UKIRF: An Item Rejection Framework for Improving Negative Items Sampling in One-Class Collaborative Filtering

Antônio David Viniski, Jean Paul Barddal, and Alceu de Souza Britto Jr.

Graduate Program in Informatics (PPGIa)
Pontifícia Universidade Católica do Paraná (PUCPR)
Rua Imaculada Conceição, 1155
Curitiba, Paraná, Brazil
{adviniski,jean.barddal,alceu}@ppgia.pucpr.br

Abstract. Collaborative Filtering (CF) is one of the most successful techniques in recommender systems. Most CF scenarios depict positive-only implicit feedback, which means that negative feedback is unavailable. Therefore, One-Class Collaborative Filtering (OCCF) techniques have been tailored to tackling these scenarios. Nonetheless, several OCCF models still require negative observations during training, and thus, a popular approach is to consider randomly selected unknown relationships as negative. This work brings forward a novel and non-random approach for selecting negative items called Unknown Item Rejection Framework (UKIRF). More specifically, we instantiate UKIRF using similarity approaches, i.e., TF-IDF and Cosine, to reject items similar to those a user interacted with. We apply UKIRF to different OCCF models in different datasets and show that it improves the recall rates up to 24% when compared to random sampling.

Keywords: Collaborative Recommendations Systems · Implicit Feedback · Negative Sampling · Similarity metrics

1 Introduction

In recent years, recommendation systems have become widely used by companies like Amazon, Netflix, and Spotify are have been proven to be effective in recommending personalized items to users, boosting businesses, and facilitating decision-making processes [5, 17]. Collaborative filtering, which aims at predicting users' preferences towards items based on historical user feedback, is considered a central technique in recommender systems [13]. The users' feedback is expressed explicitly or implicitly to reflect the user's preferences for items. Explicit feedback is often represented by a numerical grade that describes different preference levels (such as a 1-5 scale) [6, 7]. Nonetheless, collecting explicit feedback from users is difficult, complex, costly, and even unfeasible, depending on the scenario. Therefore, in many real situations, the feedback implicitly expressed by users' behavior (like clicks, bookmarks, and purchases) is easier to

obtain and attracted increasing interest from researchers and practitioners [7]. Despite the easiness of data acquisition, implicit feedback scenarios have specific problems, such as negative feedback's unavailability. The absence of user negative feedback is referred to as One-Class Collaborative Filtering (OCCF), or positive-only feedback [6]. Despite the lack of negative feedback, algorithms tailored for OCCF require strategies to assume the unknown relations between users and items as negative [15]. There are two ways to incorporate unknown inputs into the model's training: (i) consider that all missing interactions between users and items are negatives, or (ii) select a sub-sample of these missing interactions as negative. Following the former strategy is computationally prohibitive, and thus, several proposals follow the latter approach as interactions are randomly sampled and assumed as negative. We argue that there is a need for methods that perform negative interaction sampling, considering the data characteristics instead of randomly.

In this work, we propose two methods for rejection, i.e., removing items from the missing entries per user by calculating similarity measures between items in the training interactions. These methods use positive interactions to find patterns in user behavior and reject unlabeled interactions that are likely to be relevant. The first method uses Cosine similarity [14, 4] in the interaction matrix, while the second combines Cosine similarity with a TF-IDF variant [1].

The paper is organized as follows. Section 2.1 introduces the one-class collaborative filtering task. Section 2.2 introduces related works that target improving the results obtained in OCCF scenarios. Section 3 presents the proposed framework for negative items rejection. Section 4 discusses the experiments performed alongside their analysis. Finally, Section 6 concludes this paper and states envisioned future works.

2 Related Work

This section introduces the research problem and reviews popular approaches designed for implicit positive-only scenarios that sample negative examples for training recommender models. In Section 2.1 we present the One-class collaborative filtering (OCCF) challenge and formalize the problem definition. Section 2.2 shows four popular recommender models that treat the unobserved interactions as negative ones.

2.1 One-Class Collaborative Filtering

One-class collaborative filtering (OCCF) predicts users' preferences given past positive feedback available in a dataset [5]. OCCF has characteristics that differentiate it from other tasks in recommendation systems. First, negative feedback lacks, as it is cumbersome to state with certainty which items a user dislikes. For instance, the lack of interaction is ambiguous as the user may indeed dislike an item or be unaware of it. Next, implicit datasets are highly sparse as few interactions are known, and most of the user-item relationship matrix corresponds to

missing data. Furthermore, OCCF scenarios are noisy, as an interaction between a user and an item does not mean that the user prefers it. There is no explicit feedback from the user w.r.t. one's satisfaction after such interaction. Finally, implicit ratings expressed numerically indicate confidence and do not represent users' preferences as with explicit ratings, yet, it describes the frequency of interaction, e.g., how many times a user listens to a song, how frequently a user purchases an item, and so forth.

Existing solutions for OCCF differ in how they handle unobserved data. Although the one-class collaborative filtering is less visited than the multi-class setting, some approaches have been proposed in the literature to deal with missing (unknown) items [11]. According to how the unlabeled data is used, existing methods to OCCF can be classified into two categories [11], i.e., whole-data based approaches [6, 15]; and sampling based approaches [8, 18, 3, 15]. Both approaches share challenges. When considering all the missing entries as negative, two constraints are relevant. First, as most of the training instances are negative, the class imbalance problem reduces the positive class's predictive ability. Second, one must deal with the possibility of introducing false negative examples. Besides, suppose we randomly sample unobserved interactions. Consequently, it is challenging to identify representative negative examples as all of the negative and missing positive interactions are mixed and cannot be distinguished [6]. Conversely, if the sampling method considers the dataset characteristics during negative sampling, we have a smaller probability of selecting false negatives.

Notation-wise, we denote $R_{m \times n}$ to be an interaction matrix, where m and n represent the number of users and items, respectively. Therefore, OCCF methods assign a score $\hat{r}_{ui}$ for each u-th and i-th user-item pair in $R_{m \times n}$, such that $u \in U$ and $i \in I$. The value of $r_{ui} \in \{0, 1\}$ denotes the positive or unobserved interaction of the u-th user on i-th, where r_{ui} is an element of R. Consequently, $\omega_u^+ = \{i \in I \mid r_{ui} = 1\}$ and $\omega_u^- = \{i \in I \mid r_{ui} = 0\}$ denote the sets of positive and unobserved items for the u-th user. Our goal is to perform negative item sampling from ω_u^- so that items that tend to be irrelevant for the u-th user given the characteristics of the positive items ω_u^+ are dropped.

2.2 OCCF Techniques

We can categorize the approaches designed for OCCF scenarios according to how they learn the relevance order. Most algorithms exploiting OCCF focus on homogeneous positive feedback with point-wise [3], pair-wise [8], and list-wise [9] preference assumptions. Point-wise approaches regard user ratings as categorical labels or numerical values and learn the relevance scores of missing data directly [11]. The pair-wise approaches try to capture the preference order between missing data, correctly identifying the positive/negative item in each pair [18, 11]. On the other hand, an individual training example is an entire list of items in a list-wise approach, rather than individual items or item pairs. However, due to their difficulty modeling the inter-list loss and inefficiency on large-scale datasets, list-wise CF approaches are not widely used compared to point-wise and pair-wise in ranking-oriented collaborative filtering [18]. Consequently, for further

experimentation, we select the pair-wise model Bayesian Personalized Ranking for Matrix Factorization (BPRMF) [8], and the point-wise models proposed in the Neural Collaborative Framework (NCF): Generalized Matrix Factorization (GMF), Multi-layer Perceptron recommender (MLP) and Neural Matrix Factorization (NeuMF) [3].

Bayesian Personalized Ranking for Matrix Factorization (BPRMF). BPRMF [8] is a popular familiar pair-wise method. Instead of only using the user-item interactions, for each interaction $(u; i)$, BPRMF selects a number of randomly selected items (j) to be used as negative items. BPR optimization decomposes triplets in the $(u; i; j)$ format using the difference of the predictions for the u-th user w.r.t. items i ($\hat{R}_{ui} = A_u \cdot B_i^T$) and j ($\hat{R}_{uj} = A_u \cdot B_j^T$), obtaining the instance prediction: $\hat{R}_{uij} = \hat{R}_{ui} - \hat{R}_{uj}$. The prediction error $e = |1 - \hat{R}_{uij}|$ is then used to update the u-th user (A_u), the i-th and j-th item (B_i and B_j) latent factors.

Neural Collaborative Framework (NCF). NCF is a deep neural network recommender framework composed of three recommender models: GMF, MLP, and NeuMF [3]. The NCF framework presents a probabilistic approach for learning the point-wise models that pay special attention to implicit data's binary property, i.e., training models using positive and negative examples. To endow the probabilistic explanation, NCF models constrain the output $\hat{r}_{ui}$ in the range of [0, 1] using a probabilistic function in the output layer. Regarding negative instances, the authors suggest uniformly sampling them from unobserved interactions in each iteration.

Generalized Matrix Factorization (GMF). To represent the latent features of users and items, GMF uses embedding layers. Each embedding layer is a fully connected layer that projects users' sparse representation and items in a dense vector. Thus, projecting the vector to the output layer we obtained the probability prediction of user u interact with the item i: $\hat{r}_{ui} = a_{out}\left(h^T\left(A_u \odot B_i\right)\right)$, where a_{out} and h denote the output activation function and edge weights of the output layer, respectively.

Multi-layer Perceptron (MLP). Instead of the element-wise dot product between latent factors like in GMF, MLP concatenates the user and item latent features. The concatenated vector is fully connected with hidden layers to model the collaborative filtering effect and learn the interaction between latent features A_u and B_u. Therefore, the item prediction is achieved by $\hat{r}_{ui} = \sigma\left(h^T \phi_L\left(z_{L-1}\right)\right)$, where σ, z_{L-1}, and $h^T \phi_L$ denote the activation function, the last hidden layer, and the edge weights of the output layer, respectively.

Neural Matrix Factorization (NeuMF). NeuMF combines GMF and MLP architectures. More specifically, it combines the linear and non-linear kernels from GMF and MLP. Internally, NeuMF trains GMF and MLP with random initializers until convergence. To provide more flexibility to the combined model, NeuMF allows GMF and MLP to learn separated embedding and connects them by concatenating their last hidden layer.

3 Unknown Items Rejection Framework (UKIRF)

This section introduces the Unknown Items Rejection Framework (UKIRF) to improve OCCF models' performance. Our goal with this framework is to provide a pre-processing step of a collaborative filtering recommendation process, thus not requiring modifications in the recommender models.

Algorithm 1 describes UKIRF. To generate the items rejection, UKIRF uses the interaction matrix R, the set of all users U and items I. As input, UKIRF requires the number of items (N_r) for rejection per user. Line 1 denotes the `rejection_method` function call, which identifies item-item relationships using the interaction matrix R and stores these relationship data into the S similarity matrix. Details on the rejection methods proposed are given in Section 3.1. Line 2 instantiates ω_f^- as an empty dictionary-like structure responsible for storing the negative options for all users. The loop in lines 3 to 10 iterates over all users $u \in U$, in which the positive items ω_u^+ (line 4) are recovered, followed by the unobserved ω_u^- items per user. According to the number of positive observations to the u-th user, the framework decides (line 6) whether to apply the unobserved item rejection strategy or not. If $|\omega_u^+| = 0$ holds, i.e., the u-th user has no positive interactions yet; the set of unobserved items (ω_u^-) is maintained. On the other hand, if $|\omega_u^+| > 0$, UKIRF uses the `apply_rejection` function (line 7) to return the list of items (ω_u^r) that are the most similar w.r.t. to the items for which the u-th user interacted with. The `apply_rejection` has as parameters the positive items of user u-th (ω_u^+), the rejection data returned by the `rejection_method` function, and an integer N_r that denotes the number of items that shall be rejected. Thus, the UKIRF removes from ω_u^- the items stored in ω_u^r (line 8). Next, regardless of the rejection strategy chosen, the resulting ω_u^- is stored in ω_f^- ($\omega_f^-[u] = \omega_u^-$), which corresponds to the negative options for all users (line 9).

Algorithm 1: Unknown Items Rejection Framework (UKIRF)

Data: Interaction Matrix R, set of all users U, set of all items I
Input: N_r: number of items to be rejected
Output: ω_f^-

1 $S \leftarrow$`rejection_method(`R`)` ▷ Generating the item-item similarity data
2 $\omega_f^- \leftarrow \{\}$ ▷ Instantiate the dictionary to stores the negative options
3 **foreach** $u \in U$ **do**
4 $\quad \omega_u^+ \leftarrow \{i \in I \mid r_{ui} = 1\}$ ▷ Recover all u-th user positive items
5 $\quad \omega_u^- \leftarrow \{i \in I \mid r_{ui} = 0\}$ ▷ Recover all u-th user unobserved items
6 $\quad$ **if** $|\omega_u^+| \neq 0$ **then**
7 $\quad\quad \Omega_u^r \leftarrow$ `apply_rejection(`ω_u^+, S, N_r`)` ▷ Return the N_r most similar items
8 $\quad\quad \omega_u^- \leftarrow \omega_u^- \setminus \Omega_u^r$ ▷ Remove N_r items from ω_u^-
9 $\quad \omega_f^-$[u] $\leftarrow \omega_u^-$
10 **end**
11 **return** ω_f^-

After rejecting unobserved items from the system's active users, the framework returns the dictionary containing all negative options per user in U (line 11).

3.1 Similarity-based Rejection Strategies

This section presents two rejection strategies that can be used in UKIRF: Cosine-based rejection and UF-IIF rejection. These reflect the `rejection_method` function in UKIRF, which receives as input the interaction matrix R.

Cosine Similarity approach: Among the existing similarity measures, the cosine function, which is defined as the inner product of two vectors divided by the product of their lengths [14], is the most popular and is widely used similarity measure. Its calculation is efficient, especially for sparse vectors, as only the non-zero dimensions are considered [4]. This characteristic is significant in the OCCF scenarios given the sparsity present in the interaction matrix R. Given two m-dimensional vectors $\vec{v}$ and $\vec{w}$, where m is the number of users, the Cosine similarity between them is calculated as follows:

$$\text{Cosine}(\vec{v}, \vec{w}) = \frac{\vec{v} \bullet \vec{w}}{\|\vec{v}\| \, \|\vec{w}\|} = \frac{\sum_{i=0}^{n} \vec{v_i} \times \vec{w_i}}{\sqrt{\sum_{i=0}^{n} \vec{v_i}^2} \sqrt{\sum_{i=0}^{n} \vec{w_i}^2}} \tag{1}$$

The Cosine approach applies the Cosine similarity function to all item-item ($\vec{v}$, $\vec{w}$) pairs. As a result, the `rejection_method` function returns the similarity matrix $S^{n \times n}$, where n is the number of items.

User frequency-inverse item frequency approach (UF-IIF). The User frequency-inverse item frequency is a specialization of 'Term frequency-inverse document frequency' (TF-IDF), one of the most commonly used term weighting schemes in the information retrieval systems [1]. TF-IDF is a metric that multiplies the two quantities TF and IDF. TF provides the frequency of each term in the document from the document collection. On the other hand, IDF can be interpreted as the amount of information representing each term's weight in the document collection. Less frequent terms have higher IDF values. In this work, we use TF-IDF to calculate the similarity between items. First, we assume that the user is a "term" (UF), and the item is a "document" (IIF). Thus, instead of calculating the similarity between documents, we obtain the similarity of the items. In this sense, the formulation of UF and IIF measures are as follows:

$$\text{uf} = \frac{f_{u,i}}{\sum_m f_{u,i}}, \ \text{iif} = log\left(\frac{n}{\text{if}_u}\right), \ \text{uf-iif} = \text{uf} \times \text{iif}$$

where $f_{u,i}$ is the number of times the u-th user interacted with the i-th item, $\sum_m f_{u,i}$ is the total number of users who interacted with the i-item, n is the number of items present in the dataset, and if_u is the number of items the u-th user interacted with. As the UF-IIF$^{m \times n}$ matrix stores the weights of each user-item pair, we use the *Cosine* function (Equation 1) to calculate the similarity matrix S between all items.

Both Cosine and UF-IIF techniques result in a similarity matrix S that stores the similarity between items in the recommender system. Therefore, both are

Algorithm 2: Get a list of N_r similar items to items in ω_u^+

Input: ω_u^+: positive items for an user u, S: similarity matrix between items, N_r: number of items to be rejected
Output: ω_u^r: a list of N_r most similar items to items in ω_u^+

1 **Function** `apply_rejection`(ω_u^+,S,N_r)**:**
2 $P \leftarrow D^{a \times b}$, such that $a \in \omega_u^+$ and $b \in I$, and $d_{a,b} = S_{a,b}$ ▷ get a partial similarity matrix with the weight vectors of items in ω_u^+
3 $V \leftarrow \sum_{k=0}^{a} P_{k,b}$ ▷ sum of similarities considering items in ω_u^+
4 Sort V in ascending order
5 $\omega_u^r \leftarrow V_k$, such that $(|V| - N_r \leq k \leq |V|)$
6 **return** ω_u^r
7 **End Function**

used in the `apply_rejection` function given in Algorithm 2. It receives as input all the positive observations of the u-th user (ω_u^+), the S matrix returned by the `rejection_method` function, and the number of items N_r the rejection approach must reject. The first step (line 2) selects from the similarity matrix (S) a partial matrix P that denotes the similarity values $s_{a,b} = S_{a,b}$, such that $a \in \omega_u^+$ and $b \in I$. Next, line 3 generates a similarity vector V, which denotes the sum of rows (a) weights for all items in the columns (b) of the partial similarity matrix $\sum_{k=0}^{a} \leftarrow P_{k,b}$. Line 4 sorts the similarity vector V in ascending order. Finally, line 5 stores in ω_u^r such items with the highest similarity values (ω_u^r represents the return of the function `apply_rejection`).

4 Experimental Setup

4.1 Datasets

We test our proposed framework in three supermarket datasets (`SMDI_original`, `SMDI_500E` and `SMDI_200UE`) and in the Movie Lens 100k dataset [2], such that the last has been converted so that only ratings above 3.5 were considered positive. Table 1 presents the datasets characteristics. In this experimental setting, we also propose approaches to define the number of rejected items (N_r). Regarding that most real-world datasets have repeated interactions, we use the third quartile ($Q3$) and the superior limit (SL) on the number of interactions and on the number of unique interactions per user to obtain N_r values. Thus, we have four alternatives to define the N_r for each dataset, considering the unique interactions (SLU and $Q3U$) and repeated interactions (SLT and $Q3T$). Table 1 also shows the number of rejections (N_r) in each scenario.

4.2 Baselines

We compare our proposed rejection strategies with the most often used uniform random sampling and test the generated sets of unobserved items in four recommender models: BPRMF, GMF, MLP, and NeuMF. For the recommender

Table 1. Overview of the datasets used during experimentation.

Datasets	Interactions	Users	Items	Sparsity	N_r			
					SLT	SLU	Q3T	Q3U
SMDI_original	737893	9531	7141	99.57%	212	108	92	48
SMDI_500E	448791	9480	6933	99.59%	204	103	89	46
SMDI_200UE	447391	9472	6924	99.59%	204	103	89	46
Movie Lens 100k	21201	928	1172	98.05%	64	64	30	30

algorithms, we tested the following hyper-parameter values: learning rate ∈ [0.001,0.005,0.0001,0.0005], regularization rate ∈ [0.01, 0.001, 0], latent factors ∈ [5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100]. After applying the rejection method, for each positive $\{u, i\}$ user-item interaction, we randomly sampled a number (from 1 to 10) of items from the previously filtered subset of unknown items to serve as negative ones.

4.3 Assessment

For each dataset, we made a temporal split, i.e., the first 50% of the time period were selected for training and the remaining 50% for testing. This temporal split is relevant as the dataset exhibits timestamps. From the training set, 20% of the instances were used for model validation. Thus, we use the validation loss to monitor the convergence of the models [10].

Following the protocol proposed in [12], we express the accuracy of the models using Recall@K, with $K \in \{1, 10\}$. The score, shown in Equation 2, measures the average (on all users) of the proportion of recommended items that appear among the top K of the ranked list [16], where $|T|$ is the test set size.

$$\text{Recall@K} = \frac{1}{|T|} \sum_{u,i \in T} (\text{hit@K}(u, i)) \tag{2}$$

For each instance $((u, i))$ in the test set, we select a candidate list of 100 unknown items to user u, and the known item i is appended to this candidate list. The candidate list is randomly sampled from the set of unknown items of user u or from the optimized set of unknown items when considering the unknown items' rejection approaches. According to the recommender models' scores, we ranked and sorted the candidates in descending order. For each instance (u, i), hit@$K(u, i) = 1$ is said to happen when i is ranked amidst the top K items, and hit@$K(u, i) = 0$, otherwise.

We replicate each experiment 5 times in this work, so the results show the average and standard deviation of recall values. The source code and datasets used during experimentation are available at `https://github.com/adviniski/UKIRF`.

5 Results and Analysis

Table 2 depicts the results obtained in the supermarket and Movie-Lens 100k datasets. We report the Recall@K (with $K = [1, 10]$) obtained by each model alongside the N_g and N_r values that achieved the best results, such that the former is the number of negative items per positive interaction, and the latter represents the strategy for rejecting items.

For BPRMF, which selects (positive, negative) pairs per positive instance, in the `SDMI_original` dataset (Table 2), the best results in comparison with the random sampling were obtained by the Cosine removal method, with ten negatives and SLT number of removals. We have an increase of 10.90% and 3.10% to the Recall@1 and Recall@10 values, respectively. Considering the Recall@10, the MLP model had superior performance, with a recall value of 73.4%, with ten negatives items per positive item in the training phase. The results with MLP increased 4.50% for Recall@10 in comparison to random sampling. The MLP and NeuMF models presented the best results for Recall@1 values in the original dataset, with an increase of 19.90% compared with the random sampling (from 32.9% to 52.8%), both with ten negative items. On the other hand, GMF presented inferior results when compared to MLP and NeuMF. GMF acquired results close to BPRMF, however, with lower recall values.

Despite being more straightforward than UF-IIF, the Cosine function obtained better results in all methods in the `SDMI_original` dataset, while the UF-IIF acquired close results to those obtained with random sampling. As UF-IIF also uses the Cosine similarity function, we expected close results to those obtained when using only the Cosine function in the interaction matrix. These results confirm the need for preprocessing approaches in the original supermarket dataset, showing that the datasets' noisy traits influence the models' results. Since UF-IIF generates weights to all user-item interactions before using the Consine function, we expected close or better results to those obtained by Cosine. The SLT approach rejects more items and provided the best results with the Cosine function.

In opposition to the previous dataset, in `SDMI_500E`, in which users with more than 500 interactions are removed, the model GMF outperformed BPRMF. However, the increase in recall values obtained by the Cosine removal compared to Random sampling for both methods was greater than those observed above. Here the Recall@1 and Recall@10 increased 16.70% and 3.80% for GMF and 14.30% and 2.10% for BPRMF. Besides, the MLP model presented better results for both recall measures. In this dataset, the UF-IIF rejection approach obtained close results to those presented by the Cosine method. Considering Recall@1, UF-IIF outperformed Cosine with recall values of 54.4% and 53.8%, respectively. Comparing the MLP with random sampling, the UF-IIF negative rejection approach provides an increase of 24.1% in the Recall@1 values (from 30.3% to 54.4%). Considering the number of negative items selected in the training phase, for UF-IIF, the best results were obtained with ten negative items in Recall@1, while for Cosine were eight negative items. On the other hand, for the Recall@10, the best results were found with 7 and 8 for Cosine and 6 and 5 for

Table 2. Recall@K (%) values obtained by the recommender models with the different sampling strategies in the all tested datasets. N_g represents the number of negative items that yielded the best results, and N_r denotes the number of rejected items considering the approaches (SLT, SLU, Q3T, Q3U).

Model	Recall@K	Random		Cosine			UF-IIF		
		Recall (%)	N_g	Recall (%)	N_g	N_r	Recall (%)	N_g	N_r
Dataset: SMDI_original									
BPRMF	1	24.4 ± 0.0027	10	**35.3 ± 0.0020**	10	SLT	27.0 ± 0.0034	9	SLT
	10	48.6 ± 0.0055	10	**51.7 ± 0.0027**	10	SLT	48.6 ± 0.0030	10	SLU
GMF	1	22.5 ± 0.0030	3	**36.4 ± 0.0057**	1	SLT	25.9 ± 0.0034	2	SLT
	10	46.5 ± 0.0028	3	**50.4 ± 0.0060**	2	SLT	47.4 ± 0.0056	2	SLT
MLP	1	32.9 ± 0.0023	3	**52.8 ± 0.0004**	10	SLT	35.7 ± 0.0027	7	SLT
	10	68.9 ± 0.0021	10	**73.4 ± 0.0005**	10	SLT	69.0 ± 0.0029	10	Q3T
NeuMF	1	32.9 ± 0.0035	2	**52.8 ± 0.0004**	10	SLT	35.5 ± 0.0024	2	SLU
	10	68.6 ± 0.0037	10	**72.6 ± 0.0013**	4	SLT	68.7 ± 0.0048	9	SLU
Dataset: SMDI_500E									
BPRMF	1	21.5 ± 0.0038	10	35.8 ± 0.0028	9	SLT	**35.9 ± 0.0034**	8	SLT
	10	45.7 ± 0.0040	10	**48.6 ± 0.0034**	9	SLT	48.5 ± 0.0036	8	SLT
GMF	1	20.8 ± 0.0030	2	**37.5 ± 0.0027**	10	SLT	37.2 ± 0.0022	8	SLT
	10	46.6 ± 0.0026	6	**50.4 ± 0.0034**	10	SLT	50.2 ± 0.0033	8	SLT
MLP	1	30.3 ± 0.0033	10	53.8 ± 0.0023	8	SLT	**54.4 ± 0.0024**	10	SLT
	10	67.8 ± 0.0011	9	**72.2 ± 0.0015**	7	SLT	**72.2 ± 0.0008**	6	SLT
NeuMF	1	30.0 ± 0.0021	10	52.6 ± 0.0008	7	SLT	**52.8 ± 0.0008**	10	SLT
	10	67.1 ± 0.0018	10	**71.9 ± 0.0017**	8	SLT	**71.9 ± 0.0026**	10	SLT
Dataset: SMDI_200UE									
BPRMF	1	21.6 ± 0.0022	10	**36.3 ± 0.0030**	10	SLT	36.0 ± 0.0018	9	SLT
	10	46.0 ± 0.0051	9	**49.4 ± 0.0036**	10	SLT	48.9 ± 0.0023	9	SLT
GMF	1	20.8 ± 0.0037	10	**37.1 ± 0.0020**	10	SLT	**37.1 ± 0.0021**	10	SLT
	10	46.7 ± 0.0046	6	**50.3 ± 0.0073**	9	SLT	49.9 ± 0.0061	7	SLT
MLP	1	30.3 ± 0.0019	9	53.9 ± 0.0011	9	SLT	**54.3 ± 0.0016**	10	SLT
	10	67.8 ± 0.0013	10	**72.3 ± 0.0011**	5	SLT	**72.3 ± 0.0007**	6	SLT
NeuMF	1	30.2 ± 0.0022	3	53.9 ± 0.0016	10	SLT	**54.2 ± 0.0022**	10	SLT
	10	67.8 ± 0.0011	9	**72.3 ± 0.0008**	9	SLT	**72.3 ± 0.0012**	5	SLT
Dataset: Movie-Lens 100k									
BPRMF	1	2.6 ± 0.0066	10	3.6 ± 0.0070	8	SLU	**3.7 ± 0.0084**	8	SLU
	10	14.7 ± 0.0110	10	**15.5 ± 0.0130**	8	Q3T	15.4 ± 0.0128	8	SLU
GMF	1	5.0 ± 0.0000	10	**16.8 ± 0.0000**	10	SLU	16.2 ± 0.0068	9	SLT
	10	26.3 ± 0.0104	8	**33.7 ± 0.0000**	10	SLU	**33.7 ± 0.0134**	9	SLT
MLP	1	7.8 ± 0.0061	8	27.4 ± 0.0058	8	SLU	**28.2 ± 0.0054**	9	SLU
	10	43.3 ± 0.0022	9	**55.5 ± 0.0031**	8	SLT	**55.5 ± 0.0038**	10	SLT
NeuMF	1	6.8 ± 0.0079	7	**20.7 ± 0.0965**	10	SLT	19.2 ± 0.1171	10	SLU
	10	39.7 ± 0.0157	9	53.7 ± 0.0051	10	SLT	**54.2 ± 0.0039**	10	SLT

UF-IIF. For both Cosine and UF-IIF, the SLT approach to select the number of rejected items was better than others in `SDMI_500E`.

The results of the two preprocessed datasets are very close. Still, if we had to choose one of the preprocessing approaches, we could see in the results Table

2 that for most of the recommender models, the SDMI_200UE dataset provided better results. In this dataset, we removed users with more than 200 unique (distinct items) interactions, which represented supermarket cashier operators.

In the three datasets analyzed, we found the most significant differences between the results of the rejection methods and the random sampling in the first top position (Recall@1). This means that the test dataset items were ranked in the top 1 position more effectively using the rejection methods than using random sampling without any rejection of unknown items. The MLP model presents the best increase in the Recall@1 value (24.1%) from the Random to the UF-IIF approach, considering the SDMI_500E dataset. For the SDMI_200UE dataset, both MLP and NeuMF increased Recall@1 values by 24% using the UF-IIF rejection method.

Finally, the results of the recommendation models obtained in the Movie Lens 100k dataset, also presented in Table 2, showed similar behavior to those obtained in supermarket datasets. We can see the effectiveness of the Cosine and UF-IIF methods compared to the Random approach, which showed an increase of 19.60% and 20.40% in Recall@1 values, respectively, for the MLP model.

6 Conclusion

This paper has shown how to increase the goodness of implicit recommendation models via the appropriate selection of negative items during the training phase. The motivation is that random sampling is insufficient and results in non-informative updates in the model's parameters. We propose a framework for rejecting potentially relevant items to users so that these are not assumed as negative. We used Cosine similarity to find similarity between items with that user interacted with either in the interaction matrix or in the user frequency-inverse item frequency (UF-IIF) matrix. We test our approaches in real-world datasets and provide the results obtained when it is coupled with four recommendation models (BPRMF, GMF, NeuMF, and MLP). Among the negative item rejection strategies, Cosine and UF-IIF obtained better results than random sampling, increasing Recall@1 values by up to 24%.

In future works, we plan to investigate other similarity metrics to quantify the relationship between items. Furthermore, we envision testing recommender models that are not built on matrix factorization to check how negative sampling affects their efficiency.

References

1. Aizawa, A.N.: An information-theoretic perspective of tf-idf measures. Inf. Process. Manag. **39**(1), 45–65 (2003)
2. Harper, F.M., Konstan, J.A.: The movielens datasets: History and context. TiiS **5**(4), 19:1–19:19 (2016)
3. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.: Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017. pp. 173–182. ACM (2017)

4. Li, B., Han, L.: Distance weighted cosine similarity measure for text classification. In: Intelligent Data Engineering and Automated Learning - IDEAL 2013 - 14th International Conference, IDEAL 2013, Hefei, China, October 20-23, 2013. Proceedings. Lecture Notes in Computer Science, vol. 8206, pp. 611–618. Springer (2013)
5. Li, G., Zhang, Z., Wang, L., Chen, Q., Pan, J.: One-class collaborative filtering based on rating prediction and ranking prediction. Knowl. Based Syst. **124**, 46–54 (2017)
6. Pan, R., Zhou, Y., Cao, B., Liu, N.N., Lukose, R.M., Scholz, M., Yang, Q.: One-class collaborative filtering. In: Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy. pp. 502–511. IEEE Computer Society (2008)
7. Pan, W., Liu, M., Ming, Z.: Transfer learning for heterogeneous one-class collaborative filtering. IEEE Intell. Syst. **31**(4), 43–49 (2016)
8. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: bayesian personalized ranking from implicit feedback. CoRR **abs/1205.2618** (2012)
9. Shi, Y., Larson, M.A., Hanjalic, A.: List-wise learning to rank with matrix factorization for collaborative filtering. In: Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010. pp. 269–272. ACM (2010)
10. Sidana, S., Laclau, C., Amini, M., Vandelle, G., Bois-Crettez, A.: KASANDR: A large-scale dataset with implicit feedback for recommendation. In: Proceedings of the 40th Intl. ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017. pp. 1245–1248. ACM (2017)
11. Song, B., Yang, X., Cao, Y., Xu, C.: Neural collaborative ranking. In: Proceedings of the 27th ACM Intl Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018. pp. 1353–1362. ACM (2018)
12. Vinagre, J., Jorge, A.M., Gama, J.: Fast incremental matrix factorization for recommendation with positive-only feedback. In: User Modeling, Adaptation, and Personalization - 22nd Intl. Conference, UMAP 2014, Aalborg, Denmark, July 7-11, 2014. Proceedings. vol. 8538, pp. 459–470. Springer (2014)
13. Volkovs, M., Yu, G.W.: Effective latent models for binary feedback in recommender systems. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13, 2015. pp. 313–322. ACM (2015)
14. Ye, J.: Cosine similarity measures for intuitionistic fuzzy sets and their applications. Math. Comput. Model. **53**(1-2), 91–97 (2011)
15. Yu, H., Bilenko, M., Lin, C.: Selection of negative samples for one-class matrix factorization. In: Proceedings of the 2017 SIAM International Conference on Data Mining, Houston, Texas, USA, April 27-29, 2017. pp. 363–371. SIAM (2017)
16. Yuan, Q., Chen, L., Zhao, S.: Factorization vs. regularization: fusing heterogeneous social relationships in top-n recommendation. In: Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23-27, 2011. pp. 245–252. ACM (2011)
17. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: A survey and new perspectives. ACM Comput. Surv. **52**(1), 5:1–5:38 (2019)
18. Zhang, W., Chen, T., Wang, J., Yu, Y.: Optimizing top-n collaborative filtering via dynamic negative item sampling. In: The 36th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '13, Dublin, Ireland - July 28 - August 01, 2013. pp. 785–788. ACM (2013)