Distributional Contrastive Embedding for Clarification-based Conversational Critiquing

Tianshu Shen †
University of Toronto
Toronto, ON, Canada
tina.shen@mail.utoronto.ca

Zheda Mai † ‡
Optimy AI
Toronto, ON, Canada
zheda.mai@mail.utoronto.ca

Ga Wu
Twitter
Toronto, ON, Canada
gaw@twitter.com

Scott Sanner*
University of Toronto
Toronto, ON, Canada
ssanner@mie.utoronto.ca

ABSTRACT

Managing uncertainty in preferences is core to creating the next generation of conversational recommender systems (CRS). However, an often-overlooked element of conversational interaction is the role of clarification. Users are notoriously noisy at revealing their preferences, and a common error is being unnecessarily specific, e.g., suggesting "chicken fingers" when a restaurant with a "kids menu" was the intended preference. Correcting such errors requires reasoning about the level of generality and specificity of preferences and verifying that the user has expressed the correct level of generality. To this end, we propose a novel clarification-based conversational critiquing framework that allows the system to clarify user preferences as it accepts critiques. To support clarification, we propose the use of distributional embeddings that can capture the specificity and generality of concepts through distributional coverage while facilitating state-of-the-art embedding-based recommendation methods. Specifically, we incorporate Distributional Contrastive Embeddings of critiqueable keyphrases with user preference embeddings in a Variational Autoencoder recommendation framework that we term DCE-VAE. Our experiments show that our proposed DCE-VAE is (1) competitive in terms of general performance in comparison to state-of-the-art recommenders and (2) supports effective clarification-based critiquing in comparison to alternative clarification baselines. In summary, this work adds a new dimension of clarification to enhance the well-known critiquing framework along with a novel data-driven distributional embedding for clarification suggestions that significantly improves the efficacy of user interaction with critiquing-based CRSs.

KEYWORDS

Conversational Recommendation Systems, Clarification-based Critiquing, Distributional Latent Embedding

ACM Reference Format:
Tianshu Shen †, Zheda Mai † ‡, Ga Wu, and Scott Sanner. 2022. Distributional Contrastive Embedding for Clarification-based Conversational Critiquing. 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/10.1145/3485447.3512114

1 INTRODUCTION

With the rise of language-based intelligent assistants such as Apple Siri, and Google Assistant, there is a growing interest in conversational recommender systems (CRS) [7, 13, 17]. Many recent works aim to integrate recommendation into conversational interaction via *preference elicitation* [30, 33, 40] or *critiquing* [1, 5, 6, 37] that orchestrate conversational interaction while iteratively zeroing in on the user's session-based preferences. However, an often overlooked problem of conversational recommendation is that the clarity of preference statements often varies due to an observed user tendency to use overly-specific or overly-generic language [12, 24], which can significantly impact recommendation performance. Hence, preference clarification is a critical research direction for improving the efficacy of conversational recommender system interaction.

Clarifying a user's intention often requires reasoning about the level of generality and specificity of the user feedback to verify if the user has expressed the correct level of generality. Unfortunately, the generality or specificity of natural language feedback is highly-nuanced and domain-specific. For example, to generalize an over-specialized preference "chicken fingers" for restaurant recommendation, we need to identify other keyphrases whose distributional semantics significantly overlaps with the given preference. In this case, an appropriately generalized keyphrase preference would be "kids menu". While many existing works on conversational recommendation [22, 40, 41] reason about relationships between keyphrase embeddings, they do so with point embeddings that do not capture any notion of the distributional extent of keyphrase meaning required to reason about generality and specificity.

To address this technical gap, we propose a novel clarification-based conversational recommendation framework that aims to clarify potentially inaccurate critiquing feedback. The framework has two components: (1) a critiquing-based recommender system that

*Affiliate to Vector Institute of Artificial Intelligence, Toronto
†These authors contributed equally to this work
‡Contributions were made while the author was at the University of Toronto.

WWW '22, April 25–29, 2022, Virtual Event, Lyon, France

ACM ISBN 978-1-4503-9096-5/22/04…$15.00
https://doi.org/10.1145/3485447.3512114

supports learning distributions for preferential keyphrase embeddings and (2) a query clarification mechanism that provides the user with an option to choose a suggested clarification during the user-system interaction. For (1), we contribute a Distributional Contrastive Embedding extension of the Variational Autoencoder collaborative filtering recommendation framework [20] that we term DCE-VAE to support multivariate Gaussian embeddings for both keyphrases and user preferences. We then leverage DCE-VAE's keyphrase embedding distributions to generate a *Keyphrase Knowledge Tree* (KKT), which automatically extracts domain-specific subsumption relationships among keyphrases in the context of a given recommendation domain. For (2), we introduce a clarification step into the conversational critiquing workflow that leverages the KKT to suggest a potentially improved keyphrase that the user may optionally accept as a replacement critique.

We conducted various experiments to verify the proposed DCE-VAE against a state-of-the-art conversational recommender system (as well as various ablations) on two real datasets. The results demonstrate two key advantages of this work: (1) Conversational recommendation with a clarification step usually provides better performance vs. *sans* clarification. (2) Our DCE-VAE recommender model supports critique clarification better than prior work due to its ability to reason about distributional overlap among keyphrase embeddings required to reason about specificity and generality.

2 RELATED WORK

2.1 Notation

Before proceeding, we define the notation used in this paper:

- $\mathbf{r}$: The implicit feedback vector for a user with length of total number of items.
- $f_\vartheta(\mathbf{r})$: encoding function that maps a user's rating history $\mathbf{r}$ to the diagonal Gaussian embedding, where mean is denoted as $f^\mu_\vartheta(\mathbf{r})$ and variance is denoted as $f^\sigma_\vartheta(\mathbf{r})$.
- $\mathbf{z}$: : A vector with length d. This is the user latent representation (user embedding) vector. The latent representation is sampled from a Gaussian distribution $\mathcal{N}(f^\mu_\vartheta(\mathbf{r}), f^\sigma_\vartheta(\mathbf{r}))$ in VAE.
- $\mathbf{k}$: The keyphrase vector reflecting a user's keyphrase usage. We use $\mathbf{k}_\pi$ to represent a permutation of $\mathbf{k}$, where $\mathbf{k}_{\pi(i)}$ represents the ith keyphrase of the permutation.
- $f_\psi(k_i)$: An embedding lookup function for a keyphrase k_i. Similar to the rating encoder, the dual outputs of this function (i.e., two heads) represent the mean $f^\mu_\psi(k_i)$ and variance $f^\sigma_\psi(k_i)$.
- ϕ: a Kernel $(\mathbb{R}^d, \mathbb{R}^d) \times (\mathbb{R}^d, \mathbb{R}^d) \rightarrow \mathbb{R}$, measuring the similarity (distributional overlap) of two diagonal Gaussian distributions.
- $\cdot^{(t)}$: We use braced superscript to represent the time step in the conversational environment. For example, $\mathbf{a}^{(t)}$ denotes user's feedback at time step t.

2.2 Critiquing-based Recommender Systems

Critiquing [1, 5, 6, 37] is a CRS method that incrementally improves recommendations online by adapting multi-turn user feedback on previously recommended items. The workflow of critiquing-based recommender systems is illustrated in Figure 1(a), where each recommendation is followed by user critiques, and the system will recommend more appropriate items based on this feedback. This step repeats multiple times until the recommendation is accepted or the stopping criteria are met. Most previous work attempted to refine recommendations by leveraging users' feedback directly on items or known item attributes [6, 34]. However, this approach may not be effective for language-based critiques that are inherently noisy and ambiguous compared to ground truth item attributes [4].

More recent efforts focus on keyphrase-based critiquing where users interact with a large set of descriptive keyphrases mined from user reviews and embedded in the same latent space as user-item embeddings. A key question is how keyphrase critique embeddings should modulate embeddings of user preferences and the recent literature offers many potential solutions. CE-NCF [37] adds critiquing capabilities to Neural Collaborative Filtering (NCF) [10]. CE-VAE substituted the NCF backbone with VAE-CF [20] and proposed a novel variational approach to facilitate language-based modulation of preference embeddings [23]. BK-VAE used Concept Activation Vectors (CAVs) [15] to determine the alignment of keyphrase embeddings with user embeddings in VAE-CF and applied a Bayesian update to user beliefs after each critique [41].

2.3 Limitations

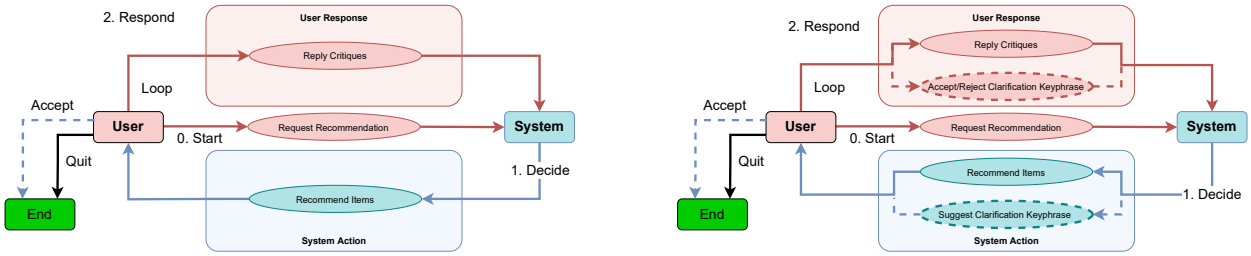
While the above works have made significant technical advances in critiquing-based CRSs, the existing critiquing workflow has two limitations that detract from its practical usage:

- A fundamental assumption of the existing workflow is that users can accurately express their preferences during the critiquing cycle. Unfortunately, users are notoriously noisy at revealing their preferences in reality [12, 21]. Through a user study, [24] shows that a significant amount of elicitation is for clarification purposes — namely achieving the right level of specificity or generality when interpreting user preference feedback.
- In the current workflow, the system makes a recommendation after receiving each user feedback. However, there might exist misalignment between the user and system's understanding of the critiqued keyphrases [4]. Therefore, making recommendations without confirming the meaning of the critiqued keyphrases with users may propagate the misunderstanding into the next round and needlessly result in additional turns to recover.

3 CLARIFICATION IN CRITIQUING-BASED CONVERSATIONAL RECOMMENDERS

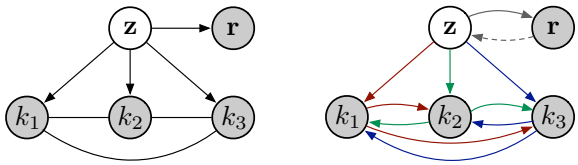
For a critiquing-based system as shown in Figure 1(a), the user feedback at each conversation iteration is presumed to be a keyphrase preference that is not represented by the current recommendations. In our clarification-based critiquing workflow shown in Figure 1(b), the system suggests a clarification keyphrase based on the most recent user critique. If the clarification keyphrase matches the user's intention, the user may accept it as the replacement of his/her initial input (critiqued keyphrase); otherwise, the clarification could be rejected. The clarification step could also be automatically bypassed when certain stopping conditions are met, which we describe later.

There are three key questions and associated challenges that we need to address in order to fulfill the clarification-based critiquing workflow of Figure 1(b): (1) How can we co-embed user preferences and keyphrases meaningfully such that we can leverage embeddings of keyphrase feedback to appropriately modulate user preference



(a) Conventional Critiquing Workflow **(b) Clarification-based Critiquing Workflow**

Figure 1: (a) In the conventional critiquing workflow, the system adjusts its recommendation after each user critique, which may not be effective when the user's critique is overly specific or generic. (b) The clarification step introduced in this work aims to confirm if the user expressed the correct level of generality in his/her critiques and attempts to improve the quality of preference feedback by proposing suggested alternatives that the user may optionally accept as a replacement critique.



(a) Probabilistic Graphical Model (b) Lowerbound Approximation

Figure 2: (a) Probabilistic Graphical Model of the likelihood of recommendations and keyphrases. (b) Model corresponding to our variational lower bound approximation of (a). Colors in (b) show the relaxations given different conditional models π described in Equations 4 and 5.

embeddings and uncertainty? (2) How can we build a keyphrase hierarchy (tree) that automatically extracts the relative specificity and generality of critiquable keyphrases in a specific recommendation context? (3) How can we generate reasonable clarification alternatives to suggest based on users' critiqued keyphrases?

3.1 Distributional Contrastive Embeddings

While co-embedding user and keyphrase information has been extensively studied along with various state-of-the-art deep critiquing approaches [2, 23], these works limit the co-embedded representations of keyphrases and user preferences into vectors (i.e., point embeddings), where representation uncertainty is disregarded. The direct consequence of such simplification is that subsumption of keyphrases cannot be easily mined from such embeddings. While mining subsumption might be unnecessary for domain experts who can accurately express their intention using short phrases, it is valuable to assist regular users with nuanced and domain-specific keyphrases for more effective user-system conversation. E.g., it may be helpful to suggest "spicy" to replace a sequence of overly-specific keyphrases such as "pepper", "hot paprika", "chili", "piquant", etc.

Hence, we now present a novel distributional co-embedding model called Distributional Contrastive Embedding (DCE). Specifically, we extend the well-known Variational Auto-encoder-based recommender system (VAE-CF [20]) with additional distributional embeddings for critiquable keyphrases.

3.1.1 Objective. Given users' historical interaction records $\mathbf{r}_u$ and the corresponding keyphrases $\mathbf{k}_u$ (extracted from item descriptions[1]), we aim to maximize the observation log likelihood such that $\sum_u^U \log p(\mathbf{r}_u, \mathbf{k}_u)$. The corresponding Probabilistic Graphical Model is shown in Figure 2(a).

The joint log-likelihood for a single user can be factorized into two components such that

$$\log p(\mathbf{r}, \mathbf{k}) = \log p(\mathbf{r}) + \log p(\mathbf{k}|\mathbf{r}), \tag{1}$$

where the marginal rating likelihood $\log p(\mathbf{r})$ can be modeled through its variational lower-bound (as a Variational Auto-encoder)

$$\log p(\mathbf{r}) \geq \underbrace{\mathbb{E}_{q_\vartheta(\mathbf{z}|\mathbf{r})}[\log p_\theta(\mathbf{r}|\mathbf{z})]}_{\text{① Interaction Prediction}} + \underbrace{\beta KL[q_\vartheta(\mathbf{z}|\mathbf{r})||p(\mathbf{z})]}_{\text{② Latent Regularization}}. \tag{2}$$

Here, β is the hyperparameter to weight the KL term as motivated in the Beta-VAE [11].

For the conditional term, $\log p(\mathbf{k}|\mathbf{r})$, however, instead of making a conditional independence assumption such that $\log p(\mathbf{k}|\mathbf{r}) = \sum_i \log p(k_i|\mathbf{r})$, we propose to explicitly model pair-wise correlations among keyphrases. Specifically, by applying the chain rule, the conditional likelihood $p(\mathbf{k}|\mathbf{r})$ can be decomposed as

$$p(\mathbf{k}|\mathbf{r}) = p(k_1|\mathbf{r})p(k_2|k_1)p(k_3|k_2, k_1)\cdots p(k_N|k_{N-1}\cdots k_1). \tag{3}$$

By assuming the probability of observing a keyphrase only depends on the first keyphrase observed $p(k_i|k_{i-1}\cdots k_1) = p(k_i|k_1)$, we can obtain the following relaxed expression

$$p(\mathbf{k}|\mathbf{r}) = p(k_1|\mathbf{r})p(k_2|k_1)p(k_3|k_1)\cdots p(k_N|k_1). \tag{4}$$

We note there are multiple relaxation options resulting in an equivalent final objective including a Markov chain conditioning assumption $p(k_i|k_{i-1}\cdots k_1) = p(k_i|k_{i-1})$ discussed in Appendix B.

Since the relaxation described above is order-sensitive (as it uses k_1 as an anchor point), optimizing it may result in a biased model. To address the order sensitivity issue, we alternatively optimize an expectation among permutations of the orders. Specifically, we define $\pi \in \Gamma$ as one of the possible permutations for the keyphrases

[1]The keyphrases could be extracted from informal descriptions of all items the user interacted with. E.g. other users' comments could be viewed as informal descriptions.

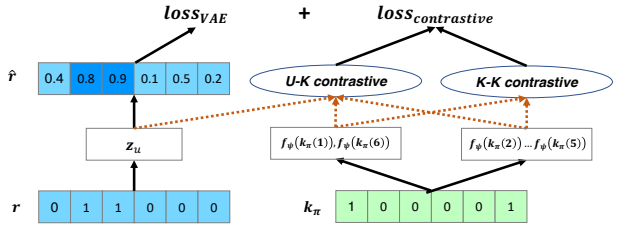


Figure 3: Architecture and training objectives of the proposed recommendation model. In addition to the VAE-CF architecture (and its loss function), we introduce additional components to explicitly model keyphrase embedding distributions through contrastive learning.

(e.g., $\pi(1)$ returns the index of the first anchoring keyphrase in the permuted keyphrase list). The unbiased estimation of $p(\mathbf{k}|\mathbf{r})$ is

$$\begin{aligned} p(\mathbf{k}|\mathbf{r}) &= \int_{\pi\in\Gamma} p(\mathbf{k}_\pi|\mathbf{r})p(\pi)d\pi = \mathbb{E}_{\pi\sim\Gamma}[p(\mathbf{k}_\pi|\mathbf{r})] \\ &= \frac{1}{|\Gamma|}\sum_{\pi\in\Gamma}\left[p(k_{\pi(1)}|\mathbf{r})\prod_{i=2}^{N}p(k_{\pi(i)}|k_{\pi(1)})\right], \end{aligned} \tag{5}$$

where the permutation prior distribution $p(\pi)$ is uniform.

Since the objective is in logarithm form $\log p(\mathbf{k}|\mathbf{r})$, we optimize its lower bound to avoid computing the log-expectation term. Specifically, the conditional term in our optimization objective is

$$\begin{aligned} &\log p(\mathbf{k}|\mathbf{r}) = \log \mathbb{E}_\pi[p(\mathbf{k}_\pi|\mathbf{r})] \geq \mathbb{E}_\pi[\log(p(\mathbf{k}_\pi|\mathbf{r})] \\ &= \frac{1}{|\Gamma|}\Big[\underbrace{\sum_{\pi\in\Gamma}[\log p(k_{\pi(1)}|\mathbf{r})}_{\text{③ user-keyphrase contrast}} + \underbrace{\sum_{i=2}^{N}\log p(k_{\pi(i)}|k_{\pi(1)})}_{\text{④ keyphrase-keyphrase contrast}}\Big]. \end{aligned} \tag{6}$$

3.1.2 Training Strategy. Directly optimizing the conditional probabilities $p(k_{\pi(1)}|\mathbf{r})$ or $p(k_{\pi(i)}|k_{\pi(1)})$ in Equation 6 is barely tractable since computing the marginal probabilities would need to iterate over all possible keyphrases, where the number of keyphrases in the conversational system may up to thousands. E.g.

$$\log p(k_{\pi(1)}|\mathbf{r}) = \log p(k_{\pi(1)},\mathbf{r}) - \underbrace{\log\sum_i p(k_{\pi(i)},\mathbf{r})}_{\text{marginal probability}}. \tag{7}$$

The challenge here is similar to that of training a skip-gram model in word embedding tasks. In particular, we can view the user rating $\mathbf{r}$ as the target word, whereas keyphrases will correspond to the context.

In this work, we leverage Noise Contrastive Estimation (NCE [8]) to optimize the conditional probabilities in a similar fashion of latest word embedding training [9, 18, 26]. Specifically, according to NCE, maximizing Equation 7 is equivalent (under some assumptions) to maximizing the following objective function

$$\log\phi(k_{\pi(1)},\mathbf{r}) - \sum_{i'\sim U[1,N]}^{N'}\log\phi(k_{\pi(i')},\mathbf{r}) \tag{8}$$

where ϕ is a kernel function that estimates the similarity of the two inputs variable, N' denotes the total number of negative samples, and i' denotes the index of the sample.

The remaining problem is how to define the kernel function. Note, while the well-known sigmoid-dot-product function

$$\phi(k_{\pi(1)},\mathbf{r}) = \sigma(f_\psi(k_{\pi(1)})^\top f_\vartheta(\mathbf{r})) \tag{9}$$

is sufficient for vector embedding models, it fails to include the desired representation uncertainty into the computation scope. Here, encoding (or simple lookup) function f_ψ and f_ϑ are parameterized for optimization, and Sigmoid function σ limits the similarity computation into a valid range $[0, 1]$.

To address the aforementioned issue, we leverage the Bhattacharyya Kernel [14]. Concretely, the Bhattacharyya Kernel takes two multivariate Gaussian distributions (with diagonal covariance) as inputs and estimate the kernel $\phi(k_{\pi(1)},\mathbf{r})$ as follows

$$\int_x \mathcal{N}\left(x; f_\psi^\mu(k_{\pi(1)}), f_\psi^\sigma(k_{\pi(1)})\right)^{\frac{1}{2}}\mathcal{N}\left(x; f_\vartheta^\mu(\mathbf{r}), f_\vartheta^\sigma(\mathbf{r})\right)^{\frac{1}{2}}dx, \tag{10}$$

where the outputs of encoding functions are, now, branched into mean and variance of the Gaussian distribution, which we call *distributional embeddings*. At this point, we can now maximize $p(k_{\pi(1)}|\mathbf{r})$ with the NCE objective described in Equation 8. The same kernel setting and NCE training strategy can also apply to maximizing $\log p(k_{\pi(i)}|k_{\pi(1)})$. And, to distinguish the two different NCE objectives, we call the former one *user-keyphrase contrast* whereas the later one *keyphrase-keyphrase contrast*.

Combining Equations (1)-(10), we have our full objective function with a total of four sub-objectives that are indexed with the circled numbers in the equations. Figure 2(b) shows the actual objective we optimize.

3.2 Keyphrase Knowledge Tree

Given the distributional embeddings learned through the previously described training objective, we are now ready to create a *Keyphrase Knowledge Tree* that facilitates user feedback clarification.

We employ a divisive clustering approach to create the knowledge tree. Specifically, with the distributional embeddings for all critiquable keyphrases $\left\{\left(f_\psi^\mu(k_i), f_\psi^\sigma(k_i)\right)|i\in\{1\dots N\}\right\}$ [2], we first initialize one single cluster and select a keyphrase that has the highest distributional similarity with all other keyphrases to represent the main concept of the root cluster. Formally, the index of the topical keyphrase is selected with the following rule:

$$t^{\text{root}} = \underset{i}{\text{argmax}}\sum_{j\neq i}\phi(k_i,k_j) \tag{11}$$

where $\phi(\cdot,\cdot)$ is the Bhattacharyya kernel with f_ψ encoding projection as we described previously.

Next, we apply the elbow method [16] to search the number of sub-clusters based on the root cluster and use k-means [39] to partition the data into sub-clusters. Here, the similarity function used in k-means is also Bhattacharyya kernel $\phi(k_i,k_j)$. We proceed recursively on each cluster with the above-mentioned steps until the cluster has one keyphrase or the maximum hierarchy level is reached.

[2] Up to this point, keyphrase perturbation π is no longer needed (as the model has been trained) so we will use the default keyphrase order in the following description.

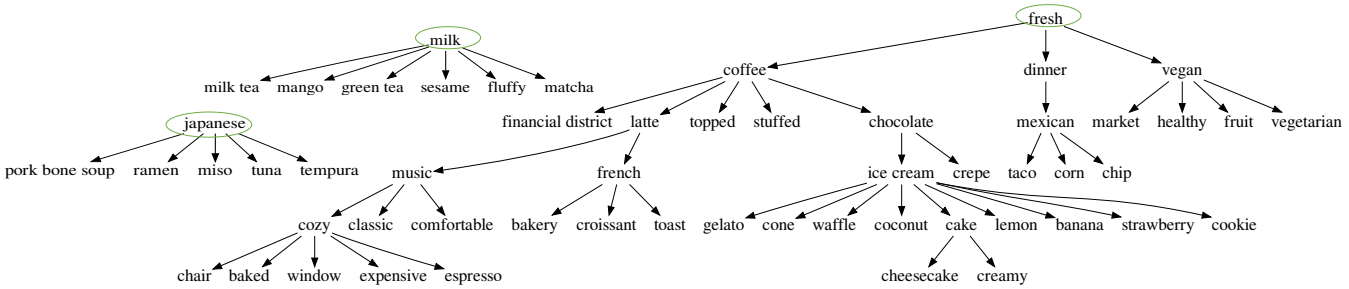


Figure 4: Hierarchical clustering of keyphrases learned through the proposed distributional embedding model. We show only three clusters (among many). It is critical to remark that generality in this tree should be viewed from the perspective of how keyphrases are applied to items in review data (e.g., the "milk" parent node subsumes variations of popular milk tea products).

Since the selected keyphrase shares the most similarity with other keyphrases in the cluster, the ancestor node keyphrase in this knowledge tree often has a more general meaning than its descendant nodes (e.g., "Japanese" should be an ancestor node of "miso"). Indeed, the topical keyphrase of a cluster usually has a more significant variance that allows it to overlap with other keyphrases, which shows the generality of its usage in the domain context. Figure 4 shows a snippet of the hierarchical keyphrase knowledge tree constructed using the algorithm discussed above. In the latter context, we refer to the keyphrase knowledge tree as $\mathbb{T}$.

We remark that the distributional embeddings introduced in this work support various distribution-based distance metrics and hence different clustering approaches to build the knowledge tree. We choose a relatively simple solution in our description to demonstrate the clarification step and leave the exploration of more sophisticated clustering approaches for future work.

3.3 Clarification Design

Now, we introduce the clarification step in the critiquing-based conversational system. A clarification step, by our definition, is an interactive iteration between user and system. For example, given a user's keyphrase input $k \in \{k_1 \cdots k_N\}$ (as part of critiquing conversation), the system suggests a potentially better keyphrase alternative $k' \in \{k_1 \cdots k_N\}$, and, then, the user takes action $a \in \{0, 1\}$ to indicate whether he/she would disagree or agree with the suggestion. If the user agrees with the proposal, the system will use k' as the replacement critique and update the user's representation accordingly for the next round of the critiquing-based conversation.

3.3.1 Clarification Proposal. The keyphrase alternative k' suggested by the system should ideally obey the following conditions:

(1) It should not have been used in the previous critiquing conversation iterations. Redundant critiquing does not help users to obtain better recommendations.
(2) It should be similar to the user's original input k in the context of the recommendation domain.
(3) It should align with the user's preferences. A user is unlikely to critique a keyphrase outside of their historical preferences since a personalized recommender system would not recommend such a product at the beginning of the conversation session.

According to the conditions listed above, we propose to produce a clarification suggestion at time step t in a conversation session, $k'^{(t)}$, as a function f_τ of historical user critiques $\mathbf{k}^s = [k^{(1)} \cdots k^{(t-1)}]$, historical system clarifications $\mathbf{k'}^s = [k'^{(1)} \cdots k'^{(t-1)}]$, user historical clarification feedback $\mathbf{a}^s = [a'^{(1)} \cdots a'^{(t-1)}]$, and user critiquing at same the time step $k^{(t)}$. Formally,

$$k'^{(t)} = f_\tau(\mathbf{k}^s, \mathbf{k'}^s, \mathbf{a}^s, k^{(t)}) = \underset{k_j \in \mathcal{E}^{(t)}}{\operatorname{argmax}}\, \phi(k_j, \mathbf{r}) \tag{12}$$

and the candidate clarification keyphrase set $\mathcal{E}^{(t)}$ is

$$\mathcal{E}^{(t)} = \left\{ k_j \middle| k_j \in \mathcal{G}(k^{(t)}; \mathbb{T}), k_j \notin set(\mathbf{k}^s \otimes (1 - \mathbf{a}^s) + \mathbf{k'}^s \otimes \mathbf{a}^s) \right\}, \tag{13}$$

where $\otimes$ represents the element-wise product and $\mathcal{G}(k^{(t)}; \mathbb{T})$ denotes the neighbor nodes of the keyphrase $k^{(t)}$ in the context of tree knowledge $\mathbb{T}$. The neighbor nodes in this work are directly connected parent and child nodes.

3.3.2 Termination Condition. The constant interruption of clarifications to the user may be disturbing, especially when the user does not require assistance. Hence, automatically ceasing the clarification step in a conversation session based on specific criteria is desired. To this end, we proposed to set a rejection tolerance threshold δ as a condition of halting clarifications. For example, if a user has rejected suggested keyphrase clarifications from the system for more than δ consecutive conversational steps such that

$$\sum_{\tau=t-\delta}^{t} \mathbf{a}^{s(t)} = 0, \tag{14}$$

then the system stops the clarification suggestions since the user either knows about their preference with high confidence or the system fails the clarification task of revealing the true scope of the user's intended preferences.

3.4 User Representation Update

Critiquing-based CRS methods often have different approaches to update the user latent representation after receiving a critique. For example, CE-NCF [37] simply zeroes out the critiquing keyphrase in a binary keyphrase vector and projects it into a user embedding by training an encoder, whereas CE-VAE [23] takes the average of the initial user embedding and the critique embedding produced by the inverse feedback loop as the updated user embedding.

Among various existing approaches, Bayesian uncertainty updating methods with a closed-form update [25, 41] have drawn recent

attention as a more theoretically elegant approach compared to the more heuristic methods of previous work. In this work, we use the Bayesian update rules proposed in BK-VAE [41] as the backbone critiquing mechanism in our experiment.

4 EXPERIMENTS

This section evaluates the proposed system by comparing it to various baseline models on two different benchmark datasets. Specifically, we proceed to evaluate the proposed model to answer the following questions:

- **RQ1:** Is the proposed Distributional Contrastive Embedding (DCE) model competitive in terms of recommendation performance in comparison to state-of-the-art recommenders?
- **RQ2:** Does the keyphrase knowledge tree constructed by our model form concise and coherent hierarchical clusters?
- **RQ3:** Is the proposed DCE model competitive with state-of-the-art critiquing methods for conversational recommendation?
- **RQ4:** Does (personalized) clarification improve conversational efficiency in the critiquing-based conversational system?

All code to reproduce these results is publicly available on Github.[3]

4.1 Experiment Settings

4.1.1 Dataset. We evaluate the performance on two publicly available datasets: The MovieLens10M (MovieLens) for movie recommendations dataset and our own private crawl of the Yelp website for business recommendations. Both datasets have keyphrase description assignments for items provided by the users. In addition, MovieLens contains social tags, typically single words or short phrases assigned by users to movies. For Yelp, we follow the preprocessing steps described in [19]. Only the keyphrases that have been assigned by at least 15 users/items for both datasets are kept.

4.1.2 Baseline. For RQ1, we compare **DCE-VAE** with the following baseline models for the recommendation performance:

- **POP:** Most popular items: not user-personalized but an intuitive baseline to test the claims of this paper.
- **AutoRec [32]:** A neural Autoencoder based recommendation system with one hidden layer and ReLU activation.
- **MF-BPR [31]:** Matrix Factorization with Bayesian Personalized Ranking, which explicitly optimizes pairwise rankings.
- **CD-AE [38]:** Collaborative Denoising Autoencoder that is specifically optimized for implicit feedback recommendation tasks.
- **BK-VAE [41]:** Bayesian Keyphrase critiquing VAE, which uses the off-the-shelf Variational Autoencoder for Collaborative Filtering (VAE-CF) [20] as the backbone of the recommender.

4.2 RQ1: Recommendation Performance

Table 1 shows the pre-critiquing recommendation performance comparison between the proposed model and the various baseline models on the Yelp-Toronto and MovieLens dataset. We note that the DCE-VAE model consistently shows better performance than the other Auto-encoder-based models. We conjecture that the keyphrase component of the DCE-VAE likelihood helps regularize and stabilize the auto-encoder latent representation learning.

[3] https://github.com/TinaBBB/DCE-Clarification-Conversational-Critiquing

4.3 RQ2: Hierarchical Clustering Performance

The Keyphrase Knowledge Tree (KKT) built up from CDE-VAE is essentially a hierarchical clustering, where a sub-tree represents a preference group. Ideally, the keyphrases under a sub-tree should have a more coherent semantic meaning than the keyphrases from different sub-trees. In this experiment, we verify the quality of the learned KKT through a semantic coherence check. Concretely, we use the pre-trained word embeddings from Glove [29] and the distances between those embeddings for single word keyphrases (only) to determine semantic coherence. We can estimate the semantic coherence of the sub-tree (i.e., *intra-cluster distance*) by estimating the average Euclidean distances between Glove embeddings of all keyphrases in the sub-tree and the root of that sub-tree. Similarly, we can also estimate the average Euclidean distance between the Glove embeddings of the sub-tree root and all keyphrases in different sub-trees (i.e., *inter-cluster distance*). If the intra-cluster distance is smaller than the inter-cluster distance, it empirically demonstrates that the sub-tree root node is more semantically coherent with keyphrases in its own sub-tree vs. other sub-trees.

Table 2 shows the intra-cluster and inter-cluster distances for the KKT for both datasets. While the distance gaps between intra-cluster and inter-cluster are not significant, we can still conclude that a sub-tree root node keyphrase is more semantically coherent with its own sub-tree vs. other sub-trees. We remark that the keyphrase knowledge tree is not optimized to maximize semantic coherence; it is merely a favorable side effect of a semantically meaningful distributional embedding.

4.4 RQ3: Critiquing Performance

Now, we progress to evaluate the DCE-VAE in the conversational recommendation setup. Here, we compare DCE-VAE with the state-of-the-art critiquing-based framework BK-VAE in a user simulation setup. In the simulation, we presume users have a desired target item (a held-out test item) and provide critiques w.r.t. that target.

Consider a binary keyphrase-item matrix indicating that the reviews for an item have mentioned the keyphrase at least 5 times. Keyphase vectors (KVs) with dimension #Items correspond to their row in this matrix; item vectors (IVs) with dimension #Keyphrases correspond to their column in this matrix. We can then compute the *average item vector* (AIV) from the top-10 recommended items. We define two user types:

- **Expert:** An expert always knows the best keyphrase to critique during the conversation to quickly retrieve the desired items. In our setup, experts always critique a keyphrase whose index has the largest difference between the target item IV and the AIV.
- **Normal:** A normal user does not always choose the Expert critique, but may instead substitute a similar keyphrase. We simulate the normal user's critiques by sampling keyphrases proportional to the cosine distance between the KV for each keyphrase and the Expert keyphrase KV.

Note that this simulation above does not contain the clarification step; for now we simply aim to compare critiquing performance.

Figure 5 shows our user simulation results averaged over 5000 independent runs. We measure recommendation quality using Hit-Rate@{5,10} that measures how often the target item is ranked

Table 1: Top-N recommendation results of the two datasets. We omit error bars as the 95% confidence interval is in the 4th digit.

Dataset	Model	R-Precision	NDCG	MAP@5	MAP@10	MAP@20	Precision@5	Precision@10	Precision@20	Recall@5	Recall@10	Recall@20
Yelp	POP	0.0335	0.0499	0.0623	0.0551	0.0474	0.0533	0.0451	0.0368	0.0199	0.0327	0.0524
	MF-BPR	0.0401	0.0590	0.071	0.0647	0.0578	0.0631	0.0554	0.0477	0.0214	0.037	0.0627
	AutoRec	0.0333	0.0498	0.0629	0.0555	0.0477	0.0537	0.0452	0.0363	0.0199	0.0328	0.0521
	CD-AE	0.0336	0.0500	0.0627	0.0554	0.0477	0.0538	0.0452	0.0367	0.0199	0.0327	0.0524
	BK-VAE	0.0521	0.0657	0.0640	0.0609	0.0567	0.0615	0.0556	0.0500	0.0247	0.0431	0.0753
	DCE-VAE	**0.0537**	**0.0686**	**0.0711**	**0.0667**	**0.0608**	**0.0664**	**0.0600**	**0.0524**	**0.0261**	**0.0452**	**0.0771**
MovieLens	POP	0.0810	0.1501	0.6138	0.5782	0.5258	0.5830	0.5206	0.4420	0.0273	0.0483	0.0810
	MF-BPR	0.1027	0.1807	0.6480	0.6263	0.5955	0.6264	0.5921	0.5452	0.0301	0.0563	0.1027
	AutoRec	0.0789	0.1459	0.5954	0.5583	0.5091	0.5591	0.5017	0.4296	0.0262	0.0466	0.0789
	CD-AE	0.0798	0.1476	0.6035	0.5652	0.5149	0.5684	0.5056	0.4341	0.0267	0.0469	0.0798
	BK-VAE	0.1057	0.1840	0.6418	0.6181	0.5873	0.6189	0.5822	0.5390	0.0308	0.0577	0.1057
	DCE-VAE	**0.1143**	**0.1982**	**0.6835**	**0.6629**	**0.6340**	**0.6633**	**0.6301**	**0.5863**	**0.0330**	**0.0621**	**0.1139**

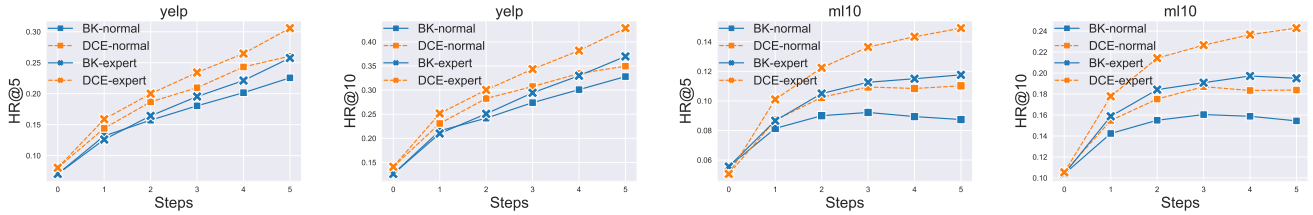


Figure 5: Pure critiquing task: HR@$\{5, 10\}$ comparison during the conversation session between BK-VAE and DCE-VAE.

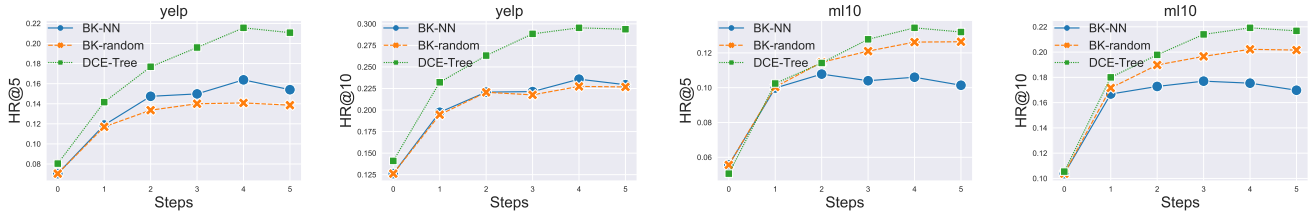


Figure 6: Clarification critiquing task: HR@$\{5, 10\}$ comparison during the conversation session between BK-VAE and DCE-VAE.

Table 2: Intra- and Extra-cluster distance measurement.

Domain	Intra-cluster Distance	Inter-cluster Distance
Yelp-Toronto	5.4471 ± 0.2056	5.7207 ± .1884
MovieLens	5.8283 ± 0.2806	6.1477 ± 0.2664

within the top-k recommendations. Here, for each recommender individually, we observe that the Expert user outperforms the Normal user as expected. But more importantly, we observe for both Expert and Normal users that the DCE-VAE consistently outperforms the BK-VAE over all steps of conversational recommendation. This suggests that DCE-VAE provides better overall critiquing performance through its use of distributional embeddings in comparison to the previous state-of-the-art BK-VAE.

4.5 RQ4: Effectiveness of Clarification

In Sections 3.2 and 3.3, we described how we propose critique clarification keyphrases to the user through a Keyphrase Knowledge Tree (KKT) $\mathbb{T}$. Here, we add a clarification step to each round of critiquing and propose three different methods (two baselines plus one using the KKT) for selecting the system suggested clarification:

- **Random**: As an uninformed baseline, the system ignores the user's originally critiqued keyphrase and proposes a random keyphrase as its clarification suggestion.
- **Nearest Neighbor (NN)**: The system selects a keyphrase whose KV representation is the closest in cosine distance to that of the user's critiqued keyphrase as its clarification suggestion.
- **Tree**: The proposed KKT-based clarification suggestion as defined in Section 3.3.

Here we focus on only Normal users with noisy critiques for whom clarification may help. In simulation, we assume a user accepts a clarification if it is consistent with the KV for the target item. From Figure 6, we observe the following:

(1) DCE-Tree outperforms BK-random and BK-NN consistently for both datasets with a remarkable performance margin.
(2) For the ml10 dataset, the BK-random model has better performance than BK-NN, indicating that many NN clarification proposals may be highly suboptimal.

Overall, we see that the KKT, which captures generality and specificity relationships among keyphrases, is better able to correct mispecifications of the user's critiques (especially leveraging the user's preference distribution as well) in comparison to NN that can only find semantically related keyphrases.

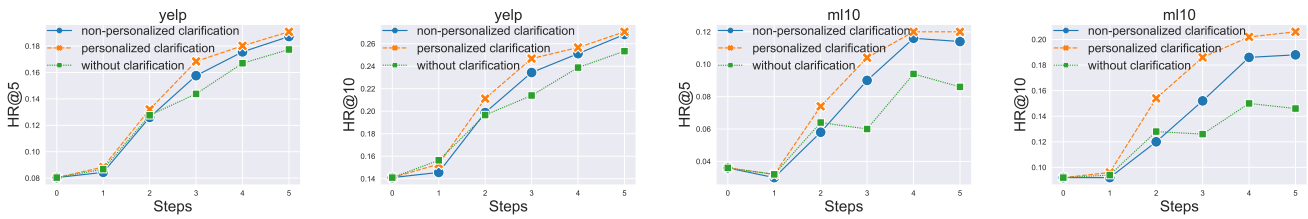


Figure 7: Clarification Performance: HR@$\{5, 10\}$ comparison during the conversation session between pure critiquing using DCE-VAE and clarification-based critiquing.

Table 3: User study examples of the clarification-based critiquing using Yelp and MovieLens datasets. The numbers in parentheses next to items indicate the initial recommendation ranking.

Dataset	Step t	Target Item	Top 3 Recommended Item	Information of Recommendations	Critiqued Keyphrase	Polarity	Clarified Keyphrase	User Acceptance
Yelp	0	Miku	Wooffles & Cream (1) Gyubee Japanese BBQ (2), Good Catch Boil House (3)	bbq, chinese, japanese, oyster, seafood, meat	tempura	preferred	japanese	true
	1		Wooffles & Cream (1), Aoyama Sushi Restaurant (5), SongCook's (6)	japanese, bbq, busy, meat, pork bone soup	expensive	preferred	downtown	false
	2		Miku (64), Ruelo Patisserie (6), Good Catch Boil House (2)	pricey, expensive, japanese, tea, seafood, oyster	-	-	-	-
MovieLens	0	Ghost	Catch Me If You Can (1), American Wedding (2), Shallow Hal (3)	romance, true story, crime, fantasy	love	preferred	chick flick	true
	1		American Wedding (2), Shallow Hal (3), Catch Me If You Can (1)	los angeles, teen, true story, fantasy, crime	new york city	preferred	comedy	false
	2		American Wedding (2), Shallow Hal (3), Serendipity (9)	romance, hilarious, teen	-	-	-	-

Personalization vs Non-personalization: We believe the clarification step should be personalized, which motivated us to introduce the keyphrase suggestion rules in Section 3.3.1. In this experiment, we verify if these personalized clarifications would assist users better than non-personalized clarifications. We remark that the non-personalized keyphrase proposal function is simply

$$k'^{(t)} = f_\tau(\mathbf{k}^s, \mathbf{k'}^s, \mathbf{a}^s, k^{(t)}) = \underset{k_j \in \mathcal{E}^{(t)}}{\operatorname{argmax}}\, \phi(k_j, k^{(t)}), \tag{15}$$

where we replaced the user's rating history $\mathbf{r}$ with the originally critiqued keyphrase $k^{(t)}$ KV (i.e., items mentioning $k^{(t)}$). In addition, we also included a baseline that does not have clarification steps to verify that clarification improves performance.

Figure 7 demonstrates the results of the experiment. Based on the observations, we conclude the following:

(1) Performance of the clarification-enhanced conversation consistently outperforms that of the critiquing-only conversations.
(2) Comparing to a non-personalized approach, performing the critiquing tasks with personalized clarifications gives better recommendation performance over multiple interactions. This observation indicates that the selection of clarification queries should ideally take into account the user's preference history.

4.6 Case Study

To anecdotally inspect the behavior of the clarification-based critiquing conversations, we recorded a conversation session on two datasets, respectively, as shown in Table 3.

For the Yelp dataset, we demonstrate the workflow for a particular sampled user whose targeted item is *Miku*. After receiving the initial recommendations, the user provides a positive preference on *"tempura"* and accepts the system's suggestion of the broader concept of *"Japanese"* as it matches their preference scope. At the second round, The user then rejects the clarification suggestion of *"downtown"* when they express a positive preference on *expensive* restaurants. With these two rounds of clarification-based critiquing, the targeted item becomes the top-recommended item.

Turning the attention to the MovieLens dataset, the target item for the sampled user is *Ghost*, and they accept the clarification keyphrase of *"chick flick"* when they indicate a positive preference towards *"love"*. The system suggestion shows that the proposed model not only simply provides more general concepts as clarification suggestions but also more specific ones by incorporating the user's preference scope (distributional embedding). In the second round, the system relates the user's critique of *"new york city"* with *"comedy"* (many movies set in "new york city" are comedies) and suggests the latter for substitution. Although the target item does not pop into the top-3 list, we can see that the change in top recommendation items is minimal, suggesting that the user's preference scope is generally vague and more interactions are needed to narrow the scope of the user preferences. It is important to remark that data quality and sparsity issues that occur with human review data may also be limiting performance in this case.

5 CONCLUSION

We presented a novel methodology for clarification in critiquing-based conversational recommender systems. To support clarification query selection, we proposed a Distributional Contrastive Embedding variant of the Variational Auto-encoder-based recommendation model, called DCE-VAE. DCE-VAE learns to distributionally co-embed keyphrase and user preference representations that permits reasoning about generality and specificity of keyphrase critiques used for the clarification step as well as beliefs in the user's preferences. For conversational recommendations, DCE-VAE can automatically construct a hierarchical Keyphrase Knowledge Tree to guide the clarification query selection process. DCE-VAE not only outperformed baseline models for critiquing-only tasks, but also for clarification-based critiquing tasks — however, these results were only in simulation and further user study validation of these techniques is important future work. Together, our contributions provide a novel workflow and methodology for clarification-based critiquing that we hope provides inspiration for future work to improve user interactions with conversational recommender systems.

REFERENCES

[1] Diego Antognini and Boi Faltings. 2021. Fast Multi-Step Critiquing for VAE-Based Recommender Systems *(RecSys '21)*. Association for Computing Machinery, New York, NY, USA, 209–219. https://doi.org/10.1145/3460231.3474249

[2] Diego Antognini and Boi Faltings. 2021. Fast Multi-Step Critiquing for VAE-based Recommender Systems. *arXiv preprint arXiv:2105.00774* (2021).

[3] Ben Athiwaratkun and Andrew Gordon Wilson. 2017. Multimodal word distributions. *arXiv preprint arXiv:1704.08424* (2017).

[4] Krisztian Balog, Filip Radlinski, and Alexandros Karatzoglou. 2021. On Interpretation and Measurement of Soft Attributes for Recommendation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 890–899.

[5] Robin D Burke, Kristian J Hammond, and Benjamin C Young. 1996. Knowledge-based navigation of complex information spaces. In *Proceedings of the national conference on artificial intelligence*, Vol. 462. 468.

[6] Li Chen and Pearl Pu. 2012. Critiquing-based recommenders: survey and emerging trends. *User Modeling and User-Adapted Interaction* 22, 1 (2012), 125–150.

[7] Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. 2021. Advances and challenges in conversational recommender systems: A survey. *AI Open* 2 (2021), 100–126. https://doi.org/10.1016/j.aiopen.2021.06.002

[8] Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 297–304.

[9] Tatsunori B Hashimoto, David Alvarez-Melis, and Tommi S Jaakkola. 2016. Word embeddings as metric recovery in semantic spaces. *Transactions of the Association for Computational Linguistics* 4 (2016), 273–286.

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

[11] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. *Iclr* 2, 5 (2017), 6.

[12] Andrea Iovine, Fedelucio Narducci, and Giovanni Semeraro. 2020. Conversational Recommender Systems and natural language: A study through the ConveRSE framework. *Decision Support Systems* 131 (2020), 113250. https://doi.org/10.1016/j.dss.2020.113250

[13] Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. 2021. A Survey on Conversational Recommender Systems. *ACM Computing Surveys (CSUR)* 54, 5 (2021), 1–36.

[14] Tony Jebara, Risi Kondor, and Andrew Howard. 2004. Probability product kernels. *The Journal of Machine Learning Research* 5 (2004), 819–844.

[15] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. 2018. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*. PMLR, 2668–2677.

[16] Trupti M Kodinariya and Prashant R Makwana. 2013. Review on determining number of Cluster in K-Means Clustering. *International Journal* 1, 6 (2013), 90–95.

[17] Wenqiang Lei, Xiangnan He, Yisong Miao, Qingyun Wu, Richang Hong, Min-Yen Kan, and Tat-Seng Chua. 2020. Estimation-action-reflection: Towards deep interaction between conversational and recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 304–312.

[18] Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems* 27 (2014), 2177–2185.

[19] Hanze Li, Scott Sanner, Kai Luo, and Ga Wu. 2020. A Ranking Optimization Approach to Latent Linear Critiquing in Conversational Recommender System. In *ACM RecSys-20*. Online.

[20] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. *arXiv preprint arXiv:1802.05814* (2018).

[21] Annie Louis, Dan Roth, and Filip Radlinski. 2020. "I'd rather just go to bed": Understanding Indirect Answers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, 7411–7425.

[22] Kai Luo, Scott Sanner, Ga Wu, Hanze Li, and Hojin Yang. 2020. Latent linear critiquing for conversational recommender systems. In *Proceedings of The Web Conference 2020*. 2535–2541.

[23] Kai Luo, Hojin Yang, Ga Wu, and Scott Sanner. 2020. Deep Critiquing for VAE-Based Recommender Systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1269–1278.

[24] Shengnan Lyu, Arpit Rana, Scott Sanner, and Mohamed Reda Bouadjenek. 2021. A Workflow Analysis of Context-driven Conversational Recommendation. In *Proceedings of the Web Conference 2021*. 866–877.

[25] Francesca Mangili, Denis Broggini, Alessandro Antonucci, Marco Alberti, and Lorenzo Cimasoni. 2020. A Bayesian approach to conversational recommendation systems. *arXiv preprint arXiv:2002.05063* (2020).

[26] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[27] Kajsa Møllersen, Subhra S Dhar, and Fred Godtliebsen. 2016. On Data-Independent Properties for Density-Based Dissimilarity Measures in Hybrid Clustering. *arXiv preprint arXiv:1609.06533* (2016).

[28] Ewa Nowakowska, Jacek Koronacki, and Stan Lipovetsky. 2014. Tractable measure of component overlap for gaussian mixture models. *arXiv preprint arXiv:1407.7172* (2014).

[29] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[30] Filip Radlinski, Krisztian Balog, Bill Byrne, and Karthik Krishnamoorthi. 2019. Coached conversational preference elicitation: A case study in understanding movie preferences. (2019).

[31] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[32] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 111–112.

[33] Anna Sepliarskaia, Julia Kiseleva, Filip Radlinski, and Maarten de Rijke. 2018. Preference elicitation as an optimization problem. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 172–180.

[34] B. Smyth, L. McGinty, J. Reilly, and K. McCarthy. 2004. Compound Critiques for Conversational Recommender Systems. In *IEEE/WIC/ACM International Conference on Web Intelligence (WI'04)*. 145–151. https://doi.org/10.1109/WI.2004.10098

[35] Flemming Topsoe. 2000. Some inequalities for information divergence and related measures of discrimination. *IEEE Transactions on information theory* 46, 4 (2000), 1602–1609.

[36] Luke Vilnis and Andrew McCallum. 2014. Word representations via gaussian embedding. *arXiv preprint arXiv:1412.6623* (2014).

[37] Ga Wu, Kai Luo, Scott Sanner, and Harold Soh. 2019. Deep Language-based Critiquing for Recommender Systems. In *Proceedings of the 13th International ACM Conference on Recommender Systems (RecSys-19)*. Copenhagen, Denmark.

[38] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 153–162.

[39] Rui Xu and Donald Wunsch. 2005. Survey of clustering algorithms. *IEEE Transactions on neural networks* 16, 3 (2005), 645–678.

[40] Hojin Yang, Scott Sanner, Ga Wu, and Jin Peng Zhou. 2021. Bayesian Preference Elicitation with Keyphrase-Item Coembeddings for Interactive Recommendation. In *Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization*. 55–64.

[41] Hojin Yang, Tianshu Shen, and Scott Sanner. 2021. Bayesian Critiquing with Keyphrase Activation Vectors for VAE-based Recommender Systems. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2111–2115.

Appendix

A HYPER-PARAMETERS AND DATASET DETAILS

Table 4: Hyper-parameters tuned on the experiments.

name	Range	Functionality	Algorithms affected
h	{50, 100, 150, 200}	Latent Dimension	AutoRec, BPR, CD-AE BK-VAE, DCE-VAE
α	{1e-4, 5e-4, 1e-3, 5e-3}	Learning Rate	AutoRec,CD-AE BK-VAE, DCE-VAE
λ	{1e-5, 5e-5 $\cdots$ 1e4}	L2 Regularization	AutoRec, BPR CD-AE, BK-VAE, DCE-VAE
λ_1	{0.001, 0.01, $\cdots$ 10}	Regularization for User-keyphrase NCE loss	DCE-VAE
β	{1e-4, 1e-3 $\cdots$ 1}	KL Regularization	BK-VAE, DCE-VAE
δ	{0.1, 0.2 $\cdots$ 1}	Corruption Rate	CD-AE, BK-VAE, DCE-VAE
ι	{1,2,3,4,5}	Negative Samples	BPR
ς	{0.1, 0.2, $\cdots$ 1}	Weighted RMSE Regularization	BK-VAE, DCE-VAE

Table 5: Summary of datasets.

Dataset	# Users	# Items	# Keyphrases	# Ratings	Sparsity
MovieLens	8,000	10,677	164	4,777,024	94.41%
Yelp	7,000	4,997	245	203,683	99.42%

B DISCUSSION OF RELAXATION FOR THE DERIVATION OF CONTRASTIVE LOSS

As mentioned in the work, there are many way of relaxing the the conditional likelihood $p(\mathbf{k}|\mathbf{r})$

$$p(\mathbf{k}|\mathbf{r}) = p(k_1|\mathbf{r})p(k_2|k_1)p(k_3|k_2,k_1)\cdots p(k_N|k_{N-1}\cdots k_1). \quad (16)$$

In this section, we provide an alternative relaxation that can reach the same final objective described in the main paper. By assuming probability of observing a keyphrase only depends on the previous observed keyphrases as Markov chain, $p(k_i|k_{i-1}\cdots k_1) = p(k_i|k_{i-1})$, we can obtain the following relaxed expression

$$p(\mathbf{k}|\mathbf{r}) = p(k_1|\mathbf{r})p(k_2|k_1)p(k_3|k_2)\cdots p(k_N|k_{N-1}). \quad (17)$$

Since we run the above relaxation multiple times based on different permutations $\pi \in \Gamma$ of keyphrase list $\mathbf{k}$, the overall objective still converge back to the final objective that we described in Figure 2 and Section 3.1.

C PROBABILISTIC DISTANCES

In this section, we introduce alternative probabilistic distance variants in addition to the Bhattacharyya kernel used for the training model. These distance variants measure the distance between two normal distribution $p = \mathcal{N}(\mu_1, \sigma_1^2)$ and $q = \mathcal{N}(\mu_2, \sigma_2^2)$. The distance functions are non-negative, and some of these become zero if two distributions are identical. The extensions for multivariate Gaussian distributions with diagonal variances can be derived by summing over the per-dimension distance.

Table 6: Best hyper-parameter setting for each algorithm.

Domain	Algorithm	h	α	λ	λ_1	β	Iteration*	δ	ι	ς
Yelp-Toronto	BPR	50	-	1e-4	-	-	30	-	1	-
	CD-AE	50	1e-4	1e-4	-	-	300	0.5	-	-
	BK-VAE	100	1e-2	1e-5	-	0.3	470	0.2	-	0.5
	AutoRec	200	1e-4	1e-5	-	-	300	0	-	-
	DCE-VAE	150	1e-2	0	0.01	0.3	250	0.2	-	0.5
MovieLens	BPR	200	-	1e-4	-	-	30	-	1	-
	CD-AE	50	1e-4	0.1	-	-	300	0	-	-
	BK-VAE	150	1e-4	1e-4	-	0.3	250	0	-	1
	AutoRec	50	1e-4	0.1	-	-	300	0	-	-
	DCE-VAE	150	1e-4	1e-4	0.1	0.3	210	0	-	1

Kullback-Leibler (KL) divergence is an asymmetric measure for the difference between two distributions:

$$\begin{aligned} KL(p,q) &= \int \log \frac{p}{q} dp \\ &= \frac{1}{2}[\log \frac{\sigma_2^2}{\sigma_1^2} + \frac{\sigma_1^2}{\sigma_2^2} + \frac{(\mu_1-\mu_2)^2}{\sigma_2^2}]. \end{aligned} \quad (18)$$

KL divergence does not satisfy the triangular inequality, and is asymmetric (e.g., $KL(p,q) \neq KL(q,p)$). The KL divergence will explode if q has a very small variance. Although existing work used KL divergence as a distance metric between Gaussian distributions [3, 36], the training objective is different from ours where asymmetric relationships between words is unknown.

Jensen-Shannon (JS) divergence is the average of the forward KL and reversed KL divergences. The JS divergence has a division term by variances $\sigma_1\sigma_2$; it can be numerically unstable when the variances are very small.

Probability product kernels[14] are the generalized inner product for two distributions:

$$PPK(p,q) = \int_z p(z)^\rho q(z)^\rho dz, \quad (19)$$

When $\rho = 1$, it is called the Expected Likelihood Kernel (ELK), and when $\rho = \frac{1}{2}$, it is the Bhattacharyya's affinity [27], or the Bhattacharyya kernel being used in this work.

Expected likelihood kernel is a special case of PPK when $\rho = 1$ in Equation 19. The log probability is computed for ELK in practice as the following:

$$ELK(p,q) = \frac{1}{2}[\frac{(\mu_1-\mu_2)^2}{\sigma_1^2+\sigma_2^2} + \log(\sigma_1^2+\sigma_2^2)]. \quad (20)$$

This was not used in the paper as the kernel method since ELK would assign higher rewards to the multivariate Gaussian embedding with smaller variances.

Wasserstein distance is a metric function of two distributions on a given metric space M. Here we note the 2-Wasserstein distance:

$$W(p,q)^2 = (\mu_1-\mu_2)^2 + \sigma_1 - \sigma_2^2. \quad (21)$$

Bhattacharyya Kernel While using the dot product between the means of two Gaussian distributions is a perfectly valid measure of similarity, it does not incorporate any uncertainty or the covariances and would not provide any benefit from our probabilistic model. The most suitable next choice for a similarity measure between two Gaussian distributions would be taking the inner product between the distributional embeddings themselves. We refer to

such similarity measure as kernel (ϕ) measures between our embedded user and keyphrase representations from the probabilistic model.

For a positive-definite reproducing kernel ϕ: $(\mathbb{R}^d, \mathbb{R}^d) \times (\mathbb{R}^d, \mathbb{R}^d) \rightarrow \mathbb{R}$, the following expression represents the formulation of the kernel measurement $\phi(P_i, P_j)$ used by our model between two Gaussian distributions P_i and P_j:

$$\phi(P_i, P_j) = \int_{x \in \mathbb{R}^d} \mathcal{N}(x; \mu_i, \Sigma_i)^{\frac{1}{2}} \mathcal{N}(x; \mu_j, \Sigma_j)^{\frac{1}{2}} dx \tag{22}$$

The above kernel formulation is called the **Bhattacharyya kernel** [14], because it is known as the Bhattacharyya's measure of affinity between distributions, related to the better known Hellinger's distance.

$$H(P, P') = \frac{1}{2} \int (\sqrt{P(x)} - \sqrt{P'(x)})^2 dx \tag{23}$$

Note that the Hellinger distance can be seen as a principled symmetric approximation of the KL divergence and is a bound on KL as shown in [35]. In the maximum likelihood setting where $\mu = \bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$, the probability product kernel would reduce to the traditional Radial Basis Function (RBF) kernel:

$$\phi(x, x') = \frac{1}{4\pi\sigma^2} exp(-||\bar{x} - \bar{x'}||^2 / 4\sigma^2) \tag{24}$$

Using the Bhattacharyya kernel is mentioned in [28] as a method to calculate distributions overlap in a statistical manner.