A Gain-Tuning Dynamic Negative Sampler for Recommendation

Qiannan Zhu
zhuqiannan@ruc.edu.cn
Gaoling School of Artificial Intelligence, Renmin University of China. Beijing Key Laboratory of Big Data Management and Analysis Methods
Beijing, China

Haobo Zhang
2018200680@ruc.edu.cn
School of Information, Renmin University of China. Beijing Key Laboratory of Big Data Management and Analysis Methods
Beijing, China

Qing He
2019200172@ruc.edu.cn
School of Finance, Renmin University of China. Beijing Key Laboratory of Big Data Management and Analysis Methods
Beijing, China

Zhicheng Dou*
dou@ruc.edu.cn
Gaoling School of Artificial Intelligence, Renmin University of China. Beijing Key Laboratory of Big Data Management and Analysis Methods
Beijing, China

ABSTRACT

Selecting reliable negative training instances is the challenging task in the implicit feedback-based recommendation, which is optimized by pairwise learning on user feedback data. The existing methods usually exploit various negative samplers (i.e., heuristic-based or GAN-based sampling) on user feedback data to improve the quality of negative samples. However, these methods usually focused on maintaining the hard negative samples with a high gradient for training, causing the false negative samples to be selected preferentially. The limitation of the false negative noise amplification may lead to overfitting and further poor generalization of the model. To address this issue, we propose a Gain-Tuning Dynamic Negative Sampling GDNS to make the recommendation more robust and effective. Our proposed model designs an expectational gain sampler, concerning the expectation of user' preference gap between the positive and negative samples in training, to guide the negative selection dynamically. This gain-tuning negative sampler can effectively identify the false negative samples and further diminish the risk of introducing false negative instances. Moreover, for improving the training efficiency, we construct positive and negative groups for each user in each iteration, and develop a group-wise optimizer to optimize them in a cross manner. Experiments on two real-world datasets show our approach significantly outperforms state-of-the-art negative sampling baselines.

CCS CONCEPTS

• **Information systems** → **Collaborative filtering**; • **Computing methodologies** → **Learning from implicit feedback**.

*The corresponding author

WWW '22, April 25–29, 2022, Virtual Event, Lyon, France.

ACM ISBN 978-1-4503-9096-5/22/04…$15.00
https://doi.org/XXXXXXX.XXXXXXX

KEYWORDS

Recommendation system, Negative sampler, Collaborative filtering

ACM Reference Format:
Qiannan Zhu, Haobo Zhang, Qing He, and Zhicheng Dou. 2022. A Gain-Tuning Dynamic Negative Sampler for Recommendation. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Recommendation system plays an important role to alleviate information overload problem and can help users to obtain useful information [7, 14, 17, 18, 20–22, 32]. Collaborative filtering (CF), as the core technique of the recommendation system, aims to learn user's potential preference from the observed user-item interactions and recommend items for users [9, 15, 23, 27]. Today's observed user-item interactions are much easier to collect from the implicit user feedback, such as purchasing, clicking, browsing, and watching etc. Each observed interaction normally indicates a user's interest in an item, which is regarded as the positive sample in recommendation. With such positive interactions, the CF model usually selects a few instances from the unobserved data as negative samples, and optimizes the positive instances with high scores and negative instances with low scores for learning user's potential preference.

Obviously, selecting reliable negative instances from user's implicit feedback plays a critical role in training recommender models. The common approach [4, 27] is to uniformly sample negative instances from the unobserved data. This uniform sampling is widely used in CF methods because of its simple and easy extension, but it suffers from the vanishing gradient as the gap between positive and negative samples is too large to provide valuable information for the latter training. To overcome the vanishing gradient problem, many methods focusing on hard negative sampling are proposed, aiming at mining the hard negative samples that have a high probability of being positive. The main idea of these methods is to design another sampling distribution to replace the uniform sampling distribution for obtaining the high-quality negative samples. For example, the typical heuristic-based methods [26, 34, 35] generate hard negative instances by item attribution (i.e., popularity, frequency) during the

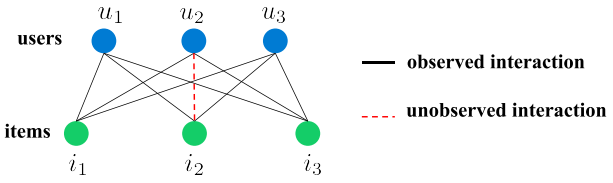

Figure 1: Illustration of the user-item interactions. The blue solid lines are the observed interactions and the red dotted lines are the unobserved interactions.

training, and the GAN-based methods [1, 24, 30] use complicate structures to generate the adversarial negative instances.

All methods mentioned above focus on finding the hard negative samples and overlook the risk of introducing false negative instances in the training stage. Factually, the false negative instances are not necessarily the user's dislike. On the contrary, they may be the user's potential interest. Given an example in Figure 1, under the collaborative assumption that users with similar interaction histories share the similar interest, the user u_2 has the larger probability to like the unobserved item i_2. While, the above negative sampling methods are prone to select the unobserved instance (u_2, i_2) as the hard negative sample because it is difficult for learning. Forcibly optimizing these false negative samples usually leads to overfitting and makes the model lose its generalization ability. Recently, state-of-the-art method SRNS [11] leverages the low variance of the false negative samples, and adds their historical variance of iterations to loss term for preventing sampling from the false negative samples. However, the simple addition combination is not enough to distinguish the false negative samples because the loss of the false negative samples is too large to be covered by the variance.

To address this issue, we propose a novel gain-tuning dynamic negative sampling model GDNS for recommendation, which can perform negative sampling efficiently and robustly. GDNS develops a gain-aware negative sampler to prefer real negative samples dynamically, and group-wise optimizer to boost performance by organizing the training instances in group formal. Specifically, in t-th iteration, the gain-aware negative sampler first obtain the expectation of the score gap between positive and negative samples, and then uses the expectational gain between $t-1$-th and t-th iteration as an indicator to distinguish the false negative sample from candidate negative samples. After selecting the real-like negative samples, the group-wise optimizer constructs positive and negative groups for each user in each epoch, and optimize them in a cross manner to improve the training efficiency. We empirically conduct extensible experiments on two real-world datasets Movielens-1m and Pinterest. The comparison results of the various measures on the two datasets show that our model achieves State-of-the-Art performance.

In summary, the contributions of this paper are as follows:

- We propose a gain-tuning dynamic negative sampling model GDNS for recommendation, which can select the reliable negative samples for training dynamically.
- We develop expectation gain driven indicator to efficiently distinguish the false negative samples with the goal of improving the generalization and robustness.

	Optimization	Robustness	Indicator
ENMF	NonSampling	×	-
Uniform	Pair-wise	×	Uniform
NNCF	Pair-wise	×	Hard-Based
AOBPR	Pair-wise	×	Hard-Based
IRGAN	Pair-wise	×	Hard-Based
RNS-AS	Pair-wise	×	Hard-Based
AdvIR	Pair-wise	×	Hard-Based
SRNS	Pair-wise	√	Hard-Var.
Ours	Group-wise	√	Gain-Based

Table 1: Model comparison with baselines.

- We design a group-aware optimization by constructing positive and negative groups for each user in each iteration, which can enhance training efficiency.

2 PRELIMINARY

Collaborative filtering based recommendation focuses on learning users' preference from the observed user-item interactions. Today, the observed user-item interactions are much easier to collect from implicit user feedback, which is prevalent in boosting the development of recommender systems.

For the implicit CF methods, we denote the user set as $\mathcal{U}$, item set as $\mathcal{I}$, and the exposure user-item data X as

$$X_{u,i} = \begin{cases} 1, & if\ (u,i)\ is\ observed. \\ 0, & otherwise. \end{cases} \tag{1}$$

where $X_{u,i} = 1$ represents that there is an observed interaction between user u and item i, which is regarded as a positive instance that the user u is interested in the item i. Otherwise, $X_{u,i} = 0$ is the unobserved interaction between the user u and item i, which does not necessarily imply the user dislike the item i. It can be considered as the user ignored the item because of its displayed position or form. Formally, the goal of the CF methods is to learn a matching function $\mathcal{U} \times \mathcal{I} \rightarrow X$ to calculate the relevance between user $u \in \mathcal{U}$ and item $i \in \mathcal{I}$. Typically, a generalized matrix factorization framework GMF [16] is used as the basic model, whose matching function on the given pair of user u and item i is:

$$r_{u,i} = W^\top (P_u \odot Q_i) = \sum_{k=1}^{d} w_k \cdot p_{u,k} \cdot q_{i,k}, \tag{2}$$

where W is a learnable parameter vector, P_u and Q_i are embedding representations of users and items, $\odot$ represents element-wise multiplication between two vectors. Also, there exist other strong models like [28, 29, 36] that have significantly achieved promising recommendation performance.

To learn an implicit CF model, a marginal hinge loss [33] is proposed, which assigns positive samples with higher scores and negative ones with lower scores. The learning objective can be formulated as minimizing following loss function, i.e.,

$$L(u,i,j) = |r_{u,j} - r_{u,i} + \gamma|_+ \tag{3}$$

where $|\cdot|_+ = \max(x, 0)$ means to get the maximum number between 0 and x, γ is a hyperparameter indicating the expectational margin between the negative and positive samples. $r_{u,i}$ and $r_{u,j}$ are the

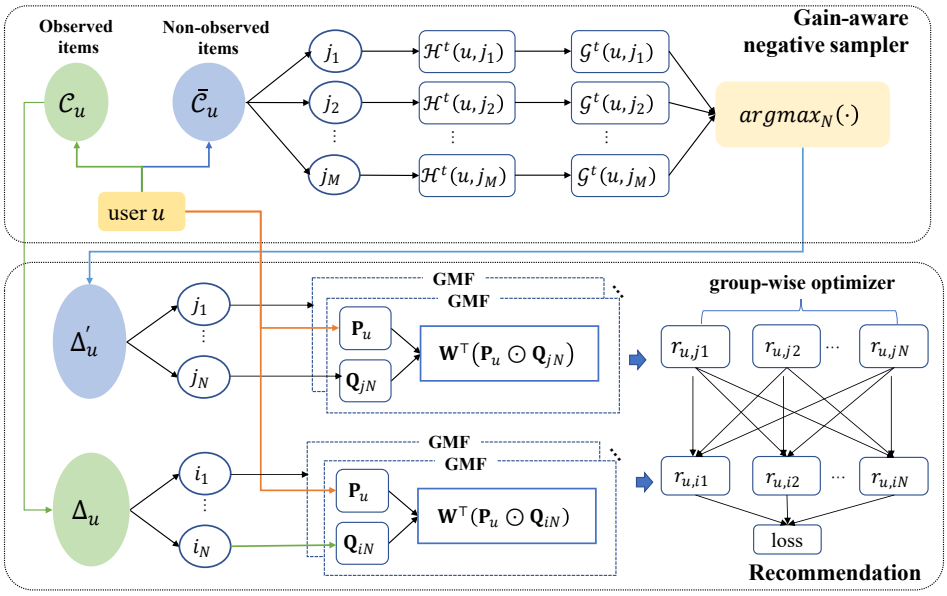

Figure 2: The model architecture of GDNS.

relevance scores of the positive instance (u, i) and negative instance (u, j), respectively. For the user u, the positive instance i is randomly chosen from the observed interaction $C_u = \{i \,|X_{u,i} = 1\}$, and the negative sample j is generated from the negative sampler $\phi(\cdot|u)$ based on the unobserved interaction $\bar{C}_u = \{j \,|X_{u,j} = 0\}$ of the user u. During the training process, the implicit CF methods can benefit more from the negative instances with better quality. In the previous work, the uniform distribution [27] is the most widely used negative sampler, suffering from low quality of negative samples. To address the problem, many menthods [11, 25] attempt to sample much harder instances, containing more information to minimize the loss function in equation (3). Nevertheless, these menthods have not enough robustness to handle false negative instances in training. Compared with the existing negative sampling methods, we focus on developing a robust negative sampler to generate reliable hard negative instances, which can provide more valuable gradient information in training stage and help to perform better on modeling user preference. In this paper, we summary the previous works of negative sampling in Table 1, which discussed the inherent difference between our model and these works.

3 METHODOLOGY

To improve the robustness and efficiency for negative sampling in implicit CF, under the deep understanding of the characteristics of the negative instances, we develop a gain-aware negative sampler to make the sampling more efficient and robust, and employ a novel group-wise optimizer to improve the training efficiency. The framework of our model is illustrated in Figure 2.

3.1 Gain-aware Negative Sampler

Given the risk of introducing false negative instances, the quality of negative samples needs to be measured in a more reliable way. In this section, we design a negative sampler that aims to learn a negative sampling distribution for generating reliable hard negative instances. Intuitively, it is reasonable to generate real negative instances from exposure data, as it records whether a user interacted with an item. Thus, we devise an exposure-aware function to measure negative signals in the exposure data. Specifically, for a user u with his interacted item set $\Delta_u \subseteq C_u$, the probability of the non-interacted item j in exposure data being a real negative sample is:

$$\mathcal{H}^t(u, j) = \mathbb{E}_{i\sim\Delta_u}\sigma(r_{u,j} - r_{u,i}) \tag{4}$$

where t represents the t-th iteration in the training stage and σ is sigmoid function. In this way, the generated negative samples are given higher prediction scores, getting close with those of positive instances, which can effectively provide larger gradients and more information during training.

Importantly, we empirically find that negative instances with large prediction scores are important for the model's learning but generally rare, i.e., following a skewed distribution. A more novel finding is that the false negative instances always have high expectation gap over many iterations of training (analyzed in section 4.6), which provides a new angle on tackling false negative problem remained in the existing approaches. Thus, compared with the expectation of the score difference between the negative and positive samples in equation (4), the expectational gain between two iterations is a more sensitive signal to monitor the difference between the negative sample and the positive item set. Correspondingly, we develop a gain-aware function to calculate the probability of item j being a real negative sample, i.e.,

$$\mathcal{G}^t_{u,j} = \alpha \cdot \mathcal{G}^{t-1}_{u,j} + (1-\alpha)\cdot\sigma\left(\frac{\mathcal{H}^{t-1}_{u,j} - \mathcal{H}^t_{u,j}}{\mathcal{H}^t_{u,j} + \epsilon}\right) \tag{5}$$

where α is a smoothing hyperparameter to make the training stable, ϵ is a small number to prevent the situation that is $\mathcal{H}^t_{u,j} = 0$. According to the gain-tuning sampler, the unobserved interactions of a given user with higher $\mathcal{G}^t_{u,j}$ are selected as the negative samples for model's optimization. This is because that the false negative samples usually have the lower $\mathcal{G}^t_{u,j}$ than the true negative samples as discussed in section 4.6. Hence, our model tends to leverage this high-gap expectation based criterion to reliably measure the quality of negative samples, i.e., selecting the candidate negative samples with higher $\mathcal{G}$ as the negative samples in training stage. This way can reduce the risk of introducing the false negative instances effectively.

3.2 Group-wise Optimizer

Previous works usually optimize the pair-wise margin-based loss as equation (3) to assign high scores to the positive instances and low scores to the negative instances. Considering (1) there is a high probability that the negative item j may be reused with the positive items of the user u in training, and (2) it is inefficient to optimize the recommendation model on a pair of positive and negative samples, we develop a group-wise optimizer to make the optimization more efficient. The group-wise ranking loss is defined as follows:

$$\mathcal{L}(u, \Delta_u, \Delta'_u) = \sum_{i \in \Delta_u} \sum_{j \in \Delta'_u} |r_{u,j} - r_{u,i} + \gamma|_+ \tag{6}$$

where the matching score $r_{u,i}$ and $r_{u,j}$ are calculated by typical GMF [16] as equation (2), Δ_u and Δ'_u are positive and negative groups for each user constructed at the beginning of each iteration. As Algorithm 1 described, given a user $u \in \mathcal{U}$, the positive group $\Delta_u = \{i_1, \dots i_N\}$ are constructed by sampling N times independently from C_u while the negative group $\Delta'_u = \{j_1, \dots j_N\}$ is constructed by selecting more reliable negative samples from $\bar{C}_u$. As the space of negative sampling is extremely huge or unknown in the real world, for each user in t-th iteration, we first construct a subset $\delta = \{j_1, \dots j_M\} \subset \bar{C}_u$ each epoch by randomly selecting M negative samples from the whole negative sample space in step 9 of Algorithm 1. Then we use the gain-tuning negative sampler to select the top N ones for building Δ'_u according to their $\mathcal{G}^{t-1}_{u,j}$. By filtering out samples with high $\mathcal{G}^{t-1}_{u,j}$, our model is expected to reduce the effect of false negative samples and further improve the performance.

Having established the group-wise ranking loss, users' preferences can be caught more effectively by the group-aware positive and negative samples than the single pairwise samples. Finally, we formulate the learning objective as minimizing following loss function, i.e.,

$$\mathcal{L} = \sum_{u \in \mathcal{U}} \mathcal{L}(u, \Delta_u, \Delta'_u) \tag{7}$$

Although our model assigns a group of positive and negative items for each user in the training stage, we still use the same evaluation protocol as baselines in the test stage. Concretely, for each user-item pair and its candidate negative items in the test set, we feed them into our model to obtain their ranking scores, and further sort all items in descending order for evaluating the performance based on the recommendation metrics.

Algorithm 1 GDNS Sampling Algorithm

Input: User set $\mathcal{U}$, Item set $\mathcal{I}$, interaction matrix X, Group size N, embedding dimension d, $n = |\mathcal{U}|$, $m = |\mathcal{I}|$
Output: P, Q, W

1: Initialize $P \in R^{m \times d}$, $Q \in R^{n \times d}$, $W \in R^d$
2: Initialize $\mathcal{H}^0 = 1^{m \times n}$, $\mathcal{G}^0 = 1^{m \times n}$
3: Construct C_u and $\bar{C}_u$ for each user u based on the interaction matrix X.
4: **for** $t \leftarrow 1$ to T epochs **do**
5: sample a mini-batch of users B from $\mathcal{U}$
6: **for** each $u \in B$ **do**
7: $\mathcal{L} \leftarrow 0$
8: $\Delta_u \leftarrow \{i_1, \dots, i_N\}$, i drawn from C_u
9: $\delta \leftarrow \{j_1, \dots, j_M\}$, j drawn from $\bar{C}_u$
10: $\Delta'_u \leftarrow \{j_1, \dots, j_N\}$, j is top N of δ on $\mathcal{G}^{t-1}_{u,j}$
11: calculate $r_{u,i} = W^\top (P_u \odot Q_i), i \in \Delta_u$
12: calculate $r_{u,j} = W^\top (P_u \odot Q_j), j \in \Delta'_u$
13: $\mathcal{H}^t_{u,j} \leftarrow 0, j \in \Delta'_u$
14: **for** each $(i, j) \in \Delta_u \times \Delta'_u$ **do**
15: $\mathcal{L} \leftarrow \mathcal{L} + L(u, i, j)$
16: $\mathcal{H}^t_{u,j} \leftarrow \mathcal{H}^t_{u,j} + \sigma(r_{u,j} - r_{u,i})$
17: **for** each $j \in \Delta'_u$ **do**
18: $\mathcal{G}^t_{u,j} = \alpha \mathcal{G}^{t-1}_{u,j} + (1 - \alpha) \cdot \sigma\left(\frac{\mathcal{H}^{t-1}_{u,j} - \mathcal{H}^t_{u,j}}{\mathcal{H}^t_{u,j} + \epsilon}\right)$
19: Backward $\mathcal{L}$ to update the P, Q and W by Backward Propagation algorithm
20: **return** P, Q, W

Why the Group-wise optimizer works? Essentially, different from isolated gradient information in the pair-wise approach, the group-wise optimizer shares the gradient information among all samples in the same group. For example, the item i_1 in the right of Figure 3 can acquire necessary gradient by the cross operation in the same groupfor improving the embedding in our model. On the contrary, the item i_1 in the left of Figure 3 can only obtain the gradient information from the pair $r_{u_1,i_1} - r_{u_1,j_1}$ on the pair-wise approaches. Moreover, the group-wise organization can boost the training efficiency with reducing computation operation, which is analyzed in following two aspects:

- This group-wise ranking loss can be easily reduce the consumption of feature encoding in the popular recommender works [8], which is restricted by feature encoding on the large volume of users and items. Taking the Movielens-1m dataset described in section 4 as an example, for the traditional pair-wise approach, the model has to encode representations of 563186 users and $563186 * 2$ negative and positive items for all 563186 training instances in each iteration. While with the group-wise optimizer, our model only needs to encode small representations of 6028 user and $6028 * N * 2$ items, where N is the group size and is set as 32 experimentally.
- Unlike the pair-wise ranking loss that constructs negative samples for each observed interaction in C, our group-wise ranking loss constructs the group of positive and negative

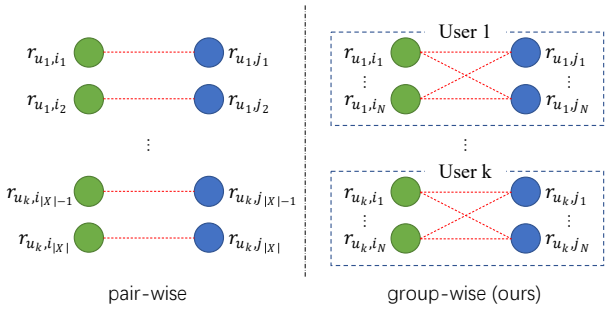

Figure 3: Comparision between pair-wise and group-wise optimization.

samples for each user. Such two loss make the construction operation as $|C|$ and $|\mathcal{U}| \times N^2$. In the real dataset, the number of observed interactions $|C|$ is much greater than the number of users $|\mathcal{U}|$. For example, there is about 93 times in Movielens-1M and 25 times in Pinterest. By sampled group Δ_u and Δ'_u instead of the whole space C and $\bar{C}$, the group-wise organization can significantly reduce computational cost to achieve convergence.

4 EXPERIMENT

In this section, we study and evaluate our approach on the Top-K recommendation task with two real-world datasets. Further, we give a deep insight into experimental results, and conduct controlled experiments with synthetic noise, so as to investigate GDNS's robustness to distinguish the false negative instances (section 4.6).

4.1 Dataset

Following the typical baselines, we evaluate our model on two popular real-world datasets in recommendation field.

- **Movielens-1m**. It contains the user's rating information on movies obtained from IMDB and Movie DataBase[1]. In our experiments, the user's ratings on movies are converted to implicit feedback data. Concretely, records with 4-rated and 5-rated are treated as positive labels while low-rated and unobserved items as negative labels [25, 31].
- **Pinterest**. It is an image sharing and social media service [2]. The Pinterest dataset contains more than 1 million images associated with users who have "pinned" them. In our experiments, user's "pinned" interactions on images are treated as positive labels while others as negative labels.

To make the comparison fair, we follow the same preprocessing settings as baselines [11] to process the datasets. First, we filter out the users or items with less than 4 records. This is a common way to alleviate the high sparsity problem of rating datasets. Second, for each user, we choose one item for validation and test respectively, and the remaining as the positive items for training. The basic statistic of datasets is shown in table 2.

[1]https://grouplens.org/datasets/movielens/1m
[2]https://sites.google.com/site/xueatalphabeta/academic-projects

Dataset	**Movielens-1m**	**Pinterest**
#User	6,028	55,187
#Item	3,533	9,916
#Train	563,186	1,390,435
#Valid	6,028	55,187
#Test	6,028	55,187

Table 2: Dataset statistics.

4.2 Baselines

To evaluate the performance of our model, we select eight typical negative sampling models as our baselines in table 1, including:

- NNCF [5], BPR [27], AOBPR [26]. Those methods generate samples with a fixed negative sampling distribution.
- ENMF [3]. This method directly treats all non-observed instances as negative samples. Due to computing all negative samples in each epoch, it is a highly competitive baseline for item recommendation.
- RGAN [31], RNS-AS [10], AdvIR [25]. Those GAN-based approaches are able to generate hard negative samples by training a external GAN module to capture the negative sampling distribution.
- SRNS [11]. This approach caches the hard negative samples each epoch to be reused. To avoid oversampling hard negative instances, SRNS considered the historical score variance of candidates negative samples during sampling.

4.3 Evaluation Protocol

In the testing phase, the recommendation methods trained with different negative samplers are asked to rank a given item list for each user. As the space of negative sampling is extremely huge or unknown in the real world, for each one positive sample in the test set, we fix the number of its relevant negative items as 100 followed by baselines [11, 16, 19]. Here, the test set provided by SRNS [11] is used to make the comparison fair. Concretely, the scoring function in equation (2) is used to calculate the scores of each test user-item pair (u, i) and its relevant negative user-item pairs $\{(u, j_k), k \in [1, 100]\}$. Then these scores are ranked in descending order to obtain the rank of the positive test sample (u, i). Finally, two metrics are employed to compare our model with baselines: (1) Recall@K. It is the proportion of correct items ranked in the top K. (2) NDCG@K. It is the Normalized Discounted Cumulative Gain in the top K. Here, we evaluate our method on metrics Recall@K and NDCG@K with $K = \{1, 3\}$. In this paper, we just report the results of $NDCG@1$, $NDCG@3$, and $Recall@3$ as $NDCG@1$ and $Recall@1$ are equivalent mathematically.

4.4 Settings

In training stage, we select the learning rate $\mu \in \{0.01, 0.03, 0.09, 0.3, 1.0\}$, the margin $\gamma \in \{0, 1, 2\}$, the smoothing hyperparameter $\alpha \in \{0.1, 0.2, 0.5, 0.7, 0.9\}$, the group size $N \in \{4, 8, 16, 32, 64, 128\}$. Specifically, the hyperparameter configuration for best-performing are selected based on NDCG@3 by grid search with early stop trick as follows: the learning rate μ is 0.09, the margin γ is 1, the smoothing hyperparameter α is 0.2, and the group size N is 32. The batch

Method	Movielens-1m			Pinterest		
	NDCG@1	NDCG@3	Recall@3	NDCG@1	NDCG@3	Recall@3
ENMF [3]	0.1846	0.3021	0.3882	0.2594	0.4144	0.5284
Uniform [27]	0.1744	0.2846	0.3663	0.2586	0.4136	0.5276
NNCF [5]	0.0829	0.1478	0.1971	0.2292	0.3699	0.4735
AOBPR [26]	0.1802	0.2905	0.3728	0.2596	0.4165	0.5319
IRGAN [31]	0.1755	0.2877	0.3708	0.2587	0.4143	0.5282
RNS-AS [10]	0.1823	0.2932	0.3754	0.2690	0.4233	0.5359
AdvIR [25]	0.1790	0.2941	0.3792	0.2689	0.4235	0.5363
SRNS [11]	0.1933	0.3070	0.3912	0.2891	0.4391	0.5486
Ours	**0.1936**	**0.3101**	**0.4007**	**0.2902**	**0.4522**	**0.5721**

Table 3: Performance comparison on two datasets.

size b is 1024 for Movielens-1m and 4096 for Pinterest to fit the memory size of our device. In particular, the dimension of embedding is fixed as 32 to make sure our model gains not just because of the dimension. We employ AdaGrad [12] to optimize the model's hyperparameters on Movielens-1m and Pinterest. Training our model on a machine with one Telsa-V100 graphics card takes about 1 and 12 hours on average. For a fair comparison, we mainly use GMF as the scoring function, and carefully tuned hyper-parameters of GDNS and baselines according to validation performance. Moreover, our model can easily generalize to other CF methods since the gain-aware negative sampler is built on the expectational gain of the score gap of user-item pairs during iterations, where the scores can be calculated by the matching function of various CF methods.

4.5 TopK Recommendation

Our paper aims to distinguish the false negative samples and generate the reliable negative samples dynamically during training, and further improve the performance of our model. In order to verify the efficiency of our model, we conduct experiments on Movielens-1m and Pinterest and report $NDCG@\{1,3\}$ and $Recall@3$ in Table 3.

Table 3 gives the convincing results of Top-K recommendation on Movielens-1m and Pinterest. From the table 3, we can observe that our model consistently outperforms all baselines on two datasets with all metrics. This indicates that GDNS can sample high-quality negative instances and thus help to learn a better CF model that ranks items more accurately. More specifically, for Movielens-1m, our model achieves 0.1936 on $NDCG@1$, 0.3101 on $NDCG@3$, and 0.4007 on $Recall@3$, which are higher performance than baseline methods. Similarly, for Pinterest, our model achieves 0.2902 gain on $NDCG@1$, 0.4522 on $NDCG@3$, and 0.5721 on $Recall@3$ compared with baselines. Besides, we have following four observations. First, hard negative sampling approaches perform more competitively among all baselines. By considering the expectational gain $\mathcal{G}$ of negative instances during iterations, our GDNS can distinguish the false negative samples from the true negative samples, since the false negative samples usually have the lower $\mathcal{G}$ than the true negative samples. Second, the fixed sampling approaches perform poorly, especially NNCF that directly adopts a power distribution based on item popularity. Third, the sampling-based approaches can be more effective than the non-sampling counterpart by improving sample quality. For example,

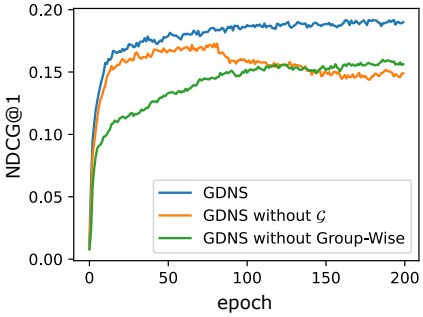

Figure 4: NDCG@1 performance comparison of our model's variants in the training process.

ENMF performs worse than RNS-AS and AdvIR on two datasets. Fourth, compared with the margin-based pairwise ranking loss, our group-wise ranking loss constructs group-aware training samples for each user in each iteration, which can reduce the calculation cost by reusing the negative samples for all user's positive interactions.

Besides the effectiveness of our model, we also qualitatively analyze the effectiveness of different components in our model, and evaluate the performance of the following variants of our model:

- **GDNS without Group-Wise.** This variant directly uses the pair-wise marginal ranking loss in equation (2) instead of our group-wise ranking loss in equation (6) to optimize the recommendation model.
- **GDNS without $\mathcal{G}$.** This variant does not take into account the gain factor $\mathcal{G}$ in negative sampler, which means that this variant makes $\mathcal{H}^t_{u,j}$ as the indicator to generate negative samples.

From figure 4, we can observe that: (1) Our model achieves better performance than these variants. It suggests that the developed components, i.e., gain-aware negative sampler and group-wise optimization, are indeed useful to boost the model's performance. (2) Without considering the gain-tuning factor $\mathcal{G}$, the performance of the variant is sharply increasing in the early stage but then drops quickly in the later stage. It is mainly because that variant without $\mathcal{G}$ tends to favor the negative samples with high scoring expectations, which may be lead to overfitting and diminish the robustness of our model. (3) Compared with the pair-wise ranking loss, our

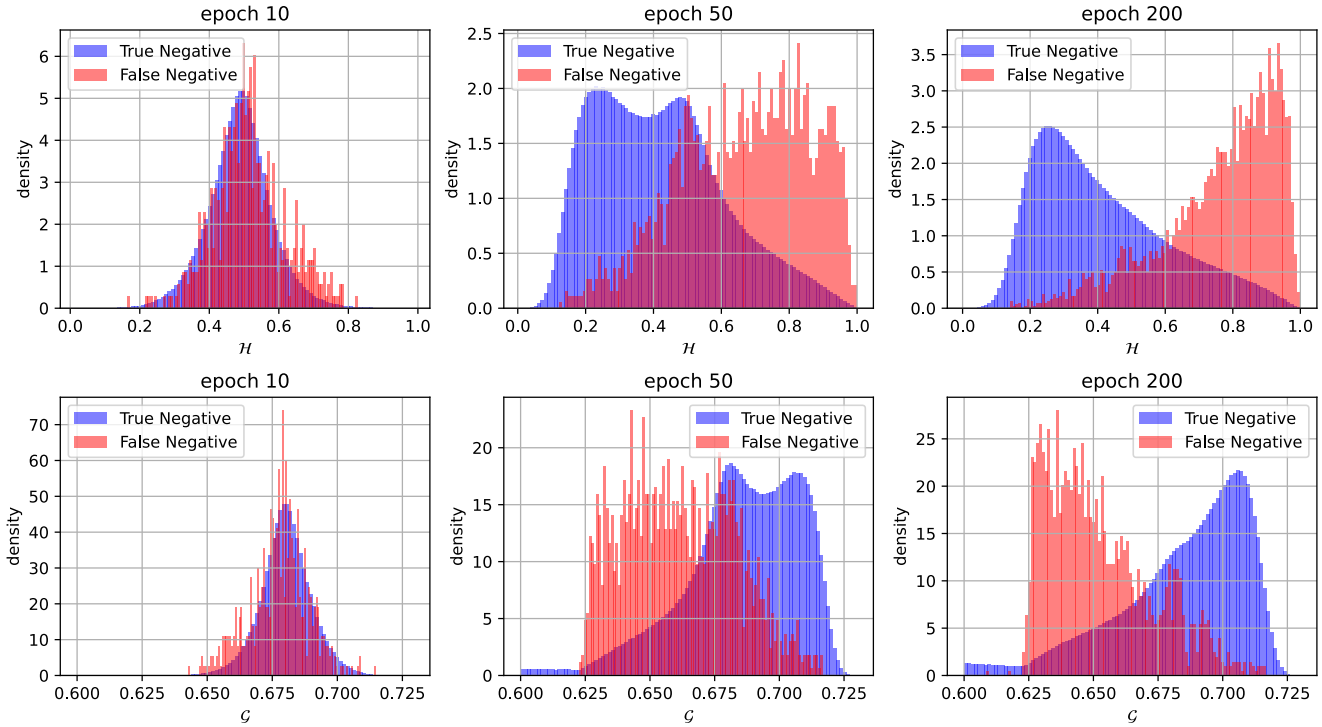

Figure 5: $\mathcal{H}$ and $\mathcal{G}$ comparison between false negative and true negative samples.

group-wise ranking loss has remarkable performance during the training process. It indicates that our group-wise ranking loss can avoid convergence to a poor local optimum, and does help to train the model more efficiently. In general, we attribute the superiority of our model to its two advantages: (1) our model can efficiently leverage the gain-tuning indicator to guide the generation of the reliable negative samples dynamically. (2) our model successfully constructs positive and negative groups for each user and uses group-wise optimization to boost training efficiency.

4.6 False Negative Analysis

In this section, we mainly to investigate whether the gain-aware negative sampler can indeed distinguishe false negative samples and further identify true negative instances. As described in previous work [11, 16, 31], the false negative sample is more likely to reflect the user's potential preferences. For a rigorous experiment, we randomly select a positive item of each user as the negative to simulate the false negative samples in the training stage. For Movielens-1m/Pinterest datasets, we randomly select 6,028 and 55,187 positive items as negative items for training. In the experiment, the constructed Movielens-1m/Pinterest datasets have 557,158/1,335,248 training samples, and the unchanged valid/test sets. To investigate whether GDNS can identify false negative samples, we track GDNS on $\mathcal{H}$ and $\mathcal{G}$ during training to analyze how these false negative samples changes. The results are shown in figure 5, which can be observed that: (1) With the density distribution of $\mathcal{H}$, it is confirmed that $\mathcal{H}$ of false negative samples is relatively larger than that of common true negative samples. Previous hard negative sampling methods are prone to select samples with larger $\mathcal{H}$ as negative samples, which would suffer from introducing the false negative samples and lead to overfitting even though high gradients can be obtained. (2) As the density distribution of $\mathcal{G}$ shown, the false negative samples exhibit a lower $\mathcal{G}$ than the common true negative samples because the false negative samples are usually difficult to be optimized and $\mathcal{H}$ changes slowly between two iterations. These phenomenon empirically prove that the false negative instances always have high expectation gap over many iterations of training, which can provide a new angle on tackling false negative problem remained in existing approaches. Thus, our gain-aware negative sampler on $\mathcal{G}$ can effectivelly mitigate the risk of introducing false negative samples and maintain the model's robustness.

4.7 Parameter Sensitivity

In this section, we mainly explore the effect of different parameter on the performance. In this experiment, expect for the parameter being tested, all other parameters are set as optimal configuration.

Group size. This section studies the model performance under different group size N settings. The experimental results in figure 6 show that: (1) Our model's performance improves with the increasing of group size. The performance with larger N is better than that with lower N. It tells that a larger N can share more gradient information between more pairs of positive and negative, which makes the optimization more efficient. (2) When the group size N is bigger than 32, enlarging the group size is hard to obtain a significant performance improvement. This is mainly because the larger group size introduces more not relatively difficult negative samples,

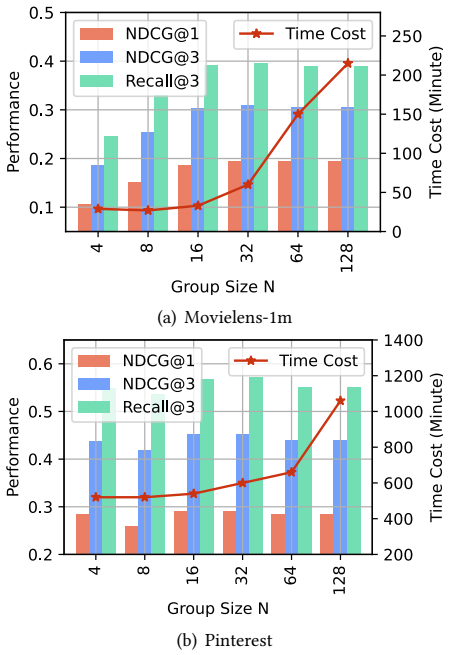

Figure 6: Performance on different group size N settings.

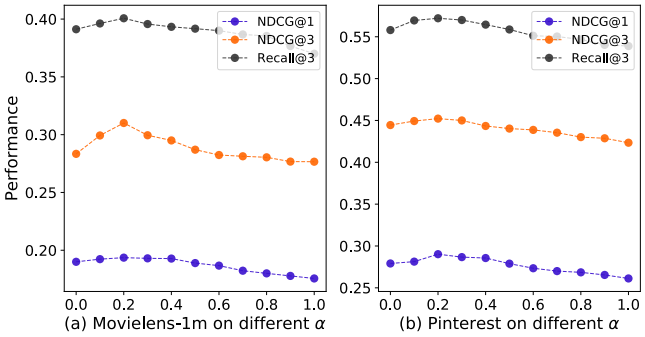

Figure 7: Performance on different smoothing parameter α.

which can not provide more gradient information to improve the performance. (3) As computation complexity analysis shows, the consumption of computing resources is also increasing with the increase of group size. Therefore, we choose 32 as the best trade-off between performance and computation.

Smoothing Parameter. In our model, the smoothing parameter α is used to make model stable on training stage. We select the smoothing parameter α from $[0, 1]$ with interval 0.1. Figure 7 gives the convinced results, which are (1) Our model achieves best performance at $\alpha = 0.2$ setting, indicating that such setting can best express the importance of expectational gain during iterations, and help make more stable training. (2) The performance first increases and then drops with the growth of α. It is because that too low α has insufficient capability of providing the information of expectational gain for training, too large α may introduce unnecessary noise and reduces generalization ability.

5 RELATED WORK

With the explosive growth of information in AI era, making personalized recommendation based on the user's interaction has been extensively investigated in research community and industry. In recent years, a variety of personalized recommendation [9, 15, 23] have been proposed. These methods usually use various unstructured information (e.g. textual reviews, visual images), and various implicit or explicit feedbacks to make personalized recommendation. As the core technique of the recommendation system, collaborative filtering based methods are important, and have been widely used in industry. This paper mainly focuses on the implicit collaborative filtering (CF) where sampling false negatives can be a severe problem. In the implicit CF problem, true negative instances are hidden inside the massive unlabeled data, along with false negative instances. Thus negative sampling for implicit CF expects an unbiased estimator that correctly identifies true negative instances during training process.

Negative sampling is the basic technique of training collaborative filtering [16] methods. Here, we classify the relative literature into three categories: (1) **Fixed Sampling** approaches [3, 27] are widely used in training collaborative filtering because of their simplicity and extensibility [27]. It randomly picks an unobserved sample as the negative, which ignores the changes over the negative sample distribution and easily fails into the vanishing gradient problem. The typical method [3] provides non-sampling way to train the neural matrix factorization model by directly using all samples in the negative sample space. (2) **GAN-based Sampling** approaches introduce Generative Adversarial Network [13] for negative sampling. Typically, it contains two additional components: a generator and a discriminator. In the scene of negative sampling, the generator is expected to produce a better negative sample distribution which the discriminator is hard to distinguish from the truth ground instance distribution [10, 25, 31]. (3) **Cache-based Sampling** approaches [2, 6, 11, 35] cache N hardest samples for each user. Each time it updates and samples from the cache pool to reuse the hard negative samples [35]. Although the cache hard negative samples can bring a big margin between positive and negative pairs to avoid the vanishing gradient problem, it also suffers from the false negative samples. SRNS [11] introduces a variance-based sampler to sample negative samples, which is state-of-the-art to avoid false negative problem.

6 CONCLUSION

This paper proposes a novel gain-tuning dynamic negative sampling model GDNS, containing a gain-ware negative sampler and a group-wise optimizer, to perform negative sampling in an efficient and robust way. In the gain-ware negative sampler, GDNS introduces a novel measure to select the reliable negative samples for training dynamically. In the group-wise optimizer, GDNS constructs positive and negative groups for each user in each iteration, which can improve training efficiency. We empirically conduct extensible experiments on two real-world datasets Movielens-1m and Pinterest and surpass State-of-the-Art models.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grant No. 62102421) , and Intelligent Social Governance Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China. I also wish to acknowledge the support provided and contribution made by Public Policy and Decision-making Research Lab of RUC.

REFERENCES

[1] Dong-Kyu Chae, Jin-Soo Kang, Sang-Wook Kim, and Jung-Tae Lee. 2018. CFGAN: A Generic Collaborative Filtering Framework based on Generative Adversarial Networks. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018,*. ACM, 137–146. https://doi.org/10.1145/3269206.3271743

[2] Haw-Shiuan Chang, Erik G. Learned-Miller, and Andrew McCallum. 2017. Active Bias: Training More Accurate Neural Networks by Emphasizing High Variance Samples. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*. 1002–1012.

[3] Chong Chen, Min Zhang, Yongfeng Zhang, Yiqun Liu, and Shaoping Ma. 2020. Efficient Neural Matrix Factorization without Sampling for Recommendation. *ACM Transactions on Information Systems* 38, 2 (Jan 2020), 14:1–14:28. https://doi.org/10.1145/3373807

[4] Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. 2017. On Sampling Strategies for Neural Network-based Collaborative Filtering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 767–776. https://doi.org/10.1145/3097983.3098202

[5] Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. 2017. On Sampling Strategies for Neural Network-based Collaborative Filtering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17)*. Association for Computing Machinery, 767–776.

[6] Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. 2017. On Sampling Strategies for Neural Network-based Collaborative Filtering. In *Proceedings of the 23rd ACM SIGKDD International Conference onmKnowledge Discovery and Data Mining*. ACM, 767–776. https://doi.org/10.1145/3097983.3098202

[7] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS@RecSys 2016*. ACM, 7–10. https://doi.org/10.1145/2988450.2988454

[8] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*. Association for Computing Machinery, 191–198. https://doi.org/10.1145/2959100.2959190

[9] Abhinandan Das, Mayur Datar, Ashutosh Garg, and Shyamsundar Rajaram. 2007. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*. ACM, 271–280.

[10] Jingtao Ding, Yuhan Quan, Xiangnan He, Yong Li, and Depeng Jin. 2019. Reinforced Negative Sampling for Recommendation with Exposure Data. (2019), 2230–2236.

[11] Jingtao Ding, Yuhan Quan, Quanming Yao, Yong Li, and Depeng Jin. 2020. Simplify and Robustify Negative Sampling for Implicit Collaborative Filtering. In *NeurIPS*.

[12] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* 12, 7 (2011).

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. *Generative Adversarial Nets*. Curran Associates, Inc., 2672–2680.

[14] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*. ijcai.org, 1725–1731. https://doi.org/10.24963/ijcai.2017/239

[15] Ruining He and Julian J. McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with one-Class Collaborative Filtering. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*. ACM, 507–517.

[16] 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 '17)*. International World Wide Web Conferences Steering Committee, 173–182. https://doi.org/10.1145/3038912.3052569

[17] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry P. Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *22nd ACM International Conference on Information and Knowledge Management, CIKM'13*. ACM, 2333–2338. https://doi.org/10.1145/2505515.2505665

[18] Bowen Jin, Chen Gao, Xiangnan He, Depeng Jin, and Yong Li. 2020. Multi-behavior Recommendation with Graph Convolutional Networks. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*. ACM, 659–668. https://doi.org/10.1145/3397271.3401072

[19] 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 (KDD '08)*. Association for Computing Machinery, 426–434. https://doi.org/10.1145/1401890.1401944

[20] Lei Li, Dingding Wang, Tao Li, Daniel Knox, and Balaji Padmanabhan. 2011. SCENE: a scalable two-stage personalized news recommendation system. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011*. ACM, 125–134. https://doi.org/10.1145/2009916.2009937

[21] Sheng Li, Jaya Kawale, and Yun Fu. 2015. Deep Collaborative Filtering via Marginalized Denoising Auto-encoder. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM*. ACM, 811–820. https://doi.org/10.1145/2806416.2806527

[22] Benjamin M. Marlin and Richard S. Zemel. 2004. The multiple multiplicative factor model for collaborative filtering. In *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004) (ACM International Conference Proceeding Series)*, Vol. 69. ACM. https://doi.org/10.1145/1015330.1015437

[23] Rong Pan, Yunhong Zhou, Bin Cao, Nathan Nan Liu, Rajan M. Lukose, Martin Scholz, and Qiang Yang. 2008. One-Class Collaborative Filtering. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008)*. IEEE Computer Society, 502–511.

[24] Dae Hoon Park and Yi Chang. 2019. Adversarial Sampling and Training for Semi-Supervised Information Retrieval. In *The World Wide Web Conference, WWW 2019*. ACM, 1443–1453. https://doi.org/10.1145/3308558.3313416

[25] Dae Hoon Park and Yi Chang. 2019. Adversarial Sampling and Training for Semi-Supervised Information Retrieval. In *The World Wide Web Conference (WWW '19)*. Association for Computing Machinery, 1443–1453.

[26] Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the 7th ACM international conference on Web search and data mining (WSDM '14)*. Association for Computing Machinery, 273–282. https://doi.org/10.1145/2556195.2556248

[27] 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 (UAI'09)*. AUAI Press, 452–461.

[28] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. AutoRec: Autoencoders Meet Collaborative Filtering. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15 Companion)*. Association for Computing Machinery, 111–112. https://doi.org/10.1145/2740908.2742726

[29] Ying Shan, T. Ryan Hoens, Jian Jiao, Haijing Wang, Dong Yu, and JC Mao. 2016. Deep Crossing: Web-Scale Modeling without Manually Crafted Combinatorial Features. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. Association for Computing Machinery, 255–262. https://doi.org/10.1145/2939672.2939704

[30] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 515–524. https://doi.org/10.1145/3077136.3080786

[31] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. Association for Computing Machinery, 515–524. https://doi.org/10.1145/3077136.3080786

[32] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. 2020. Disentangled Graph Collaborative Filtering. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*. ACM, 1001–1010. https://doi.org/10.1145/3397271.3401137

[33] 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 (KDD'18)*. Association for Computing Machinery, 974–983.

[34] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *The 36th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '13*. ACM, 785–788.

[35] Yongqi Zhang, Quanming Yao, Yingxia Shao, and Lei Chen. 2019. NSCaching: Simple and Efficient Negative Sampling for Knowledge Graph Embedding. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. 614–625. https://doi.org/10.1109/ICDE.2019.00061

[36] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep Interest Evolution Network for Click-Through Rate Prediction. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 0101 (Jul 2019), 5941–5948. https://doi.org/10.1609/aaai.v33i01.33015941