Bayesian Negative Sampling for Recommendation

Bin Liu

School of Electronic Information and Communications
Huazhong University of Science and Technology (HUST)
Wuhan, China
liubin0606@hust.edu.cn

Bang Wang*

School of Electronic Information and Communications
Huazhong University of Science and Technology (HUST)
Wuhan, China
wangbang@hust.edu.cn

Abstract—How to sample high quality negative instances from unlabeled data, i.e., negative sampling, is important for training implicit collaborative filtering and contrastive learning models. Although previous studies have proposed some approaches to sample informative instances, few has been done to discriminating false negative from true negative for unbiased negative sampling. On the basis of our order relation analysis of negatives' scores, we first derive the class conditional density of true negatives and that of false negatives. We next design a Bayesian classifier for negative classification, from which we define a model-agnostic posterior probability estimate of an instance being true negative as a quantitative negative signal measure. We also propose a Bayesian optimal sampling rule to sample high-quality negatives. The proposed Bayesian Negative Sampling (BNS) algorithm has a linear time complexity. Experimental studies validate the superiority of BNS over the peers in terms of better sampling quality and better recommendation performance. [1]

Index Terms—negative sampling, contrastive learning, implicit collaborative filtering

I. INTRODUCTION

Negative sampling originates from the Positive-Unlabel (PU) problem [1], [2]: A training dataset, called *PU-dataset*, contains both positively labelled and unlabeled instances, yet an unlabeled instance could belong to either the positive or negative class. Negative sampling is to determine a policy for guiding how to sample an unlabeled instance from a PU-dataset, so as to effectively train downstream task models. Negative sampling can find many applications in diverse tasks, such as natural language processing (NLP) [3], [4], computer vision (CV) [5], [6], as well as recommendation systems (RS) [7]–[9].

We focus on negative sampling for recommendation. Many recommendation tasks can be formulated as how to rank unlabeled items for users, yet an unlabeled item is specific to one user, called his *negative instance*. Most recommendation algorithms adopt a *pairwise learning* framework to train a recommendation model with learnable users' and items' representations for the ranking computation [10]–[13]. A pairwise comparison is to first form a training triple (u, i, j) consisting of his positive instance i and negative instance j for a user u, and the pairwise loss over all users is optimized for training a recommendation model. The dilemma of such pairwise comparisons lies in that a training triple consists of a negative instance of a user, yet such a negative instance could be potentially interested by the same user and should

be recommended by the trained model. This motivates the problem of negative sampling for recommendation. that is, how to effectively sample negative instances for training a recommendation model. Many studies have shown that negative sampling is important to improving recommendation performance [7]–[9], [14]–[17].

Recently, some negative sampling algorithms have been proposed for recommendation. We group them into the following categories. *Static negative sampling* [3], [10]–[12], [18]–[20]. Algorithms of this kind are to sample a negative instance according to a fixed sampling distribution, such as uniform sampling. *Hard negative sampling* [7], [8], [21], [22]. Algorithms of this kind favor those negative instances with representations more similar to that of positive instances in the embedding space, for example, selecting higher scored or higher ranked instances [7], [8], [23], however they are more likely to suffer from the negative problem, as reported in some recent studies [5], [6], [9]. A recent algorithm SRNS [9] of this kind advocates sampling a negative with a large variance of its predicted scores in the training precess.

In this paper, we contribute to the negative sampling studies in the field of implicit CF in three aspects: (i) On the basis of order relation analysis of negatives' scores, we derive the class conditional density of true negatives and that of false negatives, and provide an affirmative answer from a Bayesian viewpoint to *distinguish true negatives from false negatives* (**RQ1**). (ii) According to the asymptotic property of the empirical distribution function, we defined a model-agnostic estimator of an instance being true negative as a *quantitative negative signal measure* (**RQ2**). In particular, it is an unbiased posterior probability estimate of an instance being true negative that combines prior information (model-independent) with sample information (model-dependent). (iii) We also propose a Bayesian sampling rule to *sample high-quality negative instances* (**RQ3**). It is the theoretically optimal sampling rule that minimizes the empirical sampling risk. Experiment studies validate our analysis and solution in terms of sampling quality and recommendation performance.

II. NEGATIVE SAMPLING ANALYSIS

In this section, we use a general formulation of the personalized recommendation task to analyze the properties of negative sampling for training a recommendation model. We consider the following personalized recommendation problem,

---

[1] Source code and data are avaliable at https://github.com/liubin06/BNS

which has been intensively studied in the field [7], [10], [12], [24]. Let $\mathcal{M}$ denote a recommendation model. Its input is an user-item interaction matrix $\mathbf{X} = [x_{ui}] \in \mathbb{R}^{M \times N}$, consisting of M users and N items. An element $x_{ui} = 1$ indicates a user u has interacted with an item i; Otherwise, $x_{ui} = 0$. The output is for each user his recommendation list, consisting of his un-interacted items ranked according to their predicted scores.

To train the recommendation model $\mathcal{M}$, the widely used optimization objective is the following *pairwise loss*:

$$\mathcal{L}_{loss} \equiv \max_{\Theta} \sum_{(u,i,j)} \ln \sigma(\hat{x}_{ui} - \hat{x}_{uj}), \tag{1}$$

where for a user u, $\hat{x}_{ui}$ and $\hat{x}_{uj}$ is the predicted score for his already interacted item i and un-interacted item j, respectively. Θ contains model trainable parameters, and λ is a hyper-parameter in the learning process. In order to compute $\hat{x}_{ui}$, some *representation learning* techniques such as MF [25] and LightGCN [12] can be used to learn a user representation $\mathbf{w}_u$ and an item representation $\mathbf{h}_i$, such that $\hat{x}_{ui} = \text{sim}(\mathbf{w}_u, \mathbf{h}_i)$ with a similarity function sim, e.g., a cosine or dot-product function.

When training $\mathcal{M}$, negative sampling is used to construct training triples. For a user u, let $\mathcal{I}_u^+$ and $\mathcal{I}_u^-$ denote the set of his already interacted items, called *positive instances* and the set of his un-interacted items, called *negative instances*. A training triple (u, i, j) is constructed as follows: For a user u and one of his positive instance $i \in \mathcal{I}_u^+$, sample one of his negative instances $j \in \mathcal{I}_u^-$, viz., *negative sampling*. On the one hand, although the instance j is sampled from $\mathcal{I}_u^-$, it could be the case that the user u actually likes it, but the un-interaction is simply due to that he had not seen it before, that is, the item j is a *false negative* with respect to the user u. On the other hand, the item j is called a *true negative*, if the user u truly dislikes it.

The *stochastic gradient descent* (SGD) is often used to iteratively optimize the loss function for each training triple (u, i, j). For a sampled instance $j \in \mathcal{I}_u^-$, if it is a true negative to u, the loss gradient with respect to its estimated score $\hat{x}_{uj}$ is computed by

$$\frac{\partial \mathcal{L}_{loss}}{\partial \hat{x}_{uj}} = [1 - \sigma(\hat{x}_{ui} - \hat{x}_{uj})](-1), \tag{2}$$

where $\sigma(\cdot)$ is the sigmoid function. However, if the sampled instance j is actually a false negative, we do not expect that such an incorrect sampling impacts much on the model training, especially causing an opposite gradient direction. As we have no prior knowledge about the sampled instance j, we argue to replace the last term (-1) in Eq. (2) by a $\text{sgn}(\cdot)$ function, that is,

$$\frac{\partial \mathcal{L}_{loss}}{\partial \hat{x}_{uj}} = [1 - \sigma(\hat{x}_{ui} - \hat{x}_{uj})] \cdot \text{sgn}(j), \tag{3}$$

where $\text{sgn}(j) = -1$, if j is a true negative to u, and $\text{sgn}(j) = 1$ is a false negative to u. The loss gradient Eq. (3) can be decomposed into two parts, i.e., *gradient magnitude*

and *gradient direction*. This motivates our negative sampling analysis on what is a high quality negative: A sampled instance j in a training triple (u, i, j) is called a *high-quality negative*, if it is both informative and unbiased.

- **Informativeness**: The informativeness of a negative j in a triple (u, i, j) is defined as the loss gradient magnitude, i.e.,

$$\text{info}(j) = [1 - \sigma(\hat{x}_{ui} - \hat{x}_{uj})]. \tag{4}$$

- **Unbiasedness**: The unbiasedness of a negative j is defined as the probability that it is a true negative to user u, i.e.,

$$\text{unbias}(j) = \text{P}(\text{sgn}(j) = -1). \tag{5}$$

The informativeness is directly defined as how much the negative j can help updating the parameters of a recommendation model, in terms of its predicted score $\hat{x}_{uj}$. Given the predicted score $\hat{x}_{ui}$ of a positive instance, an excessively small value of $\hat{x}_{uj}$ leads to $\sigma(\hat{x}_{ui} - \hat{x}_{uj}) \rightarrow 1$ and $\text{info}(j) \rightarrow 0$, i.e., the gradient vanishes, and little can be learned from j.

The unbiasedness is actually defined as the probability of j being true negative. We notice that the so-called uniform negative sampling [10] directly set $\text{sgn}(j) = -1$ for all negatives, which introduces some *sampling bias* in model training for j being actually a false negative (positively labeled). We can understand its adverse effects in two aspects. On the one hand, a recommendation model aims at maximizing the likelihood of pairwise comparisons between positive instances and negative instances by assigning higher scores to positives and lower scores to negatives. By assigning $\text{sgn}(j) = -1$, the false negative j's score will be decreased when performing stochastic gradient descent, since its gradient is directed to the negative direction. On the other hand, it treats a false negative j that a user u may be potentially interacted as a true negative instance, causing incorrect preference learning for a recommendation model. Therefore, how to identify true negative examples from unlabeled samples is the key research question that must be solved for negative sampling.

III. THE PROPOSED ALGORITHM

A. Order analysis on sampled instances

From un-interacted instances, negative sampling aims to select true negatives for model training; While the recommendation model aims to rank false negatives. Let us define *negative classification* as the task of classifying an un-interacted instance as either a true negative or a false negative. Like the two sides of a coin, both negative sampling and recommendation include an implicit task of negative classification, that is, *how to effectively classify a sampled negative instance?*

According to the optimization objective, the implicit CF model is optimized for ranking the positive instances higher than negative instances. This suggests that the following *order relation* of predicted scores might hold in general

$$\hat{x}_{tn} \leq \hat{x}_{fn}, \tag{6}$$

where $\hat{x}_{tn}$ and $\hat{x}_{fn}$ is the predicted score of a true negative and a false negative (i.e., true positives in future testing data),

respectively. Note that this is also the optimization objective for contrastive-based learning methods [26]–[28] that essentially encourage learned feature representation for positive instance to be similar with "anchor" data point , while pushing features from the randomly sampled negative instance apart from "anchor" data point [2] in the embedding space [29]–[31].

B. Distribution Analysis

Since the scores of true negatives and false negative are predicted using the same score function in a recommendation model, it is safe to assume that $\hat{x}_{fn}$ and $\hat{x}_{tn}$ are identically and independently distributed with a same probability density function $f(x)$ and a same cumulative distribution function $F(x) = P(X \leq x)$.

Consider two IID random variables X_{tn} and X_{fn} with their corresponding realizations $\hat{x}_{tn}$ and $\hat{x}_{fn}$ that are sorted in an ascending order:

$$X_{tn} \leq X_{fn}. \tag{7}$$

For the two random variables X_{tn} and X_{fn}, there exists a sufficient small interval dx, where one and only one random variable X_{tn} with its realization $\hat{x}_{tn} \in [\hat{x}_{tn}, \hat{x}_{tn} + dx]$. Since $\hat{x}_{tn} \leq \hat{x}_{fn}$, the random variables X_{fn} with its realization $\hat{x}_{fn} \in (\hat{x}_{tn} + dx, \infty)$. The probability differential of X_{tn} can be computed by

$$
\begin{aligned}
g(\hat{x}_{tn})dx =& 2! \times P(\hat{x}_{tn} \leq X_{tn} \leq \hat{x}_{tn} + dx) \\
& \times P(\hat{x}_{tn} + dx \leq X_{fn} \leq \infty) + o(dx) \\
=& 2f(\hat{x}_{tn})dx[1 - F(\hat{x}_{tn} + dx)] + o(dx),
\end{aligned}
\tag{8}
$$

where $g(\hat{x}_{tn})$ is the class conditional density of true negatives. $f(\hat{x}_{tn})dx$ evaluates the probability of $X_{tn} \in [\hat{x}_{tn}, \hat{x}_{tn} + dx]$, and $[1 - F(\hat{x}_{tn} + dx)]$ evaluates the probability of the rest random variables $X_{fn} \in (\hat{x}_{tn} + dx, \infty)$. $o(dx)$ is the high-order infinitesimal of dx. Dividing both sides of the equation by dx, the class conditional density of true negatives can be written as:

$$
\begin{aligned}
g(\hat{x}_{tn}) =& \lim_{dx \to 0} \frac{2f(\hat{x}_{tn})dx[1 - F(\hat{x}_{tn} + dx)] + o(dx)}{dx} \\
=& 2f(\hat{x}_{tn})[1 - F(\hat{x}_{tn})].
\end{aligned}
\tag{9}
$$

Likewise, the distribution of false negatives is given by:

$$h(\hat{x}_{fn}) = 2F(\hat{x}_{fn})f(\hat{x}_{fn}). \tag{10}$$

Proposition 0.1: If $f(x)$ is a probability density function, $F(x) = \int_{-\infty}^{x} f(t)dt$ is the corresponding cumulative distribution function, then
(i) $g(x) = 2f(x)[1 - F(x)]$ is a probability density function that satisfies $g(x) \geq 0$ and $\int_{-\infty}^{\infty} g(x)dx = 1$.
(ii) $h(x) = 2f(x)F(x)$ is a probability density function that satisfies $h(x) \geq 0$ and $\int_{-\infty}^{\infty} h(x)dx = 1$.

---

[2] The "anchor" here refers to the user embedding.

Proof 0.1: Since $f(x) \geq 0$, $F(x) = \int_{-\infty}^{x} f(t)dt \in [0, 1]$, so $g(x) = 2f(x)[1 - F(x)] \geq 0$, $h(x) = 2f(x)F(x) \geq 0$.

$$
\begin{aligned}
\int_{-\infty}^{\infty} g(x)dx &= \int_{-\infty}^{\infty} 2f(x)[1 - F(x)]dx \\
&= 2\int_{-\infty}^{\infty} f(x)dx - 2\int_{-\infty}^{\infty} f(x)F(x)dx \\
&= 2 - 2\int_{-\infty}^{\infty} F(x)dF(x) \\
&= 1, \\
\int_{-\infty}^{\infty} h(x)dx &= \int_{-\infty}^{\infty} 2f(x)F(x)dx \\
&= 2\int_{-\infty}^{\infty} F(x)dF(x) \\
&= 1.
\end{aligned}
$$

1) Real Distribution: To verify the order relation of Eq (6), we use the ground-truth labels for instances in the test set to obtain the false negatives that are positively labeled but unobserved during the training process. And the rest of un-interacted items are true negatives. We adopt the classic *matrix factorization* recommendation model with random negative sampling to train the model on the training set, and record their predicted scores at each training epoch. By counting the predicted scores of true negatives and false negatives, we report their distribution densities in the model training by Fig.1.

Fig. 1 provides two insightful findings: (a) The higher the predicted score of a negative instance, the higher the probability density that it is a false negative and the lower the probability density that it is a true negative; (b) As the training continues, the distinction between two distributions gradually becomes clearer: Compared to that of true negatives, the distribution of false negatives is centered on a larger score. This suggests that the score function of a recommendation model is capable of rating the false negatives higher than true negatives.

2) Theoretical Distribution: Fig 2 exhibits the distribution morphology of false negatives and true negatives derived from the ordinal relation with different types of $f(x)$: Gaussian distribution $x \sim \mathcal{N}(\mu, \sigma)$ (symmetrical), student distribution $x \sim t(n)$ (symmetrical), and Gamma distribution $x \sim Ga(\alpha, \lambda)$ (asymmetrical) . As we can see, during the training process, the actual distribution of true/false negatives in Fig 1 gradually exhibit the same structure as depicted in Fig 2. Different scoring functions and ranking models yield different density expressions $f(\cdot)$, but this separated structure is sufficient for us to classify true negatives and false negatives.

So far, we do not know the explicit expression for the $f(x)$ and $F(x)$. Yet the calculation of empirical distribution function $F_n(x) = \frac{1}{n} \sum_j I_{|X. \leq \hat{x}_l|}$ is easy to implement, which converges to common cumulative distribution function $F(x)$ almost surely by the strong law of large numbers. Glivenko theorem [32] strengthened this result by proving uniform convergence of $F_n(\cdot)$ to $F(\cdot)$. This conclusion helps us for calculating abstract function $F(\cdot)$.
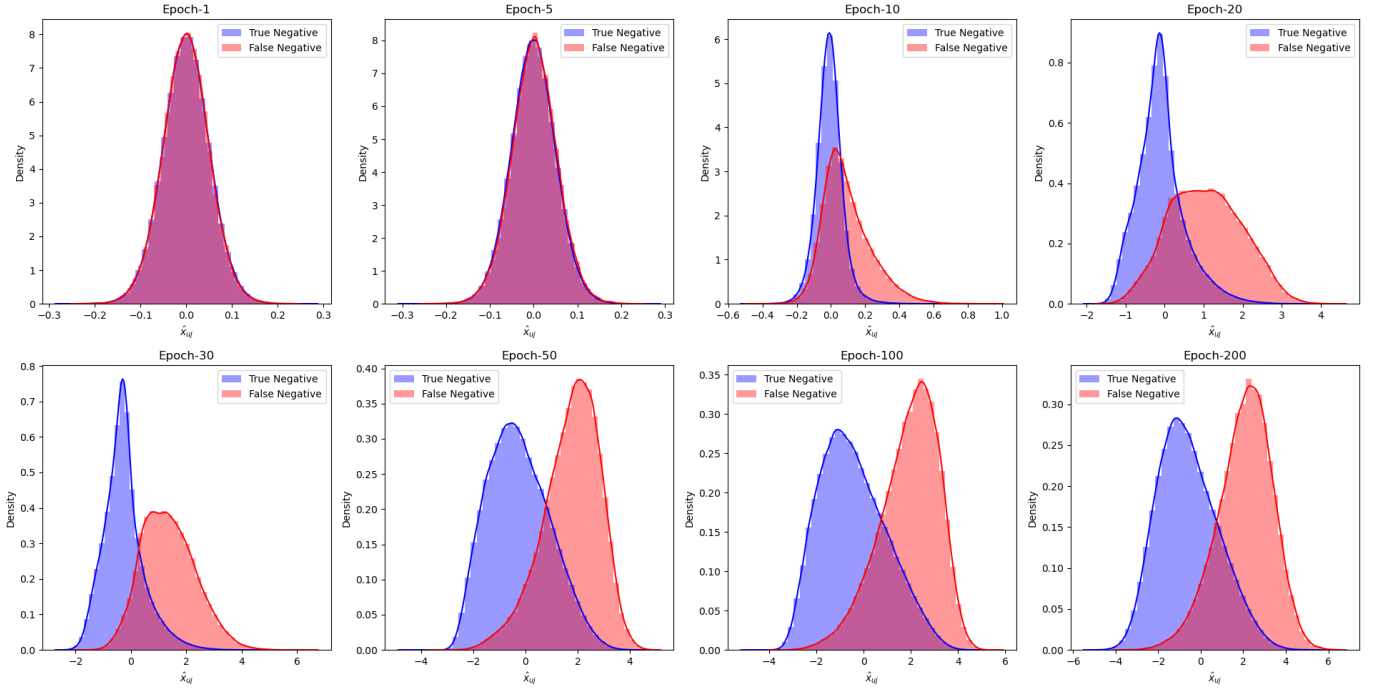
Fig. 1. Real distribution of TN and FN at different training epochs on the MovieLens-100K dataset by using the uniform random negative sampling.
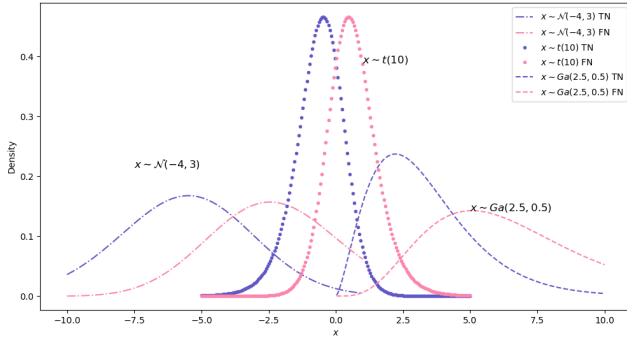

Fig. 2. Theoretical distribution of TN and FN with different types of $f(x)$.

C. Bayesian Negative Classification

For an un-interacted item l with its predicted score $\hat{x}_l$, the posterior probability of the item l being a true negative can be computed using the Bayesian formula:

$$
\begin{aligned}
P(tn|\hat{x}_l) &\propto P(\hat{x}_l|tn)P_{tn}(l) \\
&= 2f(\hat{x}_l)[1 - F(\hat{x}_l)]P_{tn}(l), \quad (11)
\end{aligned}
$$

where $P(\hat{x}_l|tn)$ is the class conditional density of true negatives given by $g(\hat{x})$, $P_{tn}(l) = 1 - P_{fn}(l)$ is the prior probability of item l being true negative instance. $f(\hat{x}_l)$ is the score distribution of un-interacted items, $F(\hat{x}_l) = \int_{-\infty}^{\hat{x}_l} f(t)dt$ is the corresponding cumulative distribution function.

Also, the posterior probability of item l being a false negative can be computed by:

$$
\begin{aligned}
P(fn|\hat{x}_l) &\propto P(\hat{x}_l|fn)P_{fn}(l) \\
&= 2F(\hat{x}_l)f(\hat{x}_l)P_{fn}(l) \quad (12)
\end{aligned}
$$

Generally, the Bayesian classifier can be obtained by maximizing the posterior probability:

$$
\underset{c \in \{fn, tn\}}{\arg\max} P(c|\hat{x}_l). \quad (13)
$$

Directly computing the score density function $f(\cdot)$ in Eq. (11) and Eq. (12) is complicated. Yet the calculation of empirical distribution function $F_n(x) = \frac{1}{n}\sum_j I_{|X_\cdot \le \hat{x}_l|}$ is easy to implement. So we define unbiasedness of an un-interacted item l in a fractional form to eliminate the density function $f(\cdot)$:

$$
\begin{aligned}
\text{unbias}(l) &= \frac{P(tn|\hat{x}_l)}{P(tn|\hat{x}_l) + P(fn|\hat{x}_l)} \quad (14) \\
&\propto \frac{f(\hat{x}_l)[1 - F(\hat{x}_l)]P_{tn}(l)}{f(\hat{x}_l)[1 - F(\hat{x}_l)]P_{tn}(l) + F(\hat{x}_l)f(\hat{x}_l)P_{fn}(l)} \\
&= \frac{[1 - F(\hat{x}_l)][1 - P_{fn}(l)]}{1 - F(\hat{x}_l) - P_{fn}(l) + 2F(\hat{x}_l)P_{fn}(l)}. \quad (15)
\end{aligned}
$$

According to Glivenko theorem [32], $F(\hat{x})$ can be approximated using $F_n(\hat{x}_l)$, i.e., the percentage of $\hat{x}_\cdot \le \hat{x}_l$

$$
F(\hat{x}_l) = \frac{\#\{\hat{x}_\cdot \le \hat{x}_l | \, l \in \mathcal{I}_u^-\}}{\#\{\mathcal{I}_u^-\}}. \quad (16)
$$

$P_{fn}(l)$ is the prior probability of l being false negative. For convenience, we assume the times of item l being interacted $pop_l \sim B(N, P_{fn}(l))$, where N is total number of interactions in training set. So

$$P_{fn}(l) = \frac{pop_l}{N}. \tag{17}$$

Lemma 0.1 (Unbiased negative signal): If $pop_l \sim B(N, P_{fn}(l))$, then $\texttt{unbias}(l)$ measure given by Eq (15) is an unbiased estimator for l being true negative.

Proof 0.2: Setting random variable $Y = 1$ if $l \in fn$, otherwise $Y = 0$,

$$Y = \begin{cases} 1, & P = \theta \\ 0, & P = 1 - \theta, \end{cases} \tag{18}$$

where θ is the probability of l being false negative. So $Y \sim B(1, \theta)$. $pop_l = Y_1 + Y_2 + \cdots + Y_N \sim B(N, \theta)$, then $P(pop_l = k) = \binom{N}{k} \theta^k (1 - \theta)^{(N-k)}$. So

$$\begin{aligned} \mathbb{E}[P_{fn}(l)] &= \mathbb{E}(\frac{pop_l}{N}) \\ &= \theta \end{aligned} \tag{19}$$

Given the observation $\hat{x}_{ul} = X$, $F(X)$ is a statistic of X. $P_{fn}(l)$ is a statistic of $\sum_i Y_i$ that is independent of X. So

$$\begin{aligned} &\mathbb{E}[\texttt{unbias}(l)] \\ &= \mathbb{E}\frac{[1 - F(X)][1 - P_{fn}(l)]}{1 - F(X) - P_{fn}(l) + 2F(X)P_{fn}(l)} \\ &= \frac{[1 - \mathbb{E}[F(X)]][1 - \mathbb{E}[P_{fn}(l)]]}{1 - \mathbb{E}[F(X)] - \mathbb{E}[P_{fn}(l)] + \mathbb{E}[2F(X)P_{fn}(l)]} \end{aligned} \tag{20}$$

$\mathbb{E}[F(X)]$ is the first order origin moment of cumulative distribution function $F(X)$

$$\begin{aligned} \mathbb{E}[F(X)] &= \int_{-\infty}^{\infty} F(x)f(x)dx \\ &= \int_{-\infty}^{\infty} F(x)dF(x) \\ &= \frac{1}{2}F^2(x)\Big|_{x=-\infty}^{x=\infty} \\ &= \frac{1}{2}. \end{aligned} \tag{21}$$

So

$$\begin{aligned} \mathbb{E}[\texttt{unbias}(l)] &= \frac{(1 - \frac{1}{2})(1 - \theta)}{1 - \frac{1}{2} - \theta + 2 \cdot \frac{1}{2} \cdot \theta} \\ &= 1 - \theta. \end{aligned} \tag{22}$$

Note $1 - \theta$ is the probability of $Y = 0$ from binomial populations $Y \sim B(1, \theta)$, so $\texttt{unbias}(l)$ is unbiased estimator of l being true negative. Fig 3 plots $\texttt{unbias}(l)$ as a function of $F(\hat{x}) \in [0, 1]$ and $P_{fn}(l) \in [0, 1]$. We observe that $\texttt{unbias}(l)$ is a decreasing function w.r.t both $F(\hat{x})$ and $P(fn)$. The monotonicity of $\texttt{unbias}(j)$ is consistent with our analysis, and the value domain of $\texttt{unbias}(j) \in [0, 1]$ conforms to the probability form.
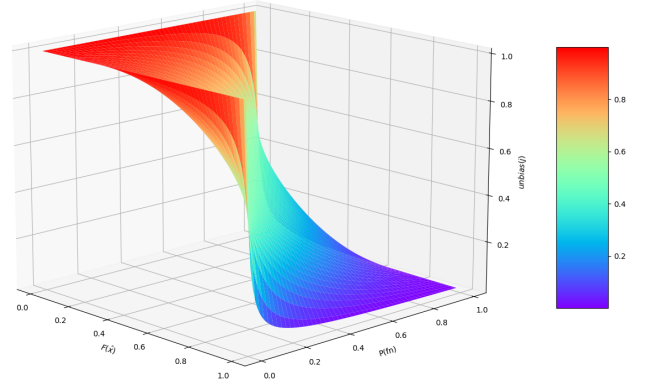

Fig. 3. Numerical plots of normalized posterior probability $\texttt{unbias}(j)$ by Eq. (15).

The definition of unbiasedness can be interpreted as the *normalized posterior probability* of l being true negative (negative signal). The density function $f(\hat{x}_l)$ is eliminated due to the fractional expression of Eq. (15). This suggests that it is model-independent metric and can be generalized to different ranking models. Eq. (15) implies that the negative signal of an un-interacted item l is formally determined by: (a) Non-threshold classification result of a ranking model denoted by $F(\hat{x}_l)$ [3]. A larger $F(\hat{x})$, i.e., a higher ranking position, indicates that to a large extent a ranking model classifies l as a false negative. Now we provide a new understanding of $F(\hat{x}_l)$. $F(\hat{x}_l)$ describes the joint probability of the observed sample $\hat{x}_l$ as a function of the parameters of the ranking model. For the specific parameter $l \in fn$, $F(\hat{x}_l)$ assigns a probabilistic prediction valued in $[0, 1]$ of l being false negative (positive). This explains why those hard negative sampling algorithms [7], [8] for over-sampling higher scored or higher ranked items are more likely to suffer from the false negative problem. (b) Prior information denoted by prior probability ($P_{tn}(l)$ or $P_{fn}(l)$).

We note that the metric of Eq. (15) covers two major paradigms of current works on distilling negative signals: (i) modeling prior information using exposure data [33], KG entities [21], connections in social networks [34], [35], etc. (ii) utilizing the sample information from ranking model such as predicted score [7], ranking position [8], scores' variance [9], etc. The former can incorporate domain knowledge, however the negative signal is independent of model status; The latter exploits sample information $\hat{x}_l$ but ignores the prior information. While being easy to sample informative instances, methods of this kind are likely to suffer false negative problems. The advantage of $\texttt{unbias}(\cdot)$ measure lies in the theoretical foundation of posterior probability, which combines priori information (model-independent) and sample information $\hat{x}_l$ (model-dependent). For the sake of simplicity and completeness of the theoretical proof, we adopted a naive approach for modeling prior probability. Some other additional information

---

[3]Ranking position and $F(\hat{x}_l)$ are with a one-to-one mapping.

and domain knowledge can also be exploited for modeling $P_{tn}(l)$.

D. Negative Sampling Algorithm

The ranking objective of Eq (1) is the analogy to AUC metric [10]. It replaces the non-differentiable Heaviside function used in AUC metric with the differentiable loss $\ln \sigma(\cdot)$, which is a common practice when optimizing for AUC [10], [36]. When performing a single negative sampling, the unlabeled instance l is directly assigned a negative gradient, resulting in a *minus effect* of the predicted score $\hat{x}_{ul}$, which will have two effects on $\mathcal{L}$: (i) decreasing of $\mathcal{L}$ by sampling a false negative, denoted as $\triangle \mathcal{L}_{fn}(l|i)$; (ii) increasing of $\mathcal{L}$ by sampling a true negative, denoted as $\triangle \mathcal{L}_{tn}(l|i)$.

Definition 0.1 (Conditional sampling risk): Given the positive instance i, we define the conditional sampling risk for sampling l as the expectation of sampling loss $\triangle \mathcal{L}(l|i)$ over the distribution of l's ground truth label c:

$$
\begin{aligned}
& R(l|i) \\
= & \mathbb{E}_{l \sim P(c|l)} \triangle \mathcal{L}(l|i) \\
= & [1 - P(tn|l)] \cdot \triangle \mathcal{L}_{fn}(l|i) + P(tn|l) \cdot \triangle \mathcal{L}_{tn}(l|i) \quad (23)
\end{aligned}
$$

where $P(tn|l)$ is the posterior probability of $l \in tn$ and can be computed by $\texttt{unbias}(l)$, and $1 - P(tn|l)$ is the posterior probability of $l \in fn$.

Definition 0.2 (Empirical sampling risk): Given a set of positive instances, we define the empirical sampling risk for sampler h as the expectation of $R(l|i)$ over the distribution of positive instances:

$$
R(h) = \mathbb{E}_{i \sim P(i)} R(l|i). \quad (24)
$$

Theorem 0.1 (Optimal sampling rule): If the conditional sampling risk $R(l|i)$ is independent of each other, for any sampler $h : \mathcal{I}_u^- \to l$,

$$
h^* = \underset{l \in \mathcal{I}_u^-}{\arg\min} R(l|i) \quad (25)
$$

uniformly superior than h that minimizes the empirical sampling risk.

Proof 0.3: Given the training set, the distribution of positive samples $P(i)$ is determined. The empirical sampling risk is

$$
R(h) = \mathbb{E}_{i \sim P(i)} R(h|i) \quad (26)
$$

where $R(h|i)$ is the conditional sampling risk given positive instance i. Then

$$
\begin{aligned}
& R(h^*) - R(h) \\
= & \mathbb{E}_{i \sim P(i)} R(h^*|i) - \mathbb{E}_{i \sim P(i)} R(h|i) \\
= & \sum_i P(i)[R(h^*|i) - R(h|i)] \\
= & \sum_i P(i)[\underset{l \in \mathcal{I}_u^-}{\arg\min} R(l|i) - R(h|i)] \\
\leq & 0. \quad (27)
\end{aligned}
$$

So the infimum of empirical sampling risk can be given in the form of optimal sampler h^*:

$$
\begin{aligned}
\inf\{R(h)\} & = R(h^*) \\
& = \mathbb{E}_{i \sim P(i)} R(h^*|i). \quad (28)
\end{aligned}
$$

The conclusion is concise and explicit: if the sampler h^* minimizes the conditional sampling risk $R(l|i)$, then the empirical sampling risk will be minimized. In turn, we can measure the quality of negative examples in terms of conditional sampling risk.

Corollary 0.1: For two negative instances $l, l' \in \mathcal{I}_u^-$, l is superior than l' if $R(l|i) \leq R(l'|i)$.

Next we will estimate sampling loss $\triangle \mathcal{L}(l|i)$ for sampling negative instance l. To simplify the analysis, we follow [10] to adopt the independence assumption, and consider the *minus effect* on a single pair's ranking objective $\tilde{\mathcal{L}} = \ln \sigma(\hat{x}_{fn} - \hat{x}_{tn})$. The Taylor expansion of ranking objective of $\tilde{\mathcal{L}}$ around point $\hat{x}_{ul}$ is

$$
\tilde{\mathcal{L}}' = \begin{cases} \tilde{\mathcal{L}} + \dfrac{\partial \mathcal{L}}{\partial \hat{x}_{ul}}(\hat{x}'_{ul} - \hat{x}_{ul}) + o(\hat{x}'_{ul} - \hat{x}_{ul}), & if \quad l = fn \\ \tilde{\mathcal{L}} - \dfrac{\partial \mathcal{L}}{\partial \hat{x}_{ul}}(\hat{x}'_{ul} - \hat{x}_{ul}) + o(\hat{x}'_{ul} - \hat{x}_{ul}), & if \quad l = tn. \end{cases} \quad (29)
$$

where $\frac{\partial \mathcal{L}}{\partial \hat{x}_{ul}} = \texttt{info}(l)$. So the unit decrease of $\hat{x}_{ul}$ (i.e., $\triangle \hat{x}_{ul} = -1$) results in $\triangle \mathcal{L}_{fn}(l|i) = \tilde{\mathcal{L}} - \tilde{\mathcal{L}}' \approx \texttt{info}(l)$, indicating the decrease of $\tilde{\mathcal{L}}$ if l is false negative (positively labeled); Otherwise $\mathcal{L}_{tn}(l|i) \approx -\texttt{info}(l)$, indicating the increase of $\tilde{\mathcal{L}}$. To take the overall ranking list into consideration, we introduce a hyperparameter λ to control the effect scale, and estimate the sampling loss as

$$
\triangle \mathcal{L}(l|i) \approx \begin{cases} \texttt{info}(l), & if \quad l = fn \\ -\lambda \texttt{info}(l), & if \quad l = tn \end{cases} \quad (30)
$$

So the conditional sampling risk for sampling instance l given positive instance i is

$$
R(l|i) = P(fn|l) \cdot \texttt{info}(l) - P(tn|l) \cdot \lambda \texttt{info}(l). \quad (31)
$$

Based on Corollary 0.1, we select high quality negative instance j by:

$$
\begin{aligned}
j & = \underset{l \in \mathcal{M}_u}{\arg\min} R(l|i) \\
& = \underset{l \in \mathcal{M}_u}{\arg\min} \ [1 - \texttt{unbias}(l) \cdot \texttt{info}(l) \\
& \quad\quad -\lambda \cdot \texttt{unbias}(l) \cdot \texttt{info}(l) \\
& = \underset{l \in \mathcal{M}_u}{\arg\min} \ \texttt{info}(l) \cdot [1 - (1 + \lambda)\texttt{unbias}(l)] \quad (32)
\end{aligned}
$$

where $\mathcal{M}_u \subseteq \mathcal{I}_u^-$ is a small candidate set containing randomly selected negative instances from $\mathcal{I}_u^-$. When $|\mathcal{M}_u| = |\mathcal{I}_u^-|$, $h = h^*$; When $\lambda \to \infty$, h reduces to $\underset{l \in \mathcal{M}_u}{\arg\max} \texttt{info}(l) \cdot \texttt{unbias}(l)$, i.e., sampling those both informative (*hard* [4]) and unbiased (*negative*) instances.

---

[4] Similar to positive instances in the embedding space.

Complexity: We summarize the proposed negative sampling algorithm as: for each negative instances in the candidate set $\mathcal{M}_u$, (i) computing interaction ratio as prior probability ($\mathcal{O}(1)$), (ii) computing $F(\hat{x}_l)$ as probabilistic prediction by ranking model (likelihood) ($\mathcal{O}(|\mathcal{I}|)$), (iii) computing the negative signal unbias(l) as posterior probability ($\mathcal{O}(1)$), and (iv) perform Bayesian negative sampling based on Eq (32) ($\mathcal{O}(1)$). So the proposed BNS has a linear time complexity. Algorithm 1 provides the pseudo-codes of our proposed negative sampling algorithm.

Algorithm 1: The proposed Bayesian negative sampling (BNS) algorithm

---

Input: Traning set $\mathcal{R} = \{(u,i)\}$, score function $s(\cdot)$, m (size of $\mathcal{M}_u$), embedding size d, weight λ.
Output: User embedding $\{\mathbf{w}_u | u \in \mathcal{U}\} \in \mathbb{R}^d$, item embedding $\{\mathbf{h}_i | i \in \mathcal{I}\} \in \mathbb{R}^d$

1 **for** $epoch = 1, 2, ..., T$ **do**
2   Sample a mini-batch $\mathcal{R}_{batch} \in \mathcal{R}$
3   **for** each $(u,i) \in \mathcal{R}_{batch}$ **do**
4   Get rating vector $\hat{\mathbf{x}}_u$.
5       $\backslash\backslash$ **Starting Negative Sampling**
6   Uniformly sample candidate set $\mathcal{M}_u \subseteq \mathcal{I}_u^-$.
7   **for** each $(u,l) \in \mathcal{M}_u$ **do**
8   Calculate info(l) by Eq (4).
9   Calculate $P_{fn}(l)$ by Eq (17). $\backslash\backslash$ *prior*
10   Calculate $F(\hat{x}_l)$ by Eq (16). $\backslash\backslash$ *likelihood*
11   Calculate unbias(l) by Eq (15).
           $\backslash\backslash$ *posterior*
12   **end**
13   Sampling negative j based on strategy Eq (32).
14   Update embeddings $\mathbf{w}_u, \mathbf{h}_i, \mathbf{h}_j$.
15   **end**
16 **end**
   Result: Final embeddings.

IV. EXPERIMENT

A. Experiment Settings

1) Dataset: We conduct experiments on three public datasets, including MovieLens-100k[5], MovieLens-1M, and Yahoo!-R3.[6] [13] They contain users' ratings on items according to a discrete five-point grading system on an interacted item. In the research of personalized ranking from implicit feedbacks [7], [8], [10], [13], [19], [20], a commonly used technique for data preprocessing is to convert all rated items to implicit feedbacks. Following their approaches, our training sets only consist of interacted items but not with their rating details. For each dataset, we randomly select 20% as test data, and the rest 80% as training data. Table I summarizes the dataset statistics.

---

[5] https://grouplens.org/datasets/movielens
[6] http://webscope.sandbox.yahoo.com/catalog.php?datatype=r.

2) Baselines: We compare the proposed algorithm with three types of negative sampling methods: (a) Fixed negative sampling distribution, including RNS [10]–[13], [19], [20], [37]: and PNS [3], [4], [18], (b) hard negative sampling with dynamic sampling distribution, including AOBPR [7] and DNS [8], and hard negative sampling based on priori statistical information to oversample high-variance negatives, including SRNS [9]. We note that all these competitors also only use the positive and unlabeled data but without additional information for negative sampling.

- RNS [10]–[13], [19], [20], [37]: (Random Negative Sampling) Uniformly sampling negatives.
- PNS [3], [4], [18]: (Popularity-biased Negative Sampling) Adopting a fixed distribution proportional to an item interaction ratio, i.e., $\propto r_j^{.75}$.
- AOBPR [7]: Over-sampling global higher ranked negatives with sampling probability proportional to $exp(-rank(j|u)/\lambda)$, where $rank(j|u)$ is the ranking position of predicted score $\hat{x}_{uj}$ in u's predicted score vector $\hat{\mathbf{x}}_u$.
- DNS [8]: Over-sampling relative higher ranked hard negatives. Its sampling probability is a liner function to the relative ranking position.
- SRNS [9]: Using priori statistical information for over-sampling high-variance negatives.

TABLE I
DATASET STATISTICS

	users	items	training set	test set
MovieLens-100k	943	1,682	80k	20k
MovieLens-1M	6,040	3,952	800k	200k
Yahoo!-R3	5,400	1,000	146k	36k

3) Experimental setup: We use the classic *matrix factorization* (MF) [25] and the recent *light graph convolution network* (LightGCN) [12] as two recommendation models. For a fair comparison, we set the identical parameters of the recommendation model for all comparing negative sampling algorithms. The codes is implemented with Numpy and Pytorch, respectively. Computations were conducted on a personal computer with Windows 10 operating system, 2.1 GHz CPU, RTX 1080Ti GPU, and 32 GB RAM.

4) Evaluation Metrics: To examine sampling quality, we measure the quality of the sampled instance from two perspectives: *sampling bias rate* and *average loss gradient magnitude*. By flipping labels of ground-truth records in the test set, we are able to obtain the false negatives (FN) that are positive labeled but unobserved during the negative sampling process. And the rest of un-interacted items are true negatives (TN). For each epoch, we record each sampled instance's label and loss gradient magnitude info(j), then define the unbiasedness and informativeness epoch-wisely by:

$$TNR = \frac{\#TN}{\#TN + \#FN}, \qquad (33)$$

$$INF = \frac{\sum_j \text{info}(j) \cdot \text{sgn}(j)}{\#TN + \#FN}, \qquad (34)$$

where $\#TN(\#FN)$ is the number of sampled true (false) negatives in each training epoch. Eq. (33) evaluates the proportion of sampled true negatives in each training epoch, i.e., true negative rate (TNR). $\mathtt{sgn}(j)$ is the indicator function: $\mathtt{sgn}(j) = 1$ if the sampled item's label is TN; Otherwise, $\mathtt{sgn}(j) = -1$ as a penalty for sampling the FN instance. The *informativeness* (INF) defined Eq. (34) can be interpreted as the average gradient magnitude with respect to selected training triple (u, i, j) in each training epoch. To evaluate recommendation performance, the widely used metrics are adopted, including P(precision), R(recall), NDCG (normalized discounted cumulative gain), to evaluate the Top-K recommendation. For their common usage, we do not provide their definitions here.

B. Experiment Results

1) Recommendation Performance: Implementation details: (a) MF [24]: embedding dimension $d = 32$, learning rate $\alpha = 0.01$, regulation constant $\lambda = 0.01$ and training epoch $T = 100$, batch size $b = 1$. (b) Light GCN [12]: embedding dimension $d = 32$, the initial learning rate $\alpha = 0.01$ and decays every 20 epochs with the decay rate=0.1, regulation constant $reg = 10^{-5}$, number of LightGCN layers $l = 1$, training epoch $T = 100$, batch size $b = 128$ for MovieLens-100K and Yahoo!-R3 datasets, $b = 1024$ for MovieLens-1M.

Table II compares the recommendation performance for the negative sampling algorithms, where the boldface and underline are used to indicate the best and the second best in each comparing group. The proposed BNS algorithm achieves the best performance in almost all cases (except two second best) of the two recommendation models, three testing datasets and three performance metrics. The results validate that our algorithm can sample high-quality negatives measured from both informativeness and unbiasedness. It is noted that the LightGCN recommendation model outperforms the MF one in general, which should thank its use of graph structure and powerful neural model for representation learning. We have run our BNS for 10 times, the standard deviations for each evaluation metric are consistently less than 0.002.

Take a close look on the results. It is interested to find that among the two static negative sampling algorithms, the RNS generally outperforms the PNS. This suggests that the popularity-based sampling distribution favoring popular items may actually introduce more biases in negative sampling. Among the three hard negative sampling algorithms, viz., AOBPR, DNS and SRNS, it is noted that the DNS often outperforms the other two. The AOBPR prioritizes those global higher ranked items, While the DNS first randomly selects a few negatives, among which favors those local relatively higher ranked items. Since the DNS balances between informativeness and unbiasedness to some extent, it can achieve the second best in many cases. The SRNS exploits the empirical observation that a negative with high-variance of its predicted scores could be a true negative. Although it is an interesting approach to identify true negative, the linear average operation of SRNS for selecting high quality instances may weaken its effectiveness.

2) Negative Sampling Quality: Implementation details: In this setion, we test two sampling criteria in terms of sampling bias and sampling quality: (i) the *posterior probability criterion* given by equation Eq. (15). (ii) *Bayesian sampling criterion* given by Eq. (32). The former aims to select true negative instances, while the latter aims to sample high-quality negative instances. The reliability of the $\mathtt{unbias}(l)$ estimation is crucial for negative sampling. We first examine whether *posterior probability criterion* is capable of selecting true negative instances, and then examine whether *Bayesian sampling criterion* is capable of selecting high quality negatives.

The *posterior probability criterion* is achieved by selecting the negative instance with the largest $\mathtt{unbias}(\cdot)$-value from the candidate set $\mathcal{M}_u$:

$$j = \underset{l \in \mathcal{M}_u}{\arg\max} \, \mathtt{unbias}(l) \tag{35}$$

$\mathcal{I}_u^-$ is a small candidate set containing randomly selected negative instances from $\mathcal{I}_u^-$. The sampling quality of different sampling methods is exhibited in Fig 4.

The *Bayesian sampling criterion* is achieved by selecting the negative instance with the smallest $R(\cdot|i)$-value from the candidate set $\mathcal{M}_u$:

$$j = \underset{l \in \mathcal{M}_u}{\arg\min} \, \mathtt{info}(l) \cdot [1 - (1+\lambda)\mathtt{unbias}(l)] \tag{36}$$

For above sampling methods, the size of $\mathcal{M}_u$ is fixed as 5.

(i) Sampling bias: *Fixed distribution sampling* (RNS and PNS) achieves relatively moderate performance. Their TNRs fluctuate around the probability of a random sample being a true negative. *Hard negative sampling* (AOBPR and DNS) has the worst performance. They adopt a greedy strategy to emphasize higher ranked negatives, also bringing higher risk of sampling false negatives per our discussion in Section III-A. The SRNS uses simple prior statistic information of variance of predicted scores, which limits the potentials of negative classification, because this prior variance may result in the sampling distribution to be overly concentrated. The proposed *Bayesian negative sampling* (BNS) achieves the best performance for its TNR closer to 1, owing to our Bayesian negative classification.

(ii) Sampling quality: The INF decreases with the increase of training epoches. This is because the trained recommendation model can rank the false negatives potentially interested by users higher than true negatives (cf. Fig. 1). Our BNS achieves the best performance after enough training epoches. The *hard negative sampling* (AOBPR and DNS) suffer more penalties due to its highest sampling bias. The SRNS adopts a linearly weighted average to combine informativeness and variance, which may not guarantee sampling unbiased and informative instances.

C. Study of BNS

We perform sensitive analysis to get deep insights on BNS.

TABLE II
COMPARISON OF RECOMMENDATION PERFORMANCE ON THE THREE DATASETS.

| Dataset	CF Model	Method	Top-5			Top-10			Top-20		
			Precision	Recall	NDCG	Precision	Recall	NDCG	Precision	Recall	NDCG
100K	MF	RNS	0.3900	0.1301	0.4143	0.3363	0.2164	0.3967	0.2724	0.3298	0.3962
		PNS	0.2647	0.0864	0.2694	0.2329	0.1475	0.2637	0.1949	0.2374	0.2709
		AOBPR	0.3970	0.1375	0.4186	0.3308	0.2165	0.3942	0.2700	0.3369	0.3980
		DNS	_0.4053_	_0.1414_	_0.4314_	0.3348	_0.2214_	_0.4042_	0.2734	_0.3413_	_0.4069_
		SRNS	0.3951	0.1342	0.4176	_0.3394_	0.2174	0.3998	_0.2747_	0.3374	0.4013
		Proposed	**0.4205**	**0.1467**	**0.4558**	**0.3463**	**0.2290**	**0.4217**	**0.2762**	**0.3466**	**0.4176**
	LightGCN	RNS	0.4261	0.1453	0.4544	0.3571	0.2319	0.4275	0.2867	0.3490	0.4248
		PNS	0.3527	0.1266	0.3816	0.3015	0.2117	0.3660	0.2461	0.3306	0.3742
		AOBPR	0.3911	0.1407	0.4200	0.3315	0.2276	0.4007	0.2680	0.3505	0.4064
		DNS	_0.4278_	_0.1475_	_0.4590_	_0.3612_	_0.2336_	_0.4331_	**0.2917**	_0.3595_	_0.4335_
		SRNS	0.4195	0.1440	0.4509	0.3564	0.2333	0.4275	0.2834	0.3520	0.4244
		Proposed	**0.4318**	**0.1518**	**0.4640**	**0.3671**	**0.2410**	**0.4368**	_0.2875_	**0.3608**	**0.4383**
1M	MF	RNS	0.3843	0.0855	0.4027	0.3353	0.1430	0.3737	0.2798	0.2244	0.3572
		PNS	0.3461	0.0753	0.3634	0.3004	0.1250	0.3356	0.2502	0.1979	0.3192
		AOBPR	0.3946	0.0954	0.4135	0.3416	0.1549	0.3837	0.2857	0.2442	0.3714
		DNS	_0.4066_	_0.0991_	_0.4272_	_0.3521_	_0.1620_	_0.3965_	_0.2945_	_0.2537_	_0.3838_
		SRNS	0.3955	0.0934	0.4225	0.3408	0.1609	0.4042	0.2779	0.2431	0.3974
		Proposed	**0.4207**	**0.1062**	**0.4324**	**0.3518**	**0.1703**	**0.4191**	**0.3045**	**0.2614**	**0.4002**
	LightGCN	RNS	0.4095	0.0953	0.4305	0.3512	0.1547	0.3985	0.2915	0.2405	0.3781
		PNS	0.3658	0.0907	0.3855	0.3152	0.1486	0.3564	0.2608	0.2314	0.3440
		AOBPR	0.4073	_0.0997_	0.4286	0.3535	**0.1626**	0.3982	0.2949	_0.2536_	_0.3849_
		DNS	_0.4130_	0.0972	_0.4342_	_0.3552_	0.1577	_0.4002_	_0.2958_	0.2468	0.3840
		SRNS	0.4026	0.0973	0.4239	0.3515	0.1526	0.3953	0.2922	0.2524	0.3815
		Proposed	**0.4228**	**0.1087**	**0.4438**	**0.3639**	_0.1612_	**0.4088**	**0.3025**	0.2527	**0.3917**
Yahoo	MF	RNS	0.1196	0.0875	0.1326	0.0935	0.1367	0.1401	0.0695	0.2015	0.1665
		PNS	0.1186	0.0876	0.1301	0.0927	0.1360	0.1378	0.0688	0.2011	0.1644
		AOBPR	0.1012	0.0741	0.1115	0.0798	0.1165	0.1184	0.0607	0.1778	0.1443
		DNS	_0.1251_	_0.0917_	_0.1390_	_0.0957_	_0.1399_	_0.1449_	_0.0697_	0.2020	_0.1697_
		SRNS	0.1141	0.0855	0.1285	0.0904	0.1358	0.1383	0.0678	_0.2025_	0.1655
		Proposed	**0.1303**	**0.0975**	**0.1470**	**0.1002**	**0.1485**	**0.1542**	**0.0711**	**0.2094**	**0.1783**
	LightGCN	RNS	0.1479	0.1101	0.1693	0.1126	0.1669	0.1760	0.0814	0.2389	0.2047
		PNS	0.1076	0.0797	0.1214	0.0809	0.1185	0.1254	0.0590	0.1708	0.1464
		AOBPR	0.1462	0.1120	0.1635	0.1048	0.1552	0.1612	0.0763	0.2229	0.1886
		DNS	_0.1530_	_0.1137_	_0.1743_	_0.1148_	_0.1697_	_0.1800_	_0.0829_	_0.2433_	_0.2089_
		SRNS	0.1457	0.1092	0.1668	0.1121	0.1636	0.1735	0.0799	0.2352	0.2017
		Proposed	**0.1550**	**0.1157**	**0.1768**	**0.1169**	**0.1729**	**0.1827**	**0.0837**	**0.2459**	**0.2117**
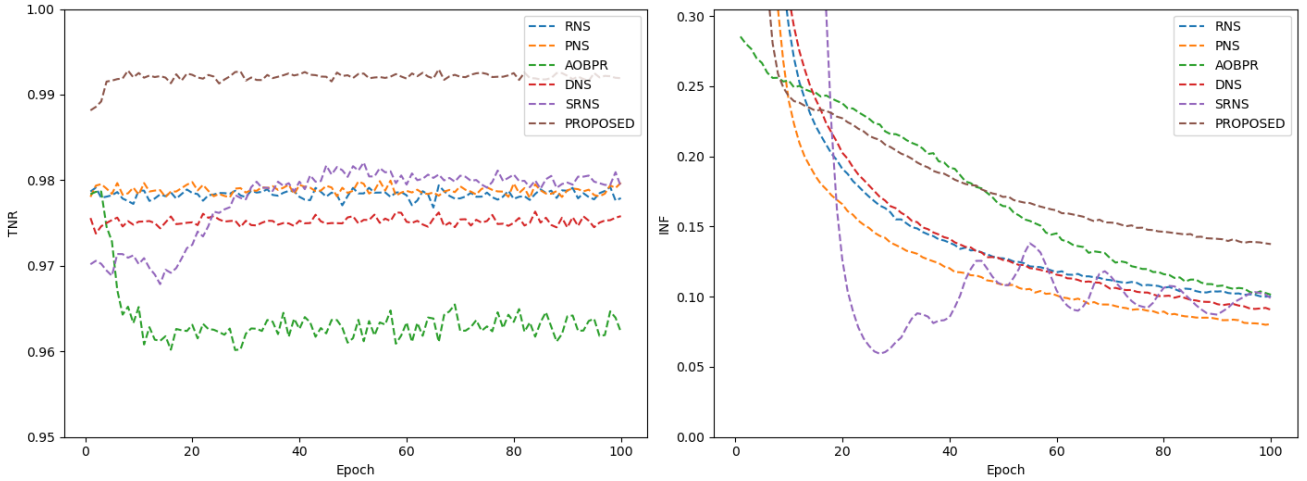
Fig. 4. The quality of selected negatives in each training epoch on the MovieLens-100K dataset.

1) Hyper-parameter selection: We first fix the size of candidate set $\mathcal{M}_u$ as 5 to investigate how λ affects the

performance. In particular, we search λ in the range of $\{0.1, 1, 5, 10, 15\}$. A larger value of λ means that putting more emphasis on the ranking gain from true negatives, and conversely more attention is paid to avoiding the risk of sampling false negatives if λ is small. We can observe from Fig. 5 that, when the value of λ increases from 0.1 to 1, $NDCG@20$ improves significantly, and achieves its maximum value when $\lambda = 5$. This does not mean that the optimal a-value is fixed during the model training.

Then we fix the optimal λ as 5 to study the impact of size of $\mathcal{M}_u$, and search $|\mathcal{M}_u|$ in the range of $\{1, 3, 5, 10, 15\}$. Note that when $|\mathcal{M}_u| = 1$, the proposed BNS reduced to the classical random negative sampling (RNS). When $|\mathcal{M}_u| > 1$, the Bayesian sampling criterion begins to play its role for selecting hard negatives. When $|\mathcal{M}_u| = |\mathcal{I}_u^-|$, the sampler h of Eq (32) is optimal sampler h^* given by Eq (25). We can observe from Fig. 5 that $NDCG@20$ achieves its maximum value when $|\mathcal{M}_u| = 5 \; or \; 10$. To reduce the time complexity, we set $|\mathcal{M}_u| = 5$.

The value of $NDCG@20$ decreases $|\mathcal{M}_u| > 10$. This experimental result is not as expected, since the larger value of $|\mathcal{M}_u|$, the easier it is to select the optimal negative. We believe that it is caused by the unreliable prior probability $P_{fn}(l)$ estimation. The unreliable prior information further results in the biased classification results denoted by $F(\hat{x}_l)$, leading to further deviation of negative signal unbias$(\cdot)$. An excessive $|\mathcal{M}_u|$ amplifies the adverse effect of negative signal bias, leads to performance degradation. More discussions can be found in Sec IV-C3.

2) Sensitivity analysis: We further study the sensitivity of BNS to λ, sample information $\hat{x}_l$, and prior information $P_{fn}(l)$.

- BNS-1: warm-start of λ. We set $\lambda = max(10 - \alpha * epoch, 2)$ to linearly decrease the value of λ as epoch number increases. That is, a larger λ to emphasis the ranking gain for sampling true negative in the initial stage, and a smaller λ to emphasis the loss for sampling false negatives in the later stages, where α is selected as 0.1.
- BNS-2: warm-start of sample information $\hat{x}_l$. We first adopt RNS to train a recommendation model for some epochs to learn a more reliable sample information $F(\hat{x}_l)$, and then resume the training by replacing it with our BNS.
- BNS-3: non-information prior distribution. For this case, BNS reduces to DNS that use only sample information $\hat{x}_l$ for negative sampling. For a single randomized trial, the probability of any item l been interacted is 1/1682, i.e., we set $P_{fn}(l) = 1/1682$ indiscriminately for any negative l, where 1682 is the total number of items.
- BNS-4: occupation information enhanced prior distribution. On the basis of Eq (17), We added a adjustment factor to improve the estimation of $P_{fn}(l)$:

$$P_{fn}(l) = \frac{pop_l}{N} \cdot (1 + \triangle o_{ul}),$$

where $\triangle o_{ul} = \frac{o_{ul} - \bar{o}_l}{\max o_l}$ indicating the deviation of the number of times of l has been preferred by u's occupation

from mean value. o_{ul} is the number of interactions of groups with same occupation as u on item l, $\bar{o}_l$ is the mean number of times of item l being interacted with by each occupation.

The experimental results are exhibited in Table III. From which we have the following findings:

Sensitivity of λ to BNS: The warm start of λ achieved better performance (BNS-1). As the degree of trade-off between sampling risk and gain, a larger value of λ means that we place more emphasis on the ranking gain from TN than rather than the risk from FN. The results show that hard negative instances are important for model learning, which is consistent with the findings of existing studies [9], [14]. We recommend a warm-start strategy of λ: larger value to emphasis the hard negatives in the initial stage, and small λ to avoid sampling false negatives in later stage.

Sensitivity of prior probability $P_{fn}(\cdot)$ to BNS: The results shows that BNS-3 achieved worse performance in the absence of prior information compared with standard BNS, while BNS-4 achieved better performance in the case of occupation-enhanced prior probability compared with standard BNS. The results show that BNS is sensitive to priori probability. The mechanism by which priori information affects negative sampling is: the unreliable prior probability $P_{fn}(\cdot)$ results in the biased classification results denoted by $F(\hat{x}.)$, leading to further deviation of negative signal unbias$(\cdot)$. An excessive size of candidate set $\mathcal{M}_u$ amplifies the adverse effect of negative signal bias, leads to performance degradation. Therefore, the larger size of $\mathcal{M}_u$ is the better if the prior probability $P_{fn}(\cdot)$ is reliable, otherwise an $\mathcal{M}_u$ of moderate size should be chosen. In particular, BNS is equivalent to DNS in the case of non-information prior distribution: BNS samples instances with appropriate $F(\hat{x})$-values and the DNS samples instances with appropriate ranking positions, while $F(\hat{x})$ and ranking position are with one-to-one mapping. By selecting the appropriate $|\mathcal{M}_u|$ and λ, BNS-3 achieves comparable performance with DNS. We recommend to select the most reliable priori information for modeling $P_{tn}(\cdot)$ or $P_{fn}(\cdot)$.

Sensitivity of sample information $\hat{x}_l$ to BNS: The warm start of the sample information $\hat{x}$ (BNS-2) did not achieve better performance than standard BNS as expected, we believe there are three reasons: (i) random sampling at the initial training stages is difficult to sample hard negatives, making the performance of BNS-2 degrade; (ii) $\hat{x}$ is endogenously determined by priori probability and the sampler h, therefore the warm-start of $\hat{x}$ has limited impact on the final ranking performance. (iii) the way we use sample information $F(\hat{x}.)$ is insensitive to small changes in $\hat{x}.$, thus the improved $\hat{x}$ has limited impact on improving the sampling quality. We believe that this property of BNS is an important manifestation of its robustness for different ranking models, as BNS is still able to sample high quality negative instances using prior information under the condition of the order relation does not hold (e.g., early training stage or weak ranking models).

3) Asymptotic optimal sampler: Next, we will show the asymptotic process of proposed sampler h to the optimal
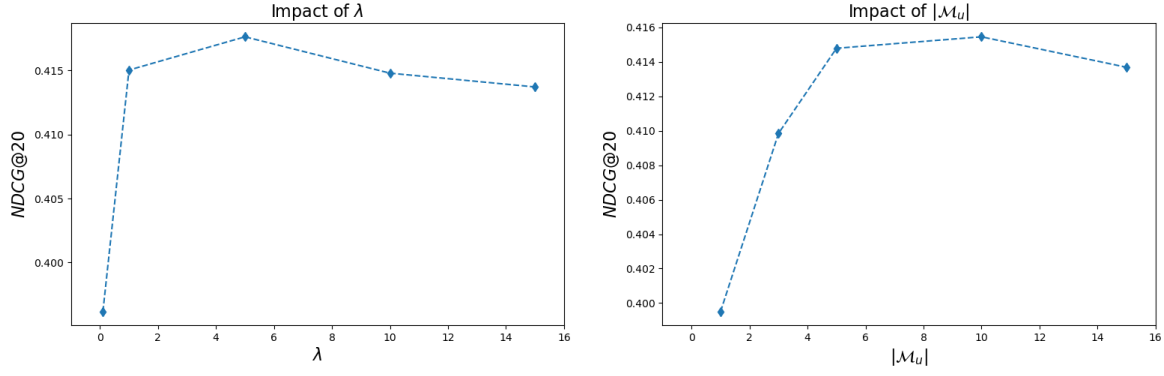
Fig. 5. The impact of λ and $|\mathcal{M}_u|$ w.r.t NDCG metric.

TABLE III
STUDY OF BNS .

Dataset	CF Model	Method	Top-5			Top-10			Top-20		
			Precision	Recall	NDCG	Precision	Recall	NDCG	Precision	Recall	NDCG
100K	MF	RNS	0.3900	0.1301	0.4143	0.3363	0.2164	0.3967	0.2724	0.3298	0.3962
		BNS	0.4205	0.1467	0.4558	0.3463	0.2290	0.4217	0.2762	0.3466	0.4176
		BNS-1	0.4237	0.1471	0.4551	0.3495	0.2305	0.4238	0.2762	0.3495	0.4197
		BNS-2	0.4148	0.1456	0.4449	0.3411	0.2245	0.4132	0.2738	0.3434	0.4125
		BNS-3	0.4048	0.1392	0.4266	0.3423	0.2282	0.4043	0.2720	0.3406	0.4030
		BNS-4	0.4262	0.1478	0.4566	0.3486	0.2305	0.4235	0.2792	0.3520	0.4216

sampler h^* given the ideal prior probability $P_{fn}(l)$. We set $P_{fn}(l) = (label(l) - 0.2)^2$, that is , $P_{fn}(l) = 0.64$ if $l \in fn$ otherwise $P_{fn}(l) = 0.04$. The asymptotic process is achieved by gradually increasing the size of $\mathcal{M}_u$, where λ is fixed as 5. Simulation results are presented in Table IV. By increasing the size of the candidate set size, the optimal sampler h^* is achieved without ranking performance degradation. This result validates our analysis in Sec IV-C1. Equipped with certain degree of a priori information, BNS achieved considerable performance even for simple dot-product based representation learning methods, demonstrating the great potential of negative sampling studies. The performance of optimal sampler (i.e., $|\mathcal{M}_u| = |\mathcal{I}_u^-|$) is an empirical upper bound for dot product-based model. Due to the existence of low-rank constraint thus limited expressiveness of matrix factorization, the recommendation performance cannot reach 1.

D. In-depth Analysis

The baselines contain the results of two illuminating ablation experiments: (i) PNS that only uses prior information (item popularity), and (ii) DNS, AOBPR and SRNS that only use sample information $\hat{x}_l$. In particular, BNS reduces to DNS in the case of an non-information prior distribution. BNS samples instances with appropriate $F(\hat{x})$-values and the DNS samples instances with appropriate ranking positions, while $F(\hat{x})$ and ranking position are with one-to-one mapping, so they achieved comparable performance. The shortcomings of the above two paradigms are obvious: the former can incorporate domain knowledge but independent of model status, resulting in a static sampling distribution; the latter exploits sample information but ignore priori knowledge, which will degrade performance especially in the scenarios of rich side information. The proposed BNS combines the priori information and sample information from the Bayesian perspective, and define the negative signal unbias(l) in a posterior probability sense. In turn, the optimal sampling rule can be given. We believe that the performance improvement of BNS stems from three aspects: (i) the prior information $P_{tn}(l)$ or $P_{fn}(l)$, and (ii) the use of sample information $\hat{x}_l$ that avoids negative sampling to conflict with classification results of the ranking model. (iii) BNS is theoretically optimal sampler that minimizes the empirical sampling risk.

V. RELATED WORK

Pairwise learning and pairwise loss have been widely applied in recommendation systems [7], [10], [19], [20]. Pairwise comparisons of positive instances and negative instances are first constructed to train a recommendation model. How to select negatives for pairwise comparisons, i.e., negative sampling, is a key to model training [7]–[9]. We review the related work of negative sampling for recommendation from two categories according to whether the sampling policy is fixed during the model training process.

TABLE IV
THE ASYMPTOTIC PROCESS TO THE OPTIMAL SAMPLER h^*.

Dataset	CF Model	BNS Size	Top-5			Top-10			Top-20		
			Precision	Recall	NDCG	Precision	Recall	NDCG	Precision	Recall	NDCG
100K	**MF**	$|\mathcal{M}_u| = 1$	0.3900	0.1301	0.4143	0.3363	0.2164	0.3967	0.2724	0.3298	0.3962
		$|\mathcal{M}_u| = 3$	0.4909	0.1567	0.5211	0.4220	0.2565	0.4942	0.3366	0.3872	0.4856
		$|\mathcal{M}_u| = 5$	0.5109	0.1612	0.5422	0.4329	0.2602	0.5092	0.3456	0.3925	0.4992
		$|\mathcal{M}_u| = 10$	0.5351	0.1696	0.5685	0.4589	0.2722	0.5365	0.3663	0.4081	0.5245
		$|\mathcal{M}_u| = 20$	0.5760	0.1828	0.6070	0.4885	0.2875	0.5695	0.3830	0.4196	0.5498
		$|\mathcal{M}_u| = 50$	0.6239	0.1989	0.6599	0.5252	0.3049	0.6146	0.4031	0.4312	0.5843
		$|\mathcal{M}_u| = 100$	0.6509	0.2104	0.6898	0.5382	0.3125	0.6346	0.4053	0.4321	0.5971
		$|\mathcal{M}_u| = 500$	0.6661	0.2183	0.7128	0.5412	0.3131	0.6487	0.4041	0.4300	0.6076
		$|\mathcal{M}_u| = |\mathcal{I}_u^-|$	0.6674	0.2184	0.7133	0.5429	0.3140	0.6495	0.4041	0.4292	0.6073

The first category is the *static negative sampling*. This kind of methods adopt a fixed sampling distribution for negative sampling during the whole training process. The most widely used is the random negative sampling (RNS) [10], [12], [13], [20], which uniformly samples negatives from un-labeled instances. Some have proposed to set sampling probability of a negative instance according to its popularity (interaction frequency), as so-called popularity-biased negative sampling(PNS) [3], [4], [18], [38], [39]. Among them, the most widely used sampling distribution is $p(j) \propto r_j^{0.75}$, where r_j is the interaction frequency of an item in the training dataset.

The second category is the *hard negative sampling* that adopts an adaptive sampling distribution targeting on sampling hard negative instances. The so-called hard negatives refer to those unlabeled instances that are similar to those positive instances in the embedding space [40]–[42]. Many hard negative sampling strategies have been proposed for personalized recommendation [7]–[9], [14]–[16]. For example, Zhang et al [8] and Steffen et al [7] propose to oversample higher scored thus higher ranked negatives that are argued to be more similar to positive instances. Some have proposed to exploit graph-based information for boosting negative sampling [21], [22], [43]–[46]. For example, Wang et al [21] and Wang et al [43] propose to leverage the types of relations on a knowledge graph to filter hard negatives. Another approach is to use the random walk on a graph for selecting hard negatives that are structurally similar to positive instances [22], [44]–[46].

Additional or prior information can be exploited for identifying and sampling hard negatives, Such side information are intuitive for distilling negative signals, such as users' connections in social networks [34], [35], geographical locations of users [47], [48], and additional interaction data such as viewed but non-clicked [33], [49]. Beside only sampling negatives from unlabeled instances, a novel class of methods is to generate a kind of virtual hard negatives from multiple negative instances or by using some generative method. For example, Huang et al. [15] propose to synthesize virtual hard negatives by hop mixing embeddings. Jun et al. [50] and Park et al. [14] design generative adversarial neural networks for generating virtual hard negatives.

VI. CONCLUSION

This paper has provided a comprehensive analysis on negative sampling for recommendation. On the basis of order relation analysis of negatives' scores, we derive the class conditional density of true negatives and that of false negatives, and provide an affirmative answer from a Bayesian viewpoint to distinguish true negatives from false negatives. Then, according to the asymptotic property of the empirical distribution function, we defined a model-agnostic posterior probability estimate of an instance being true negative as a quantitative negative signal measure. Lastly, we propose a Bayesian sampling rule to sample high-quality negative instances. It is the theoretically optimal sampling rule that minimizes the empirical sampling risk. Experiment studies have validated our arguments and findings.

The limitations of BNS are: (i) We adopt a simplistic approach for modeling prior probability, yet it is critical to the Bayesian-based methods. (ii) Due to the collaborative filtering mechanism, it is difficult to obtain the analytical solution of sampling loss $\triangle\mathcal{L}(l|i)$ from single update of a train triple. Eq (30) is just an approximation of sampling loss, which has much room for improvement. We note that negative sampling indeed faces an exploration-and-exploitation trade-off: Exploration suggests to prioritize those higher ranked informational instances been classified as false negative (positive labeled) by a ranking model; Exploitation indicates to favor those lower ranked unbiased instances been classified as true negative by the same ranking model. Future work can go further to generalize BNS to contrastive-based learning methods and investigate the optimal trade-off between such exploration and exploitation.

REFERENCES

[1] B. Jessa and D. Jesse, "Learning from positive and unlabeled data: a survey," *Machine Learning*, vol. 109, p. 719–760, 2020.

[2] G. Su, W. Chen, and M. Xu, "Positive-unlabeled learning from imbalanced data," in *Proceedings of the 30th International Joint Conference on Artificial Intelligence, Virtual Event*, 2021, pp. 2995–3001.

[3] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems*, 2013, p. 3111–3119.

[4] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web*, 2015, p. 1067–1077.

[5] X. Qin, N. Sheikh, B. Reinwald, and L. Wu, "Relation-aware graph attention model with adaptive self-adversarial training," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, pp. 9368–9376.

[6] H. Zhao, X. Yang, Z. Wang, E. Yang, and C. Deng, "Graph debiased contrastive learning with joint representation clustering," in *IJCAI*, 2021, pp. 3434–3440.

[7] S. Rendle and C. Freudenthaler, "Improving pairwise learning for item recommendation from implicit feedback," in *Proceedings of the 7th ACM international conference on Web Search and Data Mining*, 2014, pp. 273–282.

[8] W. Zhang, T. Chen, J. Wang, and Y. Yu, "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*, 2013, p. 785–788.

[9] J. Ding, Y. Quan, Q. Yao, Y. Li, and D. Jin, "Simplify and robustify negative sampling for implicit collaborative filtering," in *NeurIPS*, 2020.

[10] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, 2009, pp. 452–461.

[11] X. Wang, X. He, M. Wang, F. Feng, and T. S. Chua, "Neural graph collaborative filtering," in *SIGIR '2019*, Proceedings of the 42nd International ACM SIGIR Conference, pp. 2344–2353.

[12] H. Xiangnan, D. Kuan, W. Xiang, L. Yan, Z. Yongdong, and W. Meng, "Lightgcn: Simplifying and powering graph convolution network for recommendation." in *SIGIR*, 2020, p. 10.

[13] X. Yang and B. Wang, "Local ranking and global fusion for personalized recommendation," *Applied Soft Computing*, vol. 96, no. 1, p. 106636, 2020.

[14] D. H. Park and Y. Chang, "Adversarial sampling and training for semi-supervised information retrieval," in *WWW*, 2019, p. 1443–1453.

[15] T. Huang, Y. Dong, M. Ding, Z. Yang, W. Feng, X. Wang, and J. Tang, "Mixgcf: An improved training method for graph neural network-based recommender systems," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, p. 665–674.

[16] J. Ding, Y. Quan, X. He, Y. Li, and D. Jin, "Reinforced negative sampling for recommendation with exposure data," in *IJCAI*, 2019, pp. 2230–2236.

[17] Z. Yang, M. Ding, C. Zhou, H. Yang, J. Zhou, and J. Tang, "Understanding negative sampling in graph representation learning," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1666–1676.

[18] T. Chen, Y. Sun, Y. Shi, and L. Hong, "On sampling strategies for neural network-based collaborative filtering," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, p. 767–776.

[19] W. Pan and L. Chen, "Gbpr: Group preference based bayesian personalized ranking for one-class collaborative filterin," in *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[20] R. Yu, Y. Zhang, Y. Ye, L. Wu, C. Wang, Q. Liu, and E. Chen, "Multiple pairwise ranking with implicit feedback," in *Proceedings of the 27th ACM Int. Conference on Information and Knowledge Management, ACM*, 2018, pp. 1727–1730.

[21] X. Wang, Y. Xu, X. He, Y. Cao, M. Wang, and Chua, "Reinforced negative sampling over knowledge graph for recommendation," in *WWW*, 2020, pp. 99–109.

[22] J. Chen, C. Wang, S. Zhou, Q. Shi, Y. Feng, and C. Chen, "Samwalker: Social recommendation with informative sampling strategy," in *WWW*, 2019, pp. 228–239.

[23] T. Zhao, J. McAuley, and I. King, "Improving latent factor models via personalized feature projection for one class recommendation," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 2015, p. 821–830.

[24] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 549–558, p. 2016.

[25] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[26] M. U. Gutmann and A. Hyvärinen, "Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics." *Journal of machine learning research*, vol. 13, no. 2, 2012.

[27] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[28] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 297–304.

[29] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *International Conference on Machine Learning*, 2020, pp. 9929–9939.

[30] L. Xu, J. Lian, W. X. Zhao, M. Gong, L. Shou, D. Jiang, X. Xie, and J.-R. Wen, "Negative sampling for contrastive representation learning: A review," *arXiv preprint arXiv:2206.00212*, 2022.

[31] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang, "Self-supervised learning: Generative or contrastive," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[32] V. Glivenko, "Sulla determinazione empirica delle leggi di probabilita," *Gion. Ist. Ital. Attauri.*, vol. 4, pp. 92–99, 1933.

[33] J. Ding, Y. Quan, X. He, Y. Li, and D. Jin, "Reinforced negative sampling for recommendation with exposure data," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 2019, pp. 2230–2236.

[34] T. Zhao, J. McAuley, and I. King, "Leveraging social connections to improve personalized ranking for collaborative filtering," in *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, 2014, pp. 261–270.

[35] X. Wang, W. Lu, M. Ester, C. Wang, and C. Chen, "Social recommendation with strong and weak ties," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 2016, pp. 5–14.

[36] A. Herschtal and B. Raskutti, "Optimising area under the roc curve using gradient descent," in *Proceedings of the twenty-first International conference on Machine learning*, 2004, p. 49.

[37] S. Liu, B. Wang, and M. Xu, "Event recommendation based on graph random walking and history preference reranking," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2017, p. 861–864.

[38] M. Grbovic, N. Djuric, V. Radosavljevic, F. Silvestri, and N. Bhamidipati, "Context- and content-aware embeddings for query rewriting in sponsored search," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile,August 9-13, 2015*, 2015, pp. 383–392.

[39] H. Caselles-Dupré, F. Lesaint, and J. Royo-Letelier, "Word2vec applied to recommendation: Hyperparameters matter," in *Proceedings of the 12th ACM Conference on Recommender Systems*, 2018, pp. 352–356.

[40] J. Rao, H. He, and J. Lin, "Noise-contrastive estimation for answer selection with deep neural networks," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 1913–1916, p. 2016.

[41] Y. Zhang, Q. Yao, Y. Shao, and L. Chen, "Nscaching: simple and efficient negative sampling for knowledge graph embedding," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, 2019, pp. 614–625.

[42] Z. Sun, W. Hu, Q. Zhang, and Y. Qu, "Bootstrapping entity alignment with knowledge graph embedding," in *IJCAI*, 2018, pp. 4396–4402.

[43] Y. Wang, Z. Liu, Z. Fan, L. Sun, and P. S. Yu, "Dskreg: Differentiable sampling on knowledge graph for recommendation with relational gnn," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 3513–3517.

[44] J. Li, F. Xia, W. Wang, Z. Chen, N. Y. Asabere, and H. Jiang, "Acrec: a co-authorship based random walk model for academic collaboration recommendation," in *Proceedings of the 23rd international conference on World Wide Web*, 1209–1214, p. 2014.

[45] C. Wang, J. Chen, S. Zhou, Q. Shi, Y. Feng, and C. Chen, "Samwalker++: recommendation with informative sampling strategy," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[46] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *KDD*, 2018, pp. 974–983.

[47] F. Yuan, J. M. Jose, G. Guo, L. Chen, H. Yu, and R. S. Alkhawaldeh, "Joint geo-spatial preference and pairwise ranking for point-of-interest recommendation," in *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, 2016, pp. 46–53.

[48] W. Liu, Z.-J. Wang, B. Yao, and J. Yin, "Geo-alm: Poi recommendation by fusing geographical information and adversarial learning mechanism." in *IJCAI*, 2019, pp. 1807–1813.

[49] J. Ding, F. Feng, X. He, G. Yu, Y. Li, and D. Jin, "An improved sampler for bayesian personalized ranking by leveraging view data," in *Companion Proceedings of the The Web Conference 2018*, 2018, p. 13–14.

[50] J. Wang, L. Yu, W. Zhang, Y. Gong, Y. Xu, B. Wang, P. Zhang, and D. Zhang, "Irgan: A minimax game for unifying generative and discriminative information retrieval models," in *SIGIR*, 2017, p. 515–524.