Bayesian Negative Sampling for Recommendation

Bin Liu
School of Electronic Information and Communications
Huazhong University of Science and Technology (HUST)
Wuhan, China,
Email: 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, discriminating false negative from true negative for unbiased negative sampling remains an unsolved problem. 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.

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 process.

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, 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:

$$L_{\text{loss}} \equiv \max_{\Theta} \sum_{(u,i,j) \in \mathcal{I}_u^+} \ln \sigma(\hat{x}_{ui} - \hat{x}_{uj}),$$

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. $\Theta$ contains model trainable parameters, and $\lambda$ 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 $\text{sim}$, 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

$$\frac{\partial L_{\text{loss}}}{\partial \hat{x}_{uj}} = [1 - \sigma(\hat{x}_{ui} - \hat{x}_{uj})](1),$$

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,

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

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.,

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

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

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

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 (i.e., an unlabeled but actually a positive instance to user $u$). If the model treats the sampled false negative $j$ as a true negative instance, it will cause 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.

We note that the adverse effects of sampling bias are not limited to the BPR loss. Some other contrastive-based loss functions such as binary cross-entropy loss may also suffer from incorrect negative sampling: When sampling a false negative, the wrong gradient direction of the loss function w.r.t this sampled instance would lead to an incorrect training of the model w.r.t this sampled instance. The feature representation for this false negative would be pushed away from the anchor data point $u$; While it should be close to the anchor data point in the embedding space.

### 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 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
\[
\hat{x}_{tn} \leq \hat{x}_{fn},
\]
(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 all instances are predicted using the same encoder and the same score function, it is safe to assume that the predicted scores \(\hat{x}\) are identically and independently distributed with probability density function \(f(\hat{x})\) and cumulative distribution function \(F(\hat{x}) = P(X \leq \hat{x})\).

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:
\[
X_{tn} \leq X_{fn}.
\]
(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 \(\hat{x}_{fn} \in (\hat{x}_{fn} + dx, \infty)\). The probability differential of \(X_{tn}\) can be computed by
\[
g(\hat{x}_{tn}) \, dx = \int 2 dx \times P(\hat{x}_{tn} \leq X_{tn} \leq \hat{x}_{tn} + dx)
\times P(\hat{x}_{tn} + dx \leq X_{fn} \leq \infty) + o(dx)
(8)
\]
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 \(X_{fn} \in (\hat{x}_{fn} + dx, \infty)\), \((o(dx))\) 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:
\[
g(\hat{x}_{tn}) = \lim_{dx \to 0} \int \frac{2 f(\hat{x}_{tn})dx[1 - F(\hat{x}_{tn} + dx)] + o(dx)}{dx}
(9)
\]
Likewise, the distribution of false negatives is given by:
\[
h(\hat{x}_{fn}) = 2F(\hat{x}_{fn}) f(\hat{x}_{fn}).
(10)
\]

**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 \(g(x) = 2f(x)[1 - F(x)] \geq 0, h(x) = 2f(x)F(x) \geq 0\).

\[
\int_{-\infty}^{\infty} g(x)\, dx = \int_{-\infty}^{\infty} 2f(x)[1 - F(x)]\, dx
= 2 \int_{-\infty}^{\infty} f(x)\, dx - 2 \int_{-\infty}^{\infty} f(x)F(x)\, dx
= 2 - 2 \int_{-\infty}^{\infty} F(x)\, dx
= 1,
\]
\[
\int_{-\infty}^{\infty} h(x)\, dx = \int_{-\infty}^{\infty} 2f(x)F(x)\, dx
= 2 \int_{-\infty}^{\infty} F(x)\, dx
= 1.
\]

1) **Empirical 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 uninteracted 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 \(x \sim \text{Gamma}(\alpha, \lambda)\) (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 encoders 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 \(F_n(\cdot) = \frac{1}{n} \sum_{i=1}^{n} I(x_i \leq \cdot)\) 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 \(F(\cdot)\).
distribution function is easy to implement. So we use the fraction form of conditional probability to define unbiasedness to eliminate the density function $f(\cdot)$:

$$\text{unbias}(l) \triangleq \frac{P(tn|\hat{x}_l)}{P(\hat{x}_l)} = \frac{P(tn|\hat{x}_l)P_{tn}(l)}{P(\hat{x}_l|tn)P_{tn}(l) + P(\hat{x}_l|fn)P_{fn}(l)}$$

$$= \frac{P(\hat{x}_l|tn)P_{tn}(l)}{f(\hat{x}_l)[1 - F(\hat{x}_l)]P_{tn}(l) + F(\hat{x}_l)f(\hat{x}_l)P_{fn}(l)}$$

$$= \frac{[1 - F(\hat{x}_l)][1 - P_{fn}(l)]}{1 - F(\hat{x}_l) - P_{fn}(l) + 2F(\hat{x}_l)P_{fn}(l)}. \quad (16)$$

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

$$F(\hat{x}_l) = \frac{\#\{\hat{x} \leq \hat{x}_l | l \in \mathcal{I}_n\}}{\#\{\mathcal{I}_n\}}. \quad (17)$$

$P_{fn}(l)$ 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

$$P_{fn}(l) = \frac{pop_l}{N}. \quad (18)$$

Lemma 0.1 (Unbiased negative signal): If $pop_l \sim B(N, P_{fn}(l))$, then unbiasedness measure given by Eq (16) is an unbiased estimator for $l$ being true negative.

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

$$Y = \begin{cases} 1, & P = \theta \\ 0, & P = 1 - \theta, \end{cases} \quad (19)$$

where $\theta$ 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
\[
E[P_{fn}(l)] = \frac{E^{(pop)}}{N} = \theta
\]  

Eq. (16) 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 \(F(\hat{x}_l)\). 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 list. 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 \((P_{fn}(l) \lor P_{fn}(l))\). We note that the metric of Eq. (16) unifies 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 \(\text{unbias}(\cdot)\) measure lies in the theoretical foundation of posterior probability, which combines prior 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 and domain knowledge can also be exploited for modeling prior probability.

\[E[F(X)] = \int_{-\infty}^{\infty} F(x) f(x) dx = \int_{-\infty}^{\infty} F(x) dF(x) = \frac{1}{2} F^2(x) \bigg|_{x=-\infty}^{x=\infty} = \frac{1}{2} \tag{22}\]

\[E[\text{unbias}(l)] = \frac{(1-\frac{1}{2})(1-\theta) - \frac{1}{2} - \theta + 2 \cdot \frac{1}{2} \cdot \theta}{1 - \theta} \tag{23}\]

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

Fig. 3. Numerical plots of posterior probability \(\text{unbias}(l)\) by Eq. (16).

The definition of unbiasedness is \textit{posterior probability of} \(l\) being true negative (negative signal). The density function \(f(\hat{x}_l)\) determined by encoder and score function is eliminated due to the fractional expression of Eq. (16). This suggests that it is model-independent metric and can be generalized to different encoders. We note that the order relation also holds for other contrastive-based loss functions that essentially encourage the model to score positive instances higher than those of negative instances. The \(\text{unbias}(\cdot)\) metric is also applicable to such loss functions like the binary cross-entropy and InfoNCE \([27]\). Eq. (16) 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 \(F(\hat{x}_l)\). 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 list. 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 \((P_{fn}(l) \lor P_{fn}(l))\).

We note that the metric of Eq. (16) unifies 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 \(\text{unbias}(\cdot)\) measure lies in the theoretical foundation of posterior probability, which combines prior 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 and domain knowledge can also be exploited for modeling prior probability.

\[D. \text{Negative Sampling Algorithm}\]

The ranking objective of Eq (1) is the analogy to \textit{AUC} metric \([10]\). It replaces the non-differentiable Heaviside function used in \textit{AUC} metric with the differentiable loss \(\ln \sigma(\cdot)\), which is a common practice when optimizing for \textit{AUC} \([10]\), \([36]\). When performing a single negative sampling, the unlabeled instance \(l\) is directly assigned a negative gradient, resulting in a \textit{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 \(\Delta \mathcal{L}_{fn}(l)\); (ii) increasing of \(\mathcal{L}\) by sampling a true negative, denoted as \(\Delta \mathcal{L}_{tn}(l)\).

\textbf{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 $\Delta L(l|i)$ over the distribution of $l$’s ground truth label $c$:

$$R(l|i) = \mathbb{E}_{l \sim P(l|i)} \Delta L(l|i)$$

$$= [1 - P(trn|l)] \cdot \Delta L(f_u(l|i)) + P(trn|l) \cdot \Delta L_{tsn}(l|i)$$

(24)

where $P(trn|l)$ is the posterior probability of $l \in trn$ and can be computed by unbiased $(l)$, and $1 - P(trn|l)$ 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:

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

(25)

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

$$h^* = \arg \min_{i \in \mathcal{I}_u} R(l|i)$$

(26)

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

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

(27)

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

$$R(h^*) - R(h) = \mathbb{E}_{l \sim P(i)} [R(h^*|i) - R(h|i)]$$

$$= \sum_{i} P(i) [R(h^*|i) - R(h|i)]$$

$$\leq 0.$$  

(28)

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

$$\inf \{R(hl)\} = R(h^*|i) = \mathbb{E}_{l \sim P(i)} R(h^*|i).$$

(29)

The Taylor expansion of ranking objective of $\tilde{L}$ around point $\hat{x}_{ul}$ is

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

(30)

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

$$\Delta \mathcal{L}(l|i) \approx \left\{ \begin{array}{ll}
\text{info}(l), & \text{if } l = fn \\
-\lambda \text{info}(l), & \text{if } l = tn.
\end{array} \right.$$  

(31)

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

$$R(l|i) = P(trn|l) \cdot \text{info}(l) - P(trn|l) \cdot \lambda \text{info}(l).$$

(32)

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

$$j = \arg \min_{i \in \mathcal{I}_u} R(l|i)$$

$$= \arg \min_{i \in \mathcal{I}_u} [1 - \text{unbias}(l)] \cdot \text{info}(l)$$

$$- \lambda \cdot \text{unbias}(l) \cdot \text{info}(l)$$

$$= \arg \min_{i \in \mathcal{I}_u} \text{info}(l) \cdot [1 - (1 + \lambda)\text{unbias}(l)]$$

(33)

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

**Complexity:** We summarize the proposed negative sampling algorithm as: for each negative instances in the candidate set $\mathcal{M}_u$, (i) computing prior probability which usually takes $O(1)$ since it is statically distributed, (ii) computing $F(\hat{x}_i)$ as probabilistic prediction by ranking model (likelihood) $(O(|\mathcal{I}|))$, (iii) computing the negative signal $\text{unbias}(l)$ as posterior probability $(O(1))$, and (iv) perform Bayesian negative sampling based on Eq (33) $(O(1))$. So the proposed BNS has a linear time complexity. Algorithm 1 provides the pseudo-codes of our proposed negative sampling algorithm.

**IV. EXPERIMENT**

**A. Experiment Settings**

1) **Dataset:** We conduct experiments on three public datasets, including MovieLens-100k5, MovieLens-1M, and Yahoo!-R3.6 [13] They contain users’ ratings on items according to a

---

*Similar to positive instances in the embedding space.

5 https://grouplens.org/datasets/movielens

6 http://webscope.sandbox.yahoo.com/catalog.php?datatypes=r.
We note that all these competitors also only use the rank (by Eq. (17)). Using priori statistical information for over-bias ∈R^P DNS RNS Sample a mini-batch each PNS [3], [4], [18], (b) hard negative sampling with

do.

We compare the proposed algorithm with unbias \[9\]: Over-sampling global higher ranked (\[7\] and \[3\], \[4\], [18]: (Popularity-biased Negative Sampling)) calculate

AOBPR \[7\] and \[3\], \[4\], [18]: (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:

\[
TNR = \frac{#TN}{#TN + #FN},
\]

\[
INF = \frac{\sum_j info(j) \cdot sgn(j)}{#TN + #FN},
\]

where #TN(#FN) is the number of sampled true (false) negatives in each training epoch. Eq. (34) evaluates the proportion of sampled true negatives in each training epoch, i.e., true negative rate (TNR). sgn(j) is the indicator function: sgn(j) = 1 if the sampled item’s label is TN; Otherwise, sgn(j) = -1 as a penalty for sampling the FN instance. The informativeness (INF) defined Eq. (35) 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 α = 0.01, regulation constant reg = 0.01 and training epoch T = 100, batch size b = 1. (b) Light GCN [12]: embedding dimension d = 32, the initial learning rate α = 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.

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

| 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:

\[
TNR = \frac{#TN}{#TN + #FN},
\]

\[
INF = \frac{\sum_j info(j) \cdot sgn(j)}{#TN + #FN},
\]

where #TN(#FN) is the number of sampled true (false) negatives in each training epoch. Eq. (34) evaluates the proportion of sampled true negatives in each training epoch, i.e., true negative rate (TNR). sgn(j) is the indicator function: sgn(j) = 1 if the sampled item’s label is TN; Otherwise, sgn(j) = -1 as a penalty for sampling the FN instance. The informativeness (INF) defined Eq. (35) 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 α = 0.01, regulation constant reg = 0.01 and training epoch T = 100, batch size b = 1. (b) Light GCN [12]: embedding dimension d = 32, the initial learning rate α = 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.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
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 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 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 unbiased(·)-value from the candidate set $M_u$:

$$ j = \arg \max_{l \in M_u} \text{unbias}(l) \quad (36) $$

$T_u$ is a small candidate set containing randomly selected negative instances from $M_u$. 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 $R(\cdot|\cdot)$-value from the candidate set $M_u$:

$$ j = \arg \min_{l \in M_u} \text{info}(l) \cdot |1 - (1 + \lambda) \text{unbias}(l)| \quad (37) $$

For above sampling methods, the size of $M_u$ 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 on MovieLens-100K dataset using the MF model.

1) Hyper-parameter selection: We first fix the size of candidate set $M_u$ as 5 to investigate how $\lambda$ affects the performance. In particular, we search $\lambda$ in the range of $\{0.1, 1, 5, 10, 15\}$. A larger value of $\lambda$ 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 $\lambda$ is small. We can observe from Fig. 5 that, when the value of $\lambda$ 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 value is fixed during the model training.

Then we fix the optimal $\lambda = 5$ to study the impact of size of $M_u$, and search $|M_u|$ in the range of $\{1, 3, 5, 10, 15\}$. Note that when $|M_u| = 1$, the proposed BNS reduced to the classical random negative sampling (RNS). When $|M_u| > 1$, the Bayesian sampling criterion begins to play its role for selecting hard negatives. When $|M_u| = |T_u|$, the sampler $h$ of Eq (33) is optimal sampler $h^*$ given by Eq (26). We can observe from Fig. 5 that $NDCG@20$ achieves its maximum value when $|M_u| = 5 \text{ or } 10$. To reduce the time complexity, we set $|M_u| = 5$.

The value of $NDCG@20$ decreases $|M_u| > 10$. This experimental result is not as expected, since the larger value of $|M_u|$, 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 $F(\tilde{x}_i)$.
leading to further deviation of negative signal unbiased(·). An excessive |M_u| amplifies the adverse effect of negative signal bias, leads to performance degradation. More discussions can be found in Sec IV-C4.

2) Sensitivity analysis: We further study the sensitivity of BNS to λ, sample information  x_t, and prior information P_u(l).

- BNS-1: warm-start of λ. We set λ = max(10 − α * epoch, 2) to linearly decrease the value of λ as epoch number increases. That is, a larger λ to emphasize the ranking gain for sampling true negative in the initial stage, and a smaller λ to emphasize the loss for sampling false negatives in the later stages, where α is selected as 0.1.
- BNS-2: warm-start of sample information  x_t. We first adopt RNS to train a recommendation model for some

| Dataset | CF Model | Method | Precision | Recall | NDCG | Precision | Recall | NDCG |
|---------|----------|--------|-----------|--------|------|-----------|--------|------|
| 100K    | MF       | RNS    | 0.3900    | 0.1301 | 0.4143 | 0.3363    | 0.2164 | 0.3967 |
|         |          | PNS    | 0.2647    | 0.0864 | 0.2694 | 0.2329    | 0.1475 | 0.2637 |
|         |          | AOBPR  | 0.3970    | 0.1375 | 0.4186 | 0.3308    | 0.2163 | 0.3942 |
|         |          | DNS    | 0.4053    | 0.1414 | 0.4314 | 0.3348    | 0.2214 | 0.4042 |
|         |          | SRNS   | 0.3951    | 0.1342 | 0.4176 | 0.3394    | 0.2174 | 0.3998 |
|         |          | Proposed | 0.4205   | 0.1467 | 0.4558 | 0.3463    | 0.2290 | 0.4217 |
|         | LightGCN | RNS    | 0.3426    | 0.1453 | 0.4544 | 0.3571    | 0.2319 | 0.4275 |
|         |          | PNS    | 0.3527    | 0.1266 | 0.3816 | 0.3015    | 0.2117 | 0.3660 |
|         |          | AOBPR  | 0.3911    | 0.1407 | 0.4200 | 0.3315    | 0.2276 | 0.4007 |
|         |          | DNS    | 0.4278    | 0.1425 | 0.4580 | 0.3612    | 0.2336 | 0.4331 |
|         |          | SRNS   | 0.4195    | 0.1440 | 0.4580 | 0.3564    | 0.2333 | 0.4275 |
|         |          | Proposed | 0.4318   | 0.1518 | 0.4640 | 0.3671    | 0.2410 | 0.4368 |
| 1M      | MF       | RNS    | 0.3843    | 0.0855 | 0.4027 | 0.3353    | 0.1430 | 0.3737 |
|         |          | PNS    | 0.3461    | 0.0753 | 0.3634 | 0.3004    | 0.1250 | 0.3356 |
|         |          | AOBPR  | 0.3946    | 0.0954 | 0.4135 | 0.3416    | 0.1549 | 0.3837 |
|         |          | DNS    | 0.4066    | 0.0991 | 0.4272 | 0.3521    | 0.1620 | 0.3965 |
|         |          | SRNS   | 0.3955    | 0.0934 | 0.4225 | 0.3408    | 0.1609 | 0.4042 |
|         |          | Proposed | 0.4207   | 0.1062 | 0.4324 | 0.3518    | 0.1703 | 0.4191 |
|         | LightGCN | RNS    | 0.4095    | 0.0953 | 0.4305 | 0.3512    | 0.1547 | 0.3985 |
|         |          | PNS    | 0.3658    | 0.0907 | 0.3855 | 0.3152    | 0.1486 | 0.3564 |
|         |          | AOBPR  | 0.4073    | 0.0997 | 0.4286 | 0.3535    | 0.1626 | 0.3982 |
|         |          | DNS    | 0.4130    | 0.0972 | 0.4342 | 0.3552    | 0.1577 | 0.4002 |
|         |          | SRNS   | 0.4026    | 0.0973 | 0.4239 | 0.3515    | 0.1526 | 0.3953 |
|         |          | Proposed | 0.4228   | 0.1087 | 0.4438 | 0.3639    | 0.1612 | 0.4088 |
| Yahoo   | MF       | RNS    | 0.1196    | 0.0875 | 0.1326 | 0.0915    | 0.1367 | 0.1401 |
|         |          | PNS    | 0.1186    | 0.0876 | 0.1301 | 0.0927    | 0.1360 | 0.1378 |
|         |          | AOBPR  | 0.1012    | 0.0741 | 0.1115 | 0.0798    | 0.1165 | 0.1184 |
|         |          | DNS    | 0.1251    | 0.0917 | 0.1390 | 0.0957    | 0.1399 | 0.1449 |
|         |          | SRNS   | 0.1141    | 0.0855 | 0.1285 | 0.0904    | 0.1358 | 0.1383 |
|         |          | Proposed | 0.1303   | 0.0975 | 0.1470 | 0.1002    | 0.1485 | 0.1542 |
|         | LightGCN | RNS    | 0.1076    | 0.0797 | 0.1214 | 0.0809    | 0.1185 | 0.1254 |
|         |          | PNS    | 0.1462    | 0.1120 | 0.1635 | 0.1048    | 0.1552 | 0.1612 |
|         |          | AOBPR  | 0.1530    | 0.1137 | 0.1743 | 0.1148    | 0.1697 | 0.1800 |
|         |          | SRNS   | 0.1457    | 0.1092 | 0.1668 | 0.1121    | 0.1636 | 0.1735 |
|         |          | Proposed | 0.1550   | 0.1157 | 0.1768 | 0.1169    | 0.1729 | 0.1827 |

Fig. 4. The quality of selected negatives in each training epoch on the MovieLens-100K dataset.
epochs to learn a more reliable sample information $F(x)$, 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 $x$ for negative sampling. For a single randomized trial, the probability of any item $l$ being interacted is $1/1682$, i.e., we set $p_R(l) = 1/1682$ 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 (18), We added an adjustment factor to improve the estimation of $P_R(l)$:

$$P_R(l) = \frac{\text{pop}_l}{N} \cdot (1 + \Delta \alpha_{ul}),$$

where $\Delta \alpha_{ul} = \frac{\alpha_{ul} - \bar{\alpha}_l}{\max \alpha_{ul}}$, indicating the deviation of the number of times of $l$ has been preferred by $u$’s occupation from mean value. $\alpha_{ul}$ is the number of interactions of groups with same occupation as $u$ on item $l$, $\bar{\alpha}_l$ 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 $\lambda$ to BNS**: The warm start of $\lambda$ achieved better performance (BNS-1). As the degree of trade-off between sampling risk and gain, a larger value of $\lambda$ 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 $\lambda$: larger value to emphasize the hard negatives in the initial stage, and small $\lambda$ to avoid sampling false negatives in later stage.

**Sensitivity of prior probability $P_R(\cdot)$ 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 $P_R(\cdot)$ results in the biased classification results denoted by $F(x)$, leading to further deviation of negative signal unbiased. 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 $P_R(\cdot)$ 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 $F(x)$-values and the DNS samples instances with appropriate ranking positions, while $F(x)$ and ranking position are with one-to-one mapping. By selecting the appropriate $|\mathcal{M}_u|$ and $\lambda$, BNS-3 achieves comparable performance with DNS. We recommend to select the most reliable priori information for modeling $P_R(\cdot)$ or $P_R(\cdot)$.

**Sensitivity of sample information $x$ to BNS**: The warm start of the sample information $x$ (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) $x$ is endogenously determined by priori probability and the sampler $h$, therefore the warm-start of $x$ has limited impact on the final ranking performance. (iii) the way we use sample information $F(x)$ is insensitive to small changes in $x$, thus the improved $x$ 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) Applications to different loss functions: We also conduct experiments to examine the applicability of BNS to other loss functions. We choose two contrastive-based loss function, namely, the binary cross-entropy (BCE) and InfoNCE loss function. We use the matrix factorization model as the encoder and replace the BPR loss [10] with the BCE loss [38] and InfoNCE [27] loss, respectively. The hyper-parameters remain the same as that in using the BPR loss. Specifically, the number of negative instances $N$ of the InfoNCE loss is fixed as 1. Table IV presents the results using different loss functions. It can be seen that the BNS has achieved significant improvements compared with the random negative sampling RNS with all the three

![Fig. 5. The impact of $\lambda$ and $|\mathcal{M}_u|$ w.r.t NDCG metric.](image-url)
loss functions, exhibiting the good applicability of the BNS.

4) Asymptotic optimal sampler: Next, we will show the
asymptotic process of proposed sampler $h$ to the optimal
sampler $h^*$ given the ideal prior probability $P_{f_n}(l)$. We set
$P_{f_n}(l) = (\text{label}(l) - 0.2)^2$, that is, $P_{f_n}(l) = 0.64$ if $l \in f_n$ otherwise $P_{f_n}(l) = 0.04$. The asymptotic process is achieved
by gradually increasing the size of $\mathcal{M}_u$, where $\lambda$ is fixed as
5. Simulation results are presented in Table V. 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 recommen-
dation 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, AOibPR and SRNS that only use
sample information $\hat{x}_t$. 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 $\text{unbiased}(l)$ 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 $P_{f_n}(l)$ or $P_{f_n}(l)$, and (ii) the use of
sample information $\hat{x}_t$ 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 distribution
is fixed during the model training process.

The first category is the static negative sampling. This kind
of methods adopts a fixed sampling distribution for negative
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], [39], [40]. 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 [41]–[43]. 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],
[44]–[47]. For example, Wang et al [21] and Wang et al [44]
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], [45]–[47].

Additional or prior information can be exploited for identi-
ifying and sampling hard negatives. Such side information are
intuitive for distilling negative signals, such as users’ connec-
tions in social networks [34], [35], geographical locations of
users [48], [49], and additional interaction data such as viewed
but non-clicked [33], [50]. Beside only sampling negatives from
unlabeled instances, a novel class of methods is to generate a

| Dataset | CF Model | Method | Top-5 | Top-10 | Top-20 |
|---------|----------|--------|------|-------|-------|
| 100K    | MF       | RNS    | 0.3900 | 0.1301 | 0.4143 |
|         |          | DNS    | 0.4205 | 0.1467 | 0.4558 |
|         |          | BNS-1  | 0.4237 | 0.1471 | 0.4551 |
|         |          | BNS-2  | 0.4148 | 0.1456 | 0.4449 |
|         |          | BNS-3  | 0.4048 | 0.1392 | 0.4266 |
|         |          | BNS-4  | 0.4262 | 0.1478 | 0.4566 |

TABLE III

STUDY OF BNS.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
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. [51] 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, we derive the class conditional density of true negatives and that of false negatives. According to the asymptotic property of the empirical distribution function, we define a model-agnostic posterior probability estimate of an instance being true negative as a quantitative negative signal measure. Lastly, we propose a Bayesian negative sampling rule to sample high-quality negative instances, which is the theoretically optimal sampling rule minimizing the empirical sampling risk.

The core contribution of BNS is that it specifies the negative signal measure in the sense of posterior probability. BNS unifies the existing two paradigms for extracting negative signals by combining static prior information and dynamic sampling information from the Bayesian perspective. We note that the BNS does not specify methods for modeling prior probability. Instead, $P_n(l)$ (or $P_f(l)$) provides an interface for those methods modeling prior probability with side information. In addition, the BNS can also be applied to other contrastive-based loss functions given that a model aims at scoring positive instances higher than negative instances.

Due to the collaborative filtering mechanism, it is difficult to obtain the analytical solution of sampling loss $\Delta L(l|i)$ from single update of a train triple. Eq (31) 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.

Acknowledgements: We also want to use our BNS algorithm on MindSpore\(^7\), which is a new deep learning computing framework. These problems are left for future work.

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. Sheik, 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. 3343–3440.
[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.

\(^7\)http://www.mindspore.cn/
