applied sciences

MDPI

Article

Deep Variational Embedding Representation on Neural Collaborative Filtering Recommender Systems

Jesús Bobadilla, Jorge Dueñas, Abraham Gutiérrez * and Fernando Ortega

Departamento de Sistemas Informáticos, Escuela Técnica Superior de Ingeniería de Sistemas Informáticos, Universidad Politécnica de Madrid, 28031 Madrid, Spain; jesus.bobadilla@upm.es (J.B.); jorge.duenas.lerin@gmail.com (J.D.); fernando.ortega@upm.es (F.O.)

* Correspondence: abraham.gutierrez@upm.es

Featured Application: This paper proposes a new deep learning design to obtain accurate plots of RS information. The innovative model incorporates embedding layers of small (representable) sizes, variational layers to improve the latent space and to spread samples, and a Euclidean similarity measure to place samples according to the intuitive human interpretation of distances.

Abstract: Visual representation of user and item relations is an important issue in recommender systems. This is a big data task that helps to understand the underlying structure of the information, and it can be used by company managers and technical staff. Current collaborative filtering machine learning models are designed to improve prediction accuracy, not to provide suitable visual representations of data. This paper proposes a deep learning model specifically designed to display the existing relations among users, items, and both users and items. Making use of representative datasets, we show that by setting small embedding sizes of users and items, the recommender system accuracy remains nearly unchanged; it opens the door to the use of bidimensional and three-dimensional representations of users and items. The proposed neural model incorporates variational embedding stages to "unpack" (extend) embedding representations, which facilitates identifying individual samples. It also replaces the join layers in current models with a Lambda Euclidean layer that better catches the space representation of samples. The results show numerical and visual improvements when the proposed model is used compared to the baselines. The proposed model can be used to explain recommendations and to represent demographic features (gender, age, etc.) of samples.

Keywords: embedding; collaborative filtering; variational method; deep learning; recommender systems; recommendation explanations; data visual interpretation

Citation: Bobadilla, J.; Dueñas, J.; Gutiérrez, A.; Ortega, F. Deep Variational Embedding Representation on Neural Collaborative Filtering Recommender Systems. *Appl. Sci.* **2022**, *12*, 4168. https://doi.org/10.3390/app12094168

Academic Editor: Krzysztof Koszela

Received: 26 February 2022
Accepted: 19 April 2022
Published: 20 April 2022

1. Introduction

Recommender Systems (RS) [1] are machine learning-based personalization applications. They facilitate human/machine integration by providing accurate recommendations of items to users; mainly, items are products or services recommended to collaborative clients. Remarkable commercial companies that incorporate RS are Spotify, Netflix, TripAdvisor, and Amazon. RS can be classified according to their filtering strategy: demographic [2], content-based [3], context-aware [4], social [5], collaborative [6] and different ensembles [7]. Of the mentioned filtering approaches, the Collaborative Filtering (CF) is the most relevant since it returns the most accurate predictions and recommendations. The first CF implementations made use of the memory-based K-Nearest Neighbors (KNN) algorithm [8] due to its simplicity and because it conceptually fits with the recommendation task. Nevertheless, the KNN algorithm has some drawbacks when applied to CF RS: it is not accurate enough and it is not efficient, since successive executions are necessary to make successive recommendations. For these reasons, the KNN approach was replaced

by model-based methods, such as Matrix Factorization (MF) [9], non-Negative Matrix Factorization (NMF) [10] and Bayesian NMF (BNMF) [11].

MF models generate hidden factors for both users and items. Hidden factors can be considered as embedding representations. The rating prediction of each item to each user is obtained by just making the dot product of both (user and item) embeddings. MF models are not specifically designed to plot visual representations, but their embedding values can be processed to provide recommendation explanations [12] and to draw relationships [13]. Currently, MF models are being replaced by neural networks (NN) approaches, and consequently, hidden factors are replaced by neural embedding layers. The most relevant NN models in the CF area are DeepMF [14] and Neural Collaborative Filtering (NCF) [15]. DeepMF makes a deep learning implementation of the MF model; thus, it contains a user embedding layer, an item embedding layer, and a 'Dot' layer to join the preceding embeddings. NCF replaces the DeepMF 'Dot' layer with an MLP and eventually combines deep and shallow learning.

Embeddings are abstract, low-dimensional representations of information. There are a large number of fields where embeddings are used to encode data structures; network embeddings are largely applied to graphs [16], where they embed entities and relationships in low-dimensional spaces [17]. Gene sequences have been predicted from embeddings [18], and biomedical networks [19] have been evaluated as social graphs. Social communities are detected in the embedding-based Silhouette [20], via clustering of network node embeddings. Image applications are also a recurrent target for embedding processing, such as the person identification based on pose invariant embedding [21], image tag refinement through deep collaborative embedding [22], and handcrafted image retrieval [23] using supervised deep feature embedding. Natural language processing is also an area where tokens must be coded through different types of embedding models. Beyond the most known word2vec model, there are specific models, such as the convolution-deconvolution fusion word embedding [24], which makes a fusion of context and task information. Finally, fairness in RS is improved by means of an embedding-based combination of MF and deep learning models [25].

In the CF area, embedding layers have been used to implement autoencoders, such as the probabilistic autoencoder in [26] fed with the user-item data, the combination of stacked convolutional autoencoders, and stacked denoising autoencoders [27] to extract knowledge in RS. Context-aware information is coded using a deep learning autoencoder [28] that predicts scores and extracts features. However, the usual embedding-based architecture in the RS area exploits collaborative relationships, such as in [29] where they use embeddings to code user-item bipartite graphs for recommendation and representation learning. Relationships are also managed by means of low-dimensional dense embeddings learned from the sparse features in a wide and deep RS architecture [30]. A k-partite graph is used in [31] to characterize several types of information in recommendation tasks, and embeddings for different kinds of information are projected in the same latent space. A collaborative user network embedding has been proposed for social RS, where the cold start problem is addressed by combining MF and Bayesian personalized ranking [32]. A method to automatically set embedding sizes in RS [33] is based on the use of a reinforcement learning agent that adaptatively selects adequate sizes. Finally, internal embedding information is combined in [34] to obtain prediction and recommendation reliabilities.

Using deep learning variational approaches, we obtain wider, more representative, and more robust latent spaces and embedding representations. Neural autoencoders are, in certain cases, reinforced with a variational stage, building Variational Auto-Encoder (VAE) models. Mainly, VAEs are particularly applied to the image processing area, e.g., reconstructing images [35], creating super-resolution images by encoding low-resolution images in a dense latent space vector [36], and reducing blurring by adding a conditional sampling mechanism [37]. This paper proposes adding variational layers to the neural model suggested to improve the latent space where the embedding samples are located. It mimics the underlying VAE operative to obtain super-resolution images, reducing blurring,

and handling low-resolution samples: using VAE, the latent space is enriched, and samples are spread. Enriched embeddings are used in image processing to decode high-resolution images, unblurred images, etc., whereas we propose the use of enriched embeddings to improve the visual representation of RS information.

From the explained research, this paper proposes an innovative deep learning model that incorporates two embedding layers: one for code users and the other for code items. Both embeddings will have small sizes to make it possible to draw bi- or three-dimensional graphs of user and item samples. The accuracy loss caused by the small embedding sizes (two or three neurons each embedding) will be tested in the paper. The proposed model also incorporates a variational stage, designed to spread the latent space where item and user embeddings are represented. Both user and item embeddings will be followed by their own Gaussian variational layers whose parameter values are learned in the whole neural model. The expected results are accurate low-dimensional item and user graphs, where samples are spread in a latent space area and not 'compressed' in a reduced space region, making it easier to discriminate between adjacent samples. Finally, a 'Lambda' join layer is added to the model to implement the Euclidean distance between the embeddings of the items and the embeddings of the users. This layer replaces the 'Dot' product layer of the traditional DeepMF model or the MLP stage of the NCF model. The Euclidean Lambda layer's purpose is to keep near to related user or item embeddings and to keep far from nonrelated user or item embeddings, such as humans intuitively understand distances.

In short, this paper proposes a new deep learning design to obtain accurate plots of RS information. The innovative model incorporates embedding layers of small (representable) sizes, variational layers to improve the latent space and to spread samples, and a Euclidean similarity measure to place samples according to the intuitive human interpretation of distances. Experiments have been conducted using representative CF data sets to test the proposed model. The rest of the paper has been structured as follows: In Section 2 the proposed model is explained, Section 3 shows the experiments' design and results, Section 4 the results are discussed. Finally, Section 5 contains the main conclusions of the paper and future work.

2. Models and Methods

The current deep CF state of the art includes two remarkable neural models: DeepMF (Figure 1a) and NCF (Figure 1b). As shown in Figure 1, both DeepMF (Figure 1a) and NCF (Figure 1b) models provide two embedding layers: the first codes users and the second codes items. These are the embeddings that this paper addresses. DeepMF (Figure 1a) uses only a dot product to combine user and item factors, as well as the MF machine learning method. It is simple and it provides accurate results; nevertheless, it does not catch the nonlinear complex relations existing among users and items embedding. To solve the drawback mentioned, the NCF model (Figure 1b) incorporates an MLP that non-linearly combines factors of the user and the item, returning scalar regression values (predictions). Previously, a concatenate layer joined the embedding values of the user and the item and provided a single tensor flow to the MLP.

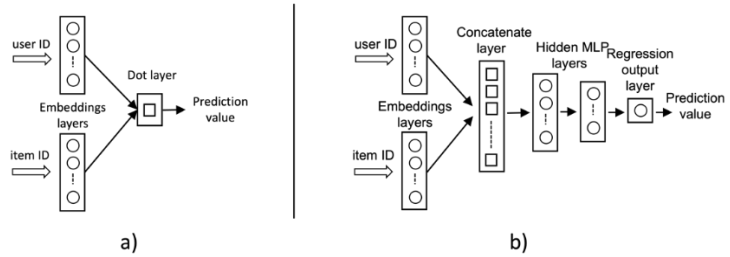

Figure 1. Deep matrix factorization (**a**) vs. neural collaborative filtering (**b**) recommender system models.

The embedding layers of the existing CF models are not designed for visual representations due to the following reasons: (1) they are vectors of excessive large sizes to be

visualized, (2) their values tend to cluster in small representation areas, and (3) The neural learning process does not consider visually understandable similarity measures (such as the Euclidean distance). To tackle the aforementioned drawbacks, we will provide three different contributions: (1) Testing the accuracy impact of reducing the embedding sizes just to two or three dimensions, (2) Expanding the embedding values representation by using the variational approach, and (3) Incorporating the Euclidean similarity measure in the deep neural model.

Contribution 1: In the CF field, it is particularly useful to visually represent users and items in such a way that clients, company managers, and technical staff can understand the existing relations among users, items, and between both users and items. This leads us to code users and items using only two or three dimensions. The key question here is: is it affordable for the accuracy we will lose in the process? As we will show in the next section, the answer is yes, tested datasets show little significant accuracy decrease.

Contribution 2: We borrow the variational method from the variational autoencoder field; they expand the embedding representation of samples, making it possible to improve clustering and classification, and to return a progressive morphing when needed. Figure 2 explains the variational approach, where each sample embedding (white circle) can be probabilistically located (grey circles) nearby (green circle) to its nonvariational fixed location (white circle). Variational methods are usually implemented by setting a Gaussian distribution in each embedding dimension. The defined set of parameters of the Gaussian distributions (blue and orange distributions) establishes the probabilistic area where the samples lay out. Our neural model specifically learns both the mean and variance of each Gaussian distribution.

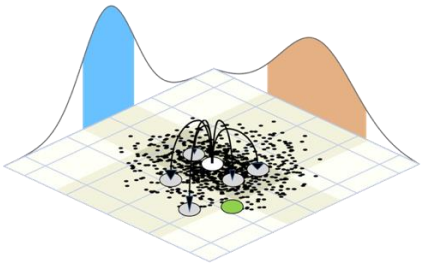

Figure 2. Representation of the variational method, where each sample embedding (white circle) is probabilistically placed (grey circles) at a nearby position (green circle).

As explained, the variational approach expands the area where the sample embeddings lay out. This is particularly adequate for our embedding representation goal since it will make it easier to visually catch our attention on the existing sample relations. As an example, in Figure 3 we show the variational result (embeddings) of the proposed model applied to the MNIST dataset, where samples have been stochastically spread to make the classification of the classes easier. Figure 3 left and right graphs show, respectively, the obtained latent space and its cumulative normal distribution. The cumulative normal distribution is frequently used to support generative tasks; in this case, it can be used to generate fake embeddings, and then to obtain fake samples (MNIST numbers). In the CF area, this opens the door to implementing augmentation data and to obtaining augmented RS datasets.

Contribution 3: Traditional DeepMF and NCF models implement, respectively, a dot layer and an MLP network (Figure 1). Both approaches (dot layer and MLP network) can be considered as similarity functions, and none of them are designed to arrange embedding representations in a visual disposition. Our proposed model replaces these functions with a visually convenient similarity measure: the Euclidean distance. It will set the embedding representation of samples in such a way that similar samples will be located at nearby locations. It is expected that what is gained in understanding the RS representation is not lost in the accuracy of the CF.

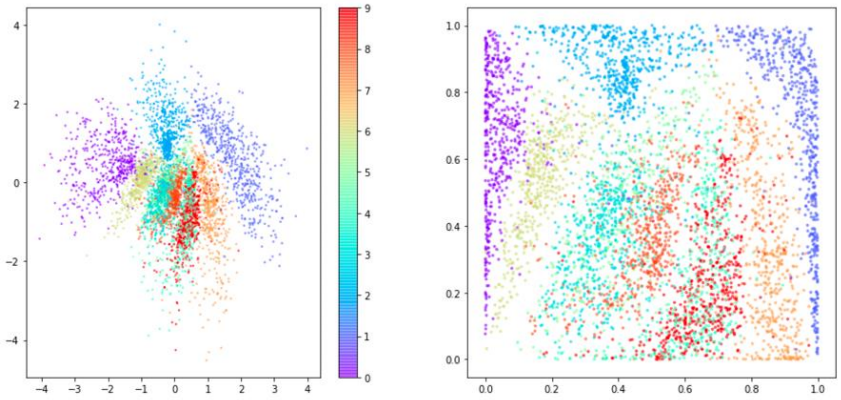

Figure 3. Representation of a VAE latent space for the MNIST dataset (left side) and its cumulative normal distribution (right side).

By combining the three mentioned contributions, we have designed the deep neural models shown in Figure 4. The user model (orange) and the item model (blue) are conceptually identical: their first stage "embedding layers" (bottom-left of Figure 4) is an embedding layer that maps user or item IDs to coded values. It is expected that users with similar behavior (similar casted votes) will be assigned similar embedding values. Same for items; items similarly voted will be coded in an equivalent way. Please note that an embedding size of two or three neurons is expected to adequately capture the diversity of the existing sets of users and items in the recommender system. In this case, we will be able to visually represent users and items by drawing graphs in two or three dimensions.

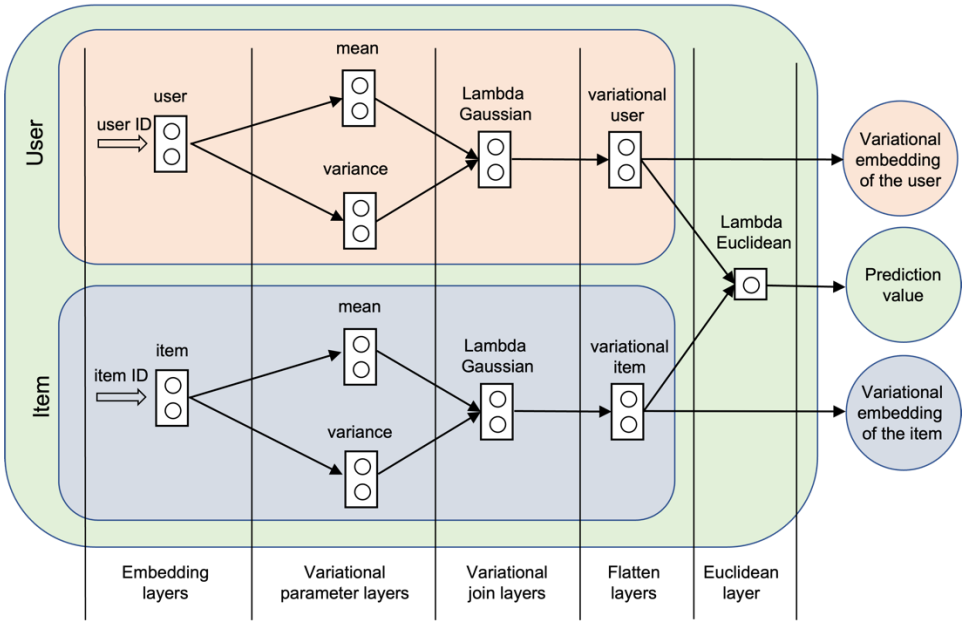

Figure 4. Proposed models: Green: collaborative filtering prediction; Orange: variational Euclidean model for users; Blue: variational Euclidean model for items.

The next stage of the proposed model: 'variational parameter layers', at the bottom of Figure 4, is responsible for learning the most adequate values of the Gaussian distributions that implement the variational behavior of our model (Figure 2). We split the user embedding into two separated tensor flows, implemented through both the 'mean' layer and the 'variance' layer, providing us with the mean and variance of each Gaussian distribution (two or three distributions, in our case). We also split the item embedding into two separated tensor flows. The user mean and the user variance layers must be

combined to obtain the user variational embedding (same for the item to obtain the item variational embedding). To implement the Figure 2 operation a 'Lambda' layer is used; this layer makes the variational sample generation. Each sample is stochastically generated attending to the Gaussian distributions that the model has learned; in the Figure 2 example, the generated sample has more probability to be spread through the orange Gaussian distribution than the blue one, since the orange one has a higher variance.

Please note that the variational sample has the same vector size as the user (or item) embedding, its 'mean' layer and its 'variance' layer. Finally, "Flatten" layers are added to the model to reshape data to unidimensional users and item vectors (of size 2 or 3).

The parallel user and item flows (orange and blue ones) provide both the user variational vector and the item variational vector ("Flatten layers" stage in Figure 4). Traditionally, they would be merged using a dot product or an MLP model (Figure 1). As explained, instead, we will incorporate a 'Lambda' layer that implements the Euclidean similarity measure ("Euclidean layer' in the bottom right of Figure 4). It will force the main model (the green one) to arrange variational user embeddings and variational item embeddings in a joined spatial area susceptible to being visually represented and easily understandable to humans. We use the regression model (green) to make training; once the model is trained, we can easily predict user variational embeddings from user IDs (orange model), and item variational embeddings from item IDs (blue model). It is important to stress that the proposed model is not designed to improve prediction accuracy (green model). The model is designed to obtain visually understandable representations of the users and the items embeddings (orange and blue inner models).

To get a deeper understanding of the proposed model, Code 1 provides the Keras/Python implementation of the model kernel.

Code 1. Keras/Python kernel of the proposed model.

```
def sampling(args):
z_mean, z_var = args
epsilon = K.random_normal(shape=(1, latent_dim), mean=0., stddev=1)
return z_mean + K.exp(z_var) * epsilon

def euclidean(args):
movie_v, user_v = args
return K.sqrt(K.sum(K.square(movie_v - user_v), axis=−1))

def variational_Euclidean(latent_dim):
user_input = Input(shape=[1])
user_embedding = Embedding(num_users + 1, latent_dim)(user_input)
user_embedding_mean = Dense(latent_dim) (user_embedding)
user_embedding_var = Dense(latent_dim) (user_embedding)
user_embedding_z = Lambda(sampling) ([user_embedding_mean, user_embedding_var])
user_vec = Flatten()(user_embedding_z)
movie_input = Input(shape=[1])
movie_embedding = Embedding(num_movies + 1, latent_dim)(movie_input)
movie_embedding_mean = Dense(latent_dim) (movie_embedding)
movie_embedding_var = Dense(latent_dim) (movie_embedding)
movie_embedding_z = Lambda(sampling) ([movie_embedding_mean,
movie_embedding_var], latent_dim)
movie_vec = Flatten()(movie_embedding_z)
similar = Lambda(euclidean)([movie_vec, user_vec])
var_eucl_pred = Model([user_input, movie_input], similar)
var_eucl_user = Model(user_input, user_vec)
var_eucl_item = Model(movie_input, movie_vec)
return var_eucl_pred, var_eucl_user, var_eucl_item
```

3. Experiments and Results

To run the designed experiments, we have chosen a set of open and representative CF databases. Table 1 shows the main parameter values of the selected datasets: MovieLens 100K [38], MovieLens1M [38], and a subset (Netflix*) of the Netflix database [39]. Please note the high number of Netflix* users compared to the MovieLens datasets. The chosen datasets have a similar structure, where their kernel is the CF information of ratings stored in files containing tuples: <user_id, item_id, rating>. Basically, they differ from each other in their sizes: number of users, items, and ratings. Additionally, the combination of the previous values determines the sparsity of the CF data. Please note that MovieLens 100K and MovieLens 1M not only differ in their number of ratings, but also in the number of users and items, and consequently in their sparsity (Table 1). Since the MovieLens 1M version is richer than the MovieLens 100K, its accuracy will be also better, as we will see in Table 2.

Table 1. Values of the main parameters of the tested datasets.

Dataset	#Users	#Items	#Ratings	Scores	Sparsity
Movielens 100K	943	1682	99,831	1 to 5	93.71
MovieLens 1M	6040	3706	911,031	1 to 5	95.94
Netflix*	23,012	1750	535,421	1 to 5	98.68

Table 2. Mean absolute error results using the proposed variational Euclidean method (embedding sizes = 2, 3, 5, 10 and the comparative accuracy obtained by setting an embedding size 2 versus an embedding size 10. The lower the MAE values, the better the result.

Dataset\Embedding Size	Mean Absolute Error (MAE)				Achieved Accuracy
	2	3	5	10	
Movielens 100K	0.7355	0.7368	0.7297	0.7213	98.04%
MovieLens 1M	0.6927	0.6875	0.6839	0.6801	98.15%
Netflix*	0.7260	0.7250	0.7248	0.7243	99.76%

From the aforementioned contributions, we will provide three different experiments to substantiate the proposed neural model: (1) CF quality impact by setting different embedding sizes, (2) numerical improvement of the proposed model versus the DeepMF baseline, (3) visual improvement of the proposed model versus the DeepMF baseline.

Experiment 1: This experiment tests the 'Contribution 1' assessment stated in the preceding section. As explained in the preceding section, it is necessary to test the RS accuracy when a bottleneck is set to the embedding layers. Since we need to visually represent embedding samples, we use embedding sizes: 2 (two-dimensional representation) or 3 (three-dimensional representation), whereas the usual implementation sizes range from 5 to 10. Our first experiment tests the accuracy loss when small embedding sizes are set. For each tested dataset (Table 1), we obtain the Mean Absolute Error (MAE) by setting embedding sizes = {2, 3, 5, 10}. Table 2 shows the MAE results, as well as the achieved accuracy percentage comparing the embedding sizes 2 and 10. As can be seen, very little accuracy is lost setting visualizable embedding sizes (2 and 3) compared to the usual sizes (5 to 10). Notably, only 2% of accuracy is lost in the worst-case scenario. The results in Table 2 open the door to visually represent the sample embeddings of items and users, knowing that the embedding values are meaningful to provide accurate CF predictions.

Experiment 2: This experiment numerically shows the improvement obtained by combining the three contributions stated in the preceding section. Once we have validated the adequacy of using visualizable embedding sizes, it is time to test the obtained improvement using the proposed approach. We will test visual improvement using the standard

intra-clustering quality measure equation that processes the distance of all the samples to their centroid. That is:

$$\text{intra-clustering} = \frac{1}{|S|} \sum_{x \in S} d(x, \overline{v}) \tag{1}$$

where:

$$d(x, \overline{v}) = \sqrt{\sum_{i \in \{1,\ldots,n\}} (x_i - \overline{v}_i)^2} \tag{2}$$

S is the set of samples, $\overline{v}$ is the S centroid and 'n' is the dimension size. Please note that whereas in the clustering field we look for low intra-clustering values, our embedding visualization aim is to spread embedding representations and to avoid them too being packed together. In this way, we will be able to better catch relations among samples. So, the higher our 'intra-clustering' quality measure, the better the results. In the CF embedding visualization field, we could call this quality measure an 'unpacking measure'. Table 3 shows the comparative results that test the non-variational dot product DeepMF baseline versus the proposed variational Euclidean model. Table 3 provides quality results for both user embeddings and item embeddings. As can be seen, representative improvements are obtained when the proposed model is used.

Table 3. Unpacking quality measure results (intra-clustering results from the quality measure defined in the 'Experiment 2') for both user and item embeddings. The higher the quality value, the better the result. "proposed" and "baselines" are absolute values, whereas "improv." shows the improvement percentage of the proposed model versus the baseline one.

Embedding Dataset/Model	Users			Items		
	Proposed	**Baseline**	**Improv.**	**Proposed**	**Baseline**	**Improv.**
Movielens 100K	0.8317	0.4789	73.66%	0.8943	0.6346	40.92%
MovieLens 1M	0.8289	0.5552	49.29%	0.9762	0.7372	32.42%
Netflix*	0.8790	0.4572	92.25%	0.9160	0.8569	6.90%

Experiment 3: This experiment visually shows the improvement obtained by combining the three contributions stated in the preceding section. The visual results of the proposed variational Euclidean model have been compared to the proposed non-variational dot product baseline (DeepMF). Figure 5 shows the returned graphs for both models when applied to the datasets in Table 1. The top graphs in Figure 5 show the baseline results, whereas the bottom graphs plot the proposed model results. As can be seen, the MovieLens 100K dataset (left graphs) displays an unpacked (extended) vision of both user and item samples when the proposed model (bottom-left graph) is used, compared to the baseline (top-left graph) one. The proposed model makes it easier to compare the relationship between samples by visually inspecting the (Euclidean) distances in the graphs. It also decreases intersections between users and items embedding representations. What we are looking at here explains the 'unpacked' quality values shown in Table 3. MovieLens 1M (center graphs) and Netflix* (right graphs) show similar layouts to MovieLens 100K, suggesting that, on CF datasets, the proposed model performs as expected.

As an example of the proposed model application, Figure 6 shows some demographic information from MovieLens 100K. Both graphs in Figure 6 show the location of the users. The graph on the left plots gender information: female (red) versus male (blue). The right graph plots age information: over 40 years of age (red) versus younger users (blue). Please note that the user plot in the bottom left graph of Figure 5 (MovieLens 100K) is not the same as the shapes shown in Figure 6; this is because they belong to different model trainings. Figure 6 is just an example that shows some type of demographic information: male versus female, and younger versus older users. Similar graphs can be obtained from different demographic features of users and from the item's type: zip code, incomings, educational level, genre of movies, type of music, year of book publication, etc. Figure 6 shows that there is not a clear pattern to cluster users attending to their gender or age; that is, in the

MovieLens 100K dataset, males and females rate movies in a similar way, analogously to the younger and older user case. What is relevant here is that we can obtain representative two and three-dimension representative graphs showing the location of CF demographic features. This big data visual information can be useful to take commercial decisions, implement segmented marketing, understand business data, improve RS information, balance data, correct biased datasets, etc.

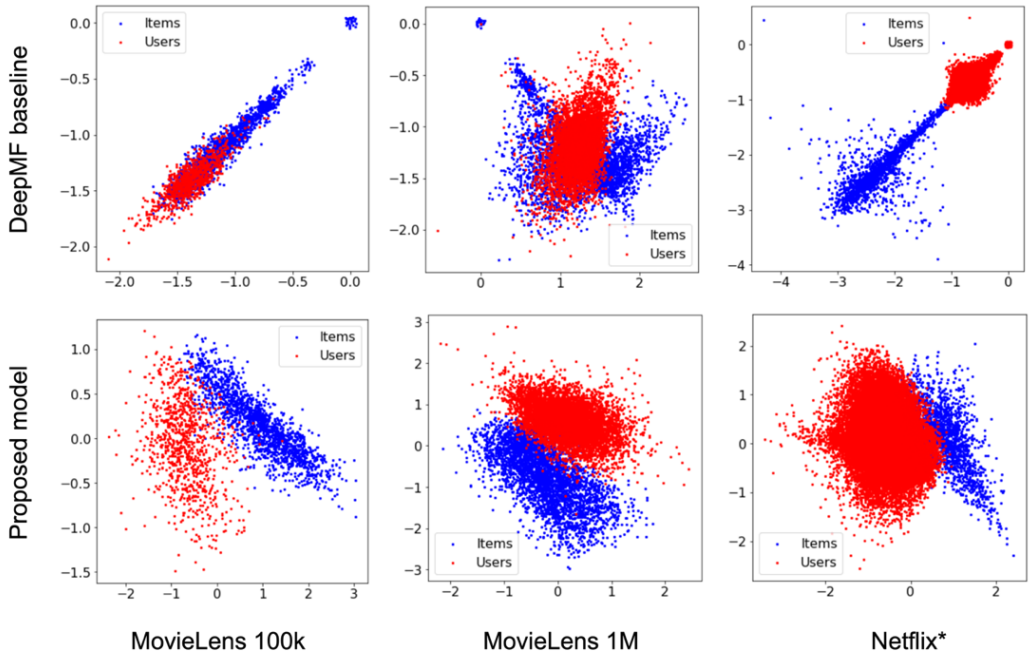

Figure 5. Embedding visualizations of users (red points) and items (blue points) for datasets in Table 1 when the DeepMF baseline is applied (top graphs) versus the proposed variational Euclidean method (bottom graphs).

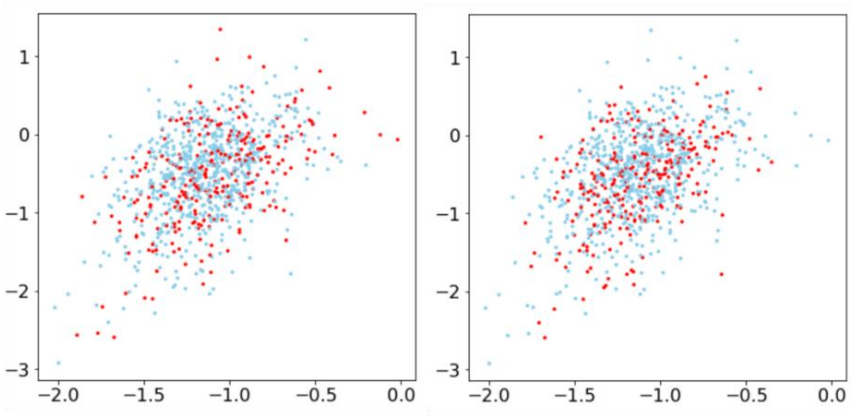

Figure 6. MovieLens 100K demographic information. The left graph shows the latent embedding location of female (red) versus male (blue) users, whereas the right graph shows over 40 years old users (red) versus younger users (blue).

To facilitate reproducibility, Table 4 shows the selected values of the involved parameters in the learning process of the proposed model.

Table 4. Parameter values chosen for the proposed model learning.

# Layers	5
# Neurons in each non-exit layer	Experiment #1: {2, 3, 5, 10}, Experiments #2 and #3: {2}
# Epochs	20
Batch size	16
Activation function of the non-exit layers	ReLu
Activation function of the exit layer	Linear
Loss function	Euclidean distance
Gaussian random distribution	Mean: 0, Variance: 1

4. Discussion

The variational proposed model has proven to adequately afford the visual representation of samples in the CF field. To fulfill this objective, we have tested the impact of limiting the embedding sizes to two neurons (obtaining two-dimensional graphs) or to three neurons (obtaining three-dimensional graphs). Results show a prediction quality of over 98% when the embedding sizes are two or three, compared to the usual five to ten embedding sizes (Table 2). Combining the variational approach and the Euclidean distance loss function, the intra-clustering quality measure improves in the proposed model compared to the DeepMF baseline (Table 3). This improvement can be visually observed by plotting each of the dataset embeddings (Figure 5). The proposed variational model has a better performance than the DeepMF baseline due to two different factors: (a) the designed variational stochasticity (which does not exist in the DeepMF model) spreads the embedding samples through the latent space (Figures 3 and 5), as the generative learning does to obtain fake images by interpolating embeddings; and (b) the proposed Euclidean function can arrange sample embeddings in the latent space in a comprehensible way to humans, compared to the non-Euclidean loss functions that usually implement the baseline model: mean squared differences, mean absolute error, etc.

5. Conclusions

Recommender Systems research is focused on accuracy, but there are some other relevant goals that should be achieved, such as the representative visualization of the collaborative filtering items and users. This can be considered a big data analytics tool that helps system managers. This paper provides an innovative neural model to make visual representations of user and item embeddings. First, we have shown that it is possible to reduce the model embedding sizes to just two or three neurons without any significant loss in prediction accuracy. Then, we have introduced Gaussian variational layers to the proposed model in order to spread the area where samples are located. Finally, a Lambda layer replaces the DeepMF Dot layer (or the NCF MLP); this layer implements the Euclidean distance. Both the Gaussian variational layers and the Lambda-Euclidean layer running together in the proposed model return suitable accuracy results and improved sample representations.

Experiment results show that the user and item embedding representations are conveniently spread through visual representation areas, making it possible to discriminate close samples and to relate between sample pairs. The centroid-based intra-cluster quality measure shows a significant improvement in the proposed neural model compared to the baseline. The plotted graphs also show better embedding representations when the proposed model is tested using the three selected representative collaborative filtering datasets. Results open the door to future works such as: (1) representing demographic features (gender, age, etc.) of samples; (2) explaining recommendations by providing a graph showing, in the same area, the active user, their recommendations and the nearest voted items to both the active user and the recommended items; and (3) incorporating three-dimensional embedding representations in 3D commercial environments.

Author Contributions: J.B. proposed the key idea for this paper and prepared to the manuscript. J.D. made the whole set of experiments and proposed several improvements. F.O. prepared the testing datasets from existing ones. A.G. helped with reviewing the manuscript and he is the corresponding author. All authors have read and agreed to the published version of the manuscript.

Funding: 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*.

Institutional Review Board Statement: The study did not require ethical approval.

Informed Consent Statement: Not applicable.

Data Availability Statement: Movielens datasets: https://grouplens.org/datasets/movielens/, accessed on 19 April 2022; Netflix* dataset: http://cf4j.etsisi.upm.es/, accessed on 19 April 2022.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Batmaz, Z.; Yurekli, A.; Bilge, A.; Kaleli, C. A review on deep learning for recommender systems: Challenges and remedies. *Artif. Intell. Rev.* **2019**, *52*, 1–37. [CrossRef]
2. Bobadilla, J.; González-Prieto, Á.; Ortega, F.; Cabrera, L. Deep learning feature selection to unhide demographic recommender systems factors. *Neural Comput. Appl.* **2020**, *33*, 7291–7308. [CrossRef]
3. Deldjoo, Y.; Schedl, M.; Cremonesi, P.; Pasi, G. Recommender Systems Leveraging Multimedia Content. *ACM Comput. Surv.* **2021**, *53*, 1–38. [CrossRef]
4. Saurabh, K.; Sunil, F.R. Context Aware Recommendation Systems: A review of the state of the art techniques. *Comput. Sci. Rev.* **2020**, *37*, 100255. [CrossRef]
5. Shokeen, J.; Rana, C. A study on features of social recommender systems. *Artif. Intell. Rev.* **2020**, *53*, 965–988. [CrossRef]
6. Bobadilla, J.; Alonso, S.; Hernando, A. Deep Learning Architecture for Collaborative Filtering Recommender Systems. *Appl. Sci.* **2020**, *10*, 2441. [CrossRef]
7. Forouzandeh, S.; Berahmand, K.; Rostami, M. Presentation of a recommender system with ensemble learning and graph embedding: A case on MovieLens. *Multimed. Tools Appl.* **2021**, *80*, 7805–7832. [CrossRef]
8. Zhu, B.; Hurtado, R.; Bobadilla, J.; Ortega, F. An Efficient Recommender System Method Based on the Numerical Relevances and the Non-Numerical Structures of the Ratings. *IEEE Access* **2018**, *6*, 49935–49954. [CrossRef]
9. Mnih, A.; Salakhutdinov, R.R. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems 21*; Koller, D., Schuurmans, D., Bengio, Y., Bottou, L., Eds.; MIT Press: Cambridge, MA, USA, 2008; pp. 1257–1264.
10. Lee, D.D.; Seung, H.S. Algorithms for Non-negative Matrix Factorization. In *Advances in Neural Information Processing Systems 13*; Leen, T.K., Dietterich, T.G., Tresp, V., Eds.; MIT Press: Cambridge, MA, USA, 2001; pp. 556–562.
11. Hernando, A.; Bobadilla, J.; Ortega, F. A non negative matrix factorization for Collaborative Filtering Recommender Systems based on a Bayesian probabilistic model. *Knowl.-Based Syst.* **2016**, *97*, 188–202. [CrossRef]
12. Hernando, A.; Moya, R.; Ortega, F.; Bobadilla, J. Hierarchical graph maps for visualization of collaborative recommender systems. *J. Inf. Sci.* **2014**, *40*, 97–106. [CrossRef]
13. Hernando, A.; Bobadilla, J.; Ortega, F.; Gutiérrez, A. Method to interactively visualize and navigate related information. *Expert Syst. Appl.* **2018**, *111*, 61–75. [CrossRef]
14. Hong-Jian, X.; Xinyu, D.; Jianbing, Z.; Shujian, H.; Jiajun, C. Deep Matrix Factorization Models for Recommender Systems. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, Melbourne, Australia, 19–25 August 2017; pp. 3203–3209. [CrossRef]
15. Xiangnan, H.; Lizi, L.; Hanwang, Z. Neural Collaborative Filtering. In Proceedings of the International World Wide Web Conference, Perth, Australia, 3–7 April 2017; ACM: New York, NY, USA, 2017. ISBN 978-1-4503-4913-0/17/04. [CrossRef]
16. Cui, P.; Wang, X.; Pei, J.; Zhu, W. A Survey on Network Embedding. *IEEE Trans. Knowl. Data Eng.* **2019**, *31*, 833–852. [CrossRef]
17. Guan, N.; Song, D.; Liao, L. Knowledge graph embedding with concepts. *Knowl.-Based Syst.* **2019**, *164*, 38–44. [CrossRef]
18. Zou, Q.; Xing, P.; Wei, L.; Liu, B. Gene2vec: Gene Subsequence Embedding for Prediction of Mammalian N6-Methyladenosine Sites from mRNA. *RNA* **2018**, *25*, 205–218. [CrossRef] [PubMed]
19. Xiang, Y.; Zhen, W.; Jingong, H.; Srinivasan, P.; Moosavinasab, S.; Huang, Y.; Lin, S.M.; Zhang, W.; Zhang, P.; Sun, H. Graph embedding on biomedical networks: Methods, applications and evaluations. *Bioinformatics* **2020**, *36*, 1241–1251. [CrossRef]
20. Škrlj, B.; Kralj, J.; Lavrač, N. Embedding-based Silhouette community detection. *Mach. Learn.* **2020**, *109*, 2161–2193. [CrossRef]
21. Zheng, L.; Huang, Y.; Lu, H.; Yang, Y. Pose-Invariant Embedding for Deep Person Re-Identification. *IEEE Trans. Image Proces.* **2019**, *28*, 4500–4509. [CrossRef]
22. Li, Z.; Tang, J.; Mei, T. Deep Collaborative Embedding for Social Image Understanding. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *41*, 2070–2083. [CrossRef]

23. Kan, S.; Cen, Y.; He, Z.; Zhang, Z.; Zhang, L.; Wang, Y. Supervised Deep Feature Embedding with Handcrafted Feature. *IEEE Trans. Image Process.* **2019**, *28*, 5809–5823. [CrossRef]
24. Shuang, K.; Zhang, Z.; Loo, J.; Su, S. Convolution–deconvolution word embedding: An end-to-end multi-prototype fusion embedding method for natural language processing. *Inf. Fusion* **2020**, *53*, 112–122. [CrossRef]
25. Bobadilla, J.; Lara-Cabrera, R.; González-Prieto, A.; Ortega, F. DeepFair: Deep Learning for Improving Fairness in Recommender Systems. *Int. J. Interact. Multimed. Artif. Intell.* **2021**, *6*, 86–94. [CrossRef]
26. Huang, T.; Zhang, D.; Bi, L. Neural embedding collaborative filtering for recommender systems. *Neural Comput. Applic.* **2020**, *32*, 17043–17057. [CrossRef]
27. Fuzheng, Z.; Nicholas, J.Y.; Defu, L.; Xing, X.; Wei-Ying, M. Collaborative Knowledge Base Embedding for Recommender Systems. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16), San Francisco, CA, USA, 13–17 August 2016; pp. 353–362. [CrossRef]
28. Jeong, S.-Y.; Kim, Y.-K. Deep Learning-Based Context-Aware Recommender System Considering Contextual Features. *Appl. Sci.* **2022**, *12*, 45. [CrossRef]
29. Chih-Ming, C.; Chuan-Ju, W.; Ming-Feng, T.; Yi-Hsuan, Y. Collaborative Similarity Embedding for Recommender Systems. In Proceedings of the The World Wide Web Conference (WWW '19), San Francisco, CA, USA, 13–17 May 2019; pp. 2637–2643. [CrossRef]
30. Heng-Tze, C.; Levent, K.; Jeremiah, H.; Tal, S.; Chandra, T.; Aradhye, H.; Anderson, G.; Corrado, G.; Chai, W.; Ispir, M.; et al. Wide & Deep Learning for Recommender Systems. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems (DLRS 2016), Boston, MA, USA, 15 September 2016; pp. 7–10. [CrossRef]
31. Zhao, W.X.; Huang, J.; Wen, J.R. Learning Distributed Representations for Recommender Systems with a Network Embedding Approach. In Proceedings of the Information Retrieval Technology (AIRS 2016), Beijing, China, 30 November–2 December 2016; Lecture Notes in Computer Science 9994. Springer: Berlin/Heidelberg, Germany, 2016; pp. 224–236. [CrossRef]
32. Zhang, C.; Yu, L.; Wang, Y.; Shah, C.; Zhang, X. Collaborative User Network Embedding for Social Recommender Systems. In Proceedings of the 2017 SIAM International Conference on Data Mining (SDM), Houston, TX, USA, 27–29 April 2017; pp. 381–389. [CrossRef]
33. Liu, H.; Zhao, X.; Wang, C.; Liu, X.; Tang, J. Automated Embedding Size Search in Deep Recommender Systems. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Online, 25–30 July 2020; pp. 2307–2316. [CrossRef]
34. Bobadilla, J.; Gutierrez, A.; Alonso, S.; González-Prieto, A. Neural Collaborative Filtering Classification Model to Obtain Prediction Reliabilities. *Int. J. Interact. Multimed. Artif. Intell.* **2021**. [CrossRef]
35. Liu, X.; Gherbi, A.; Wei, Z.; Li, W.; Cheriet, M. Multispectral Image Reconstruction from Color Images Using Enhanced Variational Autoencoder and Generative Adversarial Network. *IEEE Access* **2021**, *9*, 1666–1679. [CrossRef]
36. Liu, Z.-S.; Siu, W.-C.; Wang, L.-W.; Li, C.-T.; Cani, M.-P.; Chan, Y. Unsupervised Real Image Super-Resolution via Generative Variational AutoEncoder. In Proceedings of the EEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 1788–1797. [CrossRef]
37. Liu, Z.-S.; Siu, W.-C.; Chan, Y.-L. Photo-Realistic Image Super-Resolution via Variational Autoencoders. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 1351–1365. [CrossRef]
38. Harper, F.M.; Konstan, J.A. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.* **2015**, *5*, 1–19. [CrossRef]
39. Ortega, F.; Zhu, B.; Bobadilla, J.; Hernando, A. CF4J: Collaborative filtering for Java. *Knowl.-Based Syst.* **2018**, *152*, 94–99. [CrossRef]