

# RecVAE: a New Variational Autoencoder for Top-N Recommendations with Implicit Feedback

Ilya Shenbin  
ilya.shenbin@gmail.com  
Samsung-PDMI Joint AI Center,  
Steklov Institute of Mathematics at  
St. Petersburg, Russia

Anton Alekseev  
anton.m.alexeyev@gmail.com  
Samsung-PDMI Joint AI Center,  
Steklov Institute of Mathematics at  
St. Petersburg, Russia

Elena Tutubalina  
tutubalinaev@gmail.com  
Samsung-PDMI Joint AI Center,  
Steklov Institute of Mathematics at  
St. Petersburg, Russia

Valentin Malykh  
valentin.malykh@phystech.edu  
Neural Systems and Deep Learning  
Laboratory, Moscow Institute of  
Physics and Technology, Russia

Sergey I. Nikolenko  
sergey@logic.pdmi.ras.ru  
Samsung-PDMI Joint AI Center,  
Steklov Institute of Mathematics at  
St. Petersburg, Russia

## ABSTRACT

Recent research has shown the advantages of using autoencoders based on deep neural networks for collaborative filtering. In particular, the recently proposed *Mult*-VAE model, which used the multinomial likelihood variational autoencoders, has shown excellent results for top-N recommendations. In this work, we propose the *Recommender VAE* (RecVAE) model that originates from our research on regularization techniques for variational autoencoders. *RecVAE* introduces several novel ideas to improve *Mult*-VAE, including a novel composite prior distribution for the latent codes, a new approach to setting the  $\beta$  hyperparameter for the  $\beta$ -VAE framework, and a new approach to training based on alternating updates. In experimental evaluation, we show that *RecVAE* significantly outperforms previously proposed autoencoder-based models, including *Mult*-VAE and *RaCT*, across classical collaborative filtering datasets, and present a detailed ablation study to assess our new developments. Code and models are available at <https://github.com/ilya-shenbin/RecVAE>.

## CCS CONCEPTS

• **Computing methodologies** → **Latent variable models**; *Learning from implicit feedback*; *Regularization*; Neural networks; • **Information systems** → *Learning to rank*.

## KEYWORDS

deep learning, collaborative filtering, variational autoencoders

### ACM Reference Format:

Ilya Shenbin, Anton Alekseev, Elena Tutubalina, Valentin Malykh, and Sergey I. Nikolenko. 2020. RecVAE: a New Variational Autoencoder for Top-N Recommendations with Implicit Feedback. In *The Thirteenth ACM International Conference on Web Search and Data Mining (WSDM '20)*, February 3–7, 2020, Houston, TX, USA. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3336191.3371831>

WSDM '20, February 3–7, 2020, Houston, TX, USA

© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in *The Thirteenth ACM International Conference on Web Search and Data Mining (WSDM '20)*, February 3–7, 2020, Houston, TX, USA, <https://doi.org/10.1145/3336191.3371831>.

## 1 INTRODUCTION

Matrix factorization (*MF*) has become the industry standard as the foundation of recommender systems based on collaborative filtering. However, there are certain general issues that arise with this family of models. First, the number of parameters in any matrix factorization model is huge: it linearly depends on the number of both users and items, which leads to slow model learning and overfitting. Second, to make a prediction for a new user/item based on their ratings, one has to run an optimization procedure in order to find the corresponding user/item embedding. Third, only a small amount of ratings are known for some (often for a majority of) users and items, which could also lead to overfitting. This makes it necessary to heavily regularize matrix factorization models, and standard  $L_1$  or  $L_2$  regularizers are hard to tune.

Recently proposed models such as the *Collaborative Denoising Autoencoder* (*CDAE*) [39] partially solve these issues by using a parameterized function which maps user feedback to user embeddings. It performs regularization in an alternative way and makes it possible to predict item ratings for new users without additional iterative training. The *Variational Autoencoder for Collaborative Filtering* (*Mult*-VAE) [22] is a subsequent improvement of *CDAE* that extends it to multinomial distributions in the likelihood, which are more suitable for recommendations.

In this work, we propose the *Recommender VAE* (RecVAE) model for collaborative filtering with implicit feedback based on the variational autoencoder (VAE) and specifically on the *Mult*-VAE approach. *RecVAE* presents a number of important novelties that together combine into significantly improved performance. First, we have designed a new architecture for the encoder network. Second, we have introduced a novel composite prior distribution for the latent code  $z$  in the variational autoencoder. The composite prior is a mixture of a standard Gaussian prior and the latent code distribution with parameters fixed from the previous iteration of the model (Section 3.3), an idea originating from reinforcement learning where it is used to stabilize training [13, 30]. In the context of recommendations, we have also found that this prior improves training stability and performance. Third, we have developed a new approach to setting the hyperparameter  $\beta$  for the Kullback-Leibler term in the objective function (Section 3.4). We have found that

$\beta$  should be user-specific,  $\beta = \beta(\mathbf{x}_u)$ , and should depend on the amount of data (implicit feedback) available for a given user.

Finally, we introduce a novel approach for training the model. In *RecVAE*, training is done by alternating updates for the encoder and decoder (see Section 3.5). This approach has two important advantages. First, it allows to perform multiple updates of the encoder (a more complex network) for every update of the decoder (a very simple, single-layer network that contains item embeddings and biases). Second, it allows to use corrupted inputs (following the general idea of denoising autoencoders [16, 31]) only for training the encoder while still training the decoder on clean input data. This is again beneficial for the final model training due to the differing complexities of the encoder and decoder.

As a result of the above novelties, our model significantly outperforms all autoencoder-based previous works and shows competitive or better results in comparison with other models across a variety of collaborative filtering datasets, including *MovieLens-20M* (ML-20M), *Netflix Prize Dataset*, and *Million Songs Dataset* (MSD).

The paper is organized as follows. In Section 2 we review the crucial components and approaches we are to employ in the proposed methods as well as other relevant prior art. In Section 3 we describe the basic *Mult-VAE* approach and our modifications. Section 4 contains the results of a comprehensive experimental study for our model, and Section 5 concludes the paper.

## 2 BACKGROUND AND RELATED WORK

### 2.1 Variational autoencoders and their extensions

The *variational autoencoder* (VAE) [19, 28] is a deep latent variable model able to learn complex distributions. We begin with a brief exposition of the basic assumptions behind VAE that have been extended for collaborative filtering in *Mult-VAE* and will be further extended in this work with *RecVAE*. First, under the assumption that the dataset belongs to a low-dimensional manifold embedded in a high-dimensional space, the marginal likelihood function can be expressed via a latent code  $z$  as  $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}|z)p(z)dz$ . Since the marginal likelihood function is intractable, it is usually approximated with the evidence lower bound (ELBO):

$$\log p_\theta(\mathbf{x}) \geq \mathcal{L}_{VAE} = \mathbb{E}_{q_\phi(z|\mathbf{x})} \left[ \log p_\theta(\mathbf{x}|z) - \text{KL} \left( q_\phi(z|\mathbf{x}) \| p(z) \right) \right], \quad (1)$$

where KL is the Kullback-Leibler divergence,  $p(z)$  is the prior distribution,  $q_\phi(z|\mathbf{x})$  is a variational approximation of the posterior distribution defined as a parameterized function with  $\phi$ , and  $\theta$  are the parameters of  $p_\theta(\mathbf{x}|z)$ . This technique, known as *amortized inference*, provides additional regularization and allows to obtain variational parameters with a closed form function.

Variational autoencoders can be used not only as generative models but also for representation learning.  $\beta$ -VAE [10] is a modification of VAE designed to learn so-called *disentangled representations* by adding a regularization coefficient to the Kullback-Leibler term in the evidence lower bound. The objective function of  $\beta$ -VAE can still be considered as a valid ELBO with additional approximate posterior regularization that rescales the Kullback-Leibler divergence in

formula (1) above by a factor of  $\beta$  [12, 24]:

$$\mathcal{L}_{\beta\text{-VAE}} = \mathbb{E}_{q_\phi(z|\mathbf{x})} \left[ \log p_\theta(\mathbf{x}|z) - \beta \text{KL} \left( q_\phi(z|\mathbf{x}) \| p(z) \right) \right]. \quad (2)$$

*Denoising variational autoencoders* (DVAE) [16, 31], similar to denoising autoencoders, are trying to reconstruct the input from its corrupted version. The ELBO in this model is defined as

$$\log p_\theta(\mathbf{x}) \geq \mathcal{L}_{DVAE} = \mathbb{E}_{q_\phi(z|\tilde{\mathbf{x}})} \mathbb{E}_{p(\tilde{\mathbf{x}}|\mathbf{x})} \left[ \log p_\theta(\mathbf{x}|z) - \text{KL} \left( q_\phi(z|\tilde{\mathbf{x}}) \| p(z) \right) \right]. \quad (3)$$

It differs from VAE by the additional expectation  $\mathbb{E}_{p(\tilde{\mathbf{x}}|\mathbf{x})}$ , where  $p(\tilde{\mathbf{x}}|\mathbf{x})$  is a noise distribution, usually Bernoulli or Gaussian. Similar to how denoising autoencoders are more robust and generalize better than regular autoencoders, DVAE improves over the performance of the basic VAE and makes it possible to learn more robust approximate posterior distributions.

The *Conditional Variational Autoencoder* (CVAE) [32] is another VAE extension which is able to learn complex conditional distributions. Its ELBO is as follows:

$$\log p_\theta(\mathbf{x}|\mathbf{y}) \geq \mathcal{L}_{CVAE} = \mathbb{E}_{q_\phi(z|\mathbf{x},\mathbf{y})} \left[ \log p_\theta(\mathbf{x}|z,\mathbf{y}) - \text{KL} \left( q_\phi(z|\mathbf{x},\mathbf{y}) \| p_\theta(z,\mathbf{y}) \right) \right], \quad (4)$$

which is the same as the ELBO for regular VAE shown in (1) but with all distributions conditioned on  $\mathbf{y}$ . *VAE with Arbitrary Conditioning* (VAEAC) [17], based on CVAE, solves the imputation problem for missing features, which is in many ways similar to collaborative filtering. It has an ELBO similar to (4) where the variables are the unobserved features  $\mathbf{x}_b$  and the condition is  $\mathbf{x}_{1-b}$  for some binary mask  $b$ :

$$\log p_{\theta,b}(\mathbf{x}_b|\mathbf{x}_{1-b},b) \geq \mathcal{L}_{VAEAC} = \mathbb{E}_{q_\phi(z|\mathbf{x},b)} \left[ \log p_\theta(\mathbf{x}_b|z,\mathbf{x}_{1-b},b) - \text{KL} \left( q_\phi(z|\mathbf{x},b) \| p_\theta(z|\mathbf{x}_{1-b},b) \right) \right]; \quad (5)$$

an important point here is that the mask  $b$  can be different and even have different number of ones (unobserved features) for different inputs  $\mathbf{x}$ . However, this approach cannot be directly applied to the implicit feedback case, which we consider in this work.

### 2.2 Autoencoders and Regularization for Collaborative Filtering

Let  $U$  and  $I$  be the sets of users and items respectively in a collaborative filtering problem. Consider the implicit feedback matrix  $\mathbf{X} \in \{0,1\}^{|U| \times |I|}$ , where  $x_{ui} = 1$  if the  $u$ th user positively interacted with (liked, bought, watched etc.) the  $i$ th item (movie, good, article etc.) and  $x_{ui} = 0$  otherwise. We denote by  $\mathbf{x}_u$  the feedback vector of user  $u$ .

The basic idea of the *Collaborative Denoising Autoencoder* (CDAE) model [39] is to reconstruct the user feedback vector  $\mathbf{x}_u$  from its corrupted version  $\tilde{\mathbf{x}}_u$ . The corrupted vector  $\tilde{\mathbf{x}}_u$  is obtained by randomly removing (setting to zero) some values in the vector  $\mathbf{x}_u$ . The encoder part of the model maps  $\tilde{\mathbf{x}}_u$  to the hidden state. In CDAE, both encoder and decoder are single neural layers (so the model itself is basically a feedforward neural network with one hidden

layer), with a user input node providing user-specific weights and the rest of the weights shared across all users; the latent representation  $z_u$  and reconstructed feedback  $\hat{x}_u$  are computed as

$$z_u = \sigma(W^\top \tilde{x}_u + V_u + \mathbf{b}), \quad (6)$$

$$\hat{x}_u = \sigma(W' z_u + \mathbf{b}'), \quad (7)$$

where  $W$  and  $W'$  are input-to-hidden and hidden-to-output weight matrices respectively, and  $V_u$  is the weight vector for the user input node). Note that the matrix  $W$  can be considered as a matrix of item embeddings, and the matrix  $V$ , as the matrix of user embeddings.

Previous work also indicates that regularization plays a central role in collaborative filtering. Models based on matrix factorization (MF) with user/item embeddings almost invariably have an extremely large number of parameters that grows with dataset size; even a huge dataset with user ratings and other kinds of feedback cannot have more than a few dozen ratings per user (real users will not rate more items than that), so classical MF-based collaborative filtering models require heavy regularization [1, 2, 26]. The standard solution is to use simple  $L_2$  or  $L_1$  regularizers for the embedding weights, although more flexible priors have also been used in literature [20, 29]. The works [22, 39] present an alternative way of regularization based on the amortization of user embeddings coupled with denoising [16, 31, 37]. Several more models are reviewed in Section 4.3.

The model which is nearest to our current work in prior art is *Multinomial VAE (Mult-VAE)* [22], an extension of variational autoencoders for collaborative filtering with implicit feedback. In the next section, we begin with a detailed description of this model and then proceed to presenting our novel contributions.

### 3 PROPOSED APPROACH

#### 3.1 Mult-VAE

We begin with a description of the *Mult-VAE* model proposed in [22]. The basic idea of *Mult-VAE* is similar to VAE but with the multinomial distribution as the likelihood function instead of Gaussian and Bernoulli distributions commonly used in VAE. The generative model samples a  $k$ -dimensional latent representation  $z_u$  for a user  $u$ , transforms it with a function  $f_\theta: \mathbb{R}^k \rightarrow \mathbb{R}^{|I|}$  parameterized by  $\theta$ , and then the feedback history  $\mathbf{x}_u$  of user  $u$ , which consists of  $n_u$  interactions (clicks, purchases etc.), is assumed to be drawn from the multinomial distribution:

$$z_u \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad \boldsymbol{\pi}(z_u) = \text{softmax}(f_\theta(z_u)), \quad (8)$$

$$\mathbf{x}_u \sim \text{Mult}(n_u, \boldsymbol{\pi}(z_u)). \quad (9)$$

Note that classical collaborative filtering models also follow this scheme with a linear  $f_\theta$ ; the additional flexibility of *Mult-VAE* comes from parameterizing  $f$  with a neural network with parameters  $\theta$ .

To estimate  $\theta$  one has to approximate the intractable posterior  $p(z_u | \mathbf{x}_u)$ . This, similar to regular VAE, is done by constructing an evidence lower bound for the variational approximation where  $q(z_u)$  is assumed to be a fully factorized diagonal Gaussian distribution:  $q(z_u) = \mathcal{N}(\boldsymbol{\mu}_u, \text{diag}(\sigma_u^2))$ . The resulting ELBO is

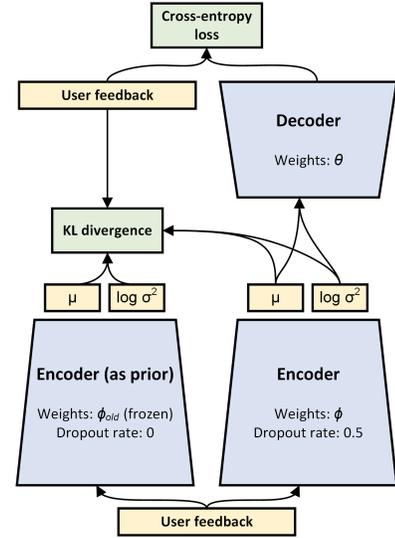


Figure 1: RecVAE architecture.

$$\mathcal{L}_{\text{Mult-VAE}} =$$

$$\mathbb{E}_{q_\phi(z_u | \mathbf{x}_u)} \left[ \log p_\theta(\mathbf{x}_u | z_u) - \beta \text{KL} \left( q_\phi(z_u | \mathbf{x}_u) \| p(z_u) \right) \right], \quad (10)$$

which follows the general VAE structure with an additional hyperparameter  $\beta$  that allows to achieve a better balance between latent code independence and reconstruction accuracy, following the  $\beta$ -VAE framework [11].

The likelihood  $p_\theta(\mathbf{x}_u | z_u)$  in the ELBO of *Mult-VAE* is multinomial distribution. The logarithm of multinomial likelihood for a single user  $u$  in *Mult-VAE* is now

$$\log \text{Mult}(\mathbf{x}_u | n_u, \mathbf{p}_u) = \sum_i \mathbf{x}_{ui} \log \mathbf{p}_{ui} + C_u, \quad (11)$$

where  $C_u$  is the logarithm of the normalizing constant which is ignored during training. We treat it as a sum of cross-entropy losses.

#### 3.2 Model Architecture

Before introducing novel regularization techniques, we provide a general description of the proposed model. Our model is inherited from *Mult-VAE*, but we also suggest some architecture changes. The general architecture is shown on Figure 1; the figure reflects some of the novelties we will discuss below in this section.

The first change is that we move to a denoising variational autoencoder, that is, move from the ELBO as shown in (10) to

$$\mathcal{L}_{\text{Mult-VAE}} = \mathbb{E}_{q_\phi(z_u | \tilde{\mathbf{x}}_u)} \mathbb{E}_{p(\tilde{\mathbf{x}}_u | \mathbf{x}_u)} \left[ \log p_\theta(\mathbf{x}_u | z_u) - \beta \text{KL} \left( q_\phi(z_u | \mathbf{x}_u) \| p(z_u) \right) \right]. \quad (12)$$

Note that while the original paper [22] compares *Mult-VAE* with *Mult-DAE*, a denoising autoencoder that applies Bernoulli-based noise to the input but does not have the VAE structure (*Mult-DAE* is a regular denoising autoencoder), in reality the authors used denoising for *Mult-VAE* as well. This is evidenced both by their code base and by our experiments, where we were able to match the

results of [22] when we used denoising and got nowhere even close without denoising (we will return to this discussion in Section 3.5). According to our intuition and experiments, the input noise for the denoising autoencoder and latent variable noise of Monte Carlo integration play different roles: the former forces the model not only to reconstruct the input vector but also to predict unobserved feedback, while the latter leads to more robust embedding learning.

Similar to *Mult*-VAE, we use the noise distribution  $p(\tilde{\mathbf{x}}|\mathbf{x})$  defined as elementwise multiplication of the vector  $\mathbf{x}$  by a vector of Bernoulli random variables parameterized by their mean  $\mu_{\text{noise}}$ .

We keep the structure of both likelihood and approximate posterior unchanged:

$$p_{\theta}(\mathbf{x}_u|z_u) = \text{Mult}(\mathbf{x}|n_u, \boldsymbol{\pi}(z_u)), \quad (13)$$

$$\boldsymbol{\pi}(z_u) = \text{softmax}(f_{\theta}(z_u)), \quad (14)$$

$$f_{\theta}(z_u) = Wz_u + \mathbf{b}, \quad (15)$$

$$q_{\phi}(z_u|\mathbf{x}_u) = \mathcal{N}(z_u|\psi_{\phi}(\mathbf{x}_u)), \quad (16)$$

where  $\psi_{\phi}(\cdot)$  is the inference network, parameterized by  $\phi$ , that predicts the mean vector and (diagonal) covariance matrix for the latent code  $z_u$ . However, we change the underlying neural networks. Our proposed architecture for the inference network is shown on Figure 3; it uses the ideas of densely connected layers from dense CNNs [15], swish activation functions [27], and layer normalization [21]. The decoder network is a simple linear layer with softmax activation, where  $\theta = \{W, \mathbf{b}\}$ . Here  $W$  and  $\mathbf{b}$  can be considered as the item embeddings matrix and the item bias vector respectively. In a similar way, we can consider encoder  $\psi_{\phi}(\cdot)$  as a function that maps user feedback to user embeddings.

### 3.3 Composite prior

Both input and output of *Mult*-VAE are high-dimensional sparse vectors. Besides, while shared amortized approximate posterior regularizes learning, posterior updates for some parts of the observed data may hurt variational parameters corresponding to other parts of the data. These features may lead to instability during training, an effect similar to a well-known “forgetting” effect in reinforcement learning. Previous work on policy-based reinforcement learning showed that it helps to regularize model parameters by bringing them closer to model parameters on the previous epoch [13, 30]; in reinforcement learning, it helps to make the final score grow more smoothly, preventing the model from forgetting good behaviours.

A direct counterpart of these ideas in our setting would be to use a standard Gaussian prior for the latent code  $z$  and add a separate regularization term in the form of the KL divergence between the new parameter distribution  $q_{\phi}(z|\mathbf{x})$  and the previous one  $q_{\phi_{old}}(z|\mathbf{x})$ , where  $\phi_{old}$  are the parameters from the previous epoch of the learning process. However, in our experiments it worked better to unite these two ideas (prior and additional regularizer) by using a composite prior

$$p(z|\phi_{old}, \mathbf{x}) = \alpha \mathcal{N}(z|0, \mathbf{I}) + (1 - \alpha) q_{\phi_{old}}(z|\mathbf{x}), \quad (17)$$

i.e., a convex combination (with  $0 \leq \alpha \leq 1$ ) of a standard normal distribution and an approximate posterior  $q_{\phi_{old}}(z|\mathbf{x})$  with fixed parameters carried over from the previous epoch. The second term regulates large steps during variational parameters optimization

and can be interpreted as an auxiliary loss function, while the first term prevents overfitting.

Note that this approach is not equivalent mathematically to a Gaussian prior and a separate KL regularizer that pulls current variational parameters to their previous values, and the fact that it works better makes this composite prior into a new meaningful contribution. We also note several works that argue for the benefits of trainable and/or complex prior distributions [35, 40], although in our experiments these approaches have not brought any improvements compared to the prior proposed above.

Conditioning the prior on variational parameters from the previous training epoch converts our model to a conditional variational autoencoder where we assume both approximate posterior and likelihood to be conditionally independent of variational parameters from the previous epoch. Comparing our model to VAEAC [17], the latter has a noised conditional prior while in our model we add noise to the approximate posterior during training. Also, unlike VAEAC, we do not train prior parameters.

### 3.4 Rescaling KL divergence

We have already mentioned the  $\beta$ -VAE framework [11] which is crucial for the performance of *Mult*-VAE and, by extension, for *Rec*VAE. However, the question of how to choose or change  $\beta$  is still not solved conclusively. Some works (see, e.g., [5]) advocate to increase the value of  $\beta$  from 0 to 1 during training, with 1 yielding the basic VAE model and the actual ELBO. For the training of *Mult*-VAE, the authors of [22] proposed to increase  $\beta$  from 0 up to some constant. In our experiments, we have not found any improvements when  $\beta$  is set to increase over some schedule, so we propose to keep scale factor fixed; this also makes it easier to find the optimal value for this hyperparameter.

Moreover, we propose an alternative view on KL divergence rescaling. Assume that the user feedback data is partially observed. We denote by  $X_u^o$  the set of items which user  $u$  has positively interacted with (according to observed data);  $X_u^f$  similarly denotes the full set of items which user  $u$  has positively interacted with (together with unobserved items). Items in  $X_u^f$  and  $X_u^o$  are represented in one-hot encoding, so that  $\mathbf{x}_u = \sum_{a \in X_u^o} \mathbf{1}_a$ , where  $\mathbf{x}_u$  is the feedback vector for user  $u$  and  $\mathbf{1}_a$  is a vector with one 1 in the position corresponding to item  $a$ . We denote  $\mathbf{x}_u = \sum_{a \in X_u^o} \mathbf{1}_a$ ,  $\mathbf{x}_u^f = \sum_{a \in X_u^f} \mathbf{1}_a$  and abbreviate  $\text{KL}_u = \text{KL}(q_{\phi}(z_u|\mathbf{x}_u)||p(z_u))$  and  $\text{KL}_u^f = \text{KL}(q_{\phi}(z_u|\mathbf{x}_u^f)||p(z_u))$ .

Consider the evidence lower bound of a variational autoencoder (1) with multinomial likelihood. It can be rewritten as follows:

$$\begin{aligned} \mathcal{L} &= \mathbb{E}_{q_{\phi}(z_u|\mathbf{x}_u^f)} \left[ \log \text{Mult}(\mathbf{x}_u^f|\boldsymbol{\pi}(z_u)) - \text{KL}_u^f \right] = \\ & \mathbb{E}_{q_{\phi}(z_u|\mathbf{x}_u^f)} \left[ \sum_{a \in X_u^f} \log \text{Cat}(\mathbf{1}_a|\boldsymbol{\pi}(z_u)) - \text{KL}_u^f \right] + C_u = \\ & \mathbb{E}_{q_{\phi}(z_u|\mathbf{x}_u^f)} \sum_{a \in X_u^f} \left[ \log \text{Cat}(\mathbf{1}_a|\boldsymbol{\pi}(z_u)) - \frac{1}{|X_u^f|} \text{KL}_u^f \right] + C_u, \end{aligned} \quad (18)$$

where  $\text{Cat}(\mathbf{1}_a | \mathbf{p}_u) = p_{ua}$  is the categorical distribution, and  $C'_u$  a constant that depends on normalizing constant of the multinomial distribution  $\text{Mult}(\mathbf{x}_u^f | \boldsymbol{\pi}(z_u))$ , which does not affect optimization.

We approximate the ELBO obtained above by summing over observed feedback only and assuming that  $q_\phi(z_u | \mathbf{x}_u) \approx q_\phi(z_u | \mathbf{x}_u^f)$  and therefore  $\text{KL}_u \approx \text{KL}_u^f$ . In the sequence of equalities below, we first approximate the sum over  $\mathbf{X}_u^f$  from (18) with a sum over  $\mathbf{X}_u^o$  with the corresponding rescaling coefficient  $|\mathbf{X}_u^f|/|\mathbf{X}_u^o|$ , then approximate  $\text{KL}_u^f$  with  $\text{KL}_u$ :

$$\begin{aligned} \mathcal{L} &\approx \frac{|\mathbf{X}_u^f|}{|\mathbf{X}_u^o|} \mathbb{E}_{q_\phi(z_u | \mathbf{x}_u^f)} \sum_{a \in \mathbf{X}_u^o} \left[ \log \text{Cat}(\mathbf{1}_a | \boldsymbol{\pi}(z_u)) - \frac{1}{|\mathbf{X}_u^f|} \text{KL}_u^f \right] + C'_u \\ &\approx \frac{|\mathbf{X}_u^f|}{|\mathbf{X}_u^o|} \mathbb{E}_{q_\phi(z_u | \mathbf{x}_u)} \sum_{a \in \mathbf{X}_u^o} \left[ \log \text{Cat}(\mathbf{1}_a | \boldsymbol{\pi}(z_u)) - \frac{1}{|\mathbf{X}_u^f|} \text{KL}_u \right] + C'_u \\ &= \frac{|\mathbf{X}_u^f|}{|\mathbf{X}_u^o|} \mathbb{E}_{q_\phi(z_u | \mathbf{x}_u)} \left[ \sum_{a \in \mathbf{X}_u^o} \log \text{Cat}(\mathbf{1}_a | \boldsymbol{\pi}(z_u)) - \frac{|\mathbf{X}_u^o|}{|\mathbf{X}_u^f|} \text{KL}_u \right] + C'_u \\ &= \frac{|\mathbf{X}_u^f|}{|\mathbf{X}_u^o|} \mathbb{E}_{q_\phi(z_u | \mathbf{x}_u)} \left[ \log \text{Mult}(\mathbf{x}_u | \boldsymbol{\pi}(z_u)) - \frac{|\mathbf{X}_u^o|}{|\mathbf{X}_u^f|} \text{KL}_u \right] + C''_u, \end{aligned}$$

where  $C'_u$  and  $C''_u$  are constants related to normalizing constants of the multinomial distribution.

Finally, we make the assumption that  $|\mathbf{X}_u^f|$  is the same for every  $u$  and denote  $\gamma = 1/|\mathbf{X}_u^f|$ . While this assumption might look strange, in effect  $|\mathbf{X}_u^f|$  is not merely unknown but is actually under our control: it is the number of items  $u$  has feedback about plus the number of items for recommendation. Therefore, we reduce all  $|\mathbf{X}_u^f|$  into a single hyperparameter  $\gamma$ :

$$\mathcal{L} \approx \frac{1}{\gamma |\mathbf{X}_u^o|} \mathbb{E}_{q_\phi(z_u | \mathbf{x}_u)} \left[ \log \text{Mult}(\mathbf{x}_u | \boldsymbol{\pi}(z_u)) - \gamma |\mathbf{X}_u^o| \text{KL}_u \right]. \quad (19)$$

In practice, we drop the  $1/\gamma |\mathbf{X}_u^o|$  factor in front of the expectation: it does not change the relation between the log likelihood and KL regularizer but rather changes the learning rate individually for each user, which slightly degraded performance in our experiments. The resulting KL divergence scaling factor,

$$\beta' = \beta'(\mathbf{x}_u) = \gamma |\mathbf{X}_u^o| = \gamma \sum_i x_{ui}, \quad (20)$$

where  $\gamma$  is a constant hyperparameter shared across all users and chosen with cross-validation, works better (see below). In total, we have proposed and motivated an approach where the  $\beta$  constant in  $\beta$ -VAE is proportional to the amount of feedback available for the current user,  $|\mathbf{X}_u^o|$ ; this is an important modification that has led to significant improvements in our experiments.

### 3.5 Alternating Training and Regularization by Denoising

Alternating least squares (ALS) [2] is a popular technique for matrix factorization. We train our model in a similar way, alternating between user and item embeddings. User embeddings are amortized by the inference network, while each item embedding is trained individually. This means that the two groups of parameters,  $\phi$  in the

---

#### ALGORITHM 1: Proposed training procedure

---

**Data:**  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_{|U|}\}$   
**Result:**  $\phi, \theta$   
**for**  $n := 1, \dots, N$  **do**  
    **for**  $m := 1, \dots, M_{\text{enc}}$  **do**  
        Sample batch  $\{\mathbf{x}_1, \dots, \mathbf{x}_b\} \sim \mathcal{D}$ ;  
        Update  $\phi$  based on  $\tilde{\mathcal{L}}$ ;  
    **end**  
     $\phi_{old} := \phi$ ;  
    **for**  $m := 1, \dots, M_{\text{dec}}$  **do**  
        Sample batch  $\{\mathbf{x}_1, \dots, \mathbf{x}_b\} \sim \mathcal{D}$ ;  
        Update  $\theta$  based on  $\tilde{\mathcal{L}}_{\text{dec}}$ ;  
    **end**  
**end**

---

encoder network and  $\theta$  in the item matrix and bias vector, are of a different nature and it might be best to train them in different ways. We propose to update  $\phi$  and  $\theta$  alternately with a different number of iterations: since the encoder is a much more complex network than the decoder, we make multiple updates of  $\phi$  for each update of  $\theta$ . This separation of training steps allows for another improvement. Both our experiments and prior art indicate that reconstruction of corrupted input data, i.e., using *denoising* autoencoders, is necessary in autoencoder-based collaborative filtering, forcing the model to learn to not only reconstruct previously observed feedback but also generalize to unobserved feedback during inference. However, we noticed that performance improves if we do not corrupt input data during the training of  $\theta$  and leave the denoising purely for  $\phi$ . Since other types of regularization (such as  $L_2$  or moving to a Bayesian decoder) also lead to degraded performance, it appears that decoder parameters are overregularized. Thus, we propose to train the decoder as part of a basic vanilla VAE, with no denoising applied. As for the encoder, however, we train it as part of the denoising variational autoencoder.

### 3.6 Summary

To summarize the proposed regularizers and changes, we first write down the ELBO for our model:

$$\mathcal{L} = \mathbb{E}_{q_\phi(z | \tilde{\mathbf{x}})} \mathbb{E}_{q(\tilde{\mathbf{x}} | \mathbf{x})} \left[ \log p_\theta(\mathbf{x} | z) - \beta'(\mathbf{x}) \text{KL} \left( q_\phi(z | \tilde{\mathbf{x}}) \| p(z | \phi_{old}, \mathbf{x}) \right) \right], \quad (21)$$

where the conditional prior distribution  $p(z | \phi_{old}, \mathbf{x})$  has been defined in (17), and the modified weight  $\beta'(\mathbf{x})$  of the KL term in (20).

To keep the input uncorrupted while training the decoder, we introduce a modified objective function for  $\theta$  updates (we skip the KL term entirely because it does not depend on  $\theta$ ):

$$\mathcal{L}_{\text{dec}} = \mathbb{E}_{q_\phi(z | \mathbf{x})} \log p_\theta(\mathbf{x} | z). \quad (22)$$

We train the model using alternating updates as shown in Algorithm 1; different parameters  $M_{\text{enc}}$  and  $M_{\text{dec}}$  reflect the asymmetry between encoder and decoder that we have discussed above. During training, we approximate both inner and outer expectation by single Monte-Carlo samples and use reparametrization similar to

regular VAE. Now we can introduce the empirical lower bound

$$\begin{aligned} \tilde{\mathcal{L}}(\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\phi}_{old}) = & \log p_{\boldsymbol{\theta}}(\mathbf{x} | \mathbf{z}^{(*)}) - \\ & - \beta'(\mathbf{x}) \left( -\log q_{\boldsymbol{\phi}}(\mathbf{z}^{(*)} | \tilde{\mathbf{x}}) + \log p(\mathbf{z}^{(*)} | \boldsymbol{\phi}_{old}, \mathbf{x}) \right), \end{aligned} \quad (23)$$

where  $\mathbf{z}^{(*)} = g(\boldsymbol{\epsilon}, \boldsymbol{\mu}, \boldsymbol{\sigma})$ , for a Gaussian posterior  $g(\boldsymbol{\epsilon}, \boldsymbol{\mu}, \boldsymbol{\sigma}) = \boldsymbol{\epsilon} \cdot \boldsymbol{\mu} + \boldsymbol{\sigma}$ , where  $[\boldsymbol{\mu}, \log \boldsymbol{\sigma}^2] = \psi_{\boldsymbol{\phi}}(\mathbf{x})$ ,  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ , and  $\tilde{\mathbf{x}} = \mathbf{x} \odot \mathbf{m}$  is the noised input,  $\mathbf{m} \sim \text{Bernoulli}(\mu_{\text{noise}})$ . We use Monte-Carlo sampling for both log likelihood and KL divergence since the KL divergence between a Gaussian and a mixture of Gaussians cannot be calculated analytically. The dropout layer on Figure 3 serves for noising; it is turned off during evaluation, decoder learning phase, and in the composite prior. Since we use the multinomial likelihood, the main component of the loss function is the classification cross-entropy as shown on Figure 1. The empirical lower bound for decoder training  $\tilde{\mathcal{L}}_{\text{dec}}$  is introduced in a similar way.

Similar to *Multi-VAE*, our model is also able to make predictions for users whose feedback was not observed during training. To do that, we predict the user embedding with the inference network  $\mathbf{z} = \psi_{\boldsymbol{\phi}}(\mathbf{x})$  based of the feedback  $\mathbf{x}$  observed at test time and then predict top items using the trained decoder  $p_{\boldsymbol{\theta}}(\mathbf{x} | \mathbf{z})$ .

## 4 EXPERIMENTAL EVALUATION

### 4.1 Metrics

Following prior art, we evaluate the models with information retrieval metrics for ranking quality:  $\text{Recall}@k$  and  $\text{NDCG}@k$ . Both metrics compares top- $k$  predictions of a model with the test set  $\mathbf{X}_u^t$  of user feedback for user  $u$ . To obtain recommendations for *RecVAE* and similar models, we sort the items in descending order of the likelihood predicted by the decoder, exclude items from the training set, and denote the item at the  $n$ th place in the resulting list as  $R_u^{(n)}$ . In this notation, evaluation metrics for a user  $u$  are defined as follows:

$$\text{Recall}@k(u) = \frac{1}{\min(M, |\mathbf{X}_u^t|)} \sum_{n=1}^k \mathbb{1} \left[ R_u^{(n)} \in \mathbf{X}_u^t \right], \quad (24)$$

where  $\mathbb{1}[\cdot]$  denotes the indicator function,

$$\begin{aligned} \text{DCG}@k(u) &= \sum_{n=1}^k \frac{2^{\mathbb{1}[R_u^{(n)} \in \mathbf{X}_u^t]} - 1}{\log(n+1)}, \\ \text{NDCG}@k(u) &= \left( \sum_{n=1}^{|\mathbf{X}_u^t|} \frac{1}{\log(n+1)} \right)^{-1} \text{DCG}@k(u), \end{aligned} \quad (25)$$

i.e.,  $\text{NDCG}@k(u)$  is defined as  $\text{DCG}@k(u)$  divided by its highest theoretically possible value.  $\text{Recall}@k$  accounts for all top- $k$  items equally, while  $\text{NDCG}@k$  assigns larger weights to top ranked items; thus, it is natural to choose a larger value of  $k$  for  $\text{NDCG}@k$ .

### 4.2 Datasets

We have evaluated *RecVAE* on the *MovieLens-20M* dataset<sup>1</sup> [9], *Netflix Prize Dataset*<sup>2</sup> [3], and *Million Songs Dataset*<sup>3</sup> [4]. We have

preprocessed the datasets in accordance with the *Multi-VAE* approach [22]. Dataset statistics after preprocessing are as follows: *MovieLens-20M* contains 9,990,682 ratings on 20,720 movies provided by 136,677 users, *Netflix Prize Dataset* contains 56,880,037 ratings on 17,769 movies provided by 463,435 users, and *Million Songs Dataset* contains 33,633,450 ratings on 41,140 songs provided by 571,355 users. In order to evaluate the model on users unavailable during the training, we have held out 10,000 users for validation and testing for *MovieLens-20M*, 40,000 users for the *Netflix Prize*, and 50,000 users for the *Million Songs Dataset*. We used 80% of the ratings in the test set in order to compute user embeddings and evaluated the model on the remaining 20% of the ratings.

### 4.3 Baselines

We compare the performance of the proposed model with several baselines, which we divide into three groups. The first group includes linear models from classical collaborative filtering. Weighted Matrix Factorization (WMF) [14] binarizes implicit feedback  $r_{ua}$  (number of times user  $u$  positively interacted with item  $a$ ) as  $p_{ua} = \mathbb{1}[r_{ua} > 0]$  and decomposes the matrix of  $p_{ua}$  similar to SVD but with confidence weights that increase with  $r_{ua}$ :

$$\min_{\mathbf{w}_u, \mathbf{w}_a} \sum_{u,a} (1 + \alpha r_{ua}) (p_{ua} - \mathbf{w}_u^T \mathbf{w}_a)^2 + \lambda (\|W_u\|_2 + \|W_a\|_2), \quad (26)$$

where  $\|\cdot\|_2$  is the  $L_2$ -norm.

The Sparse Linear Method (SLIM) for top- $N$  recommendation [25] learns a sparse matrix of aggregation coefficients  $W$  that corresponds to the weights of rated items aggregated to produce recommendation scores, i.e., the prediction is  $\tilde{r}_{ia} = \mathbf{w}_i^T \mathbf{w}_a$ , or in matrix form  $\tilde{R} = RW$ , with the resulting optimization problem

$$\min_W \frac{1}{2} \|R - RW\|_F^2 + \frac{\beta}{2} \|W\|_F^2 + \lambda \|W\|_1 \quad (27)$$

subject to  $W \geq 0$  and  $\text{diag}(W) = 0$ , where  $\|\cdot\|_F$  is the Frobenius norm and  $\|\cdot\|_1$  is the  $L_1$ -norm. The Embarrassingly Shallow Autoencoder (EASE) [33] is a further improvement on SLIM with a closed-form solution, where the non-negativity constraint and  $L_1$  regularization are dropped. Despite the name, we do not refer to this model as an autoencoder-based one.

The second is learning to rank methods such as WARP [38] and *LambdaNet* [6]. *LambdaNet* is a learning to rank approach that allows to work with objective functions that are either flat or discontinuous; the main idea is to approximate the gradient of each item's score in the ranked list for the corresponding query. The result can be treated as a direction where the item is to "move" in the ranked list for the query when sorted with respect to the newly predicted scores. WARP considers every user-item pair  $(u, i)$  corresponding to a positive interaction when training to predict scores for recommendation ranking. For every user  $u$ , other random items  $i'$  are sampled until the first one with the predicted score lower than that of the  $i$  is found. The pair  $(u, i')$  is then treated as a negative sample, and  $(u, i)$  and  $(u, i')$  are employed as positive and a negative contributions respectively for the approximation of the indicator function for the ranking loss similar to those introduced in an earlier work [36]. For more details about WARP and its performance we refer to the original work [38].

<sup>1</sup><https://grouplens.org/datasets/movielens/20m/>

<sup>2</sup><https://www.netflixprize.com/>

<sup>3</sup><http://millionsongdataset.com/>

The third group includes autoencoder-based methods that we have already discussed in Sections 2 and 3.1: CDAE [39], *Mult-DAE*, and *Mult-VAE* [22]. The proposed *RecVAE* model can also be considered as a member of this group. We also consider the very recently proposed *Ranking-Critical Training* (RaCT) [23]. This model adopts an actor-critic framework for collaborative filtering on implicit data. A critic (represented by a neural network) learns to approximate the ranking scores, which in turn improves an MLE-based nonlinear latent variable model (VAE and possibly its variations) with the learned ranking-critical objectives. The critic neural network is feature-based and is using posterior sampling as exploration for better estimates. Both actor and critic are pretrained in this model. Scores for these models have been taken from [22] and [23].

#### 4.4 Evaluation setup

*RecVAE* was trained with the *Adam* optimizer [18] with learning rate  $= 5 \cdot 10^{-4}$ , and batch size  $b = 500$ .  $M_{\text{dec}}$  is selected so that each element in the dataset is selected once per epoch, i.e.,  $M_{\text{dec}} = \frac{|U|}{\text{batch size}}$ , the Bernoulli noise parameter is  $\mu_{\text{noise}} = 0.5$ , and  $M_{\text{enc}} = 3M_{\text{dec}}$ . In addition to the standard normal distribution and the old posterior as parts of the composite prior, we also add a normal distribution with zero mean and  $\log \sigma^2 = 10$  to the mixture. Weights of these mixture components are  $3/20$ ,  $3/4$ , and  $1/10$  respectively. Since the model is sensitive to changes of the parameter  $\gamma$ , we have picked it individually for each dataset:  $\gamma = 0.005$  for *MovieLens-20M*,  $\gamma = 0.0035$  for the *Netflix Prize Dataset*, and  $\gamma = 0.01$  for the *Million Songs Dataset*. Each model was trained during  $N = 50$  epochs ( $N = 100$  for *MSD*), choosing the best model by the NDCG@100 score on a validation subset. Initially, we fit the model for *MovieLens-20M* and then fine-tuned it for the *Netflix Prize Dataset* and *MSD*.

#### 4.5 Results

Performance scores for top-N recommendations in the three datasets are presented in Table 1. The results clearly show that in terms of recommendation quality, *RecVAE* outperforms all previous autoencoder-based models across all datasets in the comparison, in particular, with a big improvement over *Mult-VAE*. Moreover, new features of *RecVAE* and RaCT are independent and can be used together for even higher performance. Nevertheless, the proposed model significantly outperforms *EASE* only on *MovieLens-20M* datasets, and shows competitive performance on the *Netflix Prize Dataset*.

For competing models in the comparison, we have used metrics reported in prior art; for *RecVAE* we have also indicated confidence intervals, showing that the difference in scores is significant.

#### 4.6 Ablation study and negative results

In order to demonstrate that each of the new features we introduced for *RecVAE* compared to *Mult-VAE* indeed helps to improve performance, we have performed a detailed ablation study, comparing various subsets of the features: (1) new encoder architecture, (2) composite prior for the latent codes, (3)  $\beta$  rescaling, (4) alternating training, and (5) removing denoising for the decoder.

Numerical results of the ablation study are presented in Table 2. We see that each new feature indeed improves the results, with all proposed new features leading to the best NDCG@100 scores on

**Table 1: Evaluation scores for *RecVAE* and baseline models on *MovieLens-20M*, *Netflix Prize Dataset*, and *MSD*. The best results are highlighted in bold. The second best ones are underlined.**

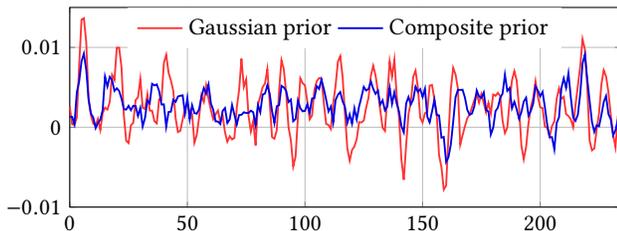
	Recall@20	Recall@50	NDCG@100
<b>MovieLens-20M Dataset</b>			
WARP [38]	0.314	0.466	0.341
LambdaNet[6]	0.395	0.534	0.427
WMF [14]	0.360	0.498	0.386
SLIM [25]	0.370	0.495	0.401
CDAE [39]	0.391	0.523	0.418
Mult-DAE [22]	0.387	0.524	0.419
Mult-VAE [22]	0.395	0.537	0.426
RaCT [23]	<u>0.403</u>	<u>0.543</u>	<u>0.434</u>
EASE [33]	0.391	0.521	0.420
RecVAE (ours)	<b>0.414</b> $\pm$ 0.0027	<b>0.553</b> $\pm$ 0.0028	<b>0.442</b> $\pm$ 0.0021
<b>Netflix Prize Dataset</b>			
WARP [38]	0.270	0.365	0.306
LambdaNet[6]	0.352	0.441	0.386
WMF [14]	0.316	0.404	0.351
SLIM [25]	0.347	0.428	0.379
CDAE [39]	0.343	0.428	0.376
Mult-DAE [22]	0.344	0.438	0.380
Mult-VAE [22]	0.351	0.444	0.386
RaCT [23]	0.357	<u>0.450</u>	0.392
EASE [33]	<b>0.362</b>	0.445	<u>0.393</u>
RecVAE (ours)	<u>0.361</u> $\pm$ 0.0013	<b>0.452</b> $\pm$ 0.0013	<b>0.394</b> $\pm$ 0.0010
<b>Million Songs Dataset</b>			
WARP [38]	0.206	0.302	0.249
LambdaNet[6]	0.259	0.355	0.308
WMF [14]	0.211	0.312	0.257
SLIM [25]	–	–	–
CDAE [39]	0.188	0.283	0.237
Mult-DAE [22]	0.266	0.363	0.313
Mult-VAE [22]	0.266	0.364	0.316
RaCT [23]	0.268	0.364	0.319
EASE [33]	<b>0.333</b>	<b>0.428</b>	<b>0.389</b>
RecVAE (ours)	<u>0.276</u> $\pm$ 0.0010	<u>0.374</u> $\pm$ 0.0011	<u>0.326</u> $\pm$ 0.0010

all three datasets. Some new features are complementary: e.g.,  $\beta$  rescaling and alternating training degrade the scores when applied individually, but together improve them; the new architecture does not bring much improvement by itself but facilitates other new features;  $\beta$  rescaling is dataset-sensitive, sometimes improving a lot and sometimes doing virtually nothing.

We have also performed extended analysis of the composite prior, namely checked how the  $\log p(z|\phi_{old}, \mathbf{x})$  regularizer that brings variational parameters closer to old ones affects model stability. This regularizer stabilizes training, as evidenced by the rate of change in the variational parameters. Figure 2 illustrates how the composite prior fixes the “forgetting” problem. It shows how NDCG@100 changes for a randomly chosen user as training progresses: each

**Table 2: Evaluation of RecVAE with different subsets of new features. The first row corresponds to *Mult*-VAE.**

New architecture	Composite prior	$\beta(x)$ rescaling	Alternating training	Decoder w/o denoising	NDCG@100		
					ML-20M	Netflix	MSD
					0.426	0.386	0.319
✓					0.428	0.388	0.320
✓	✓				0.435	0.392	0.325
✓		✓			0.435	0.390	0.321
✓			✓	✓	0.427	0.387	0.319
✓	✓	✓			0.438	0.390	0.325
	✓	✓	✓	✓	0.420	0.380	0.308
✓		✓	✓	✓	0.434	0.383	0.321
✓	✓		✓	✓	0.437	0.392	0.323
✓	✓	✓	✓	✓	0.441	0.391	0.322
✓	✓	✓	✓	✓	<b>0.442</b>	<b>0.394</b>	<b>0.326</b>

**Figure 2: Differences in NDCG@100 for a random user as a function of the training iteration.**

value is the difference in NDCG@100 for this user after each subsequent training update. Since each update changes the encoder network, it changes all user embeddings, and the changes can be detrimental for some users; note, however, that for the composite prior the changes remain positive almost everywhere while a simple Gaussian prior leads to much more volatile behaviour.

In addition, we would like to report the negative results of our other experiments. First, autoencoder-based models replace the matrix of user embeddings with a parameterized function, so it was natural to try to do the same for item embeddings. We trained RecVAE with a symmetric autoencoder that predicts top users for a given item, training it alternately with regular RecVAE and regularizing the results of each encoder with embeddings from the other model. The resulting model trained much slower, required more memory, and could not reach the results of RecVAE.

Second, we have tried to use more complex prior distributions. Mixtures of Gaussians and the variational mixture VampPrior [35] have (nearly) collapsed to a single node in our experiments, an effect previously noted in reinforcement learning [34]. The RealNVP prior [8] has yielded better performance compared to the standard Gaussian prior, but we have not been able to successfully integrate

it into the proposed composite prior: the composite prior with a Gaussian term remained the best throughout our experiments. We note this as a potential direction for further research.

Third, instead of  $\beta$ -VAE-like weighing of KL divergence, we tried to re-weigh each of the terms in the decomposed KL divergence separately. It appears natural to assume that “more precise” regularization could be beneficial for both performance and understanding. However, neither a simple decomposition into entropy and cross-entropy nor the more complex one proposed in [7] has led the model to better results.

## 5 CONCLUSION

In this work, we have presented a new model called RecVAE that combines several improvements for the basic *Mult*-VAE model, including a new encoder architecture, new composite prior distribution for the latent codes, new approach to setting the hyperparameter  $\beta$ , and a new approach to training RecVAE with alternating updates of the encoder and decoder. As a result, performance of RecVAE is comparable to EASE and significantly outperforms other models on classical collaborative filtering datasets such as *MovieLens-20M*, *Netflix Prize Dataset*, and *Million Songs Dataset*.

We note that while we have provided certain theoretical motivations for our modifications, these motivations are sometimes incomplete, and some of our ideas have been primarily motivated by practical improvements. We believe that a comprehensive theoretical analysis of these ideas might prove fruitful for further advances, and we highlight this as an important direction for future research.

## ACKNOWLEDGMENTS

This research was done at the Samsung-PDMI Joint AI Center at PDMI RAS and was supported by Samsung Research.

## REFERENCES

- [1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. on Knowl. and Data Eng.* 17, 6 (June 2005), 734–749. <https://doi.org/10.1109/TKDE.2005.99>
- [2] Robert M Bell and Yehuda Koren. 2007. Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights. In *icdm*, Vol. 7. Citeseer, 43–52.
- [3] James Bennett, Stan Lanning, et al. 2007. The netflix prize. In *Proceedings of KDD cup and workshop*, Vol. 2007. New York, NY, USA., 35.
- [4] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. 2011. The Million Song Dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*.
- [5] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, 10–21.
- [6] Christopher J Burges, Robert Ragno, and Quoc V Le. 2007. Learning to rank with nonsmooth cost functions. In *Advances in neural information processing systems*, 193–200.
- [7] Ricky TQ Chen, Xuechen Li, Roger Grosse, and David Duvenaud. 2018. Isolating sources of disentanglement in VAEs. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. Curran Associates Inc., 2615–2625.
- [8] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2017. Density estimation using Real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings*. OpenReview.net. <https://openreview.net/forum?id=HkpbnH9lx>
- [9] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4, Article 19 (Dec. 2015), 19 pages. <https://doi.org/10.1145/2827872>

- [10] 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. In *International Conference on Learning Representations*, Vol. 3.
- [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. In *ICLR*.
- [12] Matt Hoffman, Carlos Riquelme, and Matthew Johnson. 2017. The Beta VAE’s Implicit Prior. <http://bayesiandeeplearning.org/2017/papers/66.pdf>
- [13] Rein Houthoofd, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. 2016. Vime: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems*. 1109–1117.
- [14] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*. Ieee, 263–272.
- [15] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. 2016. Densely Connected Convolutional Networks. *CoRR* abs/1608.06993 (2016). arXiv:1608.06993 <http://arxiv.org/abs/1608.06993>
- [16] Daniel Im Jiwoong Im, Sungjin Ahn, Roland Memisevic, and Yoshua Bengio. 2017. Denoising criterion for variational auto-encoding framework. In *Thirty-First AAAI Conference on Artificial Intelligence*.
- [17] Oleg Ivanov, Michael Figurnov, and Dmitry P. Vetrov. 2019. Variational Autoencoder with Arbitrary Conditioning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. <https://openreview.net/forum?id=SyxtJh0qYm>
- [18] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). <http://arxiv.org/abs/1412.6980>
- [19] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). <http://arxiv.org/abs/1312.6114>
- [20] Neil D Lawrence and Raquel Urtasun. 2009. Non-linear matrix factorization with Gaussian processes. In *Proceedings of the 26th annual international conference on machine learning*. ACM, 601–608.
- [21] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer Normalization. *arXiv e-prints*, Article arXiv:1607.06450 (Jul 2016), arXiv:1607.06450 pages. arXiv:stat.ML/1607.06450
- [22] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 689–698.
- [23] Sam Lobel, Chunyuan Li, Jianfeng Gao, and Lawrence Carin. 2019. Towards Amortized Ranking-Critical Training for Collaborative Filtering. *arXiv preprint arXiv:1906.04281* (2019).
- [24] Emile Mathieu, Tom Rainforth, N. Siddharth, and Yee Whye Teh. 2019. Disentangling Disentanglement in Variational Autoencoders. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.), Vol. 97. PMLR, 4402–4412. <http://proceedings.mlr.press/v97/mathieu19a.html>
- [25] Xia Ning and George Karypis. 2011. Slim: Sparse linear methods for top-n recommender systems. In *2011 IEEE 11th International Conference on Data Mining*. IEEE, 497–506.
- [26] Deuk Hee Park, Hyea Kyeong Kim, Il Young Choi, and Jae Kyeong Kim. 2012. A Literature Review and Classification of Recommender Systems Research. *Expert Syst. Appl.* 39, 11 (2012), 10059–10072. <https://doi.org/10.1016/j.eswa.2012.02.038>
- [27] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. 2018. Searching for Activation Functions. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net. <https://openreview.net/forum?id=Hkuq2EkPf>
- [28] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014 (JMLR Workshop and Conference Proceedings)*, Vol. 32. JMLR.org, 1278–1286. <http://proceedings.mlr.press/v32/rezende14.html>
- [29] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proceedings of the 25th international conference on Machine learning*. ACM, 880–887.
- [30] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347 (2017). arXiv:1707.06347 <http://arxiv.org/abs/1707.06347>
- [31] Rui Shu, Hung H Bui, Shengjia Zhao, Mykel J Kochenderfer, and Stefano Ermon. 2018. Amortized inference regularization. In *Advances in Neural Information Processing Systems*. 4393–4402.
- [32] 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.
- [33] Harald Steck. 2019. Embarrassingly Shallow Autoencoders for Sparse Data. In *The World Wide Web Conference*. ACM, 3251–3257.
- [34] Yunhao Tang and Shipra Agrawal. 2018. Boosting Trust Region Policy Optimization by Normalizing Flows Policy. *CoRR* abs/1809.10326 (2018). arXiv:1809.10326 <http://arxiv.org/abs/1809.10326>
- [35] Jakub Tomczak and Max Welling. 2018. VAE with a VampPrior. In *International Conference on Artificial Intelligence and Statistics*. 1214–1223.
- [36] Nicolas Usunier, David Buffoni, and Patrick Gallinari. 2009. Ranking with ordered weighted pairwise classification. In *Proceedings of the 26th annual international conference on machine learning*. ACM, 1057–1064.
- [37] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research* 11, Dec (2010), 3371–3408.
- [38] Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *Twenty-Second International Joint Conference on Artificial Intelligence*.
- [39] 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.
- [40] Haowen Xu, Wenxiao Chen, Jinlin Lai, Zhihan Li, Youjian Zhao, and Dan Pei. 2019. On the Necessity and Effectiveness of Learning the Prior of Variational Auto-Encoder. *arXiv preprint arXiv:1905.13452* (2019).

## A NETWORK ARCHITECTURE

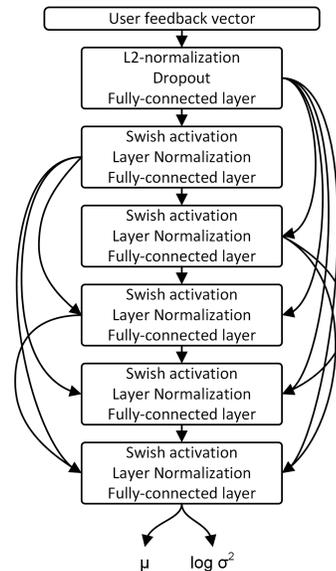


Figure 3: Architecture of the inference network  $\psi_{\phi}$ .