Deep Critiquing for VAE-based Recommender Systems

Kai Luo
University of Toronto
kluo@mie.utoronto.ca

Hojin Yang
University of Toronto
hojin@mie.utoronto.ca

Ga Wu*
University of Toronto
wuga@mie.utoronto.ca

Scott Sanner
University of Toronto
ssanner@mie.utoronto.ca

ABSTRACT

Providing explanations for recommended items not only allows users to understand the reason for receiving recommendations but also provides users with an opportunity to refine recommendations by critiquing undesired parts of the explanation. While much research focuses on improving the explanation of recommendations, less effort has focused on interactive recommendation by allowing a user to critique explanations. Aside from traditional constraint- and utility-based critiquing systems, the only end-to-end deep learning based critiquing approach in the literature so far, CE-VNCF, suffers from unstable and inefficient training performance. In this paper, we propose a Variational Autoencoder (VAE) based critiquing system to mitigate these issues and improve overall performance. The proposed model generates keyphrase-based explanations of recommendations and allows users to critique the generated explanations to refine their personalized recommendations. Our experiments show promising results: (1) The proposed model is competitive in terms of general performance in comparison to state-of-the-art recommenders, despite having an augmented loss function to support explanation and critiquing. (2) The proposed model can generate high-quality explanations compared to user or item keyphrase popularity baselines. (3) The proposed model is more effective in refining recommendations based on critiquing than CE-VNCF, where the rank of critiquing-affected items drops while general recommendation performance remains stable. In summary, this paper presents a significantly improved method for multi-step deep critiquing based recommender systems based on the VAE framework.

KEYWORDS

Deep Learning; Critiquing

ACM Reference Format:
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 (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397271.3401091

*Ga Wu contributed the probabilistic graphical model interpretation and mathematical derivation for the proposed model.

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401091

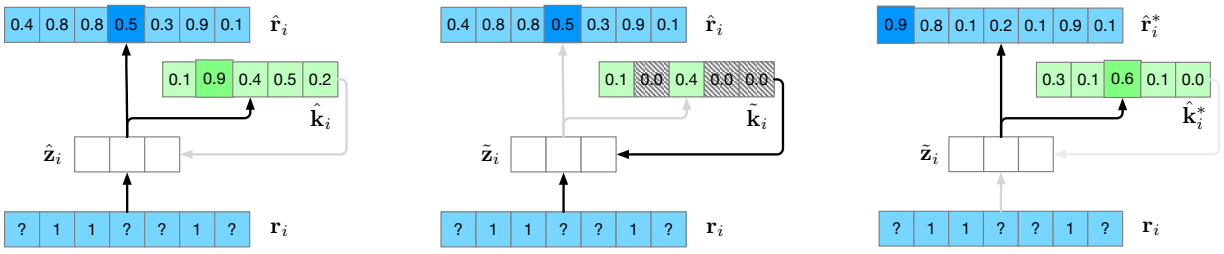
1 INTRODUCTION

Presenting short explanations along with recommended items is well-known to provide helpful context for recommendations. To this end, recent years have seen a variety of research focusing on explainable recommendations [22]; to name just a few of these works: [10] introduces topic extraction methods to explain and highlight key aspects of recommended items, [23] improves the understandability of explanations byfi lling key features of recommended items into template sentences, and [1, 6] both directly generate text explanations for recommendations using Recurrent Neural Networks.

Beyond explanations, many efforts have attempted to refine recommendations by allowing users to directly interact with recommended items or item attributes. Incremental critiquing [12, 13] was proposed to improve recommendation quality over successive recommendation cycles by taking the degree of compatibility between recommendation candidates and users' past critiques into consideration. Unfortunately, this interactive approach to critiquing recommendations based on explicit features has a major drawback: directfi ltering-based interaction on items or explicit item properties is not highly effective for domains with expressive feature sets like movies or books since critiques may not easily generalize to other items with correlated hidden properties. Rather than limiting interaction to a small set offi xed properties, it would be more flexible if users were permitted to critique using arbitrary language.

After a decade where critiquing approaches received little attention, recent work in the form of Deep Language-based Critiquing (DLC) [19] introduced a model known as CE-VNCF that provides explanations with recommendations and accepts arbitrary language-based critiques to improve these recommendations. As we show in this paper, CE-VNCF suffers from unstable training and high computational complexity since it trains with negative sampling and must loop over all items during inference. This drawback not only significantly reduces its scalability but also increases the training difficulty, which limits its use on large-scale recommendation tasks.

In this paper, we revisit deep critiquing from the perspective of Variational Autoencoder based recommendation methods and keyphrase-based interaction. Instead of basing our model on the Neural Collaborative Filtering (NCF) [4] architecture used in CE-VNCF, we leverage a more efficient Variational Autoencoder (VAE) architecture with a novel variational lower bound to support effective joint training of the entire framework. Compared to CE-VNCF, which must inefficiently perform a forward pass for all items when



(a) Prediction (b) Critiquing (c) Re-prediction

Figure 1: Step-by-stepfl ow of the Deep Critiquing with Variational Autoencoder (VAE) architecture. The light blue vector shows the sparse implicit feedback vector (bottom) and its dense VAE reconstruction (top) used for recommendation prediction; dark blue indicates the top-ranked item (not already observed as a positive interaction) to be recommended. The green vector is the keyphrase prediction; dark green indicates the best single top-ranked keyphrase to describe the user's preferences. (a) An item is recommended for user i along with a keyphrase description. (b) User i critiques keyphrases shown as gray zeroed-out entries that modify the latent embedding and (c) generates a revised recommendation and keyphrase explanation.

recommending for a user, our VAE-based model enables both item recommendation and explanation in a single forward pass. Moreover, the critiquing process becomes much simpler by modulating the user representation independently of items, resulting in the computational complexity reduction. In summary, the proposed VAE model shows: (1) better computational efficiency at both training and inference time, (2) improved stability and faster convergence, and (3) overall superior or competitive performance in comparison to state-of-the-art recommendation and critiquing methods.

2 PRELIMINARIES

Before proceeding, we define notation used throughout this paper:

- $\mathbf{r}_i$: A vector with length n (total number of items). This is the implicit feedback vector for user i; vector entries of 1 (0) denote a positive (negative or unobserved) interaction for user i with the item given by the vector index. We assume there are m users in the training data such that $i \in \{1 \cdots m\}$.
- $\mathbf{z}_i$: A vector with length d. This is user i's latent representation (user embedding) vector.
- $\mathbf{k}_i$: A vector with length s. This is the keyphrase vector that reflects user i's keyphrase usage preference.
- $\hat{\mathbf{r}}_i$, $\hat{\mathbf{k}}_i$, $\hat{\mathbf{z}}_i$: The **predicted** feedback, keyphrase, and latent representation for user i.
- $\mathbf{c}_i$: A one-hot vector with keyphrase length s whose sole positive value position indicates the index of the keyphrase to be critiqued by user i at a given step of user interaction with the recommendation system.
- $\tilde{\mathbf{z}}_i$, $\tilde{\mathbf{k}}_i$: The modified latent representation and keyphrases for user i resulting from their critiquing action $\mathbf{c}_i$.

2.1 Variational Autoencoder for Recommendation

The Variational Autoencoder (VAE) is a deep generative model used in this paper to explicitly optimize a variational lower bound on the log likelihood of all observed user feedback $\mathbf{r}_i$ in the form of $\sum_i \log p(\mathbf{r}_i)$. Here, the VAE uses an Autoencoder architecture (cf. Figure 1) to predict $\hat{\mathbf{r}}_i$ from an embedded distribution $\hat{\mathbf{z}}_i$ representing user i's preferences; the latent $\mathbf{z}_i$ is regularized through a KL divergence with a standard Normal distribution. Concretely, the VAE model optimizes the following objective:

$$\begin{aligned}\sum_i \log p(\mathbf{r}_i) &\geq \sum_i \int_{\mathbf{z}_i} q_\vartheta(\mathbf{z}_i|\mathbf{r}_i) \log \frac{p_\theta(\mathbf{r}_i|\mathbf{z}_i)p(\mathbf{z}_i)}{q_\vartheta(\mathbf{z}_i|\mathbf{r}_i)} d\mathbf{z} \\ &= \sum_i \left[E_{q_\vartheta(\mathbf{z}_i|\mathbf{r}_i)}[\log p_\theta(\mathbf{r}_i|\mathbf{z}_i)] - KL[q_\vartheta(\mathbf{z}_i|\mathbf{r}_i)||p(\mathbf{z}_i)] \right],\end{aligned} \tag{1}$$

where ϑ and θ are the parameters for the encoder and decoder networks, respectively. In practice, the proposed conditional encoder distribution $q_\vartheta(\mathbf{z}_i|\mathbf{r}_i)$ is usually a Gaussian distribution whose parameters ($\boldsymbol{\mu}_i$ and Σ_i) are predicted from the raw observation $\mathbf{r}_i$ via the encoder neural network.

The impressive generalization and reconstruction ability of the VAE model is particularly attractive to the recommendation community and has inspired many recent deep learning-based recommendation models [7, 8, 18]. While the proposed models differ in various ways, the basic model structure remains unchanged, and the common conclusion is that VAE models generally show better performance than their Autoencoder (AE) counterparts.

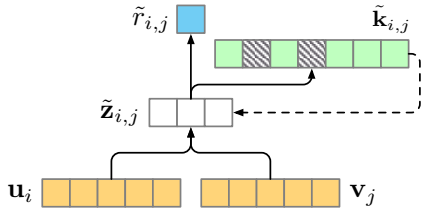
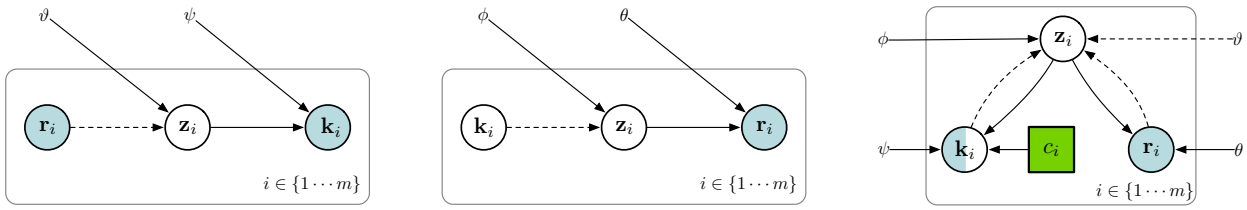


Figure 2: The previously proposed CE-VNCF recommendation architecture for Deep Language-based Critiquing [19]. This framework is based on the Neural Collaborative Filtering (NCF) architecture [4] but has an additional prediction head for producing keyphrase explanations for the recommendation. Critiquing functionality is achieved by an additional encoding network (dashed line) that encodes the critiqued keyphrases back into the latent representation.



(a) $\prod_i p(\mathbf{k}_i|\mathbf{r}_i)$ **(b)** $\prod_i p(\mathbf{r}_i)$ **(c)** $\prod_i p(\mathbf{r}_i, \mathbf{k}_i)$

Figure 3: Probabilistic Graphical Model view of the proposed model. The keyphrase $\mathbf{k}_i$ and the implicit feedback $\mathbf{r}_i$ are both generated from user i's latent representation $\mathbf{z}_i$. $\mathbf{c}_i$ is a critiquing action initiated by user i in the recommendation session.

2.2 Deep Language-based Critiquing Framework

The original Deep Language-based Critiquing framework (DLC) [19] is an end-to-end deep learning based interactive recommender system that incorporates explanation and critiquing into the Neural Collaborative Filtering (NCF) architecture as shown in Figure 2. The DLC framework is trained to maximize the conditional probability of binary interaction $r_{i,j}$ and corresponding explanation $\mathbf{k}_{i,j}$ (a binary vector for keyphrases) for *each* user i and item j given their respective embeddings as well as the critiqued keyphrases $\tilde{\mathbf{k}}_{i,j}$. The objective function for the DLC framework maximizes the joint likelihood of observed $p(r_{i,j}, \mathbf{k}_{i,j}|\mathbf{u}_i, \mathbf{v}_j, \tilde{\mathbf{k}}_{i,j})$.

The DLC framework can produce an explanation for each of the produced recommendations with a list of keyphrases. Furthermore, it accepts the interactive critiquing of users on the explained keyphrases to refine its next recommendation.

While DLC made seminal contributions to critiquing, the DLC framework shows several noticeable defects that detract from its practical usage:

- It is computationally expensive at both training and inference time due to its use of the NCF architecture for recommendation. For the training phase, the NCF model requires optimization of a triplet loss that involves negative sampling for each of the observed interactions at each training epoch. For the inference phase, in order to obtain the ranking score of items for a user, the DLC framework must loop over all items to forward propagate their score before ranking.
- The performance of the trained model heavily relies on negative sampling, which is a stochastic process that can lead to training instability as we later observe empirically.
- It can perform poorly on sequential critiquing tasks due to its "Zero-out" critiquing approach (formally defined later) that gradually attenuates the amount of information carried in the critiquing loop as we later demonstrate.
- The framework does not train with interactive critiquing. Instead, it relaxes the critiqued keyphrases with the keyphrase predictions from NCF, which can lead to a poor approximation of the intended training objective.

3 THE CRITIQUING FOR VAE BASED RECOMMENDER

This paper replaces the NCF backbone of the DLC framework with the VAE to address various drawbacks that are observed previously. In order to incorporate explanation and critiquing functionality into the VAE, we extend it with additional network components that achieve a bidirectional mapping between the user latent representation $\mathbf{z}_i$ and keyphrase preference explanation $\mathbf{k}_i$ as shown in Figure 1.

3.1 Inference Logic

Overall, we aim for the iterative recommendation and critiquing informationfl ow outlined in Figure 1. Given sparse historical interactions $\mathbf{r}_i$ of user i, the extended modelfi rst predicts the possible future interactions $\hat{\mathbf{r}}_i$ and the keyphrase interpretation of user preference $\hat{\mathbf{k}}_i$ in Figure 1(a) through

$$\hat{\mathbf{r}}_i = f_\theta(\hat{\mathbf{z}}_i) \quad \hat{\mathbf{k}}_i = f_\psi(\hat{\mathbf{z}}_i), \tag{2}$$

where

$$\hat{\mathbf{z}}_i = f_\vartheta(\mathbf{r}_i), \tag{3}$$

and f_θ, f_ψ are decoder networks for interaction and keyphrase predictions respectively, and f_ϑ is the encoder network from historical interaction.

When a user "disagrees" with the recommender's keyphrase-based explanations, they can "disable" some of the keyphrases to express their critique as shown in Figure 1(b) though the function

$$\tilde{\mathbf{k}}_i = f_\eta(\hat{\mathbf{k}}_i, \mathbf{c}_i), \tag{4}$$

where we later define specific interpretations for $\mathbf{c}_i$ in Section 3.3, including the Zero-out method used in previous work and a new proposal for Energy Redistribution we make in this work.

The model then updates the latent representation $\mathbf{z}_i$ by blending latent embedded information from the user feedback $\tilde{\mathbf{k}}_i$ and embeddings of historical interactions $\mathbf{r}_i$ in Figure 1(b) with

$$\tilde{\mathbf{z}}_i = f_\zeta(f_\phi(\tilde{\mathbf{k}}_i), f_\vartheta(\mathbf{r}_i)), \tag{5}$$

where f_ζ is the blending function, and f_ϕ represents the encoding network from the feedback $\tilde{\mathbf{k}}_i$.

Finally, the model produces new recommendations and explanations in Figure 1(c) for possible future iterations of critiquing.

3.2 Training Strategy

A straightforward strategy of learning the proposed model is that wefi rst train the original VAE model and then train the additional bidirectional mapping component in a subsequent step. However, this approach has two major defects: (1) The user latent representation learned through reconstructing its input observations may not

support keyphrase prediction and results in poor explanation and critiquing performance. (2) The explanation and critiquing components lack a probabilistic interpretation that is inherently necessary for proper variational training. Thus, we propose to train the model with a joint variational objective over all components.

In order to work around the potential issue of the simple two-step training schema, we propose to model the entire network architecture with a generative probabilistic graphical model. Concretely, we assume the observed interactions and keyphrase interpretations are all generated from a latent embedding representation of user preferences.

The derivation starts with the joint log likelihood $\sum_i^m \log p(\mathbf{r}_i, \mathbf{k}_i)$ over observed $\mathbf{r}_i$ and $\mathbf{k}_i$ and all users i as shown in Figure 3(c). In the following, we omit the user subscript i and the summation over users to reduce notational clutter. The joint likelihood could be factorized into two factors:

$$\log p(\mathbf{r}, \mathbf{k}) = \log p(\mathbf{k}|\mathbf{r}) + \log p(\mathbf{r}) \tag{6}$$

For the conditional likelihood $\log p(\mathbf{k}|\mathbf{r})$ as shown in Figure 3(c), the derivation is relatively straightfoward as it was in the Conditional Variational Autoencoder [17]. Concretely, we have:

$$\log p(\mathbf{k}|\mathbf{r}) \geq \underbrace{E_{q_\phi(\mathbf{z}|\mathbf{r})}[\log p_\psi(\mathbf{k}|\mathbf{z})]}_{\text{① Keyphrase Prediction}} - \underbrace{KL[q_\phi(\mathbf{z}|\mathbf{r})||p(\mathbf{z})]}_{\text{② Latent Regularization}}, \tag{7}$$

where we relaxed the $p(\mathbf{z}|\mathbf{r})$ with its prior $p(\mathbf{z})$. In practice, we weight the KL term with a hyperparameter β as motivated in the Beta-VAE [5]. The conditional likelihood eventually forms an Encoder-Decoder architecture, and the inputs and outputs are interactions and keyphrases respectively.

Comparing to the conditional likelihood $p(\mathbf{k}|\mathbf{r})$, the likelihood of historical interactions has a more difficult derivation. While one can adapt the Variational Autoencoder derivation, it does not support our end purpose of incorporating critiquing feedback. Instead, we provide an alternative derivation that fulfills our purpose, as shown in Figure 3(b). We start with a variational lower-bound of $\log p(\mathbf{r})$, where we decompose the usual KL in the second and third terms:

$$\begin{aligned}&\log p(\mathbf{r})\\ &\geq \underbrace{E_{q_\vartheta(\mathbf{z}|\mathbf{r})}[\log p_\theta(\mathbf{r}|\mathbf{z})]}_{\text{③ Interaction Prediction}} + \underbrace{H[q_\vartheta(\mathbf{z}|\mathbf{r})]}_{\text{④ Entropy}} + E_{q\vartheta(\mathbf{z}|\mathbf{r})}[\log p(\mathbf{z})].\end{aligned} \tag{8}$$

It would be easy to treat the prior $p(\mathbf{z})$ as a standard Normal distribution as we already did in the conditional probability term. However, to more tightly link in keyphrases, we approximate its variational lower bound by marginalizing over a reintroduced $\mathbf{k}$:

$$\begin{aligned}\log p(\mathbf{z}) &\geq E_{q(\mathbf{k}|\mathbf{z})}[\log p(\mathbf{z}|\mathbf{k})] - KL[q(\mathbf{k}|\mathbf{z})||p(\mathbf{k})]\\ &\approx \underbrace{E_{p_\psi(\mathbf{k}|\mathbf{z})}[\log p_\phi(\mathbf{z}|\mathbf{k})]}_{\text{⑤ Autoencoder}} - \underbrace{KL[p_\psi(\mathbf{k}|\mathbf{z})||p(\mathbf{k})]}_{\text{⑥ Keyphrase Regularization}},\end{aligned} \tag{9}$$

where we assume the prior distribution of the keyphrases is a standard Normal distribution, and, most importantly, we allow the inference probability $q(\mathbf{k}|\mathbf{z})$ to share its weights with the generative probability $p(\mathbf{k}|\mathbf{z})$ in Equation 7 to reduce the number of parameters to learn. This decomposition integrates learning of $\mathbf{r}$, $\mathbf{k}$, and $\mathbf{z}$ and elegantly facilitates critiquing by providing $p_\phi(\mathbf{z}|\mathbf{k})$ to infer the user's latent representation from their keyphrase preferences.

Combining Equations (6)–(9), we have our full objective function with six sub-objectives that are indexed with circled numbers in the equations. The joint objective function not only maximizes the observation likelihood but also maximizes a latent representation likelihood with two KL terms respectively regularizing the latent representation and keyphrase predictions.

It is crucial to notice that the training progress does not contain the critiquing step in the training loop. We will leverage the generalization ability of the two-way mapping between the latent and keyphrase space to support the critiquing loop.

3.3 Critiquing Function Design

For an interactive recommender system, we would prefer the user feedback interface to be as simple as possible. Thus, in our critiquable recommendation setting, we let the critiquing action simply indicate which keyphrase to "disable". Specifically, the critiquing action $\mathbf{c}_i$ in this paper is a one-hot vector with keyphrase length s whose positive value position indicates the index of the keyphrase to be critiqued.

A naive critiquing function f_η could simply be

$$\tilde{\mathbf{k}}_i = \hat{\mathbf{k}}_i \odot (1 - \mathbf{c}_i) + \mathbf{c}_i \min(\hat{\mathbf{k}}_i), \tag{10}$$

with $\odot$ indicating an elementwise product, which **Zeroes-out** the critiqued keyphrase score and keeps the rest of the vector the same. As a slight adjustment, adding back the minimum value of the critiqued keyphrase score was found to work more robustly in practice. This critiquing function is proposed in the DLC framework [19] and shown to be effective on one-step critiquing tasks.

However, we note that in the case of sequential critiquing for conversational recommendation, this simple solution does not preserve the "energy" of the system during the critiquing feedback loop and values of predictions can quickly diminish in several critiquing loops as we will show in Figure 7 in the experimental results.

In order to mitigate this vanishing energy issue in conversational recommendation, we propose an alternative critiquing function that we call **Energy Redistribution**. Concretely, the Energy Redistribution function is defined as follows:

$$\tilde{\mathbf{k}}_i = \frac{||\hat{\mathbf{k}}_i||_1}{||\hat{\mathbf{k}}_i \odot (1 - \mathbf{c}_i)||_1}(\hat{\mathbf{k}}_i \odot (1 - \mathbf{c}_i)). \tag{11}$$

This Energy Redistribution function not only zeroes-out the critiqued keyphrase but also rescales the critiqued keyphrase vector to have the same L_1 norm (i.e., "energy") before and after critiquing:

$$||\hat{\mathbf{k}}_i||_1 = ||\tilde{\mathbf{k}}_i||_1 \tag{12}$$

3.4 Blending Function Design

Based on our proposed inference logic, the critiqued keyphrase vector can be re-embedded into the latent user preference representation through an encoder network f_ϕ. But now we have the original embedding and the critiqued embedding and we need to blend them together as shown in Equation 5. While there are many options to implement the blending function[1], we take a relatively

[1]Concurrently published work on Latent Linear Critiquing (LLC) [9] implements the blending function as a linear programming task that looks for a convex combination of embeddings provided with a specific linear optimization objective.

Table 1: Summary of datasets. We selected 40 keyphrases for CDs&Vinyl and 75 keyphrases for BeerAdvocate. Coverage shows the percentage of reviews/comments that have at least one selected keyphrase.

Dataset	# Users	# Items	# Reviews	Sparsity	Keyphrase Coverage	Keyphrase Average Counts (per User)
CDs&Vinyl (Amazon)	6,056	4,395	152,670	0.5736%	75.48%	13.9969
Beer (BeerAdvocate)	6,370	3,668	263,278	1.1268%	99.29%	55.1088

simple but empirically robust approach based on the proposed critiquing architecture. Thus, in this paper, we choose to balance the combination of the critique and original embeddings:

$$\tilde{\mathbf{z}}_i = \frac{1}{2}(f_\phi(\tilde{\mathbf{k}}_i) + f_\vartheta(\mathbf{r}_i)) \tag{13}$$

We assume multiple critiques are additively accumulated in $\tilde{\mathbf{k}}_i$.

3.5 Latent Representation Sampling vs. Expectation

During the training phase, we sample the latent representation from a Normal distribution $q(\mathbf{z}_i|\mathbf{r}_i)$

$$\hat{\mathbf{z}}_i \sim \mathcal{N}(f_\vartheta^\mu(\mathbf{r}_i), f_\vartheta^\sigma(\mathbf{r}_i)), \tag{14}$$

as a standard way to maximize the variational lower-bound of the conditional likelihood $\prod_i p(\mathbf{k}_i|\mathbf{r}_i)$ by gradient descent.

However, during the inference (recommendation) phase, we omit this sampling step and instead directly use the expectation (mean) of the Normal distribution as the most likely latent representation

$$\hat{\mathbf{z}}_i = f_\vartheta^\mu(\mathbf{r}_i) \tag{15}$$

to reduce noise and avoid unpredictable behavior due to stochasticity during the critiquing process.

4 EXPERIMENTS

Now we proceed to evaluate the proposed Critiquable and Explainable VAE model (CE-VAE) in order to answer the following questions:

- Is the Variational Autoencoder (VAE) a better foundational recommender model than Neural Collaborative Filtering (NCF) in terms of recommendation performance, explanation performance, and critiquing performance?
- Does a VAE outperform a standard deterministic Autoencoder? What role does KL divergence play in terms of balancing critiquing and recommendation performance?
- Is Energy Redistribution a better critiquing approach than Zero-out and how do the methods compare at different iterations of multi-step critiquing?

All code to reproduce these results is publicly available on Github.[2]

[2] https://github.com/k9luo/DeepCritiquingForVAEBasedRecSys.

4.1 Experiment Settings

4.1.1 Dataset. We evaluate performance on two publicly available datasets: BeerAdvocate [10] and Amazon CDs&Vinyl [3, 11]. Both of the datasets have more than 100,000 reviews and product rating records. In order to simulate the One-class Collaborative Filtering environment commonly found in practical applications (e.g., only purchases are observed), we binarize the rating observations for both datasets with a rating threshold. For comparison, we follow the data processing steps in the DLC framework [19] using the codebase referenced in that paper. Table 1 shows overall dataset statistics. All experiments use 50% train, 20% validation, and 30% test data splits.

4.1.2 Baseline Models. We compare our proposed Critiquable Explainable VAE (**CE-VAE**) model to the following baseline models:

- **POP:** Most popular items – not user personalized but an intuitive baseline to test the claims of this paper.
- **AutoRec [16]:** A neural Autoencoder based recommendation system with one hidden layer and ReLU activation.
- **BPR [14]:** Bayesian Personalized Ranking, which explicitly optimizes pairwise rankings.
- **CDAE [21]:** Collaborative Denoising Autoencoder that is specifically optimized for implicit feedback recommendation tasks.
- **CE-VNCF [19]:** Critiquable and Explainable Variational Neural Collaborative Filtering – Neural Collaborative Filtering based deep critiquing model.
- **NCE-PLRec [20]:** Noise Contrastive Estimation Projected Linear Recommendation. This model augments PLRec with noise contrasted item embeddings.
- **PLRec [15]:** Also called Linear-Flow. This is the baseline projected linear recommendation approach. This is one ablation of NCE-PLRec.
- **PureSVD [2]:** A similarity based recommendation method that constructs a similarity matrix through SVD decomposition of the implicit rating matrix.
- **VAE-CF [8]:** Variational Autoencoder for Collaborative Filtering – a state-of-the-art deep learning based recommender system.

Table 2 presents our hyperparameter definitions and sweeps for architecture and algorithm tuning on the held-out validation set. The best hyperparameter settings found for each algorithm and domain are listed in Table 3.

In the case of the proposed CE-VAE, λ_2, λ_3 shared the same value during tuning while the corresponding λ term on rating prediction wasfi xed to 1 during training and prediction.

4.1.3 Critiquing Performance Metric: Falling Map [19]. Given a set of items $\mathbb{S} = \{Item_j \mid j \in \{1 \cdots n\}\}$, and a critiquable keyphrase k, if k is in the Top-K ground truth keyphrases of item j, we say the item j belongs to the item set $\mathbb{S}_k$. Ideally, after the user's i critique on k, we would want the rank of any affected items $\mathbb{S}_k$ to "fall" (move further down the ranked list) from the Top-N item recommendation list for user i after critiquing.

Using $\mathbb{S}_k$ as a surrogate to label ground truth "relevance" in a standard Mean Average Precision (MAP) metric, Falling MAP (F-MAP) measures the ranking difference of the affected items set $\mathbb{S}_k$

Table 2: Hyper-parameters tuned on the experiments.

name	Range	Functionality	Algorithms affected
r	{50, 100, 200}	Latent Dimension	PLRec, PureSVD AutoRec, CE-VAE NCE-PLRec, BPR CE-VNCF, CDAE VAE-CF
α	{1e-4, 5e-4, 1e-3, 5e-3}	Learning Rate	AutoRec,CDAE VAE-CF, CE-VAE CE-VNCF
λ_1	{1e-5, 5e-5 $\cdots$ 1e4}	L2 Regularization	PLRec, AutoRec BPR, NCE-PLRec CDAE, VAE-CF CE-VAE, CE-VNCF
λ_2	{1e-4, 1e-3 $\cdots$ 1}	Keyphrase Regularization	CE-VAE
λ_3	{1e-4, 1e-3 $\cdots$ 1}	Latent Regularization	CE-VAE
β	{1e-4, 1e-3 $\cdots$ 1}	KL Regularization	CE-VAE
ρ	{0.1, 0.2 $\cdots$ 1}	Corruption Rate	CDAE, VAE-CF CE-VAE, CE-VNCF
γ	{0.7, 0.8 $\cdots$ 1.3}	Popularity Sensitivity	NCE-PLRec
η	{1,2,3,4,5}	Negative Samples	CE-VNCF, BPR

Table 3: Best hyper-parameter setting for each algorithm.

Domain	Algorithm	r	α	λ_1	λ_2	λ_3	β	Iteration*	ρ	γ	η
Beer	PLRec	200	-	1e4	-	-	-	10	-	-	-
	BPR	50	-	1e-5	-	-	-	30	-	-	1
	NCE-PLRec	50	-	1e4	-	-	-	10	-	1.3	-
	CE-VNCF	200	1e-3	5e-5	1	1	0.01	300	0.1	-	5
	CE-VAE	100	1e-4	1e-4	0.01	0.01	1e-3	300	0.5	-	-
	PureSVD	100	-	-	-	-	-	10	-	-	-
	CDAE	100	1e-4	1e-5	-	-	-	300	0.4	-	-
	VAE-CF	50	1e-4	1e-4	-	-	0.2	300	0.5	-	-
	AutoRec	100	1e-4	1e-5	-	-	-	300	0	-	-
CDs&Vinyl	PLRec	200	-	1e4	-	-	-	10	-	-	-
	BPR	200	-	1e-4	-	-	-	30	-	-	1
	NCE-PLRec	200	-	1e3	-	-	-	10	-	1.3	-
	CE-VNCF	200	1e-4	1e-4	1	1	0.01	300	0.1	-	5
	CE-VAE	200	1e-4	1e-4	1e-3	1e-3	1e-4	600	0.5	-	-
	PureSVD	200	-	-	-	-	-	10	-	-	-
	CDAE	200	1e-4	1e-5	-	-	-	300	0.2	-	-
	VAE-CF	200	1e-4	1e-4	-	-	0.2	300	0.3	-	-
	AutoRec	200	1e-4	1e-5	-	-	-	300	0	-	-

* For PureSVD, PLRec and NCE-PLRec, iterations in this table means number of randomized SVD iterations. For AutoRec, BPR, CDAE, CE-VAE, CE-VNCF and VAE-CF, iteration shows number of epochs that processed over all users.

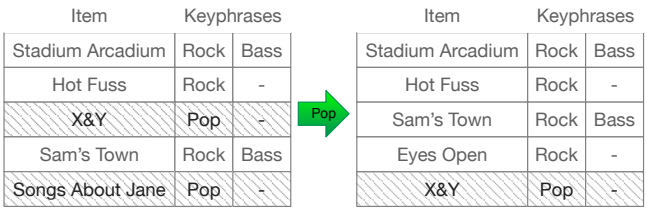


Figure 4: Example of the Falling MAP metric. Dashed entries are items that are affected by critiquing. If only two items are affected by the keyphrase critique "pop", then before critiquing, the MAP is 0.37, and after critiquing, the MAP is 0.2. As a result, the F-MAP value is 0.17. Since the F-MAP value is positive, we say the critique has been effective.

before and after critiquing keyphrase k. Specifically,

$$F\text{-}MAP(i,k,N) = MAP@N^{\text{before}}_{\mathbb{S}_k}(i) - MAP@N^{\text{after}}_{\mathbb{S}_k}(i), \qquad (16)$$

where N is the number of items to be recommended and $\mathbb{S}_k$ is constant that observed from the training data. We would expect the rank of items in $\mathbb{S}_k$ to fall after the keyphrase k is critiqued, indicated by a positive F-MAP. In our experiments, we average $F\text{-}MAP(i,k,N)$ over 7,500 random selected user and keyphrase pairs. Figure 4 shows a simple demo of F-MAP@5 for a single user.

4.2 General Recommendation Performance

Table 4 and 5 show the Top-N recommendation performance comparison between the proposed model and various baselines on the CDs&Vinyl and Beer datasets. From the tables, we obtain the following interesting observations:

- Comparing to CE-VNCF model in the DLC framework, our proposed model shows better recommendation performance for almost all metrics on the two datasets.
- Comparing to the original VAE recommender, CE-VAE performs better on the Beer dataset, but slightly worse on the CDs&Vinyl dataset. We conjecture that additional loss terms for keyphrase critiquing have little adverse effect on recommendation quality, and could even be helpful to generate a better user representation by leveraging the keyphrase history of the user.
- For the Beer dataset, CE-VAE outperforms all baseline models in terms of Precision and MAP.

We remark that our hyper-parameter tuning range is larger than that reported in the DLC paper [19]. This change of experimental setting leads to relatively negligible differences in the reported values in the tables.

4.3 Keyphrase Explanation Performance

Tables 6 and 7 show the keyphrase prediction performance comparison between the proposed model with the baseline models. Here, each model is used to rank predicted keyphrases for a user and item and standard ranking metrics are used to evaluate explanation quality against the ground truth keyphrases the user used in their review of the item.

In this experiment, the UserPop and the ItemPop are the popularity based models that leverage basic keyphrase statistics:

- **UserPop** predicts the explanation through counting and ranking the frequency of keyphrases for the users in the training dataset.
- **ItemPop** predicts the explanation through counting and ranking the frequency of keyphrases for the items in the training dataset. All users share the same predictions in this case.

Both of the deep learning-based critiquing models show better explanations than the baseline models with a significant performance gap. Remarkably, the proposed CE-VAE model shows a considerable advantage on NDCG, MAP, and Precision over the CE-VNCF model on both of the datasets. As a trade-off, in terms of Recall, the proposed model is slightly weaker.

4.4 Training Time Comparison

Table 8 shows the training time comparison between the proposed CE-VAE model and the CE-VNCF model on the two datasets. During

Table 4: Top-N recommendation results of CDs&Vinyl dataset. We omit the error bars as the confidence interval is in 4th digit.

Model	R-Precision	NDCG	MAP@5	MAP@10	MAP@20	Precision@5	Precision@10	Precision@20	Recall@5	Recall@10	Recall@20
POP	0.0078	0.0277	0.0096	0.0094	0.0088	0.0099	0.0087	0.0079	0.0088	0.0164	0.0317
AutoRec	0.0130	0.0387	0.0166	0.0154	0.0137	0.0152	0.0133	0.0113	0.0144	0.0252	0.0430
BPR	0.0621	0.1527	0.0719	0.0625	0.0524	0.0612	0.0489	0.0384	0.0751	0.1160	0.1755
CDAE	0.0090	0.0313	0.0115	0.0113	0.0105	0.0115	0.0107	0.0094	0.0108	0.0206	0.0365
CE-VNCF	0.0659	0.1639	0.0725	0.0647	0.0552	0.0640	0.0530	0.0415	0.0811	0.1313	0.1996
NCE-PLRec	0.0745	0.1667	**0.0844**	**0.0740**	**0.0617**	**0.0732**	**0.0583**	0.0436	0.0898	0.1398	0.2015
PLRec	0.0732	0.1621	0.0839	0.0734	0.0606	0.0727	0.0574	0.0424	0.0892	0.1354	0.1931
PureSVD	0.0681	0.1511	0.078	0.0678	0.0559	0.0671	0.0523	0.0389	0.0846	0.1280	0.1821
VAE-CF	**0.0760**	**0.1791**	0.0823	0.0721	0.0606	0.0708	0.0571	**0.0442**	**0.0934**	**0.1450**	**0.2144**
CE-VAE	0.0723	0.1621	0.0822	0.0717	0.0593	0.0710	0.0555	0.0415	0.0890	0.1348	0.1951

Table 5: Top-N recommendation results of Beer review dataset. We omit the error bars as the confidence interval is in 4th digit.

Model	R-Precision	NDCG	MAP@5	MAP@10	MAP@20	Precision@5	Precision@10	Precision@20	Recall@5	Recall@10	Recall@20
POP	0.0022	0.0060	0.0027	0.0026	0.0028	0.0025	0.0024	0.0031	0.0009	0.0021	0.0070
AutoRec	0.0415	0.0983	0.0544	0.0500	0.0450	0.0489	0.0437	0.0374	0.0336	0.0577	0.0958
BPR	0.0349	0.0849	0.0421	0.0390	0.0359	0.0379	0.0350	0.0314	0.0258	0.0472	0.0839
CDAE	0.0369	0.0886	0.0462	0.0427	0.0387	0.0424	0.0377	0.0328	0.0284	0.0491	0.0858
CE-VNCF	0.0441	0.1084	0.0538	0.0507	0.0467	0.0501	0.0464	0.0402	0.0358	0.0651	0.1108
NCE-PLRec	0.0505	0.1141	0.0636	0.0585	0.0522	0.0577	0.0511	0.0428	0.0412	0.0701	0.1152
PLRec	0.0455	0.1067	0.0598	0.0545	0.0487	0.0537	0.0468	0.0399	0.0387	0.0646	0.1089
PureSVD	0.0355	0.0812	0.0453	0.0410	0.0365	0.0404	0.0346	0.0303	0.0279	0.0465	0.0788
VAE-CF	**0.0515**	0.11980	0.0600	0.0556	0.0503	0.0553	0.0488	0.0423	**0.0437**	0.0742	**0.1236**
CE-VAE	0.0509	**0.11982**	**0.0657**	**0.0602**	**0.0538**	**0.0593**	**0.0523**	**0.0442**	0.0436	**0.0745**	0.1223

Table 6: Explanation Quality of CDs&Vinyl review dataset. We omit the error bars since the confidence interval is in 4th digit.

Model	NDCG@5	NDCG@10	NDCG@20	MAP@5	MAP@10	MAP@20	Precision@5	Precision@10	Precision@20	Recall@5	Recall@10	Recall@20
UserPop	0.1110	0.1339	0.1956	0.0841	0.0724	0.0615	0.0777	0.0543	0.0519	0.1347	0.1890	0.3843
ItemPop	0.1319	0.1566	0.2191	0.0933	0.0810	0.0679	0.0888	0.0604	0.0552	0.1640	0.2229	0.4214
CE-VNCF	0.4886	0.5583	0.6086	0.3419	0.2700	0.2004	0.2526	0.1714	0.1067	**0.5523**	**0.7153**	0.8646
CE-VAE	**0.4950**	**0.6137**	**0.7143**	**0.5799**	**0.5050**	**0.4152**	**0.4979**	**0.3922**	**0.2804**	0.4600	0.6630	**0.8708**

Table 7: Explanation Quality of Beer review dataset. We omit the error bars since the confidence interval is in 4th digit.

Model	NDCG@5	NDCG@10	NDCG@20	MAP@5	MAP@10	MAP@20	Precision@5	Precision@10	Precision@20	Recall@5	Recall@10	Recall@20
UserPop	0.0366	0.0610	0.1102	0.0497	0.0505	0.0546	0.0490	0.0546	0.0652	0.0316	0.0718	0.1767
ItemPop	0.0491	0.0768	0.1298	0.0669	0.0644	0.0650	0.0586	0.0624	0.0717	0.0404	0.0859	0.1997
CE-VNCF	**0.3163**	**0.4229**	0.5099	0.4089	0.3695	0.3113	0.3701	0.3075	0.2182	**0.2765**	**0.4549**	**0.6379**
CE-VAE	0.2688	0.3987	**0.5778**	**0.9015**	**0.8845**	**0.8564**	**0.8886**	**0.8556**	**0.8006**	0.1406	0.2651	0.4791

Table 8: Training times in seconds of the proposed CE-VAE model and CE-VNCF on Beer and CDs&Vinyl datasets using single Nvidia GeForce GTX 1080 Ti.

Dataset	CE-VAE	CE-VNCF
CDs&Vinyl (Amazon)	125.2802	7087.0978
Beer (BeerAdvocate)	66.7969	11752.6808

this experiment, we denote the convergence time for both candidate models as the wall-clock time required to reach the highest NDCG score on the validation set. We remark that the number of training epochs do not provide a clear indication of wall-clock time. Specifically, on the Beer dataset, both candidate algorithms converged to optimality within the same number of epochs (300 epochs). Moreover, on the CDs&Vinyl dataset, the optimal number of epochs for the CE-VNCF model is 300, but the proposed CE-VAE model requires 600 epochs to converge.

Overall, our proposed CE-VAE model is two orders of magnitude faster than the CE-VNCF model, which shows the significant advantage of removing negative sampling in the training loop. This substantial performance difference makes CE-VAE much more scalable to large datasets than CE-VNCF.

Table 9: User Case Study on the Beer and CDs&Vinyl review datasets.

Dataset	User ID	Initial Top Explanations (Explanations Listed in Order)	Top Items Recommended (Items Listed in Order)	Critiqued Keyphrase	Refined Top Explanations (Explanations Listed in Order)	Refined Top Items Recommended (Items Listed in Order)
Beer	1765	Fruit, Brown, Caramel	60 Minute IPA HopDevil Ale Hop Rod Rye Trappistes Rochefort 8 Sierra Nevada Bigfoot Barleywine Style Ale	Brown	Fruit, Sweet, Caramel	60 Minute IPA Sierra Nevada Bigfoot Barleywine Style Ale Trappistes Rochefort 8 HopDevil Ale Double Bastard Ale
CDs&Vinyl	4946	Rock, Pop, Bass	Stadium Arcadium Hot Fuss X&Y Sam's Town Songs About Jane	Pop	Rock, Pop, Metal	Stadium Arcadium Hot Fuss Sam's Town Eyes Open X&Y

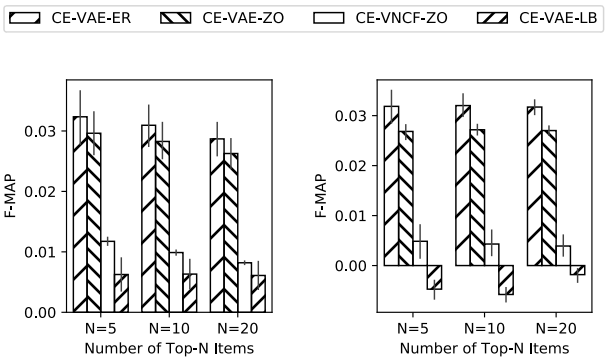


Figure 5: Falling MAP versus Critiquing Models (higher is better). Error bars show 95% CI. Thefi rst plot is on Beer and the second plot is on CDs&Vinyl.

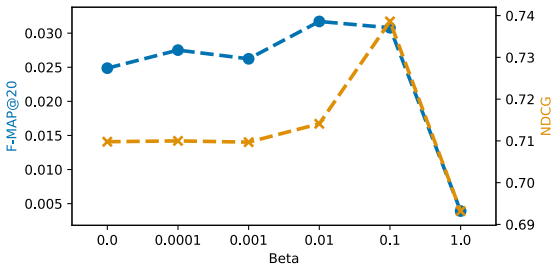


Figure 6: This plot analyzes how different beta values affect Falling MAP and keyphrase prediction NDCG.

4.5 Critiquing Performance

Figure 5 shows the critiquing performance comparison between the proposed CE-VAE model with the CE-VNCF model. For both of the datasets, the proposed CE-VAE shows significantly better critiquing performance than the CE-VNCF model. In terms of the critiquing function, we also observe a considerable performance improvement by leveraging the Energy Redistribution (ER) approach over a simple Zero-out (ZO) approach.

In addition to the previously noted experimental baselines, we also included a lower bound critiquing model (CE-VAE-LB) for comparison purposes. The lower bound model injects random noise into the latent representation $\tilde{\mathbf{z}}_i$ instead of incorporating $f_\phi(\mathbf{k}_i)$ during critiquing inference. Hence, a model that does a poor job of incorporating critiquing feedback should be close to the random CE-VAE-LB; unfortunately we see that while CE-VNCF-ZO does perform better than CE-VAE-LB, it only does so by a relatively small margin. The relatively large improvements of both CE-VAE-ER and CE-VAE-ZO over the lower bound CE-VAE-LB demonstrate the considerable advance in critiquing performance made by this work.

Table 10: Ground Truth Keyphrases of Items used in User Case Study.

Dataset	Items	Items' Ground Truth Keyphrases (Order does not matter)
Beer	60 Minute IPA	Smooth, Fruit, Citrus
	HopDevil Ale	Caramel, Fruit, Bitter, Brown, Citrus
	Hop Rod Rye	Citrus, Caramel, Brown
	Trappistes Rochefort 8	Brown, Caramel, Fruit, Sweet, Smooth
	Sierra Nevada Bigfoot Barleywine Style Ale	Golden, Sweet, Fruit, Caramel
	Double Bastard Ale	Caramel, Sweet, Fruit
CDs&Vinyl	Stadium Arcadium	Rock, Bass
	Hot Fuss	Rock
	X&Y	Pop
	Sam's Town	Rock, Bass
	Songs About Jane	Pop
	Eyes Open	Rock

4.6 Tuning β for the KL Divergence

In practice, it is critical to tune the β hyperparameter of the KL divergence term on $p(\mathbf{z})$ as mentioned in the discussion of Equation 7. Figure 6 shows an evaluation on the CDs&Vinyl dataset. This plot indicates that tuning the KL divergence hyperparameter β is critical for improving the effectiveness of the critiquing and keyphrase prediction, and it may involve a trade-off between these two objectives in case the best values do not agree with each other.

4.7 Sequential Critiquing Function Analysis

In the methodology description, we hypothesized that the Energy Redistribution critiquing function would help multi-step critiques. In this experiment, we empirically evaluate our hypothesis by running 4-step critiques over 2,000 random selected users and 8,000 random keyphrases. As shown in Figure 7, the Energy Redistribution approach preserves the average score of keyphrase prediction over multi-step critiquing (as mathematically enforced), whereas the Zero-out approach shows a gradual decline in overall scores indicating a "loss of energy" in the critiquing feedback loop that can degrade performance over multiple critiquing steps. Combined with the observation that the Energy Redistribution (ER) methods

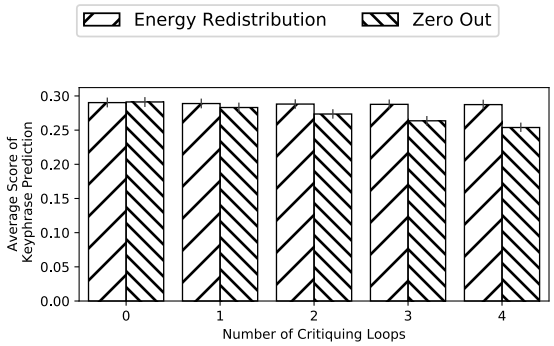


Figure 7: Average score of keyphrase prediction vs. number of critiquing loops. Compared to the Zero-out (ZO) approach, the Energy Redistribution (ER) based critiquing function is more stable over multiple steps of critiquing. Error bars show 95% CI. Step 0 (before critiquing when the ER and ZO methods are trivially equal) is intended for comparison to the subsequent critiquing iterations 1-4.

outperformed the Zero-out (ZO) methods in the previous experimental results, these observations help confirm our intuition and motivation for introducing Energy Redistribution in the multi-step critiquing setting.

4.8 Case Study

To qualitatively evaluate the performance of the critiquing in a real environment, we simulated multiple use-cases of the proposed model on the two review datasets. Table 9 shows two representative examples we encountered during our investigation. Table 10 demonstrates ground truth keyphrases of items that were used in the case study.

For the Beer dataset, based on historical interactions of user 1765, we see the model described the user preferences as keyphrase explanations: *Fruit*, *Brown*, *Caramel*. Top-5 recommended items were *60 Minute IPA*, *HopDevil Ale*, *Hop Rod Rye*, *Trappistes Rochefort 8* and *Sierra Nevada Bigfoot Barleywine Style Ale*. From Table 10, we know that those items are relevant to at least one of the predicted user's preferences. This shows that predicted user's preferences could be correlated with the predicted items' properties. We then chose to critique the keyphrase *Brown*, resulting in refined user preferences to be *Fruit*, *Sweet*, and *Caramel*. Refined item recommendations also reflect user's preferences shift. Based on Table 10, items *HopDevil Ale*, *Hop Rod Rye* and *Trappistes Rochefort 8* are associated with *Brown*. After critiquing, rank of *HopDevil Ale* dropped from 2 to 4. *Hop Rod Rye* was no longer in Top-5 recommended items after critiquing. Rank of *Trappistes Rochefort 8* increased from 4 to 3 because it highly matched with the user' current preferences. *Double Bastard Ale* was not ranked in Top-5 before critiquing and it was ranked 5 after critiquing. *Sierra Nevada Bigfoot Barleywine Style Ale*'s rank increased from 5 to 2. From Table 9 and 10, we know that both of *Double Bastard Ale* and *Sierra Nevada Bigfoot Barleywine Style Ale* contain *fruit* and *caramel* that are also in the user's initial top explanations. We can safely conclude that *Sierra Nevada Bigfoot Barleywine Style Ale* is more aligned with the user's long term preferences since it was ranked higher than *Double Bastard Ale* in the initial top recommended items ranking list as shown in Table 9. Given the fact that both items contain all of the user's refined top explanations, we know that both items highly match with the user's current preferences. Leveraging the user's long term preferences and current preferences, our proposed model ranks *Double Bastard Ale* lower than *Sierra Nevada Bigfoot Barleywine Style Ale* after critiquing as shown in Table 9.

Turning our attention to the CDs&Vinyl dataset, we see that the model predicted user 4946 liked *Rock*, *Pop* and *Bass* music based on the user's historical interactions. The Top-5 initial recommended items also reflect the user's predicted preferences. It is likely that the user was fascinated about *Pop* songs based on their historical data. This could be why *Pop* remains in the user's Top-3 predicted preferences after it was critiqued. However, it is clear that the rank of songs tjat are described by *Pop* either dropped out of Top-5 recommended items (*Songs About Jane*) or were pushed down in the ranked list (*X&Y*) after critiquing. The model recommended more rock music that matched with the user's preferences after critiquing.

As onefi nal remark on the combined case studies of Beer and CDs&Vinyl, we observe that the post-critique recommendations did not change drastically but rather demonstrate a conservative shuffling of rank order and the introduction of one new item in each case. Since these results are obtained from the tuned CE-VAE architecture that performed best overall, this indicates that personalization is important (i.e., the initial ranking) and that a critiquing method that has a nuanced impact while strongly preserving the initial personalization can yield highly competitive performance.

5 CONCLUSION

In this paper, we proposed a novel deep learning-based critiquable recommender system based on the Variational Autoencoder (VAE). Compared to thefi rst deep critiquing model CE-VNCF [19] in the literature, our proposed model shows substantially better recommendation, explanation, training time, and overall critiquing performance according to a variety of metrics. We also proposed a novel Energy Redistribution method for incorporating critiquing feedback that further allows the deep critiquing model to support multi-step critiquing and avoid degradation of the critiquing signal. Together, these contributions in our novel critiquable and explainable CE-VAE framework significantly improve the performance of the deep critiquing framework and thus increase the practicality of applying such deep critiquing models in large-scale interactive recommendation applications.

REFERENCES

[1] Felipe Costa, Sixun Ouyang, Peter Dolog, and Aonghus Lawlor. 2018. Automatic generation of natural language explanations. In *Proceedings of the 23rd International Conference on Intelligent User Interfaces Companion*. ACM, 57.

[2] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*. 39–46.

[3] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborativefi ltering. In *proceedings of the 25th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 507–517.

[4] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborativefi ltering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[5] 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.

[6] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. 2017. Neural rating regression with abstractive tips generation for recommendation. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 345–354.

[7] Xiaopeng Li and James She. 2017. Collaborative variational autoencoder for recommender systems. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 305–314.

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

[9] Kai Luo, Scott Sanner, Ga Wu, Hanze Li, and Hojin Yang. 2020. Latent Linear Critiquing for Conversational Recommender Systems. In *Proceedings of the 28th international conference on the topic of the World Wide Web (WWW-20)*. Taipei.

[10] Julian McAuley, Jure Leskovec, and Dan Jurafsky. 2012. Learning attitudes and attributes from multi-aspect reviews. In *2012 IEEE 12th International Conference on Data Mining*. IEEE, 1020–1025.

[11] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 43–52.

[12] Kevin McCarthy, Yasser Salem, and Barry Smyth. 2010. Experience-based critiquing: Reusing critiquing experiences to improve conversational recommendation. In *International Conference on Case-Based Reasoning*. Springer, 480–494.

[13] James Reilly, Kevin McCarthy, Lorraine McGinty, and Barry Smyth. 2004. Incremental critiquing. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*. Springer, 101–114.

[14] 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.

[15] Suvash Sedhain, Hung Bui, Jaya Kawale, Nikos Vlassis, Branislav Kveton, Aditya Krishna Menon, Trung Bui, and Scott Sanner. 2016. Practical linear models for large-scale one-class collaborativefi ltering. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. AAAI Press, 3854–3860.

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

[17] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*. 3483–3491.

[18] Ga Wu, Mohamed Reda Bouadjenek, and Scott Sanner. 2019. One-Class Collaborative Filtering with the Queryable Variational Autoencoder. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR-19)*. Paris, France.

[19] 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.

[20] Ga Wu, Maksims Volkovs, Chee Loong Soon, Scott Sanner, and Himanshu Rai. 2019. Noise Contrastive Estimation for Scalable Linear Models for One-Class Collaborative Filtering. *The 42th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[21] 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.

[22] Yongfeng Zhang and Xu Chen. 2018. Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192* (2018).

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