DGCN: Diversified Recommendation with Graph Convolutional Networks

Yu Zheng[1,2], Chen Gao[1,2], Liang Chen[3], Depeng Jin[1,2†], Yong Li[1,2]
[1]Beijing National Research Center for Information Science and Technology
[2]Department of Electronic Engineering, Tsinghua University
[3]Sun Yat-sen University
{jindp, liyong07}@tsinghua.edu.cn

ABSTRACT

These years much effort has been devoted to improving the accuracy or relevance of the recommendation system. Diversity, a crucial factor which measures the dissimilarity among the recommended items, received rather little scrutiny. Directly related to user satisfaction, diversification is usually taken into consideration after generating the candidate items. However, this decoupled design of diversification and candidate generation makes the whole system suboptimal. In this paper, we aim at pushing the diversification to the upstream candidate generation stage, with the help of Graph Convolutional Networks (GCN). Although GCN based recommendation algorithms have shown great power in modeling complex collaborative filtering effect to improve the accuracy of recommendation, how diversity changes is ignored in those advanced works. We propose to perform rebalanced neighbor discovering, category-boosted negative sampling and adversarial learning on top of GCN. We conduct extensive experiments on real-world datasets. Experimental results verify the effectiveness of our proposed method on diversification. Further ablation studies validate that our proposed method significantly alleviates the accuracy-diversity dilemma.

CCS CONCEPTS

• **Information systems** → **Collaborative filtering**.

KEYWORDS

Recommender systems, diversification, graph convolutional networks

ACM Reference Format:
Yu Zheng, Chen Gao, Liang Chen, Depeng Jin, Yong Li. 2021. DGCN: Diversified Recommendation with Graph Convolutional Networks. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3442381.3449835

1 INTRODUCTION

With the rapid development of the WEB, an intelligent algorithm called Recommendation System was proposed to overcome the information overflow problem [46]. The success of recommendation system has been verified in a number of scenarios including e-commerce [27, 38], online news [57, 58] and multimedia contents [9]. With the advancement of recommendation algorithms, much effort has been devoted to improving the accuracy of the recommended items. In other words, the accuracy serves as the dominant target or even the only target in most of the recent research. Chasing for higher accuracy, more attributes are incorporated [25, 28, 37, 44], and more complicated models are proposed [13, 24, 40, 65].

However, an accurate recommendation is not necessarily a satisfactory one [66]. When users access the web, finding the exactly accurate contents is just one of their many needs. For example, users spend much time in browsing news-feed applications for discovering something novel [32]. From the angle of user satisfaction, relevance is never the only rule of thumb. Many factors other than relevance, also influence how users perceive the recommennded contents, such as freshness, diversity, explainability and so on. Among those metrics affecting user satisfaction, **diversity** directly determines user engagement in recommendation scenarios [56]. Specifically, not only the similarity between the user and the recommended items matters, but the dissimilarity among the recommended items reflects the recommendation effect as well. Without the diversity of the recommended items, users are likely to be exposed of repetitive items. That is to say, although the information overload problem is alleviated, another problem of information redundancy is brought in by recommendation system [56].

In order to guarantee user satisfaction, three directions of approaches, namely post-processing, Determinantal Point Process (DPP) and Learning To Rank (LTR), have been proposed to improve the diversity of the recommended results [60]. In the early stage of diversified recommendation, a re-rank or post-processing module is appended after the generation of recommended candidates. The order of the items is determined by heuristics to balance between relevance and diversity. A bunch of solutions in this research line were proposed [3, 6–8, 41, 48, 69]. Independent with candidate generation, the re-rank strategy is decoupled from the optimization of the recommendation model. Thus the diversification signal is not reflected in upstream relevance matching models, which increases the risk of the final recommendation being suboptimal. Recently, another direction of research takes advantage of DPP [11, 19, 21, 22, 55, 56] to replace the heuristics in post-processing based mothods, but the diversification process of DPP is still conducted after the generation stage. To address this problem, a series of methods based on LTR [12, 36] were proposed which target on

†Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '21, April 19–23, 2021, Ljubljana, Slovenia
© 2021 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3449835

directly generating an ordered list of items rather than a candidate set. However, it is tricky to construct an appropriate listwise dataset for such methods. To summarize, existing solutions based on post-processing or DPP aim to improve the diversity leaving the matching process untouched. However, the overall performance of a recommendation system greatly depends on the representations of users and items learned in the matching process. Thus it remains uncertain whether the decoupled design diversifies the recommendation with acceptable loss in accuracy. In terms of LTR based methods, practical issues exist because of the difficulty in collecting feasible datasets.

Since the interactions between users and items can be naturally represented as a heterogeneous graph (a bipartite of users and items), a number of graph based recommendation algorithms were proposed which either utilize rather simple random walk [16, 30] or more complicated methods like GCN [52–54, 61, 63]. In terms of the user-item bipartite graph, higher order neighbors of a user tend to cover more diverse items, because these neighbors contain not only the given user's interacted items, but also other similar users' preferred items. Therefore, it is advantageous to perform diversification on a graph, since the high order connectivity makes diverse items more reachable. Furthermore, performing diversification in GCN also alleviates the aforementioned problem of existing works which separate diversification from the upstream relevance matching model. However, without specific designs for diversity, those high order connections might not be automatically utilized to find items which are not similar to each other. For example, the recommendation system can easily learn to provide items of the most interacted categories because they take up the majority of the edges on the graph. Nevertheless, those GCN based algorithms mainly focus on improving the accuracy, while ignoring how diversity is impacted by the much more complicated GCN model.

In our work, we focus on category diversification in recommendation with the help of GCN. We develop rebalanced neighbor discovering for GCN to make items of disadvantaged categories more reachable. We make adjustments to the negative sampling process to boost the probability of sampling *similar but negative* items. Furthermore, we employ adversarial learning to distill the implicit category preference in the learned embedding space. Through these special designs, we push the diversification process upstream into the matching stage and propose an end-to-end model called **D**iversified recommendation with **G**raph **C**onvolutional **N**etworks (DGCN). The main contributions of this paper are three-fold:

- We analyze the effect of existing diversification algorithms and propose a novel method to combine diversification with matching. The integrity of our method overcomes the problem of decoupled structure in existing works.
- Aiming to generate diverse and relevant items, we carefully design rebalanced neighbor discovering, category-boosted negative sampling and adversarial learning for GCN. An end-to-end model is developed for diversified recommendation.
- We utilize real-world datasets to evaluate the effectiveness of our proposed method. Experimental results demonstrate that diversity of recommendation is validly improved by our method. Furthermore, we conduct ablation studies to confirm the importance of each proposed component.

The remainder of the paper is organized as follows. First, we introduce a few preliminaries in Section 2. Then we elaborate our design of DGCN in Section 3 and conduct experiments in Section 4. After reviewing related work in Section 5, we make some discussions and conclude the paper in Section 6.

2 PRELIMINARIES

2.1 Diversity

As a matter of fact, diversity of recommendation can be either intra-user level or inter-user level [10]. Intra-user level diversity measures the dissimilarity of the recommended items of an individual user, while inter-user level focuses on the provided contents for different users. In this paper, we target on improving intra-user level diversity[1] as most of the related research, and leave inter-user level diverisity (also known as *decentration* or long-tail recommendation) for future work.

Diversity is often mixed up with serendipity or novelty. For example, suppose 70% of the purchased items for a user are electronic devices, 20% are clothes and the other 10% are drinks. Then a recommended list of 10 items including five or more electronic devices, one or two clothes and one or two drinks is a much more diverse result than recommending ten electronic devices. Moreover, even though the user did not purchase any books in her interaction history, she might still have interests in reading, and serendipity stands for the capability of the recommendation system to provide items appealing to users but not realized by themselves (books for the user in this case).

In this paper, we focus on category diversification [69]. When users browse the recommended items, it is not user-friendly to provide a large amount of items belonging to the same category. We utilize three widely adopted metrics for diversity in our experiments:

- **coverage**: this metric measures the number of recommended categories. Coverage reflects the holistic and overall diversity of a recommendation system.
- **entropy**: this one focuses on the distribution on different categories. Using the previous example, the entropy value of four electronic devices, three clothes and three drinks is higher than recommending seven electronic devices and three drinks.
- **gini index**: this index is popularly adopted in economics to measure the wealth or income inequality, and it was further adapted and introduced to recommendation by [2]. The number of items belonging to a specific category can be explained as the wealth of that category.

Note that in terms of coverage and entropy, higher value means stronger diversity, while for gini index it is the opposite (**lower is better**).

2.2 Recommendation Pipeline

As illustrated in Figure 1, a typical pipeline for a recommender system is composed of three stages: (1) matching (candidate generation), (2) scoring, and (3) re-ranking. Several hundreds of items are selected in matching stage from a large item pool. Then, usually complicated deep learning models are adopted in scoring stage to

[1] in this paper, *diversity* refers to *intra-user level diversity* for simplicity

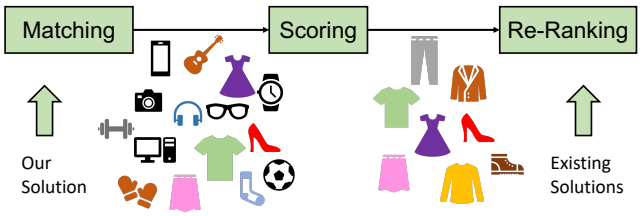


Figure 1: A typical recommendation pipeline.

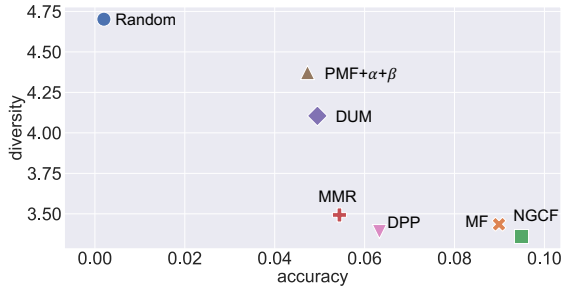


Figure 2: The accuracy-diversity dilemma.

estimate interaction probability and the top dozens of items are selected. In re-ranking, selected items are re-arranged to satisfy additional constraints.

With respect to diversity, it is widely adopted to impose heuristic rules in the re-ranking stage to diversify recommended items. Different re-ranking methods were proposed to balance between relevance and diversity [3, 7, 8, 56]. However, diversification in re-ranking is independent with upstream matching and scoring models, which makes the whole system suboptimal. Furthermore, with matching models unaware of diversification signals, the retrieved items from matching can be already redundant, which limits diversity from the source. In this paper, we aim at pushing diversification upwards. Specifically, we take diversity into consideration during matching, and propose an end-to-end method to provide diverse recommendation.

2.3 Accuracy-Diversity dilemma

Generally, when considering diversity, it is not easy to get rid of the so called accuracy-diversity dilemma [66], especially in offline evaluations. That is, higher accuracy often means losing diversity to some extent. We compared several recommendation algorithms (random, matrix factorization [35] and neural graph collaborative filtering [54]), as well as a bunch of diversification algorithms (MMR [8], DUM [3], PMF+α+β and DPP [20]), utilizing a real world e-commerce dataset collected from Taobao[2], which is the largest e-commerce platform in China. In order to verify the tradeoff between the two metrics, we plot the results of these methods in Figure 2 with accuracy and diversity as the two axes.

From Figure 2 we can observe that, with the recommendation algorithm getting more complicated, though more relevant products are provided, less categories are presented to customers. After introducing these diversification strategies, the diversity indeed gets promoted, while the accuracy is not guaranteed. Although there exist certain hyper-parameters in these diversification methods to balance the bias, later experiments show that it is rather difficult to find a satisfactory point.

[2]www.taobao.com

3 METHOD

3.1 Overview

As introduced previously, we incorporate diversification into the matching process with GCN. We aim at providing items correlated with users' interests while dissimilar with each other (diversity). Figure 3 illustrates the holistic structure of our proposed DGCN.

In real-world recommendation scenarios, the interactive behaviors between users and items are always the strongest signals with respect to user preference modeling. Thus we first construct a graph to represent the interactions, which consists of two types of nodes (users and items), and edges between them stand for those behaviors. It is worthwhile to state that edges are undirected in our constructed bipartite graph. We propose rebalanced neighbor discovering to solve the problem of inconsistency between different categories. By applying GCN on top of the the sampled sub-graph, node features propagate back and forth between users and items, which accurately simulates the collaborative filtering effect. With the goal of making diverse categories more accessible, we make adjustments to the negative sampling process, boosting the probability of selecting similar items. Furthermore, we add an adversarial task of item category classification to strengthen the diversity.

Our proposed method is featured with the following three special designs targeting on diversification: (1) Rebalanced Neighbor Discovering, (2) Category-Boosted Negative Sampling and (3) Adversarial Learning.

- **Rebalanced Neighbor Discovering** To discover more diverse items on the graph, we design a neighbor sampler based on the distribution of the neighbors. We increase the probability of selecting items of disadvantaged categories, and limit the effect of those dominant categories. With the guidance of the neighbor sampler, items of multiple categories are more reachable.
- **Category-Boosted Negative Sampling** Unlike random negative sampling, we propose to choose those *similar but negative* items with a boosted higher probability. By distorting the distribution of negative samples, representations for users and items are learned in a finer level, where the recommendation system can determinate the user preferences among similar items.
- **Adversarial Learning** We leverage the technique of adversarial learning, playing a min-max game on item category classification. With an extra adversarial task, we distill users' category preferences from item preferences, which makes the learned embeddings category-free. And consequently, neighbors in this embedding space will cover more categories.

In the following sections, we first introduce the architecture of the adopted GCN, and then we elaborate on the three special designs for diversity one after another.

3.2 GCN

Our GCN is composed of an embedding layer and a stack of graph convolutional layers, where each graph convolutional layer contains a broadcast operation and an aggregation operation.

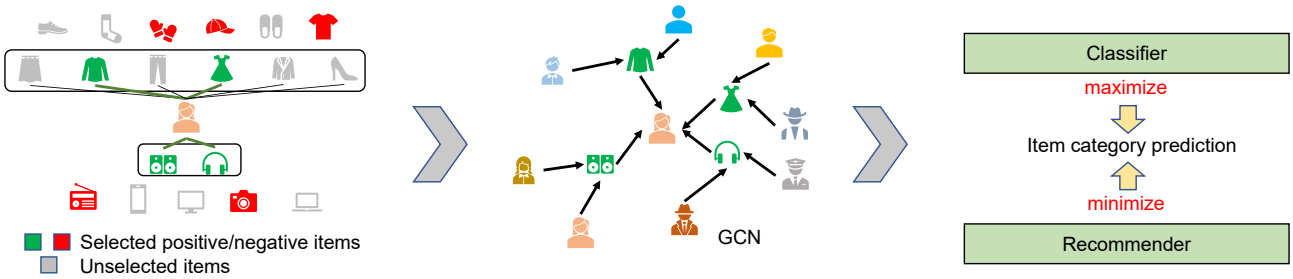


Figure 3: Overview of our proposed DGCN.

3.2.1 Embedding Layer. Inspired by representations for words and phrases [39], the embedding technique has been successfully employed in recommendation system [24]. In our work, the inputs of GCN are simply ID features for users and items. Following the widely adopted embedding strategy, we transform the one-hot ID feature to a dense vector, thus we have the following embedding look-up table:

$$E = [\boldsymbol{e}_{u1}, \ldots, \boldsymbol{e}_{uM}, \boldsymbol{e}_{i1}, \ldots, \boldsymbol{e}_{iN}], \tag{1}$$

where M is the number of users, and N is the numberof items. We represent each node with a separate embedding $\boldsymbol{e} \in \mathbb{R}^d$, in which d is the embedding size. It is worthwhile to note that the embeddings are learnable parameters and further fed into GCN for message passing on the graph. Thus the embedding can be regarded as the node feature at layer 0, i.e. $\boldsymbol{h}_v^0$ which will be introduced later.

3.2.2 Graph Convolutional Layer. We perform embedding broadcast and aggregation in the graph convolutional layer. In other words, within a graph convolutional layer, each node broadcasts its current embedding to all its neighbors and aggregates all the messages sent to it to update its embedding. In terms of neighbor aggregation, we utilize average pooling combined with a feature transform matrix and a nonlinear activation function. Formally, we denote the feature vector of node v at the k-th layer as $\boldsymbol{h}_v^k$, and the update rules are illustrated as follows:

$$\begin{aligned} \boldsymbol{h}_{AGG}^k &= MEAN(\boldsymbol{h}_j^{k-1}, \forall j \in \mathcal{N}(v)), \\ \boldsymbol{h}_v^k &= \tanh(\mathbf{W}^k \boldsymbol{h}_{AGG}^k) \end{aligned} \tag{2}$$

where $\mathcal{N}(v)$ represents the set for sampled neighbors of node v. As investigated in [59], adding self loops is of crucial importance in graph convolutional networks, since it compresses the spectrum of the normalized Laplacian. Therefore, we also insert node v itself into $\mathcal{N}(v)$. In this way, node embeddings propagate on the graph in a layer-wise manner.

3.2.3 Interaction Modeling. With respect to interaction modeling in the matching stage, heavy computation such as the inference computing of neural networks is impractical due to the large item pool and the strict latency requirements. For embedding based matching model, inner product and L2 distance are widely used. Furthermore, at online serving time, these simple rather effective interaction modeling methods can be greatly accelerated with the help of nearest neighbor search algorithms. Therefore, we use the representations of users and items at the last graph convolutional layer and take inner product of them to estimate the interaction probability:

$$p_{u,i} = < \boldsymbol{h}_u^K, \boldsymbol{h}_i^K >, \tag{3}$$

where K is the depth of the graph neural networks. During evaluations, top items with respect to $p_{u,i}$ are selected as recommended items for a given user u.

3.2.4 Prevent Overfitting. To prevent our model from overfitting, we perform dropout [49] on the feature level. To be specific, we randomly drop the intermediate node embeddings between consecutive graph convolutional layers with probability p, where p is a hyper-parameter in our method.

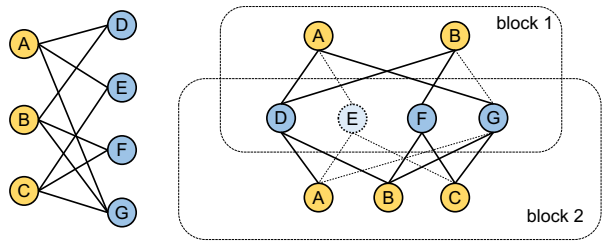
With great capability in learning representations for graph structures, GCN has been shown effective to capture the collaborative effect on the user-item bipartite graph, which improves the accuracy significantly. However, utilizing the high order connectivity for diversification has received little scrutiny. We then introduce our special designs for diversity in the proposed DGCN.

3.3 Rebalanced Neighbor Discovering

In the matching stage, usually the recommendation system retrieves items from a large corpus which is of million-scale or even billion-scale [63]. Feeding the whole graph consisting of millions of nodes to a GCN suffers from highly inefficient computation and heavy resources usage. Moreover, it is difficult to implement mini-batch training on the whole graph. Thus a neighbor sampler is often employed to sample a sub-graph from the original large one for efficient training [26]. With the help of the neighbor sampler, inductive learning on the graph is accomplished and it has been proved scalable to billion-scale graphs [63]. Specifically, the neighbor sampler generates a *Node Flow*, which is a sub-graph with multiple layers, where edges only exist in consecutive layers.

Figure 4 serves as a toy example for neighbor discovering. Specifically, during the training process, a mini-batch consists of a certain number (i.e. batchsize) of users and items, and these user nodes and item nodes in a single batch form the set of seed nodes. The neighbor sampler randomly selects their neighbors and extracts the sub-graph. For GCN deeper than one layer, this neighbor discovering process will repeat recursively, which means the sampled neighbors become seed nodes for the next hop. It is clearly illustrated in Figure 4 that edges exist in consecutive layers and connected layers form a block. Graph convolutional operations are performed block by block, where each graph convolutional layer corresponds with one block.

However, the above neighbor discovering strategy leaves aside the problem of diversity. In real-world recommendation scenarios,



(a) user-item bipartite (b) sampled node flow

Figure 4: Node Flow generated by the neighbor sampler for a 2-layer GCN. In this example, node A and node B are the initial seed nodes. For each node, the neighbor sampler randomly samples two neighbors. The first GCN layer performs convolution in block 1, and all the activated neighbors in block 1 form the seed nodes for sampling block 2, which corresponds to the second GCN layer.

items of distinct categories are supposed to be regarded differently because users perceive them differently. In other words, there exist dominant categories and disadvantaged categories according to users' interaction history, where users engage more with items of dominant categories and spend much less time in disadvantaged categories. A diversified recommendation system is capable of providing items from not only dominant categories but also disadvantaged categories. From the view of the graph, dominant categories of a user become the mainstream in the message flow on the graph, because they take up most of the in-edges to the user node. Thus, without distinguishing between dominant categories and disadvantaged categories in neighbor discovering, the learned user embeddings are likely to be too close to items embeddings of dominant categories, which makes it rather difficult to retrieve items from other categories, and thus limits the diveristy of recommendation.

In our work, we make adjustments to the neighbor discovering process, with special emphasis on category diversification. Specifically, we boost the probability of sampling items from disadvantaged categories and restrict the number of selected items from dominant categories. The rebalanced neighbor discovering algorithm is illustrated in Algorithm 1 and 2. Due to space limit, we omit the descriptions for `GetNeighbors` and `SampleWithProbability` in Algorithm 1, which are simple look-up operations in the adjacency list and random choice with a given distribution. For a user node, we first conduct histogram of item categories to find dominant ones and disadvantaged ones. By taking the inverse of the histogram, we boost the probability of sampling disadvantaged categories and lower the priority for dominant categories. A rebalance weight α is introduced to control the bias. For an item node, we equally treat all the user nodes linked to it and sample its neighbors uniformly. Through rebalanced neighbor discovering, items of more categories are selected, which in turn makes the user embeddings absorb more diverse item embeddings according to the logic of GCN. Therefore, items of superior diversity are recommended by retrieving items from the embedding space with the learned user embedding as the query vector.

Algorithm 1 Rebalanced Neighbor Discovering

INPUT: Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; GCN depth K; seed nodes set $\mathcal{S}$; number of neighbors to sample n; item-category table C; rebalance weight α
OUTPUT: Node Flow $\mathcal{L}$

$\mathcal{L} \leftarrow$ EmptyList()
for $k \leftarrow 1$ *to* K **do**
  $\mathcal{L}^k \leftarrow$ EmptySet()
  for all node $v \in \mathcal{S}$ **do**
    $\mathcal{N}(v) \leftarrow$ GetNeighbors$(\mathcal{G}, v)$
    if v is a user node **then**
      $p \leftarrow$ HistogramAndRebalance$(\mathcal{N}(v), C, \alpha)$
    else if v is an item node **then**
      $p \leftarrow$ UniformDistribution()
    end if
    $\mathcal{L}_v^k \leftarrow$ SampleWithProbability$(\mathcal{N}(v), n, p)$
    $\mathcal{L}^k \leftarrow$ Union$(\mathcal{L}^k, \mathcal{L}_v^k)$
  end for
  append set $\mathcal{L}^k$ to $\mathcal{L}$
end for
return $\mathcal{L}$

Algorithm 2 HistogramAndRebalance

INPUT: User node u's neighbors $\mathcal{N}(u)$; item-category table C; rebalance weight α
OUTPUT: Sample probability over node v's neighbors p

$H \leftarrow$ ComputeCategoryHistogram$(\mathcal{N}(v), C)$
for all node $i \in \mathcal{N}(u)$ **do**
  $p(i) \leftarrow 1/H(C(i))$
  $p(i) \leftarrow p(i)^\alpha$
end for
$p \leftarrow$ Normalize(p)
return p

3.4 Category-Boosted Negative Sampling

One of the main challenges for matching is the so called implicit feedback [45]. That is, only positive samples are accessible to the recommendation system while negative samples are inferred from the uninteracted items. This implicit protocol means negative samples are not necessarily the ones users truly dislike. In practice, negative instances are generated by randomly sampling from those uninteracted items. When training recommendation models, each positive sample is paired with a certain number (i.e. *negative sample rate*) of negative samples. By optimizing with pointwise [29] or pairwise [45] loss function, positive item embeddings are learned to be close to user embeddings, while negative item embeddings are pushed off to the opposite direction.

Several works were proposed to improve the design of the negative sampler [14, 15], aiming at promoting the recommendation accuracy. Nevertheless, few works investigate the potential of negative sampling in diversification. In our work, we propose to choose those *similar but negative* items, which means items of the same category with the positive sample. By sampling negative items from the *positive category*, the recommendation model is optimized to

Algorithm 3 Category-Boosted Negative Sampling

INPUT: Positive samples $\boldsymbol{P}$; item set $\boldsymbol{I}$; negative sample rate T; item-category table C; similar sampling weight β
OUTPUT: Training samples Ω

$\Omega \leftarrow \boldsymbol{P}$
for all positive sample $(u, i, \text{True}) \in \boldsymbol{P}$ **do**
  $\boldsymbol{N} \leftarrow \boldsymbol{I} \setminus i$
  $\boldsymbol{S} \leftarrow \boldsymbol{I}_{C(i)} \setminus i$
  for $t \leftarrow 1$ *to* T **do**
    $r \leftarrow \text{RandomFloat}(0, 1)$
    if $r < \beta$ **then**
      $i_t \leftarrow \text{Sample}(\boldsymbol{S})$
    else
      $i_t \leftarrow \text{Sample}(\boldsymbol{N})$
    end if
    $\Omega \leftarrow \Omega + (u, i_t, \text{False})$
  end for
end for
return Ω

distinguish users' preference within a category. And those negative items in the same category are less likely to be retrieved, which increases the possibility of recommending items from other more diverse categories. The negative sampling strategy is explained in Algorithm 3.

A hyper-parameter β is introduced to manage the proportion of samples from similar items. With more *similar but negative* items in training samples, the representations of user and items are learned in a finer level, which empowers the recommendation system to capture users' interests from more diverse categories. As illustrated in Figure 5, items from positive categories are sampled more as negative instances, which increases the possibility for positive items from negative categories to be recommended and thus more diverse candidates are generated.

3.5 Adversarial Learning

With respect to model training, most recommendation models are trained with a single target concerning accuracy. Though the multi-task framework has been applied in recommendation for multi-behavior modeling [18], relevance of the results still served as the core goal and diversity of recommendation was ignored. With only one optimization object of accuracy, users' category preference is implicitly learned from users' item preference. Taking the same example utilized in Section 2, the recommendation system might learn the user's interests on the whole category (i.e. electronic devices), while fails to distinguish between the user's specific preference on different electronic devices.

Without distillation of the implicit category preference captured in the recommendation model, more items of the positive categories will be recommended, which limits the chance for more diverse items to be exposed to users. Inspired by the progress made in generative models [23, 42], we propose to add an extra adversarial task of item category classification to achieve the goal of distillation and further enhance the diversity. Specifically, we augment the recommendation model with a classifier based on the learned item

Figure 5: An illustration of the sample space. Negative instances are sampled from outside positive items. We propose category-boosted negative sampling which boosts the probability of sampling from items of positive categories (the light green area).

embeddings. We hope the classifier to predict the category of the item from the item embedding as accurate as possible, and expect the recommendation model to generate item embeddings which best *fool* the classifier.

In our experiments, we adopt a fully connected layer as the classifier and use cross entropy loss for optimization. With respect to recommendation, we use log loss [29] which is shown effective in experiments. Take a single training sample (u, i, y, c) as an example, where y is either 0 or 1 represents the user-item interaction groundtruth and c is item i's corresponding category. The loss function for recommendation is formulated as follows:

$$\begin{aligned} \hat{y} &= < \boldsymbol{h}_{\boldsymbol{u}}^{\boldsymbol{K}}, \boldsymbol{h}_{\boldsymbol{i}}^{\boldsymbol{K}} >, \\ L_r(u, i, y) &= -\left[y \cdot \log \sigma(\hat{y}) + (1 - y) \cdot \log \sigma(1 - \hat{y})\right], \end{aligned} \tag{4}$$

where h_u^K and h_i^K are the representations learned by GCN (at the last layer). The loss function for item category classification is:

$$\begin{aligned} \hat{c} &= \boldsymbol{W}\boldsymbol{h}_{\boldsymbol{i}}^{\boldsymbol{K}} \\ L_c(i, c) &= -\hat{c}[c] + \log\Big(\sum_j exp(\hat{c}[j])\Big) \end{aligned} \tag{5}$$

Under the setting of adversarial learning, the object for the item category classifier is to minimize L_c, and the object for the recommendation model is to minimize $L_r - \gamma L_c$, where γ is introduced to balance the main task and the additional adversarial task.

With respect to the classifier, the classification loss is minimized by finding clusters of item embeddings. While for the recommendation model, the classification loss is reversed which pushes item embeddings of the same category far from each other and not to form clusters. Meanwhile, the main task of minimizing the recommendation loss forces the learned embedding space to retain user preference semantics.

In terms of implementation, adversarial learning can be elegantly accomplished by inserting a Gradient Reversal Layer (GRL) in the middle of the back propagation process, which was first introduced in Domain Adaptation Networks (DAN) [17]. We adopt this strategy in our work. Using the same notations of the previous section, we expect the classifier to minimize L_c, while force the GCN to maximize L_c. As illustrated in Figure 6, we insert a GRL in between of the learned item embeddings from GCN and the fully connected classifier. During the back propagation process, the gradients for minimizing the classification loss flow backward through the classifier, and after the GRL, the gradients will be reversed, which further flow to GCN. That is, we perform gradient descent on the parameters of the classifier, while perform gradient ascent on the parameters of GCN, with respect to L_c. For L_r, gradient descent

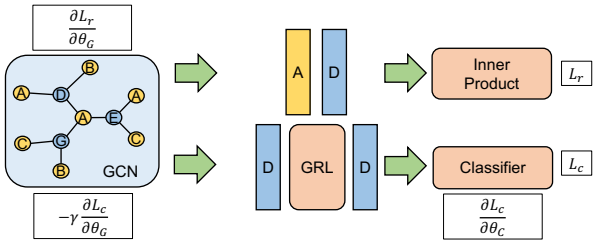


Figure 6: Implementaion of adversarial learning. Gradients are shown in boxes.

is applied to GCN. Through this subtle design, we successfully implement the adversarial learning task.

With the help of adversarial learning, the learned representations of users and items to great extent reserve the item-level interests while squeeze out the category-level interests. Therefore, positive items from negative categories are drawn near to users, and negative items from positive categories are pushed away. Consequently, neighbors in the embedding space will cover items of more diverse categories.

4 EXPERIMENTS

In this section, we conduct experiments to answer the following research questions:

- **RQ1**: How does the proposed method perform compared with other diversified recommendation algorithms?
- **RQ2**: What is the effect of each proposed component in DGCN?
- **RQ3**: How to perform trade-off between accuracy and diversity using DGCN?

4.1 Experimental Settings

4.1.1 Datasets and Evaluation Protocols. To evaluate the performance of our proposed method, we utilize three real-world datasets: Taobao, Beibei and Million Song Dataset (MSD). The three datasets vary in scale and density. Basic statistics of the datasets are summarized in Table 1.

- **Taobao**: This dataset [67, 68] contains the behaviors of users on taobao.com including click, purchase, adding item to shopping cart and item favoring during November 25 to December 03, 2017, which was provided by Alimama[3]. We regard all the aforementioned behaviors as positive samples and randomly select about 10% users with uniform probability. We adopt 10-core settings which means only retaining users and items with at least 10 interactions.
- **Beibei**: This dataset [18] is collected from one of the largest e-commerce platforms[4] in China which records the purchase behaviors during July 1 to July 31, 2017. We also utilize the 10-core settings to guarantee the data quality.
- **MSD**: This dataset [5] contains the listening history for 1M users, and is has been utilized to evaluate diversified music recommendation algorithms [11]. We extract a subset of the dataset and use 10-core settings to filter out inactive entities.

[3]https://tianchi.aliyun.com/dataset/dataDetail?dataId=649
[4]https://www.beibei.com

Table 1: Statistics of the datasets.

dataset	users	items	categories	interactions
Taobao	82633	136710	3108	4230631
Beibei	19140	17196	110	265110
MSD	65269	40109	15	2423262

For each dataset, we first rank the records according to timestamps, then we select the early 60% as training set. We divide the last 40% into two halves. The first 20% used for validation and hyper-parameter search, and we reserve the last 20% for performance comparison. To measure the top-K recommendation performance of our proposed method in consideration of both accuracy and diversity, we utilize a bunch of metrics including recall, hit ratio, coverage, entropy and gini index, while the first two metrics are about accuracy and the last three concerns diversity.

4.1.2 Baselines. To verify the effectiveness of our proposed DGCN, we compare the performance with several diversification methods as follows:

- **MMR** [8]: Maximal Marginal Relevance (MMR) is a pioneer work for diversification in search engines and is further adapted to recommendation systems [69]. This method re-ranks the contents based on greedy algorithms to minimize redundancy.
- **DUM** [3]: This method is also a greedy approach for diversification which aims at maximizing the utility of the items subject to the increase in their diversity.
- **PMF+α+β** [48]: This work formalizes the problem as a combination of three aspects: the relevance of the items, the coverage of the user's interest, and the diversity between them. Two hyper-parameters (α and β) are introduced to balance the three parts.
- **DPP** [11]: Sourced from mathematics and quantum physics, Determinantal Point Process (DPP) is recently leveraged in machine learning research, serving as an parametric model to provide a diverse subset of items from a larger pool of retrieved items. Several methods [11, 19, 21, 22, 55] were proposed to accelerate the computation of DPP.

4.1.3 Parameter Settings. We adopt log loss [29] for all methods and fix the embedding size as 32. The AMSGrad [43] variant of Adam [33] is utilized for optimization. The negative sample rate is set to 4. We train each model until convergence and utilize the early stopping technique to avoid overfitting. We perform grid search to find the best hyper-parameters. Results are averaged over all the users. We implement our proposed method with PyTorch[5], and codes are available at https://github.com/tsinghua-fib-lab/DGCN.

4.1.4 Evaluations. Since our work directly uses inner product to estimate the interaction probability, maximum inner product search can be easily integrated into the system, and we use Faiss [31] to generate candidates for evaluation which greatly reduces the time cost. During evaluation, we construct a search index (IndexFlatIP[6] for efficient nearest neighbor search based on inner product) in

[5]https://pytorch.org
[6]https://github.com/facebookresearch/faiss

Table 2: Overall Performance on Taobao dataset and Beibei dataset. (TopK = 300)

dataset	Taobao					Beibei				
metrics	recall	hit ratio	coverage	entropy	gini index	recall	hit ratio	coverage	entropy	gini index
MMR	0.0544	0.0453	74.5460	3.4931	0.5825	0.1097	0.1036	77.016	4.0184	0.4373
DUM	0.0495	0.0497	126.6621	4.1051	0.4587	0.0746	0.0724	84.3044	4.0389	0.4599
PMF + α + β	0.0473	0.0435	125.5600	4.3725	0.4648	0.1092	0.1054	73.4675	3.7528	0.5127
DPP	0.0633	0.0485	79.1154	3.3904	0.6096	0.0751	0.0745	69.3416	3.7545	0.5078
DGCN	0.0776	0.0783	84.6685	3.5779	0.5583	0.1212	0.1278	71.8546	3.7149	0.5279

Faiss using the learned item embeddings, and feed the user embeddings to the search index as query vectors. Items of the maximum inner products with the query vector will be retrieved, and recommendation metrics are further calculated based on the retrieved items. Moreover,evaluations are conducted in batch style and on GPUs for further acceleration. With the help of efficient nearest neighbor search, we successfully reduce the time cost for evaluation to a few seconds.

4.2 Overall Performance (RQ1)

We compare our proposed method with several baseline algorithms introduced previously. For each baseline method, we tuned it to be on par with our proposed method in one aspect (accuracy or diversity), and compared the effects of the other aspect, on account of the aforementioned accuracy-diversity tradeoff. Since we incorporate diversification into the matching stage, we use relatively high values of topK (300) to measure the performance of matching. Results on the Taobao and Beibei datasets are illustrated in Table 2. From the results, we have several observations:

- **The accuracy-diversity tradeoff exists widely.** In our experiments, three of the baseline methods (MMR, DUM, PMF+α+β) are based on greedy algorithms, and the other one (DPP) is based on a probability model. Comparing across different methods, generally more diverse methods provide less relevant items. For example, PMF+α+β achieves much more diverse results than DPP on Taobao dataset with over 50% relative improvement in terms of coverage, but the accuracy of PMF+α+β is much inferior to DPP. Similarly, DUM achieves the most diverse results on both datasets, however, the relevance of the recommended items by DUM is greatly damaged by diversification.
- **It is more difficult to balance the two aspects for greedy algorithms.** Although there exist certain hyper-parameters in greedy algorithms to balance the weight for accuracy and diversity, the *slope* or exchange rate of the two aspects tends to be rather large. In other words, greedy algorithms turn out to be more aggressive on diversification which makes the accuracy unacceptable. DUM is such an example which usually generates highly diverse results with relatively poor relevance.
- **Our proposed DGCN achieves a better overall performance.** Generally, DGCN generates more diverse items with reasonable relevance. Compared with DPP, our method attains a better performance with respect to both diversity and accuracy on two datasets. In comparison with MMR, our method outperforms on both diversity and accuracy on Taobao dataset, and performs roughly the same with respect to diversity on Beibei dataset, but with better relevance. Though DUM provides extremely diverse items on two datasets, the relevance of the recommended contents is not qualified enough compared with our method. As for PMF+α+β, our method attains much better accuracy on both datasets, and achieves comparable diversity on Beibei dataset.

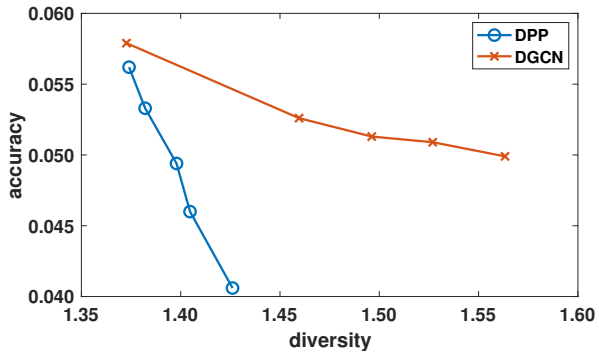


Figure 7: Accuracy-diversity curve of DPP and DGCN on MSD dataset.

To illustrate that our proposed DGCN attains a better overall performance considering both accuracy and diversity, we further conduct experiments on the benchmark MSD dataset, and plot the whole accuracy-diversity curve against the state-of-the-art DPP approach [11]. We tune the tradeoff parameters of DPP and DGCN to obtain recommended items with different accuracy and diversity. Figure 7 demonstrates the results on the MSD dataset. We can observe that the accuracy-diversity curve of the proposed DGCN is closer to the top-right corner than DPP. In other words, conditioned on equal accuracy, DGCN achieves better diversity than DPP. Meanwhile, with comparable diversity, the proposed DGCN can provide much more accurate recommendation. Therefore, the proposed DGCN achieves better overall performance compared with DPP.

4.3 Study on DGCN (RQ2)

In this section, we conduct ablation studies on each of our proposed components in DGCN. We compare the performance of our proposed method with and without the special design on rebalanced neighbor discovering, category-boosted negative sampling and adversarial learning. Table 3 illustrates the results of GCN without diversificaiton, GCN with only one diversification component and GCN with all the three components (DGCN). We also compare the results with DPP, which is widely adopted for diversified recommendation.

Table 3: Ablation study on Taobao dataset.

method	recall	coverage
DPP	0.0633	79.1154
GCN	0.1013	61.9111
Rebalance Neighbor Sampling	0.0939	71.2528
Boost Negative Sampling	0.0954	76.7391
Adversarial Learning	0.0846	79.0722
DGCN	0.0776	84.6685

From the results we can observe that each component alone contributes to improve diversity, and combining the three special designs achieves the most diverse recommendation. Specifically, a single GCN without diversification significantly outperforms DPP with respect to accuracy, however, it sacrifices the diversity which can lead to suboptimal user satisfaction. After we incorporate rebalanced neighbor discovering or category-boosted negative sampling, the diversity of our model gets promoted effectively and the model still maintains a relatively high accuracy. Moreover, combining adversarial learning with GCN achieves comparable diversity with DPP, and the recommended items are much more relevant to users' interests. According to the results in Table 3, our proposed DGCN provides the most diverse contents. In summary, without losing the superior capability of GCN, our proposed DGCN is featured of three special designs for diversification on top of GCN, which greatly improves the diversity and also guarantees the relevance of the recommended items.

We employ adversarial learning to distill users' category preferences from item preferences, aiming to make the learned representations to some extent category-free, which in turn increases the probability of recommending items from more diverse categories. We add an adversarial task of item category classification to fulfill this job. The object of the classifier is to maximize the accuracy of predicting items' categories according the learned item embeddings, while the recommendation model aims to fool the classifier as much as possible. In our experiments, we accomplish the task of adversarial training by inserting a Gradient Reversal Layer (GRL). We compare the performance of our model with and without inserting the GRL. Results are shown in Table 4. Without adversarial learning, the learned item embeddings form clusters where items of the same category are near in the embedding space, which is verified by the high category classification accuracy. Most importantly, the diversity of recommendation is rather poor, leading to the problem of information redundancy. After inserting the GRL to perform adversarial learning on category classification, we distill the category information in the embedding space. Thus it becomes much difficult to predict the category from item embeddings. According to the results, the accuracy of category classification drops drastically from 25% to 6%, which verifies the effect of the distillation process. Most importantly, with the help of adversarial learning, the recommendation diversity improves significantly with a rather acceptable accuracy.

Table 4: Ablation study of adversarial learning on Taobao dataset.

method	RS acc	RS diversity	Classifier acc
GCN w/o GRL	0.1041	55.6591	25%
GCN w/ GRL	0.0739	80.4894	6%

4.4 Trade-off between Accuracy and Diversity (RQ3)

In the proposed framework, we introduce two hyper-parameters, α and β, to control the strength of rebalanced neighbor discovering and category-boosted negative sampling. We now investigate whether these two hyper-parameters can be used to perform trade-off between accuracy and diversity.

4.4.1 ***Rebalanced Neighbor Discovering.*** We conduct experiments to study the effect of rebalanced neighbor discovering. Results of different values of α are illustrated in Figure 8. In neighbor discovering, we boost the probability of sampling from those disadvantaged categories, while limit the chances of those dominant categories. With larger α, we impose stronger boost on disadvantaged categories and perform more forceful rebalance across categories. As shown in Figure 8, the diversity of recommendation increases constantly with the growth of α. Moreover, the accuracy also gets promoted at relatively lower α and finally drops when α becomes to large, which contradicts the previously introduced accuracy-diversity tradeoff. This phenomenon of attaining improvements in both accuracy and diversity has also been observed in related diversification literatures [11, 56], which validates that diversification serves as an effective strategy to enhance user satisfaction.

4.4.2 ***Category-Boosted Negative Sampling.*** Experiments are conducted over different values of β. In our work, we make adjustments to the negative sampling process, where we aim to find those*similar but negative* items. Specifically, we sample from positive categories with probability β which is much larger than the probability by random sampling. By selecting more negative items from positive categories, the learned representations capture users' interest across categories and items of more diverse categories are recommended. Figure 9 illustrates the performance on different β with respect to accuracy and diversity. Similar to the previous experiments on neighbor discovering, the diversity of recommendation improves as we increase the probability of sampling from similar items. In addition, the accuracy is rather stable on small β and decreases when it gets too large. Through category-boosted negative sampling, our proposed DGCN provides more diverse items and guarantees the relevance of the recommended contents.

In summary, we conduct extensive experiments to evaluate our proposed DGCN with special emphasis on diversification. Overall performance on real-world datasets confirms the effectiveness of our method on improving diversity. Ablation studies of DGCN verify the function of each component. Further experiments demonstrate that trade-off between accuracy and diversity can be smoothly performed by tuning the introduced hyper-parameters.

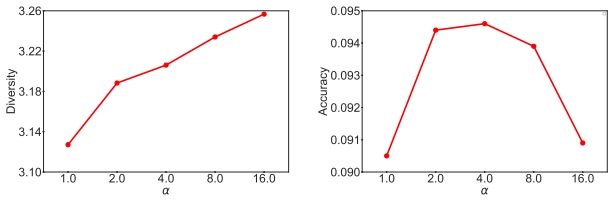


Figure 8: Performance on different α.

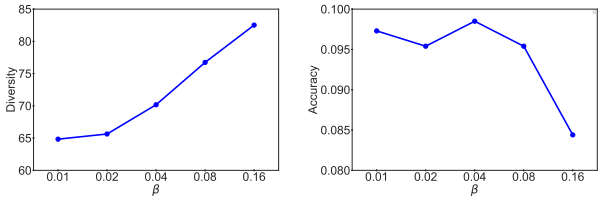


Figure 9: Performance on different β.

5 RELATED WORK

Diversified Recommendation Research on diversification in recommendation was first introduced by Ziegler *et al.* [69], which leveraged a greedy algorithm [8] from the field of information retrieval. After that, a series of post processing methods were proposed to diversify the recommendation results. Qin *et al.* [41] tackled the problem by performing a linear combination of the rating function and a entropy regularizer. Ashkan et al. [3] replaced the weighted sum object in greedy solutions with multiplication, thus removed a tuned parameter for balancing utility and diversity. Sha et al. [48] developed a framework combining relevance, coverage of user interests and diversity. Two hyper-parameters, α and β, were introduced to balance the object. Other than reranking based methods, a series of solutions called learning to rank (LTR) were proposed to generate the recommended list directly. Cheng *et al.* [12] developed a learning-based diversification method by coupling the recommendation model with a structural SVM [51]. Li *et al.*[36] proposed a ranking model and utilized a score function with the form of the product of the estimated interaction probability and category preference. Factorized category features were leveraged for optimization. Recently, Determinantal Point Process (DPP) was introduced to recommendation to generate diverse items. Several algorithms [11, 19, 21, 22, 55, 56] were proposed to reduce the heavy computation of DPP with the help of EM algorithms, greedy algorithms or tensor factorization. Unlike existing works that mainly perform diversification after matching, our proposed method combines diversification and matching with an end-to-end GCN model.

GCN based Recommendation In recent years, Graph Convolutional Networks (GCN) has made great progress in network representation learning tasks including node classification and link prediction [26, 34, 62]. Several works [4, 52–54, 63, 64] have taken advantage of GCN to learn more robust latent representations for users and items in recommendation systems. With more advanced capacity in learning graph representations, GCN has also been shown effective and efficient to be deployed in web-scale recommendation applications [63]. By applying GCN [34] on the user-item interaction bipartite graph, Berg *et al.* [4] transformed the matrix completion task in recommendation to link prediction on the graph. Ying *et al.* [63] further extended the inductive learning idea introduced in [26] to practical recommendation scenario, and proposed a scalable algorithm called PinSage. Both offline and online evaluations confirm the effectiveness of GCN in modeling user preference. Wang *et al.* [54] developed a framework which performs embedding propagation on the user-item integration graph based on GCN to model the high-order connectivity. Experimental results illustrated that the accuracy of recommendation has been successfully improved by utilizing GCN to perform representation learning. However, how diversity is impacted by the complicated GCN model remains uncertain. In our work, we focus on diversified recommendation with the help of GCN.

6 CONCLUSION

In this work, we investigated existing diversification solutions and pointed out the challenge that the decoupled design of diversification and matching could lead to suboptimal performance. Based on our analysis, we aimed to push the diversification process upwards into the matching stage, and proposed an end-to-end diversified recommendation model based on GCN with several special designs on diversity. We conducted extensive experiments on real-world datasets. Experimental results validated the effectiveness of our proposed method on improving diversity. Further ablation studies confirmed that our proposed DGCN provides diverse and relevant contents to meet users' needs.

Although the accuracy-diversity tradeoff still exists in our proposed method, it has been shown that improving the diversity does not necessarily lead to inferior accuracy [11, 50, 56]. Especially in online scenarios, promoting the diversity of the recommended contents yields substantial increases in user engagement [56]. The conflict of diversity and accuracy to some extent results from the differences between offline evaluation and online evaluation, as well as the causality of the system [47], which we believe are quite interesting and important research questions. One step further, distinct users might regard diversity differently, and the diversification process might also be personalized (*i.e.* personalized personalization) [1], which is also a promising future direction.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under 61941117, U1936217, 61971267, 61972223, 61861136003.

REFERENCES

[1] Xavier Amatriain and Justin Basilico. 2016. Past, present, and future of recommender systems: An industry perspective. In *Proceedings of the 10th ACM conference on recommender systems*. 211–214.
[2] Arda Antikacioglu and R Ravi. 2017. Post processing recommender systems for diversity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 707–716.
[3] Azin Ashkan, Branislav Kveton, Shlomo Berkovsky, and Zheng Wen. 2015. Optimal greedy diversity for recommendation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
[4] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263* (2017).

[5] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. 2011. The Million Song Dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*.

[6] Rubi Boim, Tova Milo, and Slava Novgorodov. 2011. Diversification and refinement in collaborative filtering recommender. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 739–744.

[7] Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI symposium on Principles of Database Systems*. ACM, 155–166.

[8] Jaime G Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries.. In *SIGIR*, Vol. 98. 335–336.

[9] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 335–344.

[10] Li Chen, Wen Wu, and Liang He. 2016. Personality and recommendation diversity. In *Emotions and Personality in Personalized Services*. Springer, 201–225.

[11] Laming Chen, Guoxin Zhang, and Eric Zhou. 2018. Fast greedy map inference for determinantal point process to improve recommendation diversity. In *Advances in Neural Information Processing Systems*. 5622–5633.

[12] Peizhe Cheng, Shuaiqiang Wang, Jun Ma, Jiankai Sun, and Hui Xiong. 2017. Learning to recommend accurate and diverse items. In *Proceedings of the 26th international conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 183–192.

[13] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. ACM, 191–198.

[14] Jingtao Ding, Fuli Feng, Xiangnan He, Guanghui Yu, Yong Li, and Depeng Jin. 2018. An improved sampler for bayesian personalized ranking by leveraging view data. In *Companion Proceedings of the The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 13–14.

[15] Jingtao Ding, Yuhan Quan, Xiangnan He, Yong Li, and Depeng Jin. 2019. Reinforced Negative Sampling for Recommendation with Exposure Data. IJCAI.

[16] Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. 2018. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 1775–1784.

[17] Yaroslav Ganin and Victor Lempitsky. 2014. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495* (2014).

[18] Chen Gao, Xiangnan He, Dahua Gan, Xiangning Chen, Fuli Feng, Yong Li, Tat-Seng Chua, and Depeng Jin. 2018. Learning recommender systems from multi-behavior data. *arXiv preprint arXiv:1809.08161* (2018).

[19] Mike Gartrell, Ulrich Paquet, and Noam Koenigstein. 2017. Low-rank factorization of determinantal point processes. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[20] Guillaume Gautier, Guillermo Polito, Rémi Bardenet, and Michal Valko. 2018. DPPy: Sampling Determinantal Point Processes with Python. *ArXiv e-prints* (2018). arXiv:1809.07258 http://arxiv.org/abs/1809.07258 Code at http://github.com/guilgautier/DPPy/ Documentation at http://dppy.readthedocs.io/.

[21] Jennifer Gillenwater, Alex Kulesza, Zelda Mariet, and Sergei Vassilvtiskii. 2019. A Tree-Based Method for Fast Repeated Sampling of Determinantal Point Processes. In *International Conference on Machine Learning*. 2260–2268.

[22] Jennifer A Gillenwater, Alex Kulesza, Emily Fox, and Ben Taskar. 2014. Expectation-maximization for learning determinantal point processes. In *Advances in Neural Information Processing Systems*. 3149–3157.

[23] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[24] Mihajlo Grbovic and Haibin Cheng. 2018. Real-time personalization using embeddings for search ranking at airbnb. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 311–320.

[25] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).

[26] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[27] Ruining He and Julian McAuley. 2016. VBPR: visual bayesian personalized ranking from implicit feedback. In *Thirtieth AAAI Conference on Artificial Intelligence*.

[28] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 355–364.

[29] 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*. International World Wide Web Conferences Steering Committee, 173–182.

[30] Zhengshen Jiang, Hongzhi Liu, Bin Fu, Zhonghai Wu, and Tao Zhang. 2018. Recommendation in heterogeneous information networks based on generalized random walk model and bayesian personalized ranking. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 288–296.

[31] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734* (2017).

[32] Komal Kapoor, Vikas Kumar, Loren Terveen, Joseph A Konstan, and Paul Schrater. 2015. I like to explore sometimes: Adapting to dynamic user novelty preferences. In *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM, 19–26.

[33] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[34] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[35] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 426–434.

[36] Shuang Li, Yuezhi Zhou, Di Zhang, Yaoxue Zhang, and Xiang Lan. 2017. Learning to Diversify Recommendations Based on Matrix Factorization. In *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*. IEEE, 68–74.

[37] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1754–1763.

[38] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 1 (2003), 76–80.

[39] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[40] Maxim Naumov, Dheevatsa Mudigere, Hao-Jun Michael Shi, Jianyu Huang, Narayanan Sundaraman, Jongsoo Park, Xiaodong Wang, Udit Gupta, Carole-Jean Wu, Alisson G Azzolini, et al. 2019. Deep Learning Recommendation Model for Personalization and Recommendation Systems. *arXiv preprint arXiv:1906.00091* (2019).

[41] Lijing Qin and Xiaoyan Zhu. 2013. Promoting diversity in recommendation by entropy regularizer. In *Twenty-Third International Joint Conference on Artificial Intelligence*.

[42] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[43] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. 2019. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237* (2019).

[44] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International Conference on Data Mining*. IEEE, 995–1000.

[45] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[46] Badrul Munir Sarwar, George Karypis, Joseph A Konstan, John Riedl, et al. 2001. Item-based collaborative filtering recommendation algorithms. *Www* 1 (2001), 285–295.

[47] David Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-Francois Crespo, and Dan Dennison. 2015. Hidden technical debt in machine learning systems. In *Advances in neural information processing systems*. 2503–2511.

[48] Chaofeng Sha, Xiaowei Wu, and Junyu Niu. 2016. A Framework for Recommending Relevant and Diverse Items.. In *IJCAI*. 3868–3874.

[49] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[50] Idan Szpektor, Yoelle Maarek, and Dan Pelleg. 2013. When relevance is not enough: promoting diversity and freshness in personalized question recommendation. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1249–1260.

[51] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of machine learning research* 6, Sep (2005), 1453–1484.

[52] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. 2019. Knowledge graph convolutional networks for recommender systems. In *The World Wide Web*

Conference. ACM, 3307–3313.

[53] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. KGAT: Knowledge Graph Attention Network for Recommendation. In *KDD*. 950–958.

[54] 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*. 165–174.

[55] Romain Warlop, Jérémie Mary, and Mike Gartrell. 2019. Tensorized Determinantal Point Processes for Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1605–1615.

[56] Mark Wilhelm, Ajith Ramanathan, Alexander Bonomo, Sagar Jain, Ed H Chi, and Jennifer Gillenwater. 2018. Practical diversified recommendations on youtube with determinantal point processes. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 2165–2173.

[57] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Npa: Neural news recommendation with personalized attention. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2576–2584.

[58] Chuhan Wu, Fangzhao Wu, Mingxiao An, Yongfeng Huang, and Xing Xie. 2019. Neural News Recommendation with Topic-Aware News Representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 1154–1159.

[59] Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr, Christopher Fifty, Tao Yu, and Kilian Q Weinberger. 2019. Simplifying graph convolutional networks. *arXiv preprint arXiv:1902.07153* (2019).

[60] Qiong Wu, Yong Liu, Chunyan Miao, Yin Zhao, Lu Guan, and Haihong Tang. 2019. Recent Advances in Diversified Recommendation. *arXiv preprint arXiv:1905.06589* (2019).

[61] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 346–353.

[62] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826* (2018).

[63] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 974–983.

[64] Yu Zheng, Chen Gao, Xiangnan He, Yong Li, and Depeng Jin. 2020. Price-aware recommendation with graph convolutional networks. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 133–144.

[65] Guorui Zhou, Xiaoqiang Zhu, Chenru 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*. ACM, 1059–1068.

[66] Tao Zhou, Zoltán Kuscsik, Jian-Guo Liu, Matúš Medo, Joseph Rushton Wakeling, and Yi-Cheng Zhang. 2010. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences* 107, 10 (2010), 4511–4515.

[67] Han Zhu, Daqing Chang, Ziru Xu, Pengye Zhang, Xiang Li, Jie He, Han Li, Jian Xu, and Kun Gai. 2019. Joint Optimization of Tree-based Index and Deep Model for Recommender Systems. *arXiv preprint arXiv:1902.07565* (2019).

[68] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning Tree-based Deep Model for Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1079–1088.

[69] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*. ACM, 22–32.