Applied Intelligence (2024) 54:2472–2490
https://doi.org/10.1007/s10489-024-05313-4

Wasserstein GAN-based architecture to generate collaborative filtering synthetic datasets

Jesús Bobadilla[1,2] · **Abraham Gutiérrez**[1,2]

Accepted: 1 February 2024 / Published online: 17 February 2024

Abstract
Currently, generative applications are reshaping different fields, such as art, computer vision, speech processing, and natural language. The computer science personalization area is increasingly relevant since large companies such as Spotify, Netflix, TripAdvisor, Amazon, and Google use recommender systems. Then, it is rational to expect that generative learning will increasingly be used to improve current recommender systems. In this paper, a method is proposed to generate synthetic recommender system datasets that can be used to test the recommendation performance and accuracy of a company on different simulated scenarios, such as large increases in their dataset sizes, number of users, or number of items. Specifically, an improvement in the state-of-the-art method is proposed by applying the Wasserstein concept to the generative adversarial network for recommender systems (GANRS) seminal method to generate synthetic datasets. The results show that our proposed method reduces the mode collapse, increases the sizes of the synthetic datasets, improves their ratings distributions, and maintains the potential to choose the desired number of users, number of items, and starting size of the dataset. Both the baseline GANRS and the proposed Wasserstein-based WGANRS deep learning architectures generate fake profiles from dense, short, and continuous embeddings in the latent space instead of the sparse, large, and discrete raw samples that previous GAN models used as a source. To enable reproducibility, the Python and Keras codes are provided in open repositories along with the synthetic datasets generated to test the proposed architecture (https://github.com/jesusbobadilla/ganrs.git).

Keywords WGANRS · Generative Adversarial Networks · Recommender Systems · Wasserstein distance · Synthetic datasets · Collaborative Filtering

1 Introduction

Recommender systems (RSs) are used to provide personalization facilities to users of internet services. Large companies that use RSs are Spotify, TripAdvisor, Netflix, Google Music, etc. RSs are becoming increasingly important due to its capacity to provide both accurate recommendations and recommendations designed to retain people using the service. Recommendations are provided by suggesting the products or services that have a higher probability of being liked by the user.

Consequently, it is necessary to filter the available items (products or services) in the RS. For this reason, RSs are usually classified according to their filtering approach. Social [1, 2], content-based [3], demographic [4, 5], context-aware [6], collaborative filtering (CF) [7] and their ensembles [8] are the most commonly used strategies. Social filtering recommends to the active users items that their followed, group of friends, contacts, etc., like. Content-based recommendations include items with similar content to those the active user liked. It is usual to compare descriptions or even item images. Demographic filtering selects users having demographic features such as those of the active user (similar age, same sex, same zip code or near zip code, etc.) and then extracts those item preferences. Context-aware filtering usually relies on geographic information, such as GPS coordinates. The most accurate and relevant filtering strategy is the

✉ Jesús Bobadilla
jesus.bobadilla@upm.es

Abraham Gutiérrez
abraham.gutierrez@upm.es

[1] Universidad Politécnica de Madrid, ETSISI, Ctra. de Valencia Km. 7, Madrid, Spain

[2] Technical University of Madrid, ETSISI, Ctra. de Valencia Km. 7, Madrid, Spain

collaborative strategy. In this strategy, recommendations are based on the preferences of the most similar users. The machine learning method that best fits the CF concept is the K-nearest neighbours algorithm (KNN) [9]. It is simple and directly implements the CF concept, where the neighbours are the most similar users to the active users. The main drawbacks of the KNN are that it is a memory-based method, it runs slowly, and it is not sufficiently accurate. The model-based matrix factorization (MF) [10] solves the KNN limitations. Moreover, it contains two vectors of hidden factors. The first vector is used to code (compress) the relevant information of users, whereas the second vector is used to code the relevant information of items. Both vectors belong to the same latent space, and they are combined using a dot product. Additionally, the hidden factors are optimized by minimizing the prediction errors. Non-negative matrix factorization (NMF) [11] ensures that the hidden factors are non-negative to enable some semantic interpretations of predictions.

Deep learning [12] can currently be implemented to obtain improved MF results. The simplest deep learning architecture is similar to that of MF. In this method, the hidden factors of the user are replaced by neural user embedding, and analogously, the hidden factors of items are replaced by neural item embedding. This model is called deep matrix factorization (DeepMF) [13], and it is better than MF due to the ability of its neural networks to remove complex non-linear patterns in raw data. DeepMF combines the embeddings using a dot layer. An improved DeepMF model is the variational deep matrix factorization (VDeepMF) [14], where an intermediate layer codes the parameters of a chosen distribution (usually Gaussian), and from it, a stochastic-based sampling process spreads samples in the latent space. The DeepMF (or VDeepMF) dot layer can be improved by replacing it with a multilayer perceptron (MLP) that combines the hidden factors of users and item embeddings and generates a manifold. This approach is called neural collaborative filtering (NCF) [15]. Our proposed architecture combines a DeepMF model and a Wasserstein generative adversarial network (GAN). GAN [16] networks can generate fake samples following the distribution of a source set of real data samples. The most common use of GAN networks is to generate realistic fake faces from a dataset of real human faces. Similarly, our objective is to generate synthetic (fake) CF samples from an existing dataset of CF samples, such as MovieLens [17]. Then, by collecting many fake samples, a synthetic CF dataset can be created.

Generating synthetic CF datasets makes it possible to simulate the stress situation in the RS, as it can be generated 'families' of datasets where gradually some of the parameters can be selected. For example, we can generate a family of CF datasets where the number of users grows from several thousands to millions, and then test in advance the performance of our system in different scenarios where the number of users gradually, or suddenly, grows (e.g. due to a marketing campaign or an influencer action). This simulation can avoid system failures in extreme situations. Similarly, a family of datasets can be generated with a growing number of items. It leads to more sparse scenarios where the RS accuracy could decrease. This type of simulation gives us the convenience of incorporating many products or services in a short period of time. Additionally, the generation of synthetic datasets makes possible that researchers test their machine learning models in bounded scenarios, difficult to find in real datasets, such as increasingly sparse data matrices, different cold start situations, or extreme pattern variations in the user profiles.

The state-of-the-art methods in CF generation include statistical methods that are not able to adequately determine the patterns of complex datasets. Therefore, adversarial approaches [18], and GAN-based approaches have been proposed. Preventing shilling attacks is a relevant objective in the RS field, and some GAN-based approaches act as a defence against them [19]. Data augmentation is an obvious field where GANs can be applied. Purchase profiles are used in the collaborative filtering generative adversarial network model (CFGAN) [20] model to increase the number of training samples in a dataset of commercial products. The identity-preservation generative adversarial network model (IPGAN) [21] incorporates negative sampling information to improve accuracy results. It allows two separate generative models to be incorporated, with one method managing positive data and the other method processing negative samples. Session information is used in the deep collaborative filtering generative adversarial network (DCFGAN) model [22] instead of matrices of votes combining GAN and reinforcement learning. To run recommendation training, the neural collaborative generative adversarial network (NCGAN) [23] incorporates a regular GAN that processes the intermediate CF results provided by a neural network stage. The recurrent generative adversarial network (RecGAN) [24] combines a recurrent neural network (RNN) and a GAN to process temporal patterns. Unbalanced datasets are managed using a Wasserstein GAN acting as a generator and the packing generative adversarial network (PacGAN) as a discriminator [25]. Finally, a conditional generative adversarial network (CGAN) performs a conditional generation of ratings [26].

The RS state-of-the-art method that generates synthetic CF datasets is divided into statistical and machine learning approaches. Solutions in the first group allow us to parameterize the results (to change the number of items, users, etc.), but they do not adequately capture the complex non-linear relations between users and items. Consequently, the accuracy of this method is poor. The second group makes

use of deep learning generative adversarial networks to create fake profiles or fake samples. The accuracy is improved, but the GAN architectures in this group take discrete and heavily sparse CF datasets as a source, leading to results that are obtained slowly and the subject-to-mode collapse problem. In addition, the number of items cannot be changed. Nevertheless, the existing generative adversarial network for recommender systems (GANRS) method makes its GAN generation start from dense and continuous latent space embeddings, obtaining more accurate results and enabling us to choose the number of users and the number of items in the synthetic datasets. The method proposed in this paper borrows the GANRS architecture, making the necessary changes to introduce the Wasserstein concept. Wasserstein generative adversarial networks (WGANs) are designed to reduce the inherent 'mode collapse' of the GAN architecture. In the model regularization Wasserstein generative adversarial network (MRWGAN) [27], an RS is implemented. Moreover, an autoencoder is used to implement the generative model, and a model-Wasserstein regularized distance is used as the function loss. It achieves better accuracy with missing data than the state-of-the-art methods. Analogously, an L1 regularized Wasserstein loss function has been used for autoencoder-based CF [28] to learn a low-rank representation of variables in the latent space. The Wasserstein distance has also been implemented to tackle the cold-start issue in CF, minimizing it under user embedding constraints.

GAN approaches also have their own drawbacks, particularly: a) a long training time, b) a long inference time when the GAN model is very deep, c) difficulty to set relevant CF parameters such as the number of items and the number of users in the dataset, d) possibility of suffering from the 'mode collapse' behaviour, e) difficulty to adequately learn from the sparse data sets of CF, and e) lack of fine tuning the variation of the results in successive executions.

No standard machine learning quality measures are defined to compare synthetic datasets created or generated using different statistical or generative models. This is because these models are designed to catch the complex nonlinear patterns of the source data, and there are no simple comparations able to discriminate the quality of the generated results, such as in regression (MAE, MSD, etc.) or classification (accuracy, precision, recall, etc.). To better understand this drawback, we can analyse the face image quality assessment strategies [29], which are based on the character, fidelity, and utility features of facial biometrics. In the RS field we do not have such type of information to make a similar process, since what we are generating are user/item vector profiles. In addition, face image quality makes a distinction between approaches that require a reference, a reduced reference, and no reference of faces, where only the two first cases have some accurate quality measure; precisely those situations that RS cannot manage as they do not have the equivalent 'reference' to the face image quality field. Finally, the conceptual problem of the 'quality paradox' inherent in these quality measures is heavily present in the RS scenario, where a generated dataset should not be too similar to the source, otherwise it would not be useful, and should not be too different from the source, otherwise it would not be representative. Therefore, research papers that generate synthetic datasets [23–25] test them by running several CF Machine Learning models and comparing the results obtained on different instances of the generated datasets from the same source data. This is the strategy that our paper follows in its 'Results' section.

In the seminal GANRS paper [30], relevant innovations are incorporated, and the previous RS GAN architectures are compared to generate synthetic datasets. However, a significant drawback occurs. The process to convert from the latent space generated samples (dense, small, and continuous) to the raw samples (sparse, large, and discrete) that form the synthetic dataset generates duplicated samples that must be removed. This is a common drawback in a discretization task, but if the number of duplicated samples is high, a 'collapse' in the GAN generation can occur. The innovation of our proposed Wasserstein GAN approach (WGANRS) is the introduction of the Wasserstein design (function loss, weight constraints, etc.) to the existing GANRS method in the hope that the mode collapse situations are reduced. The proposed method borrows the stages defined in [30] and replaces the regular GAN generation kernel by a Wasserstein approach (WGAN). The WGAN provides four relevant improvements: 1) it incorporates a new loss function that is interpretable and has clear stopping criteria, 2) it empirically returns better results, 3) the GAN mode collapse is significantly reduced, and 4) it provides a clear theoretical backing. The WGAN loss function is based on the earth mover's distance, and it incorporates an f_w function that acts as a discriminator model, called the 'critic'. The critic estimates the earth mover's distance, processing the highest difference between the generated distribution and the real distribution under several parameterizations of the f_w function. The critic makes the generator work harder by looking at different projections. Our most relevant predictor that measures the mode collapse mitigation will reduce the removed samples and consequently increase the number of samples of the synthetic files (their sizes). We will also test some other quality measures such as the precision, recall, and the distribution of the ratings, users, and items.

Figure 1 shows the innovation of the proposed method compared to SOTA, particularly with the most currently published baseline (GANRS) on which our method is based. As shown in a) (top of Fig. 1), SOTA methods that generate CF datasets take only raw profiles and generate synthetic RS datasets. This is an analogous process to fake face creation, which can be generated by GANs from datasets of real faces. It is known

that a recurrent problem in these processes is mode collapse [30], which leads to a lack of balanced generation of samples. Some categories are overrepresented, whereas other categories are underrepresented. As an example, we could obtain an enormous quantity of fake samples of Number 7 by using the Modified National Institute of Standards and Technology (MNIST) dataset. However, some other numbers are rarely generated. Notably, in Fig. 1a (SOTA methods), the GAN module is fed with RAW data, such as image pixels or in our case, user profiles. These RAW data are large, discrete, and sparse, leading to the mode collapse problem.

The most current research in the area is the GANRS method (our baseline). Its high-level architecture is shown in Fig. 1b, where the GAN model is not fed with RAW data. Instead, it is fed with deep learning embedded data. These embedded data are short, continuous, and dense vectors. The embedded data contain compressed information on the items and users in the RS. As a result, both the performance and the accuracy of the GANRS are improved compared to the previous SOTA models and methods.

In the GANRS baseline [30], the mode collapse problem is reduced, and the RS datasets generated are less biased than those created using SOTA methods. Regardless, the mode collapse remains, and it produces a certain degree of redundant samples. To reduce mode collapse even further, our proposed WGANRS method introduces the Wasserstein concept into the GAN kernel (Fig. 1c). The Wasserstein approach has been shown to yield better results by reducing mode collapse when applied to GANs [27]. This approach requires the introduction of a new loss function and benefits from a theoretical backing and a defined stopping criterion.

The hypothesis of the paper claims that incorporating the Wasserstein concept into the generative kernel of the GANRS method will lead to a decrease in the mode collapse problem inherent to the GAN when applied to CF scenarios. Consequently, the proposed WGANRS is expected to generate more accurate CF datasets.

The structure of the paper is as follows: In Section 2, the proposed WGANRS method (from the existing GANRS information) is explained and formalized. In Section 3, the design of the experimental executions of code is introduced, and the results are shown and analysed using the MovieLens and Netflix* datasets as a source. Moreover, the most relevant results are provided. In Section 4, the most remarkable conclusions are presented, and some future work is proposed. Additionally, the references section includes current representative papers in the main RS area and in the specific GAN generation of CF datasets.

2 Method

This section is divided into two subsections. In the first subsection, the proposed method concepts, its architecture, and the sequence of processes and stages to both train the model and generate the synthetic datasets in a feedforward prediction are explained. The second subsection contains the necessary equations to formalize the method, grouped into the main stages of the architecture. The Python and Keras codes of the proposed method is available in (https://github.com/jesusbobadilla/ganrs.git).

2.1 Concepts and architecture

The proposed deep learning architecture is based on five sequential stages in which a neural CF, a WGAN model, and a clustering process are involved. Moreover, a CF dataset is used as the source, and a synthetic dataset that has the same format as the source dataset and similar data distributions is generated. The key issue involving both the GANRS [30] seminal baseline and the WGANRS proposed architecture is that the GAN or WGAN stages are fed with dense, short, and continuous embeddings in the latent space instead of sparse, large, and discrete raw data. It makes the work of both the generator and the discriminator models easier, faster, and more accurate. The obvious drawback of the proposed design is the theoretical loss of quality involved in the compression stage (coder) and, particularly, in the subsequent decompression (decoder). However, when converting from embedded to raw samples, a significant benefit emerges: we can choose the target number of users and items, making the GANRS and WGANRS models more flexible and useful than the state-of-the-art methods. Figure 2 shows an overview of the proposed WGANRS architecture. From an existing source dataset (most-left side in Fig. 2), such as MovieLens, a synthetic dataset (most-right side in Fig. 2) is generated with the same format (to be useful to researchers) and similar patterns and distributions of the users, items, and ratings. This dataset can be generated by choosing the desired number of users, items, and starting number of samples to be useful for companies and researchers as a base for simulations, anticipating diverse future scenarios, or as ground data for new machine learning models.

The proposed WGANRS first converts (compresses) the input sparse dataset to its embedding-based representation and converts (decompresses) the generated (fake) embedding-based dataset to its raw and sparse representation. A DeepMF model (left side in Fig. 2) was chosen to perform the compression stage due to its simplicity and performance, and K-means clustering (right side in Fig. 2) was used to run the necessary decompression. In this scenario, the K-means algorithm has the advantage of setting the K^* number of users and K^{**} number of items we want the generated dataset to hold. Finally, our architecture kernel is based on a WGAN model (centre of Fig. 2) to generate fake embedding samples from real embedding samples.

The DeepMF model used for the compression task has a previous learning stage (top-left draw in Fig. 3) where the

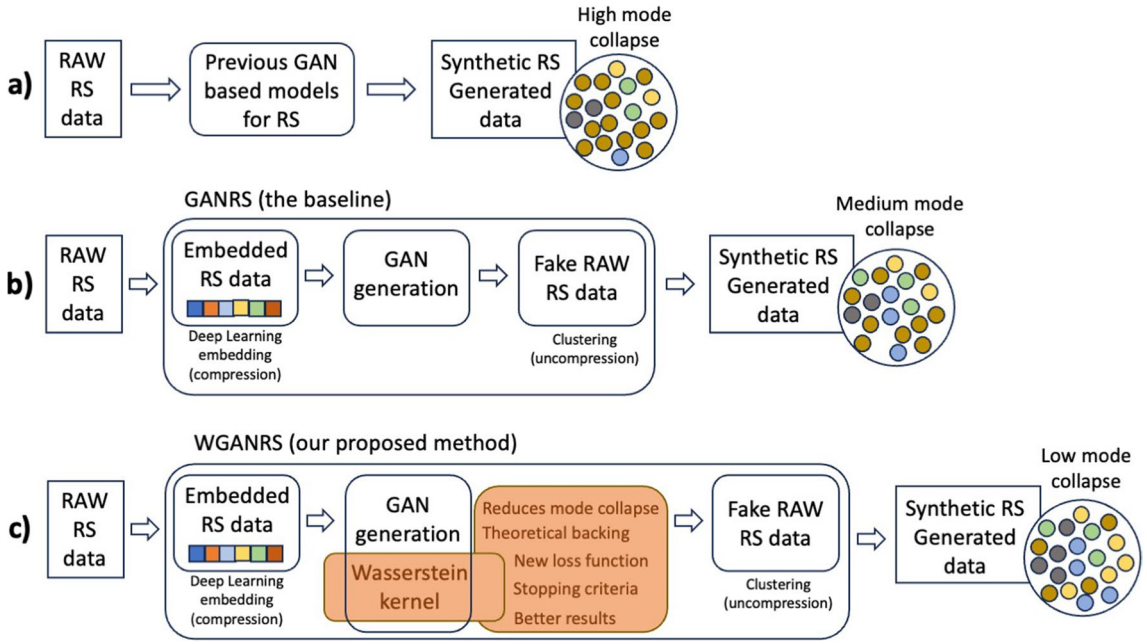

Fig. 1 Innovation of the proposed method and its expected impact in solving the GAN mode collapse. Figure 1a shows the traditional GAN approach in the CF context, Fig. 1b shows the improvement introduced in the baseline method to adequately process sparse data, and Fig. 1c details the proposed introduction of the Wasserstein kernel to reduce the mode collapse problem

embedding weights are set by means of backpropagation optimization. The DeepMF model contains two separate embedding layers: one layer for users and the other layer for items. These embeddings must have the same size, which usually ranges from 5 to 15 neurons. It is expected that similar users will be coded with similar embedding maps (codes), and the same applies for items. Once the DeepMF model has been trained, we can feedforward each existing user ID to obtain its embedding representation (top-right draw in Fig. 3). The same process is performed with all existing item IDs. Thus, we obtain a matrix $I \; x \; E$ containing the embedding representations of the items, where I is the number of items in the source dataset and E is the embedding size. Analogously, we obtain a matrix $U \; x \; E$ containing the embedding representations of the users, where U is the number of users in the source dataset.

By combining the source dataset (left side of Fig. 2), the compressed item matrix, and the compressed user matrix (top right draw in Fig. 3), we can obtain the embedding representation of the source dataset, as shown in the "embedding-based CF dataset" in the bottom left graph of Fig. 3 and in Fig. 2.

Starting from the embedding-based CF dataset as a source, the proposed WGANRS architecture generates the "fake embedding-based CF dataset" (centre of Fig. 2), and it is performed by means of a Wasserstein GAN. The first stage of this generative task is the WGAN training (bottom-left in Fig. 3), where the generator model creates synthetic samples from Gaussian stochastic vectors containing random noise. Then, the Wasserstein critic (discriminator) performs the necessary binary classification to label samples as 'real' or 'fake'. Notably, the 'fake' samples come from the generator model, whereas the 'real' samples are randomly taken from the previously generated 'real embedding-based CF dataset'. Once the training process has finished, we can forget the critic model and take the generator model to create as many fake embedding samples as needed. The whole process (training and feedforward generation) is expected to be fast due to the compressed embedding representation and accurate due to the Wasserstein restrictions to avoid mode collapse.

In the last stage of the proposed architecture, it is necessary to decompress the 'fake embedding-based CF dataset' (right side in Fig. 2 and bottom right draw in Fig. 3). In this stage, we have generated a very large number of fake samples, consisting of tuples $< user\_embedding, item\_embedding, rating >$, where both the user and the item embeddings produce vectors of real numbers. We have to convert this set of tuples to a discretized version $< user\_ID, item\_ID, rating >$, where *user_IDs* are integers in the range [1..number of users], and analogously *item_IDs* are integers in the range [1..number of items]. Once the neural network has been trained, the user embedding assigns similar codes to similar users (same with the embedding layer). This inherent property of the embedding layers makes it possible to incorporate a clustering process to the proposed

method, in charge of grouping similar users and items to the desired number of users and items in each synthetic dataset.

This is a discretization process in which the WGANRS has been designed to set the desired number of users and items in the generated dataset. To implement it, K-Means clustering [31] was performed, since it allows us to set the K^* number of users and the K^{**} number of items. Figure 4 shows the following concept: a K-means is used to cluster K^* users, whereas another K-Means process is used to cluster K^{**} items. Since similar users should have similar embeddings, it is expected that they will be grouped in the same clusters, analogously with items. Each user number in the generated dataset corresponds to the cluster number in the K-Means where the fake user embedding has been grouped. For example, the left-most sample in the fake embedding-based CF dataset (left side of Fig. 4) was grouped with its user (green colour) in the K* group and its item (blue colour) in Group 3. Consequently, the generated fake sample in the synthetic dataset is < K*, 3, rating >. In this example, the 'rating' is the value of the first sample of the source dataset (left side of Fig. 2).

Finally, due to the discretization process, duplicated samples can be found. This happens when two different generated samples share the same user ID and the same item ID. When the chosen number of users and items is high, it is more difficult to find duplicated samples since there is a wider variety of clusters, and it is expected that the users and the items will be assigned to the groups in a balanced way. Duplicated samples can be managed by simply removing the spare samples. The expected balance in the clustering groups could be 'broken' if the mode collapse is happening in the GAN. Indeed, the Wasserstein concept is used in this paper to avoid mode collapse, and the number of removed samples will be used as a quality measure in the results section. The lower the removed samples are, the better the method is.

Since two of the hyperparameters in the proposed model are the number of users and the number of items, the clustering method that better fits this information is K-Means, where directly we can set K* as the number of users and K** as the number of items. In fact, this is one of the unusual situations where the K value is known before the clustering process. Thus, other relevant clustering methods such as hierarchical, distribution, density or fuzzy-based ones are not adequate in this scenario.

In the following subsection, the WGANRS approach is formalized. Moreover, equations have been provided.

2.2 Formalization

2.2.1 CF definitions

First, we define the main sets in the RS: set of users U, items I, range of ratings V, and existing samples S.

We let U be the set of users who make use of a CF RS (1)

We let I be the set of items available to vote on in the CF RS (2)

We let V be the range of allowed votes, where $V = \{1, 2, 3, 4, 5\}$ (3)

We let S be the set of samples contained in the CF dataset, where $N = |S| =$ the total number of cast votes (4)

$$S = \{< u, i, v >_1, < u, i, v >_2, \ldots, < u, i, v >_N\}, \text{ where each } u \in \{1, \ldots, |U|\} \text{each } i \in \{1, \ldots, |I|\}, \text{and each } v \in \{1, \ldots, |V|\} \quad (5)$$

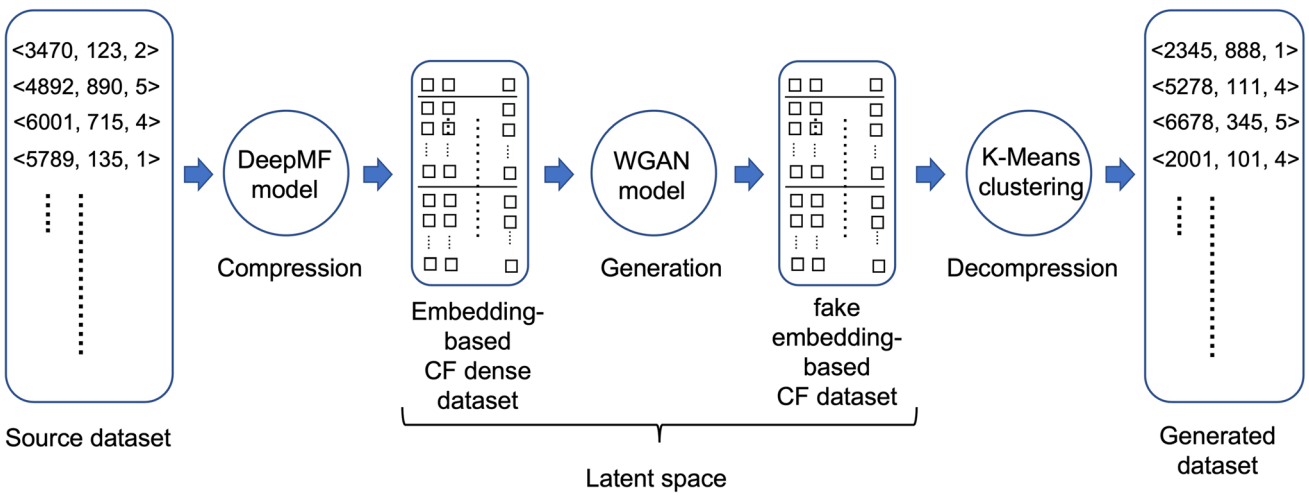

Fig. 2 Overview of the proposed WGANRS architecture

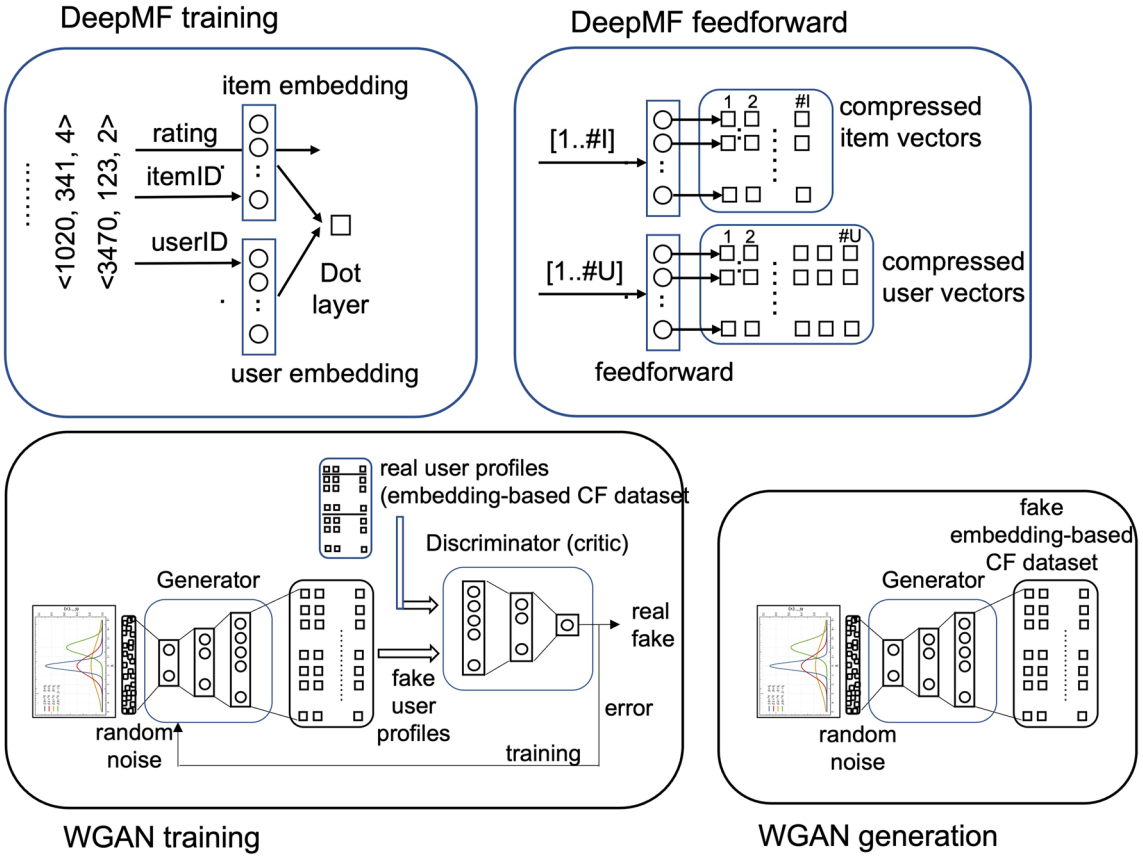

Fig. 3 DeepMF and WGAN models involved in the WGANRS architecture

The formalization of the defined CF dataset consists of a set of tuples <userID, itemID, rating (number of stars)>, where the 'rating' is the vote assigned for the 'userID' to the 'itemID'.

2.2.2 DeepMF training

The Deep MF training (top − left graph in Figure 3)is conducted to create a model that can embed each user ID and each item ID.These two embeddings feed the WGANRS generative stage. Each embedding is a compressed representation of the user or the item. The embeddings are unidimensional vectors of size $E+1$. We define $f^{eu}(u)$ as the function that compresses the user 'u' information and analogously $f^{ei}(i)$ as the function that compresses the item 'i' information (6)

We let $E+1$ be the size of the two neural layer embeddings used to vectorize each user and each item belonging to U and I, respectively. We let $f^{eu}(u) = \vec{e}^u = [e_0^u, e_1^u, \ldots, e_E^u]$, where f^{eu} is the embedding layer output of the users and $u \in \{1, \ldots, |U|\}$ (7)

We let $f^{ei}(i) = \vec{e}^i = [e_0^i, e_1^i, \ldots, e_E^i]$, where f^{ei} is the embedding layer output of the items and $i \in \{1, \ldots, |I|\}$ (8)

By combining the dense vectors of the user and item embeddings ($\vec{e}^u = [e_0^u, e_1^u, \ldots, e_E^u]$ and $\vec{e}^i = [e_0^i, e_1^i, \ldots, e_E^i]$), we can make rating predictions in the DeepMF training stage. The dot product of the user embedding and the item embedding in each$< u, i, v >_j \in S$ provides its rating prediction.

$$\hat{y}_j = f^{eu}(u) \bullet f^{ei}(i) = \vec{e}^{iu} \bullet \vec{e}^i = [e_0^u, e_1^u, \ldots, e_E^u] \bullet [e_0^i, e_1^i, \ldots, e_E^i] \quad (9)$$

Below, the most relevant equations in the back propagation algorithm are defined, and we set the output error as the mean squared differences metric. These equations are not distinctive of the proposed method.

$\frac{1}{2}(y_j - \hat{y}_j)^2$ is the output error used in the DeepMF neural network to start the back propagation algorithm, where the neural weights are iteratively improved from the δ_j values, $\Delta w_{ji} = \alpha y_j f'(Net_i) \sum_k w_{ik}\delta_k$, where k is a hidden layer, and $\Delta w_{ji} = \alpha y_j f'(Net_i) \frac{1}{2}(y_k - \hat{y}_k)^2$ if k is the output layer.$i, j,$ and k are successive sequential layers. (10)

2.2.3 DeepMF feedforward

Once DeepMF has learned, we can collect the embedding representation of each user and each item in the CF RS. Therefore, all the existing itemID and userID in the RS

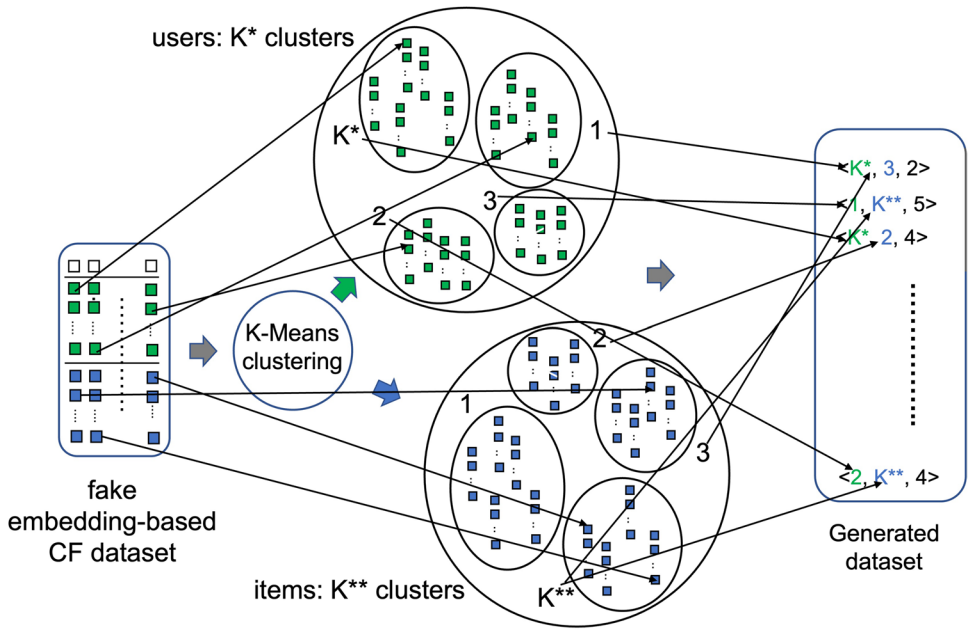

Fig. 4 Clustering stage in the WGANRS architecture

dataset feed the trained DeepMF model, and their embedded representations can then be obtained. It can be done by making the feedforward prediction operation (top-right graph in Figure 3) on the trained DeepMF model.

We let $E^* = \{< u, \vec{e}^u = \left[e_0^u, e_1^u, \ldots, e_E^u\right] >, \forall u \in U\}$ be the set of embeddings for all the RS users (11)

We let $E^*(u) = \vec{e}^u = \left[e_0^u, e_1^u, \ldots, e_E^u\right]$ (12)

We let $E^{**}(i) = \vec{e}^i = [e_0^i, e_1^i, \ldots, e_E^i]$ (14)

Let $E^{**} = \left\{< i, \vec{e}^u = \left[e_0^u, e_1^u, \ldots, e_E^u\right], \vec{e}^i = \left[e_0^i, e_1^i, \ldots, e_E^i\right] >, \forall i \in I\right\}$ be the set of embeddings for all the RS items. (13)

2.2.4 Setting the dataset of the embeddings

Now, we collect the above embedding representations of all the itemID and userID in the RS to translate the set 'S' (5) to the set 'R' (15). The set 'R' is the embedding-based dataset version of the original RAW dataset.

We let

$R = \left[< E^*(u), E^{**}(i), v >\right], \forall < u, i, v >_j \in S$ be the embedding − based dataset of real samples (15)

2.2.5 WGAN training

The core concept of the GAN methodology is to jointly train a generator model and a discriminator model. Once the architecture has been trained, the generator model creates new samples that follow the distribution of the training samples. The discriminator model attempts to differentiate between real samples and generated ones. This is a min-max optimization problem of the form:

$Min_G Max_D(\mathbb{E}_{x \sim p_{data}}[\log D(x)] + \mathbb{E}_{z \sim p_{latent}}[\log(1 - D(G(z))])$, where $G : \mathcal{Z} \rightarrow \mathcal{X}$ is the generator model, which maps from the latent space $\mathcal{Z}$ to the input space $\mathcal{X}$; $D : \mathcal{X} \rightarrow \mathbb{R}$ is the discriminator model, which maps from the input space $\mathcal{X}$ to a classification value (real/fake). $\mathbb{R} \rightarrow \mathbb{R}$ is a concave function. The above formula is the optimization function, the expression that both networks (generator and discriminator) try to optimize. G aims to minimize it, whereas D wants to maximize it.

The bottom-left graph in Fig. 3 shows the generative learning. The GAN architecture consists of a discriminator classifier (16) and a generator model (17). The generator creates fake samples (RS profiles, in our case), whereas the discriminator tries to detect fake samples. In generative adversarial training, the discriminator progressively learns to accurately differentiate fake profiles from real profiles. At the same time, the generator learns to make fake samples difficult to distinguish from real profiles. We call f^D (16) the discriminator model and f^G (17) the generator

model. Both f^D and f^G will iteratively learn and improve themselves by minimizing a loss function (18) f^{GD}.

We let f^D be the discriminator D model belonging to a GAN model (16)

We let f^G be the generator G model belonging to a GAN model (17)

We let f^{GD} be the cost function of the GAN model (18)

$f^{GD} = Min_G Max_D f(D,G) = E_R[f_w(\mathrm{R})] + E_z[f_w(G(z)]$, where E_R is the expected value for real samples, z is the random noise that feeds generator G, and E_z is the expected value for the generated fake profiles $G(z)$. f_w is the Wasserstein function based on the earth mover's distance · This function satisfies the 1 – Lipschitz constraint: $\left|\mathrm{f}(x_1) - \mathrm{f}(x_2)\right| \leq |x_1 - x_2|, \forall x_1, x_2$. R refers to (15). Notably, the Wasserstein concept has been introduced in Eq. (18). Mainly, it is implemented in the loss function code of the generative model. The Wasserstein approach has been designed to reduce the mode collapse problem inherent to the GAN. We implement it to reduce the mode collapse in our WGANRS proposed method and to consequently reduce the number of excessively generated similar profiles in the RS.

2.2.6 WGAN generation

Once the GAN has been trained, we can generate as many samples as needed. We can introduce batches of random Gaussian noise vectors 'z' and implement the feedforward process (*model.predict*). The result are batches of fake embedded profiles (bottom-right graph in Figure 3).

We let $F = f^G$ be the generated dataset of fake samples from different random noise vectors$z$ (19)

2.2.7 Clustering of items and users

Figure 4 shows the clustering process, where we set K^* (20) as the number of users in the generated dataset and K^{**} (21) as the number of items on it. The N generated fake profiles will contain both fake user embeddings and fake item embeddings (where $N >> K^*$ and $N >> K^{**}$).

We let K^* be the number of clusters used to group the embeddings of the users (20)

We let K^{**} be the number of clusters used to group the embeddings of the items (21)

For each generated fake user (each user of N), we must select the nearest user from the K* existing users (22). The $h^*(u)$ function makes this group. The same process is used for items. For each generated fake item (each one of the N), we must select the nearest item from the K** existing items (23). The $h^{**}(i)$ function makes this group.

We let $h^*(u) = c | c \in \{1, \ldots, K^*\}$ be the clustering operation that assigns a centroid to each user. (22)

We let $h^{**}(i) = c | c \in \{1, \ldots, K^{**}\}$ be the clustering operation that assigns a centroid to each item (23)

2.2.8 Setting the dataset of the item IDs and user IDs

To create the synthetic dataset, the generated set of embeddings F (19) is converted to its discretized version H (24).

We let H be the item ID and user ID discrete dataset obtained from the embedding-based dataset F of fake samples.

$$H = \left[< h^*(u), h^{**}(i), v > | \forall < E^*(u), E^{**}(i), v > \in F \right] \quad (24)$$

Sometimes, different generated samples in Set F will be discretized to the same user and item: $< h^*(u), h^{**}(i), v > = < h^{*\prime(u)}, h^{**\prime(i)}, v >$.

This particularly happens if the GAN suffers from mode collapse. In these cases, there are samples with identical information, and we create the set H where duplicated samples are removed (25).

We let $S = \{H\}$ be the synthetic generated dataset version of H where duplicated samples are removed.

The last transformation removes samples when a fake user casts different votes (v, v′) to the same item. (25)

We let $G' = \left\{ < h^*(u), h^{**}(i), v > \in H | \nexists < h^*\left(u'\right), h^{**}\left(i'\right), v' > \in H, where\ h^*(u) = h^*\left(u'\right) \wedge h^*(i) = h^{**}(i) \wedge\ v \neq v' \right\}$ (26)

3 Results

The proposed method has been tested using two open-source representative CF datasets: MovieLens 1 M (https://grouplens.org/datasets/movielens/1m/) and a subset of Netflix that we call Netflix* [32]. These two datasets were chosen because they are representative in the CF field. MovieLens is probably the most tested dataset family over the years in CF research. The results obtained in MovieLens are very informative for RS researchers. On the other hand, Netflix is not widely used due to its enormous size and because it is no longer available. We utilized the open version of Netflix* that is randomly shortened [33]. Netflix* has been selected as the dataset for this research because its internal patterns are different from those of MovieLens. MovieLens has been created in a relatively short time in an academic environment, whereas Netflix is an enormous commercial dataset that has been growing for a long period. Since this research involves catching the internal patterns of a source dataset and parameterizing and translating those patterns to a generated dataset, it is convenient to use such different sources.

Table 1 shows their main parameter values. The results of both datasets follow the same trends. Therefore, to ensure that the length of this paper is appropriate, we only explain the MovieLens results and group the Netflix* results (Figs. 10, 11, 12 and 13) in Appendix A. To test the WGANRS method, two sets of synthetic datasets have been created. The first set has a varied number of users, whereas the second set has a varied number of items. Each row in Table 2 shows the values of each set. The first row indicates that five synthetic datasets have been generated. The first dataset contains 500 users and 1000 items, the second dataset holds 1000 users and 1000 items, and so on until the last dataset has 8000 users and 1000 items. All the generated datasets were created starting from 400 thousand samples. This set of data allows us to test the impact of changing the number of users. Analogously, the second row in Table 2 shows that four synthetic datasets have been created. All four datasets have 4000 users. However, the number of items varies from 500 in the first dataset to 4000 in the last set. This allows us to measure the impact of changing the number of items.

Since we will use the GANRS method [29] as the baseline, all datasets have also been created using GANRS. Thirty-six datasets were generated, 5 + 4 using MovieLens as a source and 5 + 4 using Netflix*, both for GANRS and for WGANRS. To test these datasets, four different exections were conducted:

- *Number of generated samples*: The number of samples returned by GANRS and WGANRS was compared. The WGANRS method is expected to perform better, as it focuses particularly on avoiding the typical 'mode collapse' in GANs. The better managed the mode collapse is, the more varied the embedding samples that the WGANRS generates, the better the performance of the clustering process to create the raw samples, and finally, the lesser the number of sample collisions, increasing the sizes of the synthetic datasets.
- *Rating distributions*: It is important that the rating distribution of the generated datasets be as similar as possible to that of the source, particularly on the relevant ratings (usually 4 and 5 stars). This is an indication that the patterns of the fake profiles are correct, and they contain an adequate proportion of relevant and nonrelevant votes.
- *User and item distributions*: It is interesting to test the user and item distributions as the number of users and items varies, comparing them to the source dataset. It is expected that the Gaussian random noise used to create the stochastic vector that feeds the WGANRS generator will force the different Gaussian distributions of users and items.
- *Precision and recall*: Regarding the ground distributions, balanced votes, and high number of samples, the generated datasets are used to adequately analyse them on the CF task, and their recommendation quality measures return suitable values and trends.

The above executions cover the potential comparatives available on the generative creation of synthetic datasets in the CF area. Figure 5 shows the concept. Figure 5a (top graph) shows the traditional CF analysis. Different state-of-the-art methods or models are applied to one or several existing CF datasets, and the recommendation results can be measured using traditional quality prediction and recommendation quality metrics, such as the precision, recall, F1, and mean absolute error (MAE). However, the field of synthetic CF dataset generation is completely different. From a source dataset (Fig. 5b), we create one or several synthetic datasets. We can set different numbers of users, items, samples, etc., and each generated dataset will hold. To compare the proposed generative method with the selected state-of-the-art baseline, we must create a synthetic dataset or group of synthetic datasets using the proposed method (orange datasets in Fig. 5b) and a different dataset or group of datasets using the SOTA baseline (blue datasets in Fig. 5b). Now, we need to determine which of these datasets (or groups) are better. Comparing generated datasets (Fig. 5b) is a very different task than comparing methods or models applied to an existing dataset (Fig. 5a). Figure 5b shows three types of code execution that we can perform to decide whether the generated datasets using the proposed method (orange datasets) are better than the generated datasets using the baseline method (blue datasets):

Table 1 Main parameter values of the tested datasets

Dataset	#users	#items	#ratings	scores	sparsity
Movielens 1 M	6,040	3,706	911,031	1 to 5	95,94
Netflix*	23,012	1,750	535,421	1 to 5	98.68

1) The mode collapse impact can be measured in the CF field by removing very similar user profiles. The GAN can collapse to a reduced number of source profiles. The higher the number of deleted profiles is, the higher the mode collapse impact. The higher the deleted profiles are, the lower the number of samples in the generated dataset. Figure 5b1 shows a comparison where the *y*-axis represents the number of samples. The proposed method (orange colour) improves the SOTA baseline.
2) The generated datasets should have probability distributions that are similar to that of the source dataset. The user, item, and rating distributions of the synthetic datasets can be compared. Figure 5b2 shows the distribution of the ratings from the source dataset (green colour) compared to the 'baseline' dataset (orange colour, in this graph) and the 'proposed method' dataset (blue colour). Notably, the synthetic datasets are not expected to have the same distributions as the source dataset. The generative process should create similar datasets, not identical datasets. Identical datasets have no value, just as replicating real faces is not valuable in field of computer vision. Therefore, there is no absolute metric to measure this type of quality. Extreme distribution similarities and large distribution differences must be avoided.
3) The quality of the generated datasets can be indirectly measured by running state-of-the-art CF methods and models on them. We expect similar behaviours to those obtained when we apply these methods to the source dataset. Very different trends or absolute values in the generated dataset graphs, compared to those in the source dataset, will tell us that the generated dataset does not contain the main patterns of the source dataset. Figure 5b3 shows the precision and recall results on the source dataset (left graph) and the generated dataset (right graph) when measured with several SOTA CF deep learning models. Both trends and values are similar. As explained above, we do not expect identical behaviours and values since synthetic datasets should mimic the source patterns and not copy them. For this reason, there is no standard quality measure to compare the graphs in Fig. 5b3.

Taking into account the above considerations, several experiments have been performed on the three explained approaches, as shown in b1, b2, and b3 of Fig. 5. Individual executions and their explained results have been structured in SubSects. 3.1 to 3.4, followed by the Discussion Subsection.

Table 2 Parameter values

initial #samples	#users	#items
400,000	{500, 1000, 2000, 4000, 8000}	1,000
400,000	4,000	{500, 1000, 2000, 4000}

3.1 Number of generated samples

As explained in the previous section, the proposed method uses a WGAN to generate synthetic embedding samples. These dense and continuous samples are then converted to their sparse and discrete versions by means of the clustering process and their translation to the raw tuples in the synthetic dataset. This discretization stage causes a proportion of 'collisions' where identical or similar generated samples must be removed. The smaller the number of samples removed, the more accurate the generative model, and the richer the synthetic dataset. The Wasserstein GAN is expected to improve the results, as it is designed to prevent mode collapse inherent to the GAN models. Please note that the hypothesis is that by reducing the mode collapse, the variability of the generated embedded samples will increase, and then the clustering process will be able to spread users and item IDs in a more homogeneous way. Consequently, the number of discretized samples that are repeated (and deleted) will decrease. Overall, the total size of the generated datasets will increase as the GAN mode collapse is reduced using the Wasserstein approach.

The final number of samples generated is a relevant quality measure since it is directly related to the impact of mode collapse in the generative process. The baseline GANRS method suffers from the mode collapse problem, leading to the generation of repeated fake profiles. The method handles this situation by removing spare profiles, but it does not provide the necessary diversity of samples. The proposed WGANRS is expected to improve the results due to the Wasserstein ability to reduce the mode collapse and then to improve diversity and increase the synthetic dataset size.

Figure 6 shows the comparison of GANRS (gan) versus WGANRS (wgan) both for synthetic datasets where the number of users varies (left graph) and for synthetic datasets where the number of items varies (right graph). Overall, the proposed approach (wgan) significantly improves the baseline (gan). Specifically, it duplicates the number of generated samples. A 213% improvement in the left graph and a 191% improvement in the right graph are achieved. This is a relevant predictor of the superiority of the proposed method. Additionally, as expected, the higher the number of users or items there are, the higher the number of generated samples. This is because the clustering process can better spread the samples in the latent space as the number of centroids increases. Then, the number of duplicated samples decreases.

The results show a relevant improvement when the proposed method is applied compared to the baseline. This confirms that the paper hypothesis is fulfilled. Moreover, incorporating the Wasserstein concept into the generative kernel of the GANRS method will lead to a decrease in the mode collapse problem inherent to the GAN when applied to CF scenarios. Generated datasets have less redundant profiles. Accordingly, they are more diverse, and they contain more

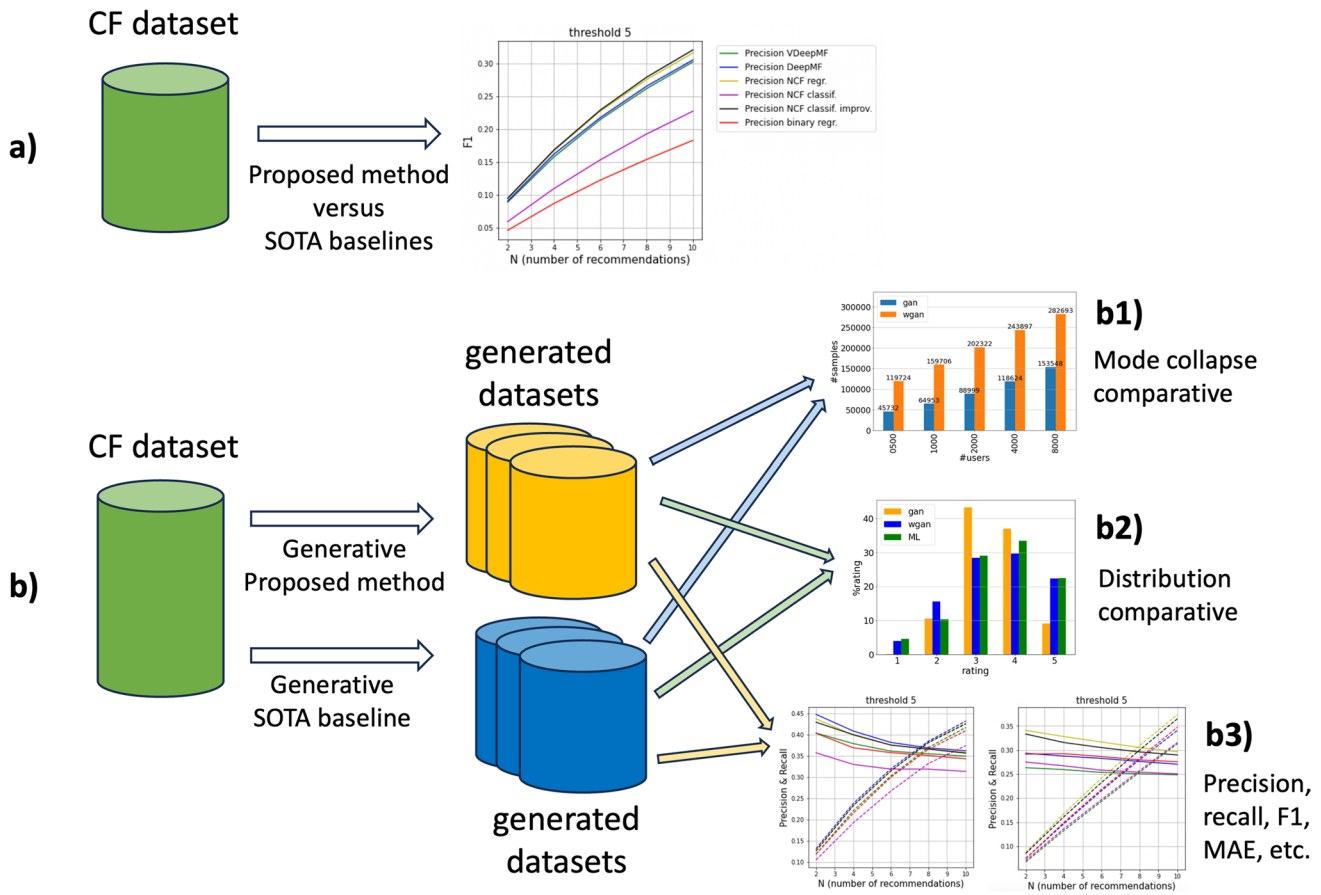

Fig. 5 Traditional CF validation of the methods and models versus the validation of the synthetic datasets generated by the GAN

samples. Overall, the proposed WGANRS method generates richer, unbiased, and longer synthetic datasets.

3.2 Rating distributions

The distribution of the ratings (one star, two stars, …, five stars) is an important quality measure in the CF synthetic dataset generation process. Recommendation models are very sensitive to the relevant versus non-relevant threshold, which is usually set to four or five stars in CF datasets containing five possible ratings. It is not enough that the Gaussian distribution of ratings in the generated dataset has a similar mean to the Gaussian distribution in the source dataset. It is also necessary that their standard deviation be analogous. Figure 7 shows that the proposed WGANRS generates a Gaussian distribution more similar to the MovieLens distribution than the baseline GANRS. Specifically, it achieves a 271.21% improvement. The improvement average obtained using the synthetic datasets in the first row of Table 2 (the number of users varies) is 304% (541% in Netflix*), whereas the second row (the number of items varies) returns a 357% improvement on average (399% in Netflix*). It is expected that these positive results will contribute to providing adequate recommendation quality results in the next subsection.

Beyond the numeric improvement values shown before, we can compare the shapes of the probability distribution in Fig. 7. The probability distribution of the source MovieLens dataset (green-colour bars) is the target. The proposed WGANRS method (blue-colour bars) is much closer to the target than the baseline GANRS method (orange-colour bars). This is the reason for the relevant numerical improvements shown in the above paragraph. Additionally, the baseline method generates a Gaussian distribution excessively centred in the average rating (three stars), whereas the proposed method adequately fits its Gaussian distribution to the correct four-star mean. Regarding the Gaussian standard deviation, the baseline method does not adequately catch the source dataset shape. Its deviation is smaller, and consequently, it does not generate enough profiles in the distribution edges (one star and five stars). In contrast, the proposed method performs nearly perfectly on both edges of the source distribution. Thus, the samples generated using the proposed WGANRS method are more diverse and unbiased than those obtained running the SOTA baseline. Additionally, the obtained result better follows the Gaussian distribution that describes the source shape of ratings. This result reinforces and

complements that obtained in Sect. 3.1. Overall, the proposed method a) reduces repeated samples, b) generates more samples, c) increases diversity, d) decreases the bias, and e) better mimics the probability distributions of the ratings.

3.3 Precision and recall

In this subsection, we show the recommendation quality results obtained on synthetic datasets obtained using MovieLens as a source. The WGANRS method was used for this experiment to generate the synthetic datasets. The state-of-the-art NCF (neural collaborative filtering) deep learning model has been used to make predictions and recommendations. The relevancy θ threshold was set to five. The top graphs in Fig. 8 show the results when the number of users varies, whereas the bottom graphs show the results when the number of items varies. Both the values and the trends obtained from the synthetic datasets (coloured curves) are similar and compatible with the source datasets (black curves), which indicates that the proposed method generates suitable synthetic datasets to be used in the RS field. Additionally, as expected, the higher the number of users, the higher the recall is, since each user profile will contain fewer relevant ratings (recall denominator). Conversely, the higher the number of users, the lower the precision is, since the denominator is the constant N (number of recommendations), whereas the numerator contains the true positives of relevant ratings, where a high number of users involves less relevant ratings per user.

Additionally, from the set of synthetic datasets where the number of users varies, the dataset that holds 1000 users provides more precision and recall results like the MovieLens source. Since MovieLens 1 M contains 6000 users, it tells us that the GAN-based method generates data patterns where recommendations are easier than using the source dataset. This result is consistent with the one reported in [30]. Most importantly, the evolution of all the recommendation curves in the generated datasets (coloured curves) follow the same trends as those exhibited by the source MovieLens (black curve), indicating that the internal patterns of the source dataset have been adequately captured by the proposed WGANRS method. Regarding the results when the number of items varies, similar conclusions can be drawn, underlying that recommendation qualities worsen in absolute values compared to the source dataset. This probably occurs because the distribution of the item ratings is highly variable compared to the distribution of the user ratings, leading to more difficult pattern extraction. There is a number of items holding a very low number of ratings.

3.4 User and item distributions

Once the rating distributions have been tested, it is also convenient to compare the user and the item distributions obtained by using both the proposed and the baseline methods. The user and item distributions of the synthetic datasets are very dependent on the Gaussian parameter values with which the noise vectors that feed the generative model have been created. In the original paper [28] that serves as a baseline, the standard deviation has been customized for each tested dataset. In contrast, by using the proposed method, we fixed it to one and then removed this hyperparameter, making it easier to fine tune the proposed approach compared to the baseline method. The top graph in Figure 9 shows the results when the number of users varies, while its bottom graph shows the result by varying the number of items. Dashed lines represent the baseline results, and solid lines show the proposed approaches. In all cases, as expected, the higher the number of users there are, the lower the number of ratings assigned to

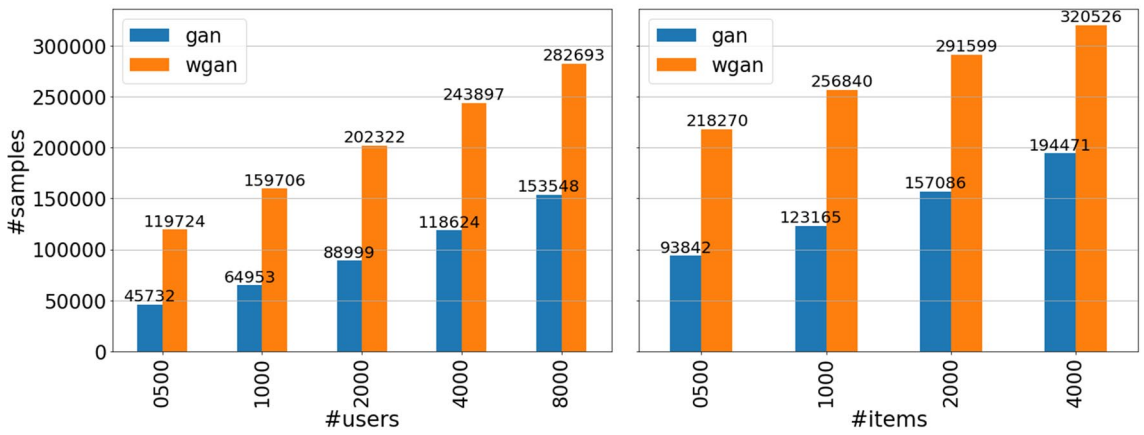

Fig. 6 Number of samples generated using the baseline GANRS method (gan) versus the proposed WGANRS method (wgan). Source dataset: MovieLens 1 M. Number of samples needed: 40,000. Left graph: generated datasets with 1000 items and a range of 500, 1000, 2000, 4000 and 8000 users. Right graph: generated datasets with 4000 users and a range of 500, 1000, 2000 and 4000 items. The higher the number of generated samples, the better the model is

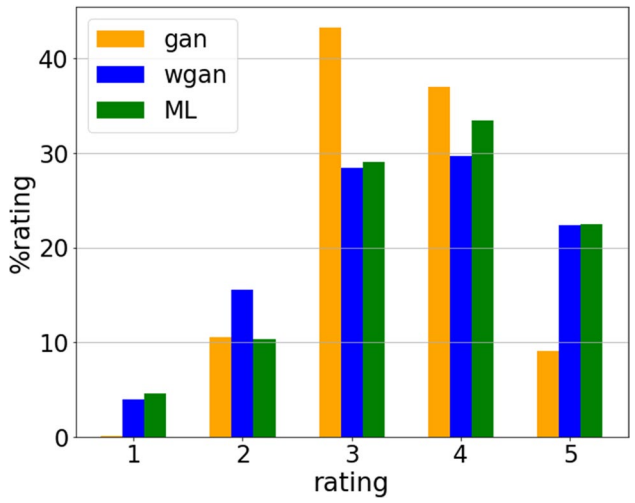

Fig. 7 Comparative rating distributions among the MovieLens 1 M (ML) source dataset, the baseline GANRS method (gan), and the proposed WGANRS method (wgan). The 8000-user and 1000-item synthetic dataset has been chosen as a representative case from the set of generated data in the paper. The closer the distribution is to the source ML distribution, the better the model is

each user, since the number of ratings in each dataset is fixed. It can also be observed that the proposed method generates Gaussian distributions with higher standard deviations than the baseline approach, which has been heuristically tailored to the dataset. Both the proposed and baseline methods generate suitable user and item distributions for the CF area.

3.5 Discussion

The experimental results show the superiority of the proposed WGANRS method compared to the GANRS baseline. Particularly relevant is the high improvement (approximately 200%) in the number of generated samples. This indicates that the proposed Wasserstein approach effectively reduces the amount of mode collapse of the GAN, The WGANRS method also effectively mimics the rating distribution of the source dataset, obtaining high improvements compared to the baseline and making it possible that their quality precision and recall values and trends are compatible with those from the source dataset. Furthermore, even no standard quality measures exist to test RS generated data, the user and item distributions obtained using the proposed approach are comparable to those of the baseline method. Additionally, the proposed method has the advantage that it is not necessary to assign heuristic values to the standard deviation of the Gaussian distribution used to create the noisy random vectors that feed the generator model of the WGAN. Finally, the results using the Netflix* dataset reinforce the results obtained by testing MovieLens. Appendix A shows the Netflix* results.

Overall, the proposed method improves both the statistical baselines and state-of-the-art generative methods. Statistical baselines are reported to reach poor accuracy. In contrast, they support adequate parameterization. Generative baselines operate quite differently. They do not support full parameterization and exceed the accuracy of statistical methods [24]. Our proposed method is proven to provide both full

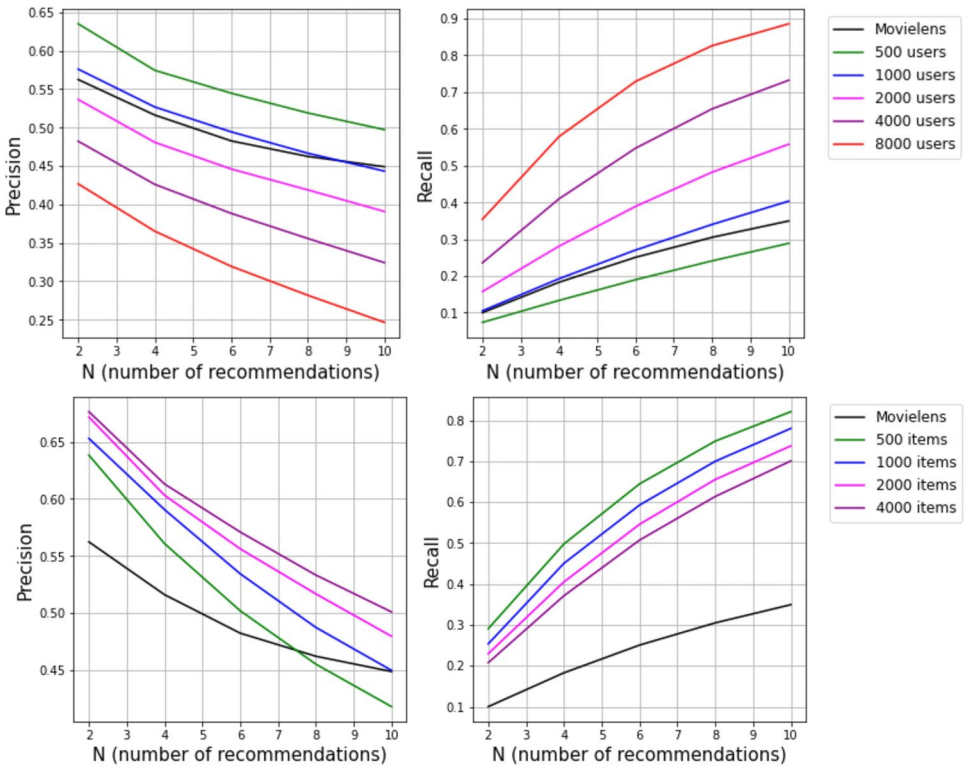

Fig. 8 Quality of the recommendation: precision and recall obtained by varying the number N of recommendations from 2 to 10. The relevancy threshold θ was set to 5. The upper graphs show the results on the synthetic datasets containing 500 to 8000 users. The lower graphs show the results on the synthetic datasets containing 500 to 4000 items. Precision can be seen in the left graphs, whereas recall is shown in the right graphs. The MovieLens dataset was used. The higher the values are, the better the results

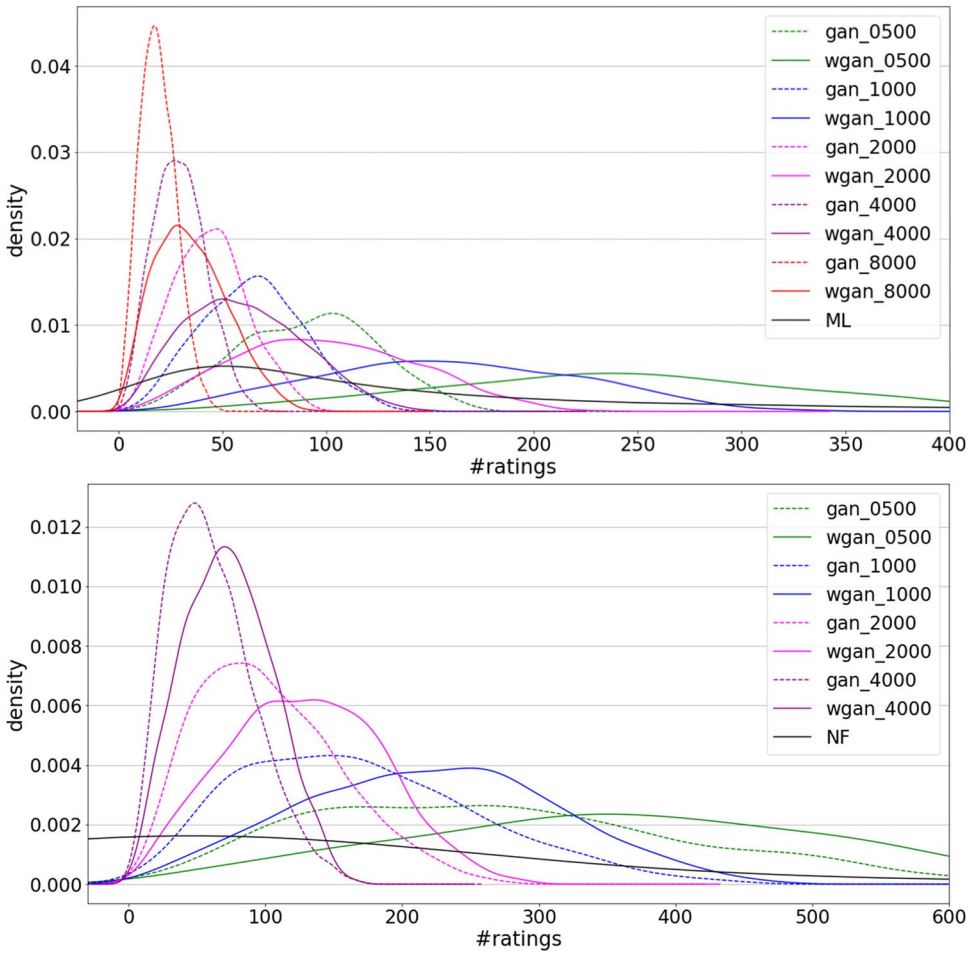

Fig. 9 Top graph: distribution of the ratings when the number of users varies from 500 to 8000; comparative of the proposed WGANRS (wgan) method and the baseline GANRS (gan) method. Bottom graph: distribution of ratings when the number of items varies from 500 to 4000; comparison of the proposed WGANRS (wgan) method and the baseline GANRS (gan) method. The source MovieLens (ML) dataset is used in both graphs

parameterization and high accuracy, in addition to a strong reduction of the mode collapse problem inherent to the GAN architectures. On the other hand, our method inherits the most positive and the most negative features of its baseline [30]. However, its accuracy and performance are very high due to the short, dense, and continuous vectors that its GAN model takes as input. Its main drawback comes from the clustering stage of the method (Fig. 4), which requires additional execution time and involves a discretization process that increases the probability of generating duplicated samples. For this reason, the Wasserstein concept has been proposed to alleviate the explained drawback. The results show that the proposed method adequately reduces the mode collapse problem, maintains the baseline advantages, reduces its disadvantages, and confirms the hypothesis of this paper.

4 Conclusions

The most relevant conclusion in this paper is that the Wasserstein approach reduces the mode collapse in the GAN generation of the CF fake samples compared to the state-of-the-art methods. This positive effect is reflected in a relevant reduction in duplicated samples and consequently in the generation of larger synthetic datasets. Furthermore, the proposed approach returns very improved distributions of ratings, which facilitates obtaining correct values and trends in recommendation quality measures. Finally, the distributions of the users and items are comparable to those of the state-of-the-art methods; these distributions act as quality measures due to the lack of standard quality measures for RS generated data. Moreover, existing hyperparameters are avoided in the proposed method. The standard deviation of the Gaussian distribution is used to create the noisy vectors that feed the generator model in the GAN. Overall, the results of the experiment show that by applying the Wasserstein distance and weight clipping to CF data, the generative process is improved compared to the state-of-the-art methods that use Wasserstein-based GANs. Proposed future work includes a) testing the proposed method on different RS datasets, with several sparsity ratios and different numbers of users or items, b) comparing the existing biases in the source datasets with the generated biases in the synthetic datasets, and c) checking the ability of the generated samples to serve as data augmentation when they are added to the source datasets.

Appendix

In this section, the same figures used for MovieLens are shown. However, in this case, the Netflix* dataset has been used as a source.

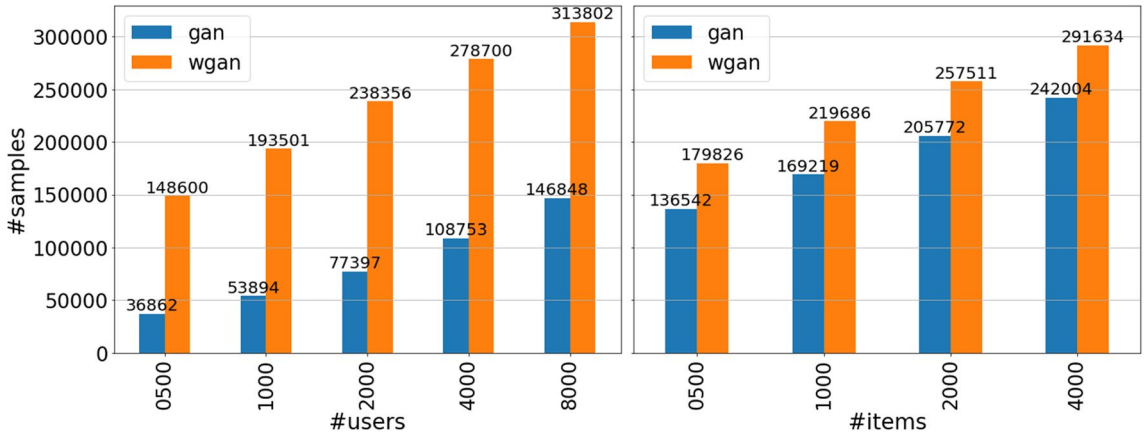

Fig. 10 Number of samples generated using the baseline GANRS method (gan) versus the proposed WGANRS method (wgan). Source dataset: Netflix*. Number of needed samples: 40,000. Left graph: generated datasets with 1000 items and a range of 500, 1000, 2000, 4000 and 8000 users; right graph: generated datasets with 4000 users and a range of 500, 1000, 2000 and 4000 items. The higher the number of generated samples, the better the model is

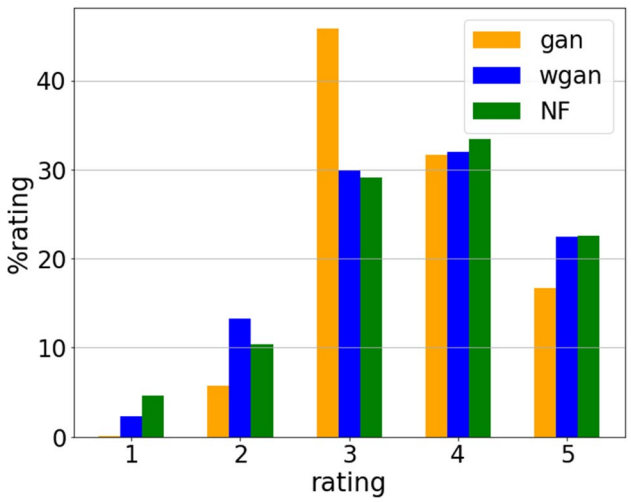

Fig. 11 Comparative rating distributions among the Netflix* source dataset, the baseline GANRS method (gan) and the proposed WGANRS method (wgan). The 8000-users and 1000-items synthetic dataset has been chosen as a representative case from the set of generated data in the paper. The closer the distribution is to the source ML distribution, the better the model is

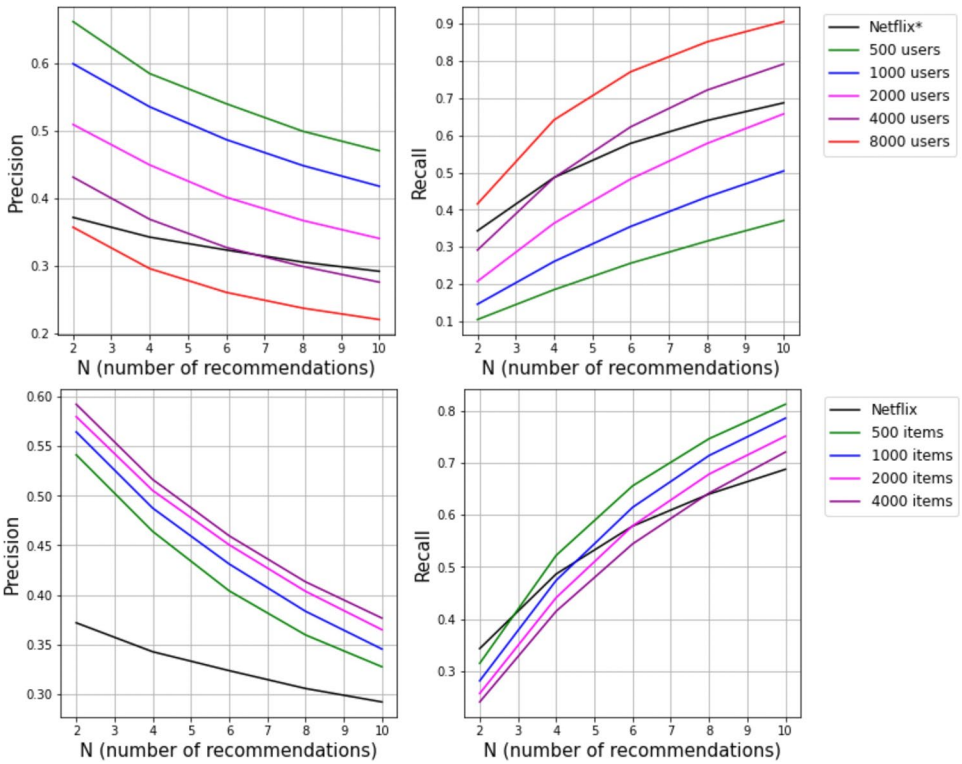

Fig. 12 Quality of recommendation: precision and recall obtained by varying the number N of recommendations from 2 to 10. The relevancy threshold θ was set to 5. The upper graphs show the results on the synthetic datasets containing 500 to 8000 users. The lower graphs show the results on the synthetic datasets containing 500 to 4000 items. Precision can be seen in the left graphs, whereas recall is shown in the right graphs. The Netflix* dataset was used. The higher the values are, the better the results

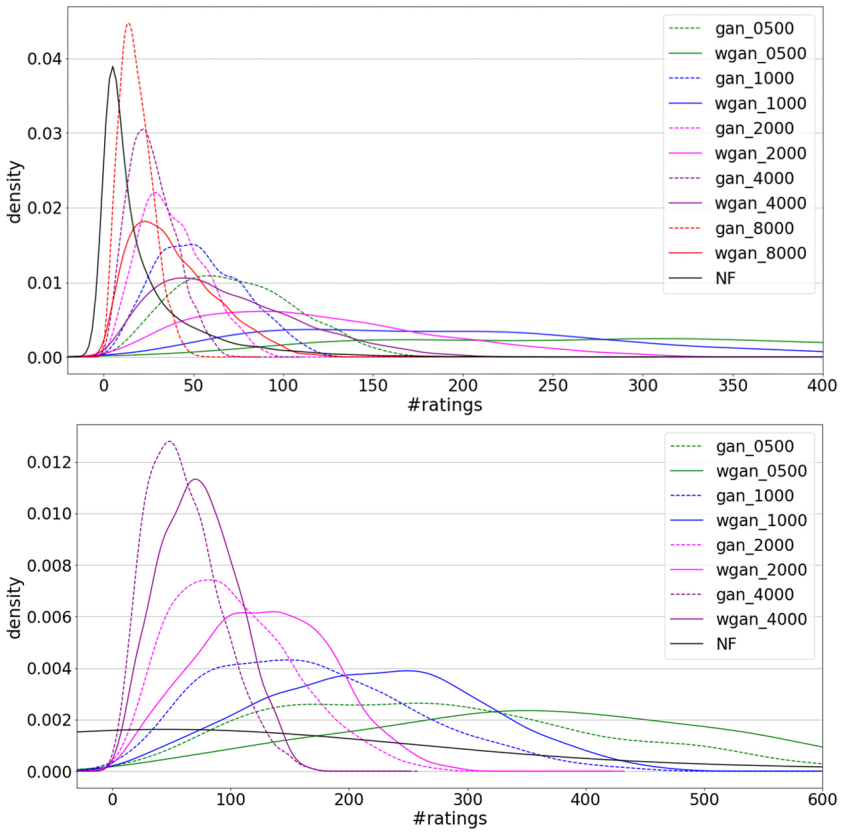

Fig. 13 Top graph: distribution of the ratings when the number of users varies from 500 to 8000. Comparison of the proposed WGANRS (wgan) method and the baseline GANRS (gan) method. Bottom graph: distribution of ratings when the number of ratings varies from 500 to 4000. Comparison of the proposed WGANRS (wgan) method and the baseline GANRS (gan) method. The source Netflix* dataset is used in both graph

Author contributions *Abraham Gutiérrez* ran most of the executions and prepared the figures and the paper format.

Jesús Bobadilla provided the paper concept, the model design, the experimental design, and wrote the paper.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This work was partially supported by the 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.

Data availability The datasets generated during and/or analysed during the current study are available in the GitHub repository, https://github.com/jesusbobadilla/ganrs.git

Declarations

Ethics for obtaining the data In this paper, all the conditions specified for the use of the open datasets taken as a source for the generative process are satisfied, including the reference to the paper stated in the README file.

Conflicts of interest The authors have no competing interests to declare that are relevant to the content of this article and agree to the publishing of its content.

References

1. Shokeen J, Rana C (2020) A study on features of social recommender systems. Artif Intell Rev 53(2):965–988. https://doi.org/10.1007/s10462-019-09684-w
2. Bobadilla J, Gutiérrez A, Alonso S, González-Prieto A (2022) Neural Collaborative Filtering Classification Model to Obtain Prediction Reliabilities. International Journal of Interactive Multimedia and Artificial Intelligence 7(4):18–26. https://doi.org/10.9781/ijimai.2021.08.010
3. Deldjoo Y, Schedl M, Cremonesi P, Pasi G (2020) Recommender systems leveraging multimedia content. ACM Computing Surveys (CSUR) 53(5):1–38. https://doi.org/10.1145/3407190
4. Bobadilla J, González-Prieto A, Ortega F, Lara-Cabrera R (2021) Deep learning feature selection to unhide demographic recommender systems factors. Neural Comput Appl 33(12):7291–7308. https://doi.org/10.1007/s00521-020-05494-2
5. Bobadilla J, Lara-Cabrera R, González-Prieto Á, Ortega F (2021) DeepFair: Deep Learning for Improving Fairness in Recommender Systems. International Journal of Interactive Multimedia and Artificial Intelligence 6(6):86–94. https://doi.org/10.9781/ijimai.2020.11.001
6. Kulkarni S, Rodd SF (2020) Context aware recommendation systems: A review of the state of the art techniques. Computer Science Review 37:100255. https://doi.org/10.1016/j.cosrev.2020.100255
7. Wang Z (2023) Intelligent recommendation model of tourist places based on collaborative filtering and user preferences. Appl Artif Intell 37(1):2203574. https://doi.org/10.1080/08839514.2023.2203574
8. Ray B, Garain A, Sarkar R (2021) An ensemble-based hotel recommender system using sentiment analysis and aspect categorization of hotel reviews. Applied Soft Computing 98:106935. https://doi.org/10.1016/j.asoc.2020.106935
9. Kabul MS, Setiawan EB (2022) Recommender System with User-Based and Item-Based Collaborative Filtering on Twitter using K-Nearest Neighbors Classification. Journal of Computer System and Informatics 3:478–484. https://doi.org/10.47065/josyc.v3i4.2204
10. Eslami G, Ghaderi F (2023) Incremental trust-aware matrix factorization for recommender systems: towards Green AI. Appl Intell 53:12599–12612. https://doi.org/10.1007/s10489-022-04150-7
11. Mehdi HA (2022) A novel constrained non-negative matrix factorization method based on users and items pairwise relationship for recommender systems. Expert Syst Appl 195:116593. https://doi.org/10.1016/j.eswa.2022.116593
12. Gheorghe P, Pérez-Jiménez M, Grzegorz R (2023) Infinite Spike Trains in Spiking Neural P Systems. Romanian Journal of Information Science and Technology 2023:251–275. https://doi.org/10.59277/ROMJIST.2023.3-4.01
13. Liu H, Zheng C, Li D, Shen X, Lin K, Wang J, Zhang Z, Zhang Z, Xiong N (2021) EDMF: Efficient Deep Matrix Factorization With Review Feature Learning for Industrial Recommender System. IEEE Trans Industr Inf 18(7):4361–4371. https://doi.org/10.1109/TII.2021.3128240
14. Bobadilla J, Ortega F, Gutiérrez A, González-Prieto Á (2022) Deep variational models for collaborative filtering-based recommender systems. Neural Comput Appl 35:7817–7831. https://doi.org/10.1007/s00521-022-08088-2
15. Hai C, Fulan Q, Jie C, Shu Z, Yanping Z (2021) Attribute-based Neural Collaborative Filtering. Expert Syst Appl 185:115539. https://doi.org/10.1016/j.eswa.2021.115539
16. Min G, Junwei Z, Junliang Y, Jundong L, Junhao W, Qingyu X (2021) Recommender systems based on generative adversarial networks: A problem-driven perspective. Inf Sci 546:1166–1185. https://doi.org/10.1016/j.ins.2020.09.013
17. Forouzandeh S, Berahmand K, Rostami M (2021) Presentation of a recommender system with ensemble learning and graph embedding: a case on MovieLens. Multimedia tools and applications 80(5):7805–7832. https://doi.org/10.1007/s11042-020-09949-5
18. Kumar A, Aggarwal RK (2022) An exploration of semi-supervised and language-adversarial transfer learning using hybrid acoustic model for hindi speech recognition. J Reliable Intell Environ 8:117–132. https://doi.org/10.1007/s40860-021-00140-7
19. Deldjoo Y, Noia DT, Merra FA (2021) Survey on Adversarial Recommender Systems: From Attack/Defense Strategies to Generative Adversarial Networks. ACM Comput Surv 54(2):1–38. https://doi.org/10.1145/3439729
20. Chae DK, Kang JS, Kim SW, Lee JT (2018) CFGAN: a generic collaborative filtering framework based on generative adversarial networks. In: Proceedings of the 27th, ACM International Conference on Information and Knowledge Management, CIKM 2018. Association for Computing Machinery, New York, NY, pp 137–146. https://doi.org/10.1145/3269206.3271743
21. Guo G, Zhou H, Chen B et al (2022) IPGAN: Generating informative item pairs by adversarial sampling. IEEE Transactions on

Neural Networks and Learning Systems 33(2):694–706. https://doi.org/10.1109/TNNLS.2020.3028572
22. Zhao J, Li H, Qu L, Zhang Q, Sun Q, Huo H, Gong M (2022) DCF-GAN: An adversarial deep reinforcement learning framework with improved negative sampling for session-based recommender systems. Inf Sci 596:222–235. https://doi.org/10.1016/j.ins.2022.02.045
23. Sun J, Liu B, Ren H, Huang W (2022) WNCGAN: A neural adversarial collaborative filtering for recommender system. Journal of intelligent & fuzzy systems 42(4):2915–2923. https://doi.org/10.3233/jifs-210123
24. Bharadhwaj H, Park H, Lim BY (2018) RecGAN: recurrent generative adversarial networks for recommendation systems. In: Proceedings of the 12th ACM Conference on Recommender Systems, RecSys september 2019. Association for Computing Machinery, New York, NY, pp 372–376. https://doi.org/10.1145/3240323.3240383
25. Shafqat W, Byun YC (2022) A Hybrid GAN-Based Approach to Solve Imbalanced Data Problem in Recommendation Systems. IEEE access 10:11036–11047. https://doi.org/10.1109/ACCESS.2022.3141776
26. Wen J, Zhu XR, Wang CD, Tian Z (2022) A framework for personalized recommendation with conditional generative adversarial networks. Knowl Inf Syst 64(10):2637–2660. https://doi.org/10.1007/s10115-022-01719-z
27. Wang Q, Huang Q, Ma K, Zhang X (2021) A Recommender System Based on Model Regularization Wasserstein Generative Adversarial Network. Inf Sci 546:1166–1185. https://doi.org/10.1016/j.ins.2020.09.013
28. Zhang X, Zhong J, Liu K (2021) Wasserstein autoencoders for collaborative filtering. Neural Comput Appl 33(7):2793–2802. https://doi.org/10.1007/s00521-020-05117-w
29. Schlett T, Rathgeb C, Henniger O, Galbally J, Fierrez J, Busch C (2022) Face Image Quality Assessment: A Literature Survey. ACM Comput Surv 54(10):1–49. https://doi.org/10.1145/3507901
30. Bobadilla J, Gutiérrez A, Yera R, Martínez L (2023) Creating Synthetic Datasets for Collaborative Filtering Recommender Systems using Generative Adversarial Networks. Knowledge Based Systems 280(1):111016. https://doi.org/10.1016/j.knosys.2023.111016
31. Ioan-Daniel B, Radu-Emil P, Alexandra-Bianca B (2022) Improvement of K-means Cluster Quality by Post Processing Resulted Clusters. Procedia Computer Science 199:63–70. https://doi.org/10.1016/j.procs.2022.01.009
32. Ortega F, Mayor J, López-Fernández D, Lara-Cabrera R (2021) CF4J 2.0: adapting collaborative filtering for java to new challenges of collaborative filtering based recommender systems. Knowledge-Based Syst 215(4):106629. https://doi.org/10.1016/j.knosys.2020.106629
33. Gong Y (2023) Distribution constraining for combating mode collapse in generative adversarial networks. J Electron Imaging 32(4):43029–43030. https://doi.org/10.1117/1.JEI.32.4.043029

Jesús Bobadilla received the B.S. and the Ph.D. degrees in computer science from the Universidad Politécnica de Madrid and the Universidad Carlos III. Currently, he is a full professor with the Department of Information Systems, Universidad Politécnica de Madrid. He is a habitual author of programming languages books working with McGraw- Hill, Ra-Ma and Alfa Omega publishers. His research interests include information retrieval, recommender systems and speech processing. He oversees the FilmAffinity.com research team working on the collaborative filtering kernel of the web site. He has been a researcher into the International Computer Science Institute at Berkeley University and into the Sheffield University.

Abraham Gutiérrez received the B.S. and the Ph.D. degrees in computer science from the Universidad Politécnica de Madrid. Currently, he is currently an associate professor with the Department of Information Systems, Universidad Politécnica de Madrid. He is the author of research papers in most prestigious international journals. He is a habitual author of programming languages books working with McGraw-Hill, Ra-Ma and Alfa Omega publishers. His research interests include P-Systems, machine learning, data analysis and artificial intelligence. He is in charge of this group innovation issues, including the commercial projects.