Disentangled Negative Sampling for Collaborative Filtering

Riwei Lai
Harbin Engineering University
Hong Kong Baptist University
lai@hrbeu.edu.cn

Li Chen
Hong Kong Baptist University
lichen@comp.hkbu.edu.hk

Yuhan Zhao
Harbin Engineering University
asa9ao@hrbeu.edu.cn

Rui Chen*
Harbin Engineering University
ruichen@hrbeu.edu.cn

Qilong Han*
Harbin Engineering University
hanqilong@hrbeu.edu.cn

ABSTRACT

Negative sampling is essential for implicit collaborative filtering to generate negative samples from massive unlabeled data. Unlike existing strategies that consider items as a whole when selecting negative items, we argue that normally user interactions are mainly driven by some relevant, but not all, factors of items, leading to a new direction of negative sampling. In this paper, we introduce a novel disentangled negative sampling (DENS) method. We first disentangle the relevant and irrelevant factors of positive and negative items using a hierarchical gating module. Next, we design a factor-aware sampling strategy to identify the best negative samples by contrasting the relevant factors while keeping irrelevant factors similar. To ensure the credibility of the disentanglement, we propose to adopt contrastive learning and introduce four pairwise contrastive tasks, which enable to learn better disentangled representations of the relevant and irrelevant factors and remove the dependency on ground truth. Extensive experiments on five real-world datasets demonstrate the superiority of DENS against several state-of-the-art competitors, achieving over 7% improvement over the strongest baseline in terms of *Recall*@20 and *NDCG*@20. Our code is publically available at https://github.com/Riwei-HEU/DENS.

CCS CONCEPTS

• **Information systems → Recommender systems**;

KEYWORDS

Negative sampling; contrastive learning; disentanglement learning

ACM Reference Format:
Riwei Lai, Li Chen, Yuhan Zhao, Rui Chen, and Qilong Han. 2023. Disentangled Negative Sampling for Collaborative Filtering. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23), February 27-March 3, 2023, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3539597.3570419

*Corresponding authors.

1 INTRODUCTION

Recommender systems have been a promising solution to cure information overload in various real-world applications, such as social media [4, 25], e-commerce [27, 28], and online advertising [39, 40]. Collaborative filtering (CF), as a key recommendation technique, focuses on learning user preferences from observed user-item interactions [11, 12, 31]. In many cases, it is not always possible to obtain high-quality explicit feedback, let alone a large amount of data to train modern deep recommendation models. Thus, implicit feedback (e.g., clicks or purchases) has become a default choice to train a CF model [30]. In implicit feedback, each observed interaction normally indicates a user's interest in an item and leads to a positive training sample. As for negative training samples, a widely adopted approach is to randomly select some uninteracted items for users. An implicit CF model is then optimized to give positive samples higher scores than negative ones [24].

As in many supervised learning problems, there is no doubt that negative samples play a decisive role in learning accurate CF models. More and more research results have shown that uniformly selecting uninteracted items is difficult to guarantee the quality of resulting negative samples and learn useful information about user preferences. Consequently, without prior information describing items, two lines of studies have been proposed. The first line consists of *static negative sampling methods*, which replace the uniform sampling distribution with other distributions (e.g., the popularity distribution of items) to prioritize informative items as negative samples [3, 33]. The other line is *hard negative sampling methods*, which oversample hard negative items during the training process [14, 29, 36]. Here the hard negative items refer to those with a high probability of being positive according to the model [6], which might bring more useful information for preference learning.

Although existing works on negative sampling have achieved some promising results, we argue that they still suffer from a common drawback: all of them consider an item as an indivisible whole and ignore the fact that *a user is normally driven by some specific factors of an item, but not all factors, to interact with it* [19, 26, 32]. For example, a user may watch a movie just because its actors match his/her preferences, while other factors (e.g., producer) might be less important in his/her decision making. Without taking into consideration items' fine-granular factors in negative sampling, existing methods may fail to identify the most suitable negative samples to precisely characterize user preferences for different items, leading to suboptimal recommendations.

To address this limitation, we propose a factor-aware sampling idea that considers different factors of items to select the best negative samples. Specifically, we call the factors that dominate a user's decision making for better revealing the user's preference *relevant factors* and the remaining factors *irrelevant factors*. With this distinction, intuitively a high-quality negative sample could be selected by contrasting the relevant factors while keeping irrelevant factors similar. For example, in the aforementioned example, a good negative sample should be a movie starring an actor that the user is not interested in, but made by the same producer. To fulfill this idea, we need to disentangle items' relevant and irrelevant factors and design a discriminative criterion to identify the best negative samples. However, this is non-trivial due to the following challenges:

- **How to disentangle the relevant and irrelevant factors of items?** An item's relevant and irrelevant factors vary from user to user. Thus, slicing the item embedding into fixed chunks, as adopted by existing algorithms [20, 32], is insufficient to disentangle the user-specific factors.
- **How to reliably measure the quality of negative samples?** The quality of negative samples depends on both relevant and irrelevant factors, whose respective impacts need to be carefully measured and balanced.
- **How to ensure the credibility of the disentanglement?** Since there is no ground truth to guide the learning, it is intrinsically difficult to ensure the credibility of the disentanglement.

In this paper, we present a novel *D*is*E*ntangled *N*egative *S*ampling (*DENS*) method. To tackle the first challenge, we utilize a *hierarchical gating module* to disentangle the relevant and irrelevant factors of interacted items via a positive gating layer equipped with a user-specific gate, whose outputs further guide the disentanglement of uninteracted items via a negative gating layer equipped with a factor-specific gate. With the disentangled factors of items, we further design a *factor-aware sampling strategy* to solve the second challenge. The factor-aware sampling strategy aims to effectively and efficiently identify the best negative samples based on both relevant and irrelevant factors, where the respective impacts of relevant and irrelevant factors are balanced dynamically throughout the training process. To overcome the last challenge of lacking ground truth to guide the disentanglement, we propose to use contrastive learning and introduce four *pairwise contrastive tasks* taking the user as an anchor. For example, we encourage the embedding of a user to be more similar to the relevant factors of the interacted item than to its irrelevant factors. Such pairwise contrastive tasks guarantee the credibility of the disentanglement in the absence of ground truth and enable to learn better disentangled representations of relevant and irrelevant factors.

It is worth noting that DENS can be seamlessly integrated with various implicit CF models. Without loss of generality, we consider LightGCN [11] as an example to illustrate the proposed DENS method. Empirically, DENS is able to achieve better performance than several state-of-the-art negative sampling methods such as DNS [36], SRNS [6], and MixGCF [14] on five public benchmark datasets. We summarize our main contributions as follows:

- To the best of our knowledge, we are the first to point out the importance of disentangling item factors in negative sampling, which leads to a new research direction for CF.
- We propose a simple yet effective DENS method, which disentangles relevant and irrelevant factors of items with contrastive learning and identifies the best negative samples by dynamically measuring and balancing the impact of different factors.
- We conduct extensive experiments to demonstrate that DENS can consistently and substantially outperform several state-of-the-art competitors, achieving over 7% improvement over the strongest competitor in terms of $Recall@20$ and $NDCG@20$.

2 RELATED WORK

2.1 Static Negative Sampling

A line of existing research focuses on identifying good distributions from which negative samples can be drawn. The simplest and also the most prevalent idea is to uniformly and randomly select negative samples from uninteracted items. Bayesian personalized ranking (BPR) [24] is a well-known instantiation of this idea. However, this idea is difficult to guarantee the quality of generated negative samples, and thus some studies [3, 33, 35] propose to replace the uniform distribution with other distributions. Yang *et al.* [35] derive a theory to quantify the impact of the negative sampling distribution, and then propose to sample negative items by approximating the distribution of positive items and accelerate the process by the Metropolis-Hastings algorithm. Inspired by the word-frequency-based and node-degree-based negative sampling distributions for network embedding [8, 21], NNCF [3] and NCE-PLRec [33] adopt an item-popularity-based sampling distribution. Under this distribution, popular items are more likely to be sampled as negative items, which helps to alleviate the widespread popularity bias issue in recommender systems [2].

2.2 Hard Negative Sampling

As another line of research, hard negative sampling methods emphasize the importance of oversampling hard negative samples because hard negative samples help to find tighter decision boundaries and allow more precise delineations of user preferences. More specifically, it is achieved by either assigning higher sampling probabilities to items with larger prediction scores [5, 6, 14, 23, 36, 41] or leveraging adversarial learning techniques [1, 15, 22, 29]. For instance, the dynamic negative sampling (DNS) strategy [36] is proposed to adaptively select negative samples with the highest prediction scores (e.g., the inner product of a user embedding and an item embedding). SRNS [6] oversamples items with both high prediction scores and high variances to tackle the false negative problem. IRGAN [29] trains a generative adversarial network to play a min-max game with the recommendation model. The sampler performs as a generator and samples negative items to confuse the model. Instead of directly selecting negative samples from uninteracted items, MixGCF [14] synthesizes hard negative samples by mixing positive information into negative samples, which further improves the performance. All the above negative sampling methods, however, neglect to consider items' *factor-level information* to select negative samples. Driven by this limitation, we propose a disentangled negative sampling method, which identifies the best negative samples based on relevant and irrelevant factors. We empirically show that considering disentangled factors is critical to achieve consistently better performance on different datasets.

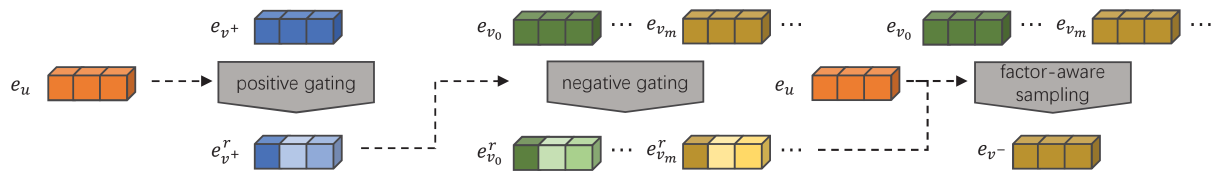


Figure 1: An illustration of DENS, where e^r denotes the relevant factors of items. Best viewed in color.

3 PROBLEM FORMULATION

In this section, we formulate the problem of negative sampling in implicit CF. Let $\mathcal{U}$ and $\mathcal{V}$ be the set of users and the set of items, respectively. We denote the set of historical interactions (i.e., implicit feedback) by $O^+ = \{(u, v^+) \mid u \in \mathcal{U}, v^+ \in \mathcal{V}\}$, where each pair (u, v^+) indicates an observed interaction between user u and item v^+. Implicit CF aims to characterize user preferences from their historical interactions. Observed interactions are generally used to form positive training samples, while uninteracted items are considered candidates to generate negative training samples. More specifically, given a positive training sample (u, v^+), a negative sampling strategy f identifies an item v^- that has not been previously interacted by u as a negative sample in order to effectively learn u's preference. A negative sampling strategy f is generally orthogonal to different CF models, and thus our DENS method can be seamlessly integrated with different existing models to further improve their performances. Without loss of generality, we consider LightGCN [11], a state-of-the-art graph neural network based CF model, as an example to illustrate the idea of DENS.

4 METHODOLOGY

In this section, we present the proposed DENS method in detail, whose workflow is illustrated in Figure 1. In the positive gating layer, we develop a user-specific gating operation to disentangle the relevant factors of interacted items. In the negative gating layer, we first randomly select M uninteracted items to form the candidate negative set. Next, we develop a factor-specific gating operation to disentangle the relevant factors of all candidate negative items by taking as input the disentangled relevant factors of the interacted item. These two components form a hierarchical structure, which is coined as the hierarchical gating module. Finally, we design a factor-aware sampling strategy to select the best negative sample from the candidate negative set based on the disentangled factors.

4.1 Hierarchical Gating Module

Motivated by the observation that a user's interaction with an item is mainly driven by the relevant factors, rather than all factors, our first task is to disentangle relevant and irrelevant factors of items. To this end, we propose a hierarchical gating module, which consists of a positive gating layer and a negative gating layer.

4.1.1 Positive Gating Layer. The positive gating layer is used to disentangle the relevant factors of *interacted items* (i.e., positive items). However, directly applying the gating operation to disentangle the relevant factors of positive items does not explicitly consider the user's specific preferences. Therefore, to capture the relevant factors that are tailored to users' preferences, we need to modify the gating operation to be user-specific. Specifically, for a pair (u, v^+), with $\mathbf{e}_u \in \mathbb{R}^d$ and $\mathbf{e}_{v^+} \in \mathbb{R}^d$ denoting the embeddings of user u and positive item v^+, respectively, the user-specific gating operation is formulated as:

$$\mathbf{e}^r_{v^+} = \mathbf{e}_{v^+} \otimes \sigma(\mathbf{W}_p \mathbf{e}_{v^+} + \mathbf{W}_u \mathbf{e}_u + \mathbf{b}_p), \tag{1}$$

where $\mathbf{e}^r_{v^+} \in \mathbb{R}^d$ is the relevant factors of positive item v^+, $\mathbf{W}_p, \mathbf{W}_u \in \mathbb{R}^{d\times d}$ and $\mathbf{b}_p \in \mathbb{R}^d$ are trainable parameters of the positive gating layer, $\sigma(\cdot)$ is the sigmoid function, and $\otimes$ is the element-wise product. By doing so, the learned relevant factors $\mathbf{e}^r_{v^+}$ can indicate user u's specific preferences on positive item v^+, and will be used as the input to the downstream negative gating layer.

4.1.2 Negative Gating Layer. The negative gating layer is employed to disentangle the *corresponding* relevant factors of *uninteracted items*. Similar to the user-specific gating operation, we propose to adopt a factor-specific gating operation that takes as input the relevant factors $\mathbf{e}^r_{v^+}$ of positive item v^+. Due to efficiency considerations, we follow the conventional methods [14, 36] to randomly select M uninteracted items of user u to form the candidate negative embedding set $\mathcal{E} = \{\mathbf{e}_{v_0}, \cdots, \mathbf{e}_{v_{M-1}}\}$, where M is usually much smaller than the total number of items $|\mathcal{V}|$, and $\mathbf{e}_{v_m}$ is the embedding of candidate negative item v_m. For each embedding $\mathbf{e}_{v_m} \in \mathcal{E}$, the factor-specific gating operation is formulated as:

$$\mathbf{e}^r_{v_m} = \mathbf{e}_{v_m} \otimes \sigma(\mathbf{W}_n \mathbf{e}_{v_m} + \mathbf{W}_r \mathbf{e}^r_{v^+} + \mathbf{b}_n), \tag{2}$$

where $\mathbf{e}^r_{v_m} \in \mathbb{R}^d$ is the corresponding relevant factors of candidate negative item v_m, and $\mathbf{W}_n, \mathbf{W}_r \in \mathbb{R}^{d\times d}$ and $\mathbf{b}_n \in \mathbb{R}^d$ are trainable parameters of the negative gating layer.

As for the irrelevant factors of items, intuitively, factors other than the relevant factors should be regarded as irrelevant factors. Since the gating operation to disentangle the relevant factors can be interpreted as assigning a weight to each factor of items, i.e., each element of item embeddings, we can calculate the irrelevant factors by the element-wise subtraction between original embeddings and disentangled relevant factors. Specifically, the irrelevant factors of positive item v^+ and candidate negative item v_m can be calculated via:

$$\mathbf{e}^i_{v^+} = \mathbf{e}_{v^+} \ominus \mathbf{e}^r_{v^+}, \quad \mathbf{e}^i_{v_m} = \mathbf{e}_{v_m} \ominus \mathbf{e}^r_{v_m}, \tag{3}$$

where $\mathbf{e}^i_{v^+}$ and $\mathbf{e}^i_{v_m} \in \mathbb{R}^d$ are the irrelevant factors of positive item v^+ and candidate negative item v_m, respectively, and $\ominus$ denotes the element-wise subtraction.

4.2 Factor-Aware Sampling Strategy

After obtaining the disentangled relevant and irrelevant factors of the positive item v^+ and all candidate negative items, next, we design a factor-aware sampling strategy to identify the best negative sample from the candidate negative set. Intuitively, a high-quality negative sample with respect to v^+ could be selected by contrasting their relevant factors while keeping irrelevant factors similar. To this end, we first need to measure the gaps between positive item v^+ and candidate negative item v_m in terms of relevant factors and irrelevant factors. Due to the considerations of efficiency and applicability, we adopt the differences between prediction scores (e.g., the inner product of a user embedding and an item embedding [11]), rather than Kullback-Leibler divergence [34] or Euclidean distance [13], via:

$$s^r = \mathbf{e}_u^\top \mathbf{e}_{v^+}^r - \mathbf{e}_u^\top \mathbf{e}_{v_m}^r, \quad s^i = \mathbf{e}_u^\top \mathbf{e}_{v^+}^i - \mathbf{e}_u^\top \mathbf{e}_{v_m}^i, \tag{4}$$

where s^r and s^i are the scores that measure the gaps between positive item v^+ and candidate negative item v_m in terms of the relevant factors and the irrelevant factors, respectively.

Next, we measure the quality of negative samples by *maximizing* the gap in terms of *relevant factors* (i.e., s^r) and *minimizing* the gap in terms of *irrelevant factors* (i.e., s^i). In order to explicitly balance the different contributions of the two types of factors, we introduce a trade-off parameter α to weigh s^r and s^i:

$$s = \alpha * s^r - (1-\alpha) * s^i. \tag{5}$$

A natural question here is why we do not use the absolute value of each gap. This is because:

- If $\mathbf{e}_u^\top \mathbf{e}_{v_m}^r$ is greater than $\mathbf{e}_u^\top \mathbf{e}_{v^+}^r$ (i.e., s^r is negative), it means user u prefers uninteracted item v_m to interacted item v^+ in terms of *relevant factors*, which indicates that the uninteracted item v_m is more likely to be a *false negative* and should **not** be selected.
- if $\mathbf{e}_u^\top \mathbf{e}_{v_m}^i$ is greater than $\mathbf{e}_u^\top \mathbf{e}_{v^+}^i$ (i.e., s^i is negative), it means user u prefers uninteracted item v_m to interacted item v^+ in terms of *irrelevant factors*, which indicates that the interaction between u and v^+ is largely due to the relevant factors of v^+. Such uninteracted item v_m contains *more useful information* and should be selected, which enables to infer the user's real preferences [38].

According to Equation (3), Equation (4), and the Distributive Law of inner product, we can rewrite Equation (5) as:

$$\begin{aligned} s &= \alpha * (\mathbf{e}_u^\top \mathbf{e}_{v^+}^r - \mathbf{e}_u^\top \mathbf{e}_{v_m}^r) - (1-\alpha) * (\mathbf{e}_u^\top \mathbf{e}_{v^+}^i - \mathbf{e}_u^\top \mathbf{e}_{v_m}^i) \\ &= \alpha * (\mathbf{e}_u^\top \mathbf{e}_{v^+}^r - \mathbf{e}_u^\top \mathbf{e}_{v_m}^r) - (1-\alpha) * [\mathbf{e}_u^\top (\mathbf{e}_{v^+} \ominus \mathbf{e}_{v^+}^r) \\ &\quad - \mathbf{e}_u^\top (\mathbf{e}_{v_m} \ominus \mathbf{e}_{v_m}^r)] \\ &= \alpha * (\mathbf{e}_u^\top \mathbf{e}_{v^+}^r - \mathbf{e}_u^\top \mathbf{e}_{v_m}^r) - (1-\alpha) * (\mathbf{e}_u^\top \mathbf{e}_{v^+} - \mathbf{e}_u^\top \mathbf{e}_{v^+}^r \\ &\quad - \mathbf{e}_u^\top \mathbf{e}_{v_m} + \mathbf{e}_u^\top \mathbf{e}_{v_m}^r) \\ &= \mathbf{e}_u^\top \mathbf{e}_{v^+}^r - (1-\alpha) * \mathbf{e}_u^\top \mathbf{e}_{v^+} - \mathbf{e}_u^\top \mathbf{e}_{v_m}^r + (1-\alpha) * \mathbf{e}_u^\top \mathbf{e}_{v_m}. \end{aligned} \tag{6}$$

Note that the first two terms of the RHS of the above equation, $\mathbf{e}_u^\top \mathbf{e}_{v^+}^r$ and $(1-\alpha) * \mathbf{e}_u^\top \mathbf{e}_{v^+}$, are the same for all candidate negative items, and have no impact on the ranking of the quality of negative samples. Therefore, we can safely remove them and derive the final factor-aware sampling strategy as follows:

$$\mathbf{e}_{v^-} = \underset{\mathbf{e}_{v_m} \in \mathcal{E}}{\arg\max}\ \mathbf{e}_u^\top ((1-\alpha) * \mathbf{e}_{v_m} \ominus \mathbf{e}_{v_m}^r). \tag{7}$$

The final key question is how to choose α. As presented in Equation (5), α affects the contributions of the two types of factors in measuring the quality of negative samples. Without the prior knowledge of the importance of each type of factors to users' decision making, we treat the selection of α as a data-driven problem and deem that it should be carefully adjusted in different scenarios. Motivated by [9], instead of setting α as a fixed parameter throughout the training process, we propose to linearly increase the value as the epoch number t increases. Specifically, we use $\alpha = \min(t/T_0, 1)$, where T_0 denotes the threshold of stopping increase.

4.3 Disentanglement with Contrastive Learning

As explained before, due to the lack of ground truth to guide the learning process, it is inherently difficult to guarantee the credibility of the disentanglement. Inspired by the superiority of contrastive learning [16, 37] in unsupervised scenarios, whose key idea is to encourage the representations of similar pairs to be close and those of dissimilar pairs to be far apart, we propose to adopt contrastive learning to supervise the disentanglement. With the user serving as an anchor, we introduce four contrastive tasks as follows:

$$\mathrm{sim}(\mathbf{e}_u, \mathbf{e}_{v^+}^r) \ > \ \mathrm{sim}(\mathbf{e}_u, \mathbf{e}_{v^+}^i), \tag{8}$$

$$\mathrm{sim}(\mathbf{e}_u, \mathbf{e}_{v^-}^i) \ > \ \mathrm{sim}(\mathbf{e}_u, \mathbf{e}_{v^-}^r), \tag{9}$$

$$\mathrm{sim}(\mathbf{e}_u, \mathbf{e}_{v^+}^r) \ > \ \mathrm{sim}(\mathbf{e}_u, \mathbf{e}_{v^-}^r), \tag{10}$$

$$\mathrm{sim}(\mathbf{e}_u, \mathbf{e}_{v^-}^i) \ > \ \mathrm{sim}(\mathbf{e}_u, \mathbf{e}_{v^+}^i), \tag{11}$$

where Equation (8) and (9) supervise the disentanglement of positive items and selected negative items, respectively, and Equation (10) and (11) supervise the disentanglement of relevant factors and irrelevant factors, respectively. The $\mathrm{sim}(\cdot,\cdot)$ is a function that measures the similarity between embeddings.

It is worth discussing the design choices as follows:

- Intuitively, users prefer the relevant factors to the irrelevant factors of *positive items* . Therefore, Equation (8) encourages the embedding of user u (i.e., $\mathbf{e}_u$) to be more similar to the relevant factors of positive item v^+ (i.e., $\mathbf{e}_{v^+}^r$) than to its irrelevant factors (i.e., $\mathbf{e}_{v^+}^i$).
- Since the negative item v^- with respect to the positive item v^+ is selected by contrasting their relevant factors while keeping irrelevant factors similar, contrary to positive items, users should prefer the irrelevant factors to the relevant factors of *negative items*. As such, Equation (9) encourages $\mathbf{e}_u$ to be more similar to the irrelevant factors of negative item v^- (i.e., $\mathbf{e}_{v^-}^i$) than to its relevant factors (i.e., $\mathbf{e}_{v^-}^r$).
- Undoubtedly, users prefer positive items to negative items, especially in *relevant factors*. Thus Equation (10) encourages $\mathbf{e}_u$ to be more similar to the relevant factors of positive item v^+ (i.e., $\mathbf{e}_{v^+}^r$) than to those of negative item v^- (i.e., $\mathbf{e}_{v^-}^r$).
- For the similar reason discussed in Equation (9), users should prefer negative items to positive items in terms of *irrelevant factors*. Therefore, Equation (11) encourages $\mathbf{e}_u$ to be more similar to the irrelevant factors of negative item v^- (i.e., $\mathbf{e}_{v^-}^i$) than to those of positive item v^+ (i.e., $\mathbf{e}_{v^+}^i$).

We refer to the Bayesian Personalized Ranking (BPR) loss function [24] to accomplish the contrastive tasks in Equation (8)-(11), which is designed to make the anchor a more similar to the positive sample p than to the negative sample q:

$$l(\mathbf{e}_a, \mathbf{e}_p, \mathbf{e}_q) = -\ln \sigma(\mathbf{e}_a^\top \mathbf{e}_p - \mathbf{e}_a^\top \mathbf{e}_q). \tag{12}$$

The inner product is used to measure the embedding similarity (i.e., $\mathrm{sim}(\cdot,\cdot)$). Therefore, the contrastive loss for the disentanglement

Algorithm 1: The training process with DENS

```
Input: Training set $O^+ = \{(u, v^+) \mid u \in \mathcal{U}, v^+ \in \mathcal{V}\}$,
       implicit CF model $F$, negative candidate size $M$.
Output: Final user embeddings $\{\mathbf{e}_u \mid u \in \mathcal{U}\}$, item
        embeddings $\{\mathbf{e}_v \mid v \in \mathcal{V}\}$, positive gating layer
        parameter $\theta_p$, negative gating layer parameter $\theta_n$.
Initialize $\{\mathbf{e}_u \mid u \in \mathcal{U}\}$, $\{\mathbf{e}_v \mid v \in \mathcal{V}\}$, $\theta_p, \theta_n$.
for $t = 1, 2, \cdots, T$ do
    Get the embeddings of users and items by $F$.
    Sample a mini-batch $O^+_{batch} \in O^+$.
    for each $(u, v^+) \in O^+_{batch}$ do
        // Negative Sampling via DENS.
        Disentangle the relevant factor $\mathbf{e}^r_{v^+}$ of $v^+$ by Eq. (1).
        Get the set of candidate negative embeddings $\mathcal{E}$.
        for each $\mathbf{e}_{v_m} \in \mathcal{E}$ do
            Disentangle the relevant factor $\mathbf{e}^r_{v_m}$ of candidate
              negative item $v_m$ by Eq. (2).
        end
        Select a negative embedding $\mathbf{e}_{v^-}$ from $\mathcal{E}$ by Eq. (7).
    end
    Update embeddings $\{\mathbf{e}_u \mid u \in \mathcal{U}\}$, $\{\mathbf{e}_v \mid v \in \mathcal{V}\}$ and
      parameters $\theta_p, \theta_n$ based on gradient $w.r.t\ \mathcal{L}$ (Eq. (15)).
end
```

can be calculated as:

$$\mathcal{L}_{\text{con}} = \sum_{(u,v^+) \in O^+} \text{mean}(l(\mathbf{e}_u, \mathbf{e}^r_{v^+}, \mathbf{e}^i_{v^+}) + l(\mathbf{e}_u, \mathbf{e}^i_{v^-}, \mathbf{e}^r_{v^-}) + l(\mathbf{e}_u, \mathbf{e}^r_{v^+}, \mathbf{e}^r_{v^-}) + l(\mathbf{e}_u, \mathbf{e}^i_{v^-}, \mathbf{e}^i_{v^+})). \tag{13}$$

4.4 Model Optimization

Finally, we adopt the proposed DENS method as the negative sampling strategy and calculate the recommendation loss (e.g., the BPR loss [24]) to optimize the parameters Θ of an implicit CF model (e.g., LightGCN [11]):

$$\mathcal{L}_{\text{rec}} = \sum_{\substack{(u,v^+) \in O^+ \\ v^- \sim f_{\text{DENS}}}} -\ln \sigma(\mathbf{e}_u^\top \mathbf{e}_{v^+} - \mathbf{e}_u^\top \mathbf{e}_{v^-}) + \lambda * \|\Theta\|_2^2, \tag{14}$$

where $v^- \sim f_{\text{DENS}}$ means that the negative item v^- is sampled by the proposed DENS method, $\|\Theta\|_2^2$ denotes the L_2 regularization to address overfitting, and λ is the coefficient controlling the strength.

We devise a multi-task learning scheme and use a hyperparameter γ to weigh the contrastive loss. The final loss function is formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \gamma * \mathcal{L}_{\text{con}}. \tag{15}$$

5 COMPLEXITY ANALYSIS

In this section, we show that DENS does **not** significantly increase the time and space complexity of training an implicit CF model. The training process with DENS is presented in Algorithm 1.

Compared with DNS [36], the main additional time cost of DENS comes from the hierarchical gating module, which disentangles the relevant factors of positive item v^+ and each item v_m in a candidate negative set of size M. Thus the additional time complexity is $O(T(1+M)N)$, where T is the number of training iterations over mini-batches and N denotes the time cost of the gating operation. In practical settings, such an overhead will not significantly slow down the training of implicit CF models. As for the space cost, DENS only requires extra space for θ_p (i.e., $\mathbf{W}_p$, $\mathbf{W}_u$, and $\mathbf{b}_p$), θ_n (i.e., $\mathbf{W}_n$, $\mathbf{W}_r$, and $\mathbf{b}_n$) and T_0, thus the additional space complexity is $O((2+4d)d+1) = O(d^2)$, where d is the dimension of embeddings. Note that the essential parameters of an implicit CF model are the embeddings of users and items, where the space complexity is $O((|\mathcal{U}| + |\mathcal{V}|)d)$. Since d is much smaller than $(|\mathcal{U}| + |\mathcal{V}|)$, the space complexity of DENS remains in the same order.

Considering the performance improvements that DENS can bring, which will be presented in the next section, we believe that these additional costs are well justified.

Table 1: The statistics of the datasets used in the experiments.

Dataset	Users	Items	Interactions	Cutting Timestamp
Amazon-Baby	4,314	5,854	42,657	Apr. 1, 2014
Amazon-Beauty	7,670	10,887	82,297	Apr. 1, 2014
Amazon-Home	23,608	22,397	230,474	Apr. 1, 2014
Amazon-Toy	4,626	10,083	52,790	Apr. 1, 2014
Last.fm	640	4,165	120,975	Apr. 1, 2010

6 EXPERIMENTS

In this section, we perform extensive experiments to evaluate our proposed DENS and answer the following research questions:

- **RQ1:** How well does DENS perform compared to previous negative sampling methods?
- **RQ2:** How well does DENS improve negative sampling via disentangling the relevant and irrelevant factors of items?
- **RQ3:** How well does DENS perform integrated with other implicit CF models such as matrix factorization (MF)?

6.1 Experimental Setup

6.1.1 Datasets and Evaluation Metrics. We consider five public benchmark datasets in the experiments. *Amazon-Baby*, *Amazon-Beauty*, *Amazon-Home* and *Amazon-Toy* are adopted from the Amazon review dataset[1] [10] with different categories. We treat purchases as implicit feedback. The *Last.fm* dataset[2] contains users' social networking, tagging, and music artist listening information collected from the Last.fm online music system. We use the artist listening information to construct implicit user-item interactions. Note that the training set is constructed with only the interactions on or before a cutting timestamp, and the remaining is used as the test set. We build the validation set by randomly sampling 10% interactions of the training set. Table 1 summarizes the statistics of the five datasets. We report the recommendation performances in terms of *Hit Ratio@K*, *Recall@K* and *NDCG@K*, with $K = \{10, 15, 20\}$, where higher values indicate better performances [11, 31].

6.1.2 Baseline Algorithms. To demonstrate the effectiveness of the proposed DENS method, we compare it with a wide range of representative negative sampling methods, including two static negative

[1]https://jmcauley.ucsd.edu/data/amazon/
[2]https://grouplens.org/datasets/hetrec-2011/

Table 2: The performance comparison with different baseline algorithms.

Dataset	Method	Top-10			Top-15			Top-20		
		Hit Ratio	Recall	NDCG	Hit Ratio	Recall	NDCG	Hit Ratio	Recall	NDCG
Baby	RNS	0.0531	0.0212	0.0115	0.0746	0.0291	0.0144	0.0919	0.0351	0.0164
	NNCF	0.0467	0.0171	0.0111	0.0620	0.0232	0.0131	0.0796	0.0296	0.0152
	DNS	0.0531	0.0205	0.0114	0.0739	0.0276	0.0139	0.0924	0.0346	0.0161
	SRNS	0.0526	0.0202	0.0113	0.0739	0.0274	0.0139	0.0929	0.0350	0.0163
	MixGCF	0.0512	0.0195	0.0111	0.0732	0.0272	0.0137	0.0937	0.0354	0.0164
	DENS	**0.0610**	**0.0231**	**0.0130**	**0.0850**	**0.0323**	**0.0163**	**0.1026**	**0.0390**	**0.0184**
Beauty	RNS	0.0568	0.0242	0.0165	0.0698	0.0319	0.0189	0.0808	0.0385	0.0206
	NNCF	0.0612	0.0267	0.0184	0.0757	0.0347	0.0209	0.0868	0.0411	0.0229
	DNS	0.0609	0.0270	0.0180	0.0757	0.0361	0.0208	0.0872	0.0418	0.0227
	SRNS	0.0614	0.0268	0.0179	0.0761	0.0355	0.0207	0.0887	0.0421	0.0228
	MixGCF	0.0616	0.0274	0.0183	0.0750	0.0350	0.0206	0.0879	0.0420	0.0229
	DENS	**0.0633**	**0.0275**	**0.0184**	**0.0787**	**0.0369**	**0.0211**	**0.0934**	**0.0437**	**0.0234**
Home	RNS	0.0272	0.0124	0.0074	0.0349	0.0162	0.0086	0.0434	0.0202	0.0098
	NNCF	0.0257	0.0119	0.0074	0.0339	0.0156	0.0086	0.0400	0.0187	0.0094
	DNS	0.0304	0.0143	0.0086	0.0395	0.0184	0.0100	0.0465	0.0217	0.0110
	SRNS	0.0307	0.0145	**0.0087**	0.0396	0.0184	0.0100	0.0464	0.0219	0.0110
	MixGCF	0.0298	0.0138	0.0085	0.0383	0.0177	0.0098	0.0462	0.0218	0.0108
	DENS	**0.0316**	**0.0145**	0.0086	**0.0420**	**0.0191**	**0.0101**	**0.0499**	**0.0230**	**0.0111**
Toy	RNS	0.0551	0.0303	0.0184	0.0681	0.0376	0.0209	0.0788	0.0444	0.0229
	NNCF	0.0548	0.0306	0.0189	0.0688	0.0388	0.0214	0.0788	0.0459	0.0234
	DNS	0.0596	0.0324	0.0203	0.0726	0.0405	0.0226	0.0861	0.0477	0.0252
	SRNS	0.0586	0.0315	0.0201	0.0723	0.0396	0.0225	0.0856	0.0473	0.0249
	MixGCF	0.0578	0.0324	0.0202	0.0723	0.0411	0.0227	0.0831	0.0471	0.0247
	DENS	**0.0601**	**0.0328**	**0.0206**	**0.0756**	**0.0425**	**0.0237**	**0.0881**	**0.0500**	**0.0259**
Last.fm	RNS	0.2765	0.0642	0.0873	0.3009	0.0817	0.0874	0.3235	0.0921	0.0870
	NNCF	0.2835	0.0681	0.0936	0.3200	0.0861	0.0937	0.3496	0.1021	0.0948
	DNS	0.3148	0.0768	0.1041	0.3443	0.0963	0.1040	0.3583	0.1084	0.1026
	SRNS	0.3009	0.0724	0.1008	0.3409	0.0938	0.1019	0.3600	0.1084	0.1026
	MixGCF	0.3096	0.0756	0.1035	0.3426	0.0959	0.1036	0.3548	0.1077	0.1020
	DENS	**0.3409**	**0.0931**	**0.1193**	**0.3809**	**0.1146**	**0.1195**	**0.4087**	**0.1335**	**0.1210**

sampling methods (RNS and NNCF) and three hard negative sampling methods (DNS, SRNS, and MixGCF).

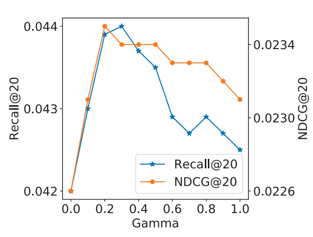
- **RNS** [24] adopts the uniform distribution to sample negative items.
- **NNCF** [3] replaces the uniform distribution with a fixed popularity-based distribution to sample more popular items as negatives.
- **DNS** [36] adaptively selects the items scored high by the recommender as negatives.
- **SRNS** [6] leverages prior statistical information to oversample high-variance items during the training process to tackle the false negative problem.
- **MixGCF** [14] develops an interpolation mixing method to inject information from positive samples to negative ones, and synthesizes harder negative samples.

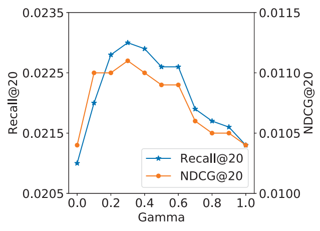
6.1.3 Implementation Details. For a fair comparison, all negative sampling methods are applied to LightGCN [11] and we strictly follow the original implementation. The embedding dimension is fixed to 64, and the embedding parameters are initialized with the Xavier method [7]. We optimize all parameters with Adam [17] and use the default learning rate of 0.001 and default mini-batch size of 2048. The L_2 regularization coefficient λ is set to 0.0001, and the default number of layers K is set to 3. The above settings are the same for all negative sampling methods. As for the hyperparameters of DENS, the candidate negative size M is searched in the range of $\{2, 4, 6, 8, 10\}$. The threshold of stopping increase α, i.e., T_0, is searched in the range of $\{50, 100, 150, \cdots, 350\}$. The multi-task weight γ of the contrastive loss $\mathcal{L}_{\text{con}}$ is searched in the range of $\{0.1, 0.2, 0.3, \cdots, 1.0\}$. The hyperparameters of all baseline algorithms are carefully tuned by grid search.

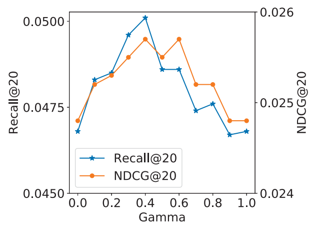
6.2 RQ1: Comparison with Baseline Algorithms

We report the performances of our proposed DENS method and all baseline algorithms in Table 2, where the best results are boldfaced and the second-best results are underlined. We can draw a few interesting observations:

- DENS consistently yields the best performance on all datasets in terms of almost all evaluation metrics. This demonstrates that DENS is capable of identifying more suitable negative samples


(a) Amazon-Beauty


(b) Amazon-Home


(c) Amazon-Toy

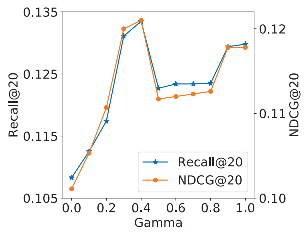

(d) Last.fm

Figure 2: Impact of different weights γ of $\mathcal{L}_{\text{con}}$.

and thus helps to learn a better CF model that ranks items more accurately. We attribute such improvements to disentangling the relevant and irrelevant factors of items and adopting a factor-aware sampling strategy for negative sampling.

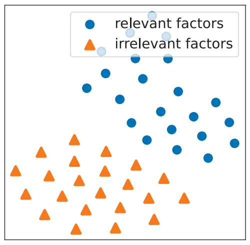
- Among all baselines, hard negative sampling methods perform competitively and static negative sampling methods perform poorly. By considering the relevant and irrelevant factors of items in negative sampling, DENS outperforms two state-of-the-art hard negative sampling methods, i.e., MixGCF and SRNS, which improve the quality of negative samples in different ways. Specifically, compared with MixGCF that *randomly* interpolates information from positive samples to synthesize more informative negative samples, DENS selects more informative negative samples by keeping the *irrelevant factors* of positive and negative samples similar, which is more effective. Compared with SRNS that leverages *prior statistical information* to oversample high-variance negatives during the training process to tackle the false negative problem, DENS can well identify the false negative samples by contrasting the *relevant factors* of positive and negative samples, which is more efficient.
- The performances of the baselines vary from dataset to dataset. None of them can consistently achieve the second-best performances on all the datasets. In particular, compared with the latest baseline method MixGCF, it is surprising to see that the simple RNS method can achieve comparable results on the Baby dataset. We believe that this observation confirms the necessity of considering items' fine-granular factors to generate more precise negative samples to reveal users' real preferences.

6.3 RQ2: Improvement of Negative Sampling via Disentangling Factors

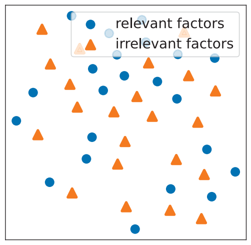
In this subsection, we discuss the impact of disentangling the relevant and irrelevant factors in negative sampling on performances.

6.3.1 Impact of Contrastive Loss $\mathcal{L}_{\text{con}}$. We first study the impact of the contrastive loss $\mathcal{L}_{\text{con}}$ in supervising the disentanglement. We carefully design three experiments from different perspectives:

- In the first experiment, we study the impact of the multi-task weight γ of $\mathcal{L}_{\text{con}}$. Figure 2 presents the varying values of *Recall*@20 and *NDCG*@20 as γ varies in the range of $\{0, 0.1, 0.2, \cdots, 1.0\}$. Among all four datasets, the worst *Recall*@20 and *NDCG*@20 are achieved when $\gamma = 0$, where the disentanglement is trained without any supervision. Such results well verify the effectiveness of our proposed contrastive loss $\mathcal{L}_{\text{con}}$. Furthermore, we can observe that $\gamma = 0.3$ or $\gamma = 0.4$ is an optimal value and the model obtains the best performance. Too small γ is not enough to guarantee the credibility of disentanglement, and too large γ may contradict the main recommendation task, which will lead to lower performances.



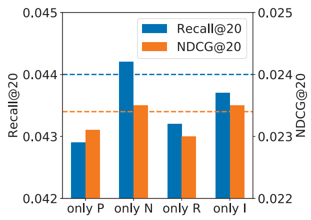
(a) With $\mathcal{L}_{\text{con}}$



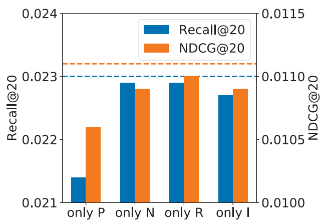
(b) Without $\mathcal{L}_{\text{con}}$

Figure 3: Visualization of the learned relevant and irrelevant factors in DENS on Last.fm dataset with and without $\mathcal{L}_{\text{con}}$.

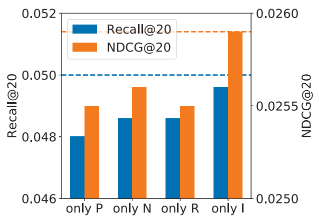
- In the second experiment, we investigate the effect of $\mathcal{L}_{\text{con}}$ in disentangling the relevant and irrelevant factors of items. Specifically, we randomly pick a user and map the disentangled relevant and irrelevant factors of all positive and negative items into a two-dimensional space. Figure 3 illustrates the learned relevant and irrelevant factors with and without the contrastive loss. We observe that with the contrastive loss, the learned relevant and irrelevant factors are clearly separated. However, when we remove the contrastive loss, the distribution of factors becomes messy, and the relevant and irrelevant factors tend to overlap with each other.
- In the third experiment, we conduct an ablation study to verify the benefits of different contrastive tasks in $\mathcal{L}_{\text{con}}$. In particular, we individually use one of the contrastive tasks in Equation (8)-(11) in turn to construct the contrastive loss and report the results in terms of *Recall*@20 and *NDCG*@20 in Figure 4. The dotted line represents the results of considering the four contrastive tasks together. We can observe that "only N" (i.e., only using the task in Eq. (9)) and "only I" (i.e., only using the task in Eq. (11)) outperform other variants in most cases, which demonstrates the effectiveness of the two tasks in supervising the disentanglement. In addition, combining the four tasks together yields further improvements in three of the reported datasets. Moreover, combined with the results in Figure 2, all the variants achieve better performances than training without any supervision, which well justifies our design of the four contrastive tasks.



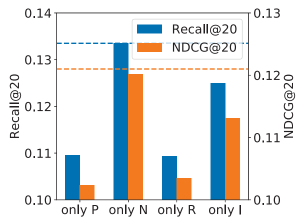
(a) Amazon-Beauty



(b) Amazon-Home



(c) Amazon-Toy

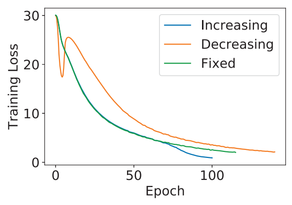


(d) Last.fm

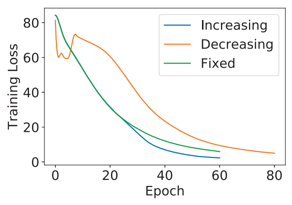
Figure 4: Impact of different contrastive tasks in $\mathcal{L}_{\text{con}}$.

Table 3: Impact of different strategies for α.

Dataset	Beauty		Home	
Strategy	Top-20		Top-20	
	Recall	NDCG	Recall	NDCG
Increasing	**0.0437**	**0.0234**	**0.0230**	**0.0111**
Decreasing	0.0398	0.0219	0.0220	0.0107
Fixed	0.0435	0.0233	0.0228	**0.0111**



(a) Amazon-Beauty



(b) Amazon-Home

Figure 5: Training loss with different strategies for α.

6.3.2 Impact of Strategies for α. We next study the impact of different strategies for α. In particular, we compare the linearly increasing strategy (i.e., $\alpha = \min(t/T_0, 1)$) with other two strategies, the linearly decreasing strategy (i.e., $\alpha = \max(0, 1 - t/T_0)$) and the fixed strategy (e.g., $\alpha = 0.4$). Table 3 presents the results of the three strategies in terms of *Recall*@20 and *NDCG*@20. Due to the space limitation, we only report the results on *Beauty* and *Home*. We observe the pattern "Increasing > Fixed > Decreasing" across the two datasets, which confirms the superiority of the linearly increasing strategy over the other two strategies. We attribute these results to the following two reasons: (1) In the early stage of the training process, the linearly increasing strategy considers more *irrelevant factors* in negative sampling, which accelerates the learning of user preferences. (2) In the later stage of the training process, based on the well-trained representations of users and items, the linearly increasing strategy characterizes user preferences more precisely by considering more *relevant factors* in negative sampling.

To verify our conjectures, we plot the curves of training loss of the three strategies in Figure 5. We can observe that, compared with the other two strategies, the linearly increasing strategy obtains lower training loss in the later stage of training, which well justifies the above reason (2). Moreover, the linearly increasing strategy achieves better testing accuracy with fewer training epochs, which validates the above reason (1).

Table 4: The performance comparison integrated with MF.

Dataset	Beauty		Home	
Method	Top-20		Top-20	
	Recall	NDCG	Recall	NDCG
RNS	0.0229	0.0122	0.0152	0.0078
NNCF	0.0243	0.0134	0.0124	0.0063
DNS	<u>0.0262</u>	<u>0.0143</u>	<u>0.0157</u>	0.0082
SRNS	0.0260	0.0142	**0.0159**	<u>0.0081</u>
DENS	**0.0266**	**0.0146**	<u>0.0157</u>	**0.0083**

6.4 RQ3: Collaboration with More CF Models

We finally study whether DENS can achieve better performances when integrated with other CF models, for example, the well-known matrix factorization (MF) [18]. Table 4 presents the performance comparison when integrated with MF. We omit MixGCF because it is a graph-based negative sampling method and not compatible with MF. As shown in Table 4, when integrated with MF, DENS can also achieve better performances in most cases. However, compared with integrating with the more advanced CF model LightGCN, the relative improvements brought by DENS are limited. This observation suggests that the quality of CF models affects the effectiveness of DENS. High-quality CF models enable learning more representative embeddings, which lays a more solid foundation for the idea of disentangling item factors in negative sampling.

7 CONCLUSION

In this paper, we studied negative sampling in CF from a brand new perspective that considers items' factor-level information. We devised a novel disentangled negative sampling (DENS) method, which effectively extracts items' relevant and irrelevant factors and selects the best negatives by carefully measuring and balancing their influence. We also introduced contrastive learning to ensure the credibility of disentanglement. Comprehensive experiments show that DENS provides a promising research direction for negative sampling to further boost existing CF models' performances.

ACKNOWLEDGMENTS

This work was supported by the National Key R&D Program of China (No. 2020YFB1710200), the Hong Kong Research Grants Council (RGC/HKBU12201620), the National Natural Science Foundation of China (No.61872105 and No.62072136), and the HKBU IG-FNRA Project (RC-FNRA-IG/21-22/SCI/01).

REFERENCES

[1] Liwei Cai and William Yang Wang. 2018. KBGAN: Adversarial Learning for Knowledge Graph Embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*. Association for Computational Linguistics, 1470–1480.

[2] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and Debias in Recommender System: A Survey and Future Directions. *CoRR* abs/2010.03240 (2020). arXiv:2010.03240 https://arxiv.org/abs/2010.03240

[3] 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 Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 767–776.

[4] Tianwen Chen and Raymond Chi-Wing Wong. 2021. An Efficient and Effective Framework for Session-based Social Recommendation. In *Proceedings of the 14th International Conference on Web Search And Data Mining (WSDM)*. ACM, 400–408.

[5] Jingtao Ding, Yuhan Quan, Xiangnan He, Yong Li, and Depeng Jin. 2019. Reinforced Negative Sampling for Recommendation with Exposure Data. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*. ijcai.org, 2230–2236.

[6] Jingtao Ding, Yuhan Quan, Quanming Yao, Yong Li, and Depeng Jin. 2020. Simplify and Robustify Negative Sampling for Implicit Collaborative Filtering. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS)*.

[7] Xavier Glorot and Yoshua Bengio. 2010. Understanding the Difficulty of Training Deep Feedforward Neural Networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*. JMLR.org, 249–256.

[8] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 855–864.

[9] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS)*. 8536–8546.

[10] Ruining He and Julian 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)*. ACM, 507–517.

[11] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 639–648.

[12] 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)*. ACM, 173–182.

[13] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge J. Belongie, and Deborah Estrin. 2017. Collaborative Metric Learning. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*. ACM, 193–201.

[14] Tinglin Huang, Yuxiao Dong, Ming Ding, Zhen Yang, Wenzheng Feng, Xinyu Wang, and Jie Tang. 2021. MixGCF: An Improved Training Method for Graph Neural Network-based Recommender Systems. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 665–674.

[15] Binbin Jin, Defu Lian, Zheng Liu, Qi Liu, Jianhui Ma, Xing Xie, and Enhong Chen. 2020. Sampling-Decomposable Generative Adversarial Recommender. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS)*.

[16] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised Contrastive Learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS)*.

[17] Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

[18] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factoriza-Tion Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.

[19] Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Huan Zhao, Pipei Huang, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. 2019. Multi-Interest Network with Dynamic Routing for Recommendation at Tmall. In *Proceedings of the 28th International Conference on Information and Knowledge Management (CIKM)*. ACM, 2615–2623.

[20] Jianxin Ma, Chang Zhou, Peng Cui, Hongxia Yang, and Wenwu Zhu. 2019. Learning Disentangled Representations for Recommendation. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NIPS)*. 5712–5723.

[21] Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS)*. 3111–3119.

[22] Dae Hoon Park and Yi Chang. 2019. Adversarial Sampling and Training for Semi-Supervised Information Retrieval. In *Proceedings of the 28th International Conference on World Wide Web (WWW)*. ACM, 1443–1453.

[23] Steffen Rendle and Christoph Freudenthaler. 2014. Improving Pairwise Learning for Item Recommendation from Implicit Feedback. In *Proceedings of the 7th International Conference on Web Search And Data Mining (WSDM)*. ACM, 273–282.

[24] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI)*. AUAI Press, 452–461.

[25] Weiping Song, Zhiping Xiao, Yifan Wang, Laurent Charlin, Ming Zhang, and Jian Tang. 2019. Session-Based Social Recommendation via Dynamic Graph Attention Networks. In *Proceedings of the 12th International Conference on Web Search And Data Mining (WSDM)*. ACM, 555–563.

[26] Qiaoyu Tan, Jianwei Zhang, Jiangchao Yao, Ninghao Liu, Jingren Zhou, Hongxia Yang, and Xia Hu. 2021. Sparse-Interest Network for Sequential Recommendation. In *Proceedings of the 14th International Conference on Web Search And Data Mining (WSDM)*. ACM, 598–606.

[27] Jianling Wang, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. Next-item Recommendation with Sequential Hypergraphs. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 1101–1110.

[28] Jianling Wang, Raphael Louca, Diane Hu, Caitlin Cellier, James Caverlee, and Liangjie Hong. 2020. Time to Shop for Valentine's Day: Shopping Occasions and Sequential Recommendation in E-commerce. In *Proceedings of the 13th International Conference on Web Search And Data Mining (WSDM)*. ACM, 645–653.

[29] 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 40rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 515–524.

[30] Wenjie Wang, Fuli Feng, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2021. Denoising Implicit Feedback for Recommendation. In *Proceedings of the 14th International Conference on Web Search And Data Mining (WSDM)*. ACM, 373–381.

[31] 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)*. ACM, 165–174.

[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)*. ACM, 1001–1010.

[33] Ga Wu, Maksims Volkovs, Chee Loong Soon, Scott Sanner, and Himanshu Rai. 2019. Noise Contrastive Estimation for One-Class Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 135–144.

[34] Xin Xia, Hongzhi Yin, Junliang Yu, Yingxia Shao, and Lizhen Cui. 2021. Self-Supervised Graph Co-Training for Session-based Recommendation. In *Proceedings of the 30th International Conference on Information and Knowledge Management (CIKM)*. ACM, 2180–2190.

[35] Zhen Yang, Ming Ding, Chang Zhou, Hongxia Yang, Jingren Zhou, and Jie Tang. 2020. Understanding Negative Sampling in Graph Representation Learning. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 1666–1676.

[36] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 785–788.

[37] Yu Zheng, Chen Gao, Jianxin Chang, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2022. Disentangling Long and Short-Term Interests for Recommendation. In *Proceedings of the Web Conference 2022*. ACM, 2256–2267.

[38] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin. 2021. Disentangling User Interest and Conformity for Recommendation with Causal Embedding. In *Proceedings of the Web Conference 2021*. ACM / IW3C2, 2980–2991.

[39] 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. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI)*. AAAI Press, 5941–5948.

[40] Guorui Zhou, Xiaoqiang Zhu, Chengru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 1059–1068.

[41] Qiannan Zhu, Haobo Zhang, Qing He, and Zhicheng Dou. 2022. A Gain-Tuning Dynamic Negative Sampler for Recommendation. In *Proceedings of the Web Conference 2022*. ACM, 277–285.