Recommender Systems: Learning Collaborative Filtering Similarity Measures Using Siamese Networks

Jesús Bobadilla, Abraham Gutierrez *

Universidad Politécnica de Madrid, Dpto. Sistemas Informáticos, Madrid (Spain)

* Corresponding author: jesus.bobadilla@upm.es (J. Bobadilla), abraham.gutierrez@upm.es (A. Gutierrez)

Received 31 January 2024 | Accepted 1 March 2025 | Published 21 March 2025

Abstract

Improving current similarity measures in the collaborative filtering Recommender Systems is relevant, since it contributes to different applications such as to get better big data representations of users and items, to implement dynamic browsers able to navigate through data, and to explain recommendation results. Currently, there are many statistically based similarity measures, some of them tailored to the extraordinarily sparse collaborative filtering scenario. Nevertheless, the hypothesis of the paper is that using neural networks, learnt similarity measures can be obtained that improve existing ones. To accomplish the task, the typical neural models cannot be used, and it is necessary to focus on the similarity learning area, in which the goal is to make the model learn, which is a similarity function able to measure how similar two objects are. Siamese networks adequately implement the similarity learning concept, and we have adapted them to collaborative filtering particularities. The results in different scenarios show significant improvements compared to the state-of-the-art.

Keywords

Collaborative Filtering,
Neural Networks,
One-Hot Encoding,
Recommender Systems,
Siamese Networks,
Similarity Measures.

DOI: 10.9781/ijimai.2025.03.006

I. Introduction

Recommender Systems (RS) [1] is the Artificial Intelligence area focused on personalization. RS recommend products or services to users. Remarkable commercial RS are Spotify, TripAdvisor, Netflix, TikTok, etc. To accomplish their task, RS can use text and images of the items (products or services), so they could recommend a Sci-Fi film based on the similarity between its synopsis and the synopsis of some other films the user liked; this is content-based filtering. There are some other filtering strategies, such as demographic filtering [2] which recommends to an active user the products that users of the same age, sex, nationality, etc. consumed. Social filtering is based on followed, followers, and trusted information [3]. Context-based filtering usually makes use of geographical information [4], such as GPS coordinates. The most accurate filtering strategy is Collaborative Filtering (CF) [5]. CF makes use of datasets that contain all the iterations between users and items; typically, they hold the explicit votes that users cast to items, or the implicit interactions between users and items, such as listened to songs, watched movies, bought products, etc. The most accurate RSs combine several filtering strategies using ensemble architectures.

The research in this paper is focused on CF RS, so we will act on data sets containing ratings assigned by users to items. This information can be stored in a bidimensional matrix where each row represents a user, each column represents an item, and each value represents an explicit vote or an implicit rating. Since users can only vote or consume a tiny proportion of the available items, the CF matrices are extraordinarily sparse [6], usually around 98% sparsity. It is relevant in this paper since we will try to design a neural model capable of measuring the existing similarity between users, where each user is represented by a sparse vector of ratings. Accurately measuring similarities between sparse vectors is much more difficult than using dense vectors.

The first CF approaches made use of the K-Nearest Neighbors (KNN) algorithm [7]. It directly implements the CF concept: 1) to find the neighbors of the active user, 2) based on the set of neighbors, to predict the ratings of those items not voted for the active user, and 3) to recommend the N highest predictions. The key to improving KNN accuracy is to design a suitable similarity measure between profile vectors and use it to find the neighbors of the active user. The better the similarity measure, the higher the accuracy. Currently, recommendations are made using machine learning matrix factorization, and deep learning models such as DeepMF [8] and Neural Collaborative Filtering [9]; they largely improve accuracy compared to KNN, and their performance is better, since once the model has been trained predictions are processed very fast.

Beyond accuracy, there are many objectives in RS, such as novelty [10], diversity [11], trust [12], recommendation explanation [13], big data analysis [14], and information browser design [15]. Most of them can take advantage of improving similarity measures to find similar users or similar items. Some CF similarity measures have been borrowed from the statistical field: Pearson correlation, cosine, sine, Jaccard, MSD, etc. whereas some others have been heuristically

Please cite this article as:
J. Bobadilla, A. Gutierrez. Recommender Systems: Learning Collaborative Filtering Similarity Measures using Siamese Networks, International Journal of Interactive Multimedia and Artificial Intelligence, vol. 9, no. 6, pp. 21-27, 2026, http://doi.org/10.9781/ijimai.2025.03.006

designed to fit some of the CF constraints, basically sparsity [6], and cold start [16]: JMSD, PIP [17], Singularities, Significances, etc.

The objective of our research is, precisely, to obtain better similarity measures to improve RS beyond accuracy aims. The hypothesis of the paper is that we can train neural network models to automatically learn the similarity measure that better adapts to the provided CF data. The neural networks are expected to find complex nonlinear patterns that relate users or items in each CF dataset, and their results will be more accurate than those returned when using existing similarity measures. Note that, unlike existing similarity measures, the neural approach requires individual trainings for different CF datasets, since each dataset will hold its own patterns. This customization and the ability of neural networks to find non-linear patterns are the pillars supporting our hypothesis.

Similarity learning is a field of neural networks in which the model is trained to predict similarity between objects. It can be explained by means of an example: let us imagine a company requiring a facial identification system to grant access to the offices; traditionally it would need a high number of pictures of each of their employees to train a neural network classifier. Additionally, each time a new employee joins to the company, the model must be retrained to incorporate the new category; this approach is not adequate. Similarity learning does not classify, and usually it only needs a sample to learn (a picture of each employee, in our example), because it makes use of the one-shot learning concept, where the model only needs a sample of each category to be trained. A similarity learning neural model predicts distances; specifically, it learns that the distance between a sample and some augmented version of it will be shorter than the distance between this sample and other samples. In our company example, we do not longer need to ask for many pictures of each employee, instead we will use image augmentation, and we do not need to retrain the model for each new employee, since the model can also apply the learnt similarity measure to the unseen pictures. Please note the similitude between the mentioned example and the CF scenario, where we only have one sample (vector of ratings) of each user (one-shot learning), and where the model is not retrained each time a new user registers and votes in the RS.

Siamese networks are designed to implement similarity learning. They contain two Multi-Layer Perceptron (MLP) to code pairs of samples (Fig. 1): a sample and its augmented version, associated to a short distance, e.g. label 0, or two different samples, associated to a large distance, e.g. label 1. The MLPs are called 'towers', and they share weights, since both the samples and their augmented versions contain similar patterns. Thus, a unique tower is defined in the neural model and two tower instances feed the architecture. Note that in the CF sparse context, the MLP can be replaced for an embedding layer that codes the discrete and sparse ratings to continuous and dense embedding representations. The coded outputs of both towers are usually merged using the Euclidean distance (Fig. 1), and then the distance is processed to convert it to the prediction of the similarity measure. The contrastive loss function is used to train the Siamese network; it contains a term to be applied if the given samples are similar and another term to be applied if they are dissimilar.

Siamese networks have currently begun to be used as a model to improve RS accuracy. Usually, their results are combined in one of the following three ways [18]: 1) feedforward, where Siamese distances are joined with any of the existing CF models, 2) clustering, where distances are used as a feature vector, and 3) learning-to-rank, where the Siamese output information reorders the recommendation list. Research examples of the feedforward approach are: a) the MOOCs RS, where distances between courses and students are used to improve CF results [19], and b) the use of graph-based dynamic matching for CF where neighborhoods are updated with the information of Siamese homogeneous graphs [20]. A Siamese generative adversarial prediction network has been designed to learn the data distribution characteristic of a HiDS matrix and then build a model to estimate the unknown entries [21]. Siamese networks have also been used to improve content-based recommendations, such as in the fashion area [22] or the music one [23], but they focus on computer vision and signal processing rather than in CF information. Whereas all these approaches test the recommendation accuracy improvements, the quality of the learnt similarity is not tested in isolation, as it is necessary for the objectives of our paper: mostly, big data and recommendation explanation. For this reason, the experiments in this paper specifically test the Siamese similarity results and not the recommendation ones.

The rest of the paper is structured as follows: Section II explains the proposed model and its formalization, Section III contains the experiments design, the obtained results, and their explanation, Section IV highlights the main conclusions and proposes several future works. Finally, the references section contains current and representative papers related to this research.

II. Model

This paper proposes a Siamese model-based network to implement the similarity learning concept. The key idea here is that our model will learn a neural similarity measure capable of providing the distance between user profiles (vectors of ratings). It is expected that a learnt measure will provide more accurate results than the classical statistical ones: cosine, sine, MSD, JMSD, correlation, etc. Training, validation, and testing samples have been set up to allow the neural network to learn and test results. For each sample (vector of ratings casted from a user) some augmented sample variations are randomly created, then the Siamese model will learn that the distance between the active sample and their augmented versions will be small, whereas the distance between the active sample and other user samples will be larger than before. Feeding our Siamese model with hundreds of thousands of such training pairs, it will learn the expected similarity measure.

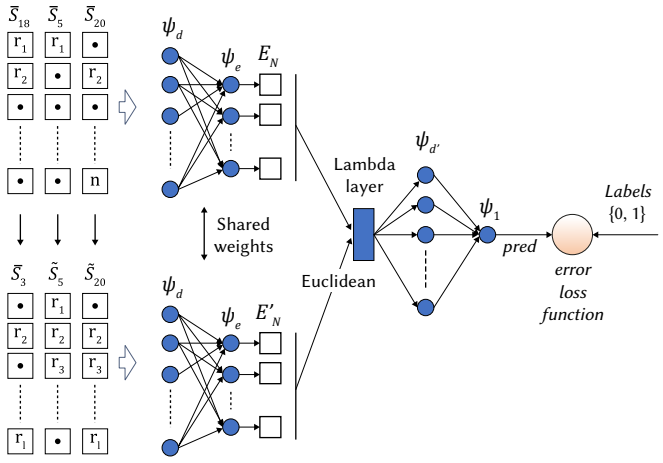

Fig. 1. Proposed Siamese neural network model.

Our model architecture is shown in Fig. 1. The symbols included in it correspond to the formalization listed below. The model includes two data paths, called towers, that join in a Lambda layer. It is important to highlight that both neural towers share their weights; they are implemented using a unique MLP and two instances that use it. The training data, shown on the left side of Fig. 1, feeds the model with pairs of vectors. Randomly disposed, half of the samples will provide, as pairs, a user vector ($\bar{s}_u$) and an augmented version of him ($\tilde{s}_u$); e.g. the first sample in the Fig. 1, containing $\bar{s}_{20}$ and $\tilde{s}_{20}$.

The associated label is 0, and the aim is that the Siamese network learns that a variation of a sample should be predicted to have a short distance from its source. A variability parameter θ sets the number of ratings that randomly will be chosen to change their values in the source sample to generate the augmented one. These values are also Gaussian randomly set to one of the possible votes (usually from 1 to 5 stars). The other half of the training samples are used to teach the model that two different users should return a higher distance than a user and its augmented sample. Fig. 1 shows it with the samples $\bar{s}_{18}$, $\bar{s}_3$, and the label 1.

As can be seen in Fig. 1, half of each sample (left-side vertical vectors) feeds the MLP in charge of translating from large and sparse integer vectors to very small and dense vectors of real numbers (the embeddings denoted with the symbols EN and E'N). Each contrastive sample provides a raw user vector (real or augmented) to the first MLP tower, and another raw user vector (real or augmented) to the second MLP tower (both towers share the same weights). The whole deep learning model learns if both vectors represent the same user (short distance, label 0) or they represent different users (large distance, label 1). The obtained embeddings will learn to similarly code users with the same tastes (similar ratings) and to dissimilarly code users with different tastes. These semantically enriched embeddings make it easy for the model to accurately predict contrastive distances.

Each vector in a sample feed into the MLP that implements the embedding function. Embeddings code sparse and discrete (integer) data to dense and continuous (float) representations. We have used two dense layers (ψ_d and ψ_e) to make this function, where 'd' and 'e' are the number of neurons in the layer. Note that the size of the ensemble will be 'e'. In this stage, we have a dense representation of the source sample (E_N), and a dense representation of the target (augmented or randomly picked) sample E'_N. As shown in Fig. 1, both ensemble vectors are joined using a Lambda layer that implements the Euclidean distance. The Lambda layer returns values in the interval [0..∞], and they must be converted to values around the labels: 0 or 1. To accomplish the task, a dense layer $\psi_{d'}$ makes the main work, followed by a dense final output layer containing a unique neuron ψ_1. Since labels can only take values 0 or 1, the 'sigmoid' activation function has been chosen in this output layer; other layers implement the 'relu' activation function. Finally, the selected loss function to minimize the learning errors is the contrastive one: $loss = (1-y)\frac{1}{2}(pred^2) + y\frac{1}{2}[\max(0, m - pred)]^2$, where 'y' is the label, 'pred' is the predicted distance, and 'm' is a margin used to ensure that dissimilar pairs greater than 'm' do not contribute to the function loss. Following to Fig. 1, a formalization of the proposed Siamese model is provided. The symbols have been kept as shown in Fig. 1 to make it easy to interpret both the equations and the concepts of Fig. 1.

Proposed Siamese model formalization:

- Let U and I be, respectively, the sets of users and items in a CF dataset. (1)
- Let V = {•, 1, 2, 3, 4, 5} be the set of votes (stars) a user can assign to an item in the CF dataset. The symbol • indicates lack of vote. (2)
- Let $\bar{s}_u = (r_1, r_2, r_3, \ldots, r_{|I|})$ be the sample vector (profile) of user u, where $r_i \in V$, represents the vote of user $u \in U$ to item $i \in I$. Please note that CF datasets are extraordinarily sparse, so most of the $r_i = \bullet$. (3)
- $f_\theta(\bar{s}_u) = \tilde{s}_u = (r_1, r_2, r_3, \ldots, r_{|I|})$ is the augmented sample vector of user u, where a random Gaussian function is used to generate random rating values belonging to V-{•}. The f_θ random function generates the $\tilde{s}_u$ vector containing θ randomly chosen modified r_i ratings. Note that the higher the variability value θ, the larger the distance between the original sample $\bar{s}_{u'}$ and the augmented sample $\tilde{s}_u$. (4)
- Let $h(u, \theta) = <\bar{s}_{u'}, f_\theta(\bar{s}_{u'}), 0>$ be the tuple containing the user u sample, its random augmented sample $\tilde{s}_u$, and the label 0. (5)
- $g(\bar{s}_u) = \bar{s}_{u'}$, where u' is randomly selected from U, and $u \neq u'$. (6)
- Let $q(u) = <\bar{s}_u, g(\bar{s}_u), 1>$ be the tuple containing the user u sample, the randomly picked sample $\bar{s}_{u'}$, and the label 1. (7)
- $D_N = \text{shuffle}(\{h(u, \theta), q(u) \; \forall \; u \in U, N \text{ times}\})$, is a dataset containing N x |U| shuffled pairs of samples h(u, θ), q(u) ready to be trained, tested, or validated. (8)
- ψ_e (ψ_d) is a neural network ensemble where the first dense layer contains d neurons, and the second dense layer contains e neurons. (9)
- $E_N = \psi_e(\psi_d(D_N[0]))$ is the activation vector at the exit of the tower 1 embedding, where the input is the first element in the tuple: $D_N[0] = h(u, \theta)$. (10)
- $E'_N = \psi_e(\psi_d(D_N[1]))$ is the activation vector at the exit of the tower 2 embedding, where the input is the second element in the tuple: $D_N[1] = q(u)$. (11)
- $pred = \psi_1\left(\psi_{d'}\left(\sqrt{E_N^2 + E_N'^2}\right)\right)$. The results of both towers are merged using the Euclidean distance, processed in a dense layer containing d' neurons and, finally, compressed to the prediction dense layer ψ_1, that contains a unique neuron. The result is the distance prediction between both inputs: $\bar{s}_{u'}$ and $\tilde{s}_u$, or $\bar{s}_{u'}$ and $\bar{s}_{u'}$. (12)
- $loss = (1-y)\frac{1}{2}(pred^2) + y\frac{1}{2}[\max(0, m - pred)]^2$ is the Siamese neural network contrastive loss in this model, where m is a margin used to ensure that dissimilar pairs greater than m do not contribute to the function loss. We have set this margin to 1. (13)

TABLE I. The Main Parameter Values Involved in the Proposed Model

Parameter	Value
|U|	1000
|I|	1700
V	{•, 1, 2, 3, 4, 5}
θ	{60, 80, 100, 125, 150}
N	50000
e	10
d	20
d'	50

Table I shows the values chosen in our experiments to implement the model. Note that $|U|$, $|I|$ and V are directly dependent on the CF dataset (Movielens 100K in our case).

The rest of parameter values should be adapted when other datasets are used.

Previously to the main experiments, we have selected several embedding sizes: N (see EN and E'N in Fig. 1) and compared their impact on the obtained qualities of the proposed Siamese neural network model. Fig. 2 shows the results; as expected, when the bottleneck threshold is reached (embeddings of size 5, in our case), the smaller the embedding size, the lower the accuracy of the Siamese network. In this model, embedding sizes smaller than 5 produce semantic information loss. Fig. 2 also shows that increasing the size

of the embeddings, above the threshold, does not improve the quality. This result has been obtained by processing the dataset Movielens 1M, and large datasets would need a large embedding size. Research collaborative filtering models usually set the embeddings size to 10.

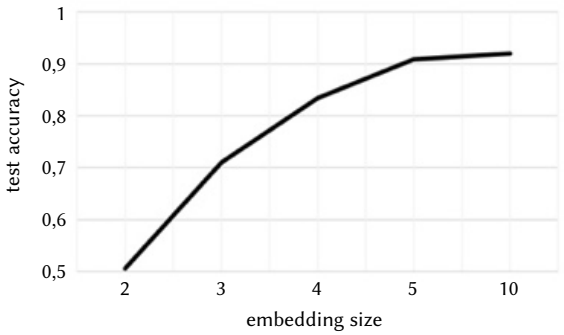

Fig. 2. The proposed Siamese model quality obtained by choosing different embedding sizes. Dataset MovieLens 1M.

We made a grid search to set the number of neurons in the dense layers of the model. The reported result for Movielens 1M was 5 neurons in the first dense layer of MLP. Choosing this hyperparameter, our Siamese model needs 18,716 parameters (using an embedding size = 10). We selected 10 neurons in the first MLP dense layer, making it possible for the designed Siamese model to process large datasets. This model only needs 37,301parameters, which is affordable for any laptop computer. The grid search set to 15 neurons the size of the last dense layer; we have increased it to 40 neurons to hold any other dataset. This decision only increases the model from 30 to 80 parameters. In short, using the grid search results, we obtain a Siamese network holding 18,716 parameters, but we have decided to enlarge the design to a model that contains 37,301 parameters able to process large datasets.

III. Experiments and Results

To test the proposed Siamese model, we have chosen both the MovieLens 100K and the MovieLens 1M datasets. They are open and popular CF datasets. We have also made use of a reduced version of the complete Netflix dataset: Netflix* [24]. Table II shows the main parameter values for these datasets. Additionally, to facilitate reproducibility, the source code is provided via GitHub [25].

TABLE II. Main Parameter Values of the Tested Datasets

Dataset	#users	#items	#ratings	scores	sparsity
MovieLens 100K	943	1682	99,831	1 to 5	93.71
MovieLens 1M	6,040	3,706	911,031	1 to 5	95,94
Netflix*	23,012	1,750	535,421	1 to 5	98.68

Experiments will test a) the learning loss and the model accuracy, b) the impact in accuracy when increasing the variability parameter θ, and c) the comparative results between the proposed model and a representative set of CF similarity measures. The chosen similarity measures are the following: Euclidean, correlation, cosine, Manhattan, Jaccard, MSD, and JMSD that will act as baselines. Note that these measures are the usual ones when the KNN algorithm is used to predict and recommend in CF RS. Finally, experiments compare the accuracy results between the proposed Siamese model and the seven baselines. Accuracy is tested by varying both the a) variability parameter and b) the N-ways testing strategy, where the active user variation (augmentation) must be correctly selected from N samples: a set of N-1 users and the augmented user. Table III shows the values of the selected parameters and the baselines.

TABLE III. Parameter Values and Baselines Used in the Experiments

Parameter	Value
Variability	{60, 80, 100, 125, 150}
N_ways	{4, 8, 16}
Baselines	Euclidean, correlation, cosine, Manhattan, Jaccard, MSD, and JMSD

The Siamese model shown in the previous section was run and provided the training and testing results shown in Fig. 3. The graph on the left of Fig. 3 shows adequate learning trajectories, where contrastive loss progressively decreases until it reaches a very low value. The right graph in Fig. 3 shows a learning with no overfitting and stable from epoch 6. 90% of hits (accuracy) have been reached by classifying between two categories: variations of the active user, and different users. Note that the training and testing data contain both categories equally, so the random baseline gets an accuracy of 50%.

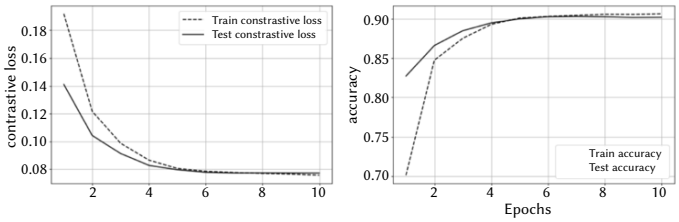

Fig. 3. The proposed training and testing results of the Siamese RS model. The left graph shows the contrastive loss (the lower the values, the better the result), whereas the right graph shows the accuracy obtained in the range [0..1] (the higher the values, the better the result). The x-axis shows the number of epochs (learning evolution).

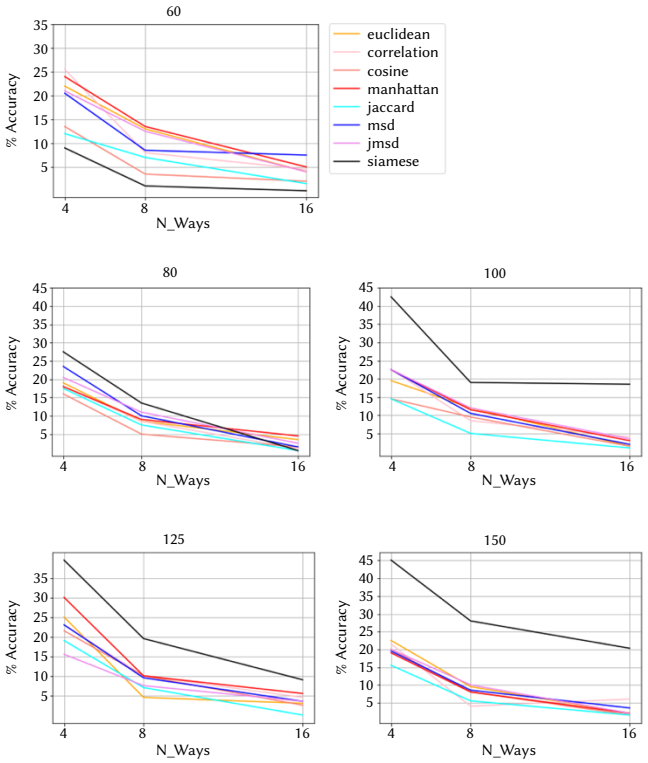

Fig. 4. Experiment results by setting the variability values θ from 60 to 150 (different graphs), and the N-ways size (x-axis). The black lines represent the proposed results of the Siamese model, whereas the colored lines show the baseline ones. The higher the values, the better the result.

From Fig. 4 we can extract some relevant conclusions:

- *The higher the variability parameter θ, the higher the accuracy returned by the proposed Siamese model and the better the comparative between the proposed model and the baselines.* In short, the higher the variability parameter, the better the model learns, and the more difficult will be to distinguish between augmented samples and different samples.
- *The variation of samples (θ) must reach a minimum value to let the Siamese network properly work.* This is the expected result, since the Siamese network design and strategy needs a minimum of variation between each sample and their augmentations to learn differences between augmented samples and different samples.
- *When the variability of the samples is high enough, the proposed method always improves the baseline result.* This is the effect of combining the two previous conclusions.
- *The baselines get similar quality results to each other.* It is known from CF KNN-based published results that the quality reached of different similarity measures highly depends on the data size, sparsity, and internal patterns but, overall, their differences are usually small.

To better compare the proposed model results when the variability values θ change, Fig. 5 extracts this information from Fig. 4. As mentioned above, an accuracy increase happens as the variability increases, and it is true for all the N-ways tested values. As can be seen, each N-way value yields different increases in accuracy; Fig. 6 averages the increase in accuracy along the three considered N-way values. Starting from the variability value $\theta = 60$, Fig. 6 shows the number of times that each variability value has improved accuracy.

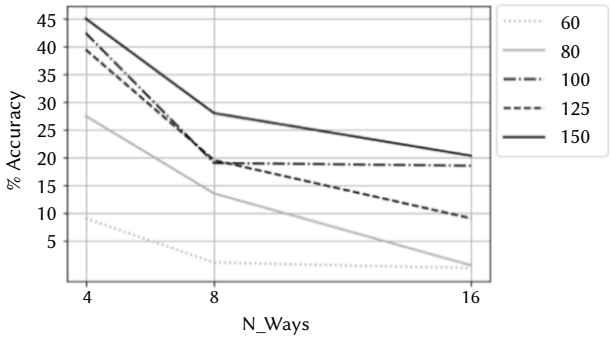

Fig. 5. Proposed Siamese model accuracy results when tested on different variability values θ. The x-axis holds the three considered N-ways sizes. The higher the accuracy, the better the result.

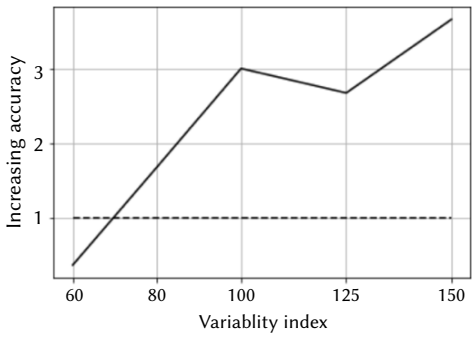

Fig. 6. Increase in the averaged accuracy of the proposed Siamese model. The x-axis shows the considered variability values θ. The y-axis shows the number of times each variability value improves accuracy.

Fig. 7 shows the results obtained using the collaborative filtering data set MovieLens 1M. As can be seen, the results follow a shape and

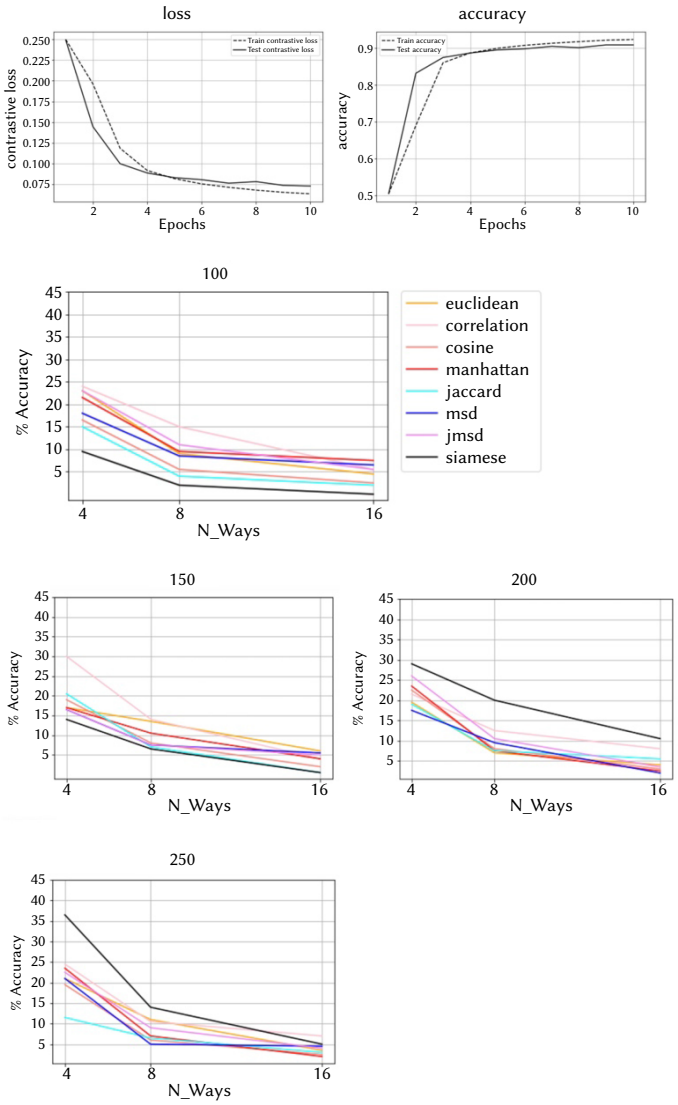

Fig. 7. Results obtained using MovieLens 1M.

pattern similar to the ones obtained by processing MovieLens 100K. It is noteworthy that this large data set also requires larger variability values (θ) than the MovieLens 100K one to generate representative augmented samples. Overall, Fig. 7 shows the scalability of the proposed Siamese model.

Fig. 8 shows the Netflix* results in the same format used in the preceding figures. The graph patterns and the accuracy values reinforce the MovieLens 100K and the MovieLens 1M explanations, emphasizing the scalability of the proposed Siamese network model.

IV. Conclusion

Improving collaborative filtering-based similarity measures is important, since they can be used in big data representations and to provide better recommendation explanations. This paper shows how neural network models can be used to learn similarities and dissimilarities between user profiles in Recommender Systems. We have adapted the Siamese network similarity learning model to the collaborative filtering scenario, feeding it with real user profiles and augmented ones. The proposed Siamese network has adequately learnt the similarity between vectors containing each user rating. Furthermore, the Siamese neural results improve the seven

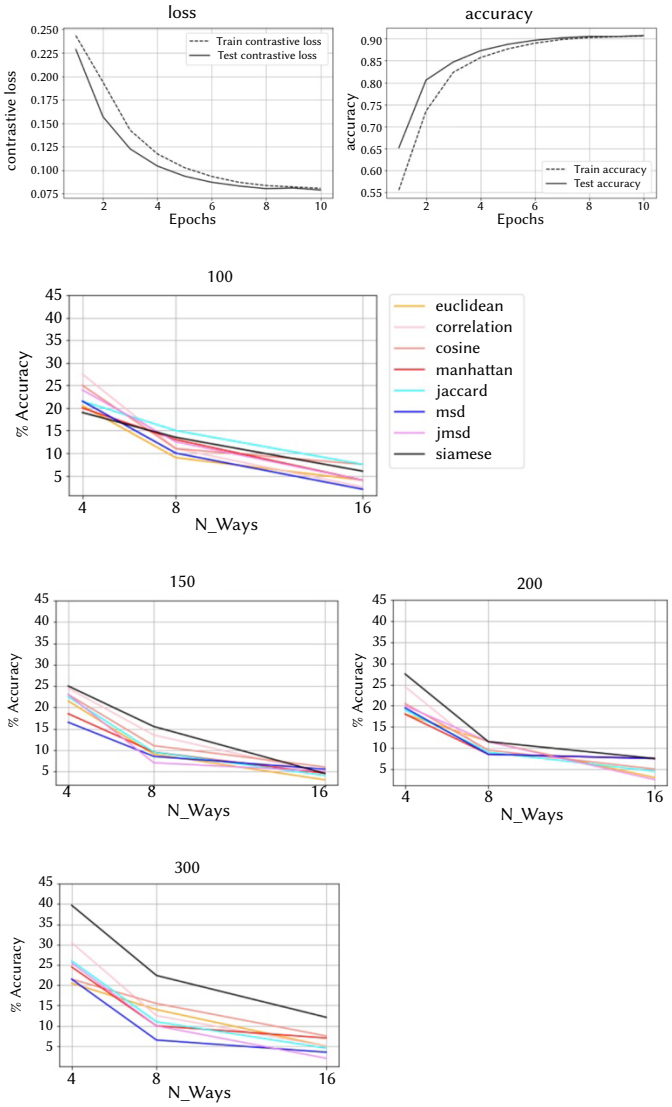

Fig. 8. Results obtained using Netflix*.

representative baselines used in the experiments. The results also show that a meaningful parameter used to set the learning samples has to be fine-tuned to get adequate accuracies. This parameter sets the distance between source samples and the augmented ones, created to train and to test the model. As expected, a balance is needed to generate representative augmented user profiles. Remarkable future works are a) to extend experiments to different collaborative filtering datasets, b) to test the impact of using the triplet loss instead of the contrastive one, and c) to implement different strategies to generate more accurate augmented samples.

Acknowledgment

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.*

References

[1] Z. Shuai, Y. Lina, S., Aixin, T. Yi, "Deep Learning Based Recommender System: A Survey and New Perspectives," *ACM computing surveys*, vol. 52, no. 1, pp. 1-38, 2019, doi: https://doi.org/10.1145/3285029.

[2] J. Bobadilla, A. González-Prieto, F. Ortega, R. Lara Cabrera, "Deep learning feature selection to unhide demographic recommender systems factors," *Neural Computing & Applications*, vol. 33, pp. 7291-7308, 2021, doi: https://doi.org/10.1007/s00521-020-05494-2.

[3] R. Bawack, E. Bonhoure, "Influencer is the New Recommender: insights for Theorising Social Recommender Systems," *Information Systems Frontiers*, vol. 25, pp. 183-197, 2023, doi: https://doi.org/10.1007/s10796-022-10262-9.

[4] S. Raza, C. Ding, "Progress in context-aware recommender systems - An overview," *Computer Science Review*, vol. 31, pp. 84-97, 2019, doi: https://doi.org/10.1016/j.cosrev.2019.01.001.

[5] J. Bobadilla, A. González-Prieto, F. Ortega, R. Lara Cabrera, "Deep learning approach to obtain collaborative filtering neighborhoods," *Neural Computing & Applications*, vol. 34, pp. 2939-2951, 2022, doi: https://doi.org/10.1007/s00521-021-06493-7.

[6] X. Liao, X. Li, Q. Xu, H. Wu, Y. Wan, "Improving Ant Collaborative Filtering on Sparsity via Dimension Reduction," *Applied Sciences*, vol. 10, no. 20, pp. 1-10, 2020, doi: https://doi.org/10.3390/app10207245.

[7] T. Anwar, V. Uma, I. Hussain, M. Pantula, "Collaborative filtering and kNN based recommendation to overcome cold start and sparsity issues: A comparative analysis," *Multimedia Tools and Applications*, vol. 81, pp. 35693-35711, 2022, doi: https://doi.org/10.1007/s11042-021-11883-z.

[8] X. Hong-Jian, D. Xinyu, Z. Jianbing, H. Shujian, C. Jiajun, "Deep Matrix Factorization Models for Recommender Systems," *IJCAI'17: Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pp. 3203-3209, 2017, doi: https://doi.org/10.5555/3172077.3172336.

[9] H. Xiangnan, L. Lizi, Z. Hanwang, "Neural Collaborative Filtering," *WWW '17: Proceedings of the 26th International Conference on World Wide Web*, pp. 173-182, 2017, doi: https://doi.org/10.1145/3038912.3052569.

[10] M. Mendoza, N. Torres, "Evaluating content novelty in recommender systems," *Journal of Intelligent Information Systems*, vol. 54, pp. 297-316, 2020, doi: https://doi.org/10.1007/s10844-019-00548-x.

[11] M. Kunaver, T. Požrl, "Diversity in recommender systems - A survey," *Knowledge-Based Systems*, vol. 123, pp. 154-162, 2017, doi: https://doi.org/10.1016/j.knosys.2017.02.009.

[12] J. Bobadilla, A. Gutiérrez, S. Alonso, A. González-Prieto, "Neural collaborative filtering classification model to obtain prediction reliabilities," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 7, no. 4, pp. 18-26, 2022, doi: https://doi.org/10.9781/ijimai.2021.08.010.

[13] A. Rago, O. Cocarascu, C. Bechlivanidis, D. Lagnado, F. Toni, "Argumentative explanations for interactive recommendations," *Artificial Intelligence*, vol. 296, pp. 103506, 2021, doi: https://doi.org/10.1016/j.artint.2021.103506.

[14] D. Medel, C.S. González-González, S.V. Aciar, "Social Relations and Methods in Recommender Systems: A Systematic Review," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 7, no. 4, pp. 7-17, 2022, doi: https://doi.org/10.9781/ijimai.2021.12.004.

[15] A. Hernando, J. Bobadilla, F. Ortega, A. Gutiérrez, "Method to interactively visualize and navigate related information," *Expert Systems with Applications*, vol. 111, pp. 61-75, 2018, doi: https://doi.org/10.1016/j.eswa.2018.01.034.

[16] D. Kumar, S. Ray, "Approaches and algorithms to mitigate cold start problems in recommender systems: a systematic literature review," *Journal of Intelligent Information Systems*, vol. 59, pp. 341-366, 2022, doi: https://doi.org/10.1007/s10844-022-00698-5.

[17] H.J. Ahn, "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem," *Information Sciences*. vol. 178, no. 1, pp. 37-51, 2008, doi: https://doi.org/10.1016/j.ins.2007.07.024.

[18] N. Serrano, A. Bellogín, "Siamese neural networks in recommendation," *Neural Computing & Applications*, vol. 35, pp. 13941-13953, 2023, doi: https://doi.org/10.1007/s00521-023-08610-0.

[19] A. Faroughi, P. Moradi, "MOOCs Recommender System with Siamese Neural Network," *2022 9th International and the 15th National Conference on E-Learning and E-Teaching (ICeLeT)*, pp. 1-6, 2022, doi: https://doi.org/10.1109/ICeLeT55619.2022.9765439.

[20] M. Jian, C. Zhang, M. Liu, X. Fu, S. Li, G. Shi, L. Wu, "Siamese Graph-Based Dynamic Matching for Collaborative Filtering," *Information Sciences*, vol. 611, pp. 185-198, 2022, doi: https://doi.org/10.1016/j.ins.2022.08.062.

[21] Q. Wang, R. Zhang, K. Ma, B. Chen, J. Chen, X. Shi, "Siamese Generative Adversarial Predicting Network for Extremely Sparse Data in in Recommendation System," *2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*, pp. 1236-1241, doi: https://doi.org/10.1109/ISPA-BDCloud-SocialCom-SustainCom52081.2021.00170.

[22] G. Gao, L. Liu, L. Wang, Y. Zhang, "Fashion clothes matching scheme based on siamese network and autoencoder," *Multimedia Systems*, vol. 25, pp. 593-602, 2019, doi: https://doi.org/10.1007/s00530-019-00617-9.

[23] M. Pulis, J. Bajada, "Siamese neural networks for content-based cold-start music recommendation," RecSys '21: Proceedings of the 15th ACM Conference on Recommender Systems, pp. 719-723, 2021, doi: https://doi.org/10.1145/3460231.3478847.

[24] F. Ortega, B. Zhu, J. Bobadilla, A. Hernando, "Cf4j: Collaborative filtering for java", *Knowledge-Based Systems*, vol. 152, pp. 94–99, 2018, doi: https://doi.org/10.1016/j.knosys.2018.04.008.

[25] J. Bobadilla, A. Gutiérrez, SiameseRS, GitHub repository, 2025. Accessed: Mar. 6, 2025 [Online]. Available: https://github.com/jesusbobadilla/SiameseRS.

Jesús Bobadilla

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

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 search 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.