Neural Computing and Applications (2023) 35:7817–7831
https://doi.org/10.1007/s00521-022-08088-2

ORIGINAL ARTICLE

Deep variational models for collaborative filtering-based recommender systems

Jesús Bobadilla[1,2] · Fernando Ortega[1,2] · Abraham Gutiérrez[1,2] · Ángel González-Prieto[2,3,4]

Received: 16 September 2022 / Accepted: 22 November 2022 / Published online: 9 December 2022

Abstract

Deep learning provides accurate collaborative filtering models to improve recommender system results. Deep matrix factorization and their related collaborative neural networks are the state of the art in the field; nevertheless, both models lack the necessary stochasticity to create the robust, continuous, and structured latent spaces that variational autoencoders exhibit. On the other hand, data augmentation through variational autoencoder does not provide accurate results in the collaborative filtering field due to the high sparsity of recommender systems. Our proposed models apply the variational concept to inject stochasticity in the latent space of the deep architecture, introducing the variational technique in the neural collaborative filtering field. This method does not depend on the particular model used to generate the latent representation. In this way, this approach can be applied as a plugin to any current and future specific models. The proposed models have been tested using four representative open datasets, three different quality measures, and state-of-the-art baselines. The results show the superiority of the proposed approach in scenarios where the variational enrichment exceeds the injected noise effect. Additionally, a framework is provided to enable the reproducibility of the conducted experiments.

Keywords Recommender systems · Collaborative filtering · Variational enrichment · Deep learning

1 Introduction

Recommender Systems (RSs) are an artificial intelligence field that provides methods and models to predict and recommend items to users (e.g., films to persons, e-commerce products to costumers, services to companies, Quality of Service (QoS) to Internet of Things (IoT) devices, etc.) [1]. Current popular RSs are Spotify, Netflix, TripAdvisor, Amazon, etc. RSs are usually categorized attending to their filtering strategy, mainly demographic [2], content-based [3], context-aware [4], social [5], Collaborative Filtering (CF) [1, 6] and filtering ensembles [7, 8]. CF is the most accurate and widely used filtering approach to implement RSs. CF models have evolved from the K-Nearest Neighbors (KNN) algorithm to the Probabilistic Matrix Factorization (PMF) [9], the non-Negative Matrix Factorization (NMF) [10] and the Bayesian non-Negative Matrix Factorization (BNMF) [11]. Currently, deep learning research approaches are growing in strength: they provide improvement in accuracy compared to the Machine Learning (ML)-based Matrix Factorization (MF) models [12]. Additionally, deep learning architectures are usually more flexible than the MF-based ones, introducing combined deep and shallow learning [13], integrated content-based ensembles [14], generative approaches [15, 16], among others.

Deep Matrix Factorization (DeepMF) [17] is a neural network model that implements the popular MF concept. DeepMF was designed to take as input a user-item matrix with explicit ratings and nonpreference implicit feedback,

✉ Ángel González-Prieto
angelgonzalezprieto@ucm.es

[1] Departamento de Sistemas Informáticos, ETSI Sistemas Informáticos, Universidad Politécnica de Madrid, C. de Alan Turing, s/n, Madrid, 28031 Madrid, Spain

[2] KNODIS Research Group, Universidad Politécnica de Madrid, C. de Alan Turing, s/n, Madrid, 28031 Madrid, Spain

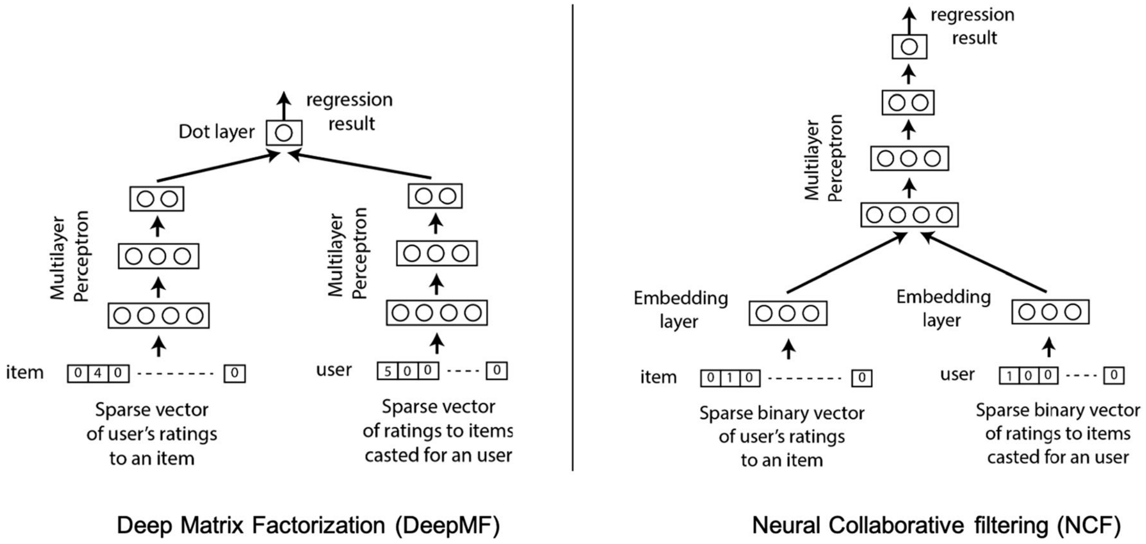
[3] Departamento de Álgebra, Geometría y Topología, Universidad Complutense de Madrid, Plaza Ciencias 3, Madrid, 28040 Madrid, Spain

[4] Instituto de Ciencias Matemáticas (CSIC-UAM-UCM-UC3M), C/ Nicolás Cabrera, 13-15, Madrid, 28049 Madrid, Spain

although current implementations use two embedding layers whose inputs are, respectively, user and items. The experimental results evidence the DeepMF superiority over the traditional approaches based on ML-focused RS, particularly the most used MF models: PMF, NMF, and BNMF. Currently, DeepMF is a popular model that is rapidly replacing the traditional MF models based on classical ML. Additionally, DeepMF has been used in the RS field to combine social behaviors (clicks, ratings,...) with images [18], and a social trust-aware RS has been implemented by using DeepMF to extract features from the user-item rating matrix for improving the initialization accuracy [19]. QoS predictions have also been addressed by using DeepMF [20]. To learn attribute representations, a DeepMF model has been used that creates a low-dimensional representation of a dataset that lends itself to a clustering interpretation [21]. Finally, the classical matrix completion task has been addressed by using the DeepMF approach [22].

The not so widely spread Neural Collaborative Filtering (NCF) model [13] may be seen as an augmented DeepMF model, where deeper layers are added to the 'Dot' one. Additionally, the 'Dot' layer can be replaced by a 'Concatenate' layer. Figure 1 shows the explained concepts. NCF slightly outperforms the DeepMF accuracy results, but it increases the required runtime to train the model and to run the forward process: it is necessary to execute the 'extra' Multi-Layer Perceptron (MLP) on top of the 'Dot' or 'Concatenate' layers. Moreover, compared to DeepMF, the NCF architecture adds new hyper-parameters to set: mainly the number of hidden layers (depth) and their size (number of neurons in each layer) of the MLP architecture.

The hypothesis of the paper is that we can improve the existing CF neural models by adding a variational stage that borrows its operative from the Variational Autoencoders (VAE). VAEs not only improve latent factor-based models, but they also manage nonlinear probabilistic latent-variable models. While VAEs have been extensively used in the image-generative area, they have rarely been covered in the CF field. Autoencoders perform a nonlinear PCA, and VAEs improve their results by performing a nonlinear factor analysis. Unfortunately, regular autoencoders do not ensure the regularity of the latent space; this is the reason why, in image processing, they do not perform fine producing new content from random encodings: Without explicit regularization, some combinations of the latent space are meaningless once decoded. The VAEs superiority comes from their 'variational' behavior, which allows to make suitable regularizations such as in the statistic variational method. Using VAEs, inputs are encoded as distributions instead of single points, making it possible to naturally express latent space regularization. The CF improvement using VAEs is because the item and the user latent factor distributions are regularized in the training stage, ensuring that their latent spaces have good properties and conveniently generalize RS predictions. The VAEs regularization has two main properties: (1) completeness: points sampled in the latent space give meaningful content once decoded, and (2) continuity: close points in the latent space provide similar contents when they are decoded. To accomplish these properties, usually regularization is done by enforcing distributions to be close to a centered and reduced standard normal distribution. Regularization involves a higher reconstruction error that can be balanced using the Kullback–Leibler divergence. The use of VAE in the CF field provides a better



Deep Matrix Factorization (DeepMF)

Neural Collaborative filtering (NCF)

Fig. 1 Deep Matrix Factorization (DeepMF) versus Neural Collaborative Filtering (NCF)

generalization; it not only can improve recommendations, but it also makes easier to use the latent codifications of items and users to make clustering, to explain recommendations, and to generate augmented datasets. The completeness and continuity properties make possible these additional benefits of the VAEs in the CF area.

The rest of the paper has been structured as follows: In Sect. 2, we describe the main ideas involved in our proposal, as well as its differences with the related work in variational CF-based recommender systems. In Sect. 3, the proposed model is explained. Section 4 shows the experiments' design, results and their discussions. Finally, Sect. 5 contains the main conclusions of the paper and the future works.

2 Fundamentals and related work

2.1 VAEs as generative models

Variational Autoencoders (VAEs) act as regular autoencoders; they aim to compress the input raw values into a latent space representation by means of an encoder neural network, whereas the decoder neural network makes the opposite operation seeking to decompress from latent space to output raw values. The main difference between classical autoencoders and VAEs is the latent space design, meaning, and operation. Classical autoencoders do not generate structured latent spaces, whereas VAEs introduce a statistical process that forces them to learn continuous and structured latent spaces. In this way, VAEs turn the samples into parameters of a statistical distribution, usually the means and variance of a Gaussian distribution. From the parameters in the multivariate distribution, we draw a random sample and a latent space sample is obtained for each training input. This operation procedure is represented in Fig. 2.

The stochasticity of the random sampling improves the robustness and forces the encoding of continuous and meaningful latent space representations, as it can be seen in Fig. 3, where it is shown the VAE latent space representation and its cumulative normal distribution.

Due to their properties, VAEs have been used as generative deep learning models in the image processing field. Reconstruction of a multispectral image has been performed by means of a VAE [23] that parameterizes the latent space of Gaussian distribution parameters. VAEs have been also used to create superresolution images as in [24], where a model is proposed to encode low-resolution images in a dense latent space vector that can be decoded for target high resolution image denoising. The blur image problem using VAE is tackled in [25] by adding a conditional sampling mechanism that narrows down the latent space, making it possible to reconstruct high resolution images. Moreover, in [26], the authors propose a flexible autoencoder model able to adapt to varying data patterns with time. By importing the VAE concept from image processing, several papers have used these models to improve RS results. For instance, denoising and variational autoencoders are tested in [27], where the authors reported the superiority of the VAE option against other models, or in [28], where variational autoencoders are combined with social information to improve the quality of the recommendations.

2.2 Our proposal: Deep variational models

The aim of this paper is to propose a neural architecture that joins the best of the DeepMF and NCF models with the VAE concept. This novel models will be called, respectively, Variational Deep Matrix Factorization (VDeepMF) and Variational Neural Collaborative Filtering (VNCF). In contrast with the autoencoder and Generative Adversarial Network (GAN) approaches in the CF field [16, 29, 30], we shall not use the generative decoder stage and we maintain the regression output layer presented in the DeepMF and the NCF models. The main advantage in the use of the VAE operation is the robustness that it confers to the latent representation. This robustness can be seen by observing Fig. 3. If we consider each dot drawn as a train sample representation in the latent space, then test samples are most likely to be correctly classified in the VAE model (right graph in Fig. 3) than being correctly classified in the regular autoencoder model (left graph in Fig. 3). In short, the variational approach stochastically 'spreads' the samples in the latent space, improving the chances of classifying correctly the training samples.

In our proposed RS CF scenario, we expect that rating values can be better predicted when a variational latent space has been learnt, because this space covers a wider, more robust, and more representative latent area. Whereas with a traditional autoencoders each sample would be coded as a value in the latent space (white circle in Fig. 4), the VAE encodes the parameters of a multivariate distribution (e.g., mean and variance of both the blue and the orange Gaussian distributions in Fig. 4). From the learnt distribution parameters, random sampling is carried out to generate stochastic latent space values (gray circles in Fig. 4). Each epoch in the learning process generates a new set of latent space values. Once the proposed model has been trained, when a $\langle$user, item$\rangle$ tuple is presented to the model, the obtained latent space value (green circle in Fig. 4) can be better predicted in the VAE scenario than in the regular autoencoder scenario: the random sampled values (gray circles) of the enriched latent space will help to associate the predicted sample (green circle) with their

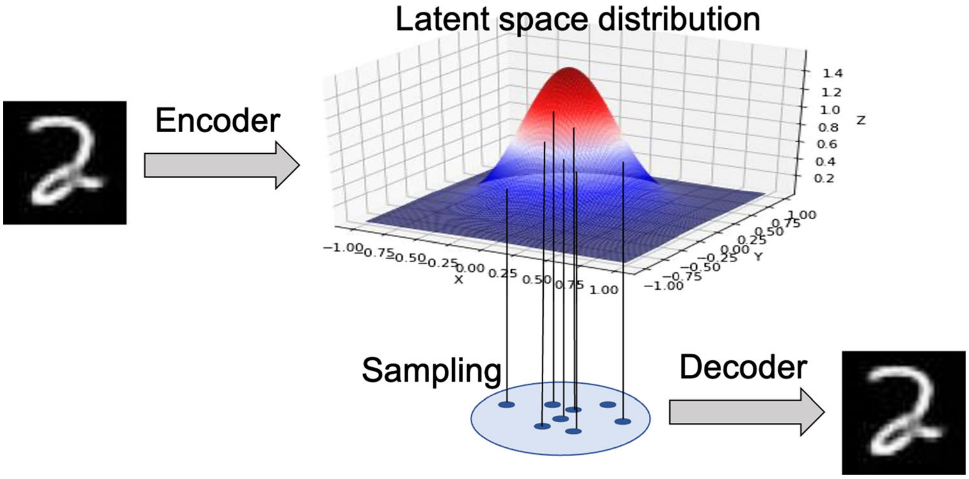


Fig. 2 Operation of a trained Variational Autoencoder (VAE) model. When a new sample is presented to the encoder stage (the handwritten digit ’2’ in this example), the model produces in the latent space a probability distribution. Typically, this distribution belongs to a known family (a multivariate normal distribution in this example), so its shape is determined by some numerical parameters (mean and standard deviation in our case). With this information, the decoder stage generates an instance by sampling this distribution (getting a slightly different digit ’2’ in this example). This introduces a stochastic component in the generation procedure that enriches the latent space and variability of the generative model

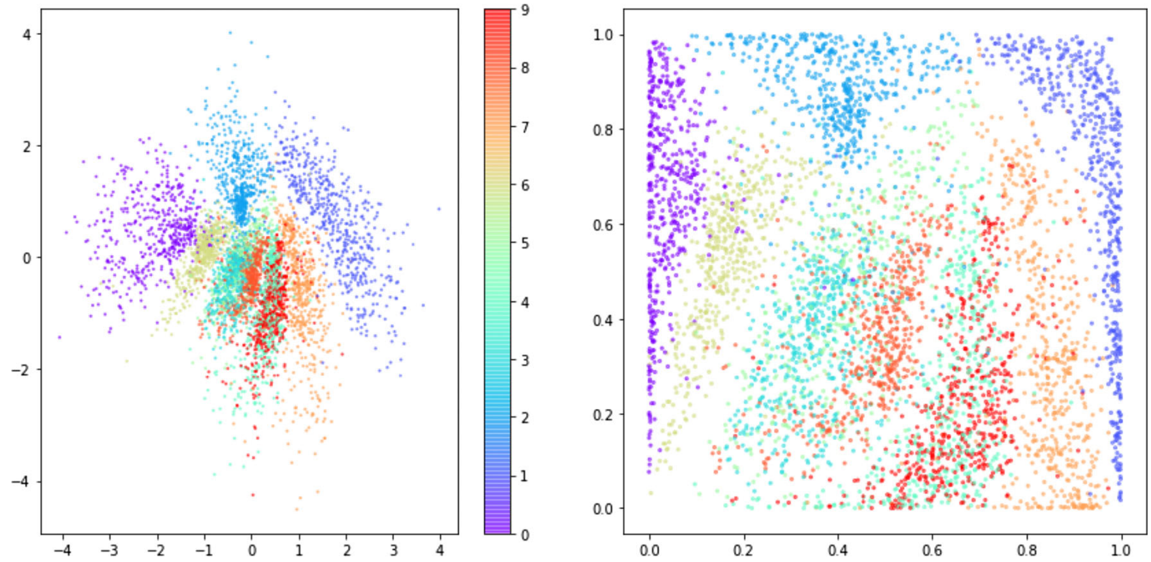


Fig. 3 Representation of a VAE latent space for the MNIST dataset (left side) and its cumulative normal distribution (right side)

associated training samples (white circle), making the prediction process much more robust and accurate.

From the above explanations, the VAE operation can be defined following Fig. 2 in its ‘Variational layers’ stage: first, two dense layers code the normal distribution parameters that set the mean and variance of the latent factors. In the CF scenario, two dense layers are arranged to code the normal distribution parameters of the items, and two other different dense layers are used to code the normal distribution parameters of the users. This variational approach regularizes the latent factors and makes it possible to reach the explained completeness and continuity goals. Once the distribution parameter layers are regularized, it is necessary to obtain a single latent factor point to code each user or item in the dataset; that is, for each user and item in the input of the model we need to combine its mean and variance. A normal random function is used to generate the latent factor point, coding the item (or the user) in the model input. Then, each latent factor point is obtained by combining: the normal random value, its item mean (or its user mean) and its item variance (or its user variance). This operation is usually performed using a neural ‘Lambda’ layer. Each Lambda layer result can be seen as a regularized version of the DeepMF nonvariational

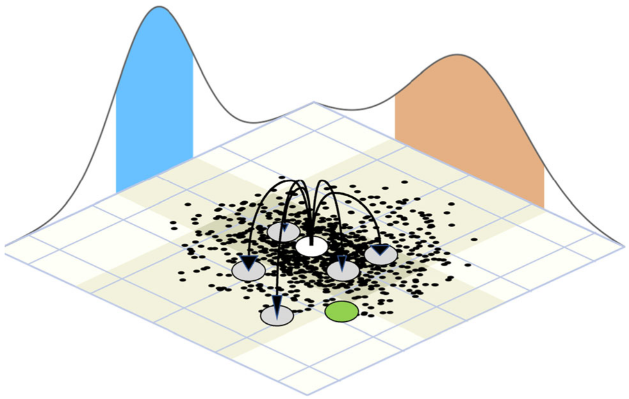


Fig. 4 Latent space representation of the proposed variational model. From the learnt means and variances of the multivariate Gaussian distribution, a random sampling process is run to spread the latent space sample values (gray circles) that will help to accurately predict the unknown sample rating values (green circle)

approach. Finally, we obtain the prediction of the rating of the user to the item by combining the 'Lambda' user and item factors using a dot product. In short, our variational approach incorporates the following substages: (1) Converting the input embedding factors to normal distribution values; and thus, making a regularization to generate continuous and complete latent factor codes, (2) Combining the normal distribution latent factor codes to obtain single latent factor values, and (3) Predicting ratings by making the dot product of the regularized latent factor values.

2.3 VAEs for recommender systems

Current CF-based variational autoencoders usually obtain raw augmented data. One strategy is to synthetize ratings from user to items or generated relevant versus not relevant votes from users to items [16, 27, 31], and another approach is to pre-train a VAE model to map data vectors into the latent space, an idea that has been intensively studied in several variants [32–36].

In any case, these strategies force practitioners to sequentially run two separated models: the generative model (GAN or VAE) that provides augmented data, and the regression CF model that makes predictions and recommendations. This approach presents three main drawbacks: (1) complexity, as two separate models are necessary, (2) large time consumption, and (3) sparsity management. As we will explain deeper in the following section, our proposed model does not generate raw augmented data. On the contrary, its innovation is based on the use of a single model to internally manage both augmentation and prediction aims. Particularly significant is the way in which the proposed model addresses the sparsity problem: we do not make augmentation on the sparse raw data (ratings cast from users to item), but an internal 'augmentation' process in the dense latent space of the model (Figs. 3 and 4). Each sample that is randomly generated from the latent space feeds the model regression layers. Thereby, we propose a model that first generates stochastic variational samples in a dense latent space, and then, these generated samples act as inputs of the regression stage of the model.

To test these ideas, the hypothesis considered in this paper is that the augmented samples will be more accurate and effective if they are generated in an inner and dense latent space rather than in a very sparse input space. It is important to realize that enriching the inner latent space can improve the recommendation results, but it also injects noise to the latent space that may potentially worsen the results. It is expected that the proposed approach will work better with poor latent spaces, whereas when it is applied to rich spaces, the spurious entropy added by the variational stage could worsen recommendations. Thus, medium-size CF datasets, or large and complex ones are better candidates to improve their results when the variational proposal is applied, whereas large datasets with predictable data distributions will probably not benefit from the noise injection of the variational architecture.

3 Proposed model

The proposed neural architecture will mix the VAE and the DeepMF (or the NCF) models. From the VAE, we take the encoder stage and its variational process, and from the DeepMF or the NCF model, we use its regression layers. This is an innovative approach in the RS field, since the VAE and GAN neural networks have only been used as a posteriori stage to make data augmentation, i.e., to obtain enriched input datasets to feed the CF DeepMF or NCF models. Hence, the traditional approach needs to separately train two models, first the VAE and then the DeepMF/NCF networks. As discussed in Sect. 2.3, these combined solutions present important disadvantages in terms of model complexity, time consumption and poor sparsity management.

In sharp contrast, our proposed approach efficiently joins the VAE and the Deep CF regression concepts to obtain improved predictions with a single training process. In the learning stage, the training samples feed the model (left hand side of Fig. 5). Each training sample consists of the tuple ⟨user, item, rating⟩ (rating casted by the user to the item). In the DeepMF/NCF architecture, each user is represented by his/her vector of voted ratings, and each item is represented by its vector of received ratings. The model learns the ratings (third element in the tuples) casted by the users to the items (first and second elements in the

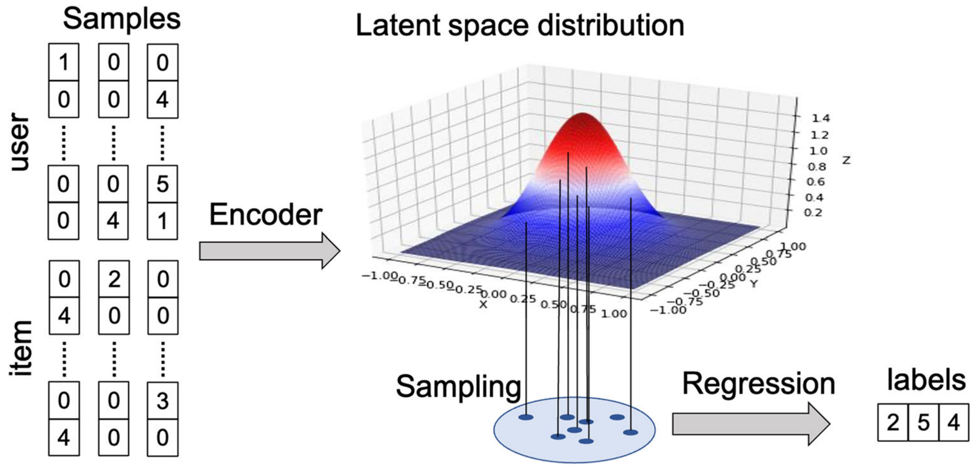


Fig. 5 Proposed VDeepMF/NCF approach. CF samples are encoded in the latent space by means of a variational process and then predictions are obtained by using a regression neural network

tuples). In other words, the ratings are outputs of the neural network (right hand side of Fig. 5).

Thanks to this architecture, the variational stage is naturally embedded into the model, so it can be flexibly used to inject variability into the samples. It is worth mentioning that this stage is also trained simultaneously to the predictive part of the model, mutually influencing each other, which we expect will lead to better results than a simple separate learning.

3.1 Formalization of the model

The architectural details of the proposed models are shown in Fig. 6. For simplicity, only the Variational Deep Matrix Factorization (VDeepMF) architecture is shown in this figure. The corresponding model for NCF, named Variational Neural Collaborative Filtering (VNCF), is analogous to the VDeepMF one: it has the same '`Embedding`' and '`Variational`' layers and we should only replace the '`Dot`' layer of DeepMF by a '`Concatenate`' layer followed by a MLP.

To fix the notation, let us suppose that our dataset contains U users and I items. In general, the aim of any deep learning model for CF-based prediction is to train a (stochastic) neural network that implements a function

$$h : \mathbb{R}^U \times \mathbb{R}^I \to \mathbb{R}.$$

This function h operates as follows. Let us codify the u-th user of the dataset (resp. the i-th item) using one-hot-encoding as the u-th canonical basis vector $\mathbf{e}_u$ (resp. the i-th canonical basis vector $\mathbf{e}_i$). Then, we have that the outcome of the model is the following

$$h(\mathbf{e}_u, \mathbf{e}_i) \in \mathbb{R} \quad = \text{Prediction of the score that the } u\text{0th user would assign to the} i\text{0th item.}$$

To train this function h, in the learning phase the neural network is fed with a set

$$\mathtt{X} = \{\langle u, i, r \rangle\}$$

of training tuples $\langle u, i, r \rangle$ of a user u that rated item i with a score r. The function h is trained to minimize the error

$$\mathcal{E}(h) = \sum_{\langle u,i,r \rangle \in \mathtt{X}} \delta(h(\mathbf{e}_u, \mathbf{e}_i), r). \tag{1}$$

Here, $\delta : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is any metric on $\mathbb{R}$. Typical choices are the so-called Mean Squared Error (MSE) and Mean Absolute Error (MAE), respectively, given by

$$\delta_{\mathrm{MSE}}(x, y) = (x - y)^2, \qquad \delta_{MAE}(x, y) = |x - y|.$$

Our proposal for the VDeepMF consist on decomposing h has a combination of a '`Embedding`', followed by a '`Variational`' stage and a final '`Dot`' layer, as shown in Fig. 6), that is

$$h = \mathtt{Dot} \circ \mathtt{Variational} \circ \mathtt{Embedding}.$$

Notice that, at the end of the day, h is a deep leaning model with novel customary layers designed for the RS problem. In this way, h can be trained to reduce the error $\mathcal{E}(h)$ of Eq. (1) with the standard Deep Learning (DL) methods, such as the backpropagation algorithm.

3.2 The embedding layer

The first '`Embedding`' layer (left hand side of Fig. 6) is borrowed from the Natural Language Processing (NLP) [13]. The idea is that this layers provides a fast translation of users and items into their respective

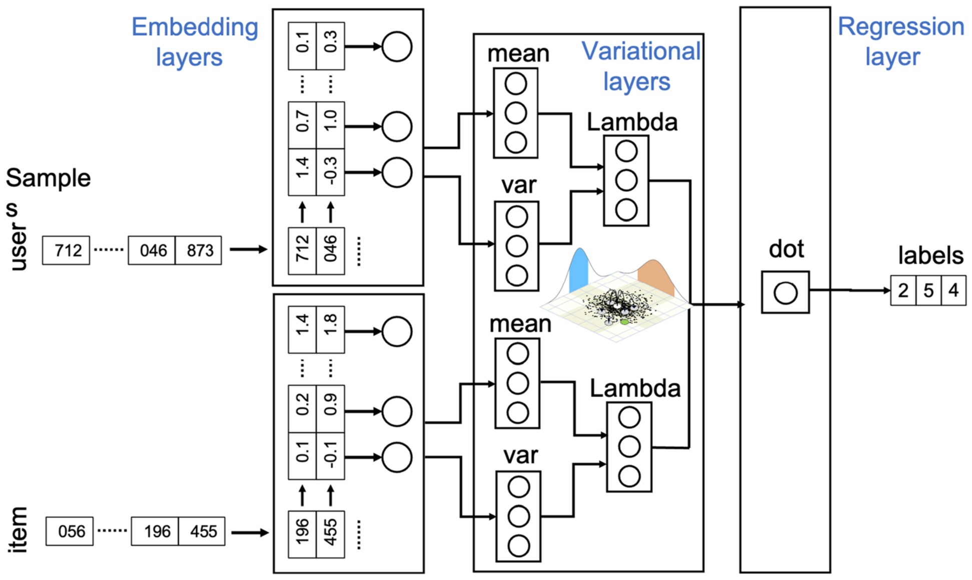


Fig. 6 Proposed VDeepMF architecture. The NCF architecture will have identical 'Embedding' and 'Variational' layers to the VDeepMF one; it will just replace the 'Dot' layer for a 'Concatenate' layer, followed by an MLP

representations in the latent spaces. To be precise, this layer implements a linear map

$$\texttt{Embedding} : \mathbb{R}^U \times \mathbb{R}^I \to \mathbb{R}^L \times \mathbb{R}^L,$$

that maps a pair $(\mathbf{e}_u, \mathbf{e}_i)$ into a pair of dense vectors $(v_u, w_i) \in \mathbb{R}^L \times \mathbb{R}^L$ that represents the u-th user and the i-th item, being $L > 0$ the dimension of the representations.

For our purpose, we implement the 'Embedding' layer as a regular MLP dense layer, in sharp contrast with other approaches in NLP such as word2vec [37, 38], GloVe [39] or ELMo [40], among others. The reason is that these later approaches seek to find an embedding preserving some metric information of the words, typically, the likelihood of finding two words together or their semantic similarity. However, in our case, since we perform context-free CF prediction, no a priori information about the similarity between users or items is available. Indeed, this is precisely the ultimate goal of the RS: to find an appropriate representations of these users and items in the latent space. For this reason, we decided not to add any extra mechanism that could bias this training process, so the 'Embedding' layer will act as a regular dense layer to be trained in parallel during the learning process.

Finally, we would like to point out that, even though from a conceptual point of view the 'Embedding' layer is just a dense layer, to save time and space, these 'Embedding' layers are typically implemented through lookup tables. In this way, instead of feeding the network with the one-hot encoding of the user u (resp. the item i), we input it via its ID as user (resp. as item). The lookup table efficiently recovers the u-th (resp. i-th) column of the embedding matrix that contains v_u (resp. w_i) so that the translation can be conducted in a more efficient way than with a standard MLP layer by exploiting the sparsity of the input.

3.3 The variational layer

The variational process is carried out by the 'Variational' stage (labeled as 'variational layers' at the middle of Fig. 6). This is the core of our proposed model.

From the latent space representation $(v_u, w_i) \in \mathbb{R}^L \times \mathbb{R}^L$ of the u-th user and the i-th item, two separated dense layers return the mean and variance parameters of two Gaussian multivariate distribution. In this way, if fix a latent space dimension $K > 0$, the first part of this 'Variational' stage (left part of the middle rectangle of Fig. 6) computes a map

$$\mathcal{S}(v_u, w_i) = (\mu_1(v_u), \sigma_1^2(v_u), \mu_2(w_i), \sigma^2(w_i)) \in \mathbb{R}^{4K}.$$

The outputs $\mu_1(v_u), \mu_2(w_i)$ of $\mathcal{S}$ will be interpreted as the means of two Gaussian distributions to the user and the item, respectively, whereas $\sigma_1^2(v_u), \sigma^2(w_i)$ will represent variance.

The second part of the 'Variational' stage (left right of the middle rectangle of Fig. 6) is ruled by a pair of random vectors $(P_{\mu_1(v_u),\sigma_1^2(v_u)}, Q_{\mu_2(w_i),\sigma^2(w_i)})$ where

$$P \sim \mathcal{N}(\mu_1(v_u), \mathrm{diag}\, \sigma_1^2(v_u)),$$
$$Q \sim \mathcal{N}(\mu_2(w_u), \mathrm{diag}\, \sigma_2^2(w_i)).$$

Here, $\mathcal{N}(\mu, \Sigma)$ denotes a K-dimensional multivariate normal distribution of mean vector μ and diagonal covariance matrix Σ, i.e., whose probability density function is

$$f(s) = \frac{1}{\sqrt{(2\pi)^K \det \Sigma}} \exp\left(-\frac{1}{2}(s-\mu)^t \Sigma^{-1}(s-\mu)\right).$$

Notice that, in our case, the covariance matrix is always diagonal.

In this setting, the task of the ‘Variational’ stage is just to sample P and Q. In this manner

$$\texttt{Variational}(v_u, w_i) = (p, q) \in \mathbb{R}^K \times \mathbb{R}^K,$$

where p is a sample of $P = P(\mathcal{S}(v_u, w_i)) \sim \mathcal{N}(\mu_1(v_u), \operatorname{diag}\sigma_1^2(v_u))$ and q is a sample of $Q = Q(\mathcal{S}(v_u, w_i)) \sim \mathcal{N}(\mu_2(w_u), \operatorname{diag}\sigma_2^2(w_i))$. This pair represents the stochastic latent representations associated with (v_u, w_i).

3.4 The join layer

This is the only layer that depends on the particular choice of the architecture. In the case of the Variational Deep Matrix Factorization (VDeepMF) architecture, this final layer is a ‘Dot’ layer (labeled as ‘regression layer’ at right hand side of Fig. 6). It is just a linear layer that simply computes the dot product of the latent vectors p and q. Therefore

$$\texttt{Dot}(p, q) = p \cdot q.$$

In the case of VNCF, this simple layer is replaced by a fully connected MLP

$$\mathcal{H} : \mathbb{R}^K \to \mathbb{R}^K \to \mathbb{R}$$

that extracts the nonlinear relations from p and q.

Therefore, summarizing the process, the proposed VDeepMF model h computes

$$\begin{aligned} h(\mathbf{e}_u, \mathbf{e}_i) &= \texttt{Dot} \circ \texttt{Variational} \circ \texttt{Embedding}(\mathbf{e}_u, \mathbf{e}_i) \\ &= P_{\mu_1(v_u), \sigma_1^2(v_u)} \cdot Q_{\mu_2(w_i), \sigma^2(w_i)}. \end{aligned}$$

Analogously, the VNCF model returns the proposed VDeepMF model h computes

$$\begin{aligned} h(\mathbf{e}_u, \mathbf{e}_i) &= \mathcal{H} \circ \texttt{Variational} \circ \texttt{Embedding}(\mathbf{e}_u, \mathbf{e}_i) \\ &= \mathcal{H}(P_{\mu_1(v_u), \sigma_1^2(v_u)}, Q_{\mu_2(w_i), \sigma^2(w_i)}). \end{aligned}$$

In both cases, $h(\mathbf{e}_u, \mathbf{e}_i)$ is a random variable that, when sampled, returns a natural number that should be interpreted as the predicted rating by h for the user u regarding item i.

4 Empirical evaluation

In this section, we describe the empirical experiments carried out to evaluate the performance of the variational approach in the DeepMF and NCF models.

4.1 Experimental setup

The experimental evaluation has been performed over four different datasets to measure the performance of the proposed method over different environments. The selected datasets are: FilmTrust [41], an small dataset that contains the ratings of thousands of items to movies; MovieLens 1 M [42], the gold standard dataset in CF-based RS; MyAnimeList [43], a dataset extracted from Kaggle[1] that contains the ratings of thousands of users to anime comics; and Netflix [44], a popular dataset with hundred of millions ratings used in the Netflix Prize competition. Table 1 shows the main parameters of these datasets. The corpus of these datasets has been randomly splitted into training ratings (80% of the ratings) and test ratings (20% of the ratings).

The evaluation of the proposed method has been analyzed from three different points of view: the quality of the predictions [45], the quality of the recommendations [46], and the quality of the recommendation lists [47].

To measure the quality of the predictions, we have compared the real rating $r_{u,i}$ of an user u to an item i of the test split R^{test} with the predicted one, $\hat{r}_{u,i}$. These comparisons have been carried out in three ways: using the MAE as in Eq. (2), using the MSE as in Eq. (3) and computing the proportion of the explained variance R^2 as in Eq. (4). Notice that, in Eq. (4), $\bar{r}$ denotes the mean of the ratings contained in the test split.

$$\text{MAE} = \frac{1}{\#R^{test}} \sum_{\langle u,i\rangle \in R^{test}} \mid r_{u,i} - \hat{r}_{u,i} \mid, \tag{2}$$

$$\text{MSE} = \frac{1}{\#R^{test}} \sum_{\langle u,i\rangle \in R^{test}} \left(r_{u,i} - \hat{r}_{u,i}\right)^2, \tag{3}$$

$$R^2 = 1 - \frac{\sum_{\langle u,i\rangle \in R^{test}} \left(r_{u,i} - \hat{r}_{u,i}\right)^2}{\sum_{\langle u,i\rangle \in R^{test}} \left(r_{u,i} - \bar{r}\right)^2}. \tag{4}$$

To measure the quality of the recommendations, we have analyzed the impact of the top N recommended items to the user u, collected in the list T_u^N. Using precision Eq. (5), we measure the proportion of relevant recommendations (i.e., the user rated the item with a rated equal or greater than a threshold θ) among the top N. Here, U denotes the set of user in the test split. In a similar vein, using recall Eq. (6), we measure the proportion of the test items rated by the user u, R_u^{test}, that were relevant to him or her and were included into the recommended items T_u^N. For the conducted experiments, the used thresholds are $\theta = 3$ for FilmTrust, $\theta = 4$ for MovieLens and Netflix, and $\theta = 8$ for MyAnimeList. These thresholds were chosen in agreement

[1] www.kaggle.com.

Table 1 Main parameters of the datasets used in the experiments

Dataset	N users	N items	N ratings	Scores	Sparsity (%)
FilmTrust	1508	2071	35,494	0.5 to 4.0	98.86
MovieLens	6040	3706	1,000,209	1 to 5	95.53
MyAnimeList	69,600	9927	6,337,234	1 to 10	99.08
Netflix	480,189	17,770	100,480,507	1 to 5	98.82

with the results of [48], where it was shown that these values represent a fair trade-off between provided coverage of the dataset and prediction accuracy.

$$\text{Precision} = \frac{1}{\#U}\sum_{u\in U}\frac{\{i \in T_u^N \mid r_{u,i} \geq \theta\}}{N}, \tag{5}$$

$$\text{Recall} = \frac{1}{\#U}\sum_{u\in U}\frac{\{i \in T_u^N \mid r_{u,i} \geq \theta\}}{\{i \in R_u^{test} \mid r_{u,i} \geq \theta\}}. \tag{6}$$

Additionally, we have measure the quality of the recommendations using the harmonic mean of the precision and the recall using F1 score Eq. (7).

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{7}$$

However, evaluating the quality of recommendations based solely on user ratings provides a biased view of the recommender's performance. Therefore, we have also determined the novelty Eq. (8) of the recommendations. Novelty [49] is calculated by assigning more weight to those items that have received fewer ratings. In other words, the novelty of an item is inversely proportional to the number of ratings received for an item ($\#R_i$) with respect to the total number of votes in the recommender system ($\#R$).

Table 2 Quality of the predictions

	FilmTrust	MovieLens	MyAnimeList	Netflix
(a) Mean Absolute Error. The lower the better.				
VDeepMF	0.6567.	**0.6827**	**0.8722**	0.7176
DeepMF	0.7957	0.6993	0.9044	**0.6830**
VNCF	0.6410	0.7263	0.9281	0.7474
NCF	**0.6361**	0.7021	0.8874	0.6903
(b) Mean Squared Error. The lower the better.				
VDeepMF	0.7324	**0.7529**	**1.3453**	0.8581
DeepMF	1.2046	0.7939	1.5017	0.7789
VNCF	0.6844	0.8179	1.4605	0.8952
NCF	**0.6743**	0.7908	1.3674.	**0.7774**
(c) R^2 score. The higher the better.				
VDeepMF	0.1438	**0.3980**	**0.4549**	0.2711
DeepMF	− 0.4082	0.3652	0.3916	0.3384
VNCF	0.1999	0.3460	0.4083	0.2396
NCF	**0.2118**	0.3677	0.4460.	**0.3397**

The best results for each quality measure are highlighted in bold

$$\text{Novelty} = \frac{1}{\#U}\sum_{u\in U}\frac{\sum_{i\in T_u^N} - \log_2\left(\frac{\#R_i}{\#R}\right)}{N}. \tag{8}$$

Finally, to measure the quality of the recommendation lists, we use the normalized Discounted Cumulative Gain (nDCG). Suppose that the recommendation list of the user u, T_u^N, is sorted decreasingly so that the items predicted as more relevant are placed in the first positions. Given $i \in T_u^N$, let $\text{pos}_{T_u^N}(i)$ be the position of the item i in the recommendation list. Analogously, suppose that the real top N recommendations to user u, R_u^N, as sorted decreasingly and denote by $\text{pos}_{R_u^N}(i)$ the position of the item $i \in R_u^N$ in the list. In this setting, the Discounted Cumulative Gain (DCG) and the Ideal Cumulative Gain (IDCG) of the user $u \in U$ are defined as in Eq. (9).

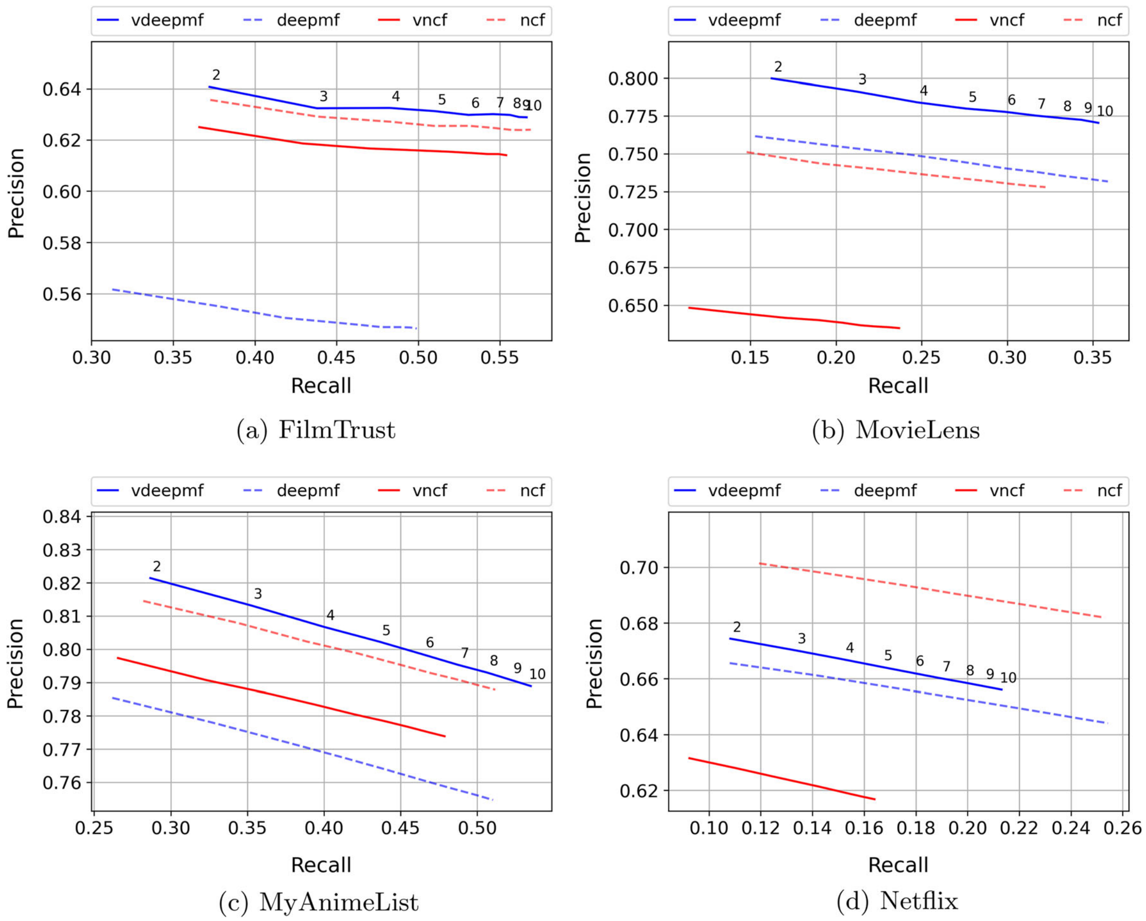
$$\begin{aligned} \text{DCG}_u &= \sum_{i\in T_u^N}\frac{2^{r_{u,i}} - 1}{\log_2\left(\text{pos}_{T_u^N}(i) + 1\right)}, \\ \text{IDCG}_u &= \sum_{i\in R_u^N}\frac{2^{r_{u,i}} - 1}{\log_2\left(\text{pos}_{R_u^N}(i) + 1\right)}. \end{aligned} \tag{9}$$

In this way, nDCG is given by the mean of the ratio between DCG and IDCG as in Eq. (10).

$$\text{nDCG} = \frac{1}{\#U}\sum_{u\in U}\frac{\text{DCG}_u}{\text{IDCG}_u}. \tag{10}$$

Due to the stochastic nature of the variational embedded space of the proposed method, the test predictions used to evaluate the proposed method have been computed as the average of the 10 predictions performed for each pair of user u and item i.

Overall, the proposed variational architecture adequately improves simple models such as the DeepMF one, approaching their results to larger models such as the NCF. This tendency can be observed in both predictions and recommendation quality measures. Additionally, shorter running times are needed to train the proposed variational approach compared to baselines. This is the expected behavior in the hypothesis of the paper, but a remarkable constraint must be considered: the variational stage works particularly well when applied to not too large datasets, whereas using large datasets, the variational approach could not be necessary. The key idea is the ability of the



(a) FilmTrust

(b) MovieLens

(c) MyAnimeList

(d) Netflix

Fig. 7 Quality of the recommendations measured by precision and recall. The higher the better

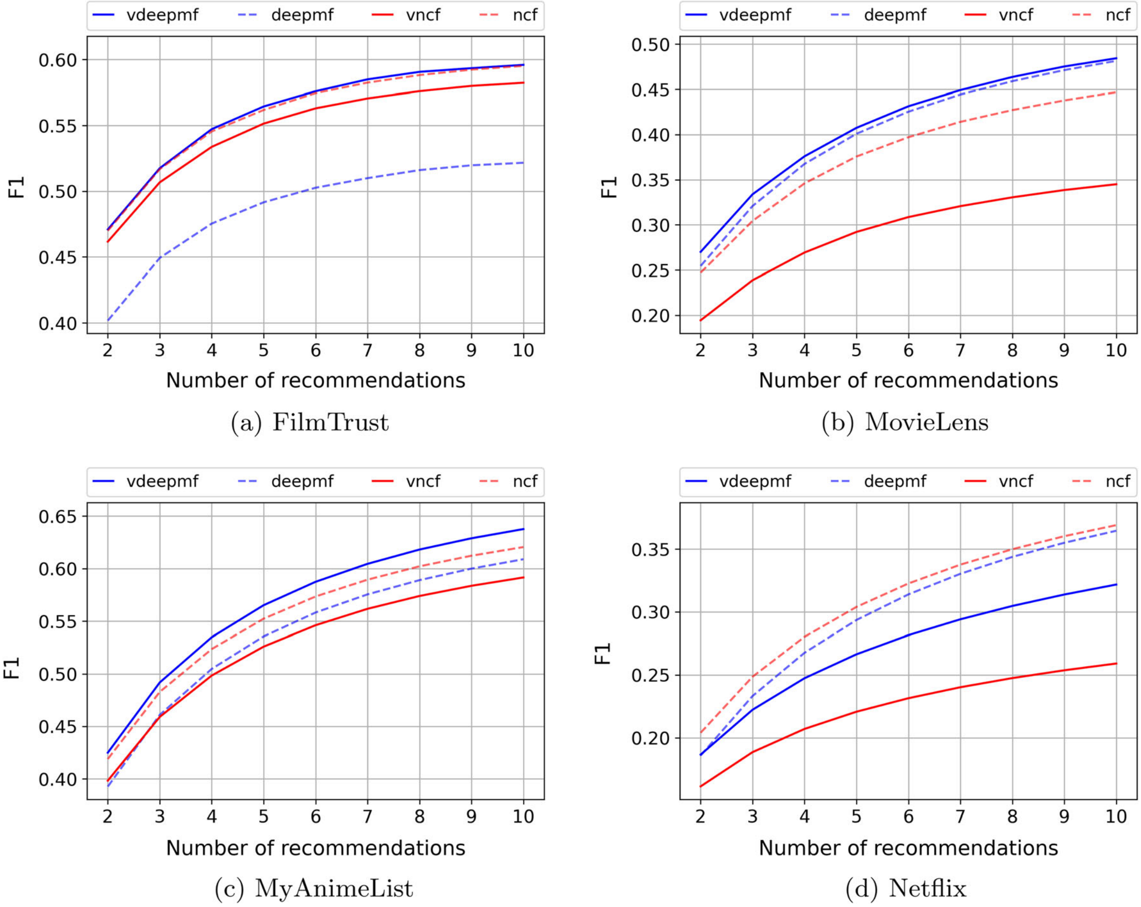
proposed model to deal with entropy: The variational stage increases entropy by generating stochastic latent factors and then enriching the latent space and making it more robust to the input sample variability. The intrinsic completeness and continuity properties of the VAE are the foundations on which the variational approach gets robust, continuous, and structured latent spaces. These enriched spaces provide the improved results obtained in the experiments.

4.2 Experimental results

Table 2 includes the quality of the predictions performed by the proposed model. Best values for each dataset are highlighted in bold. Table 2a contains the MAE Eq. (2), Table 2b contains the MSE Eq. (3), and Table 2c contains the R^2 score Eq. (4). We can observe that the proposed variational approach improves the prediction capability of DeepMF in all datasets except of Netflix and reports worse predictions when it is applied to NCF.

We justify these results by taking into account the features of the deep learning models used and the properties of each dataset. On the one hand, the larger the size of the dataset, the less necessary it is to enrich the votes with the proposed variational approach. In other words, when the dataset is small, the amount of Shannon entropy [50] that it contains might be quite limited. By using a variational method to generate new samples, we add some extra entropy that enriches the dataset, giving the chance to the regressive part of exploiting this extra data. However, large datasets usually present a large entropy in such a way that the regressive models can effectively extract very subtle information from them. In this setting, if we add a variational stage, instead of adding new relevant variability to the dataset, we only add noise that muddies the underlying patterns. For this reason, the variational approach is of no benefit in huge datasets like Netflix.

On the other hand, the NCF model is more complex than the DeepMF one, so data enrichment has less impact for complex models that are able to find more sophisticated relationships between data than simpler models. In fact,



(a) FilmTrust (b) MovieLens

(c) MyAnimeList (d) Netflix

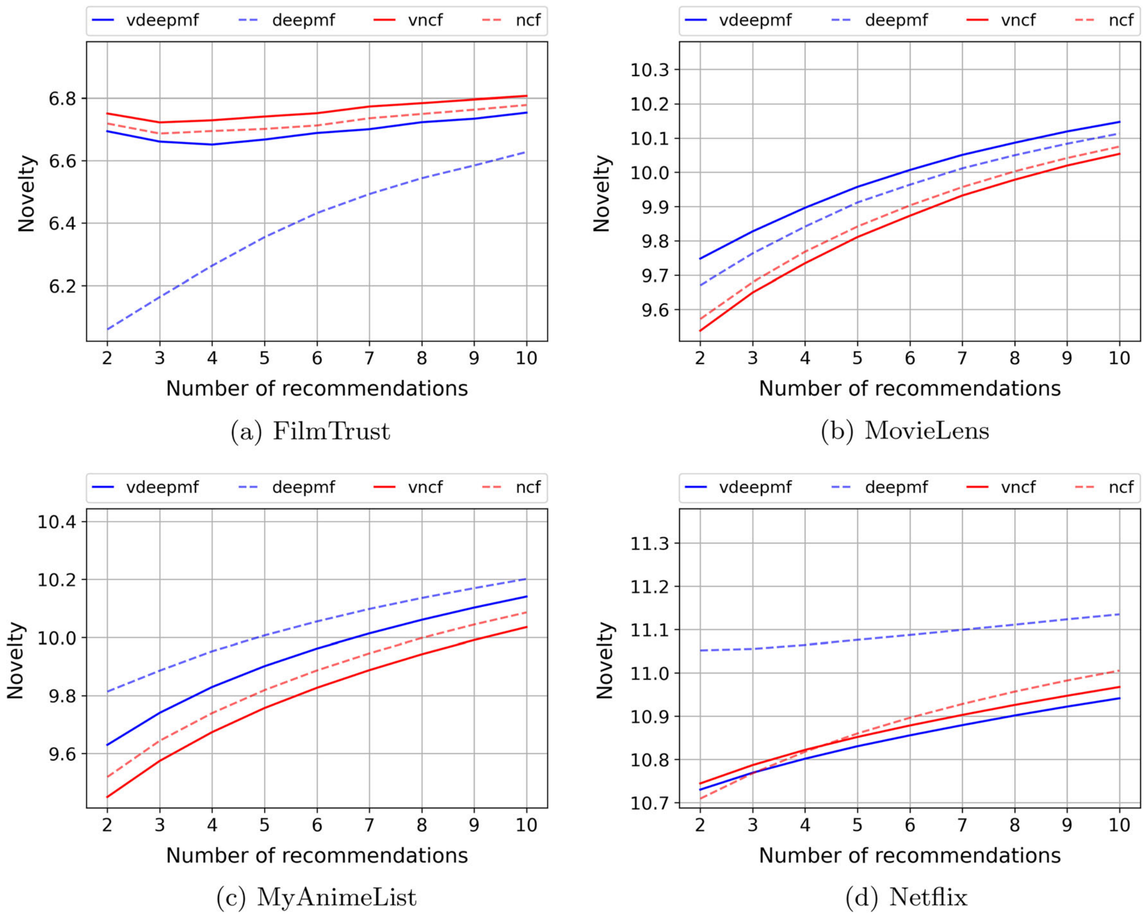
Fig. 8 Quality of the recommendations measured by F1. The higher the better

based on these results, we can assert that including the variational approach into a simple model such as DeepMF is equivalent to using a more complex model such as NCF.

Furthermore, Figs. 7 and 8 show the quality of the recommendations using precision Eq. (5), recall Eq. (6) and F1 Eq. (7) quality measures. In FilmTrust (Figs. 7a and 8a), MovieLens (Figs. 7b and 8b) and MyAnimeList (Figs. 7c and 8c), we can observe that the proposed variational approach reports a benefit for the DeepMF model and it worsens the results of the NCF model. In addition, VDeepMF model is the model that computes the best recommendations for these datasets. In contrast, in Netflix (Figs. 7d and 8d), the proposed variational approach does not improve the quality of the recommendations, with NCF being the model that provides the best recommendations for this dataset. These results are consistent with those analyzed when measuring the quality of the predictions. Consequently, it is evident that the proposed variational approach works adequately when the dataset is not too large and the model used is not too complex.

Fig. 9 contains the quality of recommendations regarding novelty Eq. (8). It is observed that, when the variational stage is added to the DeepMF model, a significant improvement of the novelty of the recommendations in small datasets is achieved. As the dataset becomes larger, the impact of the variational step is detrimental to the model. Thus, the variational stage has a positive impact on the FilmTrust (Fig. 9a) and MovieLens (Fig. 9b) datasets and a negative impact on the MyAnimeList (Fig. 9c) and Netflix (Fig. 9d) datasets. On the contrary, when a variational stage is added to the NCF model, its impact on novelty is practically zero regardless of the dataset size. This experiment, like the previous ones, reaffirms the conclusion that a variational step improves the results of simple models on small datasets.

In addition, Fig. 10 contains the nDCG results. From it, we can observe the same trends as those shown in previous experiments: in FilmTrust (Fig. 10a), the quality of the recommendation lists do not vary independently of whether the variational approach is used or not; in MovieLens (Fig. 10b) and MyAnimeList (Fig. 10c), the combination of the variational approach with simple modeling such as DeepMF, provides the best results; and in Netflix (Fig. 10d), the variational approach significantly worsens the quality of the recommendation lists.



(a) FilmTrust

(b) MovieLens

(c) MyAnimeList

(d) Netflix

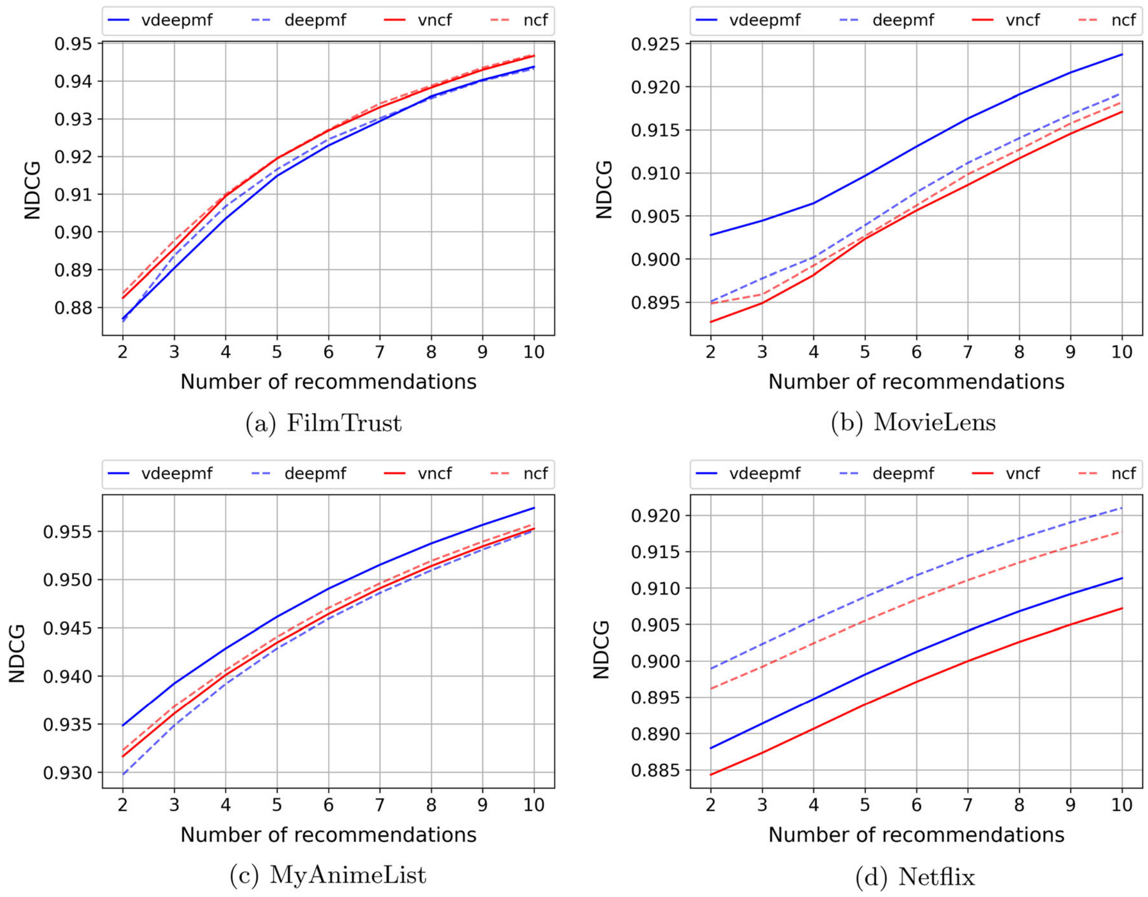
Fig. 9 Quality of the recommendations measured by novelty. The higher the better

5 Conclusions

In the latest trends, accuracy of RSs is being improved by using deep learning models such as deep matrix factorization and neural collaborative filtering. However, these models do not incorporate stochasticity in their design, unlike variational autoencoders do. Variational random sampling has been used to create augmented input raw data in the collaborative filtering context, but the inherent collaborative filtering data sparsity makes it difficult to get accurate results. This paper applies the variational concept not to generate augmented sparse data, but to create augmented samples in the latent space codified at the dense inner layers of the proposed neural network. This is an innovative approach trying to combine the potential of the variational stochasticity with the augmentation concept. Augmented samples are generated in the dense latent space of the neural network model. In this way, we avoid the sparse scenario in the variational process.

Observe that the proposed model in this paper also encodes the intrinsic locality of the users and items in the latent space. Recall that regular MF models capture the similarity of users and items in the latent space since predictions are constructed via inner product, a continuous function. In the same spirit, our variational models also preserve this locality since the output is still computed through a continuous function: the feed-forward neural network, a much more complicated function, but eventually continuous. Moreover, since the probability distributions representing each user and item in the latent space depend on continuous parameters (the mean and standard deviation of a Gaussian distribution), small variations in these parameters, corresponding to similar items or users, are also encoded as almost equal distributions, and thus, their samples tend to be also close in the distributional sense.

Thanks to these ideas, the results of the experimental analyses conducted in this paper show an important improvement when the proposed models are applied to middle-size representative collaborative filtering datasets, compared to the state-of-the-art baselines, testing both prediction and recommendation quality measures. In sharp contrast, testing on the huge Netflix dataset not only leads to no improvement, but the recommendation quality



(a) FilmTrust

(b) MovieLens

(c) MyAnimeList

(d) Netflix

Fig. 10 Quality of the recommendations lists measured by NDCG. The higher the better

Table 3 Fitting time using a Quadro RTX 8000

	FilmTrust	MovieLens	MyAnimeList	Netflix
VDeepMF	61s (15 epochs)	**601s (6 epochs)**	**7629s (9 epochs)**	12655s (3 epochs).
DeepMF	75s (25 epochs)	677s (10 epochs)	13217s (20 epochs)	15697s (4 epochs)
VNCF	**35s (7 epochs)**	1030s (9 epochs)	9945s (9 epochs)	**12650s (3 epochs)**
NCF	56s (15 epochs)	876s (10 epochs)	12111s (15 epochs)	16896s (4 epochs)

Best fitting times for each datased in bold

actually gets worse. In this manner, increasing the Shannon entropy in rich latent spaces causes that the negative effect of the introduced noise exceeds its benefit. Therefore, the proposed deep variational models should be applied to seek to a fair balance between their positive enrichment and their negative noise injection.

To emphasize this idea, in Table 3, we show the total time and epochs required by each model to be fitted to each dataset using a Quadro RTX 8000 GPU. Best time for each dataset is in bold. We can observe that including a variational layer to the model significantly reduces the required time for fitting. Variational models are able to generate Shannon entropy that is transferred to the regression stage, leading to a more effective training that requires fewer epochs to be fitted. Therefore, the fitting time needed to reach acceptable results is substantially lower.

The results presented in this work can be considered as generalizable, since they were analyzed in four representative and open CF datasets. Researchers can reproduce our experiments and easily create their own models by using the provided framework referenced in Sect. 3. The authors of this work are committed to reproducible science, so the code used in these experiments is publicly available.

Among the most promising future works, we propose the following: (1) introducing the variational process in the alternative inner layers of the relevant architectures in the collaborative filtering area, (2) screening the learning evolution in the training process, since it is faster than the classical models but it also requires early stopping in the training stage, (3) providing further theoretical explanations of the properties of the CF datasets, in terms of Shannon entropy or other statistical features, that ensure a good performance of the proposed models, (4) applying probabilistic deep learning models in the CF field to capture complex nonlinear stochastic relationships between random variables, and (5) testing the impact of the proposed concept when recommendations are made to groups of users.

Acknowledgements Á. G.-P. acknowledges the hospitality of the Department of Mathematics at Universidad Autónoma de Madrid where part of this work was developed. This work was partially supported by *Ministerio de Ciencia e Innovación* of Spain under the project PID2019-106493RB-I00 (DL-CEMG) and the *Comunidad de Madrid* under *Convenio Plurianual* with the Universidad Politécnica de Madrid in the actuation line of *Programa de Excelencia para el Profesorado Universitario*. The forth author has been partially supported by the *Madrid Government (Comunidad de Madrid – Spain)* under the Multiannual Agreement with the Universidad Complutense de Madrid in the line Research Incentive for Young PhDs, in the context of the V PRICIT (Regional Programme of Research and Technological Innovation) through the project PR27/21-029.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

Data availability The datasets analyzed during the current study are available in the repositories referred in the references [41–44].

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Beel J, Langer S, Genzmehr M, Gipp B, Breitinger C, Nürnberger A (2013) Research paper recommender system evaluation: a quantitative literature survey. In: Proceedings of the international workshop on reproducibility and replication in recommender systems evaluation, pp 15–22
2. Bobadilla J, González-Prieto Á, Ortega F, Lara-Cabrera R (2021) Deep learning feature selection to unhide demographic recommender systems factors. Neural Comput Appl 33(12):7291–7308
3. Deldjoo Y, Schedl M, Cremonesi P, Pasi G (2020) Recommender systems leveraging multimedia content. ACM Comput Surveys (CSUR) 53(5):1–38
4. Kulkarni S, Rodd SF (2020) Context aware recommendation systems: a review of the state of the art techniques. Comput Sci Rev 37:100255
5. Shokeen J, Rana C (2020) A study on features of social recommender systems. Artif Intell Rev 53(2):965–988
6. Bobadilla J, Alonso S, Hernando A (2020) Deep learning architecture for collaborative filtering recommender systems. Appl Sci 10(7):2441
7. Forouzandeh S, Berahmand K, Rostami M (2021) Presentation of a recommender system with ensemble learning and graph embedding: a case on movielens. Multimed Tools Appl 80(5):7805–7832
8. Çano E, Morisio M (2017) Hybrid recommender systems: a systematic literature review. Intell Data Anal 21(6):1487–1524
9. Mnih A, Salakhutdinov RR (2007) Probabilistic matrix factorization. Adv Neural Inf Process Syst 20:1257–1264
10. Févotte C, Idier J (2011) Algorithms for nonnegative matrix factorization with the β-divergence. Neural Comput 23(9):2421–2456
11. Hernando A, Bobadilla J, Ortega F (2016) A non negative matrix factorization for collaborative filtering recommender systems based on a bayesian probabilistic model. Knowl-Based Syst 97:188–202
12. Rendle S, Krichene W, Zhang L, Anderson J (2020) Neural collaborative filtering vs. matrix factorization revisited. In: Fourteenth ACM conference on recommender systems, pp 240–248
13. He X, Liao L, Zhang H, Nie L, Hu X, Chua T-S (2017) Neural collaborative filtering. In: Proceedings of the 26th international conference on world wide web, pp 173–182
14. Narang S, Taneja N (2018) Deep content-collaborative recommender system (dccrs). In: 2018 international conference on advances in computing, communication control and networking (ICACCCN), pp 110–116. IEEE
15. Bobadilla J, Lara-Cabrera R, González-Prieto Á, Ortega F (2021) Deepfair: deep learning for improving fairness in recommender systems. Int J Interact Multimed Artif Intell 6(6):86–94
16. Gao M, Zhang J, Yu J, Li J, Wen J, Xiong Q (2021) Recommender systems based on generative adversarial networks: a problem-driven perspective. Inf Sci 546:1166–1185
17. Xue H-J, Dai X, Zhang J, Huang S, Chen J (2017) Deep matrix factorization models for recommender systems. In: IJCAI, Melbourne, Australia, vol 17, pp 3203–3209
18. Wen J, She J, Li X, Mao H (2018) Visual background recommendation for dance performances using deep matrix factorization. ACM Trans Multimed Comput Commun Appl (TOMM) 14(1):1–19
19. Wan L, Xia F, Kong X, Hsu C-H, Huang R, Ma J (2020) Deep matrix factorization for trust-aware recommendation in social networks. IEEE Trans Network Sci Eng 8(1):511–528
20. Zou G, Chen J, He Q, Li K-C, Zhang B, Gan Y (2020) Ndmf: Neighborhood-integrated deep matrix factorization for service qos prediction. IEEE Trans Netw Serv Manage 17(4):2717–2730
21. Trigeorgis G, Bousmalis K, Zafeiriou S, Schuller BW (2016) A deep matrix factorization method for learning attribute representations. IEEE Trans Pattern Anal Mach Intell 39(3):417–429
22. Fan J, Cheng J (2018) Matrix completion by deep matrix factorization. Neural Netw 98:34–41
23. Liu X, Gherbi A, Wei Z, Li W, Cheriet M (2020) Multispectral image reconstruction from color images using enhanced variational autoencoder and generative adversarial network. IEEE Access 9:1666–1679

24. Liu Z-S, Siu W-C, Wang L-W, Li C-T, Cani M-P (2020) Unsupervised real image super-resolution via generative variational autoencoder. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pp 442–443
25. Liu Z-S, Siu W-C, Chan Y-L (2020) Photo-realistic image super-resolution via variational autoencoders. IEEE Trans Circ Syst Video Technol 31(4):1351–1365
26. Zhang S-s, Liu J-w, Zuo X, Lu R-k, Lian S-m (2021) Online deep learning based on auto-encoder. Appl Intell 51(8):5420–5439
27. Liang D, Krishnan RG, Hoffman MD, Jebara T (2018) Variational autoencoders for collaborative filtering. In: Proceedings of the 2018 world wide web conference, pp 689–698
28. Nisha C, Mohan A (2019) A social recommender system using deep architecture and network embedding. Appl Intell 49(5):1937–1953
29. Rama K, Kumar P, Bhasker B (2021) Deep autoencoders for feature learning with embeddings for recommendations: a novel recommender system solution. Neural Comput Appl 33(21):14167–14177
30. Tahmasebi H, Ravanmehr R, Mohamadrezaei R (2021) Social movie recommender system based on deep autoencoder network using twitter data. Neural Comput Appl 33(5):1607–1623
31. Li X, She J (2017) Collaborative variational autoencoder for recommender systems. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, pp 305–314
32. He M, Meng Q, Zhang S (2019) Collaborative additional variational autoencoder for top-n recommender systems. IEEE Access 7:5707–5713
33. Nahta R, Meena YK, Gopalani D, Chauhan GS (2021) Two-step hybrid collaborative filtering using deep variational bayesian autoencoders. Inf Sci 562:136–154
34. Shenbin I, Alekseev A, Tutubalina E, Malykh V, Nikolenko SI (2020) Recvae: a new variational autoencoder for top-n recommendations with implicit feedback. In: Proceedings of the 13th international conference on web search and data mining, pp. 528–536
35. Wang K, Xu L, Huang L, Wang C-D, Lai J-H (2019) Sddrs: stacked discriminative denoising auto-encoder based recommender system. Cogn Syst Res 55:164–174
36. Liu Y, Wang S, Khan MS, He J (2018) A novel deep hybrid recommender system based on auto-encoder with neural collaborative filtering. Big Data Mining Anal 1(3):211–221
37. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. Adv Neural Inform Process Syst **26**
38. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781
39. Pennington J, Socher R, Manning CD (2014) Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543
40. Peters M, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, Zettlemoyer L (1802) Deep contextualized word representations. 2018. arXiv preprint arXiv:1802.05365
41. Guo G, Zhang J, Yorke-Smith N (2013) A novel bayesian similarity measure for recommender systems. In: Proceedings of the 23rd international joint conference on artificial intelligence (IJCAI), pp 2619–2625
42. Harper FM, Konstan JA (2015) The movielens datasets: history and context. Acm Trans Interact Intell Syst (tiis) 5(4):1–19
43. Azathoth: MyAnimeList Dataset. https://www.kaggle.com/azathoth42/myanimelist. [Online; accessed 06-July-2021] (2018)
44. Bennett J, Lanning S et al (2007) The netflix prize. In: Proceedings of KDD Cup and Workshop, New York, NY, USA, vol 2007, p 35.
45. Bobadilla J, Hernando A, Ortega F, Bernal J (2011) A framework for collaborative filtering recommender systems. Expert Syst Appl 38(12):14609–14623
46. Herlocker J-L, Konstan J-A, Terveen L-G, Riedl J-T (2004) Evaluating collaborative filtering recommender systems. ACM Trans Inf Syst 22(1):5–53
47. Gunawardana A, Shani G (2015) Evaluating recommender systems. Handbook, Boston, MA
48. Ortega F, Lara-Cabrera R, González-Prieto Á, Bobadilla J (2021) Providing reliability in recommender systems through bernoulli matrix factorization. Inf Sci 553:110–128
49. Castells P, Vargas S, Wang J (2011) Novelty and diversity metrics for recommender systems: choice, discovery and relevance. In: Proceedings of the 33rd European conference on information retrieval (ECIR'11)
50. Shannon CE, Weaver W (1949) The mathematical theory of communication. University of Illinois Press, Urbana