

A Comparative Analysis of State-of-the-Art Recommendation Techniques in the Movie Domain

Dalia Valeriani, Giuseppe Sansonetti(✉), and Alessandro Micarelli

Department of Engineering, Roma Tre University, Via della Vasca Navale, 79, 00146 Rome, Italy
dal.valeriani@stud.uniroma3.it, {gsansone,micarel}@dia.uniroma3.it

Abstract. Recommender systems (RSs) represent one of the manifold applications in which Machine Learning can unfold its potential. Nowadays, most of the major online sites selling products and services provide users with RSs that can assist them in their online experience. In recent years, therefore, we have witnessed an impressive series of proposals for novel recommendation techniques that claim to ensure significative improvements compared to classic techniques. In this work, we analyze some of them from a theoretical and experimental point of view and verify whether they can deliver tangible real improvements in terms of performance. Among others, we have experimented with traditional model-based and memory-based collaborative filtering, up to the most recent recommendation techniques based on deep learning. We have chosen the movie domain as an application scenario, and a version of the classic MovieLens as a dataset for training and testing our models.

Keywords: Machine learning · Recommender systems · Collaborative filtering · Matrix factorization · Deep learning

1 Introduction and Background

Every day, Artificial Intelligence (AI) redesigns our lives [33] and spaces around us [8]. Without even realizing it, we interact - anytime and anywhere - with systems based on AI models and techniques [24]. One of the most popular examples is represented by recommender systems (RSs) [1]. In the research literature, a recommender system is defined as a decision strategy for users in complex information environments [18]. Examples of applications of RSs are in several domains: e-commerce [4], news articles [6], research papers [15], music [25], and movies [3]. RSs assist users in the fruition of cultural heritage resources [30] and points of general interest [28] as well. They also suggest personalized itineraries [12] based on the user's interests and the context of use [2]. With the spread of RSs over the years, we have witnessed more and more proposals of recommendation techniques that show off better results compared to classic approaches. The goal

© Springer Nature Switzerland AG 2020
O. Gervasi et al. (Eds.): ICCSA 2020, LNCS 12252, pp. 104–118, 2020.
https://doi.org/10.1007/978-3-030-58811-3_8

of our research activities is to verify the extent of such performance improvements. To this aim, we have implemented and tested some recommendation models starting from more classic techniques up to more innovative and timely approaches.

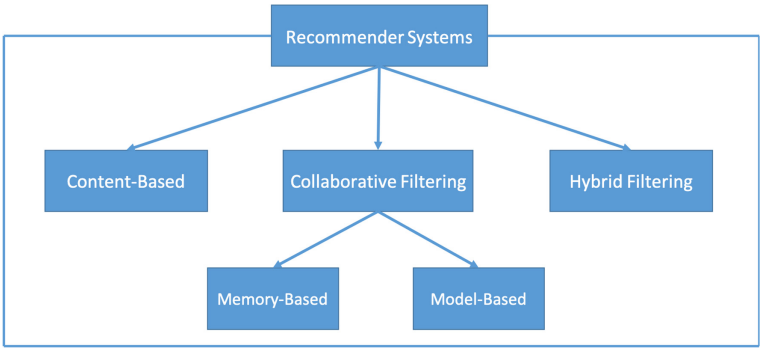


Fig. 1. A classification of recommendation techniques.

Several classifications of recommendation techniques have been proposed in the literature (see, e.g., [5,7,26,27]). According to the most general possible (see Fig. 1), RSs can be classified into three major categories based on how the recommendation is generated: content-based, collaborative filtering, and hybrid approaches. In the research work described in this paper, we mainly explored the collaborative filtering (CF) technique, with reference to two different types of approaches: memory-based (which takes advantage of all available data) and model-based (which exploits abstract representation of data) [19]. The CF technique exploits collaborative information starting from the rating matrix (i.e., user-item interactions). It can perform two types of tasks:

- prediction;
- recommendation.

In this work, we focus on the prediction task in which we want to estimate a utility function f to predict each user's preference for a new item. Typically, the set of ratings (i.e., the user-item matrix R) is divided into a training set (R_{train}) used to learn the function f, and a test set (R_{test}) used to evaluate the prediction accuracy. To assess the performance of RSs, several evaluation metrics have been proposed [32]. In this work, we consider the *root mean square error (RMSE)* to express the performance of the different models. RMSE can be defined as follows:

$$RMSE(f) = \sqrt{\frac{1}{|R_{test}|} \sum_{r_{ui} \in R_{test}} (f(u,i) - r_{ui})^2} \tag{1}$$

where $f(u, i)$ and r_{ui} are, respectively, the predicted recommendation scores and the actual rating values for all evaluated users $u \in U$ and all items $i \in I$ in the test set R_{test}.

Table 1. Statistics of the MovieLens 100K dataset.

	Users	Movies	Ratings	Sparsity
MovieLens 100K	943	1682	100,000	93.7%

The dataset chosen for experimentation purposes is *MovieLens 100K*[1], a movie dataset built ad hoc and widely used in empirical research. Table 1 shows its statistics. More precisely, it consists of 943 users, 1682 movies, and 100,000 explicit ratings with a value in the range $[1, 5]$.

The rest of this paper is structured as follows. In Sect. 2, we explore the characteristics of the dataset chosen to perform our experimental tests. We, also, describe the preprocessing steps we made on it to apply the different recommendations techniques tested. Section 3 illustrates the experimental sessions carried out and the relative results obtained. In Sect. 4, we draw our conclusions and outline some possible future developments of the work presented herein.

2 Dataset

2.1 Analysis

Before testing the different RSs, we performed an initial exploration of the dataset, in order to better understand the available data and the most interesting features for its use. A first analysis of the dataset concerned the *movies* belonging to it. Figure 2a highlights the terms that most appear in movie titles, using a larger size for those with a higher frequency. We also extracted the genres of the most popular movies in the dataset, obtaining the following list in descending order:

- Drama
- Comedy
- Thriller
- Action
- Romance
- ...

Such genres are shown in Fig. 2b, where the more popular genres are displayed with a larger size. Figure 3 shows the number of movies belonging to each genre. It can be noticed that most movies belong to the Drama category, the most popular genre, followed by the categories highlighted in the previous list.

[1] grouplens.org/datasets/movielens/100k/ (Accessed: June 23, 2020).



(a) Movie titles. (b) Movie genres.

Fig. 2. Tag clouds related to the movies in the MovieLens 100K dataset (larger size means higher frequency).

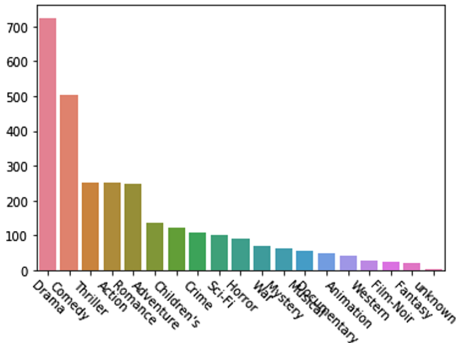


Fig. 3. Number of movies for each genre.

The subsequent analysis concerned the *users* belonging to the dataset. Specifically, we analyzed their age distribution, which led to identify the highest number of them (i.e., 35.2%) aged between 20 and 29 years, followed by a lower percentage (i.e., 25.6%) of adult users, up to the lowest percentage (i.e., 0.1%) under ten years (see Fig. 4).

After the first phase of exploration, more attention was paid to the study of the rating distribution of the whole dataset. In Fig. 5a, users appear to have been generous in their rating behavior: the average rating was 3.52 in the range $[1, 5]$, with half of the movies that received a rating value between 4 and 5. It can be also noted that the rating distribution illustrated in Fig. 5a is more on the positive side, which suggests that users are more likely to watch movies they like. The total number of users is shown in Fig. 5b, where users are represented on the y-axis and the possible rating values on the x-axis. It can be noted that almost $35,000$ users gave a rating of 4, more than $25,000$ users gave 3, and so on up to a rating of 1 expressed by over $5,000$ users.

We also wanted to investigate the rating distribution over all movies and users. In Fig. 6a, the number of ratings is shown on the y-axis, while the movie IDs are reported on the x-axis. It can be noticed that few films have been evaluated more frequently, representing the most popular items according to the classic *long tail* style curve. The same behavior occurs in Fig. 6b, where the user IDs are displayed on the x-axis. A small proportion of users gave most of the ratings.

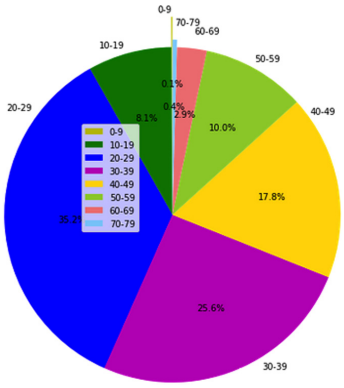
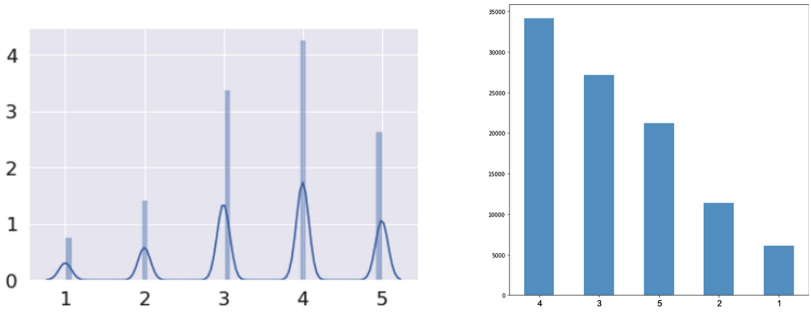
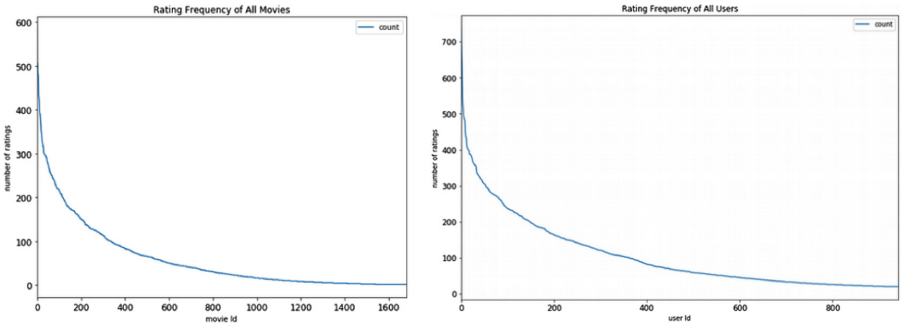


Fig. 4. Age distribution of the users in the dataset.



(a) Rating distribution. (b) Overall user ratings.

Fig. 5. Ratings reported in the dataset.



(a) Frequency on all movies. (b) Frequency on all users.

Fig. 6. Statistics on ratings.

The last analysis of the dataset involved a key aspect to fully understand the recommendation scenario: the dataset *sparsity*. Sparsity is a common challenge to overcome in many CF applications [17]. This term denotes the lack of available ratings in the R user-item matrix. If we consider a R matrix with dimensions $(n_{users} \times m_{movies})$, where each element r_{ij} represents the rating value assigned by the user i to the movie j, this matrix will likely have a very small number of entries, since most users usually rate only a few of the available movies. In the MovieLens 100K dataset, the sparsity of the R matrix can be calculated as follows:

$$\frac{n_{ratings}}{n_{users} \cdot m_{movies}} = 0.063 \tag{2}$$

which means that only 6.3% of matrix entries have a value and 93.7% of the remaining data is missing, thus making it a very sparse dataset.

2.2 Preprocessing

For experimental purposes, we carried out a preprocessing of the dataset to build the recommendation models in offline mode. Specifically, for the memory-based technique, we explored the user-based and item-based approaches with k-nearest-neighbor algorithms. These algorithms take advantage of the overall rating matrix to generate a prediction based on a set of neighboring users/items, called "neighbors" or "peer users/items", who share similar ratings with the target user. Differently, for realizing model-based RSs, we made use of:

- a simple but effective technique, named *Slope One* [23], which relies on the average rating difference between users;
- techniques based on matrix factorization [21], which map users and items in a space with reduced dimensionality by building a latent factor model necessary for predicting the rating;
- methods based on Deep Learning [22], which through deep neural networks can be used to add nonlinear transformations to existing RS approaches and interpret them in neural extensions to generate predictions.

3 Experimental Evaluations

3.1 First Experimental Session

In the first experimental session, we created models based on user-based (UB) and item-based (IB) approaches by applying different techniques known in the literature, such as filtering neighbors and normalizing the rating. The dataset was divided into 80% training set and 20% test set. In the training dataset, we determined the similarity weights for generating the prediction. Similarity weights play a double role in the neighborhood-based recommendation methods:

- they allow for the selection of neighbors;
- they provide the means to give more or less relevance to those neighbors in the prediction process.

Calculating similarity weights is one of the most critical aspects of building a recommender system as it can have a significant impact on accuracy. The similarity metrics considered in this study are the *cosine similarity* and the *Pearson's correlation coefficient*.

Cosine Similarity. This metric consists of a vector approach: users and items are represented as dimensional vectors, and similarity is measured by means of the cosine distance between these two rating vectors, respectively. The function then measures the cosine of the angle to quantify their similarity. The calculation can be performed efficiently by taking their scalar product and dividing it by the product of their L2 (Euclidean) norms. The maximum value it assumes is 1, which indicates the maximum similarity between the two vectors of users or items. For the user-based approach, the equation is as follows:

$$s(u,v) = \frac{\sum_{i \in I_{uv}} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i \in I_u} r_{ui}^2 \sum_{i \in I_v} r_{vi}^2}} \tag{3}$$

where $s(u,v)$ represents the similarity between the user u and the user v on all items evaluated. In particular, I_{uv} denotes the set of items $i \in I$ *corated*, that is, evaluated by both the user u and the user v. For the item-based approach, the equation is as follows:

$$s(i,j) = \frac{\sum_{u \in U_{ij}} r_{iu} \cdot r_{ju}}{\sqrt{\sum_{u \in U_i} r_{iu}^2 \sum_{u \in U_j} r_{ju}^2}} \tag{4}$$

where $s(i,j)$ represents the similarity between the item i and the item j on all users who rated them. In particular, U_{ij} represents the set of users $u \in U$ who evaluated both the item i and the item j.

Pearson's Correlation Coefficient. This method, unlike the cosine similarity, relies on a statistical approach that computes the correlation between the common ratings given by two users to determine their similarity. When using the Pearson's correlation, the similarity is expressed in a range $[-1, +1]$, where a high positive value suggests a high correlation, a high negative value suggests an inversely high correlation, and a zero value suggests no correlation. For the user-based approach, the equation is as follows:

$$s(u,v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)^2 \sum_{i \in I_{uv}} (r_{vi} - \bar{r}_v)^2}} \tag{5}$$

where $s(u,v)$ represents the similarity between users u and v, r_{ui} and r_{vi} represent the respective user's ratings for the item i, and $\bar{r}_u$ and $\bar{r}_v$ represent the average rating of the user u or v on all items evaluated. For the item-based

approach, the equation is as follows:

$$s(i,j) = \frac{\sum_{u \in U_{ij}} (r_{ui} - \bar{r}_i)(r_{ui} - \bar{r}_j)}{\sqrt{\sum_{u \in U_{ij}} (r_{ui} - \bar{r}_i)^2 \sum_{i \in U_{ij}} (r_{ui} - \bar{r}_j)^2}} \tag{6}$$

where, likewise, $s(i,j)$ represents the similarity between items i and j, and $\bar{r}_i$ and $\bar{r}_j$ represent the average rating on the respective items, rated by users who evaluated both of them.

Table 2. Results of the first experimental session.

	Cosine similarity	Pearson's correlation coefficient	Cosine similarity neighbor fine tuning	Mean centering neighbor fine tuning
RMSE user-based	2.88 (N = 943 users)	2.91 (N = 943 users)	2.41 (N = 40 users)	2.63 (N = 50 users)
RMSE item-based	3.37 (N = 1682 items)	3.37 (N = 1682 items)	2.71 (N = 40 items)	2.85 (N = 15 items)

Experimental Results. The rating prediction was then performed on the training dataset and evaluated on the test dataset with and without filtering peer users or items. Table 2 shows the results in terms of RMSE when the configurations for both techniques vary. The most significant results were obtained with a pre-filtering of 40 most correlated neighbors, which led to an RMSE value of approximately 2.41 for the UB approach and 2.71 for the IB approach. Higher values were obtained through the rating normalization using the mean centering (MC) technique. The idea behind MC is to determine whether a rating is positive or negative by comparing it with the average rating. Therefore, this technique remaps a user's ratings by subtracting the average value of all its ratings, indicating whether the particular rating is positive or negative compared to the average. Values above the average rating represent positive ratings, values below the average ratings represent negative ratings. Actually, the rating normalization through MC does not improve the accuracy of the model despite a fine-tuning of the neighbors involving 50 users and 15 items in the prediction task, respectively. Furthermore, the use of both similarity metrics proved to be interchangeable for the purposes of the predictive capacity by the model itself.

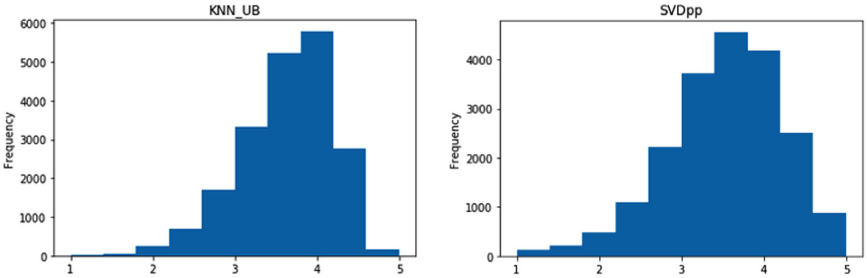
3.2 Second Experimental Session

In the second experimental session, we experimented with different prediction algorithms based on collaborative filtering and matrix factorization [11], using a k-fold cross-validation technique with $k = 5$. The most significant results we obtained are shown in Table 3. For the user-based and item-based approaches, a sensitivity analysis was performed on the number of neighbors, showing a

better value of RMSE for $N = 50$ and $N = 10$ neighbors, respectively. The Slope One technique, instead, registered a RMSE value of around 0.94. The more complex Singular Value Decomposition (SVD) and SVD++ algorithms obtained satisfactory RMSE values equal to 0.9343 and 0.9166 with $L = 50$ and $L = 10$ latent factors, respectively. The central point of these algorithms is to best approximate the initial matrix by learning the linear user-item interactions through stochastic gradient descent. It can be noted in Table 3 that SVD++ obtained better performance than SVD. The main difference compared to the simple SVD is that SVD++ can exploit implicit ratings for the prediction task, taking into account only which items users have evaluated, regardless of their rating values.

Table 3. Results of the second experimental session.

	KNN-UB	KNN-IB	Slope One	SVD	SVD++
RMSE	1.0158 ($N = 50$)	1.0256 ($N = 10$)	0.94	0.9343 ($L = 50$)	0.9166 ($L = 10$)



(a) KNN User-Based approach. (b) SVD++ algorithm.

Fig. 7. Distribution of predicted ratings through two different recommendation techniques.

A key aspect to understand the predictive capacity and behavior of a recommendation model is to analyze its rating distribution. Figure 7 shows the distribution of ratings predicted by the tested algorithms by comparing a more classic technique such as the user-based approach with a more complex technique such as SVD++. The rating values are represented on the x-axis, the number of predictions for each considered rating value is represented on the y-axis. The user-based algorithm concentrates its predictions around the average, showing a high number of predictions for rating values between 3 and 4, and a low number of predictions for the extreme values of the considered rating range. The behavior is different for the SVD++ algorithm, where the rating frequency is well distributed along the extreme values of the ratings considered. Such a distribution reflects the most accurate performance of the SVD++ model.

3.3 Third Experimental Session

In the third experimental session, we experimented with three different predictive models based on feed-forward neural architectures. Figure 8 shows the architecture of the *neural collaborative filtering (NCF)* framework [34] we tested. This architecture can generate the predicted scores with the output layer through a preprocessing of the features in dense embedding vectors and the use of neural CF layers. The two first architectures created had the following structure. For the input layers, we chose the user ID and the movie ID as unique features. In the first model (NCF_1), a CF layer and an output layer were used choosing $N = 20$ latent user/item factors. In the second model (NCF_2), the same architecture is employed with an additional CF layer.

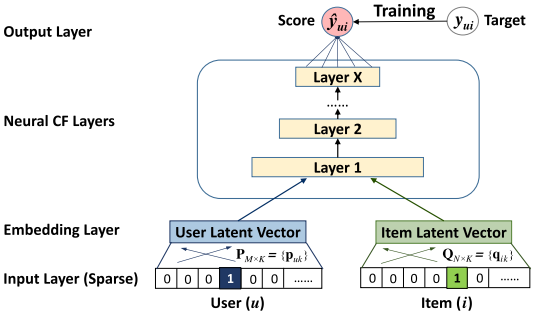


Fig. 8. Neural collaborative filtering (NCF) architecture [34].

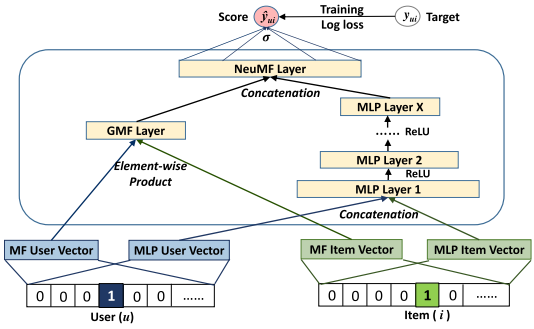


Fig. 9. Neural matrix factorization (NMF) architecture [16].

Finally, a further model [16] was implemented based on a more complex architecture, called *neural matrix factorization (NMF)*. Its architecture is depicted in Fig. 9. The main difference of this model compared to the previous two frameworks is that it combines two different neural networks:

– a standard matrix factorization network (GMF) that performs a linear modeling of the user-item features by running a simple scalar product among the latent factors of users and movies;
– a multilayer feed-forward network (MLP) that models non-linear user-item interactions to compute the final prediction score.

The key idea behind this model is to concatenate the prediction outputs from the single networks, thus creating an entirely unified network that can represent a more robust predictive structure.

Experimental Results. The three proposed architectures shared the following properties:

– The 60% of the dataset used as a training set;
– The 20% of the dataset used as a validation set;
– The 20% of the dataset used as a test set;
– Optimization function *Adam* [20] with a learning rate $\lambda = 0.0002$;
– 1875 instances per batch;
– 20 epochs with early stopping function set to 4 epochs in the training phase;
– Use of dropout layers to prevent overfitting;
– Evaluation metric for model accuracy: RMSE;
– Network loss function: mean square error.

Table 4 shows the performance of all the tested neural approaches, from the less complex neural architecture to the more structured one. It can be seen that the best result, in terms of RMSE, was achieved by the NMF model.

Table 4. Results of the third experimental session.

	NCF_1	NCF_2	NMF
RMSE	0.9327	0.9285	0.8680

3.4 Overall Experimental Results

In this section, we report the most significant results obtained in the second and third experimental session. Such results were optimized by identifying the best parameters of the different recommenders. Specifically, the algorithm specifications were as follows:

1. **KNN_IB** with $N = 10$ neighbors;
2. **KNN_UB** with $N = 50$ neighbors;
3. **SLOPE_ONE**;
4. **SVD** with $L = 50$ latent factors;
5. **SVD++** with $L = 10$ latent factors;

6. **NCF_1** with $L = 20$ latent factors (MLP);
7. **NCF_2** with $L = 20$ latent factors (MLP);
8. **NMF** with $L = 20$ latent factors (MLP) and $L = 10$ latent factors (GMF).

For the choice of the graph, we opted for a logarithmic representation capable of capturing in a more significant way the sensitive variations and the significant differences between the different RMSE values. Figure 10 summarizes all the obtained findings. The empirical results show that memory-based approaches

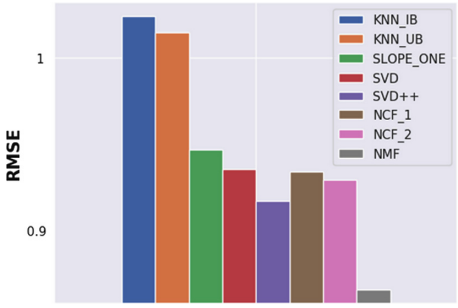


Fig. 10. Results of the second and third experimental sessions in terms of root mean square error (RMSE).

ranked in the last place in terms of predictive capacity, although they have the advantage of being able to take full advantage of user and movie data. Differently, more accurate models starting from Slope One up to SVD++ obtained more satisfactory results by addressing in the best way the typical data sparsity problem, present in the used dataset as well as in many real applications. Finally, the deep learning models proved to be accurate and robust in terms of predictive capacity, with particular reference to the NMF model, which achieved the lowest RMSE score among all the tested algorithms, thanks to the flexibility of its architecture. In order to verify the statistical significance of our experimental tests, we performed one-sample paired t-tests on the results, finding that all differences in accuracy were statistically significant for $p < 0.05$.

4 Conclusions and Future Works

In the research activities presented herein, different predictive models have been compared, thus highlighting the natural convergence to more innovative approaches that are increasingly considered in the application of RSs. Thanks to their flexibility, deep neural networks have been shown to better model user-item interactions to capture increasingly complex patterns for the benefit of the model accuracy.

In the future, it would be interesting to evaluate the behavior of architectures such as Recurrent Neural Networks (RNN) [31], and the use of unsupervised

approaches such as autoencoders [14] for modeling the recommendation task. We also plan to employ different datasets (e.g., linked open data [9]), experiment with further recommendation techniques, and integrate the simple ratings provided by users with data of different nature such as attitudes [10], temporal dynamics [29], and web browsing activities [13].

References

1. Aggarwal, C.C.: Recommender Systems: The Textbook, 1st edn. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-29659-3
2. Biancalana, C., Flamini, A., Gasparetti, F., Micarelli, A., Millevolte, S., Sansonetti, G.: Enhancing traditional local search recommendations with context-awareness. In: Konstan, J.A., Conejo, R., Marzo, J.L., Oliver, N. (eds.) UMAP 2011. LNCS, vol. 6787, pp. 335–340. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22362-4_29
3. Biancalana, C., Gasparetti, F., Micarelli, A., Miola, A., Sansonetti, G.: Context-aware movie recommendation based on signal processing and machine learning. In: Proceedings of the 2nd Challenge on Context-Aware Movie Recommendation. CAMRa 2011, pp. 5–10. ACM, New York (2011)
4. Bologna, C., De Rosa, A.C., De Vivo, A., Gaeta, M., Sansonetti, G., Viserta, V.: Personality-based recommendation in e-commerce. In: CEUR Workshop Proceedings, Aachen, Germany, vol. 997. CEUR-WS.org (2013)
5. Burke, R.: Hybrid web recommender systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) The Adaptive Web. LNCS, vol. 4321, pp. 377–408. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72079-9_12
6. Caldarelli, S., Feltoni Gurini, D., Micarelli, A., Sansonetti, G.: A signal-based approach to news recommendation. In: CEUR Workshop Proceedings, Aachen, Germany, vol. 1618. CEUR-WS.org (2016)
7. Colombo-Mendoza, L.O., Paredes-Valverde, M.A., Salas-Zárate, M.P., Bustos-López, M., Sánchez-Cervantes, J.L., Alor-Hernández, G.: Recommender systems in the offline retailing domain: a systematic literature review. In: García-Alcaraz, J.L., Sánchez-Ramírez, C., Avelar-Sosa, L., Alor-Hernández, G. (eds.) Techniques, Tools and Methodologies Applied to Global Supply Chain Ecosystems. ISRL, vol. 166, pp. 383–409. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-26488-8_17
8. D'Aniello, G., Gaeta, M., Orciuoli, F., Sansonetti, G., Sorgente, F.: Knowledge-based smart city service system. Electronics (Switzerland) **9**(6), 1–22 (2020)
9. De Angelis, A., Gasparetti, F., Micarelli, A., Sansonetti, G.: A social cultural recommender based on linked open data. In: Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization. UMAP 2017, pp. 329–332, ACM, New York (2017)
10. Gurini, D.F., Gasparetti, F., Micarelli, A., Sansonetti, G.: iSCUR: interest and sentiment-based community detection for user recommendation on Twitter. In: Dimitrova, V., Kuflik, T., Chin, D., Ricci, F., Dolog, P., Houben, G.-J. (eds.) UMAP 2014. LNCS, vol. 8538, pp. 314–319. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08786-3_27
11. Feltoni Gurini, D., Gasparetti, F., Micarelli, A., Sansonetti, G.: Temporal people-to-people recommendation on social networks with sentiment-based matrix factorization. Future Gener. Comput. Syst. **78**, 430–439 (2018)

12. Fogli, A., Sansonetti, G.: Exploiting semantics for context-aware itinerary recommendation. Pers. Ubiquit. Comput. **23**(2), 215–231 (2019). https://doi.org/10.1007/s00779-018-01189-7
13. Gasparetti, F., Micarelli, A., Sansonetti, G.: Exploiting web browsing activities for user needs identification. In: 2014 International Conference on Computational Science and Computational Intelligence, vol. 2, pp. 86–89, March 2014
14. Hassan, H.A.M., Sansonetti, G., Gasparetti, F., Micarelli, A.: Semantic-based tag recommendation in scientific bookmarking systems. In: Proceedings of the 12th ACM Conference on Recommender Systems. RecSys 2018, pp. 465–469. ACM, New York (2018)
15. Hassan, H.A.M., Sansonetti, G., Gasparetti, F., Micarelli, A., Beel, J.: BERT, ELMo, USE and InferSent sentence encoders: the panacea for research-paper recommendation? In: Tkalcic, M., Pera, S. (eds.) Proceedings of ACM RecSys 2019 Late-Breaking Results, vol. 2431, pp. 6–10. CEUR-WS.org (2019)
16. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.: Neural collaborative filtering. CoRR abs/1708.05031 (2017)
17. Idrissi, N., Zellou, A.: A systematic literature review of sparsity issues in recommender systems. Soc. Netw. Anal. Min. **10**(1), 1–23 (2020). https://doi.org/10.1007/s13278-020-0626-2
18. Isinkaye, F., Folajimi, Y., Ojokoh, B.: Recommendation systems: principles, methods and evaluation. Egypt. Inform. J. **16**(3), 261–273 (2015)
19. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender Systems: An Introduction, 1st edn. Cambridge University Press, New York (2010)
20. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Proceedings of the 3rd International Conference on Learning Representations (ICLR) (2015)
21. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
22. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
23. Lemire, D., Maclachlan, A.: Slope one predictors for online rating-based collaborative filtering. In: Proceedings of SIAM Data Mining (SDM 2005) (2005)
24. Micarelli, A., Neri, A., Sansonetti, G.: A case-based approach to image recognition. In: Blanzieri, E., Portinale, L. (eds.) EWCBR 2000. LNCS, vol. 1898, pp. 443–454. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44527-7_38
25. Onori, M., Micarelli, A., Sansonetti, G.: A comparative analysis of personality-based music recommender systems. In: CEUR Workshop Proceedings, Aachen, Germany, vol. 1680, pp. 55–59. CEUR-WS.org (2016)
26. Ricci, F.: Recommender systems in tourism. In: Handbook of e-Tourism, pp. 1–18 (2020)
27. Saha, J., Chowdhury, C., Biswas, S.: Review of machine learning and deep learning based recommender systems for health informatics. In: Dash, S., Acharya, B.R., Mittal, M., Abraham, A., Kelemen, A. (eds.) Deep Learning Techniques for Biomedical and Health Informatics. SBD, vol. 68, pp. 101–126. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-33966-1_6
28. Sansonetti, G.: Point of interest recommendation based on social and linked open data. Pers. Ubiquit. Comput. **23**(2), 199–214 (2019). https://doi.org/10.1007/s00779-019-01218-z
29. Sansonetti, G., Feltoni Gurini, D., Gasparetti, F., Micarelli, A.: Dynamic social recommendation. In: Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017. ASONAM 2017, pp. 943–947. ACM, New York (2017)

30. Sansonetti, G., Gasparetti, F., Micarelli, A., Cena, F., Gena, C.: Enhancing cultural recommendations through social and linked open data. User Model. User Adap. Inter. **29**(1), 121–159 (2019). https://doi.org/10.1007/s11257-019-09225-8
31. Sherstinsky, A.: Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Physica D Nonlinear Phenom. **404**, 132306 (2020)
32. Silveira, T., Zhang, M., Lin, X., Liu, Y., Ma, S.: How good your recommender system is? a survey on evaluations in recommendation. Int. J. Mach. Learn. Cybern. **10**(5), 813–831 (2019). https://doi.org/10.1007/s13042-017-0762-9
33. Tegmark, M.: Life 3.0: Being Human in the Age of Artificial Intelligence. Knopf Publishing Group, New York City (2017)
34. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: a survey and new perspectives. ACM Comput. Surv. **52**(1), 1–38 (2019)