Meta Graph Learning for Long-tail Recommendation

Chunyu Wei*
Tsinghua University
Bejing, China
weicy15@icloud.com

Jian Liang*
Independent Researcher
Bejing, China
liangjianzb12@gmail.com

Di Liu
Alibaba Group
Bejing, China
wendi.ld@alibaba-inc.com

Zehui Dai
Alibaba Group
Bejing, China
zehui.dzh@alibaba-inc.com

Mang Li
Alibaba Group
Hangzhou, China
mang.ll@alibaba-inc.com

Fei Wang
Cornell University
New York, United States
few2001@med.cornell.edu

ABSTRACT

Highly skewed long-tail item distribution commonly hurts model performance on tail items in recommendation systems, especially for graph-based recommendation models. We propose a novel idea to learn relations among items as an auxiliary graph to enhance the graph-based representation learning and make recommendations collectively in a coupled framework. This raises two challenges, 1) the long-tail downstream information may also bias the auxiliary graph learning, and 2) the learned auxiliary graph may cause negative transfer to the original user-item bipartite graph. We innovatively propose a novel Meta Graph Learning framework for long-tail recommendation (MGL) for solving both challenges. The meta-learning strategy is introduced to the learning of an edge generator, which is first tuned to reconstruct a debiased item co-occurrence matrix, and then virtually evaluated on generating item relations for recommendation. Moreover, we propose a popularity-aware contrastive learning strategy to prevent negative transfer by aligning the confident head item representations with those of the learned auxiliary graph. Experiments on public datasets demonstrate that our proposed model significantly outperforms strong baselines for tail items without compromising the overall performance. The code is available on https://github.com/weicy15/MGL

CCS CONCEPTS

• **Information systems** → **Recommender systems**; *Retrieval models and ranking*.

KEYWORDS

long-tail recommendation, graph learning, meta-learning

∗ Equal contributions from both authors. Correspondence to Chunyu Wei. Dr. Liang participated in this work when he was at Alibaba.

KDD '23, August 6–10, 2023, Long Beach, CA, USA

ACM ISBN 979-8-4007-0103-0/23/08…$15.00
https://doi.org/10.1145/3580305.3599428

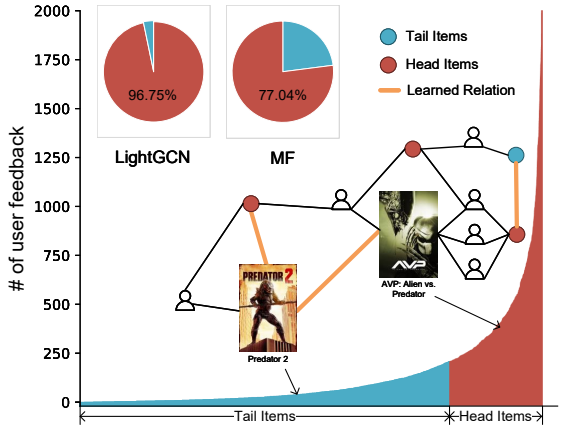

Figure 1: The long-tail distribution for items in Movielens-1M. Pie charts show recommending results trained on Movielens-1M. Orange lines is the learned item relations.

ACM Reference Format:
Chunyu Wei*, Jian Liang*, Di Liu, Zehui Dai, Mang Li, and Fei Wang. 2023. Meta Graph Learning for Long-tail Recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3580305.3599428

1 INTRODUCTION

Recommender systems help users discover potentially liked items, and have been widely deployed to alleviate information overload in diverse scenarios including e-commerce [14], online news and multimedia contents [52]. The core of recommender systems [19, 57] is to build high-quality user and item embeddings, for which the common optimization is to reconstruct user-item interactions. As user-item interactions are naturally represented as a user-item bipartite graph, Graph Convolutional Networks (GCNs) provide an efficient way to integrate multi-hop neighbors and show prominent performance in recommendation [7, 18, 48].

Despite the encouraging performance, the frequency distribution of items is usually uneven in the interaction data. The prevalent distribution is long-tail: a small fraction of popular items receive most of the user feedback, while most items only have little user

feedback. Figure 1 provides evidence of long-tail distribution on a real-world Movielens dataset. As a consequence, the models trained on these uneven datasets would easily overfit a small fraction of popular items, and lead to the popularity bias [1], which recommend items simply from their popularity, rather than user-item matching. For example, we train a standard MF [24] and LightGCN [18] on Movielens dataset, and count head and tail items in the top-20 recommendation lists of all users. From the statistics, the head items are recommended much more frequently than in the training set, which demonstrates that the popularity bias will even cause the Matthew Effect [32], i.e., the "rich get richer" feedback loop. From Figure 1, the popularity bias is more severe on graph-based models, since during multi-hop convolution, the skewed data distribution will bias the models towards the popular items much more easily.

In the real-world scenarios, we find that when giving some tail items enough exposure, they can also be widely liked by users and become popular, which means tail items are not equal to bad items. If we can learn better representations for them, we can make more accurate recommendations. Along this line, some studies utilize meta-learning to transfer the knowledge learned from head items to tail items [26, 56]. However, directly applying them to graph-based models may get sub-optimal results, since these methods take each interaction as independent following an i.i.d assumption. Relations are the key information of a graph, and the goal of graph-based model is to transfer information from the nodes to nodes.

To start with, a straightforward idea is to provide tail items with more connections as illustrated in Figure 1. Although the users' behavior data are usually sparse for the tail items, the attributes are as complete in the tail items as in the head items in most cases. For example, when a new item is launched, it will be annotated with a title or tags containing rich semantic information. One way is to generate edges between the items sharing the same attributes like [34]. However, this will cause several problems: (i) the raw feature space could be sparse and high-dimensional, which makes it difficult to find two items sharing exactly the same attributes; and (ii) it does not consider the consistency with the original bipartite structure, which will result in sub-optimal edges for the recommendation. Therefore, we propose to learn relations among items as an auxiliary graph to update the graph's structure based on the item attributes, in conjunction with the rest of the graph-based recommender training. To achieve this goal, we need to address two notable challenges:

- **Skewed Downstream Information.** To generate edges relevant to the downstream recommendation task, many existing methods [13, 51] directly coupled the graph learning step with the downstream task in an end-to-end manner. However, they do not suit well for the long-tail setting where the downstream imbalanced distribution will further amplify the biased information on the learned graph structure.
- **Negative Transfer.** In the original bipartite graph, head items can already be well represented in the context of rich connections. Thus, the auxiliary graph may introduce noise into the head items representation and cause negative transfer. Thus another key challenge is how to avoid the learned structure comprising the performance on head items while improving on tail items.

In this paper, we propose a novel Meta Graph Learning framework for Long-tail Recommendations (MGL) to address the aforementioned challenges synergistically.

To tackle the **Skewed Downstream Information**, we introduce a meta-learning approach to the learning of the auxiliary graph. The essential idea is to simulate the learning and evaluation procedure during training of the auxiliary graph learning. In the learning phase, our MGL first trains a meta edge generator to reconstruct a debiased item co-occurrence matrix from historical interactions, from which it is fine-tuned to extract the collaborative relations among items from their attributes in an unbiased manner. In the evaluation phase, the unbiased meta edge generator produces item relations as the auxiliary graph, which will participate in the representation learning of items and adapt to the downstream recommendation tasks. This method can generate the learned auxiliary graph well adapted to downstream recommendation tasks, while avoiding the meta edge generator from overfitting the downstream skewed data distribution in the meantime. We present detailed analyses to prove the effectiveness of meta-learning.

To address the **Negative Transfer** issue, we propose a popularity-aware contrastive learning strategy, which also maintain the consistency of the learned auxiliary graph and the original user-item bipartite graph. We regard the item representations from the auxiliary graph and the original bipartite graph as two views of specific items. For those head items with confident representation from the original graph, we use a multi-view contrastive loss to align their representation of both graphs. While for those tail items, we disable the contrastive loss and thus prevent unreliable information from interfering with the learning of the auxiliary graph. The intuition is that the well-represented head items in the original graph can work as share nodes to reconcile two graph structures from different latent spaces.

The contributions of this paper are summarized as follows.

- We propose MGL applying meta-learning to update the original bipartite graph for the graph-based long-tail recommendation, which helps to prevent the skewed downstream information from negatively impacting edge generation.
- We proposed a popularity-aware contrastive learning strategy to prevent negative transfer, which also maintains the consistency of the learned and original graph structure.
- Experimental results show that our method improves tail item recommendations significantly on two benchmark datasets, while achieving solid improvements for the overall performance and head item performance.

2 RELATED WORK

2.1 Long-tail Distribution

Recently, increasing attention has been paid to the problem of long-tailed distribution, e.g., recommendation [8, 30, 38, 54] and imbalanced classification in computer vision (CV) [42, 56]. Resampling is a common strategy to rebalance the skewed dataset, which includes over-sampling [5] and under-sampling [16, 17]. Another strategy is to refine the loss function, such as adding different weights or regularization for different classes [9, 33] or items [4]. Several studies have started to address the long-tailed problems on

graphs. Some GNN-based models consider degree-specific transformations on nodes of different degrees [43, 49], or use structural features including degrees to differentiate nodes [2].

In recommendations, the long-tail distribution is more obvious [4, 30], especially on the item side due to the rapid increase of new items. Some studies add different weights to user-item pairs [53] and perform clustering [35] for alleviating the long-tail problem. Another close related works are cold-start recommendations that purely focus on tail items. They usually integrate different kinds of side information [29, 58] to build cold-start representations for tail items. For example, GME [34] retrieves old ads sharing same attributes for each new ad, and investigates different ways to integrate the old ads embeddings to obtain the new item embeddings for CTR prediction models. Different from them, our work sheds lights on graph-based recommendation, and devises MGL to revise the graph structure for addressing the long-tail problem.

2.2 Graph Learning Models

GCNs perform graph convolutions to integrate vertex features and the input graph topology for node embedding learning [23]. In most cases, the input graph structure is constructed by hand-crafted rules, human experts or precomputed by the k-nearest neighbors of each node [3]. However, the input graph may miss important edges or contain redundant edges, which may be sub-optimal for graph-based downstream tasks. GAT attentively reweights edge importance based on the self-attention mechanism [46]. Some studies propose to learn distance metrics for graph Laplacian matrix with parameterized similarity calculation based on the covariance matrix of input features [28] or feature weight vector [21]. Since the input graph is not always available, another line of research proposed to learn discrete and spare graph structure from scratch by learning a discrete probability on the edges of the graph [11]. However, these models can not be directly applied to our auxiliary graph learning, since we aim to learn a structure in a different semantic space from the original bipartite graph. Also, all these approaches do not consider the long-tail problem, and we argue that these methods will revise the graph structure based on the existing skewed data, to continuously strengthen the existing data distribution and further exacerbate the impact of the long-tail distribution.

2.3 Meta-learning

Meta-learning intends to design models that can learn new skills or adapt to new environments rapidly with a few training samples. It has been successfully applied in many areas, such as CV [36], NLP [6, 22], and recommendation [26, 31, 47]. There are some attempt using meta-learning to train GNNs for quick adaptation to new tasks [20, 41], which is different to our MGL, since we use meta-learning to revise the graph structure instead of improving the GNN's adaptation. Generally, there are three types of meta-learning approaches: 1. metric-based: learn an efficient distance metric [40]; 2. model-based: use (recurrent) networks with external or internal memory [39]; and 3. optimization-based: optimize the model parameters explicitly for fast learning [10].

Our approach is most related to the optimization-based Model-Agnostic Meta-Learning (MAML) [10]. Existing methods adopting MAML [26, 31, 34, 47, 56] usually treat the learning of each item as a task to produce a model adapted for both head and tail items. However, in graph-based recommendation models, this is infeasible because each interaction don't follow an i.i.d assumption. We propose a new meta-learning objective for generating an auxiliary graph structure, such that it not only reduce the impact of the long-tail distribution, but also adapt to the downstream recommendation.

3 PRELIMINARIES

Problem Definition. Let $\mathcal{U} = \{u_1, u_2, \ldots, u_m\}$ denotes the set of users, and let $\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$ denotes the set of items. In the recommendation scenario, we typically use a binary matrix $\mathbf{R} \in \mathbb{R}^{m \times n}$ to store user-item interactions (e.g., purchases and clicks), where $r_{ui} = 1$ indicates that user u consumed item i while $r_{ui} = 0$ means that item i is unexposed to user u or user u is not interested in item i. Items are often associated with attributes, which can be denoted as $\mathbf{Y} \in \mathbb{R}^{n \times d_y}$. d_y is the attribute dimension. Following [18, 48], we represent interaction data as a user-item bipartite graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where the node set $\mathcal{V} = \mathcal{U} \cup \mathcal{I}$ and the edge set $\mathcal{E} = \{e_{ui} | r_{ui} = 1, u \in \mathcal{U}, i \in \mathcal{I}\}$. The adjacency matrix $\mathbf{A}_\mathcal{G}$ can be formulated as: $\mathbf{A}_\mathcal{G} = \begin{bmatrix} 0 & \mathbf{R} \\ \mathbf{R}^T & 0 \end{bmatrix}$. With respect to the adjacency matrix $\mathbf{A}_\mathcal{G}$, the degree matrix $\mathbf{D}_\mathcal{G} \in \mathbb{N}^{(m+n)\times(m+n)}$ is a diagonal matrix, in which each entry $\mathbf{D}_\mathcal{G}[i, i]$ denotes the number of nonzero entries in the i-th row of $\mathbf{A}_\mathcal{G}$.

Our goal is to improve recommendation on tail items, while keeping overall performance flat or better. Formally, we aim to learn an item relation matrix $\hat{\mathbf{S}} \in \mathbb{R}^{n \times n}$ based on item attributes $\mathbf{Y}$ to update $\mathbf{A}_\mathcal{G}$ as $\begin{bmatrix} 0 & \mathbf{R} \\ \mathbf{R}^T & \hat{\mathbf{S}} \end{bmatrix}$ for graph-based long-tail recommendation.

GCN Paradigm. The core of graph convolution on graph $\mathcal{G}$ is to update the ego node by aggregating the representations of its neighbor nodes, which can be formulated as follows:

$$\mathbf{E}^{(l)} = GCN(\mathbf{E}^{(l-1)}, \mathcal{G}), \tag{1}$$

where $\mathbf{E}^{(l-1)}$ is the current representations of nodes and $\mathbf{E}^{(l)}$ is the updated representations after the graph convolution layer. $\mathbf{E}^{(0)}$ is the initial inputs, which are usually the ID embeddings (trainable parameters). Many works design different combining and aggregating mechanisms to enhance the power of GCNs [15, 46, 50]. After iterating the graph convolution L times, $\mathbf{e}^{(l)}$ encodes the information from its L-order neighbors. Usually, there will be a readout function to generate the final representations for the recommendation task:

$$\mathbf{e} = f_{readout}(\{\mathbf{e}^{(l)} | l = 0, 1, \ldots, L\}). \tag{2}$$

For example, $f_{readout}$ can be concatenation [48], weighted sum [18] and retaining the last output [44].

LightGCN Brief. In this paper, we mainly implement our MGL on the simple but effective GCN-based model LightGCN. It abandons the use of feature transformation and nonlinear activation, of which the matrix form can be formulated as:

$$\mathbf{E}^{(l)} = (\mathbf{D}_\mathcal{G}^{-\frac{1}{2}} \mathbf{A}_\mathcal{G} \mathbf{D}_\mathcal{G}^{-\frac{1}{2}}) \mathbf{E}^{(l-1)}, l \in \mathbb{N}^+, \tag{3}$$

where $\mathbf{E}^{(l-1)} = [\mathbf{E}_u^{(l-1)}, \mathbf{E}_i^{(l-1)}]$ is the output of the previous layer or the initial $\mathbf{E}^{(0)}$. At last, LightGCN implement the $f_{readout}$ by weighted sum, in which the weight of each layer is set as $\frac{1}{L+1}$.

After obtaining the representations of users and items, the inner product $\hat{r}_{ui} = \mathbf{e}_u^T \mathbf{e}_i$ is used to predict the preference score, which is commonly adopted in most recommender systems. LightGCN employ the Bayesian Personalized Ranking (BPR) loss [37] to optimize the model parameters, which is a pairwise loss that assumes the observed interactions should be assigned higher prediction values than unobserved ones. The objective function is as follows:

$$\mathcal{L}_{rec} = \sum_{(u,i,j)\in O} -ln\sigma(\hat{r}_{ui} - \hat{r}_{uj}), \tag{4}$$

where $O = \{(u,i,j)|(u,i) \in \mathcal{R}^+, (u,j) \in \mathcal{R}^-\}$ is the pairwise training data, in which $\mathcal{R}^+$ denotes the observed interactions, and $\mathcal{R}^-$ denotes the unobserved interactions.

4 METHODOLOGY

In this section, we outline our MGL framework's overall architecture in Figure 2 and give detailed descriptions of its main components. The main idea of MGL is to generate item relations acquired by a meta edge generator, which forms an auxiliary graph to facilitate graph-based long-tail recommendation. Specifically, we adopt meta-learning to train the meta edge generator for better adaption to the downstream recommendation and better avoidance of the biased data distribution. Also, we innovatively introduce a popularity-aware contrastive learning strategy to maintain the consistency of the learned auxiliary graph and the original bipartite graph by maximizing the agreement of the head items between two graph structures. Thereafter, we conduct theoretical analyses on MGL, revealing its effectiveness.

4.1 Meta Edge Generator

Given the item attributes $\mathbf{Y} \in \mathbb{R}^{n\times d_y}$, we first adopt an item feature encoder $g(\cdot)$ to learn the item auxiliary embeddings, i.e., $\mathbf{p}_i = g(\mathbf{y}_i;\theta), i \in \mathcal{I}$, where $\mathbf{y}_i \in \mathcal{R}^{d_y}$ is the i-th row of $\mathbf{Y}$ and $\mathbf{p}_i \in \mathcal{R}^d$ denotes the auxiliary embedding of item i. In this work, we choose the multilayer perceptron (MLP) model as $g(\cdot)$ following [56]. For simplicity and efficiency, we adopt a vanilla design, cosine similarity, to predict if there is an edge between item i and j:

$$\mathbf{S}_{i,j} = \sigma\left(\frac{< \mathbf{p}_i, \mathbf{p}_j >}{\|\mathbf{p}_i\|\|\mathbf{p}_j\|}\right) \tag{5}$$

where $\sigma(x)$ is a sigmoid function, and $\mathbf{S}_{i,j}$ refers to the predicted relation between item i and j.

To train the meta edge generator in an unbiased manner, we construct an item co-occurrence matrix $\mathbf{S}' \in \{0,1\}^{n\times n}$ by considering the items consumed by the same user are connected:

$$\mathbf{S}' = \min(1, \mathbf{R}^T\mathbf{R}), \tag{6}$$

In this way, we greatly expand the number of connections to the tail items by exploiting the 'co-purchase' relationship. To ensure the fair exposure of each item, only no more than k neighbors are kept for each item in the co-occurrence matrix $\mathbf{S}'$:

$$\mathbf{S}'_{mask} = \mathbf{M} \odot \mathbf{S}' \odot \mathbf{M}^T, \ where \sum \mathbf{M}_{i\cdot} = k, \tag{7}$$

where $\mathbf{M}$ is the random masking matrix on the item co-occurrence matrix $\mathbf{S}'$. The loss function for training the meta edge generator is:

$$\mathcal{L}_{GL} = \|\mathbf{S} - \mathbf{S}'_{mask}\|_F^2, \tag{8}$$

where $\mathbf{S}$ refers to predicted connections between items in $\mathcal{V}$. Solely relying on reconstructing the collaborative information among items is far from adapting to the downstream recommendation. In the following, we will describe how it generates item relations to adapt to downstream recommendations through meta-learning.

4.2 Popularity-aware Contrastive Learning

To maintain the consistency of two graphs, we consider the item embeddings from both auxiliary graph and original bipartite graph as two views of specific items, as they encode two kinds of information, item attributes and consumed histories, respectively. We perform the GCN recommender on the original bipartite graph and obtain the original embeddings of items, which is formulated as:

$$\mathbf{E}^{(l)} = [\mathbf{E}^{(l)}_{user}, \mathbf{E}^{(l)}_{item}] = GCN(\mathbf{E}^{(l-1)}, \mathcal{G}), \tag{9}$$

For simplicity, we use $\mathbf{E}_{item}$ to denote items' original embeddings from the final layer of GCN. The auxiliary embeddings are used to construct the auxiliary graph and thus we use a linear decoder $h(\cdot)$ to transform the auxiliary embeddings into the same space with the original embeddings. In the original bipartite graph, head items with rich connections can be well represented in the graph-based recommender as shown in Figure 3, where we show the hit rate of each item, sorted by their degree, in vanilla LightGCN. We adopt polynomial smoothing to better show the trend w.r.t. the degree of item as the orange line. From Figure 3, the tail items with unreliable representations show poor performance. So if we indiscriminately align two views of all items, it may interfere with the learning of auxiliary graph in tail items.

Thus we use a popularity-aware contrastive loss to reconcile two graph structures from different latent spaces:

$$\mathcal{L}_{PCL} = -\sum_{v_i\in\mathcal{I}} Pop(v_i) \log \frac{exp(s(h(\mathbf{p}_i), \mathbf{e}_i)/\tau')}{\sum_{v_j\in\mathcal{I}} exp(s(h(\mathbf{p}_i), \mathbf{e}_j)/\tau')}, \tag{10}$$

where $\mathbf{e}_i$ is the i-th row of $\mathbf{E}_{item}$ representing item i's original embedding; $h(\cdot)$ is the linear projection; $s(\cdot)$ measures the similarity between two vectors, which is set as cosine similarity function; τ' is the hyper-parameter, known as the temperature in softmax.

In Equation 10, we set a coefficient $Pop(\cdot)$ w.r.t. each item to control whether the contrastive loss works for the specific item. If the item has been consumed by enough users in the original bipartite graph, $\mathcal{L}_{PCL}$ should work to reconcile its representations in two latent spaces to prevent negative transfer. Otherwise, the $\mathcal{L}_{PCL}$ should be frozen in context of the unreliable original embedding. We create a variant Sigmoid function to formulate $Pop(\cdot)$:

$$Pop(v_i) = 1 - \frac{r}{r + \exp\left(\frac{\sum_{u\in\mathcal{U}} r_{ui}}{r}\right)}, \ v_i \in \mathcal{I}, \tag{11}$$

where $\sum_{u\in\mathcal{U}} r_{ui}$ refers to the total consumption item i, known as its degree on $\mathcal{G}$, and r is the hyper-parameter to control how fast $Pop(v_i)$ will increase to about 1. Figure 3 presents the $Pop(v_i)$ curve when $r = 500$, where the popularity coefficient $Pop(\cdot)$ divides the contrastive learning into three phases. In the tail phase, the popularity coefficient is close to zero, which nearly stop the contrastive learning. In the middle and head phase, with the connections to the items gradually become rich, the item's learned representation become more and more confident, so the popularity coefficient gradually increases to 1 to encourage the contrastive learning.

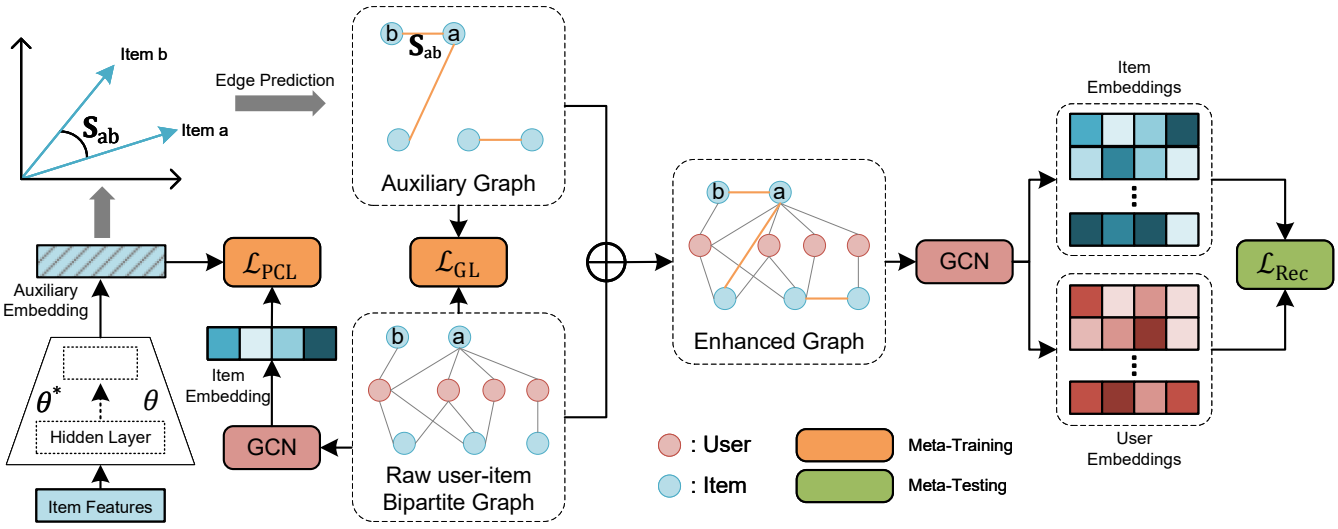

Figure 2: Framework of MGL.

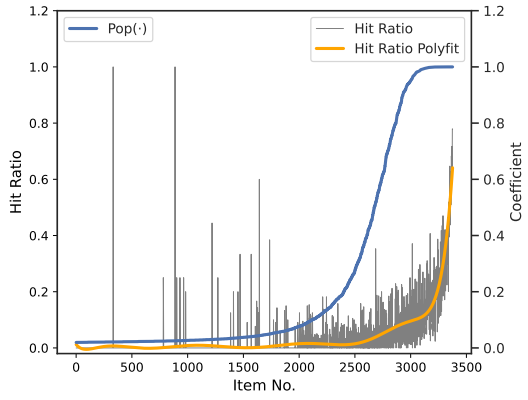

Figure 3: HR on the *Movielens* dataset *v.s.* items sorted by their degrees. Polynomial smoothing is used to eliminate extreme values caused by the lack of tail samples.

4.3 Neighbor Learning for Recommendation

After training meta edge generator to reconstruct the unbiased item collaborative information, we utilize it to generate the auxiliary graph. Note that we do not use the complete item set as candidates to generate edges. Instead, we only select **head** items as candidates to build connections towards each item (including head and tail items) on the graph and calculate the item-item similarity matrix as follows:

$$\hat{\mathbf{p}}_i = g(\mathbf{y}_i; \theta')$$

$$\hat{\mathbf{S}}_{i,j} = \begin{cases} \sigma\left(\frac{<\hat{\mathbf{p}}_i, \hat{\mathbf{p}}_j>}{\|\hat{\mathbf{p}}_i\| \|\hat{\mathbf{p}}_j\|}\right), & (\sum_{u \in \mathcal{U}} r_{uj}) > k_h \\ 0, & \text{Otherwise} \end{cases}, \tag{12}$$

where θ' is the updated parameter of the meta edge generator, and k_h is the degree threshold to distinguish head items. The intuition is same as above, that the head items have already been well represented on the original graph structure, which also makes them good candidates for reliable item-item relations and confident knowledge transfer. This can also avoid the $O(n^2)$ complexity when constructing the item-to-item graph structure, and make MGL computationally feasible real-world recommendation scenario with massive items. To prevent the learned auxiliary graph become too dense to interfere the graph convolution, we sparse the learned similarity matrix following [21, 55]. There are many methods for graph sparseness, including L1 normalization [21] and threshold interception [55]. We found threshold interception performs better in our MGL, which retains the edges with top-K computed similarities:

$$\hat{\mathbf{S}}_{i,j} = \begin{cases} \hat{\mathbf{S}}_{i,j}, & \hat{\mathbf{S}}_{i,j} \in topK(\hat{\mathbf{S}}_i) \\ 0, & \hat{\mathbf{S}}_{i,j} \notin topK(\hat{\mathbf{S}}_i) \end{cases}, \tag{13}$$

where $\hat{\mathbf{S}}_i = [\hat{\mathbf{S}}_{i,1}, \ldots, \hat{\mathbf{S}}_{i,n}]$ is the similarity vector in Equation 12.

With the auxiliary graph structure $\begin{bmatrix} 0 & 0 \\ 0 & \hat{\mathbf{S}} \end{bmatrix}$, we update the final graph structure as $\mathbf{A}_F = \begin{bmatrix} 0 & \mathbf{R} \\ \mathbf{R}^T & \hat{\mathbf{S}} \end{bmatrix}$. We denote the final graph structure as $\mathcal{G}_F$. Specifically, we adopt the same recommender in Equation 9 to obtain the final embeddings:

$$\hat{\mathbf{E}}^{(l)} = [\hat{\mathbf{E}}^{(l)}_{user}, \hat{\mathbf{E}}^{(l)}_{item}] = GCN(\hat{\mathbf{E}}^{(l-1)}, \mathcal{G}_F), \tag{14}$$

where $\hat{\mathbf{E}}^{(0)}$ is also initialized with the ID embeddings. We follow the training paradigm in Section 3 and choose the $\mathcal{L}_{rec}$ in Equation 4 as the objective function for the recommendation task.

4.4 Optimization

MGL applies a meta-learning schema to train the meta edge generator. Formally, we parameterize the meta edge generator as $g(\cdot; \theta)$, and MGL aims for training θ on the item co-occurrence matrix $\mathbf{S}'$, such that it generalizes to generate item relations $\hat{\mathbf{S}}$ for the downstream recommendation task. The process is outlined below.

Meta-Train. The meta edge generator is updated on both $\mathcal{L}_{GL}$ and $\mathcal{L}_{PCL}$, and the loss function is as follows:

$$\mathcal{F}(\theta) = \mathcal{L}_{GL} + \lambda \mathcal{L}_{PCL}, \tag{15}$$

where λ is the hyperparameter to control the weight of popularity-aware contrastive learning. The model is parameterized by θ, so the gradient of θ calculated w.r.t. this loss function is ∇_θ, and optimization will update the model as $\theta' = \theta - \alpha\nabla_\theta$.

Meta-Test. In each mini-batch, the model is also virtually evaluated on generating an auxiliary graph for recommendation. The loss with the updated parameters on the meta-test is as below:

$$\mathcal{J}(\theta', \mathbf{w}, \mathbf{E}^{(0)}) = \mathcal{L}_{rec}, \tag{16}$$

where $\mathbf{w}$ denotes the learnable parameters in the graph-based recommender *GCN*. The loss is calculated using the updated parameters θ' from meta-train, which means that for optimization we will need the second derivative with respect to θ.

Afterward, the meta-train and meta-test are optimized simultaneously, so the final objective is:

$$\min_{\theta, \mathbf{w}, \mathbf{E}^{(0)}} \mathcal{J}(\theta - \alpha\mathcal{F}'(\theta), \mathbf{w}, \mathbf{E}^{(0)}) + \beta\mathcal{F}(\theta), \tag{17}$$

where α is the meta-train step size and β weights meta-train and meta-test. The detailed procedure is illustrated in Algorithm 1.

Algorithm 1 Meta Graph Learning

Input: User-item interaction matrix $\mathbf{R}$; Item Attributes $\mathbf{Y}$
Init: Model Parameters θ, $\mathbf{w}$, $\mathbf{E}^{(0)}$; Hyperparameters α, β.
1: Calculate $\mathbf{S}'_{mask}$ with $\mathbf{R}$;
2: **for** i **in** iterations **do**
3: **Meta-train:** Compute $\mathcal{L}_{GL}$, $\mathcal{L}_{PCL}$, and $\mathcal{F}(\theta)$; (Equ. 15)
4: Update parameter $\theta' = \theta - \alpha\nabla_\theta$
5: **Meta-test:** Compute $\mathcal{J}(\theta', \mathbf{w}, \mathbf{E}^{(0)})$; (Equ. 16).
6: **Meta-optimization:** Update θ, $\mathbf{w}$, and $\mathbf{E}^{(0)}$; (Equ. 17)
7: **end for**

4.5 Analysis of MGL

Following [27], we provide detailed analysis to illustrate how the proposed meta graph learning framework can help prevent overfitting to the downstream skewed data. For simplicity, we omit other parameters except θ in $\mathcal{J}(\theta, \mathbf{w}, \mathbf{E}^{(0)})$ as $\mathcal{J}(\theta)$ in the following analysis. For Equation 16, we do the 1-st order Taylor expansion:

$$\mathcal{J}(x) = \mathcal{J}(\dot{x}) + \mathcal{J}'(\dot{x}) \times (x - \dot{x}), \tag{18}$$

where $\dot{x}$ is an arbitrary point close to x. We let $x = \theta - \alpha\mathcal{F}'(\theta)$, and we have $\dot{x} = \theta$. Then, we have:

$$\mathcal{J}(\theta - \alpha\mathcal{F}'(\theta)) = \mathcal{J}(\theta) + \mathcal{J}'(\theta) \cdot (-\alpha\mathcal{F}'(\theta)), \tag{19}$$

and the objective Equation 17 becomes:

$$\min_{\theta} \mathcal{J}(\theta) + \beta\mathcal{F}(\theta) - \alpha(\mathcal{J}'(\theta) \cdot \mathcal{F}'(\theta)), \tag{20}$$

This reveals that Equation 17 tries to (1) minimize the loss from both meta-train and meta-test, which is intuitive, and (2) maximize the dot product of $\mathcal{J}'(\theta)$ and $\mathcal{F}'(\theta)$. Recall that the dot product $a \cdot b = \|a\|_2\|b\|_2 \cos(\gamma)$ computes the similarity of two vectors, where γ is the angle between vectors a and b. So if a and b are in a similar direction, $a \cdot b$ will become larger. In our case, similar directions of $\mathcal{J}'(\theta)$ and $\mathcal{F}'(\theta)$ means the directions of improvement to θ in both meta-train and meta-test are similar. Thus, the overall objective Equation 17 is trying to *tune such that both the losses of meta-train and meta-test are minimized, and also such that they optimize in a coordinated way.* Conventional optimization $\mathcal{J}(\theta)+\beta\mathcal{F}(\theta)$ without such coordination will cause the model to focus on the easiest task among them and tune asymmetrically. With the regularization of $\mathcal{J}'(\theta) \cdot \mathcal{F}'(\theta)$, the meta edge generator will optimize in the direction, which benefits the recommendation while cannot deviate too far from reconstructing the unbiased item co-occurrence matrix, and thus prevent overfitting to the downstream skewed data.

Table 1: Descriptive statistics of the datasets.

	Movielens-1M	Bookcrossing
# Users	6,040	50,454
# Items	3,706	222,154
# Feedback	1,000,209	1,031,175
Density	0.04468	0.00009
Item Features	Title, Genres, Year	Title, Author, Year, Publisher
Feedback Portion of Top 20% Items	65.25%	50.84%

5 EXPERIMENTS

5.1 Dataset Description

We conduct experiments on two real-world datasets, i.e., *Movielens-1M* and *Bookcrossing*. Descriptive statistics are in Table 1. We can observe that the item distribution from both datasets follows a highly-skewed long-tail distribution, which shows they are well-suited for our problem. In our scenario, we do not utilize the user features, as we mainly study the long-tailed problem on the item side, though our method can also be extended to the user side. We follow similar procedures in [19, 26, 56] to engineer the features and labels. Interactions with missing features are filtered. Feature *year* is treated as the continuous features, while other features are treated as categorical features and represented by one-hot encoding. We apply bag-of-words representations item titles [12]. Consistent with [51], for each user, we randomly split his historical interactions into training, validation, and test parts with a certain ratio (8:1:1).

5.2 Experimental Setup

5.2.1 Evaluation metrics. We adopt two well-known metrics to evaluate models' performance: Normalized Discounted Cumulative Gain@K (NDCG@K) and Hit Ratio@K (HR@K). We repeated experiments five times with different initialization and reported the average values. Specially, we conduct the metric calculation on the entire item set following [25], which would give more reliable evaluation results and avoid the bias brought by sampling procedures. With the goal of improving the tail item recommendation quality without hurting the overall performance, we further report the metrics evaluated on the tail and head item set, respectively. Based on the Pareto Principle [56], we split the first 20% most frequent

Table 2: The recommendation performance of MGL versus baselines on *MovieLens-1M*.

Model	Overall				Head				Tail			
	NDCG@10	HR@10	NDCG@20	HR@20	NDCG@10	HR@10	NDCG@20	HR@20	NDCG@10	HR@10	NDCG@20	HR@20
BPR-MF [37]	0.04094	0.05960	0.05693	0.10419	0.05323	0.08358	0.07340	0.14458	0.01868	0.03387	0.02538	0.05788
NGCF [48]	0.04218	0.06306	0.05933	0.11081	0.05612	0.09072	0.07687	0.15245	0.01947	0.03540	0.02669	0.06095
LightGCN [18]	0.04807	0.06801	0.06508	0.11525	0.05556	0.08619	0.07486	0.14406	0.02299	0.04175	0.03125	0.07104
Over sampling [5]	0.04728	0.06655	0.06426	0.11345	0.05462	0.08317	0.07427	0.14194	0.02331	0.04263	0.03199	0.07330
Down sampling [16]	0.03449	0.04797	0.04639	0.08097	0.04200	0.06316	0.05785	0.11108	0.01968	0.03594	0.02649	0.05991
GAT [46]	0.04644	0.06770	0.06324	0.11409	0.06116	0.09529	0.08265	0.16068	0.02100	0.03720	0.02812	0.06220
EGLN [51]	0.05010	0.06949	0.06778	0.11839	0.06013	0.09149	0.07996	0.15089	0.02229	0.04084	0.03178	0.07461
MeLU [26]	0.04228	0.06217	0.05899	0.10864	0.05353	0.08430	0.07263	0.14135	0.02075	0.03727	0.02771	0.06155
MIRec [56]	0.04302	0.06364	0.05995	0.11065	0.05470	0.08760	0.07369	0.14409	0.02132	0.03574	0.02779	0.05844
MGL	**0.05347**	**0.07551**	**0.07295**	**0.12898**	**0.06302**	**0.09610**	**0.08518**	**0.16213**	**0.02666**	**0.05082**	**0.03695**	**0.08761**
Improv.	+6.73%*	+8.66%*	+7.64%*	+8.95%*	+3.04%*	+0.85%*	+3.07%*	+0.90%*	+14.38%*	+19.22%*	+15.51%*	+17.42%*

Table 3: The recommendation performance of MGL versus baselines on *Bookcrossing*.

Model	Overall				Head				Tail			
	NDCG@10	HR@10	NDCG@20	HR@20	NDCG@10	HR@10	NDCG@20	HR@20	NDCG@10	HR@10	NDCG@20	HR@20
BPR-MF [37]	0.00764	0.01273	0.00995	0.02034	0.01018	0.01974	0.01314	0.03066	0.00444	0.00812	0.00625	0.01433
NGCF [48]	0.01396	0.02265	0.01813	0.03648	0.01357	0.02576	0.01818	0.04289	0.01074	0.01723	0.01460	0.03076
LightGCN [18]	0.01652	0.02445	0.02062	0.03788	0.01738	0.02887	0.02109	0.04279	0.01113	0.01804	0.01430	0.02917
Over sampling [5]	0.01660	0.02356	0.02053	0.03778	0.01703	0.02894	0.02096	0.04173	0.01121	0.01882	0.01461	0.02913
Down sampling [16]	0.01216	0.01789	0.01470	0.02705	0.01328	0.02116	0.01533	0.03271	0.00940	0.01510	0.01249	0.02595
GAT [46]	0.01540	0.02305	0.01951	0.03648	0.01560	0.02676	0.01976	0.04209	0.01106	0.01794	0.01476	0.03066
EGLN [51]	0.01816	0.02717	0.02302	0.04300	0.01819	0.03237	0.02361	0.04761	0.01161	0.01844	0.01495	0.03016
MeLU [26]	0.01371	0.02237	0.01775	0.03572	0.01332	0.02426	0.01820	0.04285	0.01127	0.01885	0.01342	0.02642
MIRec [56]	0.01460	0.02134	0.02006	0.03908	0.01620	0.02906	0.02077	0.04619	0.01116	0.01943	0.01353	0.02765
MGL	**0.01966**	**0.02916**	**0.02402**	**0.04649**	**0.01869**	**0.03251**	**0.02371**	**0.04829**	**0.01488**	**0.02345**	**0.01823**	**0.03545**
Improv.	+8.23%*	+7.37%*	+4.35%*	+8.12%*	+2.72%*	+0.42%*	+0.44%*	+1.44%	+28.13%	+20.66%	+21.97%	+15.25%*

† Boldface denotes the highest score, and underline indicates the best result of the baselines. $*$ denotes that the improvement is significant at $p < 0.05$ level with a two-sample t-test.

items in *Movielens1M* and *Bookcrossing* for head items, and the rest items are treated as tail items.

5.2.2 Compared Methods. To fully demonstrate the effectiveness of MGL on graph-based long-tail recommendation, we compare MGL with the following nine methods, which can be categorized into five classes: (1) classical matrix factorization based method, i.e., BPR-MF [37]; (2) neural graph based CF models with fixed graph structures, i.e., NGCF [48] and LightGCN [18]; (3) re-sampling strategy on LightGCN, i.e., Over sampling [5] and Down sampling [16]; (4) graph learning based model, i.e., GAT [46] and EGLN [51]. (5) Meta-learning based methods, i.e., MeLU [26] and MIRec [56]. Note that for fair comparisons, we leverage the item attributes for item representation learning on all the methods. We introduce all the baseline models here:

- **BPR-MF** [37]: A competitive method improving matrix factorization (MF) with the BPR objective function for a implicit feedback-based recommendation.
- **NGCF** [48]: A graph-based model, which first encodes the collaborative signal into the user-item interaction graph structure and adopts multiple graph convolution to explore high-order connectivity.
- **LightGCN** [18]: A state-of-the-art GCN-based general recommendation model that leverages the user-item proximity to learn node representations and generate recommendations.
- **Over sampling** [5]: A training data re-sampling strategy that samples from tail items to balance the distribution of head and tail items. The over-sampling strategy is more common in practice since user feedback data is highly valuable, and we do not want to down-sample on the positives.
- **Down sampling** [16]: A method where the tail items remain unchanged and the head items are down-sampled.
- **GAT** [46]: An efficient graph neural network with the edges on the bipartite graph re-weighted by self-attention mechanism.
- **EGLN** [51]: A graph learning method leveraging the learned user and item embeddings to update the graph structures.
- **MeLU** [26]: A method based on meta-learning for cold-start item rating prediction. We have adapted it for use in implicit feedback prediction tasks.
- **MIRec** [56]: A state-of-the-art transfer learning framework for long-tail recommendation in two-tower model, which adopts the meta learning to smooth the transfer.

5.2.3 Hyper-parameter settings. We initialize the latent vectors with a Gaussian Distribution with the mean value of 0 and the standard variance of 0.01 for all the models. For fair comparisons,

the dimensions of latent factors are all fixed to 64. We apply a grid search for hyper-parameters. The learning rates are tuned amongst [0.005, 0.01, 0.02, 0.05]. To prevent overfitting, we add L_2 norm with coefficient tuned from [0.001, 0.005, 0.01, 0.02, 0.1]. We select the best models by early stopping when the HR@20 on the validation set does not increase for three consecutive epochs. We use Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1e^{-8}$, which is the same for all baseline models. Following other meta-learning methods [26, 56], we perform one gradient descent for local update in each step.

5.3 Performance Comparisons

We summarize the performance of different algorithms in terms of HR@K and NDCG@K ($k = 10, 20$) over *Movielens-1M* in Table 2 and *Bookcrossing* in Table 3. It demonstrates that MGL outperforms other methods in all evaluation metrics for both tail and head items, and the overall performance. We conduct two-sample t-tests and p-value < 0.05 indicates that the improvements of our MGL are statistically significant. Besides, We have the following observations:

- Compared with the graph learning based EGLN, MGL achieves great improvements, especially on tail items. This verifies item representations learned from skewed graph structures contain biased information, and updating the graph structure directly with them will further amplify this biased information.
- Our MGL significantly outperforms its backbone LightGCN for both head and tail items. This demonstrates the effectiveness of the auxiliary graph for the graph-based recommender, which can enhance the representation leaning of items by bringing more confident neighbors to the items. In tail items, more significant improvement can be observed, which is intuitive, because they lack enough connections on the original bipartite graph to generate confident representations.
- The re-sampling strategies adopted on LightGCN show poor performance. The results are consistent with the previous findings [56] that re-sampling could heavily change the graph structure, which may hamper the sensitive GCN.
- Although GAT shows a better performance than those non-graph based approaches, it does not show superiority compared with LightGCN except in the head items of *Movielens-1M*. We speculate that attentive weight learning is not suitable in a too sparse graph structure [51]. And historical interactions of *Movielens-1M* is much denser than those of *Bookcrossing*, especially on head items, according to Table 1.
- Two meta-learning methods show poor performance, which may because they are both implemented on the conventional two-tower model. Recent advances have proved that graph-based methods can encode high-order information, which can alleviate the interaction sparsity issue and improve the performance.

5.4 Ablation Studies

5.4.1 Effect of Meta-Learning. To investigate the effect of the meta-learning approach, we consider the following variants of MGL:

- **GL**: The edge generator is only supervised by the downstream recommendation loss $\mathcal{L}_{rec}$, i.e., $\min_{\theta,\mathbf{w},\mathbf{E}^{(0)}} \mathcal{J}(\theta, \mathbf{w}, \mathbf{E}^{(0)})$.

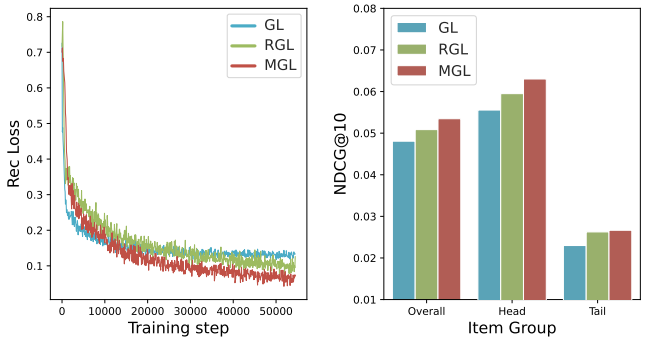

Figure 4: Effect of Meta-Learning on *Movielens-1M*

Table 4: Comparison among models on *Movielens-1M*

Model	OVerall		Head		Tail	
	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10
MGL_B	0.04297	0.06113	0.05206	0.07986	0.01123	0.02282
MGL_F	0.04842	0.06935	0.05590	0.08748	0.02452	0.04528
MGL	**0.05347**	**0.07551**	**0.06302**	**0.09610**	**0.02666**	**0.05082**

- **RGL**: The edge generator is directly supervised by $\mathcal{L}_{rec}$, $\mathcal{L}_{GL}$, and $\mathcal{L}_{PCL}$, i.e., $\min_{\theta,\mathbf{w},\mathbf{E}^{(0)}} \mathcal{J}(\theta, \mathbf{w}, \mathbf{E}^{(0)}) + \beta\mathcal{F}(\theta)$.
- **MGL**: Our proposed model, of which the final objective is $\min_{\theta,\mathbf{w},\mathbf{E}^{(0)}} \mathcal{J}(\theta - \alpha\mathcal{F}'(\theta), \mathbf{w}, \mathbf{E}^{(0)}) + \beta\mathcal{F}(\theta)$.

Fig. 4 shows the recommending training loss *w.r.t.* the number of training steps and the evaluation results on *Movielens-1M*. Specifically, when simply coupling the edge generator with the downstream recommendation task, the recommending loss drops quickly at the very beginning and turns to a steadily decreasing state afterward. We speculate that the reason is consistent with our analysis of EGLN in Section 5.3, that directly utilizing the downstream skewed graph structure as supervising signal will further amplify the biased information and make the model converge quickly. When directly adding $\mathcal{L}_{GL}$ and $\mathcal{L}_{PCL}$ as the regularization term, the recommending loss has a slower decline and is more difficult to converge, which is consistent with our analysis in Section 4.5, that the model tends to focus on the easiest task among multiple targets and thus may compromise the recommendation performance. However, under the meta-learning schema, the recommending loss of our MGL appears to have a faster declining trend after an initial sharp drop, instead of getting an early-stop, which is more likely to converge to a better local optimum. This might also be the reason why **MGL** has better performance than both **GL** and **RGL**, as illustrated by the right part of Fig. 4.

5.4.2 Different Contrastive Learning Strategy. To understand the effects of our proposed popularity-aware contrastive learning strategy, we conduct experiments on MGL and its variants, of which the results are reported in Table 4. Specifically, MGL_B denotes we implement MGL without the contrastive learning between the learned auxiliary graph and the original one (i.e., removing $\lambda\mathcal{L}_{PCL}$ in Equation 15). MGL_F denotes we implement contrastive learning

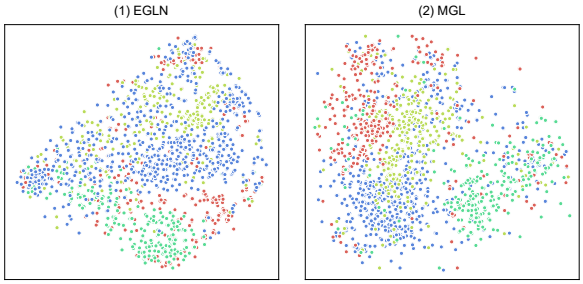

Figure 5: The 2-D visualization of tail items on *Movielens-1M*. The color represents the movie genres.

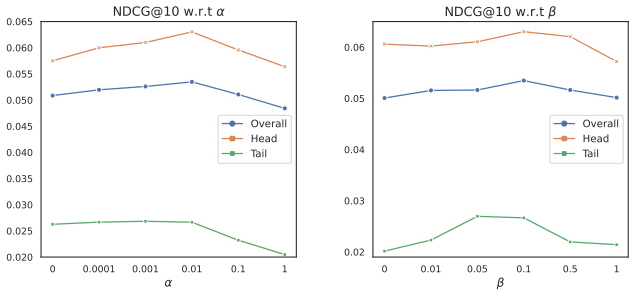

Figure 6: Hyper-parameters study on *Movielens-1M*.

on graphs in the MGL without considering the popularity of corresponding items (i.e., removing $Pop(\cdot)$ in Equation 10). Compared with MGL_B, both MGL_F and MGL achieve obviously improvements on all the item groups, which verifies the alignment between between the learned auxiliary graph and the original one. It can reduces the domain difference between learned and original graph structures and thus may transfer more useful knowledge to the meta-edge generator. However, when we also align the two views of the tail items, MGL_F performs worse than MGL, especially with a larger performance penalty on tail items. These illustrate that the tail items representations from the original graph structure are not confident and will interfere with the learning of the auxiliary graph, which also verifies the necessity of the popularity-aware coefficient $Pop(\cdot)$ in $\mathcal{L}_{PCL}$.

5.5 Embedding Visualization and Case Study

In this section, we look into the learned representations from *Movielens-1M* by visualizing the embeddings. For EGLN and MGL, we visualize the learned tail item embeddings of the top four genres using t-SNE [45] and highlight movie genres in different colors. The visualization results are presented in Figure 5. We can observe that compared with EGLN, which learns a user-item residual graph from the historical interaction, the movie clusters from MGL is more coherent w.r.t the movie genres. This proves that our MGL can better capture the semantic information of tail items. We attribute the improved performance to the learned item-to-item auxiliary graph, which can connect items attribute-similar and relevant to the recommendation, so as to enhance representation learning for tail items that lack enough historical interactions.

5.6 Parameter Sensitivity Analysis

Here we analyze the performances of MGL with hyper-parameters α and β. Limited to the space, we only show the NDCG@10 results over *Movielens-1M* dataset in Figure 6. We observe that MGL achieves the best performance with $\alpha = 0.01$ and $\beta = 0.1$. For the meta-train step size α for local update, we observe that the performance steadily increases when α increases from 0 to 0.01 and decreases quickly when α is larger than 0.01, especially in the tail item. Similar as β balancing the meta-train and meta-test, appropriate value is important for overall objective optimization.

6 CONCLUSIONS

The long-tail item distribution significantly hurts model performance, especially in graph-based recommendation models. We propose a novel framework MGL to learn relations among items as an auxiliary graph to update the graph's structure, in conjunction with the rest of the graph-based recommender training. To tackle the skewed downstream information, we introduce a meta-learning approach to the learning of the auxiliary graph. Also, we propose a popularity-aware contrastive learning strategy to prevent the negative transfer on head items, which also maintains the consistency of the learned auxiliary graph and the original user-item bipartite graph. Extensive experiments on the two benchmark datasets show that MGL consistently outperforms the state-of-the-art methods.

REFERENCES

[1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling Popularity Bias in Learning-to-Rank Recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017*, Paolo Cremonesi, Francesco Ricci, Shlomo Berkovsky, and Alexander Tuzhilin (Eds.). ACM, 42–46.

[2] Nesreen K. Ahmed, Ryan A. Rossi, John Boaz Lee, Theodore L. Willke, Rong Zhou, Xiangnan Kong, and Hoda Eldardiry. 2022. Role-Based Graph Embeddings. *IEEE Trans. Knowl. Data Eng.* 34, 5 (2022), 2401–2415.

[3] David C. Anastasiu and George Karypis. 2015. L2Knng: Fast Exact K-Nearest Neighbor Graph Construction with L2-Norm Pruning. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, James Bailey, Alistair Moffat, Charu C. Aggarwal, Maarten de Rijke, Ravi Kumar, Vanessa Murdock, Timos K. Sellis, and Jeffrey Xu Yu (Eds.). ACM, 791–800.

[4] Alex Beutel, Ed Huai-hsin Chi, Zhiyuan Cheng, Hubert Pham, and John R. Anderson. 2017. Beyond Globally Optimal: Focused Learning for Improved Recommendations. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, Rick Barrett, Rick Cummings, Eugene Agichtein, and Evgeniy Gabrilovich (Eds.). ACM, 203–212.

[5] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Intell. Res.* 16 (2002), 321–357.

[6] Junkun Chen, Xipeng Qiu, Pengfei Liu, and Xuanjing Huang. 2018. Meta Multi-Task Learning for Sequence Modeling. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 5070–5077.

[7] Lei Chen, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. 2020. Revisiting Graph Based Collaborative Filtering: A Linear Residual Graph Convolutional Network Approach. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 27–34.

[8] Zhihong Chen, Rong Xiao, Chenliang Li, Gangfeng Ye, Haochuan Sun, and Hongbo Deng. 2020. ESAM: Discriminative Domain Adaptation with Non-Displayed Items to Improve Long-Tail Performance. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 579–588.

[9] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge J. Belongie. 2019. Class-Balanced Loss Based on Effective Number of Samples. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 9268–9277.

[10] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017 (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 1126–1135.

[11] Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. 2019. Learning Discrete Structures for Graph Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 1972–1982.

[12] Yao Fu, Yansong Feng, and John P. Cunningham. 2019. Paraphrase Generation with Latent Bag of Words. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (Eds.). 13623–13634.

[13] Pallabi Ghosh, Nirat Saini, Larry S. Davis, and Abhinav Shrivastava. 2021. Learning Graphs for Knowledge Transfer With Limited Labels. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, 11151–11161.

[14] Asnat Greenstein-Messica and Lior Rokach. 2018. Personal price aware multi-seller recommender system: Evidence from eBay. *Knowl. Based Syst.* 150 (2018), 14–26.

[15] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 1024–1034.

[16] Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li. 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN 2008, part of the IEEE World Congress on Computational Intelligence, WCCI 2008, Hong Kong, China, June 1-6, 2008*. IEEE, 1322–1328.

[17] Haibo He and Edwardo A. Garcia. 2009. Learning from Imbalanced Data. *IEEE Trans. Knowl. Data Eng.* 21, 9 (2009), 1263–1284.

[18] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 639–648.

[19] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, Rick Barrett, Rick Cummings, Eugene Agichtein, and Evgeniy Gabrilovich (Eds.). ACM, 173–182.

[20] Kexin Huang and Marinka Zitnik. 2020. Graph Meta Learning via Local Subgraphs. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.).

[21] Bo Jiang, Ziyan Zhang, Doudou Lin, Jin Tang, and Bin Luo. 2019. Semi-Supervised Learning With Graph Learning-Convolutional Networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 11313–11320.

[22] Douwe Kiela, Changhan Wang, and Kyunghyun Cho. 2018. Dynamic Meta-Embeddings for Improved Sentence Representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, 1466–1477.

[23] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

[24] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.

[25] Walid Krichene and Steffen Rendle. 2020. On Sampled Metrics for Item Recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 1748–1757.

[26] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. 2019. MeLU: Meta-Learned User Preference Estimator for Cold-Start Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 1073–1082.

[27] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. 2018. Learning to Generalize: Meta-Learning for Domain Generalization. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 3490–3497.

[28] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. 2018. Adaptive Graph Convolutional Neural Networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 3546–3553.

[29] Tingting Liang, Congying Xia, Yuyu Yin, and Philip S. Yu. 2020. Joint Training Capsule Network for Cold Start Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 1769–1772.

[30] Siyi Liu and Yujia Zheng. 2020. Long-tail Session-based Recommendation. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020*, Rodrygo L. T. Santos, Leandro Balby Marinho, Elizabeth M. Daly, Li Chen, Kim Falk, Noam Koenigstein, and Edleno Silva de Moura (Eds.). ACM, 509–514.

[31] Yuanfu Lu, Yuan Fang, and Chuan Shi. 2020. Meta-learning on Heterogeneous Information Networks for Cold-start Recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 1563–1573.

[32] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Ji Yang, Minmin Chen, Jiaxi Tang, Lichan Hong, and Ed H. Chi. 2020. Off-policy Learning in Two-stage Recommender Systems. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen (Eds.). ACM / IW3C2, 463–473.

[33] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. 2021. Long-tail learning via logit adjustment. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

[34] Wentao Ouyang, Xiuwu Zhang, Shukui Ren, Li Li, Kun Zhang, Jinmei Luo, Zhaojie Liu, and Yanlong Du. 2021. Learning Graph Meta Embeddings for Cold-Start Ads in Click-Through Rate Prediction. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 1157–1166.

[35] Yoon-Joo Park and Alexander Tuzhilin. 2008. The long tail of recommender systems and how to leverage it. In *Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008*, Pearl Pu, Derek G. Bridge, Bamshad Mobasher, and Francesco Ricci (Eds.). ACM, 11–18.

[36] Juan-Manuel Pérez-Rúa, Xiatian Zhu, Timothy M. Hospedales, and Tao Xiang. 2020. Incremental Few-Shot Object Detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. Computer Vision Foundation / IEEE, 13843–13852.

[37] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, Jeff A. Bilmes and Andrew Y. Ng (Eds.). AUAI Press, 452–461.

[38] Aravind Sankar, Junting Wang, Adit Krishnan, and Hari Sundaram. 2021. ProtoCF: Prototypical Collaborative Filtering for Few-shot Recommendation. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September 2021 - 1 October 2021*, Humberto Jesús Corona Pampín, Martha A. Larson, Martijn C. Willemsen, Joseph A. Konstan, Julian J. McAuley, Jean Garcia-Gathright, Bouke Huurnink, and Even Oldridge (Eds.). ACM, 166–175.

[39] Adam Santoro, Sergey Bartunov, Matthew M. Botvinick, Daan Wierstra, and Timothy P. Lillicrap. 2016. Meta-Learning with Memory-Augmented Neural

Networks. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016 (JMLR Workshop and Conference Proceedings, Vol. 48)*, Maria-Florina Balcan and Kilian Q. Weinberger (Eds.). JMLR.org, 1842–1850.

[40] Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical Networks for Few-shot Learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 4077–4087.

[41] Qiuling Suo, Jingyuan Chou, Weida Zhong, and Aidong Zhang. 2020. TAdaNet: Task-Adaptive Network for Graph-Enriched Meta-Learning. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 1789–1799.

[42] Kaihua Tang, Jianqiang Huang, and Hanwang Zhang. 2020. Long-Tailed Classification by Keeping the Good and Removing the Bad Momentum Causal Effect. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.).

[43] Xianfeng Tang, Huaxiu Yao, Yiwei Sun, Yiqi Wang, Jiliang Tang, Charu C. Aggarwal, Prasenjit Mitra, and Suhang Wang. 2020. Investigating and Mitigating Degree-Related Biases in Graph Convoltuional Networks. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (Eds.). ACM, 1435–1444.

[44] Rianne van den Berg, Thomas N. Kipf, and Max Welling. 2017. Graph Convolutional Matrix Completion. *CoRR* abs/1706.02263 (2017).

[45] Laurens van der Maaten. 2014. Accelerating t-SNE using tree-based algorithms. *J. Mach. Learn. Res.* 15, 1 (2014), 3221–3245.

[46] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

[47] Maksims Volkovs, Guang Wei Yu, and Tomi Poutanen. 2017. DropoutNet: Addressing Cold Start in Recommender Systems. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 4957–4966.

[48] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer (Eds.). ACM, 165–174.

[49] Jun Wu, Jingrui He, and Jiejun Xu. 2019. DEMO-Net: Degree-specific Graph Neural Networks for Node and Graph Classification. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 406–415.

[50] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

[51] Yonghui Yang, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. 2021. Enhanced Graph Learning for Collaborative Filtering via Mutual Information Maximization. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 71–80.

[52] Xinyang Yi, Ji Yang, Lichan Hong, Derek Zhiyuan Cheng, Lukasz Heldt, Aditee Kumthekar, Zhe Zhao, Li Wei, and Ed H. Chi. 2019. Sampling-bias-corrected neural modeling for large corpus item recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16-20, 2019*, Toine Bogers, Alan Said, Peter Brusilovsky, and Domonkos Tikk (Eds.). ACM, 269–277.

[53] Hongzhi Yin, Bin Cui, Jing Li, Junjie Yao, and Chen Chen. 2012. Challenging the Long Tail Recommendation. *Proc. VLDB Endow.* 5, 9 (2012), 896–907.

[54] Jianwen Yin, Chenghao Liu, Weiqing Wang, Jianling Sun, and Steven C. H. Hoi. 2020. Learning Transferrable Parameters for Long-tailed Sequential User Behavior Modeling. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 359–367.

[55] Donghan Yu, Ruohong Zhang, Zhengbao Jiang, Yuexin Wu, and Yiming Yang. 2020. Graph-Revised Convolutional Network. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2020, Ghent, Belgium, September 14-18, 2020, Proceedings, Part III (Lecture Notes in Computer Science, Vol. 12459)*, Frank Hutter, Kristian Kersting, Jefrey Lijffijt, and Isabel Valera (Eds.). Springer, 378–393.

[56] Yin Zhang, Derek Zhiyuan Cheng, Tiansheng Yao, Xinyang Yi, Lichan Hong, and Ed H. Chi. 2021. A Model of Two Tales: Dual Transfer Learning Framework for Improved Long-tail Item Recommendation. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 2220–2231.

[57] Guorui Zhou, Xiaoqiang Zhu, Chengru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, Yike Guo and Faisal Farooq (Eds.). ACM, 1059–1068.

[58] Yu Zhu, Jinghao Lin, Shibi He, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2020. Addressing the Item Cold-Start Problem by Attribute-Driven Active Learning. *IEEE Trans. Knowl. Data Eng.* 32, 4 (2020), 631–644.