Providing reliability in recommender systems through Bernoulli Matrix Factorization

Fernando Ortega [a,*], Raúl Lara-Cabrera [a], Ángel González-Prieto [a], Jesús Bobadilla [a]

[a] Dpto. Sistemas Informáticos, ETSI Sistemas Informáticos, Universidad Politécnica de Madrid, Madrid, Spain

ARTICLE INFO

ABSTRACT

Beyond accuracy, quality measures are gaining importance in modern recommender systems, with reliability being one of the most important indicators in the context of collaborative filtering. This paper proposes Bernoulli Matrix Factorization (BeMF), which is a matrix factorization model, to provide both prediction values and reliability values. BeMF is a very innovative approach from several perspectives: a) it acts on model-based collaborative filtering rather than on memory-based filtering, b) it does not use external methods or extended architectures, such as existing solutions, to provide reliability, c) it is based on a classification-based model instead of traditional regression-based models, and d) matrix factorization formalism is supported by the Bernoulli distribution to exploit the binary nature of the designed classification model. The experimental results show that the more reliable a prediction is, the less liable it is to be wrong: recommendation quality improves after the most reliable predictions are selected. State-of-the-art quality measures for reliability have been tested, which shows that BeMF outperforms previous baseline methods and models.

© 2020 Elsevier Inc. All rights reserved.

1. Introduction

Recommender System (RSs) [42] are services that most people use. Relevant commercial examples are Netflix, Spotify, Amazon and TripAdvisor. Improving recommendation accuracy has focused existing research [7], and prediction models, such as the K Nearest Neighbours (KNN) algorithm [7] to the Matrix Factorization (MF) methods [24] and the latest Neural Network (NN) models [4,5] have been refined over time. In recent years, increasing importance has been given to "beyond accuracy" RS measures [19]. Diversity, coverage and serendipity are current goals in RSs, where diverse and innovative recommendations are highly desirable. It is also important that recommendations have a novelty degree [12]: to recommend *Star Wars VI (Return of the Jedi)* to a Star Wars fan can be very accurate but it is not surprising or "novel" to him or her. Remarkable current research in RS points to "reliable" recommendations [6]: when an RS recommends a restaurant by giving it five stars, a person is probably not entirely convinced about the rating. Some RS services provide additional information that allows us to infer a "reliability" for the rating; this information usually consists of the number of people who have voted for the restaurant. Almost everyone prefers a restaurant with an average vote of 4 stars and 1200 ratings than a restaurant with an average vote of 5 stars and only 5 ratings. RS methods and models for reliability automatically provide a reliability value that is associated with each recommendation.

Providing accurate reliability values that are associated with predictions is an important goal in the RS field for the following reasons:

---

a) This approach resembles human semantics: "I recommend this film because it is truly good, but I am not sure if you like war movies" (high prediction, low reliability); "I know you very well, and I am sure you will love these shoes" (high prediction, high reliability); "I definitely do not recommend this restaurant" (low prediction, high reliability); "I do not think that you are going to like this song" (low prediction, low reliability).

b) The extra "reliability" information enables new ways of showing recommendations to the users: from bidimensional plots (prediction, accuracy) to natural language, as demonstrated in the previous examples.

c) People feel more comfortable when they use fuzzy concepts (you will like it "a lot"; you "probably" will not like it) rather than absolute concepts (you will like it or will not like it).

d) Reliability values can be used to improve the accuracy of the results instead of improving explanations. As we will show later, some related works address this problem. Accuracy improvements are mainly achieved by filtering low reliability recommendations.

e) Reliability information is a powerful tool to improve users' trust in a RS: categorical mistakes (such as erroneous five-star recommendations) generate distrust on the part of the users who receive the recommendations. Conversely, people tolerate errors from noncategorical information. Reliability values are adequate information to modulate the probability of recommendations (it is "very likely" that you like...).

f) Reliability of predictions can be employed to properly handle a cold start problem. If all the predictions for a user or item have associated low reliability values, it can be considered a cold start problem, and recommendation methods that are designed ad hoc for cold start situations could be applied.

The kernel of an RS is its filtering approach; mainly, recommendations can be made based on demographic [2], content [48], social [40], context [44] and collaborative information. Due to their accurate results, a Collaborative Filtering (CF) based RS [42] is the most extended RS. Commercial RS architectures usually implement hybrid approaches [18] that include CF and other filtering sources. Research on reliability has been focused on the CF context, and methods and models have been published to provide reliability values that are mainly based on both the KNN approach and the MF approach. A KNN-based framework for providing reliability information was proposed in [17], where a specific implementation of the framework is explained. The key idea is that positive and negative factors are analyzed to obtain reliabilities: the greater are the positive factors, the higher are the reliability values; and the greater are the negative factors, the lower are the reliability values. A completely different KNN approach to obtaining reliabilities is proposed in [1], where low-quality rating profiles of users are enhanced by adding reliable votes. To estimate the confidence level of each prediction, [22] provides a full probability distribution of the item ratings rather than only a single score.

Trust information is a relevant concept that is related to reliability since both are usually correlated. In [11], the authors propose a confidence-based RS approach where trust and certainty are combined. The paper addresses the uncertainty that is derived from different rating behaviors of users on the same scale (e.g., 1 to 5 stars), and trust is applied to the opinions of the users. The use of ontologies and fuzzy linguistic models to represent users' trust in RSs is addressed in [29], where users' trust is employed in RS filtering rather than traditional similarity based on ratings. A Boltzmann machine learning model that is fed with trust information and users' preferences is proposed in [46], and the model merges both inputs to obtain improved recommendations. Trust-aware information was utilized in [3] to enrich the prediction process and obtain a "confidence" value for a recommendation. Trust-aware information [37] has been employed in other works to obtain reliability values for predictions and recommendations. Based on a reliability-based method, [33] improves the RS accuracy by providing a dynamic mechanism to construct trust networks of the users. It is particularly important to be aware that reliability values can be applied to improve the accuracy results by selecting the recommendations with the highest reliability values [17]. Some trust-aware confidence measures for rating predictions were proposed in [34]. Moreover, beyond accuracy, the entropy concept [45] can be applied to an RS to create a reliability measure. The initialization of MF is improved in [9] by using social trust information and a deep learning model.

The concept of reliability or confidence has also been recently addressed by [31,27]. In the first paper, the authors focus on several methods to embed awareness in an RS to determine whether each item should be suggested. Similar to our proposed method, the authors claim that confidence values can be used to filter the most reliable suggestions, which leads to an accuracy increase and a coverage decrease. The second paper [27] introduces the reliability concept in the review-to-rating conversion field, that is, the reliability in the conversion of textual reviews to implicit ratings. An aggregation of reliability measures was proposed in [1], where user-based reliability, rating profile reliability and item-based reliability were combined. By using this information, the authors obtain improved recommendations. Nevertheless, this solution is a memory-based solution, and thus, it has important drawbacks compared to our proposed model-based approach. Analogously, a confidence measure and a similarity confidence coefficient for link predictions are proposed in [43] in the context of the KNN memory-based method. Another memory-based approach [10] utilizes the reliability concept to improve traditional Pearson and cosine similarity measures. Previously, authors created the "credibility of a rating between users" based on the users' ratings on common items. RSs can also be improved by using context-aware prefiltering. Contextual recommendations were made in [39] using traditional CF and fuzzy rules to select items that are suitable for the specific situations of the users. In the same way, CF contextual postfiltering is performed in [38] to cast recommendations of restaurants with the aim of disseminating information about products and services. Moreover, an increase in effectiveness in advertising campaigns has been reached by combining evolutionary computation and fuzzy logic [26]. Additionally, online advertising recommendations have been improved using confidence information [35], which outperforms some current reinforcement learning schemes. Both [25,47] are research papers

that use an upper bound of confidence to improve recommendations. The former work, [25], has been tested in event-based social networks and synthetic datasets. On the other hand, in [47], a user's confidence and time context algorithm are proposed, where the user's behavior temporal information is exploited to improve the recommendation accuracy.

Despite its success, the previously mentioned research has two main drawbacks in the current RS reliability issue. First, the KNN approach is not applied due to its lack of accuracy and scalability. Conversely, our proposed method is not based on KNN but is based on the MF model, which is now a CF standard [49]. Second, social information is only available in a small subset of existing RSs, so social-based trust-aware approaches that are oriented to obtain reliability values cannot be considered universal solutions. Conversely, our proposed approach only uses the ratings matrix that contains the preferences of a set of users over a set of items, which is basic information for CF-based RSs. Since the proposed method does not rely on the KNN method, where identifying strategies for designing reliability measures are more obvious, it utilizes MF techniques and provides a brand-new MF-based approach to address RS reliabilities.

We will employ remarkable research in the MF field that has been applied to an RS in this paper as baselines. Probabilistic Matrix Factorization (PMF) [32] and Biased MF [21] are classic MF implementations. non-Negative Matrix Factorization (NMF) [23] avoids negative latent factor values and facilitates the assignment of semantic meanings to them. Bayesian non-Negative Matrix Factorization (BNMF) [16] provides an understandable probabilistic meaning to the latent factors, which lie within the range $[0, 1]$, and group users that share the same tastes. An improved CF Singular Value Decomposition (SVD++) is presented in [20]. The User Ratings Profile Model (URP) model [28] produces complete user rating profiles; each item rating is assigned by selecting a user attitude for the item. A two-level MF semantic architecture is designed in [50], which allows us to add reliabilities for any MF-based CF system as an add-on. We will use this method to obtain reliability values from the previously described MF implementations; it serves as a base for this paper's baselines.

One of the reasons that explains why the problem of reliability in RS has not been completely addressed is the lack of quality measures of reliability. Accuracy quality can be measured using a complete set of prediction and recommendation quality measures (Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), precision, recall, F1, etc.); reliability does not have analogous quality values. The underlying idea is that *"the more suitable reliability is, the better accuracy results will provide when applied: predictions with higher reliabilities should provide more accurate (less error) results, whereas we expect higher prediction errors on low reliability recommended items"* [50]. This is a recurrent concept that is addressed in several papers: *"The most common measurement of confidence is the probability that the predicted value is indeed true"* [41], *"This reliability measure is based on the usual notion that the more reliable a prediction, the less liable to be wrong"* [17], and *"Evaluations of recommenders for this task must evaluate the success of high-confidence recommendations, and perhaps consider the opportunity costs of excessively low confidence"* [15]. Using these concepts, the reliability framework proposed in [17] provides plots that show the prediction errors versus reliability inverse correlation. Based on confidence curve analysis, [30] proposed a method to estimate the reliability quality of reliability measures. The method tests the confidence curve by checking that its first value is higher than the last value (error decreases when reliability increases). The weak point of [17] is that it does not test all the confidence curve trends. In [6], the Reliability Prediction Index (RPI), which is a quality measure of reliability that tests the quality of reliabilities for predictions, is proposed. This quality measure returns a simple value, similar to customary accuracy quality measures, and scores the quality of a reliability measure for an RS.

The proposed model in this paper obtains the reliability value of each prediction and recommendation using three concepts: the machine learning classification concept, Bernoulli distribution and MF model. Instead of designing a regression model, we have chosen a classification approach, so the classes to classify correspond to the finite set of possible scores that can be assigned as rates to the items of the RS, e.g., $\{1, 2, 3, 4, 5\}$ stars in the MovieLens dataset [14]. In this way, our model returns the probabilities that a user will assign each of the possible scores to an item. In our MovieLens example, the model may return $[0.05, 0.05, 0.1, 0.6, 0.2]$, which means that the score 4 (probability 0.6) is the best prediction choice (for a given $\langle user, item \rangle$ pair). The classification model in the RS MF context is an innovative approach that is proposed in this paper: current MF implementations return a regression value for each prediction, e.g., 4.3. To design the explained classification-based MF approach, we set D individual classification tasks independent from each other, where D is the number of possible scores in the dataset ($D = 5$ in MovieLens). In this way, we have D separated classification processes. Each classification task for a given score focuses on the dichotomy that this score may or may not correspond to the score evaluated for each $\langle user, item \rangle$ pair. This behavior can be modeled using the Bernoulli distribution, that is, the discrete probability distribution of a random variable that takes the value 1 with probability p and the value 0 with probability $1 - p$. The formalization of the classification-based MF approach by using the Bernoulli distribution is detailed in Section 2. We have named this proposed method Bernoulli Matrix Factorization (BeMF) in allusion to the Bernoulli distribution and the MF model.

Fig. 1 shows the main ideas of the proposed BeMF model using a graphical example: the dataset displayed as a rating matrix is split into $D = 5$ matrices, which correspond to the possible scores (score 1, score 2, ..., score 5). Each matrix contains the basic information that is related to each corresponding score: if the rating corresponds to the actual score, we assign the code "1"; if the rating does not correspond to the score, we assign the code "0"; otherwise, the nonrated mark "–" is assigned. This set of $D = 5$ separated matrices feeds $D = 5$ independent Bernoulli factorizations, and as usual, each factorization generates its user's latent factors matrix and its item's latent factor matrix (gray row named *"Factors"*). From each pair of latent matrices, we can obtain predictions that utilize the inner product of the chosen user and item factors. Note that each prediction result does not correspond to the regression value of the rating, as is customary, but corresponds to the probability that the score will be classified as the correct score (gray row named *"Probabilities"*). The summation of these probabilities is equal
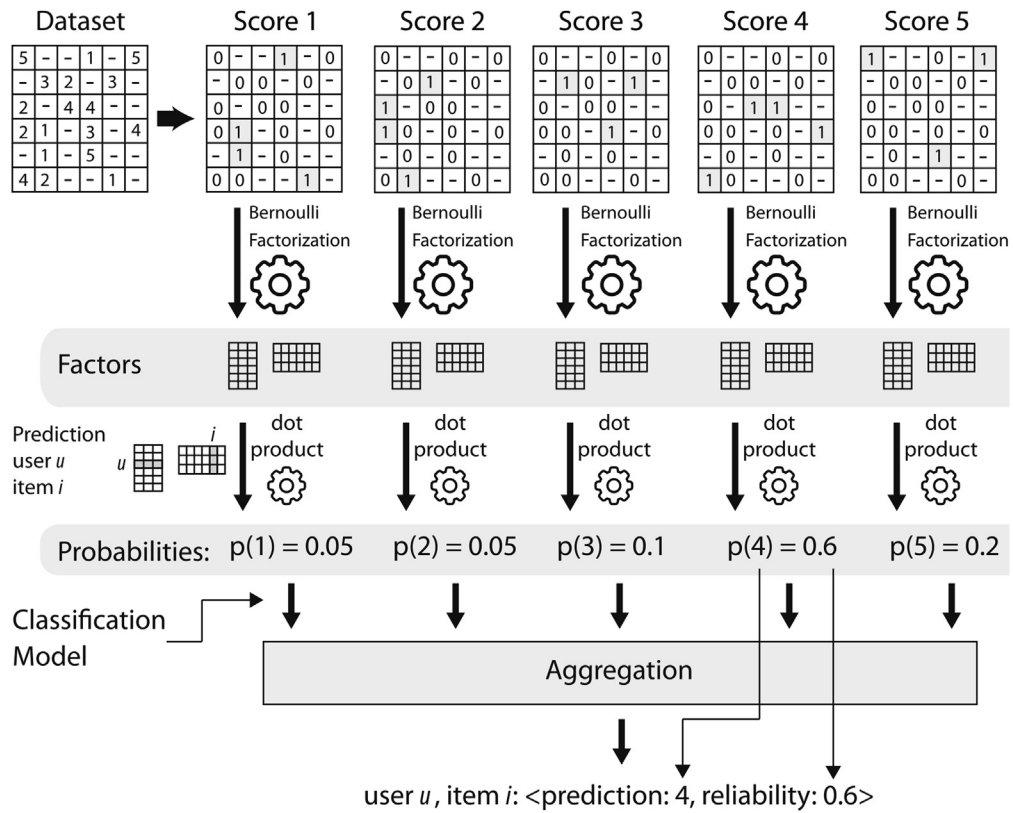
Fig. 1. Architecture of the proposed BeMF model.

to 1 since the BeMF model returns the probability distribution of the rating of user u to item i. The last step is to aggregate the $D = 5$ resulting probabilities to establish a classification "winner". Usually, the aggregation approach selects the score that is associated with the maximum probability. Note that we refer to Bernoulli factorization for the D MF parallel processes that are based on Bernoulli distribution, whereas we refer to the BeMF model for the complete architecture.

The proposed model provides an innovative approach to obtain reliability values in a CF RS. The main advantages of the model are listed as follows:

a) BeMF can be considered universal since it can be applied to a CF RS, which is are just based on ratings. Particularly, BeMF does not use social information, which is important since social information is only available in a reduced set of CF RSs. A variety of papers rely on social information and the trust and distrust of the associated information [3,33,34,37], but they cannot be applied to the most popular RS datasets.

b) BeMF is a model-based approach, and therefore, it is not limited to small- or medium-sized RSs, as discussed in papers on memory-based KNN [1,17,22]. In this way, the proposed approach is scalable, whereas the memory-based methods are not scalable.

c) The BeMF model is based on classification [4,5] rather than the usual regression approaches [16,20,21,23,28,32]. The machine learning classification approach provides more information than the regression approach; specifically, it returns a set of D probabilities. The D probabilities can be aggregated to obtain the expected reliability result. This context is richer than the isolated resultant value that is provided by the regression models.

d) The BeMF machine learning model is based on the Bernoulli distribution because it adequately adapts to the binary expected result for each of the D possible scores. The key concept is the simplification that we make by performing D simpler and more precise processes instead of the traditional complex regression task. To the best of our knowledge, a Bernoulli classification-based model has never been published in the reliability RS field. Additionally, the Bernoulli distribution directly provides, in a natural way, the expected probability that the prediction is true, that is, reliability.

e) Despite its simplicity, the use of the Bernoulli distribution endows BeMF with considerable flexibility. Eventually, the output of the BeMF model for user u and item i can be understood as a vector $(p_{u,i}^1, \ldots, p_{u,i}^D)$ of probabilities, where $p_{u,i}^s$ is the probability that u assigns the s-th score (for example, s stars) to item i. This distribution is a discrete probability distribution on the set of possible ratings. By using the underlying Bernoulli distribution, BeMF can actually model any discrete probability on the set of ratings or it fits the hyperparameters of a categorical distribution. Thus, it is completely general from a probabilistic point of view. The difference between BeMF and a model that is based on a cat-

egorical distribution is the training method: using the Bernoulli distribution for each score, we assume that the ratings are independent. For this reason, each probability p_s is computed only based on the appearances of the ratings s on the dataset. This process accelerates the training process and prevents overfitting.

This paper's hypothesis claims that an MF model can be designed to provide the reliability of each RS recommendation by using a classification-based approach and setting its learning stage in the Bernoulli distribution. Our contribution is the proposal and validation of an RS model that is capable of providing not only recommendations using a classification-based approach but also the reliability of these recommendations. Regarding the originality of our contribution, this is an innovative approach that provides explicit reliability values to RS users or improves the accuracy by selecting recommendations with the highest reliability values. Although reliability has been employed to improve accuracy, it has not become a common practice. Additionally, the reliability measures that are provided by this machine learning model are intrinsically linked to the model and do not depend on external methods or extended architectures as existing solutions.

The remainder of the paper has been structured as follows: In Section 2, the mathematical foundations of the proposed method are explained, and the resulting BeMF algorithm is provided. Section 3 shows the designed experiments and their results and discussions. Section 4 contains the main conclusions of the paper and outlines some future works.

2. Proposed model

Following the standard framework of CFs, we assume that we have N users that are evaluating M different items with a discrete set of possible scores $\mathscr{S} = \{s_1, \ldots, s_D\}$ (typically $\mathscr{S} = \{1, \ldots, 5\}$ as in the MovieLens dataset [14]). These ratings are collected in the rating matrix $R = (R_{u,i})$, where $R_{u,i} = s_\alpha$ if user $u \in [1, N]$ has assigned item $i \in [1, M]$ the score $s_\alpha \in \mathscr{S}$ and $R_{u,i} = \bullet$ if user u has not rated item i.

From this rating matrix, we generate D distinct matrices $R^{s_1}, \ldots, R^{s_D}$, which correspond to the possible scores that can be assigned to the items. In this way, fixed $s \in \mathscr{S}$; the matrix $R^s = (R^s_{u,i})$ is a (sparse) matrix such that $R^s_{u,i} = 1$ if user u voted item i with exactly score s; $R^s_{u,i} = 0$ if user u voted item i but with a different score from s and $R^s_{u,i} = \bullet$ if u did not rate item i.

Our model will attempt to fit the matrices $R^{s_1}, \ldots, R^{s_D}$ by performing D parallel matrix factorizations. The factors of each of the values will be independent, so the BeMF model is the juxtaposition of D different individual Bernoulli factorizations that fit binary matrices. Fig. 2 shows the plate diagram of the model, which is composed of D different factorizations (one factorization for each possible score), each of which is composed of N variables U_u (latent factors of each user) and M variables V_i (latent factors of each item). The hyperparameters of the model are two positive real values $\sigma_U, \sigma_V > 0$: an integer $k > 0$ (number of hidden factors) and a smooth logistic-like function ψ.

The problem of predicting the missing values of the matrix R^s is not a regression problem, as is customary in RS literature. The matrix R^s is a binary matrix, and each known value $R^s_{u,i}$ has a very specific meaning, namely, whether user u considered that item i deserves rate s. This problem is intrinsically a classification problem; in the simplest case it is a binary classification problem.

For this reason, the proposed system models the decision $R^s_{u,i}$, as a random variable of Bernoulli type with the probability of success $0 \leqslant p^s_{u,i} \leqslant 1$. This approach is compatible with the known fact that deciding to rate an item is not deterministic and depends on some stochastic psychological process that may vary from day to day. Hence, $p^s_{u,i}$ only measures the affinity of u for item i (more precisely, the reliability in the prediction, c.f., Section 2.2). In this way, the higher is the value $p^s_{u,i}$, the more likely u rates i with score s, independent of imponderable factors.

Regarding how the value $p^s_{u,i}$ is estimated, which is consistent with the usual assumptions in RS models, we will assume that the probability $p^s_{u,i}$ is given as the inner product $U^s_u \cdot V^s_i$ of some k-dimensional hidden user factors vector $U^s_u = (U^s_{u,1}, \ldots, U^s_{u,k})$ and hidden item factors vector $V^s_i = (V^s_{i,1}, \ldots, V^s_{i,k})$. However, the inner product $U^s_u \cdot V^s_i$ may attain any real
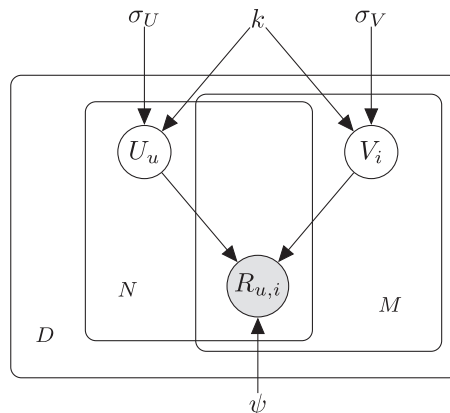

Fig. 2. Plate diagram of BeMF model.

value, so it is not a suitable quantity for $p_{u,i}^s$. To normalize the product $U_u^s \cdot V_i^s$, we use a logistic-like smooth function $\psi : \mathbb{R} \to [0, 1]$ in such a way that $0 \leqslant \psi(U_u^s \cdot V_i^s) \leqslant 1$, and thus, we may consider $p_{u,v}^s = \psi(U_u^s \cdot V_i^s)$. In contrast with other MF models, for each user and item, there is not a single instance of user and item factors but D different vectors of factors, one vector of factor for each possible rating.

2.1. Bernoulli factorization

In this section, we describe the mathematical formulation of the factorization model for each score. Hence, throughout this section, we fix $s \in \mathscr{S}$ as a possible score. To shorten the notation, we denote $R = R^s$ as the vote matrix, which corresponds to the score s. This matrix is a sparse binary matrix.

First, we fix the logistic-like function $\psi : \mathbb{R} \to [0, 1]$, i.e., a smooth increasing function with $\psi(x) \to 0$ when $x \to -\infty$ and $\psi(x) \to 1$ when $x \to \infty$. This hyperparameter will have the role of an activation function that will translate the inner product of the factors into a probability.

Fix user u and item i. The underlying probabilistic assumption of the Bernoulli factorization is that, given the latent vectors of user U_u and item V_i of dimension $k > 0$ (number of hidden factors), the rate $R_{u,i}$ is a Bernoulli distribution with the success probability $\psi(U_u \cdot V_i)$. Hence, the mass function of this random variable, $p(R_{u,i}|U_u, V_i)$, is given by

$$p(R_{u,i}|U_u, V_i) = \begin{cases} \psi(U_u V_i) & \text{if } R_{u,i} = 1, \\ 1 - \psi(U_u V_i) & \text{if } R_{u,i} = 0. \end{cases}$$

From this formula, given sample $R = (R_{u,i})$, we obtain that the associated likelihood, $\mathscr{L}(R|U, V)$, is

$$\mathscr{L}(R|U, V) = \prod_{R_{u,i} \neq \bullet} p(R_{u,i}|U_u, V_i) = \left(\prod_{R_{u,i}=1} \psi(U_u V_i) \right) \left(\prod_{R_{u,i}=0} 1 - \psi(U_u V_i) \right).$$

In this way, the log-likelihood function, $\ell(R|U, V) = \log \mathscr{L}(R|U, V)$, is given by

$$\ell(R|U, V) = \sum_{R_{u,i}=1} \log(\psi(U_u V_i)) + \sum_{R_{u,i}=0} \log(1 - \psi(U_u V_i)).$$

Now, as in PMF [32], let us assume spherical normal priors with a zero mean and standard deviation given by hyperparameters $\sigma_U, \sigma_V > 0$. In this case, we have the probability density functions

$$p(U_u) = \frac{1}{\sigma_U \sqrt{2\pi}} \exp\left(-\frac{||U_u||^2}{2\sigma^2} \right), \quad p(V_i) = \frac{1}{\sigma_V \sqrt{2\pi}} \exp\left(-\frac{||V_i||^2}{2\sigma^2} \right).$$

Hence, their prior likelihoods, $\mathscr{L}(U)$ and $\mathscr{L}(V)$ for users and items, respectively, are

$$\mathscr{L}(U) = \prod_{u=1}^N p(U_u) = \frac{1}{\sigma_U^N (2\pi)^{N/2}} \prod_{u=1}^N \exp\left(-\frac{||U_u||^2}{2\sigma_U^2} \right) = \frac{1}{\sigma_U^N (2\pi)^{N/2}} \exp\left(-\frac{\sum_{u=1}^N ||U_u||^2}{2\sigma_U^2} \right),$$

$$\mathscr{L}(V) = \prod_{i=1}^M p(V_i) = \frac{1}{\sigma_V^M (2\pi)^{M/2}} \prod_{i=1}^M \exp\left(-\frac{||V_i||^2}{2\sigma_V^2} \right) = \frac{1}{\sigma_V^M (2\pi)^{M/2}} \exp\left(-\frac{\sum_{i=1}^M ||V_i||^2}{2\sigma_V^2} \right).$$

Analogously, their prior log-likelihoods are

$$\ell(U) = -\frac{1}{2\sigma_U^2} \sum_{u=1}^N ||U_u||^2 + C_U, \quad \ell(V) = -\frac{1}{2\sigma_V^2} \sum_{i=1}^M ||V_i||^2 + C_V.$$

for some constants $C_U = -N \log(\sigma_U \sqrt{2\pi})$ and $C_V = -M \log(\sigma_V \sqrt{2\pi})$.

Moreover, the posterior likelihood, $\mathscr{L}(R)$, is

$$\mathscr{L}(R) = \mathscr{L}(R|U, V)\mathscr{L}(U)\mathscr{L}(V).$$

Therefore, the posterior log-likelihood is given by

$$\ell(R) = \ell(R|U, V) + \ell(U) + \ell(V) = \sum_{R_{u,i}=1} \log(\psi(U_u V_i)) + \sum_{R_{u,i}=0} \log(1 - \psi(U_u V_i)) - \frac{1}{2\sigma_U^2} \sum_{u=1}^N ||U_u||^2 - \frac{1}{2\sigma_V^2} \sum_{i=1}^M ||V_i||^2 + C,$$

where $C = C_U + C_V$ is a constant.

The maximum likelihood estimator is obtained by maximizing the posterior log-likelihood $\ell(R)$. For this purpose, the constant C is irrelevant, so it can be dismantled. Setting $\eta_U = \frac{1}{\sigma_U^2}$ and $\eta_V = \frac{1}{\sigma_V^2}$, the maximization problem can be converted to the minimization of the cost function

$$\mathscr{F}(U,V) = -\sum_{R_{u,i}=1} \log(\psi(U_u V_i)) - \sum_{R_{u,i}=0} \log(1 - \psi(U_u V_i)) + \frac{\eta_U}{2}\sum_{u=1}^{N}||U_u||^2 + \frac{\eta_V}{2}\sum_{i=1}^{M}||V_i||^2.$$

To optimize this cost function, we will use a standard gradient descent algorithm. Fixing user u_0 and an item i_0, denote the components of the latent vectors as $U_{u_0} = (U_{u_0,1}, \ldots, U_{u_0,k})$ and $V_{i_0} = (V_{i_0,1}, \ldots, V_{i_0,k})$. The directional derivatives of $\mathscr{F}$ in the directions $U_{u_0,a}$ and $U_{u_0,b}$ are then given by

$$\frac{\partial \mathscr{F}}{\partial U_{u_0,a}} = -\sum_{\{i\,|\,R_{u_0,i}=1\}} \frac{\psi\prime(U_{u_0}V_i)}{\psi(U_{u_0}V_i)}V_{i,a} + \sum_{\{i\,|\,R_{u_0,i}=0\}} \frac{\psi(U_{u_0}V_i)}{1-\psi(U_{u_0}V_i)}V_{i,a} + \eta_U U_{u_0,a},$$

$$\frac{\partial \mathscr{F}}{\partial V_{i_0,b}} = -\sum_{\{u\,|\,R_{u,i_0}=1\}} \frac{\psi\prime(U_u V_{i_0})}{\psi(U_u V_{i_0})}U_{u,b} + \sum_{\{u\,|\,R_{u,i_0}=0\}} \frac{\psi\prime(U_u V_{i_0})}{1-\psi(U_u V_{i_0})}U_{u,b} + \eta_V V_{i_0,b}.$$

For simplicity, let us set $\eta_U = \eta_V = \eta$. The associated gradient descent algorithm with step $\gamma > 0$ then updates the approximations at time T, which are denoted U_u^T and V_i^T, to the approximations at time $T+1$ by the rule

$$U_u^{T+1} = U_u^T + \gamma\left(\sum_{\{i\,|\,R_{u,i}=1\}} \frac{\psi\prime(U_u V_i)}{\psi(U_u V_i)}V_i - \sum_{\{i\,|\,R_{u,i}=0\}} \frac{\psi\prime(U_u V_i)}{1-\psi(U_u V_i)}V_i - \eta U_u\right),$$

$$V_i^{T+1} = V_i^T + \gamma\left(\sum_{\{u\,|\,R_{u,i}=1\}} \frac{\psi\prime(U_u V_i)}{\psi(U_u V_i)}U_u - \sum_{\{u\,|\,R_{u,i}=0\}} \frac{\psi\prime(U_u V_i)}{1-\psi(U_u V_i)}U_u - \eta V_i\right).$$

Example 2.1. If we consider $\psi = \text{logit}$ to be the logistic function, $\text{logit}(x) = \frac{1}{1+e^{-x}}$, then we have that $\text{logit}\prime(x) = \text{logit}(x)(1 - \text{logit}(x))$. Hence, the update rule reads

$$U_u^{T+1} = U_u^T + \gamma\left(\sum_{\{i\,|\,R_{u,i}=1\}} \frac{\text{logit}(U_u V_i)(1 - \text{logit}(U_u V_i))}{\text{logit}(U_u V_i)}V_i - \sum_{\{i\,|\,R_{u,i}=0\}} \frac{\text{logit}(U_u V_i)(1 - \text{logit}(U_u V_i))}{1 - \text{logit}(U_u V_i)}V_i - \eta U_u\right),$$

$$V_i^{T+1} = V_i^T + \gamma\left(\sum_{\{u\,|\,R_{u,i}=1\}} \frac{\text{logit}(U_u V_i)(1 - \text{logit}(U_u V_i))}{\text{logit}(U_u V_i)}U_u - \sum_{\{u\,|\,R_{u,i}=0\}} \frac{\text{logit}(U_u V_i)(1 - \text{logit}(U_u V_i))}{1 - \text{logit}(U_u V_i)}U_u - \eta V_i\right).$$

These update rules can be simplified as

$$U_u^{T+1} = U_u^T + \gamma\left(\sum_{\{i\,|\,R_{u,i}=1\}} (1 - \text{logit}(U_u V_i))V_i - \sum_{\{i\,|\,R_{u,i}=0\}} \text{logit}(U_u V_i)V_i - \eta U_u\right),$$

$$V_i^{T+1} = V_i^T + \gamma\left(\sum_{\{u\,|\,R_{u,i}=1\}} (1 - \text{logit}(U_u V_i))U_u - \sum_{\{u\,|\,R_{u,i}=0\}} \text{logit}(U_u V_i)U_u - \eta V_i\right).$$

2.2. BeMF model

By performing Bernoulli factorization on each possible score, we obtain a collection of user and item factor pairs $(U^{s_1}, V^{s_1}), \ldots, (U^{s_D}, V^{s_D})$ for the scores $\mathscr{S} = \{s_1, \ldots, s_D\}$. We can gather all this information to obtain the final output Φ. For each user u and item i, the output is the D-dimensional vector

$$\Phi(u,i) = \frac{1}{\sum_{\alpha=1}^{S}\psi(U_u^{s_\alpha}V_i^{s_\alpha})}\left(\psi(U_u^{s_1}V_i^{s_1}), \ldots, \psi(U_u^{s_D}V_i^{s_D})\right).$$

This vector is vector $\Phi(u,i) = (p_{u,i}^1, \ldots, p_{u,i}^D)$ with $0 \leqslant p_{u,i}^\alpha \leqslant 1$ and $\sum_\alpha p_{u,i}^\alpha = 1$. The value $p_{u,i}^\alpha$ may be interpreted as the probability that user u assigns score s_α to item i. In this way, $p_{u,i}^\alpha$ is the reliability that we have in predicting s_α. From this vector, if $\alpha_0 = \text{argmax}_\alpha \, p_{u,i}^\alpha$ is the mode of this probability distribution, we obtain.

- The prediction is given by $\hat{R}_{u,i} = s_{\alpha_0}$.
- The reliability of the prediction is given by $\rho_{u,i} = p_{u,i}^{\alpha_0}$.

Moreover, we can fix the threshold $\vartheta > 0$ (required prediction reliability), and we set $\hat{R}_{u,i} = \bullet$ (i.e., the prediction is unreliable if $\rho_{u,i} < \vartheta$). In this way, we filter unreliable predictions.

2.3. BeMF algorithm

Algorithm 1 contains the pseudocode for the training process of the BeMF model using the logistic function ($\psi = \text{logit}$) as the activation function, as shown in Example 2.1. The algorithm receives as inputs the sparse rating matrix (R), number of latent factors (k) and hyperparameters that are required by gradient descent optimization: learning rate (γ), regularization (η) and number of iterations (m). The algorithm returns as output two matrices that contain the learned latent factors: U contains the latent factors for each score s, user u and factor f, and V contains the latent factors for each score s, item i and factor f. These matrices can be used to compute both predictions and reliability, as described in Section 2.2. Note that to reduce the algorithm's processing time, the user update loop (lines 5–16) and item update loop (lines 19–30) can be executed in parallel for each user and item, respectively.

Algorithm 1: BeMF model fitting algorithm

    input : $R, k, \gamma, \eta, m, \mathcal{S}$
    output: $\mathbf{U}, \mathbf{V}$
1   Initialize $\mathbf{U} \leftarrow U(0,1), \mathbf{V} \leftarrow U(0,1)$
2   **repeat**
3   **for** *each possible score* $s \in \mathcal{S} = \{s_1, \ldots, s_D\}$ **do**
4   **for** *each user* u **do**
5   **for** *each item* i *rated by user* u: $R_{u,i}$ **do**
6   **for** *each* $f \in \{1, \ldots, k\}$ **do**
7   **if** $R_{u,i} = s$ **then**
8                    $\Delta_f^s \leftarrow \Delta_f^s + 1 - \text{logit}(U_u^s \cdot V_i^s) \cdot V_{i,f}^s$
9   **else**
10                   $\Delta_f^s \leftarrow \Delta_f^s + \text{logit}(U_u^s \cdot V_i^s) \cdot V_{i,f}^s$
11   **end**
12   **end**
13   **end**
14   **for** *each* $f \in \{1, \ldots, k\}$ **do**
15              $U_{u,f}^s \leftarrow U_{u,f}^s + \gamma \cdot (\Delta_f^s - \eta U_{u,f}^s)$
16   **end**
17   **end**
18   **for** *each item* u **do**
19   **for** *each user* u *that rated item* i: $R_{u,i}$ **do**
20   **for** *each* $f \in \{1, \ldots, k\}$ **do**
21   **if** $R_{u,i} = s$ **then**
22                    $\Theta_f^s \leftarrow \Theta_f^s + 1 - \text{logit}(U_u^s \cdot V_i^s) \cdot U_{u,f}^s$
23   **else**
24                   $\Theta_f^s \leftarrow \Theta_f^s + \text{logit}(U_u^s \cdot V_i^s) \cdot U_{u,f}^s$
25   **end**
26   **end**
27   **end**
28   **for** *each* $f \in \{1, \ldots, k\}$ **do**
29              $V_{i,f}^s \leftarrow V_{i,f}^s + \gamma \cdot (\Theta_f^s - \eta V_{i,f}^s)$
30   **end**
31   **end**
32   **end**
33   **until** *m iterations*

2.4. Running example

This section presents a running example of the BeMF model using the dataset with 4 users and 6 items shown in Table 1. To reduce the extension of the running example and make it more readable, we set the possible scores to 'like' (👍) and 'dislike' (👎). Note that despite the simplicity of this set of possible scores, it is a real set of possible scores that is employed in commercial services such as YouTube, Tinder or Steam.

The first step is to split the rating matrix into as many matrices as the number of possible scores. In this case, we have two possible scores, so we split the rating matrix of Table 1 into a matrix that encodes the 'like' ratings (Table 2 and a matrix that encodes the 'dislike' ratings (Table 2b). Recall that the absence of a rating must not be encoded into these matrices.

During the fitting process, the BeMF model must learn its parameters following Algorithm 1. These parameters are stored in four matrices that contain the latent factors of the users for the 'like' rating (), latent factors of the users for the 'dislike' rating (), latent factors of the items for the 'like' rating () and latent factors of the items for the 'dislike' rating (). For this example, we have fixed the number of latent factors to $k = 3$. Table 3 contains a random initialization of these parameters for the running example.

To learn the parameters, a gradient descent approach is employed, so we must perform m iterations following the update rules for the latent factors. For example, the update of the first factor f_1 of user u_1 for the like rating, which is denoted , is computed as

$$\Delta U^{👍}_{u_1, f_1} = (1 - logit(U_{u_1} V_{i_2})) V_{i_2, f_1} + (1 - logit(U_{u_1} V_{i_4})) V_{i_4, f_1} - logit(U_{u_1} V_{i_1}) V_{i_1, f_1}.$$

From this value, we can improve the hidden factor by updating

$$U^{👍}_{u_1, f_1} \leftarrow U^{👍}_{u_1, f_1} + \gamma \left(\Delta U^{👍}_{u_1, f_1} - \eta U^{👍}_{u_1, f_1} \right).$$

Note that only the items rated by user u_1 (i.e., i_1, i_2 and i_4) update his/her latent factors.

Table 4 contains the latent factors after one iteration ($m = 1$) using the learning rate $\gamma = 0.1$ and regularization $\eta = 0.01$.

Once the BeMF model has been trained, predictions can be computed by obtaining the score that maximizes the probability in the classification task. For example, to predict the rating of user u_1 to the item i_3, $\hat{R}_{u_1, i_3}$, we must compute the probability distribution of this rating, $\Phi(u_1, i_3)$, as

$$\Phi(u_1, i_3) = \frac{1}{logit\left(U^{👎}_{u_1} V^{👎}_{i_3}\right) + logit\left(U^{👍}_{u_1} V^{👍}_{i_3}\right)} \left(logit\left(U^{👎}_{u_1} V^{👎}_{i_3}\right), logit\left(U^{👍}_{u_1} V^{👍}_{i_3}\right) \right) \tag{1}$$

$$= \frac{1}{0.71 + 0.62} (0.71, 0.62) = (0.53, 0.47). \tag{2}$$

Henceforth, the BeMF model will return 👎 with the reliability $\rho_{u_1, i_3} = 0.53$.

Table 1
Rating matrix used in the running example. Possible scores are set to like (👍) and dislike (👎).

$R_{u,i}$	i_1	i_2	i_3	i_4	i_5	i_6
u_1	👎	👍		👍		
u_2	👍		👍	👎	👎	
u_3		👍	👍			👎
u_4	👎		👎		👍	👍

Table 2
Resulting matrix after splitting the running example's rating matrix.

$R^{👍}_{u,i}$	i_1	i_2	i_3	i_4	i_5	i_6	$R^{👎}_{u,i}$	i_1	i_2	i_3	i_4	i_5	i_6
u_1	0	1		1			u_1	1	0		0		
u_2	1		1	0	0		u_2	0		0	1	1	
u_3		1	1			0	u_3		0	0			1
u_4	0		0		1	1	u_4	1		1		0	0

(a) Like ratings.   (b) Dislike ratings.

Table 3
Initial random BeMF model parameters in the running example. The number of latent factors has been fixed to $k = 3$.

$U_{u,f}^{\mathbf{\Uparrow}}$	f_1	f_2	f_3
u_1	0.99	0.26	0.55
u_2	0.77	0.77	0.85
u_3	0.20	0.27	0.35
u_4	0.11	0.96	0.13

(a) Users latent factors for the like rating.

$U_{u,f}^{\mathbf{\Downarrow}}$	f_1	f_2	f_3
u_1	0.61	0.83	0.47
u_2	0.12	0.02	0.54
u_3	0.11	0.41	0.07
u_4	0.81	0.92	0.52

(b) Users latent factors for the dislike rating.

$V_{i,f}^{\mathbf{\Uparrow}}$	f_1	f_2	f_3
i_1	0.91	0.15	0.27
i_2	0.54	0.54	0.79
i_3	0.31	0.57	0.09
i_4	1.00	0.83	0.75
i_5	0.68	0.03	0.05
i_6	0.35	0.50	0.75

(c) Items latent factors for the like rating.

$V_{i,f}^{\mathbf{\Downarrow}}$	f_1	f_2	f_3
i_1	0.92	0.53	0.67
i_2	0.40	0.24	0.12
i_3	0.64	0.22	0.89
i_4	0.64	0.86	0.60
i_5	0.51	0.12	0.41
i_6	0.92	0.23	0.75

(d) Items latent factors for the dislike rating.

Table 4
BeMF model parameters after one iteration in the running example.

$U_{u,f}^{\mathbf{\Uparrow}}$	f_1	f_2	f_3
u_1	0.96	0.27	0.56
u_2	0.72	0.75	0.81
u_3	0.21	0.29	0.34
u_4	0.05	0.92	0.12

(a) Users latent factors for the like rating.

$U_{u,f}^{\mathbf{\Downarrow}}$	f_1	f_2	f_3
u_1	0.55	0.76	0.43
u_2	0.07	0.02	0.49
u_3	0.10	0.39	0.05
u_4	0.73	0.90	0.46

(b) Users latent factors for the dislike rating.

$V_{i,f}^{\mathbf{\Uparrow}}$	f_1	f_2	f_3
i_1	0.82	0.13	0.24
i_2	0.58	0.58	0.84
i_3	0.31	0.58	0.09
i_4	0.95	0.79	0.71
i_5	0.66	0.03	0.05
i_6	0.34	0.48	0.72

(c) Items latent factors for the like rating.

$V_{i,f}^{\mathbf{\Downarrow}}$	f_1	f_2	f_3
i_1	0.90	0.52	0.66
i_2	0.36	0.21	0.11
i_3	0.57	0.20	0.80
i_4	0.61	0.82	0.58
i_5	0.50	0.12	0.40
i_6	0.89	0.22	0.73

(d) Items latent factors for the dislike rating.

3. Proposed model evaluation

This section contains a detailed explanation of the experiments that are carried out to evaluate the proposed model. Section 3.1 describes the experimental setup that defines the datasets, baselines and quality measures that are employed during the evaluation. Section 3.2 includes the experimental results and a comparison of the performance of the proposed method regarding the selected baselines. All experiments have been conducted using Collaborative Filtering for Java (CF4J) [36]; their source code is available at https://github.com/ferortega/bernoulli-matrix-factorization.

3.1. Experimental setup

Experimental evaluation was conducted using the MovieLens [14], FilmTrust [13] and MyAnimeList datasets. These datasets have been selected to assess the impact of splitting the rating matrix into binary rating matrices using different discrete

sets of possible scores. In this way, the MovieLens dataset contains ratings from 1 star to 5 stars FilmTrust ratings range from 0.5 to 4.0 with half increments; and the MyAnimeList dataset restricts its possible scores to the range 1 to 10. Moreover, to ensure the reproducibility of these experiments, all of them were carried out using the benchmark version of these datasets, which are included in CF4J [36]. The main features of these datasets are shown in Table 5.

According to Section 1, all the MF models that are presented in the literature are capable of estimating a user's rating prediction for an item, but only a few MF models provide the reliability of their predictions and recommendations. Baselines have been selected to supply a heterogeneous representation of all the existing MF models. Table 6 contains the selected baselines and their generated outputs. Note that the proposed model, BeMF, can estimate the user's rating predictions and compute the reliability of the predictions and recommendations, as described in Section 2.2.

The chosen baselines contain several hyperparameters that must be tuned. We analyzed the prediction error of the baseline models with different values of these hyperparameters by performing a grid search optimization that minimizes the mean absolute prediction error. Table 7 contains the hyperparameters that are generated from this optimization process for each baseline and dataset.

In the same way, the BeMF model has some hyperparameters that must be tuned: k, which denotes the number of latent factors of the model; γ, which represents the learning rate of the gradient descent optimization algorithm; η, which controls the regularization to avoid overfitting; and m, the number of iterations of the fitting process. According to Algorithm 1, we have fixed the activation function to the logistic function. As with the baselines, a grid search optimization was carried out to compare the prediction error of the proposed model using thousands of combinations of the required hyperparameters. We evaluated the following intervals of values for each hyperparameter:

- k: from 2 to 8 latent factors with increments of 2.
- γ: from 0.002 to 0.02 with increments of 0.002.
- η: from 0.01 to 0.2 with increments of 0.01.
- m: from 50 to 100 with increments of 25.

Table 5
Main parameters of the datasets used in the experiments.

Dataset	Number of users	Number of items	Number of ratings	Number of test ratings	Possible scores
MovieLens	6,040	3,706	911,031	89,178	1 to 5 stars
FilmTrust	1,508	2,071	32,675	2,819	0.5 to 4.0 with half increments
MyAnimeList	69,600	9,927	5,788,207	549,027	1 to 10

Table 6
Output generated by MF models selected as baselines.

Model	Prediction	Prediction reliability	Recommendation reliability
BiasedMF [22]	✓	✗	✗
BNMF [17]	✓	✗	✓
NMF [23]	✓	✗	✗
PMF [32]	✓	✗	✗
SVD++[20]	✓	✗	✗
URP [28]	✓	✓	✓

Table 7
Baseline hyperparameters generated from grid search optimization.

Method	MovieLens	FilmTrust	MyAnimeList
PMF	factors = 8, $\gamma = 0.01$, $\lambda = 0.045$	factors = 4, $\gamma = 0.015$, $\lambda = 0.1$	factors = 10, $\gamma = 0.005$, $\lambda = 0.085$
BiasedMF	factors = 6, $\gamma = 0.01$, $\lambda = 0.055$	factors = 2, $\gamma = 0.015$, $\lambda = 0.15$	factors = 10, $\gamma = 0.01$, $\lambda = 0.085$
NMF	factors = 2	factors = 2	factors = 2
BNMF	factors = 10, $\alpha = 0.6$, $\beta = 5$	factors = 10, $\alpha = 0.4$, $\beta = 25$	factors = 4, $\alpha = 0.5$, $\beta = 5$
URP	factors = 10	factors = 4	factors = 8
SVD++	factors = 4, $\gamma = 0.0014$, $\lambda = 0.05$	factors = 2, $\gamma = 0.0014$, $\lambda = 0.02$	factors = 4, $\gamma = 0.0015$, $\lambda = 0.1$

The total number of all the possible combinations of these parameter values is 2400. In the MovieLens dataset, the minimum prediction error was obtained using $k = 2$, $\gamma = 0.006$, $\lambda = 0.16$ and $m = 100$ iterations. In the same way, in the Film-Trust dataset, the hyperparameter values that minimize the prediction error are $k = 2$, $\gamma = 0.02$, $\lambda = 0.06$ and $m = 75$ iterations. In MyAnimeList, the best predictions are estimated using $k = 4$, $\gamma = 0.004$, $\lambda = 0.1$ and $m = 100$ iterations.

The quality of the predictions and recommendations provided by a CF-based RS must be evaluated using standard quality measures. To measure the quality of the predictions, we define MAE (Eq. (3)) as the mean absolute difference between the test ratings ($R_{u,i}$) and their predictions ($\hat{R}_{u,i}$) and define coverage (Eq. (4)) as the proportion of test ratings that a CF can predict ($\hat{R}_{u,i} \neq \bullet$) concerning the total number of test ratings. Here, R^{test} is the collection of pairs $\langle u, i \rangle$ of user u and item i in the test split of the dataset, and $\#R^{test}$ denotes its cardinality.

$$\text{MAE} = \frac{1}{\#R^{test}} \sum_{\langle u,i \rangle \in R^{test}} \| R_{u,i} - \hat{R}_{u,i} \|. \tag{3}$$

$$\text{Coverage} = \frac{\#\{\langle u, i \rangle \in R^{text} | \hat{R}_{u,i} \neq \bullet\}}{\#R^{test}}. \tag{4}$$

On the other hand, fixing $n > 0$ to measure the quality of the top n recommendations, we can consider two adapted quality measures. The first quality measure is precision, which is given by the averaged proportion of successful recommendations that are included in the recommendation list of user u of maximum length n, which is denoted T_u^n, with respect to the size of the recommendation list (Eq. (5)). The second quality measure is recall, which is the averaged proportion of successful recommendations included in the recommendation list of user u, T_u^n, with respect to the total number of test items that user u likes (Eq. (6)).

$$\text{precision} = \frac{1}{N} \sum_{u=1}^{N} \frac{\{i \in T_u^n | R_{u,i} \geq \theta\}}{\#T_u^n}. \tag{5}$$

$$\text{recall} = \frac{1}{N} \sum_{u=1}^{N} \frac{\{i \in T_u^n | R_{u,i} \geq \theta\}}{\{i \in R_u^{test} | R_{u,i} \geq \theta\}}. \tag{6}$$

In the previous formulae, u runs over the users of the dataset, N is the total number of users, R_u^{test} is the collection of items rated by user u in the test split and θ is a threshold to discern if a user likes an item ($R_{u,i} \geq \theta$) or not ($R_{u,i} < \theta$).

3.2. Experimental results

The most popular MF based CF methods are constructed as regressors because they assume that rating values are continuous, and consequently, their predictions are real values. Conversely, the BeMF model has been designed to work with discrete rating values so that it solves the classification problem in which the classes are the possible scores with which items are rated. A confusion matrix is a graphical representation tool that is used to evaluate the output of a classifier. Columns contain the predicted labels, rows contain the real labels and cells denote the proportion of samples of a label that have been predicted with another label. Fig. 3 contains the confusion matrix of the BeMF model's predictions for the MovieLens (a),
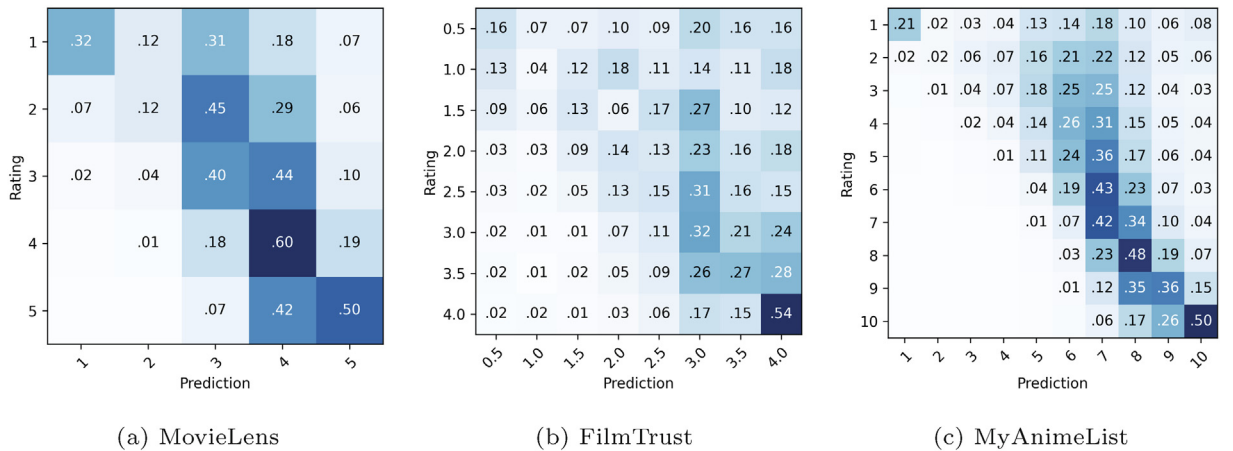

(a) MovieLens   (b) FilmTrust   (c) MyAnimeList

Fig. 3. Confusion matrix of the classification performed by BeMF.

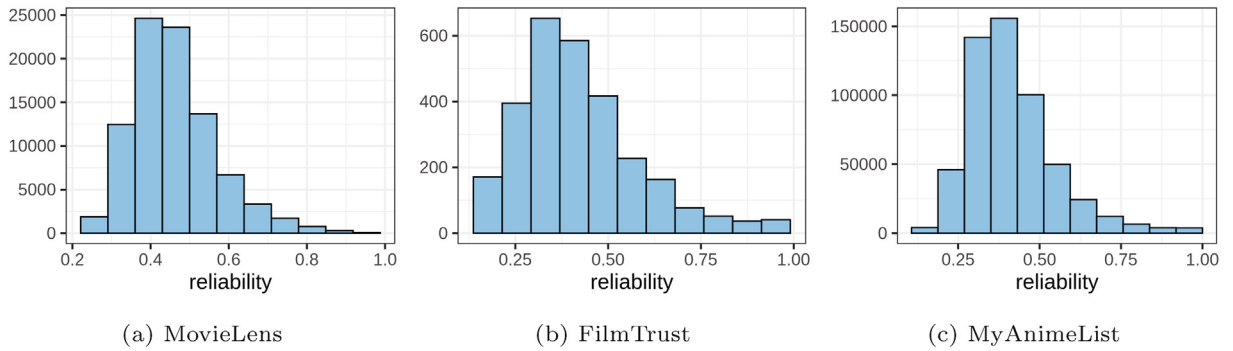(a) MovieLens   (b) FilmTrust   (c) MyAnimeList

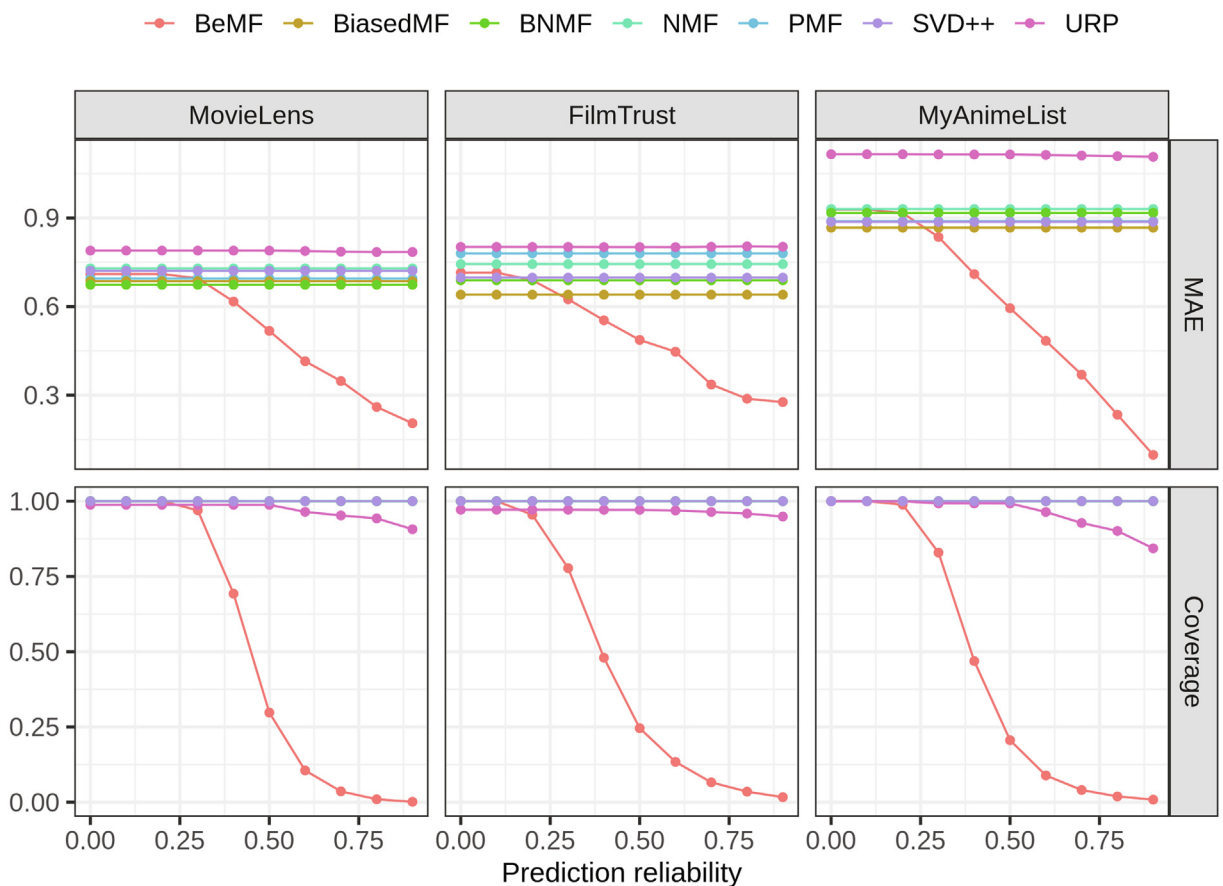**Fig. 4.** Histogram of the reliability values of BeMF's test predictions.

Fig. 5. Quality of the predictions measured by MAE and *coverage*. Predictions with lower reliability than those indicated on the *x*-axis are filtered out.

FilmTrust (b) and MyAnimeList (c) datasets. We observe that most of the predictions are correct. However, the model tends to return predictions for high scores due to the bias of the datasets employed in CF, where users tend to rate only items that they like.

The BeMF model is capable of providing not only rating prediction but also the reliability of this prediction. The reliability values indicate the model's confidence in the predictions. Fig. 4 shows the histograms with the distribution of the reliability values of the test predictions in the MovieLens (a), FilmTrust (b) and MyAnimeList (c) datasets. These histograms show that the reliability distribution shape is the same for the three tested datasets: most of the reliability values belong to the interval
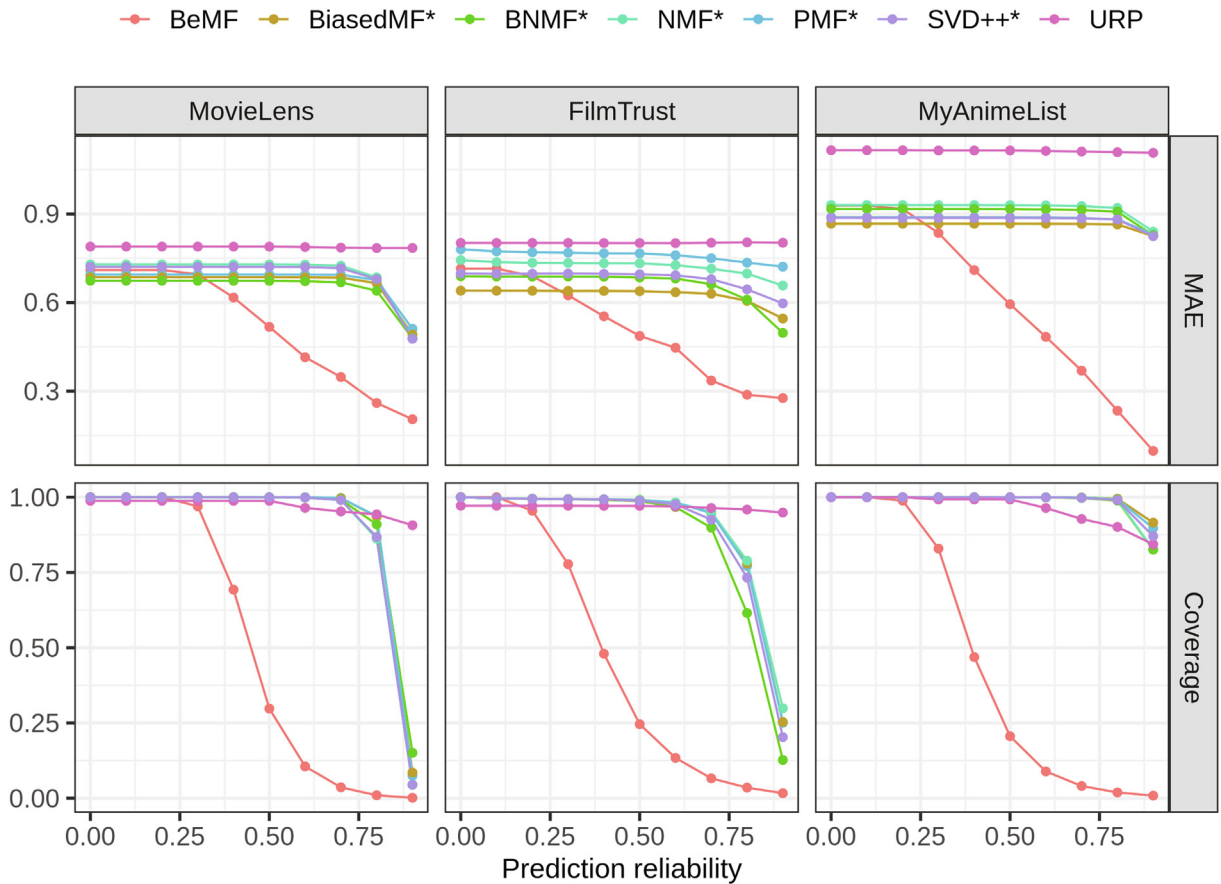
Fig. 6. Quality of the predictions measured by MAE and *coverage*. Predictions with lower reliability than those indicated on the *x*-axis are filtered out. Models with the suffix * obtain their reliability values using [50].

Table 8
RPI values for the native reliability values provided by BeMF and the reliability values enforced by [50].

Dataset	BeMF (native)	BeMF (enforced)
MovieLens	0.09344191	0.03607168
FilmTrust	0.17187947	0.03277490
MyAnimeList	0.17087788	0.03386983

$[0.3, 0.5]$, which means a confidence interval between 30% to 50% in the predicted rating, and there are few reliabilities with values greater than 0.75.

The reliability value allows us to calibrate the output of the BeMF model. By filtering out less reliable predictions, we decrease the coverage of the model (i.e., some predictions cannot be issued because the model does not have enough confidence to make them) but increase the prediction accuracy. It is reasonable to think that predictions with high reliability are more obvious than predictions with low reliability. For example, if a user has positively rated *Star Wars Episode IV (A New Hope)* and *Star Wars Episode V (The Empire Strikes Back)*, the model will have high confidence in the positive interest of the user to *Star Wars Episode VI (Return of the Jedi)* and will assign a high reliability value to this prediction. Conversely, the same model will have less confidence in the interest of the user in other sci-fi movies, such as *Interstellar* or *Gravity*, and will assign lower reliability values to these predictions.

Fig. 5 analyzes the impact of the reliability measure on the quality of the predictions. We have contrasted the prediction error against the predictability of the model using the MAE and coverage quality measures that are defined in Eqs. (3), (4), respectively. The plots were generated by filtering all the predictions with lower reliability than the reliability denoted on the x-axis. Note that only the models that return prediction reliabilities (refer to Table 6) can filter their predictions; the remaining predictions are shown as a horizontal line in the plot. All the plots exhibit the same trend: The prediction error is reduced when unreliable predictions are filtered out. Consequently, the coverage of the model also decreases. In any case,
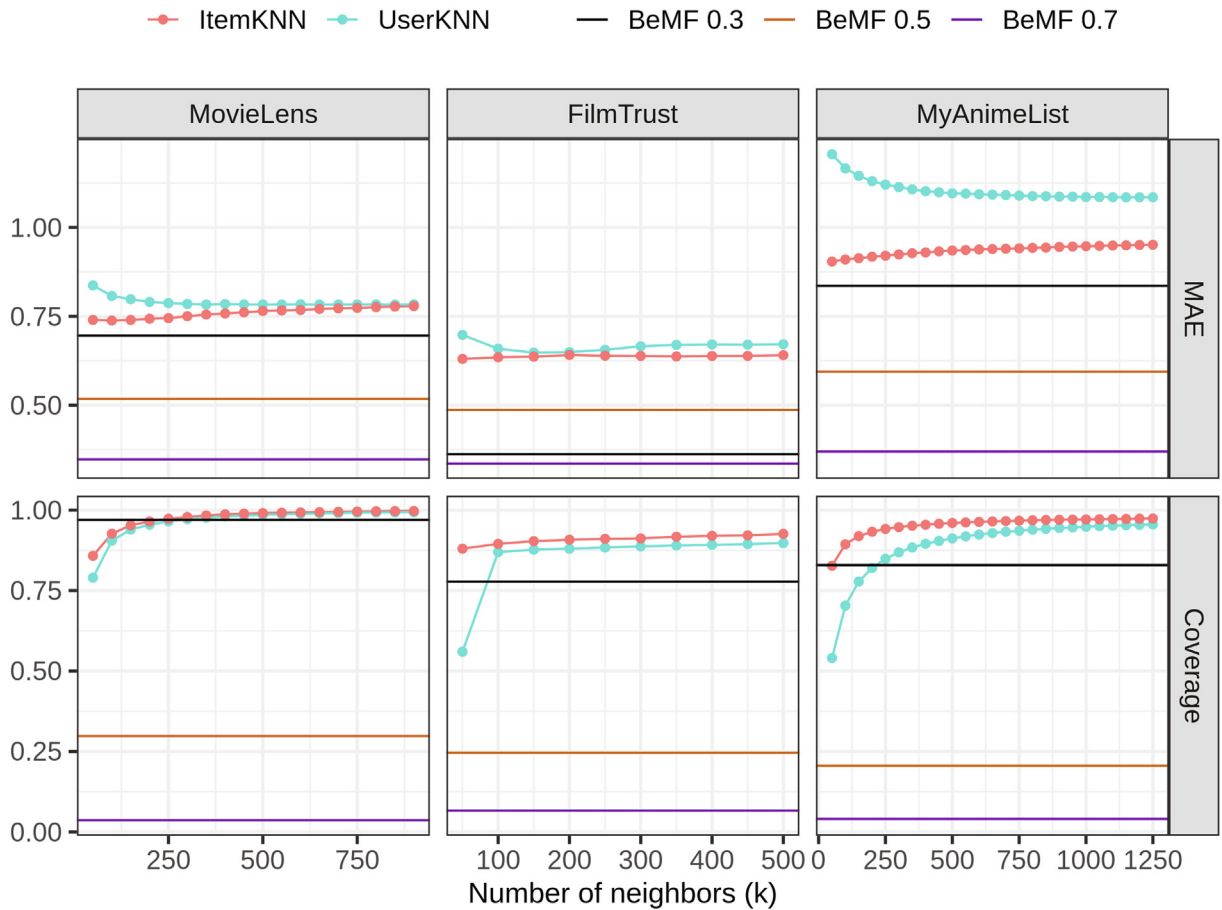
Fig. 7. Comparison of the quality of the predictions returned by CF based on KNN with respect to BeMF. BeMF is shown by filtering out predictions with a reliability lower than 0.3 (BeMF 0.3), 0.5 (BeMF 0.5) or 0.7 (BeMF 0.7).

the prediction accuracy improvement of the BeMF model with respect to the evaluated baselines is significant when the coverage is between 50% and 75%.

Despite the satisfactory results reported by the BeMF model in the previous experiment, we consider it unfair because not some recommendation models do not benefit from filtering out unreliable predictions. Several methods exist to extend CF-based RSs, which add reliability to their predictions. One of the most popular methods is presented in [50]. In this paper, the authors propose computing an auxiliary MF model that factorizes a matrix that contains the prediction errors. The authors claim that high reliability values are produced by low prediction errors, and vice versa.

Fig. 6 contains the results of repeating the experiment shown in Fig. 5 but adding a reliability value to the predictions performed by BiasedMF, BNMF, NMF, PMF and SVD++ using [50]. The trend observed in the previous experiment is again observed: When unreliable predictions are filtered out, the error decreases and the coverage decreases. However, the reliability values enforced by [50] do not achieve results that are as satisfactory as those achieved by the native reliability implemented by BeMF.

At this point, it is reasonable to consider that the tuple ⟨prediction, reliability⟩ that is returned by the BeMF model substantially improves the quality of predictions thanks to the accuracy of the reliability values. To confirm the accuracy of the native reliability values returned by the BeMF model, we have compared them against the reliability values enforced by [50] when applied to the BeMF model's predictions. To compare both quality measures, we will use RPI [6]. Table 8 contains the results of this experiment. We can observe that the native BeMF reliability values significantly improve those enforced by [50].

Previous experiments demonstrate that the satisfactory results are a consequence of including the reliability of the predictions in the model output. As we previously stated, BeMF can be calibrated to return few but very reliable predictions or many but less reliable predictions. This property is very innovative in MF but not in CF. KNN-based CF systems contain a hyperparameter that tunes the number of neighbors that are considered to issue the prediction. If a low number of neighbors is selected, few predictions are returned, but they are more accurate. Conversely, if a high number of neighbors is selected, many predictions are returned, but they are less accurate. BeMF should significantly improve the quality of the predictions
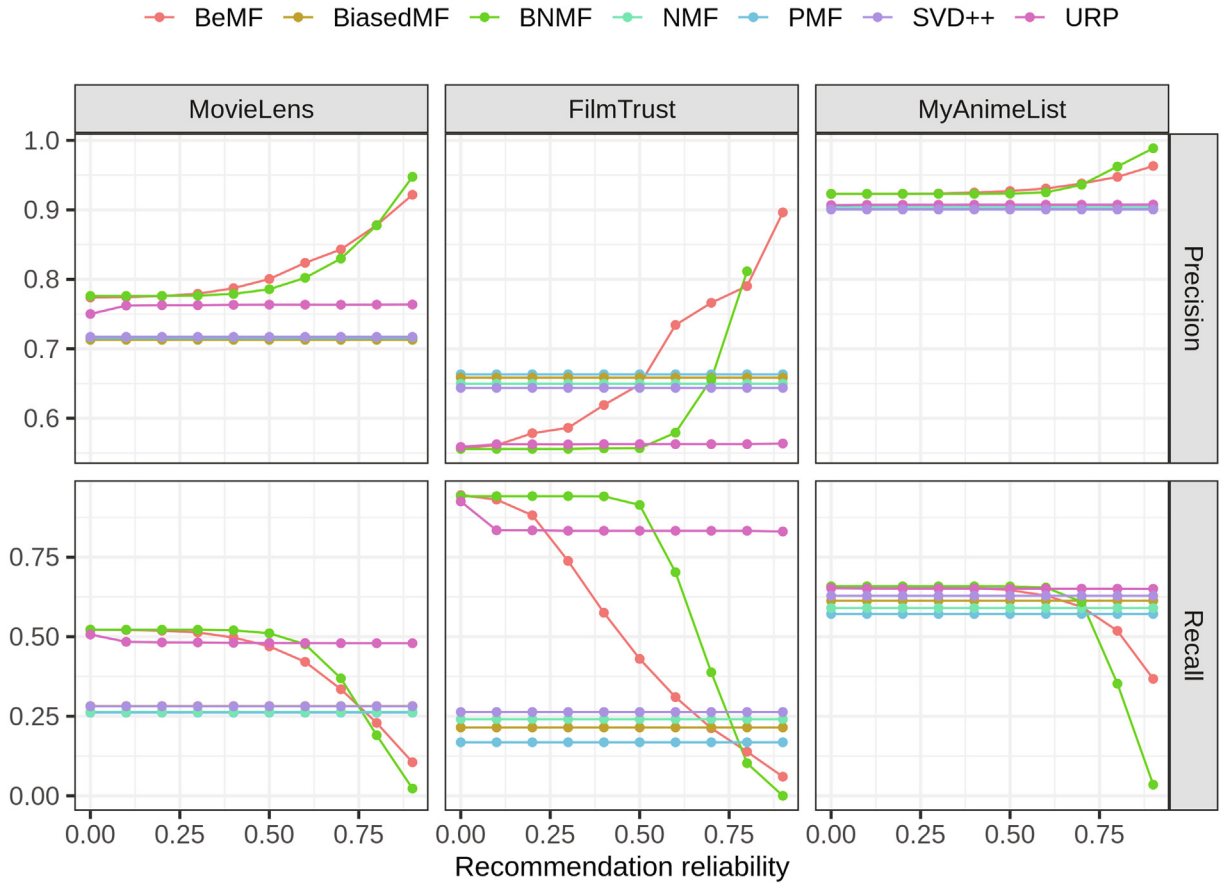
Fig. 8. Quality of the recommendations measured by precision and recall. Recommendations with a lower probability than those denoted on the *x*-axis are filtered out.

returned by KNN. To evaluate this hypothesis, BeMF has been compared with user-based KNN and item-based KNN using JMSD [8] as a similarity metric. The results of this comparison are shown in Fig. 7. We observe that the predictions returned by BeMF have less error than the predictions returned by KNN-based methods, independent of whether the predictions returned by BeMF have been filtered with a reliability lower than 0.3 (BeMF 0.3), 0.5 (BeMF 0.5) or 0.7 (BeMF 0.7).

To analyze the quality of the recommendations, we measured the precision (Eq. (5)) and recall (Eq. (6)) of the top 10 recommendations ($n = 10$). To discern whether a recommendation is right or wrong, we have fixed the threshold that determines the items that interest a user based on his/her test ratings to $\theta = 4$ in MovieLens, $\theta = 3.5$ in FilmTrust and $\theta = 7$ in MyAnimeList.

Recommendation lists, including the top 10 test items, were built in two ways. If the recommendation method provides the reliability of the recommendations (refer to Table 6), we have selected the top 10 test items with the highest probability of being liked by the user (reliability $\rho_{u,i} \geqslant \vartheta$). On the other hand, if the recommendation method does not provide the reliability of the recommendations, we have selected the top 10 test items with the highest prediction ($\hat{R}_{u,i}$), excluding those who have a prediction lower than θ. Note that ϑ and θ have different roles: the former is a threshold in the desired reliability, while the latter is a threshold in the predicted score. The results of this comparison are shown in Fig. 8.

This plot shows that when recommendations with a lower probability than those denoted on the *x*-axis are filtered out, the precision value increases and the recall value declines. BeMF is the recommendation method that provides the best precision when the item's minimum probability of being liked is between 0.5 and 0.75. BNMF achieves the best precision with probabilities higher than 0.75. However, the recall value is so low in these cases that the recommendation ability of the method is seriously impaired. Therefore, BeMF provides a better balance between the quality of the recommendations (precision) and the capability of the recommendation items (recall).

4. Conclusions

In this paper, we have presented the BeMF model, which is an MF-based CF algorithm that returns not only predictions for items that are not rated by users but also the reliability of these predictions. This outcome is achieved by addressing the recommendation process as a classification problem rather than a regression problem. The pair ⟨prediction, reliability⟩ that is

returned by the BeMF model allows us to calibrate its output to obtain the proper balance between the quality of the predictions and the quantity of the predictions (i.e., to decrease the prediction error by reducing the model's coverage). Likewise, the BeMF model allows us to estimate the reliability of a recommendation by obtaining the probability that a user has interest in an item. This approach is innovative approach that improves accuracy by selecting recommendations with the highest reliability values and providing explicit reliability values to the users.

The experimental results carried out on the MovieLens, FilmTrust and MyAnimeList datasets show the distinct superiority of the BeMF model against not only other MF models but also alternative approaches such as KNN when filtering out unreliable predictions and recommendations. BeMF has also proven to achieve a better balance between the quality of the recommendations (precision) and the capability of recommending items (recall). The authors of this work are committed to reproducible science, so the source code of all the experiments conducted in this article is publicly accessible and was tested against benchmark datasets that are included in the open source project CF4J.

One of the most important reasons for this outperformance is that reliability is a native concept in BeMF and does not depend on external methods or extended architectures as existing solutions. Using the Bernoulli distribution, BeMF can model the rating action as a stochastic process, in which the decision of assigning a particular score is not deterministic but depends on some psychological and environmentally imponderable factors of the users (e.g., its frame of mind) and the items (e.g., hype around a long-awaited movie). In this way, reliability is intrinsically linked to BeMF, and in some sense, is the main focus of the algorithm: prediction is just a byproduct that is obtained by choosing the most reliable score. This focus is in sharp contrast with reliability measures that can be artificially added to other methods, which are based on either a secondary MF to predict the expected error [50] or a similarity measure that is computed on the selected neighbors [8].

On the other hand, the URP returns a native reliability measure, which is provided via a probabilistic model of the behaviors of the users (essentially a multinomial distribution). However, in this case, the proposed distribution imposes strong topological restrictions on the obtained reliability measures. If the URP returns that the most likely score for an item from a given user is $s \in \mathscr{S}$, then the shape of the multinomial distribution forces that $s - 1$ and $s + 1$ are also very likely ratings. In particular, the obtained distributions are unimodal, so reliability is essentially a measure of closeness to the mode. Conversely, BeMF supports any discrete distribution, including multimodal distributions, as output. This finding implies that BeMF's reliability is much richer, and it may reach a valley between two modes, e.g., if an item is very controversial and is usually rated very well or very bad.

In future work, we propose conducting a more in-depth analysis of the impact of the proposed model beyond the quality of the predictions and recommendations. We propose studying the quality of BeMF beyond accuracy quality measures, such as novelty, diversity or discovery. Similarly, we propose to study the stability of the model against shilling attacks that are performed to nuke or promote particular items.

Another interesting research area is the use of other probability distributions as underlying assumptions for the MF method. In this paper, we modeled the ratings as a *D*-dimensional vector of independent Bernoulli random variables, which enables efficient parallel training for each possible rating and reduces the number of hyperparameters of the model. Nevertheless, some dependency among random scores can be imposed. This dependency, such as modeling votes as a normal distribution in PMF [32] or as a binomial distribution in BNMF [16], has been previously reported in the literature. A prospective work would be to explore other types of distributions that allow us to improve the performance in specific tasks. For instance, modeling the ratings as an exponential distribution allows the model to capture and exploit rare events, such as recommending underrated movies to specific users or providing accurate predictions to cold users.

On the other hand, this new model can be extended. Therefore, we propose incorporating social and content information to improve the quality of the predictions. We also propose adapting BeMF to make recommendations to user groups. An interesting research area would be to add time information to the model to reflect the changes in the opinions of the users over time.

CRediT authorship contribution statement

Fernando Ortega: Conceptualization, Software, Investigation, Writing - original draft, Writing - review & editing, Supervision, Project administration. **Raúl Lara-Cabrera:** Software, Resources, Data curation, Writing - original draft, Writing - review & editing, Visualization. **Ángel González-Prieto:** Conceptualization, Formal analysis, Writing - original draft, Writing - review & editing. **Jesús Bobadilla:** Validation, Writing - original draft, Writing - review & editing, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

References

[1] S. Ahmadian, M. Afsharchi, M. Meghdadi, A novel approach based on multi-view reliability measures to alleviate data sparsity in recommender systems, Multimedia Tools and Applications 78(13) (2019) 17763–17798. ISSN 1573-7721..

[2] M.Y.H. Al-Shamri. User profiling approaches for demographic recommender systems, Knowledge-Based Systems 100 (2016) 175–187. ISSN 0950-7051..

[3] M.M. Azadjalal, P. Moradi, A. Abdollahpouri, M. Jalili, A trust-aware recommendation method based on Pareto dominance and confidence concepts, Knowledge-Based Systems 116 (2017) 130–143. ISSN 0950-7051..

[4] Z. Batmaz, A. Yurekli, A. Bilge, C. Kaleli. A review on deep learning for recommender systems: challenges and remedies, Artificial Intelligence Review 52 (1) (2019) 1–37. ISSN 1573-7462..

[5] J. Bobadilla, S. Alonso, A. Hernando, Deep learning architecture for collaborative filtering recommender systems, Applied Sciences 10(7) (2020) 2441. ISSN 2076-3417..

[6] J. Bobadilla, A. Gutiérrez, F. Ortega, B. Zhu, Reliability quality measures for recommender systems, Information Sciences 442 (2018) 145–157.

[7] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowledge-Based Systems 46 (2013) 109–132. ISSN 0950-7051..

[8] J. Bobadilla, F. Serradilla, J. Bernal, A new collaborative filtering metric that improves the behavior of recommender systems, Knowledge-Based Systems 23 (6) (2010) 520–528.

[9] S. Deng, L. Huang, G. Xu, X. Wu, Z. Wu, On deep learning for trust-aware recommendations in social networks, IEEE Transactions on Neural Networks and Learning Systems 28(5) (2016) 1164–1177. ISSN 2162-2388..

[10] S. Fan, H. Yu, H. Huang, An improved collaborative filtering recommendation algorithm based on reliability, in: 2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), 2018, pp. 45–51.

[11] F.S. Gohari, F.S. Aliee, H. Haghighi, A new confidence-based recommendation approach: Combining trust and certainty, Information Sciences 422 (2018) 21–50. ISSN 0020-0255..

[12] P. Gravino, B. Monechi, V. Loreto, Towards novelty-driven recommender systems, Comptes Rendus Physique 20(4) (2019) 371–379. ISSN 1631-0705..

[13] G. Guo, J. Zhang, N. Yorke-Smith, A novel bayesian similarity measure for recommender systems, in: Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI), 2013, pp. 2619–2625.

[14] F.M. Harper, J.A. Konstan, The movielens datasets: History and context, ACM Transactions on Interactive Intelligent Systems 5 (4) (2015) 1–19.

[15] J.L. Herlocker, J.A. Konstan, L.G. Terveen, J.T. Riedl, Evaluating collaborative filtering recommender systems, ACM Transactions on Information Systems 22(1) (2004) 5–53. ISSN 1046-8188..

[16] A. Hernando, J. Bobadilla, F. Ortega, A non negative matrix factorization for collaborative filtering recommender systems based on a Bayesian probabilistic model, Knowledge-Based Systems 97 (2016) 188–202. ISSN 0950-7051..

[17] A. Hernando, J. Bobadilla, F. Ortega, J. Tejedor, Incorporating reliability measurements into the predictions of a recommender system, Information Sciences 218 (2013) 1–16. ISSN 0020-0255..

[18] V. Yu. Ignat'ev, D.V. Lemtyuzhnikova, D.I. Rul', I.L. Ryabov, Constructing a hybrid recommender system, Journal of Computer and Systems Sciences International 57(6) (2018) 921–926. ISSN 1555-6530..

[19] M. Kaminskas, D. Bridge, Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems, ACM Transactions on Interactive Intelligent Systems 7(1) (2016) 1–42. ISSN 2160-6455..

[20] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'08, Association for Computing Machinery, New York, NY, USA, 2008, pp. 426–434.

[21] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42(8) (2009) 30–37. ISSN 1558-0814..

[22] Y. Koren, J. Sill, OrdRec: An ordinal model for predicting personalized item rating distributions, Proceedings of the Fifth ACM Conference on Recommender Systems, RecSys'11, Association for Computing Machinery, New York, NY, USA, 2011, pp. 117–124.

[23] D.D. Lee, H.S. Seung, Algorithms for non-negative matrix factorization, in: T.K. Leen, T.G. Dietterich, V. Tresp, (Eds.), Advances in Neural Information Processing Systems, vol. 13, MIT Press, 2001, pp. 556–562..

[24] H. Li, Y. Liu, Y. Qian, N. Mamoulis, W. Tu, D.W. Cheung. HHMF: hidden hierarchical matrix factorization for recommender systems, Data Mining and Knowledge Discovery 33(6) (2019) 1548–1582. ISSN 1573-756X..

[25] Y. Liang, C. Huang, X. Bao, K. Xu, Sequential dynamic event recommendation in event-based social networks: An upper confidence bound approach, Information Sciences 542 (2021) 1–23. ISSN 0020-0255..

[26] Q. Madera, O. Castillo, M. García-Valdez, A. Mancilla, A method based on interactive evolutionary computation and fuzzy logic for increasing the effectiveness of advertising campaigns, Information Sciences 414 (2017) 175–186. ISSN 0020-0255..

[27] D. Margaris, C. Vassilakis, D. Spiliotopoulos, What makes a review a reliable rating in recommender systems? Information Processing & Management 57(6) (2020) 102304. ISSN 0306-4573..

[28] B. Marlin, Modeling user rating profiles for collaborative filtering, Proceedings of the 16th International Conference on Neural Information Processing Systems, NIPS'03, MIT Press, Cambridge, MA, USA, 2003, pp. 627–634.

[29] C. Martinez-Cruz, C. Porcel, J. Bernabé-Moreno, E. Herrera-Viedma, A model to represent users trust in recommender systems using ontologies and fuzzy linguistic modeling, Information Sciences 311 (2015) 102–118. ISSN 0020-0255..

[30] M.A. Mazurowski. Estimating confidence of individual rating predictions in collaborative filtering recommender systems, Expert Systems with Applications 40(10) (2013) 3847–3857. ISSN 0957-4174..

[31] R.M. Mesas, A. Bellogín, Exploiting recommendation confidence in decision-aware recommender systems, Journal of Intelligent Information Systems 54(1) (2020) 45–78. ISSN 1573-7675..

[32] A. Mnih R.R. Salakhutdinov, Probabilistic matrix factorization, in: Advances in Neural Information Processing Systems (2008) pp. 1257–1264..

[33] P. Moradi, S. Ahmadian, A reliability-based recommendation method to improve trust-aware recommender systems, Expert Systems with Applications 42(21) (2015) 7386–7398. ISSN 0957-4174..

[34] P. Moradi, S. Ahmadian, F. Akhlaghian, An effective trust-based recommendation method using a novel graph clustering algorithm, Physica A: Statistical Mechanics and its Applications 436 (2015) 462–481. ISSN 0378-4371..

[35] N. Nguyen-Thanh, D. Marinca, K. Khawam, D. Rohde, F. Vasile, E.S. Lohan, S. Martin, D. Quadri, Recommendation System-based Upper Confidence Bound for Online Advertising. ArXiv e-prints, 2019..

[36] F. Ortega, B. Zhu, J. Bobadilla, A. Hernando, CF4J: Collaborative filtering for Java, Knowledge-Based Systems 152 (2018) 94–99.

[37] M.G. Ozsoy, F. Polat, Trust based recommendation systems, Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM'13, Association for Computing Machinery, New York, NY, USA, 2013, pp. 1267–1274.

[38] X. Ramirez-Garcia, M. García-Valdez, Post-Filtering for a Restaurant Context-Aware Recommender System, Springer, Cham, 2014, pp. 695–707.

[39] X. Ramirez-Garcia, M. Garcia-Valdez, A Pre-filtering based context-aware recommender system using fuzzy rules, Springer, Cham, 2015, pp. 497–505. ISBN 978-3-319-17746-5..

[40] A. Rezvanian, B. Moradabadi, M. Ghavipour, M.M. Daliri Khomami, M.R. Meybodi, Social Recommender Systems, Springer International Publishing, Cham, 2019, pp. 281–313.

[41] G. Shani, A. Gunawardana, Evaluating Recommendation Systems, Springer, US, Boston, MA, 2011, pp. 257–297.

[42] S.S. Sohail, J. Siddiqui, R. Ali, Classifications of recommender systems: a review, Journal of Engineering Science and Technology Review 10(4) (2017) 132–153. ISSN 1791-2377..

[43] Z. Su, X. Zheng, J. Ai, L. Shang, Y. Shen, Link prediction in recommender systems with confidence measures, Chaos: An Interdisciplinary Journal of Nonlinear Science 29(8) (2019) 083133. ISSN 1054-1500..

[44] N.M. Villegas, C. Sánchez, J. Díaz-Cely, G. Tamura, Characterizing context-aware recommender systems: A systematic literature review, Knowledge-Based Systems 140 (2018) 173–200. ISSN 0950-7051..

[45] W. Wang, G. Zhang, J. Lu, Collaborative filtering with entropy-driven user similarity in recommender systems, International Journal of Intelligent Systems 30(8) (2015) 854–870. ISSN 0884-8173..

[46] X. Wu, X. Yuan, C. Duan, J. Wu, A novel collaborative filtering algorithm of machine learning by integrating restricted Boltzmann machine and trust information, Neural Computing and Applications 31(9) (2019) 4685–4692. ISSN 1433-3058..

[47] G. Xu, Z. Tang, C. Ma, Y. Liu, M. Daneshmand, A collaborative filtering recommendation algorithm based on user confidence and time context, Journal of Electrical and Computer Engineering 2019 (2019). ISSN 2090-0147..

[48] H. Zamani A. Shakery, A language model-based framework for multi-publisher content-based recommender systems, Information Retrieval Journal 21 (5) (2018) 369–409. ISSN 1573-7659..

[49] S. Zhang, L. Liu, Z. Chen, H. Zhong, Probabilistic matrix factorization with personalized differential privacy, Knowledge-Based Systems 183 (2019) 104864. ISSN 0950-7051..

[50] B. Zhu, F. Ortega, J. Bobadilla, A. Gutiérrez, Assigning reliability values to recommendations using matrix factorization, Journal of Computational Science 26 (2018) 165–177.