Accuracy-diversity trade-off in recommender systems via graph convolutions

Isufi, Elvin; Pocchiari, Matteo; Hanjalic, Alan

DOI
10.1016/j.ipm.2020.102459

Publication date
2021

Document Version
Final published version

Published in
Information Processing and Management

Information Processing and Management 58 (2021) 102459

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Accuracy-diversity trade-off in recommender systems via graph convolutions

Elvin Isufi *, Matteo Pocchiari, Alan Hanjalic

Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Delft, the Netherlands

ARTICLE INFO

Keywords:
Accuracy-diversity
Collaborative filtering
Graph filters
Graph convolutional neural networks
Graph signal processing

ABSTRACT

Graph convolutions, in both their linear and neural network forms, have reached state-of-the-art accuracy on recommender system (RecSys) benchmarks. However, recommendation accuracy is tied with diversity in a delicate trade-off and the potential of graph convolutions to improve the latter is unexplored. Here, we develop a model that learns joint convolutional representations from a nearest neighbor and a furthest neighbor graph to establish a novel accuracy-diversity trade-off for recommender systems. The nearest neighbor graph connects entities (users or items) based on their similarities and is responsible for improving accuracy, while the furthest neighbor graph connects entities based on their dissimilarities and is responsible for diversifying recommendations. The information between the two convolutional modules is balanced already in the training phase through a regularizer inspired by multi-kernel learning. We evaluate the joint convolutional model on three benchmark datasets with different degrees of sparsity. The proposed method can either trade accuracy to improve substantially the catalog coverage or the diversity within the list; or improve both by a lesser amount. Compared with accuracy-oriented graph convolutional approaches, the proposed model shows diversity gains up to seven times by trading as little as 1% in accuracy. Compared with alternative accuracy-diversity trade-off solutions, the joint graph convolutional model retains the highest accuracy while offering a handle to increase diversity. To our knowledge, this is the first work proposing an accuracy-diversity trade-off with graph convolutions and opens the doors to learning over graphs approaches for improving such trade-off.

1. Introduction

Despite accuracy is still the most dominant criterion guiding the design and evaluation of recommender systems (RecSys), numerous studies have shown that recommendation diversity –decreasing the similarity of the items in the recommended item list– significantly improves user satisfaction (Aggarwal et al., 2016; Bradley & Smyth, 2001; Kaminskas & Bridge, 2016). However, accuracy and diversity do not always go hand in hand and the development of a recommender system to consider both criteria typically requires dealing with an accuracy-diversity trade-off, a.k.a. balance or dilemma Kunaver and Požrl (2017), Wu et al. (2019).

The thin balance between accuracy and diversity is tied with the complexity and irregularity of the user-item relationships. Dealing with this complexity and irregularity has produced creative adaptations of existing RecSys paradigms, such as modifying accuracy-oriented algorithms into diversity-oriented counterparts (Gan & Jiang, 2013; Said, Kille, Jain, & Albayrak, 2012). As an example,

* Corresponding author.
E-mail addresses: e.isufi-1@tudelft.nl (E. Isufi), m.pocchiari@student.tudelft.nl (M. Pocchiari), a.hanjalic@tudelft.nl (A. Hanjalic).

https://doi.org/10.1016/j.ipm.2020.102459
Received 29 July 2020; Received in revised form 7 November 2020; Accepted 3 December 2020
Available online 14 December 2020

the nearest neighbor (NN) collaborative filtering connects entities (users, items) based on pairwise similarities and leverages these connections to interpolate the missing values from proximal entities. To secure accuracy for the target user, the system learns from the preferences of the most similar (nearest) neighboring entities. However, this typically leads to low recommendation diversity, as the NNs are too similar to the target user. In the search for a better accuracy-diversity trade-off, (Said et al., 2012) proposed to look at the furthest neighbors (FNs) instead, i.e., a subset of k users that are most dissimilar to the target-user in terms of preferences. The assumption here is that recommending items FNs disliked most could bring more diversity while preserving an acceptable level of accuracy. Other RecSys algorithms focusing on improving diversity by connecting entities based on their dissimilarity include (Anelli, Di Noia, Di Sciascio, Ragone, & Trotta, 2019; Zeng, Shang, Zhang, Lü, & Zhou, 2010; Zhou et al., 2010). Alternative approaches aiming at trading accuracy with diversity include re-ranking (Adomavicius & Kwon, 2008; Zhang & Hurley, 2008), leveraging side information (Hurley, 2013; Panniello, Tuzhilin, & Gorgoglione, 2014), or merging different models operating with different criteria (Zeng et al., 2010; Zhou et al., 2010).

We believe that in order to obtain sufficient depth in understanding the accuracy-diversity trade-off, RecSys approaches are needed that can fully capture the abovementioned complex and irregular user-item relationships. Graphs have proved themselves as excellent tools to develop such approaches (Ortega, Frossard, Kovačević, Moura, & Vandergheynst, 2018), which made graph-based RecSys one of the most rapidly developing areas. Examples of graph-based RecSys approaches are diffusion-based recommendations (Nikolakopoulos, Berberidis, Karypis, & Giannakis, 2019), random walks (Abbassi & Mirrokni, 2007), and graph neural network-based recommendations (Monti, Bronstein, & Bresson, 2017; Sun et al., 2019; Ying et al., 2018), to name a few.

In parallel to the increasing importance of graphs in the RecSys domain, the signal processing and machine learning communities have developed processing tools for data over graphs (Ortega et al., 2018; Wu et al., 2020). The quintessential tool in these areas is the graph convolution. Graph convolutions extend to graphs the operation of convolution used to process temporal and spatial signals and serve as the building block for graph convolutional neural networks (GCNNs) (Gama, Isufi, Leus, & Ribeiro, 2020). Graph convolutions, both in their linear or GCNN form, have been successfully applied to RecSys reaching state-of-the-art accuracy (Yang, 2019; Ying et al., 2018). Despite the promise, graph convolutions have only been used to over-fit accuracy, leaving unexplored their ability to diversify recommendations and, ultimately, improve the accuracy-diversity trade-off.

In this work, we explore the potential of graph convolutions to improve the accuracy-diversity trade-off for recommender systems. We conduct this exploration by developing a novel model composed of two graph convolutional components, one providing accuracy-oriented recommendations from a NN graph, and one providing diversity-oriented recommendations from a FN graph. Differently from current works, we train a single joint model to fit the data, rather than using two separate models. Our specific contribution in this paper can be summarized as follows:

i) We propose a novel accuracy-diversity trade-off framework for RecSys via graph convolutions. The model operates on a NN graph to improve accuracy and on a FN graph to improve diversity. Each graph can capture user-user or item-item relationships, allowing to also include the hybrid settings, such as a user-NN and an item-FN graph. To the best of our knowledge, this is the first contribution providing an accuracy-diversity trade-off by using these hybrid setups. The proposed model relies only on the available ratings, which we find important since side information, such as metadata or context, can be unavailable or require extensive work to be accurately used.
ii) We develop design strategies that estimate the joint model parameters in view of both accuracy and diversity. These design strategies are versatile to both rating and ranking frameworks. When the joint model is composed of linear graph convolutional filters, we analyze the optimality of the design problem and provide solutions for it.
iii) We analyze the joint model in the graph-spectral domain to provide an alternative interpretation of how the proposed approach balances accuracy with diversity. The joint model presents a band-stop behavior on both the NN and the FN graph, and builds recommendations by focusing on the extremely low and high graph frequencies.
iv) We evaluate two types of trade-offs: i) an accuracy-diversity trade-off w.r.t. catalog coverage (i.e., aggregated diversity), and ii) an accuracy-diversity trade-off w.r.t. list diversity (i.e., individual diversity). The first trade-off shows the models' ability to recommend niche items and personalize recommendations. The second trade-off shows the models' ability to diversify items in the list. The proposed models can either trade accuracy to boost substantially one diversity metric, or improve by a lesser amount.

The remainder of this paper is organized as follows. Section 2 places our contribution in the context of current literature. Section 3 reviews NN collaborative filtering from a graph convolutional learning perspective. Section 4 provides a high-level overview of the proposed approach. Sections 5 and 6 contain the design strategies for rating and ranking, respectively. Section 7 provides the graph-spectral analysis of the joint models, while Section 8 contains the numerical results. Section 9 discusses our findings.

2. Related work

Accuracy-diversity trade-off. Along with the initial work (Smyth & McClave, 2001), also Bridge and Kelly (2006) promotes the accuracy-diversity trade-off as a joint objective for effective RecSys. A popular direction to tweak this trade-off is by two-step approaches, in which re-ranking is applied to a retrieved list to boost diversity (Adomavicius & Kwon, 2009; Zhang & Hurley, 2008). The work in Adomavicius and Kwon (2008) re-ranks items based on rating variance in the neighborhood of a user, while (Adomavicius & Kwon, 2011) uses re-ranking to cover a larger portion of the catalog. Also Eskandanian and Mobasher (2020), Hamedani and Kaedi (2019) diversify items to improve coverage in a user-personalized manner. The work in Hamedani and Kaedi (2019) optimizes the

recommendation list to improve accuracy and diversity but reduce item popularity, while uses matching problems to improve coverage while minimizing the accuracy loss. Instead, Hurley and Zhang (2011) proposes a new metric to quantify diversity within a list and develops an optimization algorithm to improve it. Methods and algorithms in this category rely heavily on the initial recommendation list, which makes it difficult to attribute to which extent the improved trade-off is due to re-ranking or to the properties of the list.

Another category of approaches considers a single algorithm and leverage side information, such as metadata or context, to improve diversity. The work in Hurley (2013) builds an item-item dissimilarity graph from features and uses this graph in a learning-to-rank framework. Also the work in Gogna and Majumdar (2017) uses item features to provide a single method for matrix completion. Differently, Panniello et al. (2014) leverages context and evaluates different pre-filtering, post-filtering, and modeling schemes in terms of accuracy and diversity. Our approach, instead, balances accuracy with diversity without relying on side-information in both a learning-to-rate and learning-to-rank framework.

A third category of approaches modifies conventional accuracy-oriented algorithms to improve diversity. Authors in Liu, Shi, and Guo (2012) build similarities by avoiding the influence of popular objects or high-degree users on the direction of random walk, which is shown heuristically to improve diversity. The work in Gan and Jiang (2013) adjusts the calculation of user similarities in the classic NN approach to improve diversity. A broader analysis following this line is recently presented in Anelli et al. (2019). The work in Wasilewski and Hurley (2016) follows up on Hurley (2013) and uses the regularizer in the learning-to-rank loss to improve diversity. In our view, the latter overloads the regularizer with an additional objective. Since the primary goal of the regularizer is to generalize the model to unseen data, leveraging it also to improve diversity leads to a triple accuracy-diversity-generalization trade-off, which is challenging to handle. Likely, one of the three objectives will be treated as a byproduct, which reduces the possibilities to steer the optimization of the accuracy-diversity trade-off. Differently, Said, Fields, Jain, and Albayrak (2013), Said et al. (2012) connect users based on their dissimilarities and propose the so-called furthest neighbor (FN) collaborative filtering –contrarily to the vanilla NN collaborative filtering. By using information from neighbors that a user disagrees with, this approach was shown to improve diversity by affecting accuracy by little. Yet, the degree to which FNs affect the accuracy-diversity trade-off remains insufficiently investigated. In our approach, we leverage both the NN and the FN in a joint convolutional model to better understand this influence.

While changing the inner-working mechanism of a single model can improve diversity, a single model often lacks the ability to capture the complex relationships contained in highly-sparse RecSys datasets. A fourth category of approaches overcomes this issue by working with an ensemble of models, also referred to as joint or hybrid models. These models have a higher descriptive power that can better balance accuracy with diversity at the expense of complexity, which is often of the same order of magnitude. Authors in Zeng et al. (2010) propose a joint collaborative filtering algorithm that leverages the influence of both similar and dissimilar users. The dissimilarity is computed by counting the items two users have consumed individually but not jointly. The predicted ratings from the similar and dissimilar users are merged into a final score and the influence of each group is controlled by a scalar. The way dissimilarity is computed ignores the fact that users may have consumed the same item, but rated it differently. Also, building dissimilarities from non-consumed items ignores the fact that a user may also like an item other users have consumed separately. To avoid the latter, we account for the ratings when building dissimilarities between entities. The authors of Zhou et al. (2010) follow a similar strategy as Zeng et al. (2010) and mix a heat spreading with a random walk to provide an accuracy-diversity trade-off. A probabilistic model to balance accuracy with diversity is further proposed in Javari and Jalili (2015). The latter considers the order in which items are consumed and proposes a joint model, which on one branch maximizes accuracy, while on the other, diversity. In contrast to this, we train the whole model jointly.

Graph convolutions in RecSys. Graph convolutions have been introduced to the RecSys domain only recently (Berg, Kipf, & Welling, 2017; Huang, Marques, & Ribeiro, 2017; Monti et al., 2017). The approach proposed in Huang et al. (2017), subsequently extended to Huang, Marques, and Ribeiro (2018), showed the NN collaborative filter is a non-parametric graph convolutional filter of order one. This work also showed that higher-orders parametric graph convolutional filters improve rating prediction. These graph convolutional filters are the basis to form GCNNs Gama et al. (2020) and we will use them to balance accuracy with diversity. The work in Monti et al. (2017) merges GCNNs with a recurrent neural network to complete the user-item matrix. Instead, Berg et al. (2017) completes the matrix with a variational graph autoencoder, in which graph convolutions are performed by an order-one graph convolutional filter. The work in Chen, Wu, Hong, Zhang, and Wang (2020) uses the same graph convolution as Huang et al. (2018), but uses it in a learning-to-rank setting. Although starting from different standpoints and naming the method differently, the two approaches are identical from a technical perspective. Taken together, Huang et al. (2018) and Chen et al. (2020) showed that linear graph convolutions may often suffice in highly sparse RecSys datasets. We shall corroborate this behavior also in the accuracy-diversity trade-off setting.

The authors in Sun et al. (2019) deployed a GCNN with filters of order one on an augmented graph comprising the user-item bipartite interaction graph and also user-user and item-item proximal graphs. The work in Wang, He, Wang, Feng, and Chua (2019b) also learns from the user-item bipartite interactions through an order one GCNN, but augments the propagation rule to promote exchanges from similar items. Authors in Ying et al. (2018) combine random walks with graph convolutions to perform recommendations in large-scale systems containing millions of nodes. Authors in Wang et al. (2019a) first build a user-specific knowledge graph and then apply graph neural networks to compute personalized recommendations. They also regularize the loss to enforce similar scores in adjacent items. Lastly, GCNNs have been used for location recommendation in Zhong et al. (2020). Two GCNNs are run over two graphs, a point-of-interest graph and a social relationship graph to identify these points-of-interest for a user.

Altogether, these works show the potential of graph convolutions in changing the RecSys landscape. However, all approaches focus only on accuracy and ignore recommendation diversity. In this work, we consider graph convolutions to establish an accuracy-diversity trade-off in both a learning-to-rate and learning-to-rank setup.

3. Learning from similar nearest neighbors

Consider a recommender system setting comprising a set of users $\mathcal{U} = \{1, \ldots, U\}$ and a set of items $\mathcal{I} = \{1, \ldots, I\}$. Ratings are collected in the user-item matrix $\mathbf{X} \in \mathbb{R}^{U \times I}$, in which entry X_{ui} contains the rating of user u to item i. Ratings are mean-centered[1], so that we can adopt the common convention $X_{ui} = 0$ if value (u, i) is missing. The objective is to populate matrix $\mathbf{X}$ by exploiting the relationships between users and items contained in the available ratings. We capture these relationships through a graph, which is built following the principles of NN collaborative filtering. This graph is used to predict ratings and the k items with the highest predicted rating form the recommendation list.

In user-based NNs, relationships are measured by the Pearson correlation coefficient. Consider a matrix $\mathbf{B} \in \mathbb{R}^{U \times U}$, in which entry B_{uv} measures the correlation between users u and v. Matrix $\mathbf{B}$ is symmetric and can be seen as the adjacency matrix of a *user-correlation* graph $\mathcal{G}_{\mathrm{u}} = (\mathcal{U}, \mathcal{E}_{\mathrm{u}})$. The vertex set of $\mathcal{G}_{\mathrm{u}}$ is the user set $\mathcal{U}$ and the edge set $\mathcal{E}_{\mathrm{u}}$ contains an edge $(u, v) \in \mathcal{E}_{\mathrm{u}}$ only if $B_{uv} \neq 0$. Each item i is treated separately and user ratings are collected in vector $\mathbf{x}^i \in \mathbb{R}^U$ (corresponding to the ith column of $\mathbf{X}$). Vector $\mathbf{x}^i$ can be seen as a signal on the vertices of $\mathcal{G}_{\mathrm{u}}$, which uth entry $[\mathbf{x}^i]_u := X_{ui}$ is the rating of user u to item i, or zero otherwise (Shuman, Narang, Frossard, Ortega, & Vandergheynst, 2013); see Fig. 2 (a). Predicting ratings for item i translates into interpolating the missing values of graph signal $\mathbf{x}^i$. These values are estimated by shifting available ratings to neighboring users. First, we transform the global graph $\mathbf{B}$ to an item-specific graph $\mathbf{B}_i$ which contains only the top-n positively correlated edges per user and normalize their weights; see Fig. 2(b)–(c). The NN shifted ratings to immediate neighbors can be written as

$$\widehat{\mathbf{x}}^i = \mathbf{B}_i \mathbf{x}^i \tag{1}$$

which holds true because matrix $\mathbf{B}_i$ respects the sparsity of the user-graph adapted to item i.

In item-based NNs, the procedure follows likewise. First, we construct an item-item correlation matrix $\mathbf{C} \in \mathbb{R}^{I \times I}$ in which entry C_{ij} is the Pearson correlation coefficient between items i and j. Matrix $\mathbf{C}$ is symmetric and it is the adjacency matrix of an item-correlation graph $\mathcal{G}_{\mathrm{i}} = (\mathcal{I}, \mathcal{E}_{\mathrm{i}})$. The vertex set of $\mathcal{G}_{\mathrm{i}}$ matches the item set $\mathcal{I}$ and the edge set $\mathcal{E}_{\mathrm{i}}$ contains an edge $(i, j) \in \mathcal{E}_{\mathrm{i}}$ only if $C_{ij} \neq 0$. Then, we consider the complementary scenario and treat each user u separately. We collect the ratings of user u to all items in the graph signal $\mathbf{x}_u \in \mathbb{R}^I$ (corresponding to the uth row of $\mathbf{X}$). Finally, item-based NN interpolates the missing values in $\mathbf{x}_u$ through shifts to neighboring items. Building a user-specific graph $\mathbf{C}_u$ from $\mathbf{C}$, keeping only the top-n positively correlated edges per item, and normalizing the weights, we predict the ratings as

$$\widehat{\mathbf{x}}_u = \mathbf{C}_u \mathbf{x}_u. \tag{2}$$

In either scenario, matrices $\mathbf{B}$, $\{\mathbf{B}_i\}_i$, $\mathbf{C}$, and $\{\mathbf{C}_u\}_u$ can be regarded as instances of a general graph adjacency matrix variable $\mathbf{S}$ of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ containing $|\mathcal{V}|$ nodes and $|\mathcal{E}|$ edges. We denote the available rating signal by $\mathbf{x}$ and the estimated rating signal by $\widehat{\mathbf{x}}$ so that we write estimators (1) and (2) with the unified notation

$$\widehat{\mathbf{x}} = \mathbf{S}\mathbf{x}. \tag{3}$$

As it follows from (3), NN estimators rely only on ratings present in the immediate surrounding of a node. But higher-order neighbors carry information that can improve prediction and their information should be accounted for accordingly to avoid destructive interference. Graph convolutional filters have proven themselves to be the tool for capturing effectively multi-resolution neighbor information when learning over graphs (Ortega et al., 2018), including recent success in multi-resolution NN collaborative filtering (Huang et al., 2018; Monti et al., 2017). We detail in the sequel the graph convolutional filter and the respective GCNN extension.

3.1. Nearest neighbor graph convolutional filters

Estimator (3) accounts for the immediate neighbors to predict ratings. Similarly, we can account for the two-hop neighbors via the second-order shift $\mathbf{S}^2\mathbf{x}$. Writing $\mathbf{S}^2\mathbf{x} = \mathbf{S}(\mathbf{S}\mathbf{x})$ shows the second-order shift builds a NN estimator $\mathbf{S}(\cdot)$ on the previous one $\mathbf{S}\mathbf{x}$. We can also consider neighbors up to K-hops away as $\mathbf{S}^K\mathbf{x} = \mathbf{S}(\mathbf{S}^{K-1}\mathbf{x})$. To balance the information coming from the different resolutions $\mathbf{S}\mathbf{x}, \mathbf{S}^2\mathbf{x}, \ldots, \mathbf{S}^K\mathbf{x}$, we consider a set of parameters $\mathbf{h} = [h_0, \ldots, h_K]^\top$ and build the Kth order NN predictor

$$\widehat{\mathbf{x}} = \sum_{k=0}^{K} h_k \mathbf{S}^k \mathbf{x} := \mathbf{H}(\mathbf{S})\mathbf{x} \tag{4}$$

where $\mathbf{H}(\mathbf{S}) = \sum_{k=0}^{K} h_k \mathbf{S}^k$ is referred to as graph convolutional filter of order K (Ortega et al., 2018; Shuman et al., 2013).[2] The ratings $\widehat{\mathbf{x}}$ in (4) are built as a shift-and-sum of the available ratings $\mathbf{x}$. Particularizing $\mathcal{G}$ to $\mathcal{G}_{\mathrm{u}}$, (4) becomes a graph convolutional filter estimator

[1] Mean-centering can be done across users, items, or both. The proposed methods work with any choice.

[2] The term $k = 0$, $\mathbf{S}^0\mathbf{x} = \mathbf{x}$, does not contribute to predicting ratings. We kept it in (4) since this term plays a role in GCNN in Section 3.2. Particularizing the graph to the directed cycle and the signal to a periodic discrete signal, operation (4) reduces to the temporal convolution operation.

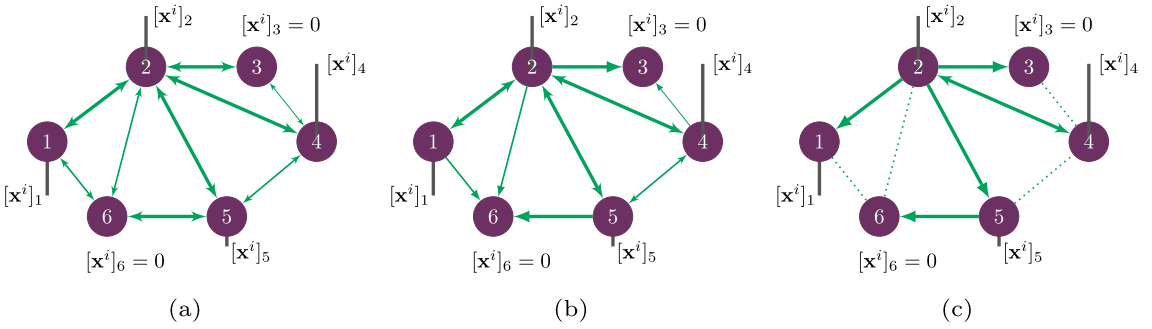

Fig. 1. Building an item-specific graph $\mathbf{B}_i$ from the global correlation graph $\mathbf{B}$. (a) User graph $\mathbf{B}$. Nodes represent users and arrows correlations. Ratings to item i are a graph signal $\mathbf{x}_i$ shown by vertical bars. (b) Treat each undirected edge as two directed edges. Remove any directed edge starting from a user who did not rate item i. (c) Keep the $n = 1$ strongest incoming edge for each user representing the nearest neighbor. The adjacency matrix of this graph is $\mathbf{B}_i$.

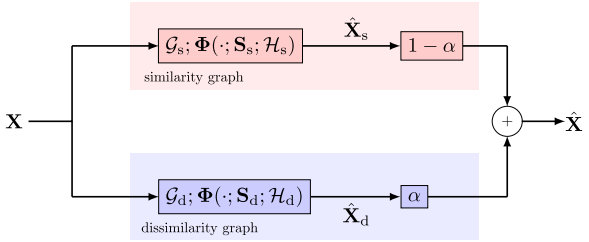

Fig. 2. Rating prediction with similar and dissimilar Graphs. We construct a NN graph $\mathscr{G}_\mathrm{s}$ capturing similarities between entities, and a FN graph $\mathscr{G}_\mathrm{d}$ capturing dissimilarities between entities. On each graph, we run a graph convolutional module $\mathbf{\Phi}(\cdot)$ with respective parameter set $\mathscr{H}_{(\cdot)}$ [cf. (4), (9)]. The estimated outputs are combined through a parameter α to obtain the final joint estimate $\widehat{\mathbf{X}}$.

over the user NN graph with estimated ratings for item i

$$\widehat{\mathbf{x}}^i = \sum_{k=0}^{K} h_k \mathbf{B}_i^k \mathbf{x}^i := \mathbf{H}(\mathbf{B}_i)\mathbf{x}^i. \tag{5}$$

Particularizing $\mathscr{G}$ to $\mathscr{G}_\mathrm{i}$, (4) becomes a graph convolutional filter over the item NN graph with estimated ratings for user u

$$\widehat{\mathbf{x}}_u = \sum_{k=0}^{K} h_k \mathbf{C}_u^k \mathbf{x}_u := \mathbf{H}(\mathbf{C}_u)\mathbf{x}_u. \tag{6}$$

Moreover, the vanilla NN collaborative filters (1) and (2) are the particular case of (5) and (6), respectively, obtained for $K = 1$ and $h_0 = 0$ and $h_1 = 1$.

Graph convolutional filters are defined by the $K+1$ parameters $\mathbf{h}$. Further, since the shift operator matrix $\mathbf{S}$ matches the NN structure, it is sparse, therefore, obtaining the output $\widehat{\mathbf{x}}$ in (4) amounts to a complexity of order $\mathscr{O}(|\mathscr{E}|K)$. These properties are important to deal with scarcity of data and scalability.

3.2. Nearest neighbor graph convolutional neural networks

Besides numerical efficiency, graph convolutional filters have high mathematical tractability and are the building block for graph convolutional neural networks (Gama et al., 2020). To build a GCNN with the filter in (4), consider the composition of a set of L layers. The first layer $\ell = 1$ comprises a bank of F_1 filters $\mathbf{H}_1^f(\mathbf{S})$ each defined by coefficients $\{h_{1k}^f\}_k$. Each of these filters outputs graph signals $\mathbf{u}_1^f = \mathbf{H}_1^f(\mathbf{S})\mathbf{x}$, which are subsequently passed through a pointwise nonlinearity $\sigma(\cdot)$ to produce a collection of F_1 features $\mathbf{x}_1^f$ that constitute the output of layer $\ell = 1$, i.e.,

$$\mathbf{x}_1^f = \sigma\big[\mathbf{u}_1^f\big] = \sigma\big[\mathbf{H}_1^f(\mathbf{S})\mathbf{x}\big] = \sigma\left[\sum_{k=0}^{K} h_{1k}^f \mathbf{S}^k \mathbf{x}\right] \quad \text{for } f = 1, \ldots, F_1. \tag{7}$$

At subsequent intermediate layers $\ell = 2, \ldots, L-1$, the output features $\{\mathbf{x}_\ell^g\}_g$ of the previous layer $\ell - 1$ become inputs to a bank of $F_\ell F_{\ell-1}$ convolutional filters $\mathbf{H}_\ell^{fg}(\mathbf{S})$ each of which outputs the features $\mathbf{u}_\ell^{fg} = \mathbf{H}_\ell^{fg}(\mathbf{S})\mathbf{x}_{\ell-1}^g$. The filter outputs obtained from a common

input $\mathbf{x}_{\ell-1}^{g}$ are aggregated and the result is passed through a nonlinearity $\sigma(\cdot)$ to produce the F_ℓ output features

$$\mathbf{x}_{\ell}^{f}=\sigma\left[\sum_{g=1}^{F}\mathbf{u}_{\ell}^{fg}\right]=\sigma\left[\sum_{g=1}^{F}\mathbf{H}_{\ell}^{fg}(\mathbf{S})\mathbf{x}_{\ell-1}^{g}\right]=\sigma\left[\sum_{g=1}^{F}\sum_{k=0}^{K}h_{\ell k}^{fg}\mathbf{S}^{k}\mathbf{x}_{\ell-1}^{g}\right]\quad\text{for } f=1,\ldots,F_{\ell}. \tag{8}$$

Operation (8) is the propagation rule of a generic layer ℓ of the GCNN, which final outputs are the F_L features $\mathbf{x}_L^1,\ldots,\mathbf{x}_L^{F_L}$. These final convolutional features are passed through a shared multi-layer perceptron per node to map the F_L features per node n, $[x_{Ln}^1;\ldots;x_{Ln}^{F_L}]$, into the output estimate $\widehat{x}_n$.

The GCNN can be seen as a map $\mathbf{\Phi}(\cdot)$ that takes as input a graph signal rating $\mathbf{x}$, an entity-specific graph $\mathbf{S}$, and a set of parameters $\mathscr{H}=\{h_{\ell k}^{fg}\}$ for all layers ℓ, orders k, and feature pairs (f,g). This map produces the estimate

$$\mathbf{\Phi}(\mathbf{x};\mathbf{S};\mathscr{H}):=\widehat{\mathbf{x}}. \tag{9}$$

The GCNN leverages the coupling between the rating and the NN graph in the input layer to learn higher-order representations in the intermediate layers. This coupling is captured by the bank filters as per (4). Consequently, the GCNN inherits the numerical benefits of the graph convolutional filter. Denoting by $F=\max_\ell F_\ell$ the maximum number of features for all layers, the number of parameters defining the GCNN is of order $\mathscr{O}(F^2KL)$ while its computational complexity amounts to $\mathscr{O}(F^2KL|\mathscr{E}|)$. The latter are in the same order of magnitude as for the graph convolutional filter [cf. (4)]. The filter can in fact be viewed as a particular GCNN map $\mathbf{\Phi}(\cdot)$ [cf. (9)] limited to linear aggregations; refer also to Gama et al. (2020) for more details. In the remainder of this paper, we will denote by $\mathbf{\Phi}(\cdot)$ both the filter and the GCNN and refer to them with the common terminology *graph convolutions*.

4. Accounting for disimilar furthest neighbors

We work with a NN similarity-graph $\mathscr{G}_{\mathrm{s}}=\{\mathscr{V},\mathscr{E}_{\mathrm{s}}\}$ [cf. Section 3; Fig. 2] and a FN dissimilarity-graph $\mathscr{G}_{\mathrm{d}}=\{\mathscr{V},\mathscr{E}_{\mathrm{d}}\}$. The dissimilarity graph is built by following the opposite principles of NNs, i.e., connecting each entity to its top-$\overline{n}$ most negatively related ones. To illustrate the latter, consider $\mathbf{C}$ captures item-item correlations while vector $\mathbf{x}_u$ contains the ratings of user u to all items. The user-specific dissimilarity graph has the adjacency matrix $\overline{\mathbf{C}}_u$ obtained from $\mathbf{C}$ by: i) removing any edge starting from an item not rated by user u; ii) keeping for each item i the $\overline{n}$ most negatively connections; iii) normalizing the resulting matrix to make $\overline{\mathbf{C}}_u$ right stochastic. In other words, defining $\overline{\mathscr{N}}_{iu}$ as the set containing the $\overline{n}$ most dissimilar items to i rated by user u, entry (i,j) of $\overline{\mathbf{C}}_u$ is

$$\left[\overline{\mathbf{C}}_u\right]_{ij}=\begin{cases}C_{ij}\Big/\displaystyle\sum_{j'\in\overline{\mathscr{N}}_{iu}}C_{ij'} & \text{if } j\in\overline{\mathscr{N}}_{iu}\\ 0 & \text{if } j\notin\overline{\mathscr{N}}_{iu}\end{cases}. \tag{10}$$

This procedure for building the user-specific FN graphs differs from the NN approach only in step ii). The normalization step ensures a similar magnitude of signal shifting [cf.(3)] on both NN and FN graphs and implies the entries of $\overline{\mathbf{C}}_u$ are positive, i.e., a larger value indicates a stronger dissimilarity. In the considered datasets, positive correlations go up to 1.0 while the negative correlations down to -0.2.

On each graph $\mathscr{G}_{\mathrm{s}}$ and $\mathscr{G}_{\mathrm{d}}$ we have a convolutional module $\mathbf{\Phi}_{\mathrm{s}}(\mathbf{x};\mathbf{S}_{\mathrm{s}};\mathscr{H}_{\mathrm{s}})$ and $\mathbf{\Phi}_{\mathrm{d}}(\mathbf{x};\mathbf{S}_{\mathrm{d}};\mathscr{H}_{\mathrm{d}})$, outputting an estimate of the user-item matrix $\widehat{\mathbf{X}}_{\mathrm{s}}$ and $\widehat{\mathbf{X}}_{\mathrm{d}}$, respectively. We combine the two outputs in the joint estimate

$$\widehat{\mathbf{X}}=(1-\alpha)\widehat{\mathbf{X}}_{\mathrm{s}}+\alpha\widehat{\mathbf{X}}_{\mathrm{d}} \tag{11}$$

where scalar $\alpha\in]0,1[$ balances the influence of similar and dissimilar connections; Fig. 2. Each graph $\mathscr{G}_{\mathrm{s}}$ or $\mathscr{G}_{\mathrm{d}}$ can be a user or an item graph and the graph convolutional modules $\mathbf{\Phi}(\cdot)$ can be linear [cf. (4)] or nonlinear [cf. (9)]. This framework yields eight combinations to investigate the trade-off. We limit ourselves to situations where the graph convolutional modules are the same on both graphs and focus on the four combinations in Table 1. To ease exposition, we shall discuss the theoretical methods with the hybrid combination user NN graph (i.e., $\mathscr{G}_{\mathrm{s,u}}$ with adjacency matrix $\mathbf{B}_i$ for item i) and item FN graph (i.e., $\mathscr{G}_{\mathrm{d,i}}$ with adjacency matrix $\overline{\mathbf{C}}_u$ for user u). This setting implies we predict rating $\widehat{X}_{ui}$ by learning, on one side, from the coupling $(\mathscr{G}_{\mathrm{s,u}},\mathbf{x}^i)$, and, on the other side, from the coupling $(\mathscr{G}_{\mathrm{d,i}},\mathbf{x}_u)$.

Joint models like the one we consider are popular beyond the RecSys literature. The works in Hua et al. (2019), Sevi, Rilling, and Borgnat (2018) consider two different shift operators of the same graph to model signal diffusion with graph convolutional filters [cf. (4)]. This strategy is subsequently extended to GCNNs in Dehmamy, Barabási, and Yu (2019). Instead, Chen, Niu, Lan, and Liu (2019), Ioannidis, Marques, and Giannakis (2019) exploit different relationships between data to build GCNNs. The common motivation in all these works is that a model based on a single graph (often capturing similarities between nodes (Mateos, Segarra, Marques, & Ribeiro, 2019)) or a single shift operator is insufficient to represent the underlying relationships. Therefore, we argue a joint model capturing different interactions helps representing better the data. A model based only on NNs fails giving importance to items that differ from the main trend. FNs account for this information and aid diversity. However, the information from FNs should be accounted for properly during training to keep the accuracy at the desired level. We detail this aspect in the upcoming two sections.

Table 1
Combinations between the similarity-graph $\mathcal{G}_{\text{s}}$ and the dissimilarity-graph $\mathcal{G}_{\text{d}}$.

Dissimilar $\mathcal{G}_{\text{d}}$			
		User	Item
Similar $\mathcal{G}_{\text{s}}$	User	User-user (UU)	User-item (UI)
	Item	Item-user (IU)	Item-item (II)

5. Learning for rating

In this section, we estimate the joint model parameters w.r.t. the mean squared error (MSE) criterion. Analyzing the MSE quantifies also the trade-off for all items in the dataset (unbiasing the results from the user preferences in the list).[3] The MSE also provides insights into the role played by the FNs. To this end, consider a training set of user-item pairs $\mathcal{T} = \{(u,i)\}$ for the available ratings in $\mathbf{X}$. Consider also the user-similarity graph $\mathcal{G}_{\text{s,u}}$, the item-dissimilarity graph $\mathcal{G}_{\text{d,i}}$, and their respective graph convolutions $\mathbf{\Phi}_{\text{s}}(\mathbf{x}^i; \mathbf{B}_i; \mathscr{H}_{\text{s}})$ and $\mathbf{\Phi}_{\text{d}}(\mathbf{x}_u; \overline{\mathbf{C}}_u; \mathscr{H}_{\text{d}})$. We estimate parameters $\mathscr{H}_{\text{s}}$ and $\mathscr{H}_{\text{d}}$ by solving the regularized problem

$$\begin{aligned} \underset{\mathscr{H}_{\text{s}}, \mathscr{H}_{\text{d}}}{\text{minimize}} \quad & \frac{1}{\mu}\text{MSE}_{(u,i)\in\mathcal{T}}\left(\mathbf{\Phi}_{\text{s}}\left(\mathbf{x}^i; \mathbf{B}_i; \mathscr{H}_{\text{s}}\right) + \mathbf{\Phi}_{\text{d}}\left(\mathbf{x}_u; \overline{\mathbf{C}}_u; \mathscr{H}_{\text{d}}\right); \mathbf{X}_{\mathcal{T}}\right) + \frac{1}{2}\left(\frac{\|\mathscr{H}_{\text{s}}\|_2^2}{1-\alpha} + \frac{\|\mathscr{H}_{\text{d}}\|_2^2}{\alpha}\right) \\ \text{subject to} \quad & 0 < \alpha < 1 \end{aligned} \tag{12}$$

where $\text{MSE}_{(u,i)\in\mathcal{T}}(\cdot;\cdot)$ measures the fitting error w.r.t. the available ratings $\mathbf{X}_{\mathcal{T}}$, while the second term acts as an accuracy-diversity regularizer.[4]

Scalar α controls the information flow from the NN and the FN graph. For $\alpha \to 0$, we have $\|\mathscr{H}_{\text{d}}\|_2^2/\alpha \gg \|\mathscr{H}_{\text{s}}\|_2^2/(1-\alpha)$, therefore, problem (12) forces parameters $\mathscr{H}_{\text{d}}$ to a smaller norm rather than using them to fit the data. Hence, this setting mainly leverages information from the similar graph $\mathbf{\Phi}_{\text{s}}(\cdot)$, ultimately, reducing the ensemble to an accuracy-oriented NN graph convolutional model. For $\alpha \to 1$, we have the opposite case $\|\mathscr{H}_{\text{s}}\|_2^2/(1-\alpha) \gg \|\mathscr{H}_{\text{d}}\|_2^2/\alpha$, which implies the information from the similar graph plays little role in fitting since parameters $\mathscr{H}_{\text{s}}$ are forced towards zero. Hence, problem (12) mainly exploits information from FNs to reduce the MSE. Intermediate values of α closer to zero than to one lead to models where most information is leveraged from the NNs to keep the MSE low, while some information is taken from FNs to add diversity. We refer to α as the trade-off parameter. Scalar μ balances the fitting error with the overall regularization and allows generalizing the model to unseen data.

5.1. Graph convolutional filter

Recall the graph convolutional filter in (4) and consider graphs $\mathcal{G}_{\text{s,u}}$ and $\mathcal{G}_{\text{d,i}}$ can have different number of nodes. To account for this technicality in the design phase, we first transform the filters into a more manageable form. The filter output on the user-similarity graph $\mathbf{B}_i$ can be written as

$$\mathbf{\Phi}_{\text{s}}\left(\mathbf{x}^i; \mathbf{B}_i; \mathbf{h}_{\text{s}}\right) = \sum_{k=0}^{K} h_{s,k}\mathbf{B}_i^k\mathbf{x}^i := \mathbf{B}_{\text{s},i}\mathbf{h}_{\text{s}} \tag{13}$$

where the $U \times (K+1)$ matrix $\mathbf{B}_{\text{s},i} = [\mathbf{B}_i^0\mathbf{x}^i, \ldots, \mathbf{B}_i^{K_s}\mathbf{x}^i]$ contains the shifted ratings of item i over graph $\mathbf{B}_i$ and vector $\mathbf{h}_{\text{s}} = [h_{s,0}, \ldots, h_{s,K_s}]^\top$ the parameters. The uth row of $\mathbf{B}_{\text{s},i}$ is the $1 \times (K+1)$ vector $[\mathbf{B}_{\text{s},i}]_{u:}$ containing the shifted ratings of user u for item i. We then stack the $|\mathcal{T}|$ row vectors $[\mathbf{B}_{\text{s},i}]_{u:}$ for all pairs $(u,i) \in \mathcal{T}$ in

$$\mathbf{M}_{\text{s}} = \left[\ldots; \left[\mathbf{B}_{\text{s},i}\right]_{u:}; \left[\mathbf{B}_{\text{s},j}\right]_{u:}; \ldots; \left[\mathbf{B}_{\text{s},k}\right]_{v:}; \left[\mathbf{B}_{\text{s},l}\right]_{v:}; \ldots\right] \in \mathbb{R}^{|\mathcal{T}|\times(K+1)}.$$

The τth row of $\mathbf{M}_{\text{s}}$ corresponds to the τth (u,i) tuple. Denoting by $\mathbf{x}_{\mathcal{T}} = \text{vec}(\mathbf{X}_{\mathcal{T}})$ the $|\mathcal{T}| \times 1$ vector of available ratings, we can write the filter output for all training samples as $\widehat{\mathbf{x}}_{\text{s},\mathcal{T}} = \mathbf{M}_{\text{s}}\mathbf{h}_{\text{s}}$. Likewise, we can write the filter output over the item-dissimilarity graph $\overline{\mathbf{C}}_u$ as

$$\mathbf{\Phi}_{\text{d}}\left(\mathbf{x}_u; \overline{\mathbf{C}}_u; \mathbf{h}_{\text{d}}\right) = \sum_{k=0}^{K} h_{d,k}\overline{\mathbf{C}}_u^k\mathbf{x}_u := \overline{\mathbf{C}}_{\text{d},u}\mathbf{h}_{\text{d}} \tag{14}$$

where matrix $\overline{\mathbf{C}}_{\text{d},u} = [\overline{\mathbf{C}}_u^0\mathbf{x}_u, \ldots, \overline{\mathbf{C}}_u^{K_d}\mathbf{x}_u] \in \mathbb{R}^{I\times(K+1)}$ collects the shifted ratings of user u w.r.t. graph $\overline{\mathbf{C}}_u$ and vector $\mathbf{h}_{\text{d}} = [h_{d,0}, \ldots, h_{d,K}]^\top$ the filter parameters. Then, we construct the $|\mathcal{T}| \times (K+1)$ matrix $\mathbf{M}_{\text{d}}$ by collecting the rows $[\overline{\mathbf{C}}_{\text{d},u}]_{i:}$ $\forall$ $(u,i) \in \mathcal{T}$ so that to write $\widehat{\mathbf{x}}_{\text{d},\mathcal{T}} =$

[3] We shall see that often methods trained with the MSE perform better in the list w.r.t. trained for ranking.

[4] In (12), we allowed ourselves a slight abuse of notation and indicated with $\|\mathscr{H}_{(\cdot)}\|_2^2$ the ℓ_2- norm squared of the vector containing the coefficients in set $\mathscr{H}_{(\cdot)}$.

$\mathbf{M}_{\mathrm{d}}\mathbf{h}_{\mathrm{d}}$.

With these in place, the design problem (12) particularizes to

$$\begin{aligned} \underset{\mathbf{h}_{\mathrm{s}},\mathbf{h}_{\mathrm{d}}}{\text{minimize}} \quad & \frac{1}{2\mu}\|\mathbf{x}_{\mathscr{T}}-\mathbf{M}_{\mathrm{s}}\mathbf{h}_{\mathrm{s}}-\mathbf{M}_{\mathrm{d}}\mathbf{h}_{\mathrm{d}}\|_2^2+\frac{1}{2}\left(\frac{\|\mathbf{h}_{\mathrm{s}}\|_2^2}{1-\alpha}+\frac{\|\mathbf{h}_{\mathrm{d}}\|_2^2}{\alpha}\right) \\ \text{subject to} \quad & 0<\alpha<1 \end{aligned} \tag{15}$$

which is a regularized-least squares problem in the filter coefficients $\mathbf{h}_{\mathrm{s}}$ and $\mathbf{h}_{\mathrm{d}}$. The closed-form solution for (15) can be found by setting the gradient to zero, i.e.,

$$-\frac{1}{\mu}\mathbf{M}_{\mathrm{s}}^{\top}\left(\mathbf{x}_{\mathscr{T}}-\mathbf{M}_{\mathrm{s}}\mathbf{h}_{\mathrm{s}}-\mathbf{M}_{\mathrm{d}}\mathbf{h}_{\mathrm{d}}\right)+\frac{1}{1-\alpha}\mathbf{h}_{\mathrm{s}}=\mathbf{0} \tag{16a}$$

$$-\frac{1}{\mu}\mathbf{M}_{\mathrm{d}}^{\top}\left(\mathbf{x}_{\mathscr{T}}-\mathbf{M}_{\mathrm{s}}\mathbf{h}_{\mathrm{s}}-\mathbf{M}_{\mathrm{d}}\mathbf{h}_{\mathrm{d}}\right)+\frac{1}{\alpha}\mathbf{h}_{\mathrm{d}}=\mathbf{0} \tag{16b}$$

or equivalently solving the linear system of equations

$$\frac{1}{\mu}\begin{bmatrix}\mathbf{M}_{\mathrm{s}}^{\top}\mathbf{x}_{\mathscr{T}}\\ \mathbf{M}_{\mathrm{d}}^{\top}\mathbf{x}_{\mathscr{T}}\end{bmatrix}=\begin{bmatrix}\mathbf{M}_{\mathrm{s}}^{\top}\mathbf{M}_{\mathrm{s}}-\frac{1}{1-\alpha}\mathbf{I} & \mathbf{M}_{\mathrm{s}}^{\top}\mathbf{M}_{\mathrm{d}}\\ \mathbf{M}_{\mathrm{d}}^{\top}\mathbf{M}_{\mathrm{s}} & \mathbf{M}_{\mathrm{d}}^{\top}\mathbf{M}_{\mathrm{d}}-\frac{1}{\alpha}\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{h}_{\mathrm{s}}\\ \mathbf{h}_{\mathrm{d}}\end{bmatrix}. \tag{17}$$

If the matrix inversion in (17) is ill-conditioned, we can always solve (15) with of-the-shelf iterative methods. The above procedure leads to an optimal balance between the information coming from the NNs and the FNs.

5.2. Graph convolutional neural network

We now consider models $\mathbf{\Phi}_{\mathrm{s}}(\mathbf{x}^i;\mathbf{B}_i;\mathscr{H}_{\mathrm{s}})$ and $\mathbf{\Phi}_{\mathrm{d}}(\mathbf{x}_u;\overline{\mathbf{C}}_u;\mathscr{H}_{\mathrm{d}})$ are GCNNs running respectively over graphs $\mathbf{B}_i$ and $\overline{\mathbf{C}}_u$. Particularizing (12) to this setting implies solving

$$\begin{aligned} \underset{\mathscr{H}_{\mathrm{s}},\mathscr{H}_{\mathrm{d}}}{\text{minimize}} \quad & \frac{1}{2\mu}\sum_{(u,i)\in\mathscr{T}}\left|X_{ui}-\left[\mathbf{\Phi}_{\mathrm{s}}\left(\mathbf{x}^i;\mathbf{B}_i;\mathscr{H}_{\mathrm{s}}\right)\right]_u-\left[\mathbf{\Phi}_{\mathrm{d}}\left(\mathbf{x}_u;\overline{\mathbf{C}}_u;\mathscr{H}_{\mathrm{d}}\right)\right]_i\right|^2+\frac{1}{2}\left(\frac{\|\mathscr{H}_{\mathrm{s}}\|_2^2}{1-\alpha}+\frac{\|\mathscr{H}_{\mathrm{d}}\|_2^2}{\alpha}\right) \\ \text{subject to} \quad & 0<\alpha<1 \end{aligned} \tag{18}$$

where $[\mathbf{\Phi}_{\mathrm{s}}(\mathbf{x}^i;\mathbf{B}_i;\mathscr{H}_{\mathrm{s}})]_u$ is the user-similarity GCNN output for user u and $[\mathbf{\Phi}_{\mathrm{d}}(\mathbf{x}_u;\overline{\mathbf{C}}_u;\mathscr{H}_{\mathrm{d}})]_i$ is the item-dissimilarity GCNN output for item i. Problem (18) preserves the trade-offs of the general version (12), but it is non-convex and little can be said about its global optimality. However, because of the compositional form of the GCNN, we can estimate parameters $\mathscr{H}_{\mathrm{s}}$ and $\mathscr{H}_{\mathrm{d}}$ via standard back-propagation since the graph convolutional filters are linear operators in the respective parameters (Goodfellow, Bengio, & Courville, 2016). The following remark is in order.

Remark 1. In (18), we considered the accuracy-diversity parameter α only in the regularizer and not also in the fitting part as in (11). We found that including the latter to the MSE term leads to a more conservative solution towards diversity. We have consistently seen that keeping α only in the regularizer allows for a better trade-off. Furthermore, the regulariser in (18) does not need be rational in α, but can be in any form as long as it balances the NNs with the FNs. An alternative is $\Omega(\mathscr{H}_{\mathrm{s}};\mathscr{H}_{\mathrm{d}};\alpha)=\frac{1}{2}(\alpha\|\mathscr{H}_{\mathrm{s}}\|_2^2+(1-\alpha)\|\mathscr{H}_{\mathrm{d}}\|_2^2)$.

6. Learning for ranking

This section designs the joint model for ranking. We considered the Bayesian personalized ranking (BPR), which is a state-of-the-art learn-to-ranking framework (Rendle, Freudenthaler, Gantner, & Schmidt-Thieme, 2009). BPR considers the rating difference a user u has given to two items i and j. Let symbol $i\succ_u j$ indicate user u rated item i more than item j and augment the training set as $\mathscr{T}\subseteq\mathscr{U}\times\mathscr{I}\times\mathscr{I}$ to contain triples of the form $\mathscr{T}=\{(u,i,j)|i\succ_u j\}$. For each available tuple (u,i) we created four triples $\{(u,i,j)\}_j$ such that $X_{ui}>X_{uj}$ following (Rendle et al., 2009). Subsequently, the estimated ratings for tuples (u,i) and (u,j) are respectively

$$\begin{aligned} \widehat{X}_{ui}(\mathscr{H}_{\mathrm{s}},\mathscr{H}_{\mathrm{d}}) &= \left[\mathbf{\Phi}_{\mathrm{s}}\left(\mathbf{x}^i;\mathbf{B}_i;\mathscr{H}_{\mathrm{s}}\right)\right]_u+\left[\mathbf{\Phi}_{\mathrm{d}}\left(\mathbf{x}_u;\overline{\mathbf{C}}_u;\mathscr{H}_{\mathrm{d}}\right)\right]_i \\ \widehat{X}_{uj}(\mathscr{H}_{\mathrm{s}},\mathscr{H}_{\mathrm{d}}) &= \left[\mathbf{\Phi}_{\mathrm{s}}\left(\mathbf{x}^j;\mathbf{B}_j;\mathscr{H}_{\mathrm{s}}\right)\right]_u+\left[\mathbf{\Phi}_{\mathrm{d}}\left(\mathbf{x}_u;\overline{\mathbf{C}}_u;\mathscr{H}_{\mathrm{d}}\right)\right]_j \end{aligned} \tag{19}$$

and the utility function is

$$\widehat{X}_{uij}(\mathscr{H}_{\mathrm{s}},\mathscr{H}_{\mathrm{d}})=\widehat{X}_{ui}(\mathscr{H}_{\mathrm{s}},\mathscr{H}_{\mathrm{d}})-\widehat{X}_{uj}(\mathscr{H}_{\mathrm{s}},\mathscr{H}_{\mathrm{d}}) \tag{20}$$

which expresses the rating difference as a parametric relationship between user u, item i, and item j. The utility function is used to estimate parameters $\mathscr{H}_{\mathrm{s}}, \mathscr{H}_{\mathrm{d}}$ by maximizing the likelihood

$$p(i \succ_u j | \mathscr{H}_{\mathrm{s}}, \mathscr{H}_{\mathrm{d}}) := \sigma\left(\widehat{X}_{uij}(\mathscr{H}_{\mathrm{s}}, \mathscr{H}_{\mathrm{d}})\right) = \left(1 + e^{-\widehat{X}_{uij}(\mathscr{H}_{\mathrm{s}}, \mathscr{H}_{\mathrm{d}})}\right)^{-1} \tag{21}$$

where $\sigma(x) = (1 + e^{-x})^{-1}$ is the logistic sigmoid function (Rendle et al., 2009). By applying the natural logarithm (monotonic increasing) to (21) and regularizing it, we can estimate the joint convolutional model parameters by solving the regularized optimization problem

$$\begin{aligned} \underset{\mathscr{H}_{\mathrm{s}}, \mathscr{H}_{\mathrm{d}}}{\text{minimize}} \quad & -\frac{1}{\mu} \sum_{(u,i,j) \in \mathscr{T}} \ln \sigma\left(\widehat{X}_{uij} \mathscr{H}_{\mathrm{s}}, \mathscr{H}_{\mathrm{d}}\right) + \alpha \left\| \mathscr{H}_{\mathrm{s}} \right\|_2^2 + (1-\alpha) \left\| \mathscr{H}_{\mathrm{d}} \right\|_2^2 \\ \text{subject to} \quad & 0 < \alpha < 1 \end{aligned} . \tag{22}$$

Differently from (5), the regularizer in (22) is linear in α. We opted for this choice because the linear was more robust to μ. Nevertheless, the regulariser in (22) respects the same trend as that in (5): for $\alpha \to 0$, NNs are mainly used for fitting since $\alpha \left\| \mathscr{H}_{\mathrm{s}} \right\|_2^2 \to 0$; vice-versa, for $\alpha \to 1$ the FNs are mainly used for fitting since $(1-\alpha) \left\| \mathscr{H}_{\mathrm{d}} \right\|_2^2 \to 0$.

6.1. Graph convolutional filter

Particularizing the convolutional models to filters [cf. (4)], (19) becomes

$$\begin{aligned} \widehat{X}_{ui}(\mathscr{H}_{\mathrm{s}}, \mathscr{H}_{\mathrm{d}}) &= \left[\mathbf{B}_{\mathrm{s},i}\right]_{u:} \mathbf{h}_{\mathrm{s}} + \left[\overline{\mathbf{C}}_{\mathrm{d},u}\right]_{i:} \mathbf{h}_{\mathrm{d}} \\ \widehat{X}_{uj}(\mathscr{H}_{\mathrm{s}}, \mathscr{H}_{\mathrm{d}}) &= \left[\mathbf{B}_{\mathrm{s},j}\right]_{u:} \mathbf{h}_{\mathrm{s}} + \left[\overline{\mathbf{C}}_{\mathrm{d},u}\right]_{j:} \mathbf{h}_{\mathrm{d}} \end{aligned} \tag{23}$$

where $[\mathbf{B}_{\mathrm{s},i}]_{u:}$ is the uth row of the similar user-NN graph matrix $\mathbf{B}_{\mathrm{s},i}$ [cf. (13)] and $[\overline{\mathbf{C}}_{\mathrm{d},u}]_{i:}$ is the ith row of the dissimilar item-FN graph matrix $\overline{\mathbf{C}}_{\mathrm{d},u}$ [cf. (14)]. Substituting (23) into (22) leads to

$$\begin{aligned} \underset{\mathbf{h}_{\mathrm{s}}, \mathbf{h}_{\mathrm{d}}}{\text{minimize}} \quad & -\frac{1}{\mu} \sum_{(u,i,j) \in \mathscr{T}} \ln \sigma\left(\widehat{X}_{uij}\left(\mathbf{h}_{\mathrm{s}}, \mathbf{h}_{\mathrm{d}}\right) + \left(\alpha \left\|\mathbf{h}_{\mathrm{s}}\right\|_2^2 + (1-\alpha) \left\| \mathbf{h}_{\mathrm{d}} \right\|_2^2\right)\right. \\ \text{subject to} \quad & \widehat{X}_{uij}(\mathbf{h}_{\mathrm{s}}, \mathbf{h}_{\mathrm{d}}) = \left(\left[\mathbf{B}_{\mathrm{s},i}\right]_{u:} \mathbf{h}_{\mathrm{s}} + \left[\overline{\mathbf{C}}_{\mathrm{d},u}\right]_{i:} \mathbf{h}_{\mathrm{d}}\right) - \left(\left[\mathbf{B}_{\mathrm{s},j}\right]_{u:} \mathbf{h}_{\mathrm{s}} + \left[\overline{\mathbf{C}}_{\mathrm{d},u}\right]_{j:} \mathbf{h}_{\mathrm{d}}\right) \\ & 0 < \alpha < 1 \end{aligned} . \tag{24}$$

Function $-\ln\sigma(\widehat{X}_{uij}(\mathbf{h}_{\mathrm{s}}, \mathbf{h}_{\mathrm{d}}))$ is convex since it involves a log-sum-exp of an affine function Boyd, Boyd, and Vandenberghe (2004). Consequently, problem (24) is convex in $\mathbf{h}_{\mathrm{s}}$ and $\mathbf{h}_{\mathrm{d}}$. Convexity guarantees we can find a minimizer for (24) but not a closed-form solution. In fact, finding an analytical solution for logistic fitting problems is notoriously difficult except for particular instances (Lipovetsky, 2015). However, we can get the optimal parameters for (24) through the stochastic gradient descent updates

$$\mathbf{h}_{\mathrm{s}} \leftarrow \mathbf{h}_{\mathrm{s}} + \frac{\gamma}{\mu}\left[e^{-\widehat{X}_{uij}(\mathbf{h}_{\mathrm{s}}, \mathbf{h}_{\mathrm{d}})} \sigma\left(\widehat{X}_{uij}\left(\mathbf{h}_{\mathrm{s}}, \mathbf{h}_{\mathrm{d}}\right)\right)\left(\left[\mathbf{B}_{\mathrm{s},i}\right]_{u}^{\top} - \left[\mathbf{B}_{\mathrm{s},j}\right]_{u}^{\top}\right) - 2\alpha \mathbf{h}_{\mathrm{s}}\right] \tag{25a}$$

$$\mathbf{h}_{\mathrm{d}} \leftarrow \mathbf{h}_{\mathrm{d}} + \frac{\gamma}{\mu}\left[e^{-\widehat{X}_{uij}(\mathbf{h}_{\mathrm{s}}, \mathbf{h}_{\mathrm{d}})} \sigma\left(\widehat{X}_{uij}(\mathbf{h}_{\mathrm{s}}, \mathbf{h}_{\mathrm{d}})\right)\left(\left[\overline{\mathbf{C}}_{\mathrm{d},u}\right]_{i}^{\top} - \left[\overline{\mathbf{C}}_{\mathrm{d},u}\right]_{j}^{\top}\right) - 2(1-\alpha)\mathbf{h}_{\mathrm{d}}\right] \tag{25b}$$

where γ is the stepsize. These optimal parameters guarantee the best learning-to-rank solution for any balance between the NNs and FNs (α) and between fitting and generalization (μ).

6.2. Graph convolutional neural network

When $\mathbf{\Phi}_{\mathrm{s}}(\mathbf{x}^i; \mathbf{B}_i; \mathscr{H}_{\mathrm{s}})$ and $\mathbf{\Phi}_{\mathrm{d}}(\mathbf{x}_u; \overline{\mathbf{C}}_u; \mathscr{H}_{\mathrm{d}})$ are GCNNs, the BPR optimization problem is that in (22). Because of the nonlinearity, it is difficult to establish if a global minimum exists and we should seek for a satisfactory local minimum. Since cost (22) is differentiable w.r.t. $\mathscr{H}_{\mathrm{s}}$ and $\mathscr{H}_{\mathrm{d}}$, we can achieve this local minimum through conventional backpropagation.

Either estimated for rating or ranking, the coefficients of the joint model dictate the filter behavior (either directly or within the GCNN layers) on the NN and FN graphs. Besides analyzing the filter behavior in the node domain (as multi-hop rating aggregation) and in the respective cost functions (as accuracy-diversity trade-off), we can also get insight on the trade-off by analyzing the graph convolutional modules in the graph spectral domain (Ortega et al., 2018). We discuss this aspect next.

7. Spectral explanation

We conduct here a spectral analysis of graph convolutions to show they act as band-stop filters on both NN and FN graphs. First, we recall the concept of Fourier transform for signals on directed graphs (Sandryhaila & Moura, 2014). Assuming the shift operator $\mathbf{S}$ is diagonalizable, we can write $\mathbf{S} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}$ with eigenvector matrix $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_N]$ and complex eigenvalues $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_N)$. The graph Fourier transform (GFT) of signal $\mathbf{x}$ is

$$\tilde{\mathbf{x}} = \mathbf{U}^{-1}\mathbf{x}. \tag{26}$$

The ith GFT coefficient $\tilde{x}_i$ of $\tilde{\mathbf{x}}$ quantifies the contribution of the ith eigenvector $\mathbf{u}_i$ to expressing the variability of $\mathbf{x}$ over the graph. The latter is analogous to the discrete Fourier transform for temporal or spatial signals. In this analogy, the complex eigenvalues $\lambda_i \in \boldsymbol{\Lambda}$ are referred to as the graph frequencies (Sandryhaila & Moura, 2013; Shuman et al., 2013). The inverse transform is $\mathbf{x} = \mathbf{U}\tilde{\mathbf{x}}$.

To measure the graph signal variability, we follow Sandryhaila and Moura (2014) and order the graph frequencies λ_i based on their distance from the maximum eigenvalue $\lambda_{\max}(\mathbf{S})$. This ordering is based on the notion of total variation (TV), which for the eigenpair $(\lambda_i, \mathbf{u}_i)$ is defined as

$$\text{TV}(\mathbf{u}_i) = \left|1 - \frac{\lambda_i}{\lambda_{\max}(\mathbf{S})}\right| \| \mathbf{u}_n \|_1 \tag{27}$$

where $\| \cdot \|_1$ is the ℓ_1-norm. The closer λ_i to the maximum eigenvalue $\lambda_{\max}(\mathbf{S})$, the smoother the corresponding eigenvector $\mathbf{u}_i$ over the graph (i.e., values on neighboring nodes are similar). If signal $\mathbf{x}$ changes little (e.g., similar users have similar ratings), the corresponding GFT $\tilde{\mathbf{x}}$ has nonzero entries mostly in entries $\tilde{x}_i$ which index i corresponds to a low graph frequency $\lambda_i \to \lambda_{\max}(\mathbf{S})$ (low TV). Contrarily, if signal $\mathbf{x}$ varies substantially in connected nodes, the GFT $\tilde{\mathbf{x}}$ has nonzero values also in entries $\tilde{x}_i$ which index i corresponds to a high graph frequency $\lambda_i \gg \lambda_{\max}(\mathbf{S})$ (high TV); refer to Ortega et al. (2018), Sandryhaila and Moura (2014) for further detail.

With this analogy in place, we substitute the eigendecomposition $\mathbf{S} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}$ into the graph convolutional filter (4) and obtain the filter input-output relationship in the spectral domain

$$\tilde{\hat{\mathbf{x}}} = \sum_{k=0}^{K} h_k \boldsymbol{\Lambda}^k \tilde{\mathbf{x}} := \mathbf{H}(\boldsymbol{\Lambda})\tilde{\mathbf{x}} \tag{28}$$

where $\tilde{\hat{\mathbf{x}}} = \mathbf{U}^{-1}\hat{\mathbf{x}}$ is the GFT of the output and $\mathbf{H}(\boldsymbol{\Lambda}) = \sum_{k=0}^{K} h_k \boldsymbol{\Lambda}^k$ contains the filter frequency response on the main diagonal. Relation (28) shows in first place graph convolutional filters respect the convolutional theorem because they act as a pointwise multiplication between the filter transfer function $\mathbf{H}(\boldsymbol{\Lambda})$ and the input GFT $\tilde{\mathbf{x}}$. Therefore, analyzing $\mathbf{H}(\boldsymbol{\Lambda})$ shows how the filter processes the input ratings $\mathbf{x}$ to estimate $\tilde{\hat{\mathbf{x}}}$. We evaluate the frequency responses of the filter and respective GCNN when deployed on the similar NN and the dissimilar FN graphs for the MovieLens 100K dataset. The latter allows for a direct comparison with the vanilla NN and the graph convolutional NN filter (Huang et al., 2018).

7.1. Graph convolutional filters

Recall the joint model with user-NN filter $\mathbf{H}(\mathbf{B}_i) = \sum_{k=0}^{K} h_{s,k} \mathbf{B}_i^k \mathbf{x}^i$ [cf. (13)] and an item-FN filter $\mathbf{H}(\overline{\mathbf{C}}_u) = \sum_{k=0}^{K} h_{d,k} \overline{\mathbf{C}}_u^k \mathbf{x}_u$ [cf. (14)]. Substituting the eigendecompositions $\mathbf{B}_i = \mathbf{U}_{s,i}\boldsymbol{\Lambda}_{s,i}\mathbf{U}_{s,i}^{-1}$ and $\overline{\mathbf{C}}_u = \overline{\mathbf{U}}_{d,u}\overline{\boldsymbol{\Lambda}}_{d,u}\overline{\mathbf{U}}_{d,u}^{-1}$, we can write the outputs in the graph frequency domain respectively as

$$\tilde{\hat{\mathbf{x}}}^i = \sum_{k=0}^{K} h_{s,k} \boldsymbol{\Lambda}_{s,i}^k \tilde{\mathbf{x}}^i \quad \text{and} \quad \tilde{\hat{\mathbf{x}}}_u = \sum_{k=0}^{K} h_{d,k} \overline{\boldsymbol{\Lambda}}_{d,u}^k \tilde{\mathbf{x}}_u. \tag{29}$$

In (29), $\mathbf{H}(\boldsymbol{\Lambda}_{s,i}) := \sum_{k=0}^{K} h_{s,k} \boldsymbol{\Lambda}_{s,i}^k$ and $\mathbf{H}(\overline{\boldsymbol{\Lambda}}_{d,u}) := \sum_{k=0}^{K} h_{d,k} \overline{\boldsymbol{\Lambda}}_{d,u}^k$ are the responses of filters $\mathbf{H}(\mathbf{B}_i)$ and $\mathbf{H}(\overline{\mathbf{C}}_u)$, respectively. To estimate the responses, we first get the parameters from (15) for rating or (22) for ranking and order the eigenvalues $\lambda_{n,i}$ (resp. $\lambda_{n,u}$) of each $\mathbf{B}_i$ (resp. $\overline{\mathbf{C}}_u$) as per the total variation in (27). Subsequently, for each $\mathbf{B}_i$ (resp. $\overline{\mathbf{C}}_u$) we record the frequency responses $\{\mathbf{H}(\boldsymbol{\Lambda}_{s,i})\}_i$ (resp. $\{\mathbf{H}(\overline{\boldsymbol{\Lambda}}_{d,u})\}_u$) and average them across all items I (resp. users U) to get a single frequency response over the user-NN graph (resp. item-FN graph). The frequency responses are shown in Fig. 3 for different values of α.

In all cases, we observe a band-stop behavior since more than 90% of the response in the middle frequencies is zero. The latter corroborates the behavior of the vanilla and graph convolutional NN filter (Huang et al., 2018). Another behavior inherited from the NN/FN graphs is that filters preserve the extreme low and high graph frequencies. Low graph frequencies are signals with a small total variation [cf. (27)], while high graph frequencies are signals with a high total variation.

- *In the user-NN graph*, low frequencies represent signals where similar users give similar ratings. This part is the global trend of preferences among similar users, which is leveraged to predict ratings. High frequencies represent discordant ratings between similar users for a particular item and can be seen as a primitive source for diversity.
- *In the item-FN graph*, the spectral behavior is the same but implications are different. Low frequencies represent ratings with a small difference in *dissimilar* neighboring items; implying, a user u gave similar ratings to dissimilar items. These low frequencies may

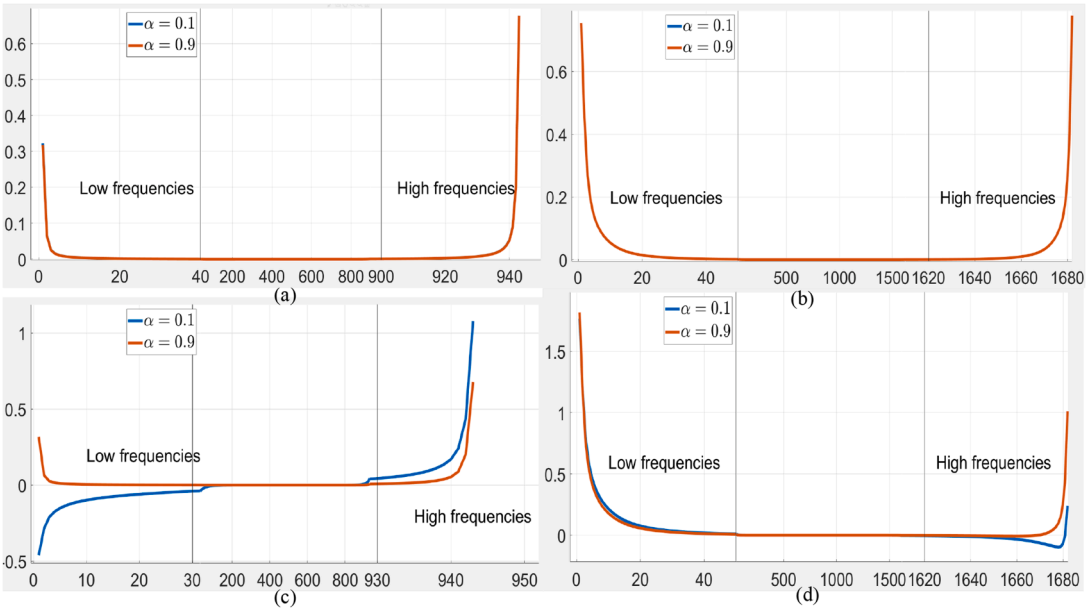

Fig. 3. Frequency responses of the graph filters over a user-NN and an item-FN graph. The horizontal axis is the graph frequency index, while the vertical axis is the estimated frequency response. (Top) Filters designed w.r.t. the mean squared error criterion in (15). (Bottom) Filters designed w.r.t. the Bayesian personalized ranking criterion in (22). (a–c) Frequency response of the user-NN graph filter. (b–d) Frequency response of the item-FN graph filter.

also be because users rate negatively a subset of dissimilar connected items and positively another subset of dissimilar connected items. The high pass components represent ratings changing significantly between neighboring dissimilar items; e.g., one of the two dissimilar items sharing an edge is rated positively while the other negatively. This part contributes towards keeping high the recommendation accuracy while relying on negative correlations between items.

These insights show the joint linear models eliminate irrelevant features (band-stop behavior), smooth out ratings (low frequencies), and preserve discriminative features to aid diversity (high frequencies). This phenomenon is observed for different values of α (importance on NNs vs. FNs) and design criteria (MSE [cf. (15)] vs. BPR [cf. (24)]). The frequency response changes less with α in the MSE design (lines differ by 10^{-3}) than in BPR. This might be because the MSE focuses on the average rating prediction for all items (preferred or not), while the BPR prioritizes a subset of most preferred items. In BPR, we also observed a stronger band-stop behavior for $\alpha \to 1$ meaning the joint model focuses even more on extreme frequencies to predict ratings. This suggests the model relies on the average trend on both graphs (lower frequencies) and on highly dissimilar values in adjacent entities (higher frequencies).

7.2. Graph convolutional neural networks

We now analyze the frequency response of the filters in the GCNN (8). Fig. 4 illustrates the latter for a one-layer GCNN with $F = 2$ filters over each graph. We observe again the strong band-stop behavior. In the NN graph, the stopped band is narrower than in the FN graph, and it is narrower if the GCNN is learned for ranking than rating. The band-stop behavior and the increased focus on the extremly low and high graph frequencies suggest the GCNN leverages the information in a similar way as the linear counterpart. We refer to the previous section to avoid repetition. Lastly, we remark the band-stop behavior is also observed in the vanilla NN [cf. (1)-(2)] and in the linear graph convolutional NN filter (Huang et al., 2018).

8. Numerical experiments

This section corroborates the proposed schemes through experiments with three real datasets of different sparsity, namely, MovieLens100k and MovieLens1M Harper and Konstan (2015), Douban Ma, Zhou, Liu, Lyu, and King (2011) and Flixster Jamali and Ester (2010). Table 2 summarizes their features. We evaluate the trade-offs of the joint models for all combinations in Table 1. We considered both the linear [cf. (4)] and the nonlinear graph convolutional models [cf. (8)] designed for rating [cf. (12)] and ranking [cf. (22)], leading to 16 combinations. We considered the same data pre-processing and train-test split as in Monti et al. (2017). The code to reproduce our results and apply the model to other data is available as an open-source software.[5]

[5] https://github.com/esilezz/accdiv-via-graphconv

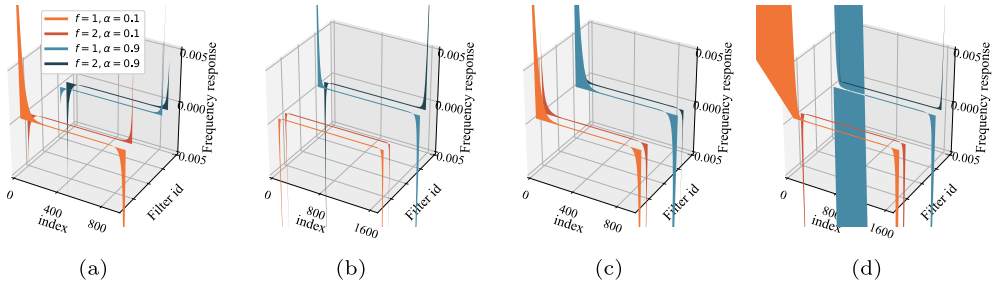

Fig. 4. Frequency responses of the bank of $F = 2$ graph convolutional filters of a one-layer GCNN [cf. (7)] over a user NN graph and an item FN graph. The z-axis is the frequency response that is cropped to improve visibility. The other two axis are the graph frequency index and the filter number. (a) and (b) Filters designed w.r.t. the mean squared error [cf. (15)]. (c) and (d) Filters designed w.r.t. the Bayesian personalized ranking [cf. (22)]. (a) and (c) Filters on the user NN graph. (b) and (d) Filters on the item FN graph.

We quantified accuracy through the root MSE (RMSE) –the lower the better– and the normalized discounted cumulative gain @k (NDCG@k) –the higher the better– and diversity through the aggregated diversity @k (AD@k) and individual diversity @k (ID@k) –both the higher the better (Herlocker, Konstan, Terveen, & Riedl, 2004; Zhang & Hurley, 2008; Ziegler, McNee, Konstan, & Lausen, 2005). The RMSE measures the fitting error for all ratings, while the NDCG@k accounts also for the item relevance in a list of length k. The AD@k is a global at-the-dataset metric and measures the fraction of all items included in *all* recommendation lists of length k. The ID@k is a local at-the-user metric and measures the average diversity in *each* recommendation list. A high ID@k does not imply a high AD@k and vice-versa Adomavicius and Kwon (2011), Wang and Yin (2013). Appendix A provides further detail[6]

We considered a GCNN architecture composed of a single hidden layer with two parallel filters. We trained the GCNN using the ADAM optimizer with the default parameters (Kingma & Ba, 2014) and sought different learning rates γ and fitting-regularizer parameter μ. To limit the search of different hyperparameters, we proceeded with the following rationale. First, we performed an extensive parameter analysis in the MovieLens100k dataset, since this dataset is common in the two most similar graph convolutional works (Huang et al., 2018; Monti et al., 2017) and in the accuracy-diversity trade-off works (Adomavicius & Kwon, 2009; 2011; Zeng et al., 2010). We then used the best performing setting in this dataset and corroborated the trade-offs in the remaining three. Second, we chose the hyperparameters of the similarity graph (number of nearest neighbor, filter order, length of the recommendation list) from the linear graph convolutional filter optimized for rating [cf. (15)] (Huang et al., 2018). Besides being a faster design method to seek different parameters, this strategy allowed evaluating also the accuracy-diversity trade-off of the graph convolutional NN filter. Finally, we kept fixed these parameters for the NN graph and evaluated different combinations on the FN graph.

8.1. Accuracy-diversity trade-off for rating

We first study the trade-off when the joint models are trained for rating [cf. Section 5]. For the NN module, we used the parameters derived in Appendix B. For the FN module, we fixed the number of neighbors to the arbitrary common value 40, evaluated different filter orders $K \in \{1, 2, 3\}$, and show the best results.

Fig. 5 shows the results for the combinations in Table 1 as a function of $\alpha \in [0.1, 0.9]$. As we increase the influence of FNs ($\alpha \to 1$), the RMSE increases. The linear filters are more robust to α than the GCNN. We attribute the latter to the convexity of their design problem. Increasing α increases diversity, while the AD and ID exhibit opposite behavior. Values of α up to 0.5 offer a good trade-off as the RMSE remains unaffected but diversity increases substantially.

To further quantify the trade-off, we allow the RMSE to deteriorate by at most 3% w.r.t. the NN setup [cf. Appendix B] and pick a value of α that respects such constraint. Table 3 compares the different models. For a user NN graph, the joint models (i.e., UU and UI) boost substantially one diversity metric. We believe this is because models build only on user-NN graphs are conservative to both diversity metrics [cf. Fig. B.11], therefore, the margin for improvement is larger. Contrarily, for an item NN graph, the joint models (i.e., IU and II) are conservative and improve by little both diversity metrics. We also highlight the case of II-GCNN which improves the RMSE and AD while keeping the same ID.

8.2. Accuracy-diversity trade-off for ranking

With the same setting of the last section, we now evaluate the trade-off when the joint models are optimized for ranking [cf. Section 6]. These results are shown in Fig. 6. A higher importance to FNs ($\alpha \to 1$) reduces the NDCG@20 but improves diversity. Both the filter and the GCNN are less sensitive to α when designed for ranking. While for the filter we may still attribute this robustness to the

[6] We have also evaluated the models with different metrics including: the mean absolute value (MAE), a surrogate of the RMSE for rating; precision and recall @k, which are ranking-oriented metrics for accuracy; and entropy diversity, which measures the models' ability to recommend items in the long-tail. We have observed these metrics respect the accuracy-diversity trade-off we report and have omitted them for conciseness.

Table 2
Features of the considered datasets.

Data set	Users	Items	Ratings	Sparsity
MovieLens100k	943	1,682	100,000	6.3×10^{-2}
MovieLens1M	6,040	3,952	1,000,000	4.2×10^{-2}
Douban	3,000	3,000	136,891	1.5×10^{-2}
Flixster	3,000	3,000	26,173	2.9×10^{-3}

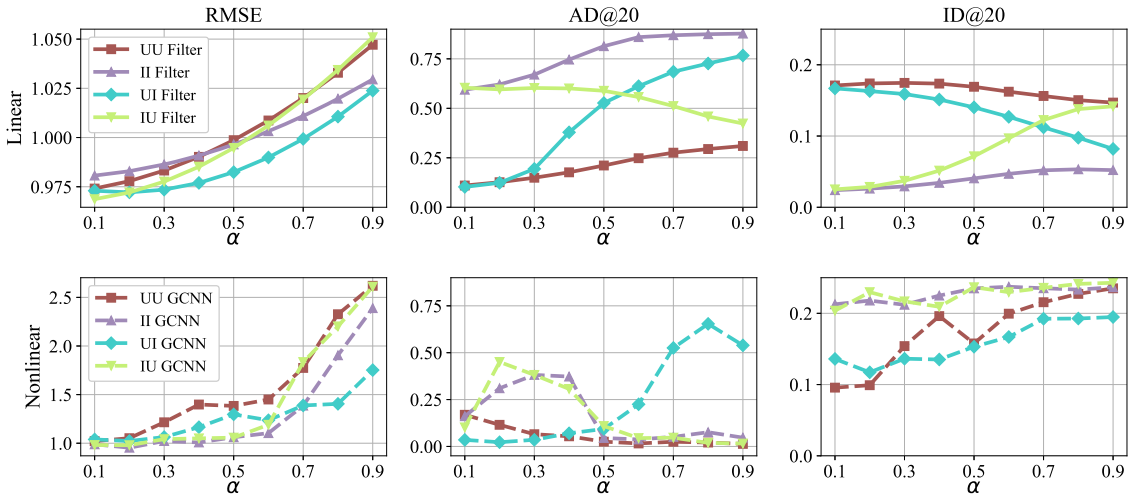

Fig. 5. RMSE, AD@20 and ID@20 as a function of the accuracy-diversity parameter α for models optimized for rating. As more information from the dissimilar connections is included, the RMSE deteriorates but diversity improves. The RMSE of the GCNN is more sensitive to α as its hyper-parameters are not tuned.

Table 3
RMSE, AD@20 and ID@20 for different models optimized for rating. In brackets we show the standard deviation of the absolute error for the RMSE and the standard deviation of the ID@20.

		RMSE	AD@20	ID@20
User Linear	User NN	0.96 (0.58)	0.19	0.02 (0.02)
	UU filter	0.98 (0.60)	0.15	0.17 (0.02)
	UI filter	0.98 (0.61)	0.53	0.14 (0.04)
User GCNN	User GCNN	1.03 (0.65)	0.02	0.15 (0.05)
	UU GCNN	1.05 (0.63)	0.12	0.10 (0.03)
	UI GCNN	1.06 (0.64)	0.04	0.14 (0.03)
Item Linear	Item NN	0.96 (0.62)	0.65	0.03 (0.01)
	II filter	0.98 (0.59)	0.62	0.03 (0.03)
	IU filter	0.98 (0.59)	0.60	0.03 (0.03)
Item GCNN	Item GCNN	0.97 (0.65)	0.29	0.22 (0.08)
	II GCNN	0.95 (0.60)	0.31	0.22 (0.02)
	IU GCNN	0.98 (0.62)	0.45	0.23 (0.07)

optimality of the design problem, the results for the GCNN suggest the BPR leverages better the information from FNs. Note also the filter on the UI combination pays little in NDCG but gains substantially in AD and ID.

To further quantify these results, in Table 4 we show the diversity gain when reducing the NDCG by at most 3%. We note that it is often sufficient to deteriorate the NDCG by 1% and gain substantially in diversity. Bigger diversity improvements are achieved when one of the two graphs is item-based. Lastly, we notice the joint GCNN models gain less in diversity compared with linear filters. The GCNN can be further improved by tuning its parameters.

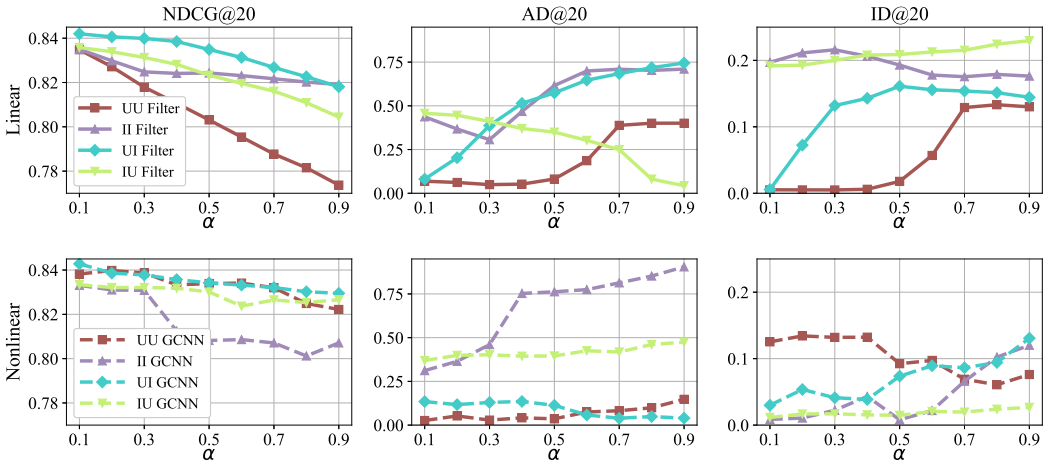

Fig. 6. NDCG@20, AD@20 and ID@20 as a function of the accuracy-diversity parameter α for ranking-optimized models. As more information from FNs is included, the NDCG@20 deteriorates but diversity improves. The NDCG is less sensitive to α compared with the RMSE for both the joint graph convolutional filter and GCNN.

Table 4
NDCG@20, AD@20 and ID@20 for the models working on the NN graph and for the joint models optimized for ranking. In brackets we show the standard deviation of NDCG@20 and ID@20.

		NDCG@20	AD@20	ID@20
User Linear	User NN	0.84 (0.08)	0.19	0.02 (0.02)
	UU filter	0.83 (0.08)	0.07	0.01 (1×10^{-3})
	UI filter	0.83 (0.07)	0.65	0.16 (0.03)
User GCNN	User GCNN	0.84 (0.07)	0.10	0.12 (0.04)
	UU GCNN	0.82 (0.09)	0.15	0.08 (0.03)
	UI GCNN	0.83 (0.08)	0.11	0.07 (0.02)
Item Linear	Item NN	0.83 (0.07)	0.65	0.03 (0.03)
	II filter	0.82 (0.08)	0.70	0.18 (0.05)
	IU filter	0.83 (0.08)	0.41	0.20 (0.04)
Item GCNN	Item GCNN	0.83 (0.09)	0.40	0.02 (0.02)
	II GCNN	0.83 (0.08)	0.46	0.02 (0.02)
	IU GCNN	0.83 (0.07)	0.47	0.03 (0.03)

8.3. *Comparisons with accuracy-oriented models*

In this section, we analyze how the trade-offs of the joint models compare with those achieved by five accuracy-oriented alternatives including state-of-the-art[7] user NN filter [cf. (5)], item NN filter [cf. (6)], and the multi-graph convolutional neural network (MGCNN) (Jamali & Ester, 2010); but also the conventional methods of low-rank matrix completion (LR-MC) (Mazumder, Hastie, & Tibshirani, 2010) and matrix factorization optimized w.r.t. BPR (MF-BPR) (Rendle et al., 2009). Save the last, the first four are designed for rating. We first compare the models in MovieLens100k dataset and then in Douban and Flixster. We consider only the UI combination.

MovieLens100k. Fig. 7 contrasts the RMSE and NDCG@20 with the diversity metrics the AD@20 (left) and ID@20 (right) for $\alpha \in [0.1,0.9]$. The accuracy of GCNN is more sensitive to α than the other models. The GCNN gives also more importance to diversity within the list (ID) rather than covering the catalog (AD). This indicates a few items are recommended by the GCNN but are different between them. Contrarily, the joint linear filters are more robust to accuracy losses, gain in AD, but pay in ID. Contrasting the proposed approaches with the other alternatives, we observe:

- Rating-optimized models (MGCNN, user NN filter, item NN filter, and LR-MC) achieve a lower RMSE but face problems in AD. The item NN filter achieves a reasonable AD but its ID is very low. The MGCNN over-fits the RMSE by prioritizing a few popular items to all users as shown by the low AD and high ID. The joint linear filter can substantially improve the AD by paying little in RMSE, while the GCNN requires additional tuning. The improved AD comes often at expenses of ID, yet values of $\alpha \approx 0.3$ offer a good balance between the two. We can further improve the ID with the IU combination [cf. Fig. 5].

[7] The latter generalize the vanilla NN collaborative filter.

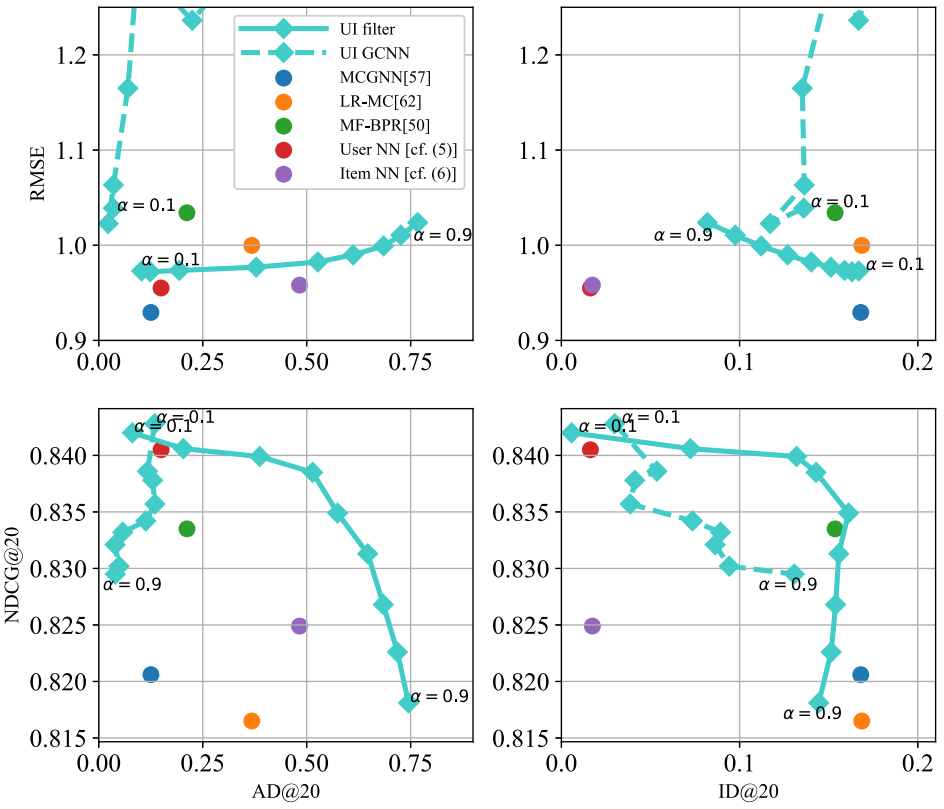

Fig. 7. RMSE and NDCG@20 versus AD@20 and ID@20 as a function of parameter α. Results are shown for the joint models on the UI graph combinations and for five baselines on MovieLens100k. Increasing α for the joint model degrades the accuracy where the GCNN method is more sensitive. Contrarily, diversity increases and we see an opposite behavior between the AD@20 and ID@20.

- The ranking-optimized method (BPR-MF) achieves a high NDCG but still lower than the rating-design user NN filter. This high accuracy is again linked to filling the list with a small group of different items. The joint models optimized for ranking overcome this limitation by making the list slightly more similar (lowering ID) but increasing the catalog coverage (improving AD). This strategy keeps the NDCG high.

Overall, we conclude that a high accuracy from the NNs is tied with an increase of list diversity (ID) but also with a scarce catalog

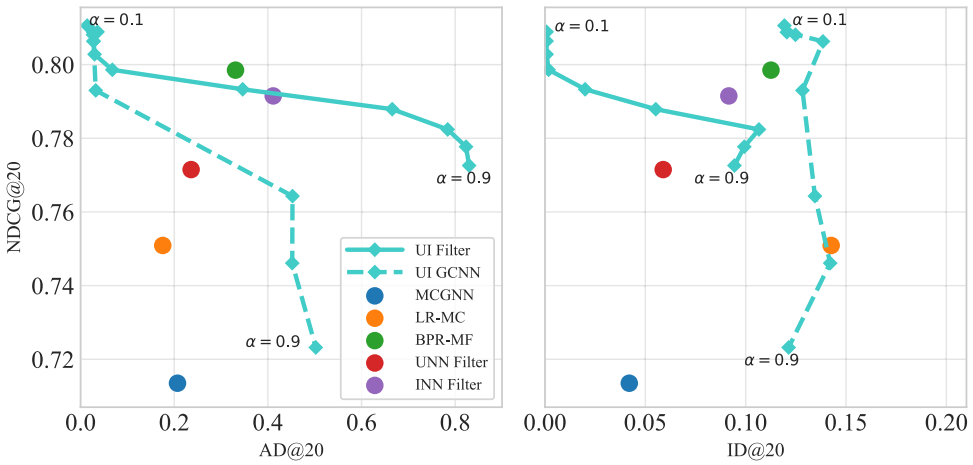

Fig. 8. Accuracy vs. diversity comparisons of the proposed approach for differnet values of α and five baselines in MoveLens1M. We see again that increasing α increases both AD@20 and ID@20 while the NDCG@20 reduces. The GCNN is again more sensitive than the linear counterpart.

coverage (AD). The proposed joint models can keep a reasonable accuracy while contributing to a higher diversity.

MovieLens1M. Likewise, we show in Fig. 8 the performance in the MoveLens1M dataset. We illustrate only the NDCG but a similar trend is achieved also for the RMSE. Overall, we can see the same trend and trade-off as above. While we can see the lienar counterpart can scale and preserve the trade-off without needed a parameter tuning, this is not the case for the GCNN. Especially, if we want to target a trade-off w.r.t. the individual diversity.

Douban & Flixster. We now compare the different models in two datasets containing fewer interactions compared with MovieLens100k; see Table 2. The sparsity of these datasets brings additional challenges when evaluating the NDCG@k. For a list of length 20 there is only one test user for Fixster and none for Douban. To have statistically meaningful results, we measured the NDCG for a list of length $k = 5$, which leads 1,373 test users for Douban and 126 for Fixster. However, to have a unified diversity comparison with the MovieLens100k dataset, we computed the diversity for a list of length 20.

In Table 5, we show the performance for Douban and Fixster datasets, respectively. For our models, we report the extreme values $\alpha = 0.1$ and $\alpha = 0.9$ and a hand-picked value of α. As α increases, the joint models lose in accuracy but gain in diversity. We see again the sensitivity of GCNNs to α for which the RMSE may also reach unacceptable values. The joint models optimized for ranking can always provide a better NDCG w.r.t. MF-BPR while offering a higher diversity. In general, the best trade-off by the joint models is achieved by the GCNN designed for rating and filters designed for ranking.

8.4. Comparison with accuracy-diversity algorithms

In this final section, we compare the joint UI linear model with two alternatives that propose a similar accuracy-diversity trade-off Zeng et al. (2010), Zhou et al. (2010). The hybrid approach in Zeng et al. (2010) mixes a user-based vanilla NN with a user-based vanilla FN. FNs are computed based on the number of items consumed separately and only FNs are multiplied by a single scalar $\alpha \in [-1.4, 0.5]$. The hybrid approach in Zhou et al. (2010) merges a heat diffusion with a random walk to balance accuracy with diversity over item-item graphs. This approach controls the influence of each model similarly to our method through a scalar $\alpha \in [0, 1]$. Both works predict the probability of an item being consumed by a user rather than the rating. Therefore, we compare the accuracy w.r.t. the NDCG. As a baseline, we also consider the vanilla user FN and item FN collaborative filters.

In Fig. 9, we show the trade-offs of the different methods for all three datasets. We see the proposed joint model achieves consistently the highest NDCG while offering a margin to improve accuracy. This behavior is better highlighted in MovieLens100K dataset for which the method hyperparameters have been chosen. We attribute the latter to the fact that the joint model learns its parameters to improve ranking accuracy rather than being a simple fusion of two separate entities. The hybrid strategy from Zhou et al. (2010) focuses entirely on the catalog coverage as can be seen by the high AD. This strategy heavily affects both the NDCG and ID for which this approach performs the worst. The hybrid strategy from Zeng et al. (2010) offers a trade-off in both diversity metrics but the role of the two graphs depends largely on the dataset sparsity. In Flixster, for instance, we see this strategy offers little trade-off as the performance for all values of α but -1.4 is the same. To some extent, this trend is also present in our joint model, yet it has more control over diversity while retaining the highest NDCG.

9. Discussion

The accuracy-diversity trade-off represents a crucial factor for improving user satisfaction when personalizing recommendations. However, achieving the 'right' trade-off is challenging, not only because of its subjective aspects that are difficult to quantify, but also

Table 5
Performance comparison in Douban and Flixster dataset. We show the RMSE for methods trained for rating and the NDCG@5 for methods trained for ranking. For the intermediate value of α, we show in brackets the difference in percentage compared with the best value of competing alternatives.

	Douban					Flixster				
	α	RMSE	NDCG@5	AD@20	ID@20	α	RMSE	NDCG@5	AD@20	ID@20
MCGNN	-	**0.80 (0.58)**	-	0.03	**0.12 (0.05)**	-	**0.93 (0.62)**	-	0.08	0.07 (0.05)
LR-MC	-	1.39 (0.72)	-	0.80	0.04 (0.02)	-	3.17 (1.74)	- 0.45	0.09 (0.05)	
MF-BPR	-	-	**0.75 (0.09)**	0.78	0.05 (0.03)	-	-	**0.72 (0.10)**	0.50	**0.10 (0.06)**
User NN	-	0.76 (0.60)	-	0.52	0.04 (0.02)	-	1.04 (0.65)	-	**0.58**	0.06 (0.03)
Item NN	-	0.8 (0.61)	-	**0.99**	0.05 (0.03)	-	1.12 (0.62)	-	0.33	0.03 (0.03)
Rating linear	0.1	**0.76 (0.59)**	-	0.51	0.04 (0.02)	0.1	1.04 (0.64)	-	0.58	0.06 (0.03)
	0.6	0.84 (0.62)	-	0.75	0.05 (0.03)	0.6	1.05 (0.65)	-	0.69	0.06 (0.05)
	0.9	0.92 (0.65)	-	0.96	0.05 (0.05)	0.9	1.06 (0.66)	-	**0.71**	0.06 (0.05)
Rating GCNN	0.1	0.83 (0.62)	-	0.49	0.10 (0.06)	0.1	1.46 (0.86)	-	0.60	0.10 (0.06)
	0.3	0.85 (0.63)	-	0.44	0.12 (0.07)	0.4	1.46 (0.89)	-	0.56	0.11 (0.07)
	0.9	3.31 (1.54)	-	0.63	0.11 (0.05)	0.9	2.84 (1.48)	-	0.48	0.12 (0.08)
Ranking linear	0.1	-	**0.8 (0.07)**	0.44	**0.15 (0.08)**	0.1	-	**0.75 (0.07)**	0.34	**0.14 (0.09)**
	0.4	-	**0.80 (0.08)**	0.48	0.12 (0.05)	0.3	-	**0.75 (0.08)**	0.35	0.13 (0.08)
	0.9	-	0.77 (0.07)	0.77	0.07 (0.03)	0.9	-	0.74 (0.09)	0.36	0.13 (0.08)
Ranking GCNN	0.1	-	**0.8 (0.08)**	0.47	0.13 (0.06)	0.1	-	0.74 (0.07)	0.30	0.13 (0.08)
	0.4	-	**0.80 (0.07)**	0.9	0.05 (0.04)	0.6	-	0.74 (0.07)	0.35	0.13 (0.07)
	0.9	-	0.76 (0.07)	0.91	0.05 (0.04)	0.9	-	0.73 (0.09)	0.36	0.13 (0.07)

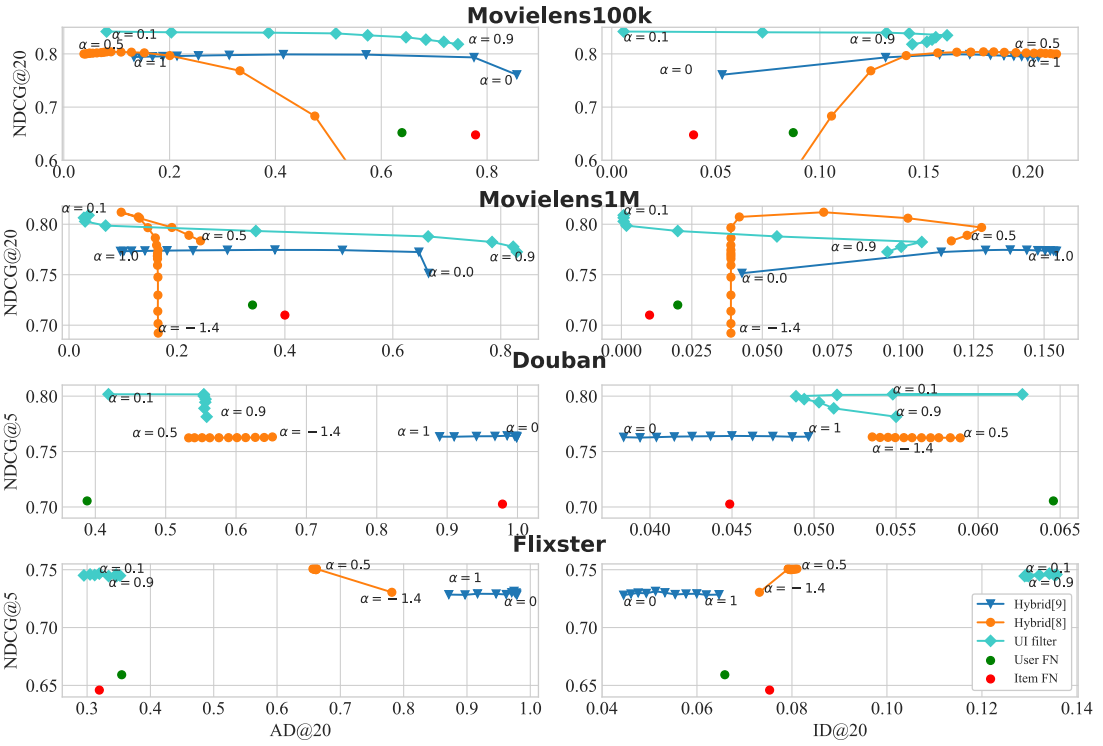

Fig. 9. Accuracy-diversity trade-off of the joint UI filter optimized for ranking [cf. (24)] and of the hybrid approaches from Zeng et al. (2010) and Zhou et al. (2010). The results are shown for different values of the single parameter α controlling the trade-off in all methods. We also show the vanilla user-FN and item-FN for reference. For all methods, we see that chaning α leads to a degradation of accuracy while improving diversity. The proposed approach pays the least in terms of NDCG while offering an improvement in diversity.

because of the complex and irregular user-item relationships that influence both accuracy and diversity. This paper focused on the latter and investigated the potential of graphs that have a proven history as core mathematical tools for representing such data. More specifically, it focused on graph convolutions as the means of dealing with the data complexity and irregularity and to achieve an effective accuracy-diversity trade-off for recommender systems. The overall conclusion of this paper is that graph convolutions have large potential to learn an accuracy-diversity trade-off from the ratings in user-item matrix without relying on side information. Results in three datasets showed graph convolutions attained the highest accuracy while improving diversity compared with other alternatives operating in a similar setting.

The proposed approach relies on the joint information from the nearest and the furthest neighbors in both a learning-to-rate and learning-to-rank setting. We analyzed how this information is leveraged during parameter design and from a graph spectral domain perspective. We formulated a learning problem that has accuracy as main focus but that accounts for the trade-off through a regularizer. When the graph convolutional model is composed only of linear filters, we proved the learning problem is convex and provided solutions for it. Convexity rendered the linear model more robust to hyperparameter choice, while the nonlinear model required careful tuning. In the graph spectral domain, we showed graph convolutions operate as bandstop filters in both the nearest neighbor and in the furthest neighbors graphs. This analysis concluded the joint model exploits the general agreement about preferring or not an item but also complete disagreements between connected nearest and furthest neighbors.

We developed an accuracy-to-coverage trade-off, in which accuracy is traded to recommend niche items; and an accuracy-to-individual diversity trade-off, in which accuracy is traded to improve the diversity in the list. The joint convolutional model offers a balance in each setting that is difficult to be achieved with a single model. Comparisons with nearest neighbor accuracy oriented approaches –including state-of-the-art graph convolutional RecSys methods but also vanilla and graph convolutional nearest neighbor collaborative filtering– showed a diversity improvement by up to seven times while paying about 1% in accuracy. Comparisons with the vanilla furthest neighbor collaborative filtering showed consistently a higher accuracy because of the information from the nearest neighbors. The trend in these findings is in line with that in Said et al. (2012), Zeng et al. (2010). Overall, we have seen graph convolutions can trade accuracy to improve substantially one diversity criteria or improve both by a lesser amount.

The current manuscript has also open aspects. One main question we left unaddressed is why nonlinear GCNN models do not outperform the linear counterparts. Although we have seen a GCNN case that improves accuracy and both diversity metrics, most of the results suggested the joint model should be less complex, the sparser the dataset. This is not entirely surprising as shown also in Chen et al. (2020), Huang et al. (2018). Second, we considered the information coming from the furthest neighbor to improve diversity. However, the latter is only a choice and may require careful learning to keep accuracy at satisfactory levels. In alternative, we

may consider safer choices for the dissimilarity graph, which in turn need revisiting the learning problem to still have diversity gains. Third, an extensive analysis is needed to contrast the proposed approach with re-ranking-based solutions. It would be of interest to identify to what extent the dissimilarity graph contrasts the need for re-ranking and which re-ranking method can yield a better trade-off when applied to the list retrieved by the joint convolutional approaches. Lastly, more research is needed toward explainability. The graph spectral analysis helps in this regard but research is still needed to identify the link between the different spectral components and the items included in the recommendation lists. Nonetheless, to the best of our knowledge this is the first work showcasing the potential of graph convolutions to establish an accuracy-diversity trade-off for recommender systems.

CRediT authorship contribution statement

Elvin Isufi: Conceptualization, Methodology, Writing - original draft, Project administration. **Matteo Pocchiari:** Software, Data curation, Validation, Investigation, Writing - review & editing, Visualization. **Alan Hanjalic:** Supervision, Writing - review & editing.

Appendix A. Metrics

Denote the test set by $\mathcal{T}_s$ and the length of recommendation list by k.

RMSE. For X_{ui} being the true value and $\widehat{X}_{ui}$ the estimated rating for tuple $(u,i) \in \mathcal{T}_s$, the RMSE is defined as

$$\text{RMSE} = \frac{\sqrt{\sum_{(u,i)\in\mathcal{T}_s} \left|\widehat{X}_{ui} - X_{ui}\right|^2}}{|\mathcal{T}_s|}. \tag{A.1}$$

A lower value of RMSE indicates a better fit; hence, a better performance.

NDCG @k. Denote by $\mathcal{I}_{uk} = \{i_{u1}, \ldots, i_{uk}\}$ the set of k items predicted with the highest ratings for user u, i.e., $\widehat{X}_{ui_{u1}} \geq \widehat{X}_{ui_{u2}} \geq \ldots \geq \widehat{X}_{ui_{uk}}$. We first define the discounted cumulative gain (DCG) for which we consider the true ratings $X_{ui} := \text{rel}_i$ (called also the relevance for item i) for items $i \in \mathcal{I}_{uk}$ ordered w.r.t. the predicted order in $\mathcal{I}_{uk}$, i.e., $\text{rel}_1 \geq \text{rel}_2 \geq \ldots \geq \text{rel}_k$. The DCG for user u in a list of length k is defined as

$$\text{DCG}_u@k = \sum_{i=1}^{k} \frac{\text{rel}_i}{\log_2(i+1)}. \tag{A.2}$$

The $\text{DCG}_u@k$ accounts for the ordering of the true values in the predicted list $\mathcal{I}_{uk}$. This ordering can at most be the ideal one $X_{ui_{u1}} = \widehat{X}_{ui_{u1}} \geq X_{ui_{u2}} = \widehat{X}_{ui_{u2}} \geq \ldots \geq X_{ui_{uk}} = \widehat{X}_{ui_{uk}}$, i.e., when the algorithm orders the items in the predicted list $\mathcal{I}_{uk}$ following the true order. In this instance, we refer to it as the ideal DCG for user u ($i\text{DCG}_u@k$). Then, the NDCG@k for the a list k over all users $\mathcal{U}$ is defined as

$$\text{NDCG}@k = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} \frac{\text{DCG}_u@k}{i\text{DCG}_u@k}. \tag{A.3}$$

A high value of NDCG@k indicates a better recommendation in the list of order k; hence, a better performance.

Aggregated diversity. The aggregated diversity measures the fraction of items $\mathcal{I}$ included in the union of all the recommendation lists $\mathcal{I}_{uk}$, i.e.,

$$\text{AD}@k = \frac{1}{|\mathcal{I}|}\bigcup_{u=1}^{|\mathcal{U}|} \mathcal{I}_{uk}. \tag{A.4}$$

A higher aggregated diversity indicates the algorithm recommends a larger portion of the items present in the catalog, consequently, a better performance.

Individual diversity. The individual diversity for a list of length k (ID@k) measures the average diversity within the recommendation lists of all users. For $d(i,j)$ being a distance metric of two items i and j quantifying their dissimilarity, the individual diversity is computed as

$$\text{ID}@k = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} \frac{2}{k(k-1)} \sum_{(i,j)\in\mathcal{I}_{uk}, i\neq j} d(i,j) \tag{A.5}$$

where the inner sum computes the individual diversity for the list $\mathcal{I}_{uk}$ of user u and the outer sum averages across all users. A higher ID indicates the average recommendation list is more diverse. Notice the ID requires computing a distance between items (often based on item features). To use this metric also in featureless items, we followed Kunaver, Dobravec, and Košir (2015) and computed the Euclidean distance based on the first seven SVD latent features for items i and j.

Appendix B. Filter Order Analysis

In this section, we analyze the role of the filter orders in the accuracy diversity tradeoff. We first start with the similarity graph to fix the parameters for this setting and then we discuss the impact of the two filter orders for the joint model.

Similarity Graph. We consider the user-NN and item-NN graphs [cf. Section 3]. For each graph, we evaluated different nearest neighbors $n \in \{5, 10, \ldots, 40\}$, filter orders $K \in \{1, 2, 3\}$, and list length $k \in \{10, 20, \ldots, 100\}$.

NN and filter order. We first analyzed combinations between different NN and filter orders. We fixed the length of the list to $k = 10$ which is a common choice in the literature (Adomavicius & Kwon, 2011; Karypis, 2001; Niemann & Wolpers, 2013; Wang & Yin, 2013; Ziegler et al., 2005). Fig. B.10 shows the RMSE, the AD@10, and the ID@10 for both scenarios. The number of NNs plays a role in the trade-off. More NNs reduce the RMSE but degrade both diversity metrics. This is because each entity gets connected with more similar entities whose combined effect smoothness ratings. For almost all NNs, there is always an order $K > 1$ that improves both accuracy and diversity of the vanilla NN collaborative filter [cf.(1)-(2)].

From these results, we choose the combination that achieves the lowest RMSE. For the user-based scenario, we have: 30 − NN, $K = 3$, RMSE= 0.955, AD@10 = 0.150, and ID@10 = 0.017. For the item-based scenario, we have: 35 − NN, $K = 2$, RMSE= 0.958, AD@10 = 0.482, and ID@10 = 0.017.

Length recommendation list. In Fig. B.11, we show the effect the recommendation list has on trade-off NDCG@k-AD@k, and NDCG@k-ID@k. A longer list improves diversity, but reduces accuracy. This is rather expected because chances to include different items increase in a longer list. At the same time, a longer list makes more challenging identifying the correct order, therefore, reducing the NDCG@k; see also Valcarce, Bellogín, Parapar, and Castells (2018).

Comparing the user NN with the item NN, we see little difference in terms of NDCG, while there is more different in AD and ID.

- User NN achieves a lower AD but a higher ID. This implies the algorithm prioritizes a few relevant items in the catalog but diversifies the list of each user, respectively. In our opinion, this is because the user NN has a narrow view of all items in the catalog as it explores user-similarities. The model fails to account for the broad range of items (each item is treated individually) and prioritizes popular choices, which are different between them. The plateau the user NN reaches for relatively low ID further corroborates the latter.
- Item NN achieves a higher AD but a lower ID. I.e., the model covers a larger portion of the catalog (recommends different items to different users) but, to a specific user, it recommends similar items. Item NN is less user-centric since it leverages item similarities and ignores the influence of other similar users. Consequently, the model has a broader view on items to build recommendations; this explains an AD that is up to four times higher compared with the user NN. Nevertheless, since each user is treated individually less importance is given to diversifying items within the list. We can see this model as highly personalizing the user list because different items are recommended to different users but these items are highly similar.

Based on these results, we set the list length to $k = 20$ since this value achieves the highest NDCG for both the user NN and the item NN. While a longer list can be an option to improve diversity, it is not user-satisfactory to search within it.

Similar-Dissimilar Graphs. Next, we analyze the impact of the filter order in the user-item linear filter combination. Fig. B.12 shows the heatmap of the different metrics (RMSE, NDCG@20, AD@20, and ID@20) as a function of the filter order K in the user-similarity graph and $\overline{K}$ in the item-dissimilarity graph. We can see two main trade-offs.

First, limiting the effects of the dissimilarity graph by setting $\overline{K} = 1$, we see the accuracy increases with the similarity filter order K. This is expected because increasing K implies exploiting more the similar connections for fitting the accuracy-metrics; see also Huang et al. (2018). In addition, also the individual diversity improves, whereas the aggregated diversity deteriorates. Thus, prioritising more the information from the similarity graph benefits an accuracy-to-individual diversity trade-off but a worse catalog coverage. The latter may be interpreted as providing less personalzied recommendations and working with a few niche items that are diverse among them.

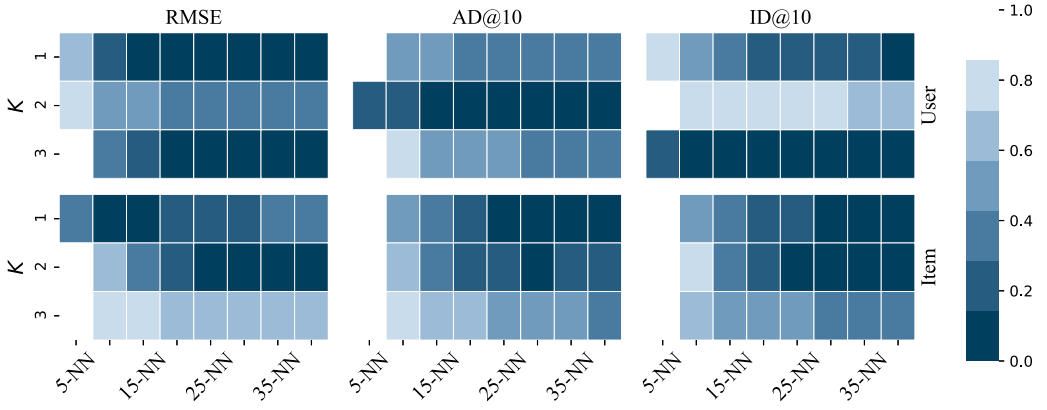

Fig. B.10. RMSE, AD@10, and ID@10 for different filter orders K (vertical axis) and nearest neighbors (horizontal axis). (Top) User-based graph convolutional filter [cf. (5)]. (Bottom) Item-based graph convolutional filter [cf. (6)]. Increasing the NN improves the RMSE but reduces diversity. The filter orders $K > 1$ shows the vanilla NN $K = 1$ [cf.(1)-(2)] can be improved by multi-hop neighbors. Results are scaled in the range [0,1] to improve visibility.

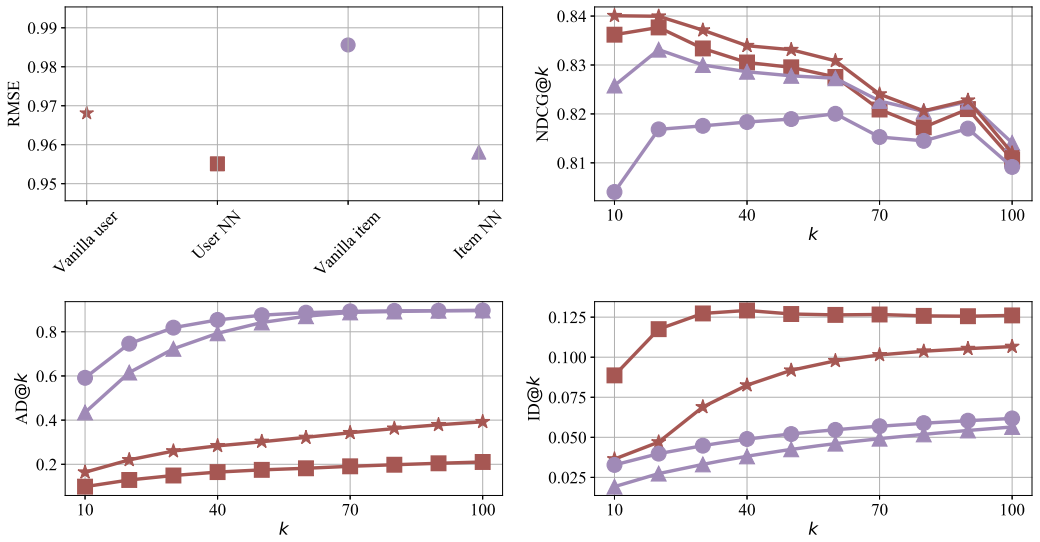

Fig. B.11. RMSE, NDCG@k, AD@k and ID@k as recommendation set's size k varies. We see the NDCGk reduces for lists more than 20 items while diversity increases.

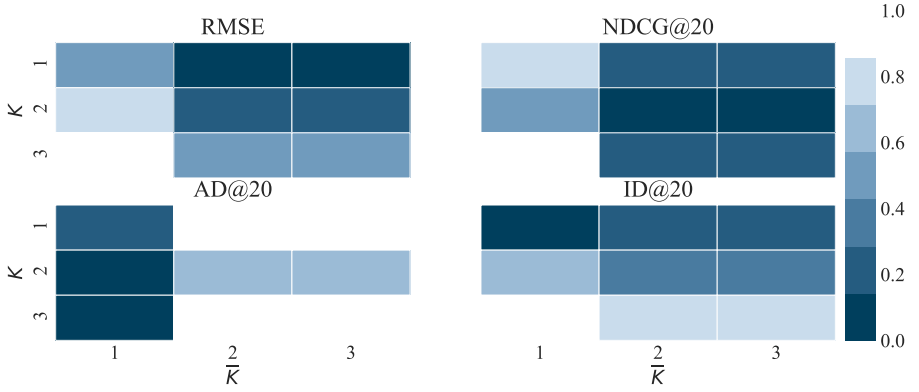

Fig. B.12. Performance metrics for accuracy (RMSE and NDCG@20) and diversity (AD@20 and *ID*@20) as a function of the filter order K on the user-similarity graph and $\overline{K}$ on the item-dissimilarity graph. Results are normalized to ease comparison.

Second, increasing the effect of the dissimilarity graph (i.e., $\overline{K}$) reduces the accuracy but increases diversity. In this instance, the trade-off is in favour of the aggregated diversity. That is, the more the furthest neighbors are included, the more different items are recommended to different users. All this at expenses of having less diversity within the list of a user. Remark in the latter case the gain in aggregated diversity is substantial compared to the loss in individual diversity.

References

Abbassi, Z., & Mirrokni, V. S. (2007). A recommender system based on local random walks and spectral methods. *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on web mining and social network analysis* (pp. 102–108).

Adomavicius, G., & Kwon, Y. (2008). Overcoming accuracy-diversity tradeoff in recommender systems: A variance-based approach, *vol. 8. Proceedings of wits*. Citeseer.

Adomavicius, G., & Kwon, Y. (2009). Toward more diverse recommendations: Item re-ranking methods for recommender systems. *Workshop on information technologies and systems*. Citeseer.

Adomavicius, G., & Kwon, Y. (2011). Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Transactions on Knowledge and Data Engineering, 24*(5), 896–911.

Aggarwal, C. C., et al. (2016). *Recommender systems* (vol. 1). Springer.

Anelli, V. W., Di Noia, T., Di Sciascio, E., Ragone, A., & Trotta, J. (2019). The importance of being dissimilar in recommendation. *Proceedings of the 34th ACM/SIGAPP symposium on applied computing* (pp. 816–821).

Berg, R. v. d., Kipf, T. N., & Welling, M. (2017). Graph convolutional matrix completion. arXiv:1706.02263.

Boyd, S., Boyd, S. P., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Bradley, K., & Smyth, B. (2001). Improving recommendation diversity. *Proceedings of the twelfth irish conference on artificial intelligence and cognitive science, Maynooth, Ireland* (pp. 85–94). Citeseer.
Bridge, D., & Kelly, J. P. (2006). Ways of computing diverse collaborative recommendations. *International conference on adaptive hypermedia and adaptive web-based systems* (pp. 41–50). Springer.
Chen, L., Wu, L., Hong, R., Zhang, K., & Wang, M. (2020). Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach, *vol. 34. Proceedings of the AAAI conference on artificial intelligence* (pp. 27–34).
Chen, S., Niu, S., Lan, T., & Liu, B. (2019). PCT: Large-scale 3D point cloud representations via graph inception networks with applications to autonomous driving. *2019 IEEE international conference on image processing (ICIP)* (pp. 4395–4399). IEEE.
Dehmamy, N., Barabási, A.-L., & Yu, R. (2019). Understanding the representation power of graph neural networks in learning graph topology. *Advances in neural information processing systems* (pp. 15387–15397).
Eskandanian, F., & Mobasher, B. (2020). Using stable matching to optimize the balance between accuracy and diversity in recommendation. *Proceedings of the 28th ACM conference on user modeling, adaptation and personalization, UMAP '20* (pp. 71–79). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3340631.3394858.
Gama, F., Isufi, E., Leus, G., & Ribeiro, A. (2020). From graph filters to graph neural networks. *IEEE Signal Processing Magazine*. arXiv:2003.03777.
Gan, M., & Jiang, R. (2013). Improving accuracy and diversity of personalized recommendation through power law adjustments of user similarities. *Decision Support Systems, 55*(3), 811–821.
Gogna, A., & Majumdar, A. (2017). Balancing accuracy and diversity in recommendations using matrix completion framework. *Knowledge-Based Systems, 125*, 83–95.
Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.
Hamedani, E. M., & Kaedi, M. (2019). Recommending the long tail items through personalized diversification. *Knowledge-Based Systems, 164*, 348–357.
Harper, F. M., & Konstan, J. A. (2015). The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TIIS), 5*(4), 1–19.
Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS), 22*(1), 5–53.
Hua, F., Richard, C., Chen, J., Wang, H., Borgnat, P., & Goncalves, P. (2019). Learning combination of graph filters for graph signal modeling. *IEEE Signal Processing Letters*.
Huang, W., Marques, A. G., & Ribeiro, A. (2017). Collaborative filtering via graph signal processing. *2017 25th European signal processing conference (EUSIPCO)* (pp. 1094–1098). IEEE.
Huang, W., Marques, A. G., & Ribeiro, A. R. (2018). Rating prediction via graph signal processing. *IEEE Transactions on Signal Processing, 66*(19), 5066–5081.
Hurley, N., & Zhang, M. (2011). Novelty and diversity in top-n recommendation–analysis and evaluation. *ACM Transactions on Internet Technology (TOIT), 10*(4), 1–30.
Hurley, N. J. (2013). Personalised ranking with diversity. *Proceedings of the 7th ACM conference on recommender systems* (pp. 379–382).
Ioannidis, V. N., Marques, A. G., & Giannakis, G. B. (2019). A recurrent graph neural network for multi-relational data. *ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 8157–8161). IEEE.
Jamali, M., & Ester, M. (2010). A matrix factorization technique with trust propagation for recommendation in social networks. *Proceedings of the fourth ACM conference on recommender systems* (pp. 135–142).
Javari, A., & Jalili, M. (2015). A probabilistic model to resolve diversity–accuracy challenge of recommendation systems. *Knowledge and Information Systems, 44*(3), 609–627.
Kaminskas, M., & Bridge, D. (2016). Diversity, serendipity, novelty, and coverage: Asurvey and empirical analysis of beyond-accuracy objectives in recommender systems. *ACM Transactions on Interactive Intelligent Systems (TIIS), 7*(1), 1–42.
Karypis, G. (2001). Evaluation of item-based top-n recommendation algorithms. *Proceedings of the tenth international conference on information and knowledge management* (pp. 247–254).
Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv:1412.6980.
Kunaver, M., Dobravec, Š., & Košir, A. (2015). Using latent features to measure the diversity of recommendation lists. *2015 38th International convention on information and communication technology, electronics and microelectronics (MIPRO)* (pp. 1230–1234). IEEE.
Kunaver, M., & Požrl, T. (2017). Diversity in recommender systems–a survey. *Knowledge-Based Systems, 123*, 154–162.
Lipovetsky, S. (2015). Analytical closed-form solution for binary logit regression by categorical predictors. *Journal of Applied Statistics, 42*(1), 37–49.
Liu, J.-G., Shi, K., & Guo, Q. (2012). Solving the accuracy-diversity dilemma via directed random walks. *Physical Review E, 85*(1), 016118.
Ma, H., Zhou, D., Liu, C., Lyu, M. R., & King, I. (2011). Recommender systems with social regularization. *Proceedings of the fourth ACM international conference on web search and data mining* (pp. 287–296).
Mateos, G., Segarra, S., Marques, A. G., & Ribeiro, A. (2019). Connecting the dots: Identifying network structure via graph signal processing. *IEEE Signal Processing Magazine, 36*(3), 16–43.
Mazumder, R., Hastie, T., & Tibshirani, R. (2010). Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research, 11* (Aug), 2287–2322.
Monti, F., Bronstein, M., & Bresson, X. (2017). Geometric matrix completion with recurrent multi-graph neural networks. *Advances in neural information processing systems* (pp. 3697–3707).
Niemann, K., & Wolpers, M. (2013). A new collaborative filtering approach for increasing the aggregate diversity of recommender systems. *Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 955–963).
Nikolakopoulos, A. N., Berberidis, D., Karypis, G., & Giannakis, G. B. (2019). Personalized diffusions for top-n recommendation. *Proceedings of the 13th ACM conference on recommender systems* (pp. 260–268).
Ortega, A., Frossard, P., Kovačević, J., Moura, J. M., & Vandergheynst, P. (2018). Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE, 106*(5), 808–828.
Panniello, U., Tuzhilin, A., & Gorgoglione, M. (2014). Comparing context-aware recommender systems in terms of accuracy and diversity. *User Modeling and User-Adapted Interaction, 24*(1–2), 35–65.
Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). BPR: Bayesian personalized ranking from implicit feedback. *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, UAI '09* (pp. 452–461). Arlington, Virginia, USA: AUAI Press.
Said, A., Fields, B., Jain, B. J., & Albayrak, S. (2013). User-centric evaluation of a k-furthest neighbor collaborative filtering recommender algorithm. *Proceedings of the 2013 conference on computer supported cooperative work* (pp. 1399–1408).
Said, A., Kille, B., Jain, B. J., & Albayrak, S. (2012). Increasing diversity through furthest neighbor-based recommendation. *Proceedings of the WSDM, 12*.
Sandryhaila, A., & Moura, J. M. (2013). Discrete signal processing on graphs. *IEEE Transactions on Signal Processing, 61*(7), 1644–1656.
Sandryhaila, A., & Moura, J. M. (2014). Discrete signal processing on graphs: Frequency analysis. *IEEE Transactions on Signal Processing, 62*(12), 3042–3054.
Sevi, H., Rilling, G., & Borgnat, P. (2018). Harmonic analysis on directed graphs and applications: From fourier analysis to wavelets. arXiv:1811.11636.
Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., & Vandergheynst, P. (2013). The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine, 30*(3), 83–98.
Smyth, B., & McClave, P. (2001). Similarity vs. diversity. *International conference on case-based reasoning* (pp. 347–361). Springer.
Sun, J., Zhang, Y., Ma, C., Coates, M., Guo, H., Tang, R., & He, X. (2019). Multi-graph convolution collaborative filtering. *2019 IEEE International conference on data mining (ICDM)* (pp. 1306–1311). IEEE.
Valcarce, D., Bellogín, A., Parapar, J., & Castells, P. (2018). On the robustness and discriminative power of information retrieval metrics for top-n recommendation. *Proceedings of the 12th ACM conference on recommender systems* (pp. 260–268).
Wang, H., Zhang, F., Zhang, M., Leskovec, J., Zhao, M., Li, W., & Wang, Z. (2019a). Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 968–977).

Wang, J., & Yin, J. (2013). Combining user-based and item-based collaborative filtering techniques to improve recommendation diversity. *2013 6th International conference on biomedical engineering and informatics* (pp. 661–665). IEEE.
Wang, X., He, X., Wang, M., Feng, F., & Chua, T.-S. (2019b). Neural graph collaborative filtering. *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval* (pp. 165–174).
Wasilewski, J., & Hurley, N. (2016). Incorporating diversity in a learning to rank recommender system. *The twenty-ninth international flairs conference*.
Wu, Q., Liu, Y., Miao, C., Zhao, Y., Guan, L., & Tang, H. (2019). Recent advances in diversified recommendation. arXiv:1905.06589.
Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Philip, S. Y. (2020). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*.
Yang, H. (2019). AliGraph: A comprehensive graph neural network platform. *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 3165–3166).
Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., & Leskovec, J. (2018). Graph convolutional neural networks for web-scale recommender systems. *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 974–983).
Zeng, W., Shang, M.-S., Zhang, Q.-M., Lü, L., & Zhou, T. (2010). Can dissimilar users contribute to accuracy and diversity of personalized recommendation? *International Journal of Modern Physics C, 21*(10), 1217–1227.
Zhang, M., & Hurley, N. (2008). Avoiding monotony: improving the diversity of recommendation lists. *Proceedings of the 2008 ACM conference on recommender systems* (pp. 123–130).
Zhong, T., Zhang, S., Zhou, F., Zhang, K., Trajcevski, G., & Wu, J. (2020). Hybrid graph convolutional networks with multi-head attention for location recommendation. *World Wide Web*, 1–27.
Zhou, T., Kuscsik, Z., Liu, J.-G., Medo, M., Wakeling, J. R., & Zhang, Y.-C. (2010). Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences, 107*(10), 4511–4515.
Ziegler, C.-N., McNee, S. M., Konstan, J. A., & Lausen, G. (2005). Improving recommendation lists through topic diversification. *Proceedings of the 14th international conference on world wide web* (pp. 22–32). ACM.