Data Mining and Knowledge Discovery (2020) 34:1560–1588
https://doi.org/10.1007/s10618-020-00708-6

Simple and effective neural-free soft-cluster embeddings for item cold-start recommendations

Shameem A. Puthiya Parambath[1] · **Sanjay Chawla**[1]

Received: 7 November 2019 / Accepted: 25 July 2020 / Published online: 3 August 2020

Abstract

Recommender systems are widely used in online platforms for easy exploration of personalized content. The best available recommendation algorithms are based on using the observed preference information among collaborating entities. A significant challenge in recommender system continues to be item cold-start recommendation: how to effectively recommend items with no observed or past preference information. Here we propose a two-stage algorithm based on soft clustering to provide an efficient solution to this problem. The crux of our approach lies in representing the items as soft-cluster embeddings in the space spanned by the side-information associated with the items. Though many item embedding approaches have been proposed for item cold-start recommendations in the past—and simple as they might appear—to the best of our knowledge, the approach based on soft-cluster embeddings has not been proposed in the research literature. Our experimental results on four benchmark datasets conclusively demonstrate that the proposed algorithm makes accurate recommendations in item cold-start settings compared to the state-of-the-art algorithms according to commonly used ranking metrics like Normalized Discounted Cumulative Gain (NDCG) and Mean Average Precision (MAP). The performance of our proposed algorithm on the MovieLens 20M dataset clearly demonstrates the scalability aspect of our algorithm compared to other popular algorithms. We also propose the metric *Cold Items Precision* (CIP) to quantify the ability of a system to recommend cold-start items. CIP can be used in conjunction with relevance ranking metrics like NDCG and MAP to measure the effectiveness of the cold-start recommendation algorithm.

Responsible editor: Ira Assent, Carlotta Domeniconi, Aristides Gionis, Eyke Hüllermeier.

✉ Shameem A. Puthiya Parambath
spparambath@hbku.edu.qa

Sanjay Chawla
schawla@hbku.edu.qa

[1] Qatar Computing Research Institute, HBKU Research Complex, Doha, Qatar

Keywords Recommender systems · Item recommendation · Item cold-start problem · Soft-cluster embeddings

1 Introduction

Personalized recommender systems assist users in exploring large collections of items efficiently to deal with the problem of information overload by filtering the items into small selections tailored to an individual's personal preference. This is achieved by inferring the users' intrinsic preferences for different items. In typical use cases, items can be simple tweet messages, food recipes, electronic gadgets or vehicles. Two popular approaches for recommendation are: (i) content based and (ii) collaborative. Content-based systems recommend items which are similar in content to the ones a user favoured in the past whereas collaborative systems recommend items that users with similar tastes favoured in the past. It is well established in research and practice that collaborative systems tend to outperform content based systems (Adomavicius and Tuzhilin 2005; Saveski and Mantrach 2014). Collaborative algorithms make use of past or observed preference information among collaborating entities (users and items) to recommend top-N items for a user. The preference information among collaborating entities are often represented using a user-item preference or rating matrix where each entry of the matrix stands for a rating score given by a user for an item. The rating information can be due to either explicit or implicit feedback; in explicit feedback settings, users assign a preference score that quantifies the relative degree of favouritism of a user for the item and is often represented as an ordinal number. In implicit feedback settings, preference information is inferred from the implicit user-item interaction like watching a show can be inferred as a positive feedback whereas skipping it is inferred as a negative feedback (Hu et al. 2008). We limit our exposition to the explicit feedback settings as it is very popular with many real world problems. For example, in the Netflix Challenge, movies were rated in the ordinal scale 1, 2, 3, 4, 5, one denoting the least favoured item and five denoting the most favoured.

The collaborative methods for recommendation can be loosely classified under three schemes: (i) user based, (ii) item based and (iii) latent factor based. Setting aside the relative merits and demerits of these three schemes, it is argued that all these approaches work better than content-based systems (Adomavicius and Tuzhilin 2005; Saveski and Mantrach 2014), but it suffers from the cold-start problem. The user and item collaborative algorithms work only in warm-start settings i.e., when past rating data for users and items are available. On the other hand, latent factor models rely on matrix factorization schemes. In the case of matrix factorization based algorithms, user and item features can be extracted only for those users and items for which some rating values are observed. A major challenge in collaborative recommendation is: how to provide top-N recommendations when rating data is completely missing for an item or a user. In the recommendation literature this scenario is termed as cold-start recommendation. The cold-start problem can be either due to a cold-start item i.e. an item is not yet rated by any user (item cold-start problem) or due to a cold-start user i.e. a user did not rate any item (user cold-start problem) or both. In this work, we concentrate on the item cold-start problem, which naturally arises when a

	i_1	i_2	$\cdots$	i_k	i_{k+1}	$\cdots$	i_m	i_{m+1}	$\cdots$	i_{n-1}	i_n
u_1	1	1	$\cdots$	0	0	$\cdots$	1	1	$\cdots$	0	-
u_2	1	1	$\cdots$	0	0	$\cdots$	1	1	$\cdots$	-	-

(a) user-item rating matrix

	i_1	i_2	$\cdots$	i_k	i_{k+1}	$\cdots$	i_m	i_{m+1}	$\cdots$	i_{n-1}	i_n
f_1	0	0	$\cdots$	1	1	$\cdots$	1	1	$\cdots$	1	1
f_2	1	1	$\cdots$	0	0	$\cdots$	1	1	$\cdots$	1	1

(b) item side-information matrix

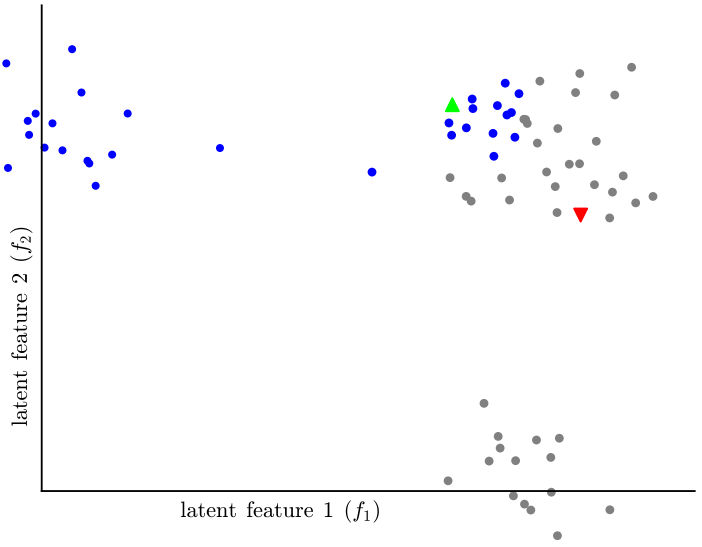

Fig. 1 Irrelevant (grey) and relevant (blue) items for two users with exact rating profiles (see the user-item rating matrix (a). In matrix (a) 1 indicates item is preferred, 0 indicates item is disfavored and—indicates rating is not observed) form three clusters. Our idea is to rank the cold item ▲ (item i_n in tables) over the irrelevant warm item ▼ (item i_{n-1} in tables), rated only by the first user, for the second user, as the cold item is placed nearer to the relevant items. In matrix (b), 1/0 indicates that the item is supported/unsupported by the corresponding latent factor. The scores obtained by our algorithm for the cold and warm items are 2.35 and 1.48 respectively (Color figure online)

new item is added to the items catalogue. Formally, the problem can be defined as: Given a user-item rating matrix containing cold-start items, make a personalized top-N recommendation of items which contains relevant (favoured) cold-start items also, if any.

We propose an efficient two-stage algorithm[1] for item cold-start recommendation within the collaborative matrix factorization framework. Our algorithm starts by extracting vector embeddings for the cold and warm items assuming that the items can be soft-clustered in the space spanned by the enriched side-information. In the second step, the extracted soft-cluster embeddings of the warm items are used to estimate latent user embeddings by approximating a low rank factor model on the observed rating values. Our idea for cold-start recommendation is based on the assumption that an item lying closer to the relevant items cluster is favoured compared to an observed irrelevant item. The intuition behind our algorithm is graphically depicted in Fig. 1.

Consider two users u_1 and u_2 and a set of n items. The user u_1 rated $(n-1)$ items $\{i_1 \ldots i_{n-1}\}$ whereas user u_2 rated $n-2$ items $\{i_1 \ldots i_{n-2}\}$. The users have shown identical tastes i.e. the rating profiles for the $(n-2)$ items $\{i_1 \ldots i_{n-2}\}$ for

[1] Source code for the algorithm: https://git.io/JJYmO.

both users are exactly the same. Both the users favoured the item sets $\{i_1 \dots i_{k-1}\}$ and $\{i_m \dots i_{n-2}\}$, but disfavoured the items set $\{i_k \dots i_{m-1}\}$. The item i_n is unobserved (represented using) for both the users whereas the item i_{n-1} is disfavoured by u_1 but unobserved for u_2. This information is represented in Table: (a) of Fig. 1. Table: (a) corresponds exactly to the rating matrix associated with the data. The items $\{i_1 \dots i_n\}$ are associated with at most two latent factors f_1 and f_2. In the simplest use case, if we assume that the items $i_1 \dots i_n$ are movies and each movie can be associated with at most two genres, f_1 and f_2 represent the genres associated with the movies. In such cases, each movie can be represented as a real vector of genre association values. For example, if the movie titled **a** has genre association values 0.4 and 0.8, it can be represented as the vector $\begin{bmatrix} 0.4 \\ 0.8 \end{bmatrix}$.

In the example given in Fig. 1, the items $\{i_1 \dots i_{k-1}\}$ and $\{i_k \dots i_{m-1}\}$ are supported[2] by the latent factors f_2 and f_1 respectively. The items $\{i_m \dots i_n\}$ are supported by both latent factors f_1 and f_2. This information is shown in the Table: (b) of Fig. 1. Table: (b) corresponds exactly to the side-information matrix associated with the data. It turns out that the preferences of u_1 and u_2 are fully defined by the latent factors f_1 and f_2 in the sense that both users disfavor items with support only by f_1 and prefer items with support only by f_2. The items with support by both f_1 and f_2 have mixed preferences. This is represented using the scatter plot in Fig. 1.[3] Items with support only by f_1 i.e. the items for which $f_1 = 1$ and $f_2 = 0$ in the item side-information matrix are plotted around the line labelled f_1 and the items with support only by f_2 i.e. the items for which $f_1 = 0$ and $f_2 = 1$ in the item side-information matrix are plotted around the line labelled f_2. The cluster of points at the top right corner represents the items with support by both f_1 and f_2. Items relevant to both users i.e. items with high preference scores are colored in blue and irrelevant items are colored in gray.

Given the rating and side-information matrix, the task here is to recommend an unrated item to the user u_2. The possible candidate items are i_{n-1} and i_n. Since the item i_n is not observed by either u_1 or u_2, it is a cold item whereas i_{n-1} is an irrelevant item for u_1 but an unobserved warm item for u_2. In Fig. 1, ▲ and ▼ represent the cold and warm unobserved items respectively. Our idea is to rank the cold item (▲), placed nearer to the cluster of relevant items (blue point cloud), ahead of the relatively far placed irrelevant warm item (▲) when recommending for user u_2. To do so, we first extract the cluster association values from the enriched side-information matrix and use it to fit user feature vectors using rating information. Since the cold item has greater association towards the latent factor f_2 or smaller association towards the latent factor f_1, the soft-cluster embeddings for the cold item will reflect it. The rating scores obtained by our algorithm for the cold and warm items are 2.35 and 1.48 respectively.

We also observe that the majority of the item cold-start recommendation algorithms are evaluated using commonly used ranking metrics like NDCG or MAP only. But in practical settings, recommended items are a mix of both warm and cold items. For effective evaluation, it is important to quantify the ability of an algorithm to recommend

[2] Support of a vector $\mathbf{x}$ is the set of all j such that $\mathbf{x}_j > 0$.

[3] To get a sense of the point cloud and differentiate between individual points, the points are plotted with a small random noise added to it.

cold items. We propose the metric *Cold Items Precision* (CIP) to measure the ability of an algorithm to recommend cold items. CIP can be used in conjunction with NDCG and MAP to measure the effectiveness of the cold-start recommendation algorithm. In addition, the two-step or split optimization strategy we use in our settings might be of independent theoretical interest, particularly when applied to cold-start scenarios.

We advise the readers to keep in mind that similar to item cold-start recommendation algorithms proposed by Vartak et al. (2017) and Saveski and Mantrach (2014), our proposed algorithm can handle item or user cold-start problem only; it cannot handle both the item and user cold-start problems. In addition, we also need the side-information matrix to contain positive correlations between different side-information categories to obtain significant performance improvement over other algorithms (see Sect. 4). We reckon that this might be the case with other content + collaborative based algorithms also. The remainder of this paper is organized as follows: after a brief overview of the state-of-the-art personalized item cold-start recommendation algorithms in Sect. 2, we describe our framework and algorithm in Sect. 3. At the end of Sect. 3, we propose an extension of the algorithm to handle user cold-start problems. In Sect. 4, we report the results of our experimental study on three benchmark datasets against strong non-deep learning based baselines. Though recent studies (Ludewig and Jannach 2018; Lin 2019; Dacrema et al. 2019; Ludewig et al. 2019) showed that deep learning based algorithms perform relatively poorly compared to non-deep learning based baselines in common information retrieval tasks, there is a growing interest within the community to adapt deep learning based models. In this regard, we compare our algorithm against state-of-the-art deep graph embedding models for cold-start recommendations. We conclude the paper in Sect. 5 after describing the results of a qualitative study.

2 Related work

Collaborative item cold-start recommendation algorithms can be broadly classified into two categories: (i) based on the idea of enriching matrix factorization based collaborative algorithms with item side-information or metadata and (ii) based on eliciting user feedback for cold-start items by recruiting a set of users to get initial ratings. Up to some extent, content-based algorithms are unsusceptible to the cold-start problem, and a natural strategy to deal with the problem is to adapt collaborative algorithms to use the content information. Matrix factorization based collaborative filtering schemes consider preference information of both users and items to extract latent user and item features from the observed rating matrix. These latent user and item feature embeddings are used to predict the top-N recommendations. To handle the cold-start recommendation problem, one can devise methods to incorporate the side-information data in the matrix factorization scheme.

A large class of cold-start recommendation algorithms are based on the collective matrix factorization idea proposed by Singh and Gordon (2008). Collective matrix factorization based algorithms assume the existence of a shared subspace between different modalities of the data. The shared item subspace can be approximated using the available side-information and rating data. Typical collective matrix factorization based algorithms optimize a non-convex joint objective function, defined in terms

of all the available data modalities, using cyclic block coordinate descent otherwise known as alternating minimization. Saveski and Mantrach (2014) proposed an algorithm based on the collective matrix factorization technique that uses the item metadata and the rating data, and exploits the local geometric structure of the metadata space. The proposed algorithm, called *locally collective matrix factorization*, decomposes the side-information and the collaborative matrices in a common low-dimensional space while preserving the local geometrical structure of the data. This is done by adding a manifold regularizer (Belkin et al. 2006) to the collective matrix factorization objective function. Very recently, Gouvert et al. (2018) also used the collective matrix factorization ideas to solve the cold-start recommendation in the music recommendation setting. The proposed algorithm learns the feature vectors for the warm and cold songs by jointly optimizing a loss function defined in terms of the listening count data of the songs and the associated tags information. Typical to the collective matrix factorization models, the authors assume the existence of a shared subspace between song listen count and tags.

Factorization machines (Rendle 2010, 2012) based techniques are also very popular for cold-start recommendations. Factorization machines based algorithms also make use of the item side-information to predict the relevance scores for cold items. In specific application settings with abundant item metadata, such as question-answering, many cold-start recommendation algorithms based on the idea of using factorization machines are proposed in the past (Sun et al. 2018; Piazza et al. 2017). In factorization machines, each user-item interaction is represented as a unique feature-label pair where the feature is constructed by using the user and item side-information data. The model parameters are learned using standard supervised learning techniques. The higher order correlations between different available side-information can also be part of the feature vector. Factorization machine based models are particularly suitable in sparse settings when the categorical side-information data is represented using one-hot encoding. Since factorization machine objective function does not make use of the user-user correlations explicitly, algorithms based on factorization machines might lose some collaborative information between different users. Though this problem can be alleviated by encoding observed user-user correlation as part of the feature vector, it might result in an exponential increase in the dimensionality of the feature space.

Gantner et al. (2010) proposed a two-stage algorithm, similar to ours, by estimating attribute-to-feature mappings from the rating matrix using supervised learning techniques. In the first stage, the proposed method learns user and item features by fitting a low rank latent factor model. The extracted user features are used with item metadata to learn item weight vectors for different metadata categories for both cold and warm items in the second stage. In addition to higher computational costs, the above algorithm may result in severe overfitting as the observed rating values are shared in the learning phases of both stages of the algorithm. The proposed algorithm uses the observed rating matrix to learn the user and item features in the first phase and the same observed rating matrix is used to learn the weight matrix in the second phase of the algorithm. Our proposed method learns item embeddings using cluster assumption in an unsupervised manner without making use of the observed rating values with lesser computational cost.

Park and Chu (2009) extended the pairwise preference ranking model to accommodate cold-start items. In the proposed method, the observed rating matrix is assumed to be a weighted cross-product between user and item metadata vectors. The rating scores are estimated by fitting a pairwise loss function in the regression framework. Zhang et al. (2014) extended the collaborative latent factor model to include bias terms for the metadata categories associated with the items. The final recommendation is produced by co-training two regressors learned from the observed user-item rating values and corresponding metadata data.

Barjasteh et al. (2015) proposed an algorithm based on decoupling the rating estimation and item feature vector estimation into two separate steps. The algorithm extracts a representative item subspace from the item similarity matrix after estimating the entries of a fully recoverable submatrix of the original observed rating matrix using standard matrix completion techniques. The full rating matrix is estimated using the extracted representative subspace and the information contained in the completed submatrix. Chou et al. (2016) proposed a cold-start next-song recommendation algorithm in sequential data settings. The proposed algorithm predicts the next-song using tensor factorization that exploits content features extracted from music audio. Recently, Sedhain et al. (2017) proposed a Low-Rank Linear model for user cold-start recommendation settings. In the proposed algorithm, the observed rating values are used to estimate a low-dimensional weight matrix from a subset of parametrized low-rank matrices. This is a generalization of the pure content based cold-start recommendation algorithm and the proposed algorithm is equivalent to the pure content based algorithm when the weight matrix is replaced with the one-hot encoded side-information matrix of the items. This approach can also be considered as a special case of the factorization machines (Rendle 2010) with only a single order of interactions used for feature engineering.

Clustering based approaches are also used for collaborative filtering in the past (Ungar and Foster 1998; Salah et al. 2016; Verstrepen et al. 2017). The majority of the clustering algorithms work by co-clustering the users and items based on the rating values. Recently, Vlachos et al. (2019) proposed a co-clustering based algorithm for cold-start recommendation algorithm in the presence of positive only ratings. As in the case of our algorithm, the proposed algorithm makes use of any side-information data, but assumes that all the items are either preferred or the preference status is not known. The side-information data is incorporated into the co-clustering objective to form a joint optimization objective, and the model parameters are estimated using alternating minimization techniques typical to the collective matrix factorization based approaches.

Recently, many item representation learning models have been proposed. Kula (2015) proposed learning item embeddings by aggregating the embeddings for each category of the side-information. These embeddings are extracted by fitting a low rank matrix model on the rating data. With the advent of deep representation learning methods, many deep-neural network based architectures have been proposed to solve the item cold-start problem. Vasile et al. (2016) proposed learning item embeddings from the text data associated with the product co-purchase details and item side-information using deep neural networks. Wei et al. (2016) proposed a 2-stage approach where the item embeddings for both cold and warm items are learned using deep neural networks,

from the side-information in the first stage and are used in the second stage where SVD based matrix factorization techniques are used to estimate the unobserved rating values. This work considers recommendation as a rating prediction task whereas we pose recommendation as a top-N ranking problem. Similarly, Vartak et al. (2017) proposed a meta-learning item cold-start recommendation algorithm using deep neural networks. The proposed method works in implicit feedback settings and recommendation is considered as a binary classification task. Our proposed method differs from the above methods as our embeddings are soft-cluster embeddings which are not constrained to work only with the textual description of the items. It is generally believed that the subpar performance of the deep learning models for information retrieval tasks as opposed to the language processing and vision tasks is due to the sparsity in the input features (Sun et al. 2018; McMahan et al. 2013). Moreover, in light of recent studies on the efficacy of deep learning methods for recommendation tasks, one should be sceptical about the use of these methods (Dacrema et al. 2019; Ludewig et al. 2019). In spite of that, deep architecture models are the backbone of many large scale industrial recommender systems like YouTube (Covington et al. 2016). Unlike the settings we study in this paper, the majority of the deep learning models for cold-start recommendation do not consider the extreme cold-start scenario. Recent advances in deep learning models for cold-start recommendation make use of the graph embedding techniques and learn user and item embeddings by injecting higher order connectivity relations between users and items using the Laplacian of the user-item implicit interaction matrix (Zheng et al. 2018; Wang et al. 2019b).

User feedback elicitation based cold-start recommendation algorithms are mostly used in online learning settings and work by eliciting new items or new user preferences by actively querying preference information. A typical workflow in such an approach can be described as: whenever a new item is introduced, the system initiates many trials and at each trial the new item is presented to a seed user for her opinion, thus gradually building a rating profile for the item. Zhou et al. (2011) proposed functional matrix factorization within the active learning framework, employing a decision tree model for selecting the seed users for each new item and the decision tree model was combined with the low-rank matrix factorization to fit a prediction model. Feature vectors for the cold-start items were adaptively built by the seed users' responses by modeling the cold-start item feature vector as a function of user response. The decision tree outputs were mapped to the item feature space and these item features were used in the subsequent low-rank approximation. The proposed method alternatively learns the best mapping function and item features. Aharon et al. (2015) proposed a smart exploration strategy to identify a set of users for rating the cold-start items in online settings. Similarly, Anava et al. (2015) proposed an algorithm to recruit a set of users to rate the cold-start items with constraints on the number of users to recruit.

3 Cold-start item recommendations

We are given a set $\mathcal{I}$ of n items and a set $\mathcal{U}$ of m users. Each user is assumed to have an inherent personal preference over the set of items $\mathcal{I}$ i.e. each item $i \in \mathcal{I}$ might be favoured differently by different users. Users' personal preferences for the items are

represented using a preference score in the form of ordinal rating values, typically in the range $\{1 \cdots 5\}$ with higher values indicating favored items and lower values indicating unfavoured items. We also assume that a user may have the same preference score for different items and the preference relation is rational. The rating values are tabulated as a $m \times n$ user-item rating matrix $\mathbf{R}$ where each entry $\mathbf{R}_{ij}$ represents the rating values by the user i for the item j. The unobserved rating values are denoted using zeroes i.e. $\mathbf{R}_{ij} = 0$ if user i did not rate item j.

In addition to the above standard collaborative settings assumption, we also assume that some side-information related to the items is available. We refer to any attribute or metadata associated with items other than ratings as side-information. We represent the available side-information associated with the items set $\mathcal{I}$ using a $n \times p$ characteristic matrix called item side-information matrix $\mathbf{A}$ where p is the number of available side-information categories. For example, if items are movies, then attributes such as genre, cast, director etc. are referred to as side-information, which can be represented as a one-hot encoded characteristic matrix. The subspace spanned by the rows of the characteristic matrix forms side-information space. In the standard item cold-start scenario, we assume that the item set $\mathcal{I}$ is a union of two disjoint sets $\mathcal{I}_w$ and $\mathcal{I}_c$ i.e. $\mathcal{I} = \mathcal{I}_w \cup \mathcal{I}_c$ and $\mathcal{I}_w \cap \mathcal{I}_c = \varnothing$, where $\mathcal{I}_w$ is the set of warm items and $\mathcal{I}_c$ is the set of cold items i.e. items for which no rating values are observed ($\mathbf{R}_{ij} = 0 \; \forall j \in \{1 \ldots n\}$). In the item cold-start recommendation problem, one is interested in recommending potentially relevant unobserved items to users such that recommendation contains relevant cold-start items from $\mathcal{I}_c$ also, if any.

3.1 Soft cluster membership item embeddings

The crux of our approach lies in extracting rich vector embeddings for items, both warm and cold, from the available side-information data. To extract the vector embeddings, we assume that the items form k weak clusters in the side-information space and each item is associated with one or more of these k clusters. For example, if the items are movies and the side-information space is made up of different genres, each movie will be associated with multiple genres or some latent factors extracted from genres with different degrees of association. The ith entry of an item vector embedding indicates the degree to which the corresponding item is associated with ith cluster.

Instead of working directly on the item side-information matrix, we wish to take into account possible dependencies between different categories within the side-information. To explain this idea, consider the movie recommendation example. A horror movie is expected to have stronger association with movies under the thriller or suspense genres than family movies. We expect such associations to be reflected in the soft-cluster embedding vector. For this purpose, we define the enriched side-information matrix or simply enriched matrix $\mathbf{X}$ as the product of the side-information matrix $\mathbf{A}$ with the co-occurrence frequency $\mathbf{A}^\top\mathbf{A}$. The side-information matrix $\mathbf{A}$ is assumed to be the one-hot encoded matrix.

$$\mathbf{X} = \mathbf{A}\mathbf{A}^\top\mathbf{A} \tag{1}$$

NMF based clustering Given a data matrix, k-means clustering can be used to hard cluster the data points. The k-means can be equivalently cast as a matrix factorization problem (Ding et al. 2005). When the data matrix contains only non-negative entries, it is natural to assume that the factor matrices are also non-negative. Thus, we can frame k-means clustering of non-negative data points as a Non-negative Matrix Factorization (NMF) problem and the factor matrices can be interpreted as the cluster membership matrix and matrix of centroids (Ding et al. 2010). Moreover, it is argued that NMF has clustering capabilities which are superior to k-means (Li and Ding 2006).

Formally, non-negative factorization of $\mathbf{X}$ of the form

$$\underset{\substack{\mathbf{G}^\top\mathbf{G}=\mathbf{N}_k \\ \mathbf{G}_{ij}\in\{0,1\},\mathbf{F}\geq 0}}{\operatorname{argmin}} \quad \|\mathbf{X} - \mathbf{G}\mathbf{F}\|^2 \tag{2}$$

where $\mathbf{G} \in \Re_+^{n\times k}$, $\mathbf{F} \in \Re_+^{k\times p}$ and $\mathbf{N}_k$ is the $k \times k$ diagonal integer matrix, is equivalent to k-means clustering (Li and Ding 2006). The diagonal elements of $\mathbf{N}_k$ indicate the number of data points in each cluster.

Here $\mathbf{G}$ and $\mathbf{F}$ can be interpreted as the cluster membership matrix and the feature representation of the cluster centroids respectively. The ith column of $\mathbf{F}$ corresponds to the centroid of the ith cluster. The constraints $\mathbf{G}_{ij} \in \{0, 1\}$ and $\mathbf{G}^\top\mathbf{G} = \mathbf{N}_k$ guarantee that each data point is associated with exactly one cluster. Thus the optimization problem in Eq. (2) results in hard clustering as in the case of k-means. Now, relaxing the constraints $\mathbf{G}_{ij} \in \{0, 1\}$ and $\mathbf{G}^\top\mathbf{G} = \mathbf{N}_k$, the optimization problem in Eq. (2) reduces to the soft-clustering as noted by Li and Ma (2004) and Ding et al. (2005). Here, $\mathbf{G}_{ij}$ indicates the degree to which the ith item is associated with jth cluster. Lee and Seung (2001) proposed a simple multiplicative update rule to find a local minima of the soft-clustering optimization problem.

$$\mathbf{F}_{ij} = \mathbf{F}_{ij}\frac{(\mathbf{G}^\top\mathbf{X})_{ij}}{(\mathbf{G}^\top\mathbf{G}\mathbf{F})_{ij}} \quad \mathbf{G}_{ij} = \mathbf{G}_{ij}\frac{(\mathbf{X}\mathbf{F}^\top)_{ij}}{(\mathbf{G}\mathbf{F}\mathbf{F}^\top)_{ij}}$$

Convex-NMF Based Clustering: Ding et al. (2010) showed that by constraining $\mathbf{F}$ to be the linear combinations of the rows of $\mathbf{X}$, a better estimate for the cluster centroids can be obtained. The authors proposed convex-NMF where the centroid matrix $\mathbf{F}$ is assumed to be a linear combination of the columns of mixed-sign matrix $\mathbf{X}$. The optimization problem for the convex-NMF takes the form

$$\underset{\mathbf{G}\geq 0,\mathbf{P}\geq 0}{\operatorname{argmin}} \ \|\mathbf{X} - \mathbf{G}\mathbf{P}\mathbf{X}\|^2 \tag{3}$$

Here, since the enriched matrix $\mathbf{X}$ is non-negative, our update rule is slightly different from the one proposed by Ding et al. (2010). The objective function given in Eq. (3)

is non-increasing under the update rules

$$\mathbf{G}_{ij} = \mathbf{G}_{ij} \frac{(\mathbf{X}\mathbf{X}^\top \mathbf{P}^\top)_{ij}}{(\mathbf{G}\mathbf{P}\mathbf{X}\mathbf{X}^\top \mathbf{P}^\top)_{ij}} \quad \mathbf{P}_{ij} = \mathbf{P}_{ij} \frac{(\mathbf{G}^\top \mathbf{X}\mathbf{X}^\top)_{ij}}{(\mathbf{G}^\top \mathbf{G}\mathbf{P}\mathbf{X}\mathbf{X}^\top)_{ij}}$$

The above update rule can be easily obtained using gradient descent with adaptive learning rate. For example, gradient update rule for $\mathbf{G}$ can be written as:

$$\mathbf{G}_{ij} = \mathbf{G}_{ij} + \eta_{ij}(\mathbf{X}\mathbf{X}^\top \mathbf{P}^\top - \mathbf{G}\mathbf{P}\mathbf{X}\mathbf{X}^\top \mathbf{P}^\top)_{ij}$$

taking $\eta_{ij} = \frac{\mathbf{G}_{ij}}{(\mathbf{G}\mathbf{P}\mathbf{X}\mathbf{X}^\top \mathbf{P}^\top)_{ij}}$, we get $\mathbf{G}_{ij} = \mathbf{G}_{ij} \frac{(\mathbf{X}\mathbf{X}^\top \mathbf{P}^\top)_{ij}}{(\mathbf{G}\mathbf{P}\mathbf{X}\mathbf{X}^\top \mathbf{P}^\top)_{ij}}$

Similarly, gradient update rule for $\mathbf{P}$ can be written as:

$$\mathbf{P}_{ij} = \mathbf{P}_{ij} + \eta_{ij}(\mathbf{G}^\top \mathbf{X}\mathbf{X}^\top - \mathbf{G}^\top \mathbf{G}\mathbf{P}\mathbf{X}\mathbf{X}^\top)_{ij}$$

taking $\eta_{ij} = \frac{\mathbf{P}_{ij}}{(\mathbf{G}^\top \mathbf{G}\mathbf{P}\mathbf{X}\mathbf{X}^\top)_{ij}}$, we get $\mathbf{P}_{ij} = \mathbf{P}_{ij} \frac{(\mathbf{G}^\top \mathbf{C}\mathbf{C}^\top)_{ij}}{(\mathbf{G}^\top \mathbf{G}\mathbf{P}\mathbf{C}\mathbf{C}^\top)_{ij}}$

Proof for convergence of the above update rule follows the same argument as given by Ding et al. (2010).

3.2 Extracting user features

Once the soft-cluster membership vector embeddings for items are extracted, the second stage in our algorithm is to estimate the user features. Following Steck (2013), we use regularized weighted non-negative matrix factorization to estimate the user features given item features. We use the Frobenius norm on the user feature matrix as the regularizer. As the ratings and item features are assumed to be non-negative, we impose non-negativity constraints on the user features also. The optimization problem to estimate the user features given the item feature matrix ($\mathbf{G}$) takes the form

$$\underset{\mathbf{U} \geq 0}{\operatorname{argmin}} \|\mathbf{W} \odot (\mathbf{U}\mathbf{G}^\top - \mathbf{R})\|^2 + \frac{1}{2}\|\mathbf{U}\|^2 \tag{4}$$

here $\odot$ indicates the Hadamard product and $\mathbf{W}$ is the weight matrix defined as

$$\mathbf{W}_{ij} = \begin{cases} 1 \text{ if } \mathbf{R}_{ij} \neq 0, \\ 0 \text{ if } \mathbf{R}_{ij} = 0 \end{cases} \tag{5}$$

The optimization problem in Eq. (4) is a constrained convex problem and it can be solved efficiently. Due to the Hadamard product term, the solution to the above problem cannot be expressed in a simple closed form. The problem can be re-written such that the solution for the ith row of $\mathbf{U}$ can be obtained by solving non-negative least square (nnls) problem. Let $\mathbf{W}^i$ be the $n \times n$ diagonal matrix with ith row of $\mathbf{W}$

Algorithm 1: Soft-cluster Embedding based Cold start item recommendation (SEC)

Input : $\mathbf{R}, \mathbf{A}$

1 Construct enriched matrix $\mathbf{X} = \mathbf{A}\mathbf{A}^\top\mathbf{A}$;
2 Extract item embeddings matrix $\mathbf{G}$ by solving Eq. (2);
3 Extract user embeddings $\mathbf{U}$ by solving Eq. (4);
4 Estimate the rating matrix $\hat{\mathbf{R}} = \mathbf{U}\mathbf{G}^\top$;

Output: Top-N- unobserved items in the descending order of $\hat{\mathbf{R}}$

as the diagonal entries of $\mathbf{W}^i$, ith row of $\mathbf{U}$ (represented as column vector $\mathbf{U}_i$) can be obtained by solving the below regularized nnls

$$\underset{\mathbf{U}\geq 0}{\operatorname{argmin}} \|\mathbf{Q}^i\mathbf{U}_i - \mathbf{b}_i\|^2 + \frac{1}{2}\|\mathbf{U}_i\|^2$$

where $\mathbf{Q}^i = \mathbf{W}^i\mathbf{G}$ and $\mathbf{b}_i = \mathbf{W}^i\mathbf{R}_i$. The problem can be converted to standard nnls problem $\operatorname{argmin}_{\mathbf{U}\geq 0} \|\hat{\mathbf{Q}}^i\mathbf{U}_i - \hat{\mathbf{b}}_i\|$ by introducing the matrix $\hat{\mathbf{Q}}^i = \begin{bmatrix} \mathbf{Q}^i \\ \frac{\mathbf{I}}{\sqrt{2}} \end{bmatrix}$ and the vector $\hat{\mathbf{b}}_i = \begin{bmatrix} \mathbf{b}_i \\ 0 \end{bmatrix}$ where $\mathbf{I}$ is the $k \times k$ identity matrix. In practice, matrix product terms $\mathbf{Q}^i = \mathbf{W}^i\mathbf{G}$ and $\mathbf{b}_i = \mathbf{W}^i\mathbf{R}_i$ can be computed in linear time using broadcasting techniques available in popular linear algebra packages.

3.3 Ranking in item cold-start settings

Finally, once the item and user embeddings are extracted, preference scores for the unobserved items including the cold items can be obtained as $\hat{\mathbf{R}} = \mathbf{U}\mathbf{G}^\top$. Top-$N$ recommendation is carried out by ranking the top-N unobserved items according to the estimated preference scores. The complete algorithm is given in Algorithm 1.

In practice, the embeddings for a new out-of-train item can be obtained as follows. The enriched side-information vector for the new item can be obtained using vector-matrix multiplication. Given the one-hot encoded side-information vector a for a new item, $x = aA^TA$ gives the corresponding enriched side-information vector. Here A^TA is the stored historical co-occurrence frequency matrix. The item embedding vector G_a can be obtained by solving Eqs. (2) or (3) for the fixed $\mathbf{F}$ and x. The cluster centroids matrix ($\mathbf{F}$) is fixed as it is computed using historical data.

We would like to point out that in our algorithm, joint optimization of soft clustering and matrix completion objectives resulted in poor performance due to error propagation in the clustering and completion tasks as observed by Barjasteh et al. (2015). Hence, a two-stage approach is necessary to get lower prediction error and better performance. Later, we study this effect in detail.

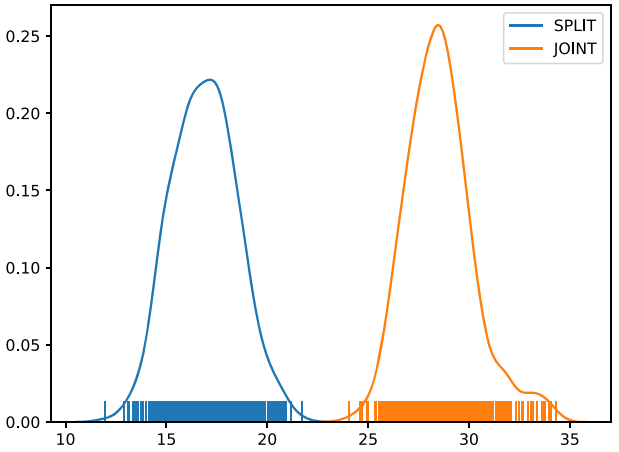

Fig. 2 Approximation error distribution for joint versus split optimization

3.4 Handling user cold-start problem

Algorithm 1 can be readily modified to handle the user cold-start problem. If $\mathbf{B}$ is the user side-information matrix, we define the enriched user matrix as $\mathbf{Y} = \mathbf{B}\mathbf{B}^\top\mathbf{B}$. We can extract the soft cluster membership embedding vectors for users by solving the relaxed version of Eq. (2) for $\mathbf{Y}$. Given the extracted user features $\mathbf{U}$, item features $\mathbf{G}$ can be estimated by solving the optimization problem 4 for $\mathbf{G}$. The remaining steps follow as in Algorithm 1.

3.5 Split versus joint optimization

We conclude this section by highlighting the importance of the split (two-step) optimization procedure in our proposed algorithm. Our hypothesis claims that the cold and warm items can be soft-clustered in the enriched side-information space and the user rating for an item can be obtained by the inner product between the user feature and the item cluster association vectors. Our hypothesis also suggests that performance improvement reported in Sect. 4 can be obtained by split optimization as opposed to the joint-optimization proposed in other approaches for cold-start recommendations (Saveski and Mantrach 2014; Zhou et al. 2011; Krohn-Grimberghe et al. 2012; Singh and Gordon 2008). We empirically validate this hypothesis below.

We create 1000 instances of the optimization problem by randomly drawing the values of $\mathbf{G}$, $\mathbf{F}$ and $\mathbf{U}$ from a random uniform distribution. The resulting values of $\mathbf{X}$ and $\mathbf{R}$ are given as input to the joint and split optimization strategies. The distribution of the approximation errors, Frobenius norm of the difference between true $\mathbf{U}$ and the estimated $\mathbf{U}$ by the corresponding optimization strategies, are plotted in Fig. 2. As shown in the figure, split optimization results in lower approximation error (mean error of ~ 17) compared to the joint optimization (mean error of ~ 27). It should also be noted that variance of the split optimization is slightly higher than the joint optimization. In

practice, joint optimization of (2) and (4) can be done using alternating minimization techniques (Jain et al. 2017). Although the joint optimization problem is bi-convex and closed form solutions exist for each variable update, it is time consuming compared to the split optimization strategy. In case of joint optimization one has to solve three non-negative least square problems for each row of the variable matrix, whereas in case of split optimization, results can be obtained using a multiplicative update rules and a single non-negative least square solution, thus reducing the time complexity in addition to the performance improvement.

In our specific case, split optimization works better because of the fact that cluster centroids $\mathbf{F}$ do not depend on users' preferences and are consequently independent of the user feature matrix $\mathbf{U}$. In joint optimization, at each iteration $\mathbf{F}$ is re-computed according to the value of $\mathbf{U}$ through $\mathbf{G}$, as $\mathbf{G}$ value is calculated based on $\mathbf{U}$. In split optimization, $\mathbf{F}$ is not affected by the value of $\mathbf{U}$ and thus results in better approximation of the centroids. Our experimental study on real world datasets also confirms the above observation.

4 Experiments

The main purpose of this section is to compare the performance of different state-of-the-art algorithms against the proposed one. Though many recent studies (Yang et al. 2019; Dacrema et al. 2019; Ludewig et al. 2019) demonstrated that deep learning based algorithms perform inferior to content + collaborative algorithms in recommendation tasks, there is a growing interest within the community to adapt deep learning based models. Hence, we carry out two sets of experiments: in the first set of experiments we compare our algorithms against state-of-the-art content + collaborative algorithms and in the second set of experiments we compare the proposed algorithm against recent graph embeddings based deep learning algorithms. In both cases, we evaluate different approaches in a movie cold-start recommendation setting. We also perform qualitative analysis of our results to explain the recommendations provided by the proposed algorithm.

4.1 Comparison against non-deep learning frameworks

4.1.1 Datasets

We used three benchmark movie datasets for our first set of evaluation (ii) MovieLens *20M* (ii) MovieLens *1M* and (iii) Yahoo! Movies. MovieLens *20M* contains 20,000,263 ratings of 138,493 users across 27,278 movies. MovieLens *1M* contains 1,000,209 rating values of 6040 users for 3706 movies. In both MovieLens *20M* and *1M* datasets, every user has rated at least 20 movies. In case of the Yahoo! Movies, we removed movies with missing genres and users with less than 15 ratings and the final dataset contained 138,310 rating values for 3429 users and 8067 movies. In case of MovieLens *1M* and Yahoo! Movies, we used genres as the side-information and the characteristic matrix is constructed based on the genres associated with movies.

Consequently, the dimension of the side-information space is 18 and 25, the number of total genres for MovieLens and Yahoo! Movies respectively. MovieLens *20M* dataset contains additional movie tags given by different users. The dataset contains 35,169 unique (disregarding the case-sensitivity) tags across 19,545 movies. In the final processing we filtered out the tag data which does not appear a minimum of 200 times. The side information matrix is built using the genres and remaining tag information. Final dimension of the side-information space was 442.

4.1.2 Baselines

We chose four item cold-start recommendation algorithms as baselines: Decoupled matrix Completion and Transduction (DCT) (Barjasteh et al. 2015), Linear Low-Rank Regression (LCO) (Sedhain et al. 2017), Local Collective Embeddings (LCE) (Saveski and Mantrach 2014) and Attribute-to-Feature Mappings (AFM) (Gantner et al. 2010). LCE employs a joint optimization strategy but DCT and AFM employ split optimization strategy like in our case. LCE can be considered as a special case of factorization machines which uses only 1-level of variable interactions. Other commonly employed baselines like random recommendations are excluded in our study as previous studies showed that they always perform poorly compared to our baselines (Barjasteh et al. 2015; Saveski and Mantrach 2014). Our proposed algorithm is called Soft-cluster Embedding based Cold-start recommendation (SEC).

Decoupled matrix Completion and Transduction (DCT) The algorithm is composed of matrix *complete* and information *transduction* steps. In the *complete* step, a sub-matrix extracted from the original rating matrix is completed using matrix completion techniques and in the *transduction* step a representative subspace from the item similarity matrix is extracted using top eigenvectors. Completed sub-matrix and representative subspace are used to estimate the missing rating values.

Low-Rank Linear Cold-Start Recommendation (LCO) LCO models the rating values as the weighted sum of the available side-information data Sedhain et al. (2017). The observed rating values are used to estimate a low-dimensional weight matrix from a subset of parametrized low-rank matrices. LCO algorithm was previously used in the user cold-start settings and here we adapt it to item cold-start settings by using item side-information data. Pure content based cold-start recommendation can be considered as a special case of LCO algorithm where the weight matrix is replaced with the side-information matrix of the warm items.

Local Collective Embedding (LCE) This is an extension to the collective matrix factorization approach proposed by Singh and Gordon (2008). The LCE algorithm exploits the local geometric structure of the low dimensional latent space by imposing that two items closer in the intrinsic geometry should be closer in the low-dimensional space (Saveski and Mantrach 2014). To achieve this, a manifold regularizer (Belkin et al. 2006) is added to the collective matrix factorization objective function.

Attribute-to-Feature Mappings (AFM) For completeness, we compare our algorithm against the algorithm proposed by Gantner et al. (2010). Though the proposed method is a bit older than other baselines, since the work is closely related to our work, we include that in our comparison. AFM is a two-step algorithm where user and item embeddings are learned from side-information by modeling the rating matrix as a

linear combination of the embeddings. The extracted user features are used with item metadata to learn the item weight matrix for different side-information categories for both cold and warm items in the second stage.

4.1.3 Evaluation metrics

In addition to the standard ranking metrics Normalized DCG (NDCG) and Mean Average Precision (MAP), we define a new metric called *Cold Items Precision* (CIP) to quantify the number of cold-start items recommended.

Cold Items Precision (CIP) Majority of the previous work on cold-start recommendation, including our baselines, used only ranking metrics like NDCG or MAP to evaluate the performance. But in practice, items to be recommended are a mix of both warm and cold items. By using only ranking metrics, one fails to quantify the ability of the algorithm to recommend cold items. For an effective comparison, in addition to the ranking metrics, one has to measure the number of recommended cold items. For this, we define the metric *Cold Items Precision* (CIP):

$$CIP = \frac{\#\text{ of recommended cold items}}{\#\text{ of recommendations}}$$

A higher CIP value means that more cold items are included in the recommendation in addition to the unobserved warm items. In the case of MAP, since it is a binary ranking metric, we discretize the rating values such that 4 and 5 ratings are considered relevant and others as irrelevant.

4.1.4 Experimental protocol

We followed the same experimental protocol as in the baseline algorithm Barjasteh et al. (2015). We employed fivefold validation by partitioning the items set into 5 equal disjoint subsets, and using one as testing and the rest as training set. The ratio of testing to training data is 1:4, and the experiment is repeated for each of these 5 disjoint partitions. Our task is to recommend a fixed number of relevant warm and/or cold movies from the unobserved test data. The representative users are determined based on the training set only. We consider recommendation as top-N ranking problem and follow the evaluation strategy described in Parambath et al. (2016) and Steck (2013). In the case of the SEC, the number of clusters is set to 60 for MovieLens *20M*, and 25 for MovieLens *1M* and Yahoo! Movies dataset. For the baseline algorithms, hyperparameters are chosen using grid search, wherever applicable, and reported results correspond to the optimal set of hyperparameters. We use Friedman and Nemenyi post-hoc test to check the statistical significance of results (Demšar 2006).

4.1.5 Results and discussion

In the case of SEC, we experimented with NMF and Convex-NMF clustering schemes. NMF and Convex-NMF results were very similar, with the NMF scheme giving slightly better results compared to Convex-NMF. Below, we report the results of our algorithm

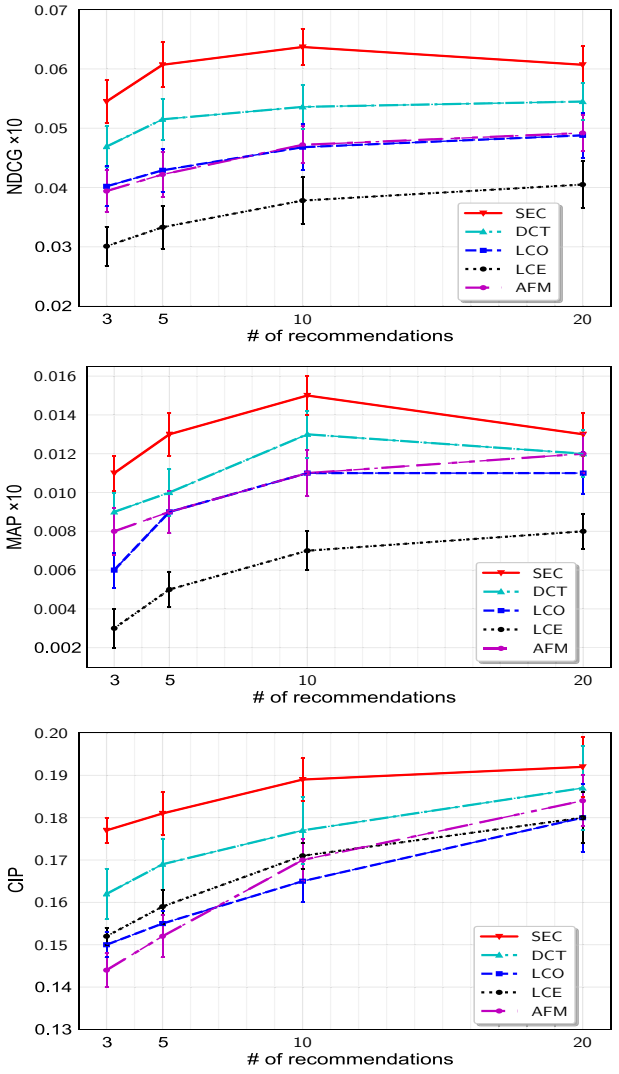

Fig. 3 Comparison of the performances of different non-DL algorithms on cold-start movie recommendation task (MovieLens *20M*, SEC = ours)

with NMF scheme. Later, we compare the performance between the NMF and Convex-NMF schemes on MovieLens *1M* dataset.

Figures 3, 4 and 5 present the results of our experimental study with error bars in terms of NDCG, MAP and CIP on the MovieLens *20M*, MovieLens *1M* and the Yahoo! Movies datasets respectively. Our proposed SEC algorithm outperformed all the baselines in terms of the relevance ranking metrics NDCG and MAP. The first major inference we can make from our experimental study is that the choice of the optimization strategy (split-vs-joint optimization detailed in Sect. 3) affects the results

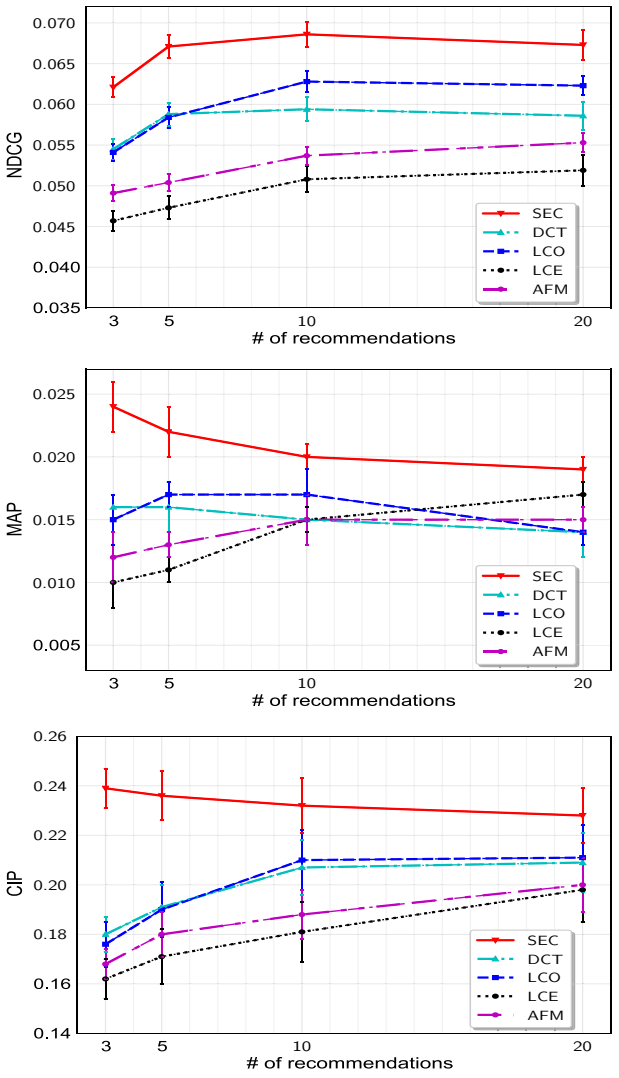

Fig. 4 Comparison of the performances of different non-DL algorithms on cold-start movie recommendation task (MovieLens *1M*, SEC = ours)

on real world datasets as well i.e., joint optimization based techniques perform poorly compared to split optimization techniques. Split optimization based algorithms like SEC, DCT and AFM performed better than the joint optimization based LCE. This can be seen for all three datasets used.

In the case of MovieLens *20M* dataset, even for higher values of $N \geq 10$, SEC consistently outperformed its peers. In practical cases, due to budget constraints like the number of available slots in the webpage or due to the screen size, the value of N is limited to 5 or less. Hence, it is very important to note the performance of any

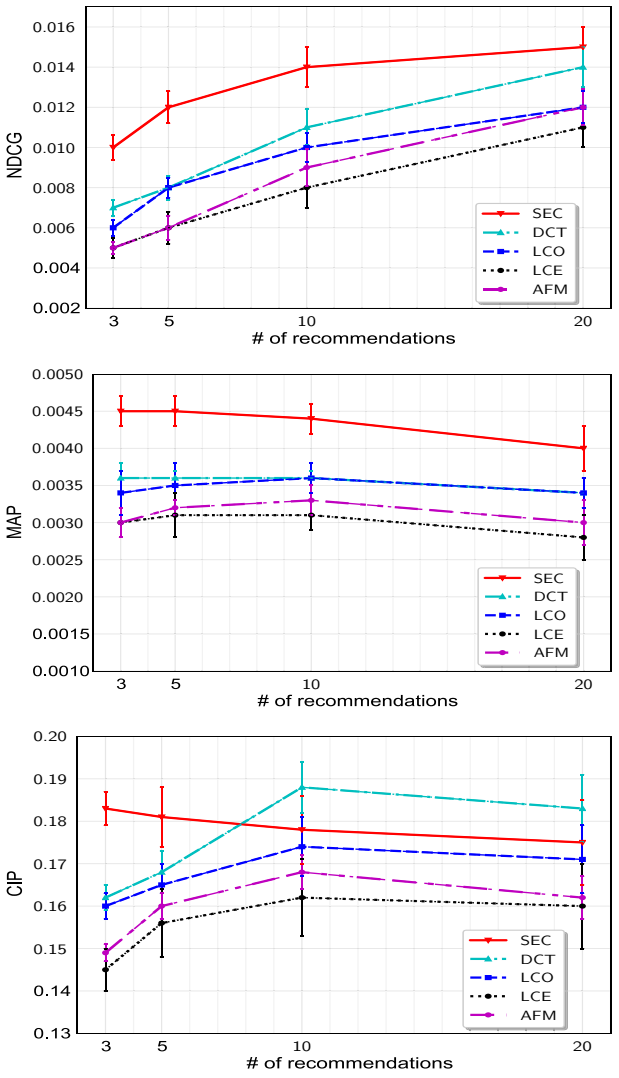

Fig. 5 Comparison of the performances of different non-DL algorithms on cold-start movie recommendation task (Yahoo! Movies, SEC = ours)

recommendation algorithm for lower values of N. A good recommendation algorithm should have higher values for NDCG and MAP for lower values of N. SEC results in better performance on MovieLens *20M*, MovieLens *1M* and Yahoo! Movies for smaller number of recommendations (N) values. As the value of N increases, the majority of the relevant items are also included in the recommendation list by the baseline algorithms. Moreover, SEC results are statistically significant compared to the second-best methods LCO and DCT. It should also be noted that less than half of the movies (11,516 out of 27,278) in the MovieLens *20M* dataset have more than ten

4 or 5 ratings whereas in MovieLens *1M* 2811 movies out of 3706 have more than ten 4 or 5 ratings. Hence in general performance metric values for MovieLens *1M* is higher than for MovieLens *20M* dataset.

In case of MovieLens *1M*, SEC improved NDCG values by more than 10% compared to LCO and DCT, the second-best performer. Though AFM is closely related to our work, due to the overfitting effect we discussed in Sect. 2, its performance is not up to par with other baselines. It can also be noted that as the recommendation size increases, MAP values for both MovieLens *20M*, MovieLens *1M* and Yahoo! Movies decrease. This is due to the low discretization threshold we use for the binary ranking metric MAP in the experiments. We deem an item as relevant only when the observed rating value is 4 or 5. Though the average number of relevant ratings per user in MovieLens and Yahoo! Movies is relatively high, the median relevant ratings per user is very small for the MovieLens and Yahoo! Movies datasets. From the above observation it can be inferred that for the majority of the users, our proposed algorithm recommends all the relevant test items in the top-5 and top-3 rankings. Hence, on average as the recommendation size increases to more than 5, the MAP values decrease.

In the case of MovieLens, both *1M* and *20M*, SEC was able to recommend more cold items as compared to others. Higher values of CIP and MAP for SEC clearly indicate that most of the cold items SEC recommended have ratings values of 4 or 5. In the case of Yahoo! Movies, when the recommendation size is more than 5, DCT recommended more cold items than SEC, but lower MAP and NDCG values for DCT clearly show that most of these items are not very relevant. It should also be noted that even though the proposed CIP measures the number of cold items in the recommendations, it alone does not capture the relevance of the cold items. A good cold-start recommendation algorithm should have higher values for CIP, NDCG and MAP, as NDCG and MAP measure overall usefulness of the recommendation. Our proposed algorithm achieves these requirements.

Parameter analysis SEC model has one hyperparameter: k, the number of clusters. Figure 6 shows a typical behaviour of SEC when k is varied. The results are averaged over 5 runs of the algorithm on the MovieLens data. Using smaller values of k results in underfitting the data whereas larger values result in k overfitting. We noticed a similar trend in the case of MovieLens *20M* and Yahoo! Movies as well. One can choose the best value for k that fits the data using standard techniques (Pham et al. 2005). There is a growing interest in parameter free clustering techniques (Sarfraz et al. 2019), and considering such techniques for cold-start recommendation will free us from hyperparameter tuning. We leave this as a future work.

Running time comparison We compare the running time complexity of the proposed algorithm against baseline algorithms. The soft clustering stage of the SEC algorithm has two update steps. Running time for the $\mathbf{F}$ and $\mathbf{G}$ update steps are in the order of $\mathcal{O}(t((n+p)k^2+npk))$ where $t \sim 100$ is the number of iterations to convergence. Often in practice, $k << n$. The second stage in the SEC algorithm: solving for $\mathbf{U}_i$ can be done in $\mathcal{O}(k^2)$ using fast projected gradient methods (Polyak 2015). Time complexity of LCO algorithm is in the order of $\mathcal{O}(k^3 + k^2(n + p))$ (Sedhain et al. 2017). DTC requires solving matrix completion problem on a submatrix, which amounts to solving non-negative least squares iteratively. The running time for matrix completion takes $\mathcal{O}(t(m + n)k^3)$ and extracting the top eigenvectors from the item similarity matrix

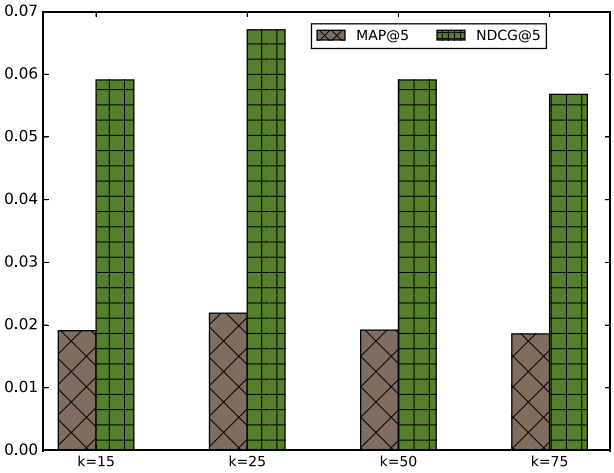

Fig. 6 Performance of SEC as a function of the hyperparameter k (# of clusters)

takes $\mathcal{O}(k^3)$ time. LCE optimization problem can be solved using multiplicative update rules like in our case and running time is in the order of the clustering step of the SEC algorithm (Saveski and Mantrach 2014). In addition to the time savings, our algorithm gives a clear and simple intuition on how items are selected for recommendation (see Fig. 1).

Richness of side-information Our approach relies on extracting soft cluster membership vector embeddings for items using the item side-information. Hence, it is very important that side-information contains meaningful correlation information between different sub-categories. To study the importance of the richness of the side-information, we create synthetic data such that it follows the same rating distribution as the MovieLens data, with same n and m values, but with random side-information such that correlation among different sub-categories in the data is low.

The results of our study is plotted in Fig. 7. SEC algorithm with item embeddings extracted from random synthetic side-information data (RAND) performed worse than the worst performer LCE algorithm. Thus, in practice, it is important that side data contains meaningful information for the SEC algorithm to work.

Raw versus enriched side-information We investigate the effectiveness of the item embeddings obtained using raw side-information and enriched side-information. We run SEC by extracting item embeddings directly from the raw side-information $\mathbf{A}$ and comparing the results against SEC with item embeddings obtained using the enriched side-information $\mathbf{X}$. SEC with item embeddings using enriched data resulted in better performance compared to the SEC with item embeddings using raw data. This result is not surprising as the enriched data captures possible dependencies between different item metadata categories compared to raw side data. However, SEC with raw side-information still performed better than our baseline algorithms. The plot for the comparison against raw vs enriched side-information on the MovieLens dataset is given in Fig. 8. For a fair comparison, we also plotted the second-best performing algorithm, LCO.

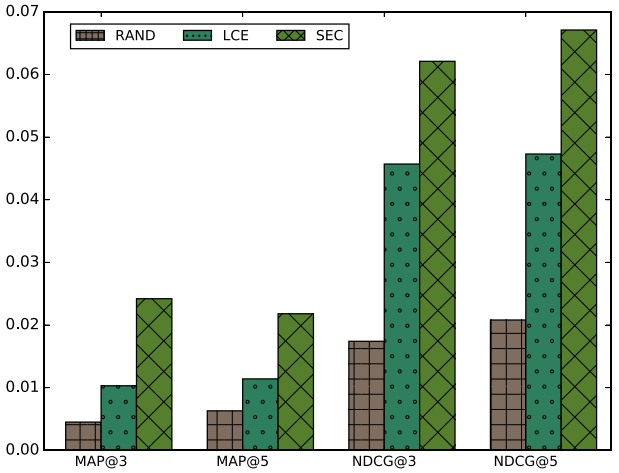

Fig. 7 Comparison of SEC algorithm with synthetic side-information (RAND) against real side-information (SEC) and lowest performed algorithm in our study (LCE)

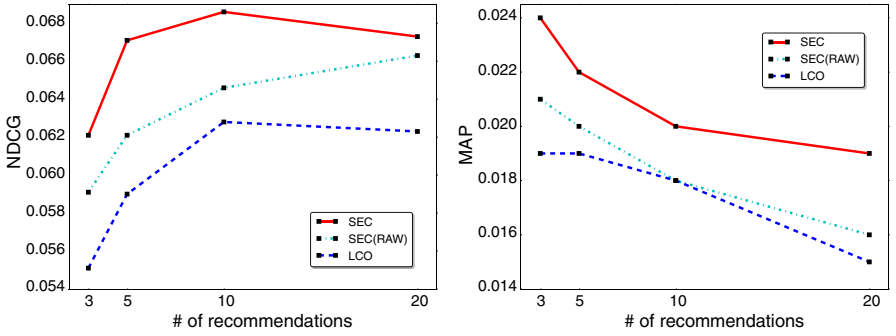

Fig. 8 Raw versus enriched side-information effect on recommendation

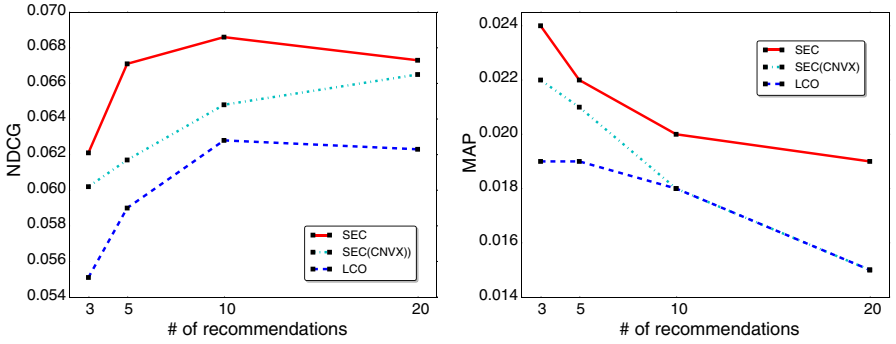

Fig. 9 NMF versus convex-NMF clustering scheme effect on recommendation

NMF versus convex-NMF clustering Finally, we compare the performance of the cold-start recommendation using vanilla NMF (non-convex version) against convex NMF clustering schemes. Our experimental results are plotted in Fig. 9. SEC and SEC(CNVX) represent the vanilla NMF and convex NMF schemes respectively whereas LCO represents the second-best performer in our experimental study. Though Convex NMF gives better estimation of the cluster centroids, in practice, vanilla NMF based clustering gives better performance. A possible hypothesis for these results is that item clusters are more dispersed than centered. The values for the ranking metrics NDCG and MAP are higher for the vanilla NMF based cold-start recommendation compared to the Convex NMF scheme. It is evident from Fig. 9 that Convex-NMF performs suboptimally compared to vanilla NMF, whereas it performs better than LCO, the second-best algorithm in our experimental study. Like in the case of Raw versus Enriched Side-Information study, we can conclude that vanilla NMF is the preferred clustering scheme to get state-of-the-art results. However, Convex NMF based clustering also results in better performance compared to other cold-start recommendation algorithms.

4.2 Comparison against deep learning algorithms

In this section, we conduct experiments to compare our algorithm against three deep learning (DL) based algorithms. For completeness, we also include the results of the DCT algorithm, which came second in our experimental study with non-DL based algorithms on MovieLens *20M* dataset. We choose the following three deep learning algorithms for our study.

4.2.1 Baselines

Neural Graph Collaborative Filtering (NGCF) (Wang et al. 2019a) NGCF makes use of higher order connection information between users and items to improve traditional deep collaborative recommendation algorithms. NGCF (Wang et al. 2019a) randomly initializes the user and item embeddings and refines the embeddings by injecting higher order collaborative signals in the form of the Laplacian of the user-item graph and propagating through multiple neural network layers. The Laplacian is constructed from the block diagonal user-item bipartite graph. Higher order connectivity information is encoded by stacking multiple propagation layers. In our experiments, we used the source code provided by the authors.[4]

Graph Convolutional Matrix Completion (GCMC) GCMC is based on semi-supervised graph convolutional networks (Berg et al. 2018). It works by generating embeddings for users and items considering only the first-order relationships between users and items. Given the adjacency matrix, graph auto-encoder is constructed by minimizing the reconstruction error between the predicted ratings and the observed ratings and the decoder is defined as a function acting on the user and item embeddings

[4] https://git.io/JvVkv.

and returning the reconstructed rating value. In our experiments, we used the source code provided by the authors.[5]

Hybrid Collaborative Filtering with Autoencoders (HCFA) Publicly released code for NGCF and GCMC does not make use of the item side-information associated with the data. Hence, we use one more deep learning based algorithm termed HCFA (Strub et al. 2016). HCFA use stacked denoising autoencoders to learn a non-linear representation of users and items by integrating the side information with the sparse rating data. The item side-information is integrated to the item embeddings by simply appending the side-information to the rating information. In our experiments, we used the source code provided by the authors.[6]

4.2.2 Dataset and experimental protocol

The code released by the authors for baseline DL approaches failed to run using MovieLens *20M* dataset due to memory issues.[7] Hence for this set of experiments we used the MovieLens *10M* dataset. Unlike in the matrix factorization based approaches, NGCF and GCMC work by building user-item bipartite graphs from observed relevant ratings. This implies that the training set should contain at least one positive rating for every item and for every user. Hence the approach cannot be applied to extreme cold-start settings. In extreme item cold-start settings, one assumes that no rating information about an item is available a priori. So we test our algorithm in non-extreme cold-start item settings. We remind the readers that we used the code as it is in the corresponding repositories and used the default configuration settings given by the authors.

We randomly split the ratings data into training and test sets respectively. We then removed the users and items with zero relevant ratings from the training set. We assume that a movie is relevant to a user if it is rated at least with a rating value of 4. Finally, we make sure that sets of test users are a proper subset of training users by removing all the test users and items that do not appear in the training set. This whole procedure is carried out five times randomly, and the reported results are averaged over five splits. For all three deep learning based algorithms, we used default parameter settings mentioned in the respective code. For evaluation, we ranked all the items in the test set for every test user based on the output score obtained from the corresponding algorithms.

Since CIP cannot be used in non-extreme cold-start settings, to estimate the cold-start recommendation capacity of the algorithms, we report the fraction of predicted items in top 20 recommendations which has less than 1, 2 and 3 ratings in the training set. We define CIP@k as the CIP of an item with less than or equal to k ratings in the training set, in addition to the relevance ranking metrics DCG and MAP.

[5] https://git.io/JvVI5.

[6] https://git.io/JvVL2.

[7] We used a 512 GB RAM machine with Nvidia GPUs to run all the experiments.

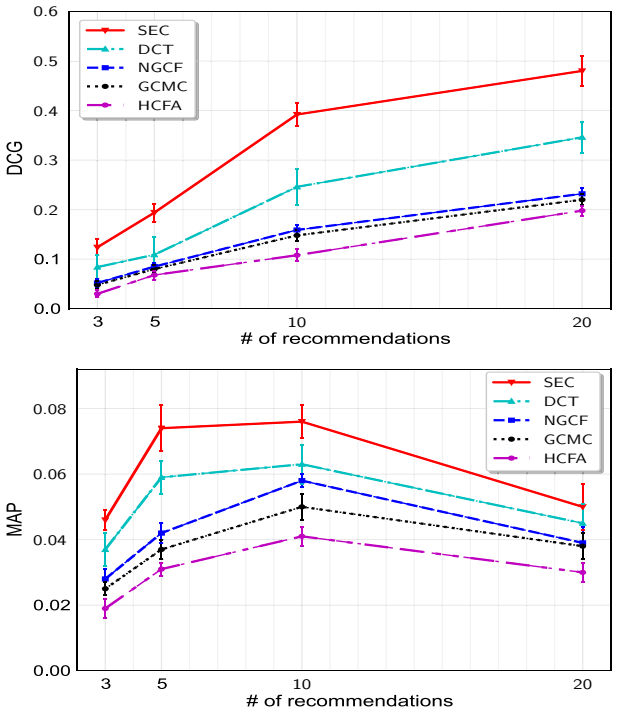

Fig. 10 Performance comparison of SEC algorithm against DL based algorithms (MovieLens *10M*, SEC = ours)

Table 1 CIP@k values for different algorithms for $k = 1, 2, 3$

Baseline	CIP@3	CIP@4	CIP@5
SEC (ours)	0.59	0.74	0.82
DCT	0.30	0.54	0.73
NGCF	0.18	0.31	0.46
GCMC	0.15	0.26	0.42
HCFA	0.43	0.81	0.94

4.2.3 Results

DCG and MAP values for different algorithms are plotted in Fig. 10. Deep Learning based algorithms perform poorly compared to the non-DL based SEC and DCT. Our results conform to the recent analysis of the performance of deep learning based algorithms on different recommendation related tasks (Dacrema et al. 2019). The same result has been noted in specific application settings like ad click prediction (McMahan et al. 2013).

Table 1 contains CIP@k values for different algorithms for $k = \{1, 2, 3\}$. Though HCFA returns many cold-items (possibly due to the explicit use of side-information), the majority of them are not relevant as indicated by the low MAP and DCG values.

Table 2 Subset of 5 relevant training set movies for a candidate user, and Top 5 recommendations for the user on Movielens *1M* dataset

Relevant training items	SEC (ours)	DCT	LCO
Total recall	***Lethal weapon 2***	Legends of the fall	From dusk till dawn
Rush hour	Get shorty	***X-Men***	Montana
Leathal weapon	**Gat carter**	Serial Mom	**Air America**
Romancing the stone	From dusk till dawn	**U-571**	***Lethal weapon 2***
Enemy of the state	***Star wars: episode V***	Montana	Get shorty

Items in bold letters are relevant test items whereas the entries with bold italicized letters are relevant cold items

In general, deep learning based algorithms perform suboptimal compared to SEC and DCT when recommending relevant cold-start items.

4.3 Qualitative analysis

To conclude the experimental validity of the proposed algorithm, we conduct a qualitative analysis of the proposed algorithm against two strong baselines in our experiments using MovieLens *1M* dataset. For this analysis, we randomly choose a user and Top 5 recommendations for this user. We compare the set of recommendations made by different algorithms for this user and see how well it matches with the users' true preferences. Our results are given in Table 2. From the list of relevant training movies, it is very clear that the user likes movies from Action, Crime, Thriller genres. The top 5 recommendations provided by the SEC algorithm spans the genres Action, Crime, Thriller and Adventure. Though none of the movies in the training set correspond to Adventure genre, SEC algorithm was able to associate the correlation between strongly connected Action and Adventure genres in the soft-cluster embeddings. As a result, the top 5 recommendations made by SEC algorithm contain a relevant cold movie, Star Wars: Episode V—The Empire Strikes Back, whereas other baselines fail to do so. Not surprisingly, all the cold-start recommendation algorithms correctly recommended Action, Crime, Thriller movies for this user. Overall, top 5 recommendations made by the proposed SEC algorithm covers three relevant movies which span the relevant genres out of which two are cold items. The second-best performing algorithms DCT and LCO also recommend many Action genre movies but contain a smaller number of relevant and cold items compared to the proposed algorithm.

5 Conclusion

We presented a new method to solve the item cold-start recommendation problem. The algorithm can be applied in any item cold-start recommendation scenario with access to item side-information. Our exhaustive experiments on benchmark datasets showed that the algorithm performs well in terms of popular relevance metrics compared to strong content, content + collaborative and deep learning based baselines. Qualitative

analysis of the experimental results explains the recommendation provided by the proposed algorithm.

One of the main issues associated with the proposed algorithm is that the number of clusters (k) has to be fixed beforehand using heuristics or based on cross-validation results. An interesting topic for future work is: how to select the number of soft-clusters automatically. There is a growing interest in parameter free clustering techniques, and considering such techniques for cold-start recommendation will free us from hyperparameter tuning. Another promising line of future research is to consider the bandit version of the soft-clustering based recommendation algorithm. Majority of the current collaborative bandit algorithms use hard clustering to estimate the mean reward of the unobserved items. We plan to extend the work to soft-clustering bandits for recommendations.

Acknowledgements Open Access funding provided by the Qatar National Library.

References

Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans Knowl Data Eng 17:734–749

Aharon M, Anava O, Avigdor-Elgrabli N, Drachsler-Cohen D, Golan S, Somekh O (2015) Excuseme: asking users to help in item cold-start recommendations. In: Proceedings of the 9th ACM conference on recommender systems (RecSys). ACM, pp 83–90

Anava O, Golan S, Golbandi N, Karnin Z, Lempel R, Rokhlenko O, Somekh O (2015) Budget-constrained item cold-start handling in collaborative filtering recommenders via optimal design. In: Proceedings of the 24th international conference on World Wide Web, international World Wide Web conferences steering committee, pp 45–54

Barjasteh I, Forsati R, Masrour F, Esfahanian AH, Radha H (2015) Cold-start item and user recommendation with decoupled completion and transduction. In: Proceedings of the 9th ACM conference on recommender systems (RecSys). ACM, pp 91–98

Belkin M, Niyogi P, Sindhwani V (2006) Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. J Mach Learn Res 7(Nov):2399–2434

Berg RVD, Kipf TN, Welling M (2018) Graph convolutional matrix completion. In: ACM SIGKDD DeepLearning workshop

Chou SY, Yang YH, Jang JSR, Lin YC (2016) Addressing cold start for next-song recommendation. In: Proceedings of the 10th ACM conference on recommender systems (RecSys). ACM, pp 115–118

Covington P, Adams J, Sargin E (2016) Deep neural networks for youtube recommendations. In: Proceedings of the 10th ACM conference on recommender systems, pp 191–198

Dacrema MF, Cremonesi P, Jannach D (2019) Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In: Proceedings of the 13th ACM conference on recommender systems, ACM, RecSys '19, pp 101–109

Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7(Jan):1–30

Ding C, He X, Simon HD (2005) On the equivalence of nonnegative matrix factorization and spectral clustering. In: Proceedings of the 2005 SIAM international conference on data mining (SDM). SIAM, pp 606–610

Ding CH, Li T, Jordan MI (2010) Convex and semi-nonnegative matrix factorizations. IEEE Trans Pattern Anal Mach Intell 32(1):45–55

Gantner Z, Drumond L, Freudenthaler C, Rendle S, Schmidt-Thieme L (2010) Learning attribute-to-feature mappings for cold-start recommendations. In: 2010 IEEE 10th international conference on data mining (ICDM). IEEE, pp 176–185

Gouvert O, Oberlin T, Févotte C (2018) Matrix co-factorization for cold-start recommendation. In: Proceedings of the 19th international society for music information retrieval conference, ISMIR. ACM: Association for Computing Machinery, pp 792–798

Hu Y, Koren Y, Volinsky C (2008) Collaborative filtering for implicit feedback datasets. In: 2008 Eighth IEEE international conference on data mining. IEEE, pp 263–272

Jain P, Kar P et al (2017) Non-convex optimization for machine learning. Found Trends® Mach Learn 10(3–4):142–336

Krohn-Grimberghe A, Drumond L, Freudenthaler C, Schmidt-Thieme L (2012) Multi-relational matrix factorization using bayesian personalized ranking for social network data. In: Proceedings of the fifth ACM international conference on Web search and data mining. ACM, pp 173–182

Kula M (2015) Metadata embeddings for user and item cold-start recommendations. In: Bogers T, Koolen M (eds) Proceedings of the 2nd workshop on new trends on content-based recommender systems co-located with RecSys 2015, CEUR-WS.org, CEUR workshop proceedings, vol 1448, pp 14–21

Lee DD, Seung HS (2001) Algorithms for non-negative matrix factorization. In: Advances in neural information processing systems (NIPS), pp 556–562

Li T, Ding C (2006) The relationships among various nonnegative matrix factorization methods for clustering. In: Sixth international conference on data mining (ICDM'06). IEEE, pp 362–371

Li T, Ma S (2004) IFD: iterative feature and data clustering. In: Proceedings of the 2004 SIAM international conference on data mining (SDM). SIAM, pp 472–476

Lin J (2019) The neural hype and comparisons against weak baselines. In: ACM SIGIR Forum, vol 52. ACM, pp 40–51

Ludewig M, Jannach D (2018) Evaluation of session-based recommendation algorithms. User Model User Adap Inter 28:331–390

Ludewig M, Mauro N, Latifi S, Jannach D (2019) Performance comparison of neural and non-neural approaches to session-based recommendation. In: Proceedings of the 13th ACM conference on recommender systems, RecSys '19. ACM, pp 462–466

McMahan HB, Holt G, Sculley D, Young M, Ebner D, Grady J, Nie L, Phillips T, Davydov E, Golovin D, et al. (2013) Ad click prediction: a view from the trenches. In: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 1222–1230

Parambath SA, Usunier N, Grandvalet Y (2016) A coverage-based approach to recommendation diversity on similarity graph. In: Proceedings of the 10th ACM conference on recommender systems (RecSys). ACM, pp 15–22

Park ST, Chu W (2009) Pairwise preference regression for cold-start recommendation. In: Proceedings of the third ACM conference on recommender systems (RecSys). ACM, pp 21–28

Pham DT, Dimov SS, Nguyen CD (2005) Selection of k in k-means clustering. Proc Inst Mech Eng C J Mech Eng Sci 219(1):103–119

Piazza A, Kröckel P, Bodendorf F (2017) Emotions and fashion recommendations: evaluating the predictive power of affective information for the prediction of fashion product preferences in cold-start scenarios. In: Proceedings of the international conference on web intelligence, pp 1234–1240

Polyak RA (2015) Projected gradient method for non-negative least square. Contemp Math 636:167–179

Rendle S (2010) Factorization machines. In: 2010 IEEE international conference on data mining. IEEE, pp 995–1000

Rendle S (2012) Factorization machines with libfm. ACM Trans Intell Syst Technol (TIST) 3(3):1–22

Salah A, Rogovschi N, Nadif M (2016) A dynamic collaborative filtering system via a weighted clustering approach. Neurocomputing 175:206–215

Sarfraz S, Sharma V, Stiefelhagen R (2019) Efficient parameter-free clustering using first neighbor relations. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 8934–8943

Saveski M, Mantrach A (2014) Item cold-start recommendations: learning local collective embeddings. In: Proceedings of the 8th ACM conference on recommender systems (RecSys). ACM, pp 89–96

Sedhain S, Menon AK, Sanner S, Xie L, Braziunas D (2017) Low-rank linear cold-start recommendation from social data. In: Thirty-first AAAI conference on artificial intelligence

Singh AP, Gordon GJ (2008) Relational learning via collective matrix factorization. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD). ACM, pp 650–658

Steck H (2013) Evaluation of recommendations: rating-prediction and ranking. In: Proceedings of the 7th ACM conference on recommender systems (RecSys). ACM, pp 213–220

Strub F, Mary J, Gaudel R (2016) Hybrid collaborative filtering with autoencoders. arXiv preprint arXiv:1603.00806

Sun J, Vishnu A, Chakrabarti A, Siegel C, Parthasarathy S (2018) Coldroute: effective routing of cold questions in stack exchange sites. Data Min Knowl Disc 32(5):1339–1367

Ungar LH, Foster DP (1998) Clustering methods for collaborative filtering. In: AAAI workshop on recommendation systems, Menlo Park, CA, vol 1, pp 114–129

Vartak M, Thiagarajan A, Miranda C, Bratman J, Larochelle H (2017) A meta-learning perspective on cold-start recommendations for items. In: Advances in neural information processing systems, pp 6904–6914

Vasile F, Smirnova E, Conneau A (2016) Meta-prod2vec: product embeddings using side-information for recommendation. In: Proceedings of the 10th ACM conference on recommender systems. ACM, pp 225–232

Verstrepen K, Bhaduriy K, Cule B, Goethals B (2017) Collaborative filtering for binary, positiveonly data. ACM SIGKDD Explor Newslett 19(1):1–21

Vlachos M, Dünner C, Heckel R, Vassiliadis VG, Parnell TP, Atasu K (2019) Addressing interpretability and cold-start in matrix factorization for recommender systems. IEEE Trans Knowl Data Eng 31(7):1253–1266

Wang X, He X, Wang M, Feng F, Chua T (2019a) Neural graph collaborative filtering. In: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, SIGIR 2019, Paris, France, July 21–25, 2019, pp 165–174

Wang X, He X, Wang M, Feng F, Chua TS (2019b) Neural graph collaborative filtering. In: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, pp 165–174

Wei J, He J, Chen K, Zhou Y, Tang Z (2016) Collaborative filtering and deep learning based hybrid recommendation for cold start problem. In: 2016 IEEE 14th International conference on dependable, autonomic and secure computing, 14th international conference on pervasive intelligence and computing. IEEE, pp 874–877

Yang W, Lu K, Yang P, Lin J (2019) Critically examining the "neural hype": weak baselines and the additivity of effectiveness gains from neural ranking models. In: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, SIGIR'19. ACM, pp 1129–1132

Zhang M, Tang J, Zhang X, Xue X (2014) Addressing cold start in recommender systems: a semi-supervised co-training algorithm. In: Proceedings of the 37th international ACM SIGIR conference on research and development in information retrieval. ACM, pp 73–82

Zheng L, Lu CT, Jiang F, Zhang J, Yu PS (2018) Spectral collaborative filtering. In: Proceedings of the 12th ACM conference on recommender systems, pp 311–319

Zhou K, Yang SH, Zha H (2011) Functional matrix factorizations for cold-start recommendation. In: Proceedings of the 34th international ACM SIGIR conference on research and development in information retrieval. ACM, pp 315–324