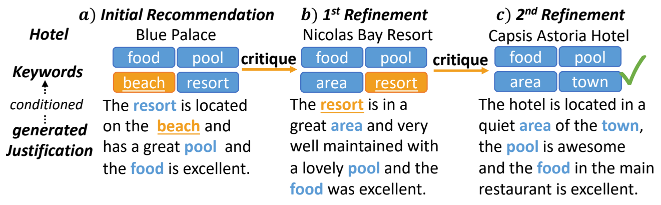

a) Initial Recommendation
b) 1st Refinement
c) 2nd Refinement
Hotel
Blue Palace
Nicolas Bay Resort
Capsis Astoria Hotel
Keywords
food
pool
beach
resort
critique
food
pool
area
resort
critique
food
pool
area
town
conditioned
generated
Justification
The resort is located on the beach and has a great pool and the food is excellent.
The resort is in a great area and very well maintained with a lovely pool and the food was excellent.
The hotel is located in a quiet area of the town, the pool is awesome and the food in the main restaurant is excellent.

Figure 1: A flow of conversational critiquing over two time steps. a) The system proposes to the user a recommendation with a keyphrase explanation and a justification. The user can interact with the explanation and critique phrases. b) A new recommendation is produced from the user's profile and the critique. 3) This process repeats until the user accepts the recommendation and ceases to provide critiques.

natural language lead to substantial increases in perceived recommendation quality and trustworthiness compared to simpler explanations, such as aspect, template, or similarity.

A second, and more important, benefit of explanations is that they provide a basis for feedback: if a user is unsatisfied with a recommendation, understanding what generated it allows them to *critique* it (Fig. 1). Critiquing – a conversational method of incorporating user preference feedback regarding item attributes into the recommended list of items – has several advantages. First, it allows the system to correct and improve an incomplete or inaccurate model of the user's preferences, which improves the user's decision accuracy [Chen and Pu, 2012]. Compared to preference elicitation, critiquing is more flexible: users can express preferences in any order and on any criteria [Reilly *et al.*, 2005].

Useful explanations are hard to generate. Prior research has employed users' reviews to capture their preferences and writing styles (e.g., [Dong *et al.*, 2017]). From past reviews, they generate *synthetic* ones that serve as personalized *explanations* of ratings given by users. However, many reviews are noisy, because they partly describe experiences or endorsements. It is thus nontrivial to identify meaningful justifications inside reviews. [Ni *et al.*, 2019] proposed a pipeline for identifying justifications from reviews and asked humans to annotate them. [Chen *et al.*, 2019; Chen *et al.*, 2020] set the justification as the first sentence. However, these notions of justification were ambiguous, and they assumed that a review contains only one justification.

Recently, [Antognini *et al.*, 2021] solved these shortcomings by introducing a justification extraction system with no prior limits imposed on their number or structure. This is important because a user typically justifies his overall rating with multiple explanations: one for each aspect the user cares about [Musat and Faltings, 2015]. The authors showed that there is a connection between faceted ratings and snippets within the reviews: for each subrating, there exists at least one text fragment that alone suffices to make the prediction. They employed a sophisticated attention mechanism to favor long, meaningful word sequences; we call these ***markers***. Building upon their study, we show that these *markers* serve to create better user and item profiles and can inform better user-item pair justifications. Fig. 2 illustrates the pipeline.

From explanations to critiquing. To reflect the overlap between the profiles of a user and an item, one can produce a set of keyphrases and then a synthetic justification. The user can correct his profile, captured by the system, by *critiquing* certain aspects he does not like or that are missing or not relevant anymore and obtain a new justification (Fig. 1). [Wu *et al.*, 2019] introduced a keyphrase-based critiquing method in which attributes are mined from reviews, and users interact with them. However, their models need an extra autoencoder to project the critique back into the latent space, and it is unclear how the models behave in multi-step critiquing.

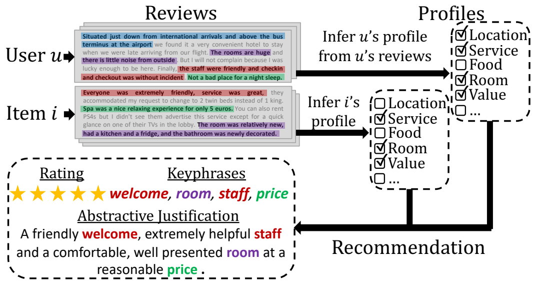
We overcome these drawbacks by casting the critiquing as an unsupervised attribute transfer task: altering a keyphrase explanation of a user-item pair representation to the critique. To this end, we entangle the user-item pair with the explanation in the same latent space. At inference, the keyphrase classifier modulates the latent representation until the classifier identifies it as the critique vector.

In this work, we address the problem recommendation with fine-grained explanations. We first demonstrate how to extract multiple relevant and personalized justifications from the user's reviews to build a profile that reflects his preferences and writing style (Fig. 2). Second, we propose T-RECS, a recommender with explanations. T-RECS explains a rating by first inferring a set of keyphrases describing the intersection between the profiles of a user and an item. Conditioned on the keyphrases, the model generates a synthetic personalized justification. We then leverage these explanations in an unsupervised critiquing method for single- and multi-step critiquing. We evaluate our model using two real-world recommendation datasets. T-RECS outperforms strong baselines in explanation generation, effectively re-ranks recommended items in single-step critiquing. Finally, T-RECS also better models the user's preferences in multi-step critiquing while generating consistent textual justifications.

2 Related Work

Textual Explainable Recommendation. Researchers have investigated many approaches to generating textual explanations of recommended items for users. [McAuley and Leskovec, 2013] proposed a topic model to discover latent factors from reviews and explain recommended items. [Zhang *et al.*, 2014] improved the understandability of topic words and aspects by filling template sentences.

Another line of research has generated synthetic reviews as explanations. Prior studies have employed users' reviews and



Situated just down from international arrivals and above the bus terminus at the airport we found it a very convenient hotel to stay when we were late arriving from our flight. The rooms are huge and there is little noise from outside. But I will not complain because I was lucky enough to be here. Finally, the staff were friendly and checkin and checkout was without incident. Not a bad place for a night sleep.

Everyone was extremely friendly, service was great, they accommodated my request to change to 2 twin beds instead of 1 king. Spa was a nice relaxing experience for only 5 euros. You can also rent PS4s but I didn't see them advertise this service except for a quick glance on one of their TVs in the lobby. The room was relatively new, had a kitchen and a fridge, and the bathroom was newly decorated.

A friendly welcome, extremely helpful staff and a comfortable, well presented room at a reasonable price.

Figure 2: For reviews written by a user u and a set of reviews about an item i, we extract the justifications for each aspect rating and implicitly build an interest profile. T-RECS outputs a personalized recommendation with two explanations: the keyphrases reflecting the overlap between the two profiles, and a synthetic justification conditioned on the latter.

tips to capture their preferences and writing styles. [Catherine and Cohen, 2017] predicted and explained ratings by encoding the user's review and identifying similar reviews. [Chen *et al.*, 2019] extended the previous work to generate short synthetic reviews. [Sun *et al.*, 2020] optimized both tasks in dual forms. [Dong *et al.*, 2017] proposed an attribute-to-sequence model to learn how to generate reviews given categorical attributes. [Ni and McAuley, 2018] improved review generation by leveraging aspect information using a seq-to-seq model with attention. Instead of reviews, others have generated tips [Li *et al.*, 2017; Li *et al.*, 2019]. However, the tips are scarce and uninformative [Chen *et al.*, 2019]; many reviews are noisy because they describe partially general experiences or endorsements [Ni *et al.*, 2019].

[Ni *et al.*, 2019] built a seq-to-seq model conditioned on the aspects to generate relevant explanations for an existing recommender system; the fine-grained aspects are provided by the user in the inference. They identified justifications from reviews by segmenting them into elementary discourse units (EDU) [Mann and Thompson, 1988] and asking annotators to label them as "good" or "bad" justifications. [Chen *et al.*, 2019] set the justification as the first sentence. All assumed that a review contains only one justification. Whereas their notions of justification were ambiguous, we extract multiple justifications from reviews using *markers* that justify subratings. Unlike their models, ours predicts keyphrases on which the justifications are conditioned and integrates critiquing.

Critiquing. Refining recommended items allows users to interact with the system until they are satisfied. Some methods are example critiquing [Williams and Tou, 1982], in which users critique a set of items; unit critiquing [Burke *et al.*, 1996], in which users critique an item's attribute and request another one instead; and compound critiquing [Reilly *et al.*, 2005] for more aspects. The major drawback of these approaches is the assumption of a fixed set of known attributes.

[Wu *et al.*, 2019] circumvented this limitation by extending the neural collaborative filtering model [He *et al.*, 2017]. First, the model explains a recommendation by predicting a set of keywords (mined from users' reviews). In [Chen *et al.*, 2020], based on [Chen *et al.*, 2019], the model samples only one keyword via the Gumbel-Softmax function. Our work applies a deterministic strategy similar to [Wu *et al.*, 2019].

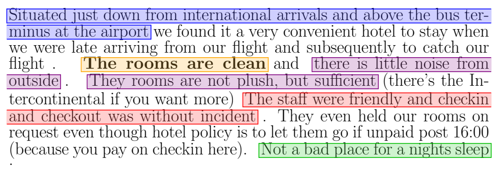

Situated just down from international arrivals and above the bus terminus at the airport we found it a very convenient hotel to stay when we were late arriving from our flight and subsequently to catch our flight . **The rooms are clean** and there is little noise from outside . They rooms are not plush, but sufficient (there's the Intercontinental if you want more) The staff were friendly and checkin and checkout was without incident . They even held our rooms on request even though hotel policy is to let them go if unpaid post 16:00 (because you pay on checkin here). Not a bad place for a nights sleep .

Figure 3: Extracted justifications from a hotel review. The inferred *markers* depict the excerpts that explain the ratings of the aspects: Service, Cleanliness, Value, Room, and Location. We denote in **bold** the EDU-based justification from the model of [Ni *et al.*, 2019].

Second, [Wu *et al.*, 2019] project the critiqued keyphrase explanations back into the latent space, via an autoencoder that perturbs the training, from which the rating and the explanation are predicted. In this manner, the user's critique modulates his latent representation. The model of [Chen *et al.*, 2020] is trained in a two-stage manner: one to perform recommendation and predict one keyword and another to learn critiquing from online feedback, which requires additional data. By contrast, our model is simpler and learns critiquing in an unsupervised fashion: it iteratively edits the latent representation until the new explanation matches the critique. Finally, [Luo *et al.*, 2020] examined various linear aggregation methods on latent representations for multi-step critiquing. In comparison, our gradient-based critiquing iteratively updates the latent representation for each critique.

3 Extracting Justifications from Reviews

In this section, we introduce the pipeline for extracting high-quality and personalized justifications from users' reviews. We claim that a user justifies his overall experience with multiple explanations: one for each aspect he cares about. Indeed, it has been shown that users write opinions about the topics they care about [Zhang *et al.*, 2014]. Thus, the pipeline must satisfy two requirements: 1. extract text snippets that reflect a rating or subrating, and 2. be data driven and scalable to mine massive review corpora and to construct a large personalized recommendation justification dataset.

[Antognini *et al.*, 2021] proposed the multi-target masker (MTM) to find text fragments that explain faceted ratings in an unsupervised manner. MTM fulfills the two requirements. For each word, the model computes a distribution over the aspect set, which corresponds to the aspect ratings (e.g., service, location) and "not aspect." In parallel, the model minimizes the number of selected words and discourages aspect transition between consecutive words. These two constraints guide the model to produce long, meaningful sequences of words called ***markers***. The model updates its parameters by using the inferred *markers* to predict the aspect sentiments jointly and improves the quality of the *markers* until convergence.

Given a review, MTM extracts the *markers* of each aspect. A sample is shown in Fig. 3. Similarly to [Ni *et al.*, 2019], we filter out *markers* that are unlikely to be suitable justifications: including third-person pronouns or being too short. We use the constituency parse tree to ensure that *markers* are verb phrases. The detailed processing is available in Appendix A.

4 T-RECS: A Multi-Task Transformer with Explanations and Critiquing

Fig. 4 depicts the pipeline and our proposed T-RECS model. Let U and I be the user and item sets. For each user $u \in U$ (respectively an item $i \in I$), we extract *markers* from the user's reviews on the training set, randomly select N_{just}, and build a justification reference J^u (symmetrically J^i).

Given a user u, an item i, and their justification history J^u and J^i, our goal is to predict 1. a rating $\boldsymbol{y}_r$, 2. a keyphrase explanation $\boldsymbol{y}_{kp}$ describing the relationship between u and i, and 3. a natural language justification $\boldsymbol{y}_{just} = \{w_1, ..., w_N\}$, where N is the length of the justification. $\boldsymbol{y}_{just}$ explains the rating $\boldsymbol{y}_r$ conditioned on $\boldsymbol{y}_{kp}$.

4.1 Model Overview

For each user and item, we extract *markers* from their past reviews (in the train set) and build their justification history J^u and J^i, respectively (see Section 3). T-RECS is divided into four submodels: an **Encoder** E, which produces the latent representation $\boldsymbol{z}$ from the historical justifications and latent factors of the user u and the item i; a **Rating Classifier** C^r, which classifies the rating $\hat{\boldsymbol{y}}_{\boldsymbol{r}}$; a **Keyphrase Explainer** C^{kp}, which predicts the keyphrase explanation $\hat{\boldsymbol{y}}_{kp}$ of the latent representation $\boldsymbol{z}$; and a **Decoder** D, which decodes the justification $\hat{\boldsymbol{y}}_{just}$ from $\boldsymbol{z}$ conditioned on $\hat{\boldsymbol{y}}_{kp}$, encoded via the **Aspect Encoder** A. T-RECS involves four functions: $\boldsymbol{z} = E(u, i)$;$\hat{\boldsymbol{y}}_r = C^r(\boldsymbol{z})$;$\hat{\boldsymbol{y}}_{kp} = C^{kp}(\boldsymbol{z})$; $\hat{\boldsymbol{y}}_{just} = D(\boldsymbol{z}, A(\hat{\boldsymbol{y}}_{kp}))$.

The above formulation contains two types of personalized explanations: a list of keyphrases $\hat{\boldsymbol{y}}_{kp}$ that reflects the different aspects of item i that the user u cares about (i.e., the overlap between their profiles) and a natural language explanation $\hat{\boldsymbol{y}}_{just}$ that justifies the rating, conditioned on $\hat{\boldsymbol{y}}_{kp}$. The set of keyphrases is mined from the reviews and reflects the different aspects deemed important by the users. The keyphrases enable an interaction mechanism: users can express agreement or disagreement with respect to one or multiple aspects and hence critique the recommendation.

Entangling user-item.
A key objective of T-RECS is to build a powerful latent representation. It accurately captures user and item profiles with their writing styles and entangles the rating, keyphrases, and a natural language justification. Inspired by the superiority of the Transformer for text generation tasks [Radford *et al.*, 2019], we propose a Transformer-based encoder that learns latent personalized features from users' and items' justifications. We first pass each justification J^u_j (respectively J^i_j) through the Transformer to compute the intermediate representations $\boldsymbol{h}^u_j$ (respectively $\boldsymbol{h}^i_j$). We apply a sigmoid function on the representations and average them to get $\boldsymbol{\gamma}^u$ and $\boldsymbol{\gamma}^i$:

$$\boldsymbol{\gamma}^u = \frac{1}{|J^u|} \sum\nolimits_{j \in J^u} \sigma(\boldsymbol{h}^u_j) \quad \boldsymbol{\gamma}^i = \frac{1}{|J^i|} \sum\nolimits_{j \in J^i} \sigma(\boldsymbol{h}^i_j).$$

In parallel, the encoder maps the user u (item i) to the latent factors $\boldsymbol{\beta}^u$ ($\boldsymbol{\beta}^i$) via an embedding layer. We compute the latent representation $\boldsymbol{z}$ by concatenating the latent personalized features and factors and applying a linear projection: $\boldsymbol{z} = E(u, i) = W[\boldsymbol{\gamma}^u \parallel \boldsymbol{\gamma}^i \parallel \boldsymbol{\beta}^u \parallel \boldsymbol{\beta}^i] + \boldsymbol{b}$, where $\parallel$ is the concatenation operator, and W and $\boldsymbol{b}$ the projection parameters.

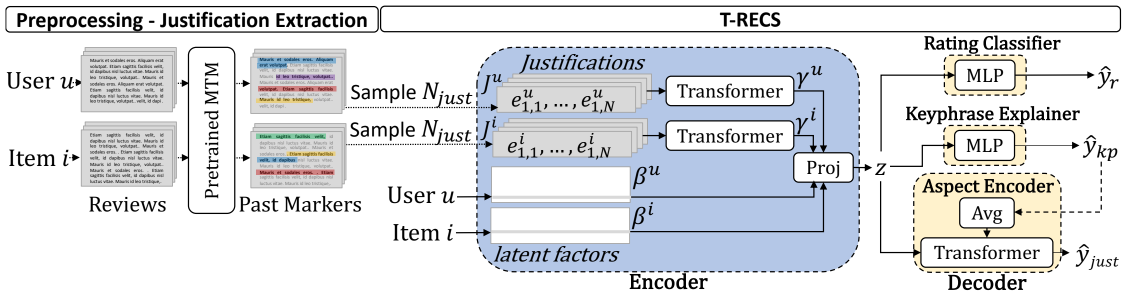


Figure 4: (Left) Preprocessing for the users and the items. For each user u and item i, we first extract *markers* from their past reviews (highlighted in color), using the pretrained multi-target masker (see Section 3), that become their respective justifications. Then, we sample N_{just} of them and build the justification references J^u and J^i, respectively. (Right) T-RECS architecture. Given a user u and an item i with their justification references J^u, J^i and latent factors β^u, β^i, T-RECS produces a joint embedding $\boldsymbol{z}$ from which it predicts a rating $\hat{\boldsymbol{y}}_r$, a keyphrase explanation $\hat{\boldsymbol{y}}_{kp}$, and a natural language justification $\hat{\boldsymbol{y}}_{just}$ conditioned on $\hat{\boldsymbol{y}}_{kp}$.

Rating Classifier & Keyphrase Explainer.

Our framework classifies the interaction between the user u and item i as positive or negative. Moreover, we predict the keyphrases that describe the overlap of their profiles. Both models are a two-layer feedforward neural network with LeakyRelu activation function. Their respective losses are:

$$\mathcal{L}_r(C^r(\boldsymbol{z}), \boldsymbol{y}_r) = (\hat{\boldsymbol{y}}_r - \boldsymbol{y}_r)^2$$

$$\mathcal{L}_{kp}(C^{kp}(\boldsymbol{z}), \boldsymbol{y}_{kp}) = -\sum_{k=1}^{|K|} y^k_{kp} \log \hat{y}^k_{kp}$$

where $\mathcal{L}_r$ is the mean square error, $\mathcal{L}_{kp}$ the binary cross-entropy, and K the whole set of keyphrases.

Justification Generation.

The last component consists of generating the justification. Inspired by "plan-and-write" [Yao *et al.*, 2019], we advance the personalization of the justification by incorporating the keyphrases $\hat{\boldsymbol{y}}_{kp}$. In other words, T-RECS generates a natural language justification conditioned on the 1. user, 2. item, and 3. aspects of the item that the user would consider important. We encode these via the Aspect Encoder A that takes the average of their word embeddings from the embedding layer in the Transformer. The aspect embedding is denoted by $\boldsymbol{a}_{kp}$ and added to the latent representation: $\tilde{\boldsymbol{z}} = \boldsymbol{z} + \boldsymbol{a}_{kp}$. Based on $\tilde{\boldsymbol{z}}$, the Transformer decoding block computes the output probability $\hat{y}^{t,w}_{just}$ for the word w at time-step t. We train using teacher-forcing and cross-entropy with label smoothing:

$$\mathcal{L}_{just}(D(\boldsymbol{z}, \boldsymbol{a}_{kp}), \boldsymbol{y}_{just}) = -\sum_{t=1}^{|\boldsymbol{y}_{just}|} CE(y^{t,w}_{just}, \hat{y}^{t,w}_{just})$$

We train T-RECS end-to-end and minimize jointly the loss $\boldsymbol{\mathcal{L}} = \lambda_r \mathcal{L}_r + \lambda_{kp}\mathcal{L}_{kp} + \lambda_{just}\mathcal{L}_{just}$, where λ_r, λ_{kp}, and λ_{just} control the impact of each loss. All objectives share the latent representation $\boldsymbol{z}$ and are thus mutually regularized by the function $E(u, i)$ to limit overfitting by any objective.

4.2 Unsupervised Critiquing

The purpose of critiquing is to refine the recommendation based on the user's interaction with the explanation, the keyphrases $\hat{\boldsymbol{y}}_{kp}$, represented with a binary vector. The user critiques either a keyphrase k by setting $\hat{y}^k_{kp} = 0$ (i.e., disagreement) or symmetrically adding a new one (i.e.,$\hat{y}^k_{kp} = 1$).

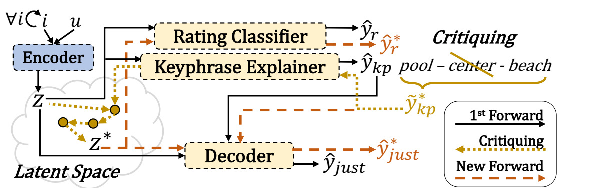


Figure 5: Workflow of considering to recommend items to a user u. We illustrate it for a given item i. **Black** denotes the forward pass to infer the rating $\hat{\boldsymbol{y}}_r$ with the explanations $\hat{\boldsymbol{y}}_{kp}$ and $\hat{\boldsymbol{y}}_{just}$. **Yellow** indicates the critiquing: the user critiques the binary-vector keyphrase explanation $\hat{\boldsymbol{y}}_{kp}$ (e.g., *center*) to $\tilde{\boldsymbol{y}}^*_{kp}$, which modulates the latent space into $\boldsymbol{z}^*$ for each item. **Orange** shows the new forward pass for the subsequent recommendation $\hat{\boldsymbol{y}}^*_r$ and explanations $\hat{\boldsymbol{y}}^*_{kp}$, $\hat{\boldsymbol{y}}^*_{just}$.

We denote the critiqued keyphrase explanation as $\tilde{\boldsymbol{y}}^*_{kp}$. The overall critiquing process is depicted in Fig. 5. Inspired by the recent success in editing the latent space on the unsupervised text attribute transfer task [Wang *et al.*, 2019], we employ the trained Keyphrase Explainer C^{kp} and the critiqued explanation $\tilde{\boldsymbol{y}}^*_{kp}$ to provide the gradient from which we update the latent representation $\boldsymbol{z}$ (depicted in **yellow**). More formally, given a latent representation $\boldsymbol{z}$ and a binary critique vector $\tilde{\boldsymbol{y}}^*_{kp}$, we want to find a new latent representation $\boldsymbol{z}^*$ that will produce a new keyphrase explanation close to the critique, such that $|C^{kp}(\boldsymbol{z}^*) - \tilde{\boldsymbol{y}}^*_{kp}| \leq T$, where T is a threshold. In order to achieve this goal, we iteratively compute the gradient with respect to $\boldsymbol{z}$ instead of the model parameters C^{kp}_θ. We then modify $\boldsymbol{z}$ in the direction of the gradient until we get a new latent representation $\boldsymbol{z}^*$ that C^{kp} considers close enough to $\tilde{\boldsymbol{y}}^*_{kp}$ (shown in **orange**). We emphasize that we use the gradient to modulate $\boldsymbol{z}$ rather than the parameters C^{kp}.

Let denote the gradient as $\boldsymbol{g}_t$ and a decay coefficient as ζ. For each iteration t and $\boldsymbol{z}^*_0 = \boldsymbol{z}$, the modified latent representation $\boldsymbol{z}^*_t$ at the t^{th} iteration can be formulated as follows:

$$\boldsymbol{g}_t = \nabla_{\boldsymbol{z}^*_t}\mathcal{L}_{kp}(C^{kp}(\boldsymbol{z}^*_t), \tilde{\boldsymbol{y}}^*_{kp}); \quad \boldsymbol{z}^*_t = \boldsymbol{z}^*_{t-1} - \zeta^{t-1}\boldsymbol{g}_t/||\boldsymbol{g}_t||_2$$

Because this optimization is nonconvex, there is no guarantee that the difference between the critique vector and the inferred explanation will differ by only T. In our experiments in Section 5.4, we found that a limit of 50 iterations works well, and that the newly induced explanations remain consistent.

5 Experiments

5.1 Experimental Settings

Datasets. We evaluate the quantitative performance of T-RECS using two real-world, publicly available datasets: BeerAdvocate [McAuley and Leskovec, 2013] and HotelRec [Antognini and Faltings, 2020]. They contain 1.5 and 50 million reviews from BeerAdvocate and TripAdvisor. In addition to the overall rating, users also provided five-star aspect ratings. We binarize the ratings with a threshold t: $t > 4$ for hotel reviews and $t > 3.5$ for beer reviews. We further filter out all users with fewer than 20 interactions and sort them chronologically. We keep the first 80% of interactions per user as the training data, leaving the remaining 20% for validation and testing. We sample two justifications per review.

We need to select keyphrases for explanations and critiquing. Hence, we follow the processing in [Wu *et al.*, 2019] to extract 200 keyphrases (distributed uniformly over the aspect categories) from the *markers* on each dataset.

Implementation Details. To extract *markers*, we trained MTM with the hyperparameters reported by the authors. We build the justification history J^u,J^i, with $N_{just} = 32$. We set the embedding and attention dimension to 256 and to 1024 for the feed-forward network. The encoder and decoder consist of two layers of Transformer with 4 attention heads. We use a batch size of 128, dropout of 0.1, and Adam with learning rate 0.001. For critiquing, we choose a threshold and decay coefficient $T = 0.015, \zeta = 0.9$ and $T = 0.01, \zeta = 0.975$ for hotel and beer reviews. We tune all models on the dev set. For reproducibility purposes, we provide details in Appendix I.

5.2 RQ 1: Are *markers* appropriate justifications?

We derive baselines from [Ni *et al.*, 2019]: we split a review into elementary discourse units (EDUs) and apply their classifier to get justifications; it is trained on a manually annotated dataset and generalizes well to other domains. We employ two variants: EDU One and EDU All. The latter includes all justifications, whereas the former includes only one.

We perform a human evaluation using Amazon's Mechanical Turk (see Appendix H for more details) to judge the quality of the justifications extracted from the Markers, EDU One, and EDU All on both datasets. We employ three setups: an evaluator is presented with 1. the three types of justifications; 2. only those from Markers and EDU All; and 3. EDU One instead of EDU All. We sampled 300 reviews (100 per setup) with generated justifications presented in random order. The annotators judged the justifications by choosing the most convincing in the pairwise setups and otherwise using best-worst scaling. We report the win rates for the pairwise comparisons and a normalized score ranging from -1 to +1.

Table 2 shows that justifications extracted from Markers are preferred, on both datasets, more than 80% of the time. Moreover, when compared to EDU All and EDU One, Markers achieve a score of 0.74, three times higher than EDU All. Therefore, justifications extracted from the Markers are significantly better than EDUs, and a single justification cannot explain a review. Fig. 3 shows a sample for comparison.

Table 1: Descriptive statistics of the datasets.

						Avg. #KP per		
Dataset	**#Users**	**#Items**	**#Inter.**	**Dens.**	**KP Cov.**	**Just.**	**Rev.**	**User**
Hotel	72,603	38 896	2.2M	0.08%	97.66%	2.15	3.79	115
Beer	7,304	8,702	1.2M	2.02%	96.87%	3.72	6.97	1,210

Table 2: Human evaluation of explanations in terms of the win rate and the **b**est-**w**orst scaling. A score significantly different than Markers (post hoc Tukey HSD test) is denoted by ** for $p < 0.001$.

		Hotel	*Beer*
Winner	**Loser**	**Win Rate**	**Win Rate**
Markers	EDU All	81%**	77%**
Markers	EDU One	93%**	90%**

	Hotel			*Beer*		
Model	**Score**	**#B**	**#W**	**Score**	**#B**	**#W**
EDU One	-0.95**	1	96	-0.93**	2	95
EDU All	0.21**	24	3	0.20**	23	3
Markers	**0.74**	75	1	**0.73**	75	2

5.3 RQ 2: Does T-RECS generate high-quality, relevant, and personalized explanations?

Natural Language Explanations. We consider five baselines: ExpansionNet [Ni and McAuley, 2018] is a seq-to-seq model with a user, item, aspect, and fusion attention mechanism that generates personalized reviews. DualPC [Sun *et al.*, 2020] and CAML [Chen *et al.*, 2019] generate an explanation based on a rating and the user-item pair. Ref2Seq improves upon ExpansionNet by learning only from historical justifications of a user and an item. AP-Ref2Seq [Ni *et al.*, 2019] extends Ref2Seq with aspect planning [Yao *et al.*, 2019], in which aspects are given during the generation. All models use beam search during testing and the same keyphrases as aspects. We employ BLEU, ROUGE-L, BertScore [Zhang *et al.*, 2020], the perplexity for the fluency, and R_{KW} for the explanation consistency as in [Chen *et al.*, 2020]: the ratio of the target keyphrases present in the generated justifications.

The main results are presented in Table 3 (more in Appendix D). T-RECS achieves the highest scores on both datasets. We note that 1. seq-to-seq models better capture user and item information to produce more relevant justifications, and 2. using a keyphrase plan doubles the performance on average and improving explanation consistency.

We run a human evaluation, with the best models according to R_{KW}, using best-worst scaling on the dimensions: overall, fluency, informativeness, and relevance. We sample 300 explanations and showed them in random order. Table 4 shows that our explanations are largely preferred on all criteria.

Keyphrase Explanations. We compare T-RECS with the popularity baseline and the models proposed in [Wu *et al.*, 2019], which are extended versions of the NCF model [He *et al.*, 2017]. E-NCF and CE-NCF augment the NCF method with an explanation and a critiquing neural component. Also, the authors provide variational variants: VNCF, E-VNCF, and CE-VNCF. Here, we omit NCF and VNCF because they are trained only to predict ratings. We report the following metrics: NDCG, MAP, Precision, and Recall at 10.

Table 3: Generated justifications on automatic evaluation.

	Model	BLEU	R-L	BERT$_{\text{Score}}$	PPL↓	R$_{\text{KW}}$
Hotel	ExpansionNet	0.53	6.91	74.81	28.87	60.09
	DualPC	1.53	16.73	86.76	28.99	13.12
	CAML	1.13	16.67	87.77	29.10	9.38
	Ref2Seq	1.77	16.45	86.74	29.07	13.19
	AP-Ref2Seq	7.28	33.71	88.31	21.31	90.20
	T-RECS	**7.47**	**34.10**	**90.23**	**17.80**	**93.57**
Beer	ExpansionNet	1.22	9.68	72.32	22.28	82.49
	DualPC	2.08	14.68	85.49	21.15	10.60
	CAML	2.43	14.99	85.96	21.29	10.18
	Ref2Seq	3.51	15.96	85.27	22.34	12.10
	AP-Ref2Seq	15.89	46.50	91.35	12.07	91.52
	T-RECS	**16.54**	**47.20**	**91.50**	**10.24**	**94.96**

Table 4: Human evaluation of justifications in terms of best-worst scaling for **O**verall, **F**luency, **I**nformativenss, and **R**elevance. Most scores are significantly different than T-RECS (post hoc Tukey HSD test) with $p < 0.002$. † denotes a nonsignificant score.

	Hotel				*Beer*			
Model	**O**	**F**	**I**	**R**	**O**	**F**	**I**	**R**
ExpansionNet	-0.58	-0.67	-0.52	-0.56	-0.03	-0.31	0.10	-0.01
Ref2Seq	-0.27	-0.19	-0.30	-0.26	-0.69	-0.34	-0.71	-0.69
AP-Ref2Seq	0.30	0.32	0.29	0.29	0.22	0.25†	0.21†	0.25
T-RECS	**0.55**	**0.54**	**0.53**	**0.53**	**0.49**	**0.39**	**0.39**	**0.45**

Table 5 shows that T-RECS outperforms the CE-(V)NCF models by 60%, Pop by 20%, and E-(V)NCF models by 10% to 30% on all datasets. Pop performs better than CE-(V)NCF, showing that many keywords are recurrent in reviews. Thus, predicting keyphrases from the user-item latent space is a natural way to entangle them with (and enable critiquing).

5.4 RQ 3: Can T-RECS enable critiquing?

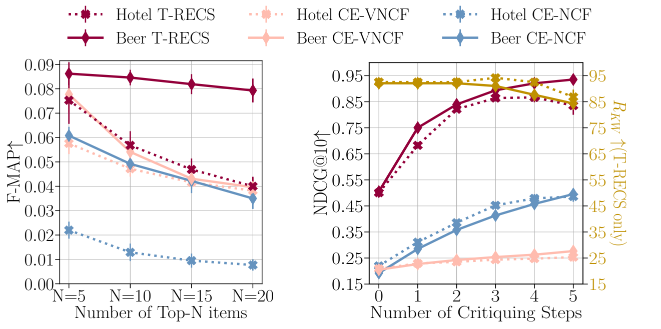
Single-Step Critiquing. For a given user, T-RECS recommends an item and generates personalized explanations, where the user can interact by critiquing one or multiple keyphrases. However, no explicit ground truth exists to evaluate the critiquing. We use F-MAP [Wu *et al.*, 2019] to measure the effect of a critique. Given a user, a set of recommended items $\mathbb{S}$, and a critique k, let $\mathbb{S}_k$ be the set of items containing k in the explanation. The F-MAP measures the ranking difference of the affected items $\mathbb{S}_k$ before and after critiquing k, using the Mean Average Precision at N. A positive F-MAP indicates that the rank of items in $\mathbb{S}_k$ fell after k is critiqued. We compare T-RECS with CE-(V)NCF and average the F-MAP over $5,000$ user-keyphrase pairs.

Fig. 6a presents the F-MAP performance on both datasets. All models show an anticipated positive F-MAP. The performance of T-RECS improves considerably on the beer dataset and is significantly higher for $N \leq 10$ on the hotel dataset. The gap in performance may be caused by the extra loss of the autoencoder, which brings noise during training. T-RECS only iteratively edits the latent representation at test time.

Multi-Step Critiquing. Evaluating multi-step critiquing via ranking is difficult because many items can have the keyphrases of the desired item. Instead, we evaluate whether

Table 5: Keyphrase explanation quality at $N = 10$.

	Hotel				*Beer*			
Model	**NDCG**	**MAP**	**P**	**R**	**NDCG**	**MAP**	**P**	**R**
Pop	0.333	0.208	0.143	0.396	0.250	0.229	0.176	0.253
E-NCF	0.341	0.215	0.137	0.380	0.249	0.220	0.179	0.262
CE-NCF	0.229	0.143	0.092	0.255	0.192	0.172	0.136	0.197
E-VNCF	0.344	0.216	0.139	0.386	0.236	0.210	0.170	0.248
CE-VNCF	0.229	0.134	0.107	0.297	0.203	0.178	0.148	0.215
T-RECS	**0.376**	**0.236**	**0.158**	**0.436**	**0.316**	**0.280**	**0.228**	**0.332**



(a) Falling MAP for different top-N. Error bars show the standard deviation.

(b) Keyphrase prediction over multi-step critiquing with 95% confidence interval. We also report the explanation consistency R_{KW} for T-RECS.

Figure 6: Single- (top) and multi-step (bottom) critiquing.

a system obtains a complete model of the user's preferences following [Pu *et al.*, 2006]. A user expresses his keyphrase preferences iteratively according to a randomly selected liked item. After each step, we evaluate the keyphrase explanations. For T-RECS, we also report the explanation consistency R_{KW}. We run up to five-steps critiques over $1,000$ random selected users and up to $5,000$ random keyphrases for each dataset. Fig. 6b shows that T-RECS builds through the critiques more accurate user profiles and consistent explanations. CE-NCF's top performance is significantly lower than T-RECS, and CE-VNCF plateaus, surely because of the KL divergence regularization, which limits the amount of information stored in the latent space. The explanation quality in T-RECS depends on the accuracy of the user's profile and may become saturated once we find it after four steps.[1]

6 Conclusion

Recommendations can carry much more impact if they are supported by explanations. We presented T-RECS, a multi-task learning Transformer-based recommender, that produces explanations considered significantly superior when evaluated by humans. The second contribution of T-RECS is the user's ability to react to a recommendation by *critiquing* the explanation. We designed an unsupervised method for multi-step critiquing with explanations. Experiments show that T-RECS obtains stable and significant improvement in adapting to the preferences expressed in multi-step critiquing.

[1]We could not compare T-RECS with [Chen *et al.*, 2020] because the authors did not make the code available due to copyright issues.

References

[Antognini and Faltings, 2020] Diego Antognini and Boi Faltings. Hotelrec: a novel very large-scale hotel recommendation dataset. In *the Language Resources and Evaluation Conference*, 2020.

[Antognini and Faltings, 2021] Diego Antognini and Boi Faltings. Rationalization through concepts. In *Findings of the Association for Computational Linguistics: ACL 2021*, Online, 2021. Association for Computational Linguistics.

[Antognini *et al.*, 2021] Diego Antognini, Claudiu Musat, and Boi Faltings. Multi-dimensional explanation of target variables from documents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35, 2021.

[Burke *et al.*, 1996] Robin D. Burke, Kristian J. Hammond, and Benjamin C. Young. Knowledge-based navigation of complex information spaces. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, page 462–468, 1996.

[Catherine and Cohen, 2017] Rose Catherine and William Cohen. Transnets: Learning to transform for recommendation. In *Proceedings of the ACM conference on recommender systems*, 2017.

[Chen and Pu, 2012] Li Chen and Pearl Pu. Critiquing-based recommenders: survey and emerging trends. *User Modeling and User-Adapted Interaction*, 22(1-2), 2012.

[Chen *et al.*, 2018] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. Neural attentional rating regression with review-level explanations. In *Proceedings of the 2018 World Wide Web Conference*, pages 1583–1592, 2018.

[Chen *et al.*, 2019] Zhongxia Chen, Xiting Wang, Xing Xie, Tong Wu, Guoqing Bu, Yining Wang, and Enhong Chen. Co-attentive multi-task learning for explainable recommendation. In *IJCAI*, pages 2137–2143, 2019.

[Chen *et al.*, 2020] Zhongxia Chen, Xiting Wang, Xing Xie, Mehul Parsana, Akshay Soni, Xiang Ao, and Enhong Chen. Towards explainable conversational recommendation. IJCAI, 2020.

[Dong *et al.*, 2017] Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. Learning to generate product reviews from attributes. In *the Conference of the European Association for Computational Linguistics*, pages 623–632, 2017.

[Erkan and Radev, 2004] Günes Erkan and Dragomir R Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of AI research*, 22:457–479, 2004.

[Giannakopoulos *et al.*, 2017] Athanasios Giannakopoulos, Diego Antognini, Claudiu Musat, Andreea Hossmann, and Michael Baeriswyl. Dataset construction via attention for aspect term extraction with distant supervision. In *IEEE International Conference on Data Mining Workshops*, pages 373–380, 2017.

[He *et al.*, 2017] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182, 2017.

[Holtzman *et al.*, 2020] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.

[Hoyer, 2004] Patrik O Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of machine learning research*, 5(Nov):1457–1469, 2004.

[Kendall, 1938] M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[Kunkel *et al.*, 2018] Johannes Kunkel, Tim Donkers, Catalin-Mihai Barbu, and Jürgen Ziegler. Trust-related effects of expertise and similarity cues in human-generated recommendations. In *2nd Workshop on Theory-Informed User Modeling*, 2018.

[Kunkel *et al.*, 2019] Johannes Kunkel, Tim Donkers, Lisa Michael, Catalin-Mihai Barbu, and Jürgen Ziegler. Let me explain: Impact of personal and impersonal explanations on trust in recommender systems. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 2019.

[Li *et al.*, 2017] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. Neural rating regression with abstractive tips generation for recommendation. In *the ACM SIGIR conference on Research and Development in Information Retrieval*, 2017.

[Li *et al.*, 2019] Piji Li, Zihao Wang, Lidong Bing, and Wai Lam. Persona-aware tips generation. In *The World Wide Web Conference*, 2019.

[Luo *et al.*, 2020] Kai Luo, Scott Sanner, Ga Wu, Hanze Li, and Hojin Yang. Latent linear critiquing for conversational recommender systems. In *Proceedings of the 29th International Conference on the World Wide Web*, 2020.

[Mann and Thompson, 1988] William C Mann and Sandra A Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text-interdisciplinary Journal for the Study of Discourse*, 8(3):243–281, 1988.

[McAuley and Leskovec, 2013] Julian McAuley and Jure Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *the ACM conference on Recommender systems*, pages 165–172. ACM, 2013.

[Mnih and Salakhutdinov, 2008] Andriy Mnih and Russ R Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2008.

[Mukherjee and Awadallah, 2020] Subhabrata Mukherjee and Ahmed Hassan Awadallah. Uncertainty-aware self-training for text classification with few labels. *arXiv preprint arXiv:2006.15315*, 2020.

[Musat and Faltings, 2015] Claudiu Cristian Musat and Boi Faltings. Personalizing product rankings using collaborative filtering on opinion-derived topic profiles. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[Ni and McAuley, 2018] Jianmo Ni and Julian McAuley. Personalized review generation by expanding phrases and attending on aspect-aware representations. In *Proceedings of the Association for Computational Linguistics*, pages 706–711, 2018.

[Ni *et al.*, 2019] Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP-IJCNLP)*, 2019.

[Nikolakopoulos *et al.*, 2019] Athanasios N. Nikolakopoulos, Dimitris Berberidis, George Karypis, and Georgios B. Giannakis. Personalized diffusions for top-n recommendation. In *the 13th ACM Conference on Recommender Systems*, pages 260–268, 2019.

[Niu *et al.*, 2020] Yilin Niu, Fangkai Jiao, Mantong Zhou, Ting Yao, Jingfang Xu, and Minlie Huang. A self-training method for machine reading comprehension with soft evidence extraction. In *the Association for Computational Linguistics*, 2020.

[Pu *et al.*, 2006] Pearl Pu, Paolo Viappiani, and Boi Faltings. Increasing user decision accuracy using suggestions. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 121–130, 2006.

[Radford *et al.*, 2019] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[Reilly *et al.*, 2005] James Reilly, Kevin McCarthy, Lorraine McGinty, and Barry Smyth. Explaining compound critiques. *Artificial Intelligence Review*, 2005.

[Ricci *et al.*, 2011] Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer, 2011.

[Sun *et al.*, 2020] Peijie Sun, Le Wu, Kun Zhang, Yanjie Fu, Richang Hong, and Meng Wang. Dual learning for explainable recommendation: Towards unifying user preference prediction and review generation. In *Proceedings of World Wide Web*, 2020.

[Tintarev and Masthoff, 2015] Nava Tintarev and Judith Masthoff. Explaining recommendations: Design and evaluation. In *Recommender systems handbook*, pages 353–382. Springer, 2015.

[Wang *et al.*, 2019] Ke Wang, Hang Hua, and Xiaojun Wan. Controllable unsupervised text attribute transfer via editing entangled latent representation. In *Annual Conference on Neural Information Processing Systems*, pages 11034–11044, 2019.

[Williams and Tou, 1982] Michael D Williams and Frederich N Tou. Rabbit: an interface for database access. In *Proceedings of the ACM Conference*, pages 83–87, 1982.

[Wu *et al.*, 2019] Ga Wu, Kai Luo, Scott Sanner, and Harold Soh. Deep language-based critiquing for recommender systems. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 137–145, 2019.

[Yao *et al.*, 2019] Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7378–7385, 2019.

[Zhang and Curley, 2018] Jingjing Zhang and Shawn P Curley. Exploring explanation effects on consumers' trust in online recommender agents. *International Journal of Human–Computer Interaction*, 34(5):421–432, 2018.

[Zhang *et al.*, 2014] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *the 37th ACM SIGIR conference on Research & development in information retrieval*, pages 83–92, 2014.

[Zhang *et al.*, 2020] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *ICLR 2020*, 2020.

[Zhao *et al.*, 2018] Qian Zhao, Jilin Chen, Minmin Chen, Sagar Jain, Alex Beutel, Francois Belletti, and Ed H Chi. Categorical-attributes-based item classification for recommender systems. In *the 12th ACM Conference on Recommender Systems*, 2018.

A Processing & Filtering *Markers*

The method described in [Antognini *et al.*, 2021] extracts, most of the time, *markers* that consist of long, continuous spans of words. However, sometimes, the *markers* are too short because some reviews do not include enough words to justify a certain aspect rating, or the *markers* stop in the middle of a sentence. Although both are theoretically not wrong, we aim to create fluent and grammatically correct justifications. To this end, we exploit the constituency parse tree to ensure that *markers* are noun/verb phrases. We apply the following steps to the entire set of reviews for each dataset:

1. Compute the constituency parse tree of each review;
2. For each noun and adjective node in the constituency parse tree of a *marker*, if the parent node is a verb or noun phrase, we add its children to the *marker*. We follow the rules in [Giannakopoulos *et al.*, 2017];
3. Filter out *markers* having less than four tokens or including first and third-person pronouns.

In cases where faceted ratings are not available at large cases, [Mukherjee and Awadallah, 2020; Niu *et al.*, 2020] proposed elegant solutions to infer them from 20 or fewer samples. If faceted ratings are not present, one can use the concept rationalizer of [Antognini and Faltings, 2021] to infer *markers*.

B Keyphrase Samples

None of our datasets contains initially preselected keyphrases. We extract 200 keyphrases from the *markers* used to model the user and item profiles. They serve as a basis for the explanation and the critiquing. Table 6 shows some keyphrases for each dataset. We apply the following processing for each dataset, similarly to [Wu *et al.*, 2019]:

1. Group by aspect the *markers* from all reviews. The aspect sets come from the available faceted ratings;[2]
2. For each group of *markers*:
   - Tokenize and lemmatize the entire set of *markers*;
   - Extract unigram lists of high-frequency noun and adjective phrases;
   - Keep the top-k most likely unigrams;
3. Represent each review as a one-hot vector indicating whether each keyphrase occurred in the review.

Another possibility to extract keywords from reviews is to leverage Microsoft Concept Graph[3] as in [Chen *et al.*, 2019; Chen *et al.*, 2020]. However, the provided API is limited to single instance conceptualization.

C Justification Examples

Table 7 and Table 8 present different justifications that are extracted from the hotel and beer reviews. We observe that the *markers* justify the subratings. Although they might be some overlaps between EDU All and Markers, justifications from EDU All often are incomplete or not relevant.

[2]For the hotel reviews: service, cleanliness, value, location, and room. For beer reviews: appearance, smell, mouthfeel, and taste.

[3]https://concept.research.microsoft.com/

Table 6: Some keyphrases mined from the inferred *markers*. We grouped them by aspect for a better understanding.

Dataset	Aspect	Keyphrases
Hotel	Service	bar, lobby, housekeeping, guest
	Cleanliness	carpet, toilet, bedding, cigarette
	Value	price, wifi, quality, motel, gym
	Location	airport, downtown, restaurant shop
	Room	bed, tv, balcony, fridge, microwave
Beer	Appearance	golden, dark, white, foamy
	Aroma	fruit, wheat, citrus, coffee
	Palate	creamy, chewy, syrupy, heavy
	Taste	bitter, sweet, balanced, nutty

D Full Natural Language Explanations Results

We also compare T-RECS with more models than these of Section 5.3: LexRank [Erkan and Radev, 2004], NRT [Li *et al.*, 2017], Item-Rand, Ref2Seq Top-k, and ACMLM [Ni *et al.*, 2019]. LexRank [Erkan and Radev, 2004] is a unsupervised multi-document summarizer that selects an unpersonalized justification among all historical justifications of an item. NRT generates an explanation based on rating and the word distribution of the review. Item-Rand is an unpersonalized baseline which outputs a justification randomly from the justification history J^i of item i. Ref2Seq Top-k is an extension of Ref2Seq, where we explore another decoding strategy called Top-k sampling [Radford *et al.*, 2019], which should be more diverse and suitable on high-entropy tasks [Holtzman *et al.*, 2020]. Finally, ACMLM is an aspect conditional masked language model that randomly chooses a justification from J^i (similar to Item-Rand) and then iteratively edits it into new content by replacing random words. We also include more metrics and $\mathrm{R}_{\mathrm{Sent}}$, which computes the percentages of generated justifications sharing the same polarity as the targets according to a sentiment classifier.[4]

The complete results are presented in Table 9. Interestingly, Item-Rand performs closely to LexRank: the best justification, according to LexRank, is slightly better than a random one. On the other hand, ACMLM edits the latter by randomly replacing tokens with the language model but produces poor quality justification, similarly to [Ni *et al.*, 2019]. Finally, we also observe that the polarities of the generated justifications for beers match nearly perfectly, unlike in hotels where the positive and negative nuances are much harder to capture.

E Full Keyphrase Explanation Results

Table 10 contains complementary results to the keyphrase explanation quality experiment of Section 5.3.

[4]We employ the sentiment classifiers trained jointly with Multi-Target Masker of [Antognini *et al.*, 2021], used to infer the *markers* from which the justifications are extracted.

Table 7: Comparisons of the extracted justifications from different models for two hotels on the hotel dataset. Colors denote aspects while underline denotes EDUs classified as good justifications.

Model	**Casa del Sol Machupicchu**	**Southern Sun Waterfront Cape Town**
Review	the hotel was decent the staff was very friendly. the free pisco sour class with kevin was a nice bonus! however, the rooms were lacking. the wifi was incredibly slow and there was no air conditioning, so it got very hot at night. we couldn't open the windows either because there were so many bugs, birds, and noise. overall, the location is convenient, but was is not worth the price.	this is my second year visiting cape town and staying here. excellent location to business district, convention center, v&a waterfront and access short distance to table mountain. very nice hotel, very friendly staff. breakfast is very good. rooms are nice but bed mattress could be improved as bed is somewhat hard. overall a very nice hotel.
Rating	Overall: 3.0, Service: 3.0, Cleanliness: 4.0, Value: 2.0, Location: 4.0, Room: 3.0	Overall: 4.0, Service: 5.0, Cleanliness: 5.0, Value: 4.0, Location: 5.0, Room: 3.0
Markers	- the rooms were lacking. - the hotel was decent and the staff was very friendly. - overall , the location is convenient , but was is not worth the price. - we could n't open the windows either because there were so many bugs , birds , and noise. - the wifi was incredibly slow and there was no air conditioning , so it got very hot at night.	- breakfast is very good. - very nice hotel , very friendly staff. - rooms are nice but bed mattress could be improved as bed is somewhat hard. - excellent location to business district , convention center , v&a waterfront and access short distance to table mountain.
EDU All	- the hotel was decent - the free pisco sour class with kevin was a nice bonus.	- excellent location to business district , convention center , v&a waterfront and access short distance to table mountain. - very nice hotel , very friendly staff. breakfast is very good. - rooms are nice. - overall a very nice hotel.
EDU One	- the hotel was decent	- very nice hotel , very friendly staff . breakfast is very good

Table 8: Comparisons of the extracted justifications from different models for two beers on the beer dataset. Colors denote aspects while underline denotes EDUs classified as good justifications.

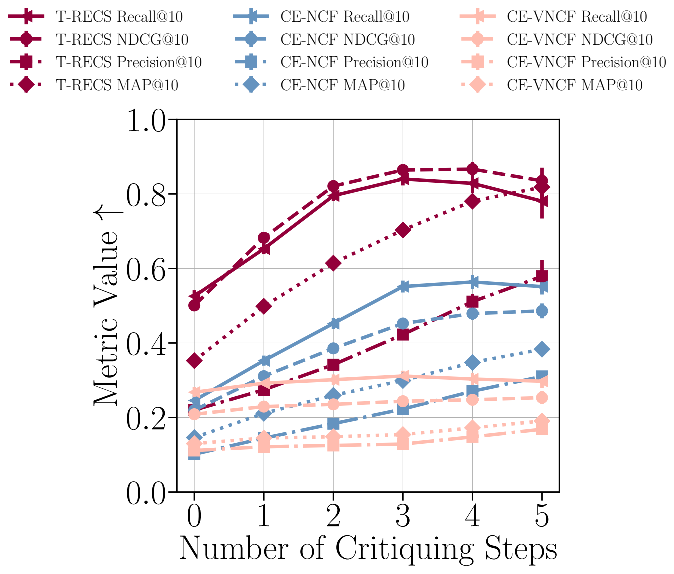
Model	**Saison De Lente**	**Bell's Porter**
Review	poured from a 750ml bottle into a chimay branded chalice. a: cloudy and unfiltered with a nice head that lasts and leaves good amounts of lacing in its tracks. s: sour and bready with apple and yeast hints in there as well. t: dry and hoppy with a nice crisp sour finish. m: medium bodied, high carbonation with big bubbles. d: easy to drink, but i didn't really want more after splitting a 750ml with a buddy of mine.	this beer pours black with a nice big frothy offwhite head. smells or roasted malts, and chocolate. tastes of roasted malt with some chocolate and a hint of coffee. the mouthfeel has medium body and is semi-smooth with some nice carbination. drinkability is decent i could drink a couple. overall a good choice from bell's.
Rating	Overall: 3.0, Appearance: 3.5, Smell: 4.0, Mouthfeel: 3.5, Taste: 3.5	Overall: 3.5, Appearance: 4.0, Smell: 3.5, Mouthfeel: 3.5, Taste: 4.0
Markers	- dry and hoppy with a nice crisp sour finish. - medium bodied , high carbonation with big bubbles. - sour and bready with apple and yeast hints in there as well. - cloudy and unfiltered with a nice head that lasts and leaves good amounts of lacing in its tracks.	- smells or roasted malts , and chocolate. - this beer pours black with a nice big frothy offwhite head. - tastes of roasted malt with some chocolate and a hint of coffee. - the mouthfeel has medium body and is semi smooth with some nice carbination.
EDU All	- medium bodied , high carbonation with big bubbles. - easy to drink	- smells or roasted malts , and chocolate. - tastes of roasted malt with some chocolate and a hint of coffee. - drinkability is decent - overall a good choice
EDU One	- easy to drink	- drinkability is decent

Table 9: Performance of the generated personalized justifications on automatic evaluation.

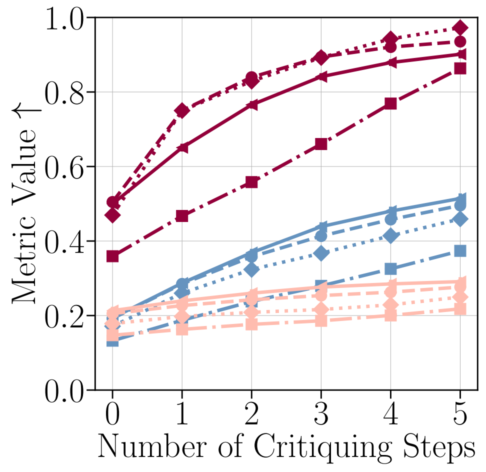
	Model	B-1	B-2	B-3	B-4	R-1	R-2	R-L	BERT$_{\text{Score}}$	PPL↓	R$_{\text{KW}}$	R$_{\text{Sent}}$
Hotel	Item-Rand	11.50	2.88	0.91	0.32	12.65	0.87	9.75	84.20	-	6.92	56.88
	LexRank	12.12	3.31	1.10	0.41	14.74	1.16	10.61	83.91	-	10.32	58.51
	ExpansionNet	4.03	1.95	1.01	0.53	34.22	9.65	6.91	74.81	28.87	60.09	61.38
	NRT	17.00	6.51	3.06	1.51	18.21	2.88	16.08	86.31	29.75	11.44	64.46
	DualPC	18.91	6.88	3.18	1.53	20.12	3.08	16.73	86.76	28.99	13.12	63.54
	CAML	10.93	4.11	2.09	1.13	15.44	2.37	16.67	87.77	29.10	16.17	65.14
	Ref2Seq	17.57	7.03	3.44	1.77	19.07	3.43	16.45	86.74	29.07	13.19	64.40
	Ref2Seq Top-k	12.68	3.46	1.11	0.40	12.67	0.95	10.30	84.29		6.38	58.11
	AP-Ref2Seq	32.04	19.03	11.76	7.28	38.90	14.53	33.71	88.31	21.31	90.20	69.37
	ACMLM	8.60	2.42	1.12	0.62	9.79	0.55	7.23	81.90	-	13.24	60.00
	T-RECS (Ours)	**33.53**	**19.76**	**12.14**	**7.47**	**40.29**	**14.74**	**34.10**	**90.23**	**17.80**	**93.57**	**70.12**
Beer	Item-Rand	10.96	3.02	0.91	0.29	10.28	0.75	8.25	83.39	-	6.70	99.61
	LexRank	12.23	3.58	1.12	0.38	13.81	1.16	9.90	83.42	-	10.79	99.88
	ExpansionNet	6.48	3.59	2.06	1.22	**54.53**	18.24	9.68	72.32	22.28	82.49	**99.99**
	NRT	18.54	8.53	4.40	2.43	17.46	3.61	15.56	85.26	21.22	11.43	**99.99**
	DualPC	18.38	8.10	3.95	2.08	17.61	3.38	14.68	85.49	21.15	10.60	**99.99**
	CAML	12.94	6.5	3.80	2.43	14.63	2.43	14.99	85.96	21.29	10.18	**99.99**
	Ref2Seq	18.75	9.47	5.51	3.51	18.25	4.52	15.96	85.27	22.34	12.10	**99.99**
	Ref2Seq Top-k	13.92	5.02	2.10	1.01	12.36	1.50	10.52	84.14		8.51	99.83
	AP-Ref2Seq	44.84	30.57	21.68	15.89	51.38	23.27	46.50	91.35	12.07	91.52	**99.99**
	ACMLM	7.76	2.54	0.91	0.34	8.33	0.87	6.17	80.94	-	10.33	**99.99**
	T-RECS (Ours)	**46.50**	**31.56**	**22.42**	**16.54**	53.12	**23.86**	**47.20**	**91.50**	**10.24**	**94.96**	**99.99**

Table 10: Performance of personalized keyphrase explanation quality.

		NDCG@N			**MAP@N**			**Precision@N**			**Recall@N**		
	Model	N=5	N=10	N=20	N=5	N=10	N=20	N=5	N=10	N=20	N=5	N=10	N=20
Hotel	UserPop	0.2625	0.3128	0.3581	0.2383	0.1950	0.1501	0.1890	0.1332	0.0892	0.2658	0.3694	0.4886
	ItemPop	0.2801	0.3333	0.3822	0.2533	0.2083	0.1608	0.2041	0.1431	0.0959	0.2866	0.3961	0.5245
	E-NCF	0.2901	0.3410	0.3889	0.2746	0.2146	0.1618	0.1943	0.1366	0.0919	0.2746	0.3802	0.5057
	CE-NCF	0.1929	0.2286	0.2634	0.1825	0.1432	0.1085	0.1290	0.0918	0.0631	0.1809	0.2548	0.3469
	E-VNCF	0.2902	0.3441	0.3925	0.2746	0.2158	0.1634	0.1947	0.1391	0.0932	0.2746	0.3860	0.5132
	CE-VNCF	0.1727	0.2289	0.2761	0.1530	0.1336	0.1115	0.1275	0.1071	0.0767	0.1795	0.2965	0.4200
	T-RECS (Ours)	**0.3158**	**0.3763**	**0.4319**	**0.2919**	**0.2356**	**0.1807**	**0.2223**	**0.1581**	**0.1068**	**0.3109**	**0.4358**	**0.5812**
Beer	UserPop	0.2049	0.2679	0.3357	0.2749	0.2404	0.2014	0.2366	0.1901	0.1445	0.1716	0.2767	0.4207
	ItemPop	0.1948	0.2495	0.3131	0.2653	0.2291	0.1894	0.2267	0.1759	0.1342	0.1618	0.2529	0.3886
	E-NCF	0.1860	0.2485	0.3158	0.2488	0.2204	0.1877	0.2162	0.1789	0.1389	0.1571	0.2618	0.4040
	CE-NCF	0.1471	0.1922	0.2422	0.1967	0.1721	0.1446	0.1687	0.1363	0.1050	0.1227	0.1974	0.3033
	E-VNCF	0.1763	0.2362	0.3055	0.2389	0.2097	0.1797	0.2031	0.1696	0.1356	0.1471	0.2478	0.3943
	CE-VNCF	0.1512	0.2025	0.2595	0.1987	0.1784	0.1532	0.1774	0.1475	0.1155	0.1293	0.2146	0.3352
	T-RECS (Ours)	**0.2394**	**0.3163**	**0.3946**	**0.3127**	**0.2799**	**0.2369**	**0.2800**	**0.2284**	**0.1717**	**0.2048**	**0.3320**	**0.4970**



(a) Results on the **hotel** dataset.



(b) Results on the **beer** dataset.

Figure 7: Multi-step critiquing performance. Keyphrase prediction over multi-step critiquing in terms of Recall@10, NDCG@10, Precision@10, and MAP@10 with 95% confidence interval. a) Results on the hotel dataset, b) on the beer dataset.

F Additional Metrics Multi-Step Critiquing

More metrics of the multi-step critiquing experiment in Section 5.4 are available in Fig. 7.

G RQ 4: Do T-RECS justifications benefit the overall recommendation quality?

In this section, we investigate whether justifications are beneficial to T-RECS and improve overall performance. We assess the performance on three different axes: rating prediction, preference prediction, and Top-N recommendation.

Rating & Preference Prediction

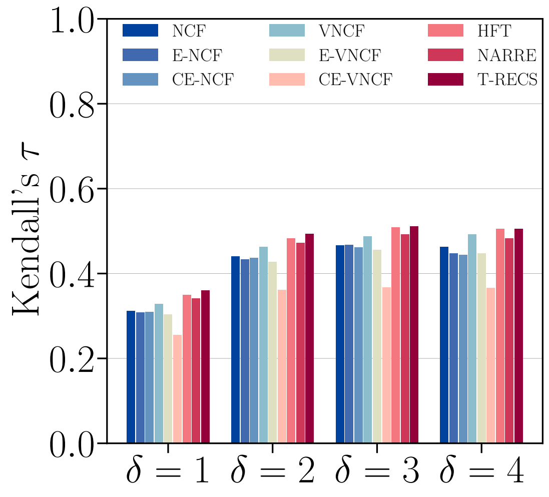
We first analyze recommendation performance by the mean of rating prediction. We utilize the common Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) metrics. However, the rating prediction performance alone does not best reflect the quality of recommendations, because users mainly see the relative ranking of different items [Ricci *et al.*, 2011; Musat and Faltings, 2015]. Consequently, we measure also how well the item rankings computed by T-RECS agree with the user's own rankings as given by his own review ratings. We measure this quality by leveraging the standard metric Kendall's τ rank correlation [Kendall, 1938], computed on all pairs of rated-items by a user in the testing set. Overall, there are 153 954 and 1 769 421 pairs for the hotel and beer datasets, respectively.

Table 11: Performance of the rating prediction.

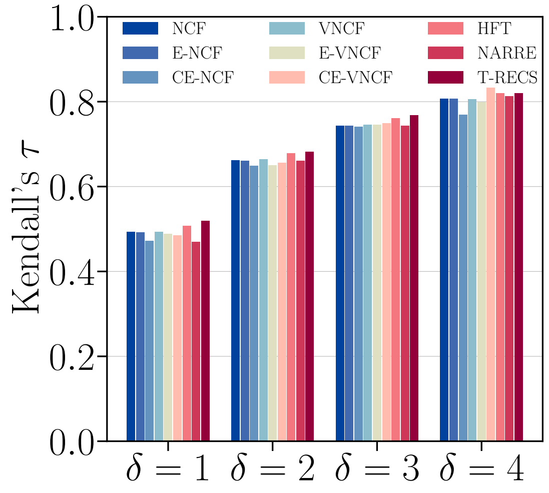
	Hotel			*Beer*		
Model	**MAE**	**RMSE**	$\tau \uparrow$	**MAE**	**RMSE**	$\tau \uparrow$
NMF	0.3825	0.6171	0.2026	0.3885	0.4459	0.4152
PMF	0.3860	0.5855	0.0761	0.3922	0.4512	0.4023
HFT	0.3659	0.4515	0.4584	0.3616	0.4358	0.4773
NARRE	0.3564	0.4431	0.4476	0.3620	0.4377	0.4506
NCF	0.3619	0.4358	0.4200	0.3638	0.4341	0.4696
E-NCF	0.3579	0.4382	0.4145	0.3691	0.4326	0.4685
CE-NCF	0.3552	0.4389	0.4165	0.3663	0.4390	0.4527
VNCF	0.3502	0.4313	0.4408	0.3666	0.4300	0.4706
E-VNCF	0.3494	0.4365	0.4072	0.3627	0.4457	0.4651
CE-VNCF	0.3566	0.4545	0.3502	**0.3614**	0.4330	0.4619
T-RECS	**0.3306**	**0.4305**	**0.4702**	**0.3614**	**0.4295**	**0.4909**

We examine the following baseline methods together with T-RECS: NMF [Hoyer, 2004] is a non-negative matrix factorization model for rating prediction. PMF [Mnih and Salakhutdinov, 2008] is a probabilistic matrix factorization method using ratings for collaborative filtering. HFT [McAuley and Leskovec, 2013] is a strong latent-factor baseline, combined with a topic model aiming to find topics in the review text that correlate with the users' and items' latent factors. NARRE [Chen *et al.*, 2018] is a state-of-the-art model that predicts ratings and reviews' usefulness jointly. Finally, we include the six methods of [Wu *et al.*, 2019] described in Section 5.3.

The results are shown in Table 11. T-RECS consistently outperforms all the baselines, by a wide margin on the hotel dataset, including models based on collaborative filtering with/without review information or models extended with an explanation component and/or a critiquing component. Interestingly, the improvement in the hotel dataset in terms of MAE and RMSE is significantly higher than on the beer dataset. We hypothesize that this behavior is due to the sparsity (see Table 1), which has also been observed in the hotel domain in prior work [Musat and Faltings, 2015; Antognini and Faltings, 2020]. On the beer dataset, we note that reviews contain strong indicators and considerably improve the performance of NARRE and HFT compared to collaborative filtering methods. The extended (V)NCF models with either an explanation and/or a critiquing component improve MAE performance. Therefore, explanations can benefit the recommender systems to improve rating prediction.



(a) Kendall's τ correlation on the **hotel** dataset.



(b) Kendall's τ correlation on the **beer** dataset.

Figure 8: Performance of the preference prediction using Kendall's τ and $\delta = |y_r^i - y_r^j|$.

Table 11 also contains the results in terms of preference prediction. T-RECS achieves up to 0.0136 higher Kendall correlation compared to the best baseline. Surprisingly, we note that CE-VNCF, NMF, and PMF show much worse results on the hotel datasets than on the beer dataset. This highlights that hotel reviews are noisier than beer reviews and emphasizes the importance of capturing users' profiles, where T-RECS does best in comparison to other models.

Preference Prediction

In this experiment, we study a more fine-grained rank correlation. Following [Musat and Faltings, 2015], we analyze the pairwise ranking of rated items by a user, and we impose a minimum value for the rating difference between two items i and j, such that $\delta = |y_r^i - y_r^j|$; the rating difference δ symbolizes the minimum preference strength.

Fig. 8 contains the Kendall's τ evaluation for multiple δ on both datasets. Overall, T-RECS increases the Kendall correlation similarly to other models but performs better on average. We observe that HFT's performance is similar to T-RECS, although slightly lower for most cases. On the beer dataset, we surprisingly note that CE-VNCF obtains a negligible higher score for $\delta = 4$, while significantly underperforming for $\delta < 4$, and especially on the hotel dataset. Finally, the Kendall's τ correlation increases majorly with the strength of preference pairs on the beer dataset and plateaus over $\delta \geq 2$ on the hotel dataset. It highlights that hotel reviews are noisier than beer reviews, and it emphasizes the importance of capturing users' profiles, where T-RECS does best in comparison to other models.

Table 12: Performance of the Top-N recommendation.

		NDCG@N		**Precision@N**		**Recall@N**	
	Model	N=10	N=20	N=10	N=20	N=10	N=20
Hotel	NCF	0.1590	0.2461	0.0231	0.0200	0.2310	0.3991
	E-NCF	0.1579	0.2432	0.0234	0.0200	0.2336	0.4004
	CE-NCF	0.1585	0.2431	0.0235	0.0201	0.2352	0.4028
	VNCF	0.1492	0.2431	0.0220	0.0197	0.2204	0.3932
	E-VNCF	0.1505	0.2395	0.0219	0.0192	0.2188	0.3842
	CE-VNCF	**0.1738**	**0.2662**	0.0221	0.0190	0.2210	0.3809
	T-RECS	0.1674	**0.2662**	**0.0236**	**0.0207**	**0.2358**	**0.4144**
Beer	NCF	0.2172	0.3509	0.0250	0.0212	0.2499	0.4231
	E-NCF	0.2087	0.3363	0.0243	0.0205	0.2426	0.4103
	CE-NCF	0.2226	0.3456	0.0252	0.0205	0.2517	0.4105
	VNCF	0.1943	0.3329	0.0235	0.0211	0.2345	0.4213
	E-VNCF	0.1387	0.2813	0.0158	0.0168	0.1579	0.3362
	CE-VNCF	0.2295	0.3598	0.0263	0.0218	0.2630	0.4352
	T-RECS	**0.2372**	**0.3674**	**0.0269**	**0.0219**	**0.2693**	**0.4390**

Recommendation Performance

We evaluate the performance of T-RECS on the last dimension: Top-N recommendation. We adopt the widely used leave-one-out evaluation protocol [Nikolakopoulos *et al.*, 2019; Zhao *et al.*, 2018]; in particular, for each user, we randomly select one liked item in the test set alongside 99 randomly selected unseen items. We compare T-RECS with the state-of-the-art methods in [Wu *et al.*, 2019]. Finally, we rank the item lists based on the recommendation scores produced by each method, and report the NDCG, Precision, and Recall at different N.

Table 12 presents the main results. Comparing to CE-(V)NCF models, which contain an explanation and critiquing components, our proposed model shows better recommendation performance for almost all metrics on the two datasets. On average, the variants of (V)NCF reach higher results than the original method, which was not the case in the rating prediction and relative rankings tasks (see Section G), unlike T-RECS that shows consistent results.

H Human Evaluation Details

We use Amazon's Mechanical Turk crowdsourcing platform to recruit human annotators to evaluate the quality of extracted justifications and the generated justifications produced by each model. To ensure high-quality of the collected data, we restricted the pool to native English speakers from the

U.S., U.K., Canada, or Austria. Additionally, we set the worker requirements at a 98% approval rate and more than 1000 HITS.

The user interface used to judge the quality of the justifications extracted from different methods, in Section 5.2, is shown in Fig. 9. Another human assessment evaluates the generated justifications (see Section 5.3) on the four dimensions: 1. overall measures the overall subjective quality; 2. fluency represents the readability; 3. informativeness indicates whether the justification contains information pertinent to the user; 4. relevance measures how relevant the information is to an item. The interface is available in Fig. 10

I Additional Training Details

I.1 Tuning

We build the justification history J^u,J^i, with $N_{just} = 32$. In T-RECS, we set the embedding, latent, and self-attention dimension size to 256, and the dimension of the feed-forward network to 1024. The encoder and decoder consist of two layers of Transformer with 4 attention heads. We use a batch size of 128, dropout of 0.1, 4000 warm-up steps, smoothing parameter $\varepsilon = 0.1$, and Adam with learning rate $0.001, \beta_1 = 0.9, \beta_2 = 0.98$, and $\epsilon = 10^{-9}$. The Rating Classifier and Keyphrase Explainer are two layers of 128 and 64 dimensions with LeakyReLU ($\alpha = 0.2$). For critiquing, we choose a threshold and decay coefficient $T = 0.015, \zeta = 0.9$ and $T = 0.01, \zeta = 0.975$ for hotel and beer reviews, respectively. We use the code from the authors for most models. We tune all models on the dev set. We have operated a random search over 10 trials. We chose the models achieving the lowest validation loss. The range of hyperparameters are the following for T-RECS (similar for other models):

- Learning rate: $[0.001, 0.0001]$;
- Max epochs: $[100, 200, 300]$;
- Batch size: $[128]$;
- Hidden size encoder/decoder: $[256]$;
- Attention heads: $[4]$;
- Number of layers: $[2]$;
- Dropout encoder: $[0.0, 0.1, 0.2, 0.3, 0.4, 0.5]$;
- Dropout decoder: $[0.0, 0.1, 0.2, 0.3]$;
- General dropout: $[0.0, 0.1, 0.2]$;
- Warmup: $[2000, 4000, 8000, 16000]$;
- λ_r, λ_{kp}, λ_{just}: [1.0];

Most of the time, the model converges under 20 epochs. For critiquing, we employed:

- Decay coefficient ζ: $[0.5, 0.75, 0.8, 0.9, 0.95]$;
- Max iterations: $[25, 50, 75, 100, 200]$;
- Threshold: $[0.015, 0.01, 0.005]$;

I.2 Hardware / Software

- **CPU**: 2x Intel Xeon E5-2680 v3 (Haswell), 2x 12 cores, 24 threads, 2.5 GHz, 30 MB cache;
- **RAM**: 16x 16GB DDR4-2133;
- **GPU**: 2x Nvidia Titan X Maxwell;
- **OS**: Ubuntu 18.04;
- **Software**: Python 3.6, PyTorch 1.3.0, CUDA 10.0.

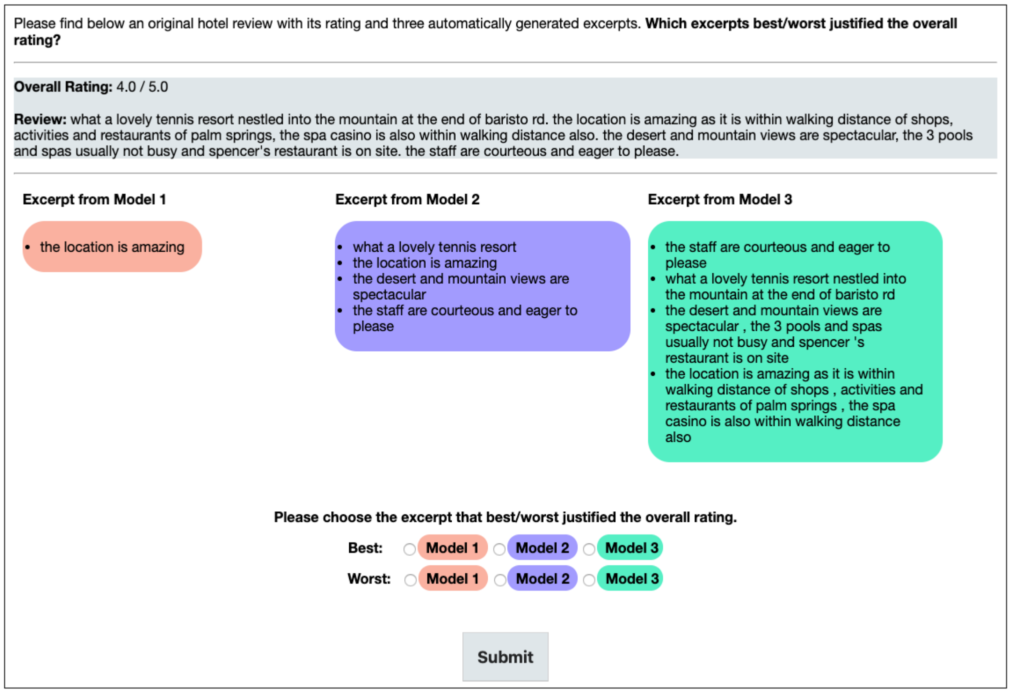
I.3 Running Time

Currently, our code is not optimized, and the critiquing experiment has been run on CPU. In this setting, the critique of a user takes less than 2.5 seconds. By optimizing the code, batching critiques together, and leveraging GPUs/TPUs, we would expect a significant gain. In practice, we could also limit the critique to the Top-N items (e.g., 100 or 1000). To keep track of the critiques and user representations, one could use a database to avoid recomputing the previous critiques' representations.

I.4 Addressing Users without Reviews

The cold-start problem is not particular to our method but a general problem in recommendation. However, our system could infer $\boldsymbol{\gamma^u}$ and $\boldsymbol{\beta^u}$ for users without reviews and then computes the initial latent representation $\boldsymbol{z}$ as in Section 4. We propose the following strategies:

1. **Users with ratings but no reviews**: we could leverage collaborative filtering techniques to identify users with similar ratings and build an initial representation.
2. **New users**: following the observation of [Zhang *et al.*, 2014; Musat and Faltings, 2015]: "users write opinions about the topics they care about" (mentioned in Section 3), we could ask new users to write about the different aspects they deem important. Their initial representation is an aggregation of other users with similar writing. Another option is to ask the users to select items they like based on a subset of items and build a profile from these preferences (see above).



Please find below an original hotel review with its rating and three automatically generated excerpts. **Which excerpts best/worst justified the overall rating?**

Overall Rating: 4.0 / 5.0

Review: what a lovely tennis resort nestled into the mountain at the end of baristo rd. the location is amazing as it is within walking distance of shops, activities and restaurants of palm springs, the spa casino is also within walking distance also. the desert and mountain views are spectacular, the 3 pools and spas usually not busy and spencer's restaurant is on site. the staff are courteous and eager to please.

Excerpt from Model 1

- the location is amazing

Excerpt from Model 2

- what a lovely tennis resort
- the location is amazing
- the desert and mountain views are spectacular
- the staff are courteous and eager to please

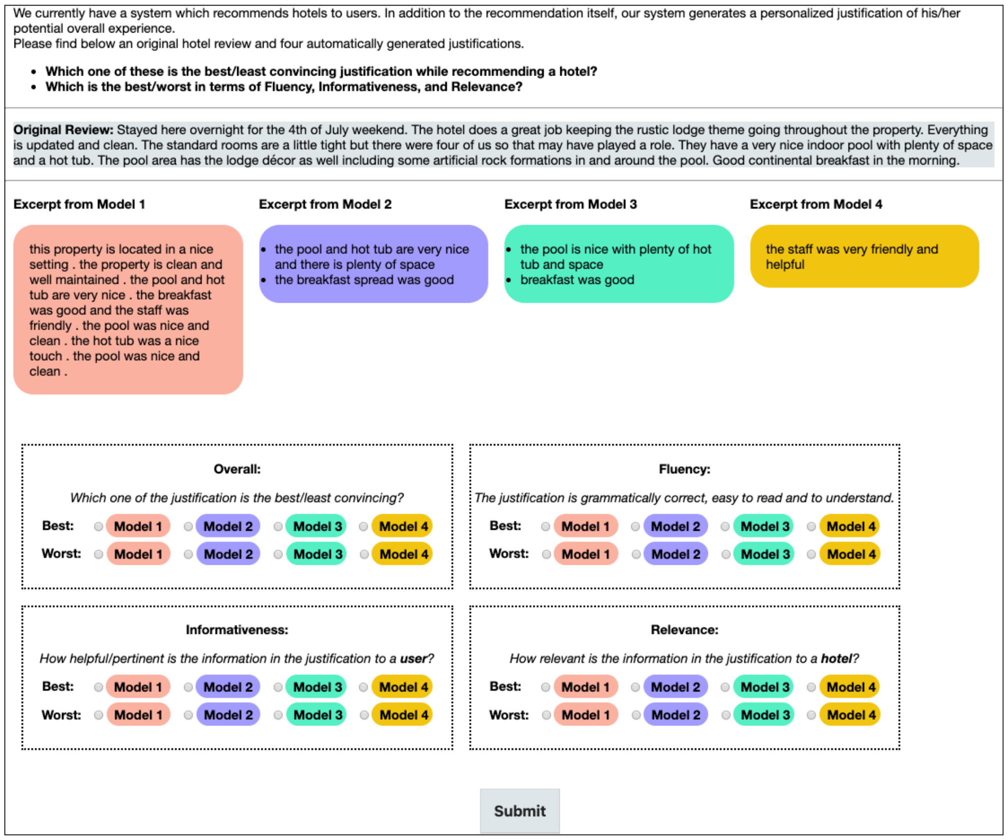
Excerpt from Model 3

- the staff are courteous and eager to please
- what a lovely tennis resort nestled into the mountain at the end of baristo rd
- the desert and mountain views are spectacular , the 3 pools and spas usually not busy and spencer 's restaurant is on site
- the location is amazing as it is within walking distance of shops , activities and restaurants of palm springs , the spa casino is also within walking distance also

Please choose the excerpt that best/worst justified the overall rating.

Best:	Model 1	Model 2	Model 3
Worst:	Model 1	Model 2	Model 3

Figure 9: Annotation platform for judging the quality of extracted justifications from different methods. The justifications are shown in random order for each comparison. In this example, our method corresponds to the third model.



We currently have a system which recommends hotels to users. In addition to the recommendation itself, our system generates a personalized justification of his/her potential overall experience.
Please find below an original hotel review and four automatically generated justifications.

- **Which one of these is the best/least convincing justification while recommending a hotel?**
- **Which is the best/worst in terms of Fluency, Informativeness, and Relevance?**

Original Review: Stayed here overnight for the 4th of July weekend. The hotel does a great job keeping the rustic lodge theme going throughout the property. Everything is updated and clean. The standard rooms are a little tight but there were four of us so that may have played a role. They have a very nice indoor pool with plenty of space and a hot tub. The pool area has the lodge décor as well including some artificial rock formations in and around the pool. Good continental breakfast in the morning.

Excerpt from Model 1

this property is located in a nice setting . the property is clean and well maintained . the pool and hot tub are very nice . the breakfast was good and the staff was friendly . the pool was nice and clean . the hot tub was a nice touch . the pool was nice and clean .

Excerpt from Model 2

- the pool and hot tub are very nice and there is plenty of space
- the breakfast spread was good

Excerpt from Model 3

- the pool is nice with plenty of hot tub and space
- breakfast was good

Excerpt from Model 4

the staff was very friendly and helpful

Overall:

Which one of the justification is the best/least convincing?

Best:	Model 1	Model 2	Model 3	Model 4
Worst:	Model 1	Model 2	Model 3	Model 4

Fluency:

The justification is grammatically correct, easy to read and to understand.

Best:	Model 1	Model 2	Model 3	Model 4
Worst:	Model 1	Model 2	Model 3	Model 4

Informativeness:

*How helpful/pertinent is the information in the justification to a **user**?*

Best:	Model 1	Model 2	Model 3	Model 4
Worst:	Model 1	Model 2	Model 3	Model 4

Relevance:

*How relevant is the information in the justification to a **hotel**?*

Best:	Model 1	Model 2	Model 3	Model 4
Worst:	Model 1	Model 2	Model 3	Model 4

Figure 10: Annotation platform for judging the quality of generated justifications from different methods, on four dimensions. The justifications are shown in random order for each comparison. In this example, our method corresponds to the second model.