Learning-Efficient Yet Generalizable Collaborative Filtering for Item Recommendation

Yuanhao Pu [1] Xiaolong Chen [1] Xu Huang [2] Jin Chen [3] Defu Lian [2 4] Enhong Chen [2 4]

Abstract

The weighted squared loss is a common component in several Collaborative Filtering (CF) algorithms for item recommendation, including the representative implicit Alternating Least Squares (iALS). Despite its widespread use, this loss function lacks a clear connection to ranking objectives such as Discounted Cumulative Gain (DCG), posing a fundamental challenge in explaining the exceptional ranking performance observed in these algorithms. In this work, we make a breakthrough by establishing a connection between squared loss and ranking metrics through a Taylor expansion of the DCG-consistent surrogate loss—softmax loss. We also discover a new surrogate squared loss function, namely **R**anking-**G**eneralizable **Squared** (**RG**^{2}) loss, and conduct thorough theoretical analyses on the DCG-consistency of the proposed loss function. Later, we present an example of utilizing the **RG**^{2} loss with Matrix Factorization (MF), coupled with a generalization upper bound and an ALS optimization algorithm that leverages closed-form solutions over all items. Experimental results over three public datasets demonstrate the effectiveness of the **RG**^{2} loss, exhibiting ranking performance on par with, or even surpassing, the softmax loss while achieving faster convergence.

1. Introduction

Collaborative filtering is a typical technique in item recommendations that leverages similarities between user behaviors to predict user preferences. The objective functions in CF for item recommendation tasks, which performs recommendation with implicit feedback, can be separated into two tracks: **sampling** methods and **non-sampling** methods. **Sampling** methods, such as BPR (Rendle et al., 2012), select a subset of items as negative samples to help distinguish positive samples, significantly reducing computational complexity and speeding up the training process. **Non-sampling** methods leverage all un-interacted items as negatives for comparison, ensuring more accurate ranking performance over all items. Typical objective functions include Softmax Loss and Weighted Squared Loss. Softmax Loss aims to maximize the probability of interacted items compared with uninteracted items (Sun et al., 2019; Rendle, 2021), whose optimization goal is consistent with ranking metrics such as Normalized Discounted Cumulative Gain (NDCG) (Bruch et al., 2019; Ravikumar et al., 2011), endowing it with significant advantages for item recommendation.

The Weighted Squared Loss, commonly referred to as the loss in Weighted Regularized Matrix Factorization (WRMF) (Hu et al., 2008; Pan et al., 2008), is widely employed in item recommendation. This loss assigns a fixed value of 1 to interacted items and smaller values, accompanied by lower weights, to un-interacted items. The research community has widely acknowledged such type of non-sampling method, which incorporates all un-interacted items into the training process, due to its superior ranking performance (Rendle, 2021; Chen et al., 2023; 2020; Yuan et al., 2021). By considering all instances, this approach contributes to better distinguishing between interacted and un-interacted items, leading to improved recommendation accuracy. The effectiveness of the weighted squared loss can be observed in the continued competitive performance of iALS (Rendle et al., 2021; 2022), which utilizes this loss, compared to more recent approaches such as SLIM (Ning & Karypis, 2011), EASE (Steck, 2019), and VAE (Liang et al., 2018). iALS consistently demonstrates strong ranking-oriented metrics, despite being developed over a decade ago. Moreover, the simplicity of the weighted squared loss function also allows for an applicable derivation (Bayer et al., 2017; Takács & Tikk, 2012), leading to more efficient convergence with the help of optimization methods leveraging closed-form solutions or higher-order gradients. Compared

[1] School of Artificial Intelligence and Data Science, University of Science and Technology of China, China [2] School of Computer Science and Technology, University of Science and Technology of China, China [3] School of Business and Management, Hong Kong University of Science and Technology, Hong Kong, China [4] State Key Laboratory of Cognitive Intelligence, China. Correspondence to: Defu Lian <liandefu@ustc.edu.cn>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

to commonly used stochastic gradient descent (SGD) methods (Amari, 1993), those methods often achieve desirable performance within a few iterations, demonstrating its effectiveness and efficiency for item recommendations.

However, existing research on weighted squared loss functions has primarily relied on conjectures about their superior performance rather than thorough investigations of the underlying mechanisms. Notably, there is an absence of comprehensive, in-depth, and theoretical analysis that explore the relationship between the objectives of squared loss functions and ranking metrics. In comparison, the softmax loss function serves as a valuable counterpart, which has been revealed as a consistent surrogate loss to DCG. However, the non-linear operations inherent in softmax raise challenges in optimizing ranking objectives across the entire item corpus, which hinders the application of closed-form solution-based or higher-order gradient-based optimization algorithms. Considerable bias and variance are also introduced during the estimation of the gradient with mini-batch updating, which harms both efficacy and efficiency.

In this work, we revisit the connections between the squared loss function and ranking-oriented metrics. We discover that a surrogate squared loss, referred to as the **R**anking-**G**eneralizable **squared** (**RG**^{2}) loss, closely aligns with the ranking objective DCG. To achieve this, we apply the Taylor expansion rule to the DCG-consistent softmax loss, resulting in a squared-form surrogate loss that serves as a good approximation and upper bound of the softmax loss. Through rigorous analysis, we establish the consistency of the RG^2 loss with DCG. We then apply the RG^2 loss to item recommendation tasks with MF models, where we derive an generalization upper bound and incorporate the Alternating Least Squares (ALS) optimization method, which optimizes the model with closed-form solutions. This integration leads to improved ranking performance compared to the existing iALS approach, as the RG^2 loss closely approximates softmax and aligns with the ranking metric.

The main contributions of this paper are presented in the following folds:

- We propose a novel squared loss function named RG^2 loss, which is an efficient approximation and upper bound of Softmax loss inspired by Taylor expansion. This novel approach unveils the connection between the squared loss and the ranking metric, offering new insights into recommender system optimization.
- We establish the theoretical properties of RG^2 loss, demonstrating its consistency with the ranking metric DCG, which highlights its reliability.
- We incorporate the RG^2 loss to MF-based CF tasks, and provide an upper bound regarding generalization. Besides, we make a rational application of ALS optimization method, which further improves the ranking performance compared to the traditional non-sampling method, pushing the boundaries of recommender system accuracy.
- Experimental results on three public real-world datasets indicate the effectiveness of our loss function, with particular advantages in efficiency improvements without sacrificing recommendation performance.

2. Preliminaries

2.1. Softmax Loss

The softmax loss function was first introduced to handle multi-class classification tasks and was soon generalized to various applications, including recommender systems. Consider a recommendation scenario predicting whether item $i \in \mathcal{I}$ is preferred in context $u \in \mathcal{U}$ (e.g., user, location, behavior history, etc.) with a positive preference set $\mathcal{D} = \{(u, i) \mid \text{item } i \text{ is preferred in context } u\}$. Logits $o_i^{(u)} = f_\theta(u, i)$ represent the output of model f denoting the score of the predicted preference for item i in context u. The softmax operation transforms them into:

$$p(o_i^{(u)}) = \frac{\exp(o_i^{(u)})}{\sum_{i \in \mathcal{I}} \exp(o_{i_k}^{(u)})} \tag{1}$$

which essentially converts $\boldsymbol{o}^{(u)} = [o_{i_1}^{(u)}, \cdots, o_{i_N}^{(u)}]_{i \in \mathcal{I}}$ into a probability distribution over $N = |\mathcal{I}|$ items for each input context u. Subsequently, the softmax loss, also commonly referred to as Categorical Cross-Entropy, is computed for each context-item pair (u, i) as follows:

$$\mathcal{L}(\boldsymbol{o}^{(u)}) = -\log\left(p(o_i^{(u)})\right) \tag{2}$$

The standard form of Softmax Loss can be expressed as:

$$\mathcal{L}_{\text{SM}} = -\frac{1}{|\mathcal{D}|} \sum_{(u,i) \in \mathcal{D}} \log\left(p(o_i^{(u)})\right) + \lambda\psi(\theta) \tag{3}$$

where $\lambda\psi(\theta)$ stands for some regularization term (usually l_2-norm) in practice.

In the subsequent sections, we will illustrate the alignment between the softmax loss and ranking-oriented metrics, such as DCG, which explains the superior performance of the softmax loss in item recommendation tasks. However, the non-linear operations within the softmax function, particularly the normalization of exponential calculations, pose challenges for efficient optimization using optimization methods using closed-form solutions or higher-order gradients, especially when dealing with large-scale item corpora.

2.2. Weighted Squared Loss

The weighted squared loss function is another typical non-sampling loss for handling implicit feedback in item recommendation. It assigns different weights to interacted and un-interacted items, indicating higher confidence in positive instances and lower in negative ones. A specific form of weighted squared loss can follow WRMF (Hu et al., 2008) as:

$$\mathcal{L}_{\text{WRMF}} = \sum_{u\in\mathcal{U}, i\in\mathcal{I}} w_{ui} \left(o_i^{(u)} - r_{ui}\right)^2 + \lambda\psi(\theta) \tag{4}$$

where

$$r_{ui} = \begin{cases} 1, & (u,i)\in\mathcal{D} \\ 0, & (u,i)\notin\mathcal{D} \end{cases}, \quad w_{ui} = \begin{cases} \alpha+1, & (u,i)\in\mathcal{D} \\ 1, & (u,i)\notin\mathcal{D} \end{cases}$$

and $\alpha \gg 1$ represents a higher weight corresponding to positive instances during training, which is adjusted empirically as a hyperparameter in different scenarios.

The weighted squared loss mentioned is frequently employed in dual encoders. iALS (Rendle et al., 2022) utilizes this loss to optimize matrix factorization models, while SAGram (Krichene et al., 2018) applies it to optimize non-linear encoders. The effectiveness of this loss stems from the utilization of ALS or second-order gradient descent optimization methods, which involve updating the user and item latent matrices using the closed-form solutions or the second-order Hessian matrix. The competitive performance demonstrated by iALS, outperforming contemporary approaches despite being proposed over a decade ago, further emphasizes the efficacy of the weighted squared loss. However, the optimization objectives associated with this loss currently lack a clear theoretical foundation to fully understand the underlying mechanism, as the squared form of the loss appears to operate independently of ranking metrics. Consequently, further investigation is necessary to uncover the underlying principles.

3. RG^2: Ranking-Generalizable Squared Loss

3.1. Theoretical Properties of Softmax Loss

The widespread application of softmax loss is not only attributed to its excellent performance in recommendation but also to its profound theoretical properties. The efficacy of softmax can be understood from the following two perspectives. Firstly, the softmax loss indirectly regulates the upper bound of important ranking metrics like NDCG,

Definition 3.1.

$$\text{DCG}(\boldsymbol{o}^{(u)}, \mathcal{D}) = \sum_{i\in\mathcal{I}} \frac{2^{r_{ui}} - 1}{\log(1+\pi^{(u)}(i))}, \tag{5}$$

$$\text{NDCG}(\boldsymbol{o}^{(u)}, \mathcal{D}) = \frac{\text{DCG}(\boldsymbol{o}^{(u)}, \mathcal{D})}{\max_{\boldsymbol{o}} \text{DCG}(\boldsymbol{o}, \mathcal{D})}, \tag{6}$$

where $\pi^{(u)}(i)$ stands for the rank of $o_i^{(u)}$ in $\boldsymbol{o}^{(u)} \in \mathbb{R}^N$, $r_{ui} \in \{0,1\}$ stands for whether $(u,i)\in\mathcal{D}$.

Proposition 3.2. *(Bruch et al., 2019) Softmax loss is a bound on mean Normalized Discounted Cumulative Gain in log-scale, i.e.*

$$-\log \overline{NDCG} \le -\frac{1}{|\mathcal{D}|}\sum_{u\in\mathcal{U}} \frac{1}{|\mathcal{I}_u|}\sum_{i\in\mathcal{I}_u} \log\left(p(o_i^{(u)})\right) \tag{7}$$

where $\mathcal{I}_u = \{i \mid (u,i)\in\mathcal{D}\}$.

This characteristic makes it particularly effective in handling ranking-oriented recommendations. Secondly, from a more fundamental viewpoint, the softmax loss is a surrogate loss that is top-k calibrated and DCG-consistent.

Definition 3.3. (Top-k preserving) Given vectors $\boldsymbol{s}, \boldsymbol{\eta} \in \mathbb{R}^N$, $\boldsymbol{s}$ is top-k preserving with respect to $\boldsymbol{\eta}$, denoted as $P_k(\boldsymbol{s}, \boldsymbol{\eta})$, if for $\forall n \in [N]$,

$$\begin{aligned} \eta_n > \eta_{[k+1]} &\implies s_n > s_{[k+1]} \\ \eta_n < \eta_{[k]} &\implies s_n < s_{[k]} \end{aligned} \tag{8}$$

Here $s_{[j]}$ stands for the j-th largest value in $\boldsymbol{s}$.

Definition 3.4. (Inverse Top-k preserving function) Given $A, B \subseteq \mathbb{R}^N$, a function $f : A \to B$ is inverse top-k preserving if $\forall \boldsymbol{s} \in A, P_k(\boldsymbol{s}, f(\boldsymbol{s}))$

Definition 3.5. (Order preserving) Given $\boldsymbol{s}, \boldsymbol{\eta} \in \mathbb{R}^N$, $\boldsymbol{s}$ is order preserving with respect to $\boldsymbol{\eta}$, denoted as $\boldsymbol{s} \hookrightarrow \boldsymbol{\eta}$ if

$$\forall i,j, \eta_i > \eta_j \implies s_i > s_j \tag{9}$$

A function $g(\cdot)$ is order preserving iff $g(\boldsymbol{s}) \hookrightarrow \boldsymbol{s}$.

Definition 3.6. (Top-k calibration): Let $\Delta_N = \{\boldsymbol{\eta} \in \mathbb{R}^N | \sum_{i=1}^N \eta_i = 1\}$. A loss function $\psi : \mathbb{R}^N \times \mathcal{Y} \to \mathbb{R}$ is top-k calibrated if for all $\boldsymbol{\eta} \in \Delta_N$,

$$\inf_{\boldsymbol{s}\in\mathbb{R}^N \cap \neg P_k(\boldsymbol{s},\boldsymbol{\eta})} L_\psi(\boldsymbol{s},\boldsymbol{\eta}) > \inf_{\boldsymbol{s}\in\mathbb{R}^N} L_\psi(\boldsymbol{s},\boldsymbol{\eta}) = L_\psi(\boldsymbol{\eta}) \tag{10}$$

Definition 3.7. (Bregman Divergence) Given $\boldsymbol{s}, \boldsymbol{t} \in \mathbb{R}^N$ and a convex, differentiable function $\phi : \mathbb{R}^N \to R$, the Bregman divergence D_ϕ is defined by

$$D_\phi(\boldsymbol{s}, \boldsymbol{t}) = \phi(\boldsymbol{t}) - \phi(\boldsymbol{s}) - \langle \nabla\phi(\boldsymbol{s}), \boldsymbol{t} - \boldsymbol{s}\rangle. \tag{11}$$

Proposition 3.8. *(Yang & Koyejo, 2020) Rewrite the softmax loss as a Bregman divergence, i.e.,*

$$-\log(p(o_i^{(u)})) = D_\phi(g(\boldsymbol{o}^{(u)}), \boldsymbol{e}_i) \tag{12}$$

where $\phi(\boldsymbol{o}) = \sum_j o_j \log o_j, g(\boldsymbol{o})_i = p(o_i)$. Since ϕ is strictly convex and differentiable, g is inverse top-k preserving, then the softmax loss is top-k calibrated.

Proposition 3.9. *Since ϕ is strictly convex and differentiable, g is inverse order-preserving, and the softmax loss is DCG-consistent.*

Remark 3.10. The importance of normalization in surrogate losses for NDCG-consistency was emphasized in (Ravikumar et al., 2011). However, this term is not critical for item recommendation since implicit feedback can be modeled as single-click behaviors, which transform all normalization terms to 1 (Bruch, 2021). To prevent ambiguity, we uniformly discuss DCG-consistency without normalization.

This reveals the softmax loss's theoretical soundness in optimizing these ranking-related objectives, offering an ideal alternative for recommendation models.

3.2. Revealing Connections by Taylor Expansion

Reflecting on the loss form in Eq.(2), we analyse with a direct Taylor expansion at $\mathbf{0}$ with respect to any given $\boldsymbol{o}^{(u)}$:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{o}^{(u)}) &\overset{Taylor}{=} \mathcal{L}(\mathbf{0}) + \nabla\mathcal{L}(\mathbf{0})^\top\boldsymbol{o}^{(u)} + \frac{1}{2}\boldsymbol{o}^{(u)\top}\nabla^2\mathcal{L}(\mathbf{0})\boldsymbol{o}^{(u)} \\
&= \log N - o_i^{(u)} + \frac{1}{N}\mathbf{1}_N^\top\boldsymbol{o}^{(u)} + \frac{1}{2N}\boldsymbol{o}^{(u)\top}(\boldsymbol{I} - \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^\top)\boldsymbol{o}^{(u)} \\
&\leq \log N - o_i^{(u)} + \frac{1}{N}\mathbf{1}_N^\top\boldsymbol{o}^{(u)} + \frac{1}{2N}\boldsymbol{o}^{(u)\top}\boldsymbol{o}^{(u)}
\end{aligned}$$

The final inequality comes from the semi-positive definition of matrix $\mathbf{1}_N\mathbf{1}_N^\top$. Here, we omit the remainder for the sake of simplicity, while the softmax loss is still upper bounded by the approximation under a rational assumption. The detailed process of expansion and upper bound proofs are referred to Appendix B.2. To further simplify the above form, we formulate

$$\begin{aligned}
\mathcal{L}(\boldsymbol{o}^{(u)}) &\leq \log N - o_i^{(u)} + \frac{1}{2N}\|\boldsymbol{o}^{(u)} + \mathbf{1}_N\|^2 - \frac{1}{2N}\mathbf{1}_N^\top\mathbf{1}_N \\
&= \left(\log N - \frac{1}{2}\right) - o_i^{(u)} + \frac{1}{2N}\|\boldsymbol{o}^{(u)} + \mathbf{1}_N\|^2
\end{aligned}$$

Through the above expansion analysis with respect to the softmax loss, we surprisingly find the connection between the squared-form formula $\|\boldsymbol{o}^{(u)} + \mathbf{1}_N\|^2$ and the softmax loss. At the same time, we observe and define the **RG**^{2} loss[1] function as:

$$\mathcal{L}_{\text{eps}} = -\sum_{(u,i)\in\mathcal{D}} o_i^{(u)} - \frac{1}{2N}\|\boldsymbol{o}^{(u)} + \mathbf{1}_N\|^2 \tag{13}$$

Intuitively, the newly designed loss function is composed of two parts. The first part tends to maximize the outputs corresponding to all interacted items, while the second part aims to converge the scores for all items toward a fixed value of -1. An interesting observation is that such loss can be presented in a squared form by some simple transformations. The detailed process is in Appendix B.3.

$$\mathcal{L}_{\mathrm{RG}^2} = -\frac{1}{|\mathcal{D}|}\sum_{u,i} -\frac{|\mathcal{I}_u|}{2N}\left(o_i^{(u)} + 1 - r_{ui}\frac{N}{|\mathcal{I}_u|}\right)^2 + \lambda\psi(\theta) \tag{14}$$

where $r_{ui} = 1$ if $(u,i) \in \mathcal{D}$ and 0 otherwise. Similar to WRMF, the proposed squared loss tends to converge the scores for all positive instances toward a large value while all negative ones toward -1.

Remark 3.11. Due to the flexibility of the **RG**^{2} loss, we use the subscript 'eps' to denote the result obtained from Taylor expansions and 'RG^2' for its squared-form transformation.

3.3. Consistency

Having derived a novel form of the loss function, our next step is to establish its consistency with the ranking metric.

Assuming that the interaction set $\mathcal{D}$ i.i.d drawn from sample space $\mathcal{X} = \{(u,i)\}_{u\in\mathcal{U},i\in\mathcal{I}}$ with some distribution, the expected DCG loss is given as:

$$\mathcal{L}_{\text{DCG}}(\boldsymbol{o}^{(u)}) = \mathbb{E}_{\mathcal{D}\sim\mathcal{X}}\left[-\operatorname{DCG}(\boldsymbol{o}^{(u)}, \mathcal{D})\right] \tag{15}$$

Consider a potential surrogate ϕ, whose expected loss can then be given as:

$$\Phi(\boldsymbol{o}^{(u)}) = \mathbb{E}_{\mathcal{D}}[\phi(\boldsymbol{o}^{(u)}, \mathcal{D})] \tag{16}$$

Hence, the definition of consistency is provided as follows:

Definition 3.12. A surrogate ϕ is consistent with DCG if for any distribution on sample space $\mathcal{X}$ and for any sequence $\{{\boldsymbol{o}^{(u)}}_n\}_{n=1}^\infty$, s.t.

$$\Phi(\boldsymbol{o}_n^{(u)}) \to \Phi^* \tag{17}$$

we have

$$\mathcal{L}_{\text{DCG}}(\boldsymbol{o}_n^{(u)}) \to \mathcal{L}_{\text{DCG}}^* \tag{18}$$

where

$$\begin{aligned}
\Phi^* &= \min_{\boldsymbol{o}^{(u)}} \Phi(\boldsymbol{o}^{(u)}) \\
\mathcal{L}_{\text{DCG}}^* &= \min_{\boldsymbol{o}^{(u)}} \mathcal{L}_{\text{DCG}}(\boldsymbol{o}^{(u)})
\end{aligned} \tag{19}$$

whose minimum are assumed to be achievable.

Following the above definition, we are to prove the consistency of $\mathcal{L}_{\mathrm{RG}^2}$ with the targeted optimization objective DCG. The detailed proof is in Appendix.B.4.

Theorem 3.13. *For any given $\boldsymbol{o} \in \mathbb{R}^N$, consider the surrogate loss*

$$\Phi(\boldsymbol{o}) = \mathbb{E}_{\mathcal{D}}[\mathcal{L}_{RG^2}(\boldsymbol{o})] \tag{20}$$

then the following satisfies with some constant C,

$$\mathcal{L}_{DCG}(\boldsymbol{o}) - \mathcal{L}_{DCG}^* \leq C(\Phi(\boldsymbol{o}) - \Phi^*)^{\frac{1}{2}} \tag{21}$$

hence proving the DCG-consistency of surrogate loss $\mathcal{L}_{\mathrm{RG}^2}$.

In fact, the consistency proof about the surrogate loss $\mathcal{L}_{\mathrm{RG}^2}$ can also be understood in terms of order-preserving.

Proposition 3.14. *Consider $g(\boldsymbol{r}) = \frac{N}{|\mathcal{I}_u|}\boldsymbol{r} - 1$ which is linear therefore invertible order-preserving. $\mathcal{L}_{RG^2}$ reaches its minimum when ${\boldsymbol{o}^{(u)}}^* = g(\mathbb{E}[\boldsymbol{r}])$, which is intuitive for a squared loss form. Thus revealing $\mathcal{L}_{RG^2}$ is DCG-consistent.*

[1] Note that such form omits constant terms, hence leading to a high probability of loss value less than zero.

The above discussion demonstrates that the proposed squared loss also has DCG-consistency compared to the softmax loss, providing an explanation for the squared-form loss to have good performance in ranking metrics.

3.4. Instantiation with Matrix Factorization

In this section, we utilize the RG^2 loss in MF model with analysis on a generalization upper bound and optimize it using the ALS method. The details for graph-based models are provided in Appendix. C.2.

Matrix factorization (MF) is a typical and flexible collaborative filtering method that has been comprehensively developed in various recommendation scenarios (Koren, 2008; Hu et al., 2008; Koren et al., 2009; Rendle et al., 2012; Lian et al., 2014). It decomposes the large interaction matrix $R \in \{0,1\}^{M\times N}$ where $r_{ui} = 1$ if $(u,i) \in \mathcal{D}$, into two lower-dimensional matrices. Consider a MF-based recommendation scenario with M users and N items. Let $P \in \mathbb{R}^{M\times K}, Q \in \mathbb{R}^{N\times K}$ stand for the representation matrices of users and items to be learned, where K stands for the dimension of latent embeddings. The predicted matrix is computed as $O = P \cdot Q^\top$, with an apparent property rank$(O) \le K$. The flexibility of MF lies in the selection of different objective functions, which significantly impact the model's performance and recommendation quality. By utilizing our proposed RG^2 loss, rewriting Eq.(14) into MF-based form with Frobenius-norm regularizers, we have

$$\mathcal{L}_{\text{RG}^2} = \sum_{u,i=1}^{M,N} W_{ui}(S_{ui} - P_{u\cdot} \cdot Q_{\cdot i}^\top)^2 + \lambda(\|P\|_F^2 + \|Q\|_F^2) \tag{22}$$

where

$$S_{ui} = r_{ui}\frac{N}{|\mathcal{I}_u|} - 1; W_{ui} = \frac{|\mathcal{I}_u|}{2N|\mathcal{D}|}. \tag{23}$$

Compared to original WRMF loss in Eq.(4), our proposed $\mathcal{L}_{\text{RG}^2}$ highlights the effect of positive pairs by increasing the predicted values instead of setting different weights, and avoids the manual adjustment of hyperparameter α.

Before we analyse the efficient ALS method in detail, we first provide a generalization upper bound for RG^2.

3.4.1. Generalization Analysis

To analyse RG^2's generalization performance in MF, we give out a definition of the hypothesis space. Suppose the interaction set $\mathcal{D} = \{(u,i) \in \mathcal{X}\}$ is i.i.d drawn from some unknown distribution $\mathcal{P}(\mathcal{X})$. Denote hypothesis function $h(u,i) = o_i^{(u)} = P_{u\cdot} \cdot Q_{\cdot i}^\top$. The hypothesis space can be expressed as:

$$\mathcal{H} := \left\{ h_O : (u,i) \to h_O(u,i) = o_i^{(u)} \mid \text{rank}(O) \le K \right\} \tag{24}$$

Correspondingly, the empirical error and generalization error in our settings are formed as:

$$\hat{\mathcal{R}}_{\mathcal{D}}(h) = \frac{1}{|\mathcal{D}|}\sum_{u,i=1}^{M,N} \frac{|\mathcal{I}_u|}{2N}\left(o_i^{(u)} + 1 - r_{ui}\frac{N}{|\mathcal{I}_u|}\right)^2 \tag{25}$$

$$\mathcal{R}(h) = \mathbb{E}_{\mathcal{D}\sim\mathcal{P}_{\mathcal{X}}^m}\left[\frac{|\mathcal{I}_u|}{2N}\left(o_i^{(u)} + 1 - r_{ui}\frac{N}{|\mathcal{I}_u|}\right)^2\right] \tag{26}$$

Remark 3.15. This is a classical form of squared loss. By assuming an upper bound B of the loss function, the error is restricted by a $2B$-Lipschitz continuity.

To obtain the upper bound of the generalization error, consider the pseudo-dimension of the hypothesis space $\mathcal{H}$ with the following estimates:

Proposition 3.16. *(Srebro et al., 2004) The pseudo-dimension of the low-rank matrix hypothesis space $\mathcal{H}$ is at most $K(M+N)\log\frac{16eM}{K}$, where M, N, K represent the number of users, items, and embedded dimensions.*

This estimate is important for approximating the complexity of the hypothesis space due to the following inequality with respect to the L_1-covering number,

Lemma 3.17. *(Anthony et al., 1999) Let $\mathcal{H}$ be a nonempty set of real functions mapping from a domain $\mathcal{X}$ into $[0,1]$ and suppose that $\mathcal{H}$ has finite pseudo-dimension d. Then*

$$\mathcal{N}_1(\varepsilon, \mathcal{H}, m) < e(d+1)\left(\frac{2e}{\varepsilon}\right)^d \tag{27}$$

Remark 3.18. Even though the hypothesis space we define may not satisfy mapping all inputs into $[0,1]$, the optimization behavior of the loss function ensures that the hypothesis space is embedded into a bounded interval, which does not harm generalizability.

With the Lipschitz condition of the loss function, L_1-covering number of hypothesis space $\mathcal{H}$ establishes a direct relation to the generalized error bound, from which obtaining the following theorem.

Theorem 3.19. *(Anthony et al., 1999) Suppose $\mathcal{H}$ be a nonempty set of real functions mapping from a domain $\mathcal{X}$ into $[0,1]$. Let P be any probability distribution on $\mathcal{X}\times\mathbb{R}$, $\forall\varepsilon > 0$ with any positive integer m. Then for any loss function $\ell(h(x),y)$ with $|\ell(h(x),y)| \le B$ with L-Lipschitz,*

$$\begin{aligned} P^m &\left(\exists h \in \mathcal{H} \quad s.t. \quad |\mathcal{R}(h) - \hat{\mathcal{R}}_{\mathcal{D}}(h)| \ge \varepsilon\right) \\ &\le 4\mathcal{N}_1\left(\frac{\varepsilon}{8L}, \mathcal{H}, 2m\right)\exp\left(\frac{-m\varepsilon^2}{32B^4}\right) \end{aligned} \tag{28}$$

We have listed all the conditions required to prove an upper bound on generalization error. By combining the above conclusions, we obtain the following theorem.

Theorem 3.20. *For all conditions in our settings, with probability at least $1-\delta$, we have*

$$\mathcal{R}(h) \leq \hat{\mathcal{R}}_{\mathcal{D}}(h) + \sqrt[d+2]{\frac{4(d+1)B^3(32eB)^{d+1}}{|\mathcal{D}|\delta}} \tag{29}$$

where $d = K(M+N)\log\frac{16eM}{K}$.

Proof. Starting from Theorem 3.19, we have

$$\begin{aligned} & P^m\left(|\mathcal{R}(h) - \hat{\mathcal{R}}_{\mathcal{D}}(h)| \geq \varepsilon\right) \\ \leq & 4\mathcal{N}_1\left(\frac{\varepsilon}{16B}, \mathcal{H}, 2m\right)\exp\left(-\frac{|\mathcal{D}|\varepsilon^2}{32B^4}\right) \\ \leq & 4e(d+1)(\frac{32eB}{\varepsilon})^d \exp\left(-\frac{|\mathcal{D}|\varepsilon^2}{32B^4}\right) \\ \leq & 4e(d+1)(\frac{32eB}{\varepsilon})^d \frac{32B^4}{|\mathcal{D}|\varepsilon^2} \end{aligned} \tag{30}$$

The last inequality comes from $\exp(-x) \leq \frac{1}{x}$. By setting

$$\delta = 4e(d+1)(\frac{32eB}{\varepsilon})^d \frac{32B^4}{|\mathcal{D}|\varepsilon^2}, \tag{31}$$

we obtain the final form of ε in Eq.(29). □

We further combine this upper bound with Theorem 3.13 to obtain a lower bound for ranking metric DCG.

Proposition 3.21.

$$\begin{aligned} & \mathbb{E}_{\mathcal{D}}\left[-DCG(\boldsymbol{o}^{(u)}, \mathcal{D})\right] \leq C(\mathcal{R}(h) - \Phi^*)^{\frac{1}{2}} + \mathcal{L}^*_{DCG} \\ \leq & C\left(\hat{\mathcal{R}}_{\mathcal{D}}(h) + \sqrt[d+2]{\frac{4(d+1)B^3(32eB)^{d+1}}{|\mathcal{D}|\delta}} - \Phi^*\right) + \mathcal{L}^*_{DCG} \end{aligned}$$

Note that $\mathcal{L}^*_{DCG} = \mathbb{E}_{\mathcal{D}}\left[IDCG(\boldsymbol{o}^{(u)}, \mathcal{D})\right]$. Then with elementary transformations, we have

$$\begin{aligned} & \mathbb{E}_{\mathcal{D}}\left[DCG(\boldsymbol{o}^{(u)}, \mathcal{D})\right] \geq \mathbb{E}_{\mathcal{D}}\left[IDCG(\boldsymbol{o}^{(u)}, \mathcal{D})\right] \\ & - C\left(\hat{\mathcal{R}}_{\mathcal{D}}(h) - \inf_h \mathcal{R}(h) + \mathcal{T}_{\mathcal{D}}(d, B, e, \delta)\right) \end{aligned} \tag{32}$$

where $\mathcal{T}_{\mathcal{D}}(d, B, e, \delta) = \sqrt[d+2]{\frac{4(d+1)B^3(32eB)^{d+1}}{|\mathcal{D}|\delta}}$.

It is worth noting that the generalization lower bound of DCG derived from this proposition is constrained by the pseudo-dimension d of the hypothesis space and the number of interactions $|\mathcal{D}|$ with $O((d+1)^{\frac{1}{d+2}} \cdot C^{1-\frac{1}{d+2}})$ and $O(|\mathcal{D}|^{-\frac{1}{d+2}})$, respectively.

3.4.2. Optimization with ALS

Given that the proposed RG^2 loss involves linear operations, it is well-suited for utilization in the ALS optimization algorithm. This algorithm updates the model parameters based on the closed-form solutions. Regarding Eq.(22), by fixing one of the matrix P or Q, the derivative with respect to the other term is calculated as:

Algorithm 1 Weighted Alternating Least Squares.

```
1:  Input: Data Matrix $R \in \{0,1\}^{M,N}$
2:  Output: $P$ and $Q$
3:  Initialize $P$ and $Q$
4:  $M, N \leftarrow R.shape[0], R.shape[1]$
5:  $\mathbf{u} \leftarrow [\sum R_{i\cdot}]_{1 \leq i \leq M}$
6:  $W \leftarrow \frac{1}{2N}\text{Diag}(\mathbf{u})R$
7:  $S \leftarrow \frac{1}{N}\text{Diag}(\mathbf{u})^{-1}R - 1$
8:  repeat
9:      Update $P_{i,\cdot}, \forall\ i$ with Eq.(33)
10:     Update $Q_{i,\cdot}, \forall\ i$ with Eq.(34)
11: until convergence
```

$$\begin{aligned} \frac{\partial \mathcal{L}^*}{\partial P_{uk}} &= 2\sum_j W_{uj}(P_{u\cdot}Q_{i\cdot}^\top - S_{uj})Q_{ik} + 2\lambda(\sum_j W_{uj})P_{uk} \\ \frac{\partial \mathcal{L}^*}{\partial P_{u\cdot}} &= \left[\frac{\partial \mathcal{L}^*}{\partial P_{u1}}, \cdots, \frac{\partial \mathcal{L}^*}{\partial P_{uM}}\right] \\ &= 2P_{u\cdot}\left(Q^\top \widetilde{W}_{i\cdot}Q + \lambda\left(\sum_j W_{uj}\right)I\right) - 2S_{u\cdot}\widetilde{W}_{u\cdot}Q \end{aligned}$$

where $\widetilde{W}_{u\cdot}, \widetilde{W}_{\cdot i}$ are diagonal matrices with the elements of $W_{u\cdot}, W_{\cdot i}$ on the diagonal.

Therefore, the objective reaches its minimum at a closed-form solution by setting derivatives to zero as below:

$$P_{u\cdot} = S_{u\cdot}\widetilde{W}_{u\cdot}Q\left(Q^\top \widetilde{W}_{u\cdot}Q + \lambda\left(\sum_i W_{ui}\right)I\right)^{-1} \tag{33}$$

$$Q_{i\cdot} = S_{\cdot i}^\top \widetilde{W}_{\cdot i}P\left(P^\top \widetilde{W}_{\cdot j}P + \lambda\left(\sum_u W_{ui}\right)I\right)^{-1} \tag{34}$$

The equations above serve as the update rule for model parameters as illustrated in Algorithm.1. First, the matrices P, Q are initialized uniformly in line 3, and the matrices W, S are calculated according to Eq.(23) in lines 4-7. After initialization, we update the matrix P referring Eq.(33) and then Q referring Eq.(34) alternatively. The training phase is repeated until convergence. We also provide the optimization based on the Gauss-Newton method in Appendix D.

3.4.3. Complexity Analysis

We give out a brief analysis on the time complexity of Alg.1. Recall that P, Q are matrices shaped $M \times K$ and $N \times K$, with $K \ll \min\{M, N\}$. Note all users share the same constant matrices S and W and thus they can be pre-computed. The update procedure of $P_{u\cdot}$ comprises the following matrix operations. The complexity of multiplying $Q^\top \widetilde{W}_{u\cdot}$ is

$O(NK)$ since $\widetilde{W}_{u\cdot}$ is diagonal. The subsequent multiplication with Q yields a complexity of $O(NK^2)$, with $O(K^3)$ for the matrix inversion. S is divided into a sparse interaction matrix and an all-one matrix, leading to a complexity of $O(|\mathcal{D}|K+MK+NK)$ calculating $S_{u\cdot}\widetilde{W}_{u\cdot}Q$, with another $O(MK^2)$ on multiplying $S_{u\cdot}\widetilde{W}_{u\cdot}Q$ with the inversion part. The overall complexity yields $O(|\mathcal{D}|K^2+(M+N)K^3)$ for updating the user and item latent matrix. After T iterations, the total complexity achieves $O(T(|\mathcal{D}|K^2+(M+N)K^3))$.

4. Experimental Results

4.1. Experimental Settings

4.1.1. Dataset and Evaluation

Dataset. We evaluate our method on three public datasets: MovieLens-10M, Amazon-electronics, and Steam Games collected from different real-world online platforms, involving domains of movies, shopping, and games, which are abbreviated as **MovieLens**, **Electronics** and **Steam**. **MovieLens** comprises approximately 10 million movie ratings ranging from 0.5 to 5, in increments of 0.5. **Electronics** collects the customer's reviews on electronics products on the Amazon platform, where each review consists of a rating ranging from 0 to 5 and the reviews about the product. **Steam** is a dataset crawled from the large online video game distribution platform *Steam* (Kang & McAuley, 2018), comprising the player's reviews plus rich information such as playing hours. As for **MovieLens** and **Electronics**, we treat items rated below 3 as negatives and the remains as positives. For **Steam**, since there is no explicit rating, we treat all samples as positives. We employ the widely used k-core filtering strategy to filter out the users and items with interactions less than 5. The detailed statistics of those datasets after filtering are illustrated in Table 1.

Table 1: Statistics of datasets.

Dataset	#User	#Item	#Interact	Sparsity
MovieLens	69,815	9,888	8,240,192	98.81%
Electronics	192,403	63,001	1,689,188	99.99%
Steam	281,204	11,961	3,484,497	99.90%

Data Split. We partition all datasets into training, validation, and test sets with a split ratio of $\{0.8, 0.1, 0.1\}$ for each user, respectively. The validation set is utilized to assess the model's performance, while the metrics derived from the test set serve as the foundation for comparative analysis.

Metrics. We adopt two widely used ranking evaluation metrics, Mean Reciprocal Rank with cutoff set as K(**MRR@K**) and **NDCG@K**, to measure the ranking performance of different methods, which aligns with the theoretical understandings in previous discussions. The definition of **NDCG@K** is given in Def.3.1 with a top-K cutoff, while

$$\text{MRR@K} = \frac{1}{|\mathcal{Q}|}\sum_{i=1}^{|\mathcal{Q}|}\frac{1}{\text{rank}_i}$$

where $\mathcal{Q}$ represents all queries in the test set, rank_i stands for the rank of the first relevant item in top-K of the recommended list. We set K = 10 on all datasets and use **NDCG@10** as the early stop indicator to demonstrate the broad validity of our loss on ranking metrics.

4.1.2. Baselines

To validate the effectiveness of the proposed novel loss function, we incorporate various types of loss functions as baseline methods, including sampling-based methods (**BPR**, **BCE**, **S-Softmax**, **UIB**,**SML**), variants of log Softmax loss (**Sparsemax**), and the competitive method **WRMF** optimized with the ALS method.

- **BPR** (Rendle et al., 2012): Bayesian Personalized Ranking Loss is designed for personalized ranking in implicit feedback, which maximizes the score difference between interacted and non-interacted items.

$$\mathcal{L}_{\text{BPR}} = -\frac{1}{|\mathcal{D}|}\sum_{u\in\mathcal{U}}\sum_{i\in\mathcal{I}_u, j\in\mathcal{I}_u^-}\log\left(\sigma\left(o_i^{(u)} - o_j^{(u)}\right)\right)$$

- **BCE** (He et al., 2017): Binary Cross-Entropy Loss regards the observed items as positives and unobserved items as negatives.

$$\mathcal{L}_{\text{BCE}} = -\sum_{(u,i)\in\mathcal{D}}\log\sigma\left(o_i^{(u)}\right) - \sum_{(u,j)\in\mathcal{D}^-}\log\left(1-\sigma\left(o_j^{(u)}\right)\right)$$

- **Sampled Softmax** (Covington et al., 2016; Yi et al., 2019): Sampled Softmax is an efficient approximation depending on importance sampling, with a sampling set $\mathcal{N}$ according to the sampling probability $q_i^{(u)}$.

$$\mathcal{L}_{\text{S-Softmax}} = -\frac{1}{|\mathcal{D}|}\sum_{(u,i)\in\mathcal{D}}\frac{e^{o_i^{(u)}}/q_i^{(u)}}{e^{o_i^{(u)}}/q_i^{(u)} + \sum_{j\in\mathcal{N}} e^{o_j^{(u)}}/q_j^{(u)}}$$

- **Sparsemax** (Martins & Astudillo, 2016): Sparsemax is a variant of Softmax that returns sparse posterior distributions by assigning zero probability to some classes.

$$\mathcal{L}_{\text{Sparse}} = -\sum_{(u,i)\in\mathcal{D}}\left(-o_i^{(u)} + \frac{1}{2}\sum_{j\in S^{(u)}}^{N}\left({o_j^{(u)}}^2 - {\tau^{(u)}}^2\right) + \frac{1}{2}\right)$$

where $\tau^{(u)} = \left(\left(\sum_{j\in S^{(u)}} o_j^{(u)}\right) - 1\right)/|S^{(u)})|$, $S^{(u)} = \{j \in \mathcal{I} \mid \text{sparsemax}_j\left(\boldsymbol{o}^{(u)}\right) > 0\}$ and $\text{sparsemax}\left(\boldsymbol{o}^{(u)}\right) = \underset{\boldsymbol{p}\in\Delta^{K-1}}{\arg\min}\|\boldsymbol{p} - \boldsymbol{o}^{(u)}\|^2$.

Table 2: Comparisons of recommendation performance. S-Softmax and Softmax are the abbreviations of Sampled Softmax and Softmax loss functions. **Bold** and underline numbers represent the best and the second-best results respectively.

Dataset	Metric	BPR	BCE	S-Softmax	Sparsemax	UIB	SML	WRMF	Softmax	$\mathbf{RG}^2$
MovieLens	MRR@10	0.3476	0.3602	0.3785	0.3177	0.3836	0.3386	0.4475	0.4487	**0.4723**
	NDCG@10	0.2116	0.2228	0.2378	0.1738	0.2426	0.2045	0.2797	0.2849	**0.2963**
Electronics	MRR@10	0.0124	0.0098	0.0119	0.0078	0.0101	0.0074	0.0146	0.0172	**0.0173**
	NDCG@10	0.0154	0.0126	0.0152	0.0100	0.0130	0.0098	0.0178	0.0212	**0.0212**
Steam	MRR@10	0.0434	0.0435	0.0466	0.0317	0.0410	0.0307	0.0464	**0.0493**	0.0492
	NDCG@10	0.0521	0.0532	0.0549	0.0359	0.0506	0.0353	0.0544	**0.0579**	0.0575

- **UIB** (Zhuo et al., 2022): This loss introduces a learnable auxiliary score b_u for each user to represent the User Interest Boundary (UIB) and penalizes samples that exceed the decision boundary. With $\phi(\cdot)$ being MarginLoss, the loss form is given as follows:

$$\mathcal{L}_{\text{UIB}} = \sum_{(u,i)\in\mathcal{D}} \phi(b_u - o_i^{(u)}) + \sum_{(u,i)\in\mathcal{D}} \phi(o_i^{(u)} - b_u)$$

- **SML** (Li et al., 2020): SML improves the limitation of CML by introducing dynamic margins. Let $d(u,i)$ denote the distance function between embeddings of user u and item i, the loss form is given as follows:

$$\mathcal{L}_{\text{SML}} = \sum_{(u,i)\in\mathcal{D}} \sum_{(u,i^-)\notin\mathcal{D}} ([d(u,i) - d(u,i^-) + m_u]_+$$
$$+\lambda[d(u,i) - d(i,i^-) + n_i]_+) - \gamma(\frac{1}{|\mathcal{U}|}\sum_u m_u + \frac{1}{|\mathcal{I}|}\sum_i n_i),$$
$$m_u \in (0,l], n_i \in (0,l], \forall u, i \in \mathcal{U}, \mathcal{I}$$

- **Softmax**: Log softmax loss maximizes the probability of the observed items normalized over all items by Eq.(3).
- **WRMF** (Hu et al., 2008; Rendle et al., 2021): WRMF uses ALS methods to optimize the loss value in Eq.(4).
- $\mathbf{RG}^2$: Our Ranking-Generalizable Squared loss also utilizes ALS methods to optimize the loss value in Eq.(14).

4.1.3. Implementation Details

We conduct all experiments on a highly-modularized recommendation library RecStudio (Lian et al., 2023). The loss functions and baselines are implemented on a linear model[2], i.e., a matrix factorization model, with the embedding size set to 64. For **BPR** and **BCE** loss function, we draw 10, 20, and 10 negative samples uniformly for each positive in the training procedure for MovieLens, Electronics, and Steam, respectively. For **Sampled Softmax**, the proposal distribution is set as uniform sampling, and the numbers of negative samples are set as 100, 200, and 100, respectively. As for **UIB** and **SML**, the respective negative sampling numbers are 10, 10, 1 and 1, 100, 50. We use a single Nvidia RTX-3090 with 24GB memory in training for all methods. Except for **WRMF**, all the baselines are optimized with the Adam(Kingma & Ba, 2014) optimizer, which is a variant of SGD. As for the SGD optimization, the batch size is set to 2048. Learning rate and weight decay are tuned in $\{0.1,0.01,0.001\}$ and $\{0, 10^{-6}, 10^{-5}, 10^{-4}\}$, respectively. As for **WRMF**, the hypermeters λ and α are tuned in $\{0,$ $0.1, 0.01, 0.001\}$ and $\{0.5, 1, 2, 4, 8\}$, respectively. There is only one tunable hypermeter λ in our loss function, which is tuned in $\{0, 0.1, 0.01, 0.001\}$. The code is available at `https://github.com/yuanhao53/RG2`.

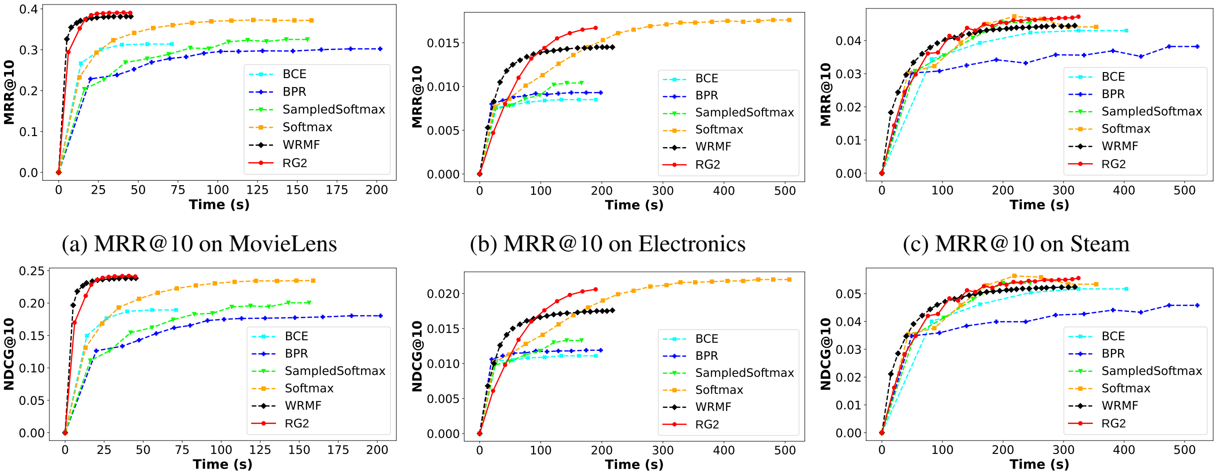
4.2. Overall Recommendation Performance

To validate the effectiveness of our loss function in recommendation performance, we compare our method with baselines in terms of the ranking metrics in the experiments. The results are presented in Table 2 and results with confidence interval are reported in Appendix. C.1. From the results, we can summarize the following findings.

Non-sampling methods (**WRMF**, **Softmax** and $\mathbf{RG}^2$) show significant enhancement in performance compared with sampling ones (**BPR, BCE, S-Softmax, UIB** and **SML**). Despite optimized by SGD, **Softmax** loss demonstrates significant superiority over all (SGD-based) sampling baselines, which obtains 20.49% and 20.19% average relative improvements compared with the best sampling baseline in terms of MRR@K and NDCG@K on all datasets, which aligns with its theoretical properties in consistency with ranking metrics. Furthermore, due to the randomness introduced by sampling methods, bias has emerged as a critical factor constraining the performance of all sampling-based methods. **Sparsemax**, as a sparse variant of Softmax, sets most probabilities in the output distribution as zero, resulting in the underfitting of representations of inactive users and cold items, thereby suffering severe performance degradation.

For ALS-based non-sampling methods, **WRMF** employs a deterministic optimization process, which avoids the variance introduced by SGD. It achieves a 7.53% average rel-

[2] Experimental results on graph-based model (LightGCN) are provided in Appendix.C.2.



(a) MRR@10 on MovieLens (b) MRR@10 on Electronics (c) MRR@10 on Steam

(d) NDCG@10 on MovieLens (e) NDCG@10 on Electronics (f) NDCG@10 on Steam

Figure 1: Comparisons of convergence speed on all datasets.

ative degradation on all metrics with **Softmax** but outperforms other baselines.

Our proposed **RG**^{2} achieves a balance between efficiency and effectiveness. Firstly, our method showcases a comparable ranking performance with **Softmax**, specifically 4.63% and 0.31% average relative improvements yet a 0.70% relative decrease on MovieLens, Electronics, and Steam in terms of all metrics, respectively, but a faster convergence speed. This substantiates the effectiveness of our approximation and the advantage of ALS optimization, which preserves a deterministic updating direction through a closed-form solution compared with the first-order gradient-based SGD optimization. Secondly, our **RG**^{2} achieves a comparable speed but better performance with **WRMF**, demonstrating the better alignment of **RG**^{2} with ranking metrics within the same ALS optimization method. The detailed analysis of convergence is referred to in Section 4.3.

4.3. Comparison of Convergence Speed

Furthermore, to investigate the convergence speed of our method, we record the metrics after each epoch evaluated on the validation set during the training process over all three datasets, as illustrated in Figure 1, where we record the data point of each epoch and their running time.

As shown in the figures, both **RG**^{2} and **WRMF** demonstrate faster convergence speed than **Softmax** in terms of NDCG and MRR, proving the efficacy of ALS optimization that directly optimizes the convex problem with a closed-form solution. Notably, our RG^2 shows a comparable final performance to **Softmax** which exceeds **WRMF** and all other baselines, indicating that RG^2 possesses both efficacy from softmax loss approximation and efficiency from ALS optimization. Besides, sampling-based approaches, such as **BPR** and **SampledSoftmax** can reduce the training time of each epoch compared with **Softmax**, especially for datasets with more items, such as Electronics. However, they all exhibit poor performances.

5. Conclusion

In conclusion, our exploration into squared-form loss functions and the introduction of the RG^2 loss mark strides in the domain of recommender systems. The RG^2 loss, ingeniously approximating and upper bounding the Softmax loss through Taylor expansion, represents an innovative forward in item recommendation. This innovation maintains RG^2's alignment with DCG, thus ensuring relevance and performance on ranking metrics. Our rigorous empirical analysis, conducted across three public datasets with both MF and GNN-based models, has confirmed the superiority of the RG^2 approach. Through the adaptation to both ALS and SGD optimizations, RG^2 not only achieves performance comparable to the established benchmarks of Softmax loss but, in several instances, surpasses them, demonstrating remarkable efficiency improvements in the training process without compromising on performance. Such advancements underscore the potential of squared-form loss functions in enhancing the scalability and effectiveness of recommender systems.

Acknowledgements

The work was supported by grants from the National Natural Science Foundation of China (No. 62022077).

Impact Statement

This paper presents work whose goal is to advance the field of Recommender Systems, which is a typical application of Machine Learning. There are many potential societal consequences of our work, including personalized performance enhancement for web platforms, e-commerce or online advertisement, none of which we feel must be specifically highlighted here.

References

Amari, S.-i. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5(4-5):185–196, 1993.

Anthony, M., Bartlett, P. L., Bartlett, P. L., et al. *Neural network learning: Theoretical foundations*, volume 9. cambridge university press Cambridge, 1999.

Bayer, I., He, X., Kanagal, B., and Rendle, S. A generic coordinate descent framework for learning from implicit feedback. In *Proceedings of the 26th international conference on world wide web*, pp. 1341–1350, 2017.

Bengio, Y. and Senécal, J.-S. Adaptive importance sampling to accelerate training of a neural probabilistic language model. *IEEE Transactions on Neural Networks*, 19(4): 713–722, 2008.

Bruch, S. An alternative cross entropy loss for learning-to-rank. In *Proceedings of the web conference 2021*, pp. 118–126, 2021.

Bruch, S., Wang, X., Bendersky, M., and Najork, M. An analysis of the softmax cross entropy loss for learning-to-rank with binary relevance. In *Proceedings of the 2019 ACM SIGIR international conference on theory of information retrieval*, pp. 75–78, 2019.

Chen, C., Zhang, M., Zhang, Y., Liu, Y., and Ma, S. Efficient neural matrix factorization without sampling for recommendation. *ACM Transactions on Information Systems (TOIS)*, 38(2):1–28, 2020.

Chen, C., Ma, W., Zhang, M., Wang, C., Liu, Y., and Ma, S. Revisiting negative sampling vs. non-sampling in implicit recommendation. *ACM Transactions on Information Systems*, 41(1):1–25, 2023.

Cossock, D. and Zhang, T. Subset ranking using regression. In *Learning Theory: 19th Annual Conference on Learning Theory, COLT 2006, Pittsburgh, PA, USA, June 22-25, 2006. Proceedings 19*, pp. 605–619. Springer, 2006.

Covington, P., Adams, J., and Sargin, E. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pp. 191–198, 2016.

Ding, J., Yu, G., He, X., Quan, Y., Li, Y., Chua, T.-S., Jin, D., and Yu, J. Improving implicit recommender systems with view data. In *IJCAI*, pp. 3343–3349, 2018.

Guo, H., Tang, R., Ye, Y., Li, Z., and He, X. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247*, 2017.

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., and Chua, T.-S. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pp. 173–182, 2017.

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., and Wang, M. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 639–648, 2020.

Hu, Y., Koren, Y., and Volinsky, C. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE international conference on data mining*, pp. 263–272. Ieee, 2008.

Huang, P.-S., He, X., Gao, J., Deng, L., Acero, A., and Heck, L. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pp. 2333–2338, 2013.

Huang, X., Lian, D., Chen, J., Zheng, L., Xie, X., and Chen, E. Cooperative retriever and ranker in deep recommenders. In *Proceedings of the ACM Web Conference 2023*, pp. 1150–1161, 2023.

Kang, W.-C. and McAuley, J. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pp. 197–206. IEEE, 2018.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Koren, Y., Bell, R., and Volinsky, C. Matrix factorization techniques for recommender systems. *Computer*, 42(8): 30–37, 2009.

Krichene, W., Mayoraz, N., Rendle, S., Zhang, L., Yi, X., Hong, L., Chi, E., and Anderson, J. Efficient training on very large corpora via gramian estimation. In *International Conference on Learning Representations*, 2018.

Li, M., Zhang, S., Zhu, F., Qian, W., Zang, L., Han, J., and Hu, S. Symmetric metric learning with adaptive margin for recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 4634–4641, 2020.

Lian, D., Zhao, C., Xie, X., Sun, G., Chen, E., and Rui, Y. Geomf: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 831–840, 2014.

Lian, D., Liu, Q., and Chen, E. Personalized ranking with importance sampling. In *Proceedings of The Web Conference 2020*, pp. 1093–1103, 2020a.

Lian, D., Wang, H., Liu, Z., Lian, J., Chen, E., and Xie, X. Lightrec: A memory and search-efficient recommender system. In *Proceedings of The Web Conference 2020*, pp. 695–705, 2020b.

Lian, D., Huang, X., Chen, X., Chen, J., Wang, X., Wang, Y., Jin, H., Fan, R., Liu, Z., Wu, L., et al. Recstudio: Towards a highly-modularized recommender system. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2890–2900, 2023.

Liang, D., Krishnan, R. G., Hoffman, M. D., and Jebara, T. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 world wide web conference*, pp. 689–698, 2018.

Martins, A. and Astudillo, R. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International conference on machine learning*, pp. 1614–1623. PMLR, 2016.

Ning, X. and Karypis, G. Slim: Sparse linear methods for top-n recommender systems. In *2011 IEEE 11th international conference on data mining*, pp. 497–506. IEEE, 2011.

Pan, R., Zhou, Y., Cao, B., Liu, N. N., Lukose, R., Scholz, M., and Yang, Q. One-class collaborative filtering. In *2008 Eighth IEEE international conference on data mining*, pp. 502–511. IEEE, 2008.

Ravikumar, P., Tewari, A., and Yang, E. On ndcg consistency of listwise ranking methods. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 618–626. JMLR Workshop and Conference Proceedings, 2011.

Rendle, S. Item recommendation from implicit feedback. In *Recommender Systems Handbook*, pp. 143–171. Springer, 2021.

Rendle, S. and Freudenthaler, C. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pp. 273–282, 2014.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. Bpr: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*, 2012.

Rendle, S., Krichene, W., Zhang, L., and Koren, Y. Ials++: Speeding up matrix factorization with subspace optimization. *arXiv preprint arXiv:2110.14044*, 2021.

Rendle, S., Krichene, W., Zhang, L., and Koren, Y. Revisiting the performance of ials on item recommendation benchmarks. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pp. 427–435, 2022.

Shen, Y., He, X., Gao, J., Deng, L., and Mesnil, G. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd international conference on world wide web*, pp. 373–374, 2014.

Srebro, N., Alon, N., and Jaakkola, T. Generalization error bounds for collaborative prediction with low-rank matrices. *Advances In Neural Information Processing Systems*, 17, 2004.

Steck, H. Embarrassingly shallow autoencoders for sparse data. In *The World Wide Web Conference*, pp. 3251–3257, 2019.

Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., and Jiang, P. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pp. 1441–1450, 2019.

Takács, G. and Tikk, D. Alternating least squares for personalized ranking. In *Proceedings of the sixth ACM conference on Recommender systems*, pp. 83–90, 2012.

Weston, J., Bengio, S., and Usunier, N. Wsabie: Scaling up to large vocabulary image annotation. 2011.

Wu, J., Wang, X., Gao, X., Chen, J., Fu, H., Qiu, T., and He, X. On the effectiveness of sampled softmax loss for item recommendation. *arXiv preprint arXiv:2201.02327*, 2022.

Yang, F. and Koyejo, S. On the consistency of top-k surrogate losses. In *International Conference on Machine Learning*, pp. 10727–10735. PMLR, 2020.

Yi, X., Yang, J., Hong, L., Cheng, D. Z., Heldt, L., Kumthekar, A., Zhao, Z., Wei, L., and Chi, E. Sampling-bias-corrected neural modeling for large corpus item recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pp. 269–277, 2019.

Yuan, B., Li, Y.-S., Quan, P., and Lin, C.-J. Efficient optimization methods for extreme similarity learning with nonlinear embeddings. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 2093–2103, 2021.

Zhang, W., Chen, T., Wang, J., and Yu, Y. 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*, pp. 785–788, 2013.

Zhou, G., Zhu, X., Song, C., Fan, Y., Zhu, H., Ma, X., Yan, Y., Jin, J., Li, H., and Gai, K. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 1059–1068, 2018.

Zhuo, J., Zhu, Q., Yue, Y., and Zhao, Y. Learning explicit user interest boundary for recommendation. In *Proceedings of the ACM Web Conference 2022*, pp. 193–202, 2022.

A. Related Works

Recommender systems have become an indispensable component in a wide range of applications, significantly enhancing user experience by providing personalized suggestions (Huang et al., 2013; Guo et al., 2017; Zhou et al., 2018; Lian et al., 2020b). Collaborative filtering is a typical technique in recommender systems, leveraging similarities between user behaviors to predict user preferences. As mentioned above, the objective functions in CF for item recommendation tasks can be separated into two tracks: **sampling** methods and **non-sampling** methods.

A.1. Non-sampling Loss for Recommenders

In recommendation scenarios, the feedback data is characterized by a highly sparse nature, with only a small fraction of user-item pairs displaying interaction. The selection of an appropriate loss function is crucial for optimizing the performance of recommendation algorithms, especially when dealing with a large number of un-interacted items. In recent research, the community has raised a consensus that taking all user-item pairs into consideration, rather than leveraging small sample fractions, leads to performance improvement (Rendle, 2021; Chen et al., 2023; 2020; Yuan et al., 2021). This kind of approach is generally regarded as non-sampling losses.

• **Squared Loss**. One straightforward approach is to treat all interacted samples as positive and non-interacted samples as negative (Ning & Karypis, 2011). By assigning target scores of 1 and 0, respectively, the algorithm can perform regression on these samples. The weighted regression squared loss (Hu et al., 2008; Ding et al., 2018) further improved the performance by introducing weights to capture the confidence of each user-item pair. Due to the absence of non-linear operations, these loss functions exhibit the advantageous property of possessing closed-form solutions. Consequently, optimization becomes more tractable and efficient by utilizing methods like alternating least squares (Hu et al., 2008; Takács & Tikk, 2012) and coordinate descent (Bayer et al., 2017). However, with the increasing number of users/items and the complex encoders for embeddings, the update of all maintained large matrices in these methods can become computationally costly. To address this, researchers have proposed approaches such as the Newton method (Yuan et al., 2021) or the Gram-Matrix trick (Krichene et al., 2018), which facilitate efficient learning of non-linear embeddings. Another challenge is the limited correlation observed between the regression squared loss function and ranking metrics, which lacks deep exploration compared to its well-established exceptional ranking performance.

• **Softmax Loss**. Unlike the squared loss, which treats ranking problems as regression tasks, the softmax loss takes a different approach, which assumes that user interests follow a multinomial distribution (Shen et al., 2014; Liang et al., 2018; Sun et al., 2019) and aims to maximize the likelihood function. The softmax loss has been proven to align well with the ranking metrics (Bruch et al., 2019; Ravikumar et al., 2011; Huang et al., 2023). This alignment brings notable advantages to the softmax loss, particularly in scenarios with implicit feedback. However, the presence of the exponential operation in the softmax loss prevents the derivation of a closed-form solution or higher-order gradient over all items, necessitating stochastic gradient descent (SGD) methods for optimization. SGD approximates the global gradient by sampling a subset of data in each iteration, which would result in slower convergence compared to methods utilizing closed-form solutions. Besides, the training process involves the summation of all predicted scores over the entire dataset, which has significant computational complexity. Feasible solutions include approximating the item set scores using sampling methods (Wu et al., 2022) or employing sparse approximations of the original probability vector (Martins & Astudillo, 2016).

A.2. Sampling Loss for Recommenders

Non-sampling methods that consider relationships across the entire item set generally deliver superior accuracy. However, their computational costs are generally prohibitive, particularly when dealing with a significantly large number of items. Negative sampling has been introduced as an efficient approach to mitigate this challenge. This kind of method involves sampling a subset of items as negative samples and approximating the computation over the entire item set. Particularly, Bayesian Personalized Ranking (BPR) loss (Rendle et al., 2012), a pair-wise ranking loss, uniformly samples items from the un-interacted items as negatives to help distinguish the positive samples. To select more informative items, several sampling strategies have been proposed, including WARP (Weston et al., 2011), AOBPR (Rendle & Freudenthaler, 2014), DNS (Zhang et al., 2013), to enhance the performance during the training process. Additionally, PRIS (Lian et al., 2020a) assigns different weights to different ranking pairs based on their importance, which attends further enhancements. Another commonly used approach is sampled softmax (Yi et al., 2019; Bengio & Senécal, 2008), which estimates the gradient expectation through importance sampling. Although these sampling-based methods effectively reduce the computational overhead for large-scale recommender systems by reducing the number of computed items, it is crucial to recognize that these approximations can introduce biases and variances in the estimation for the first-order gradient. This bias comes from the limited number of sampled items and the biased sampling distribution. The challenges become even more significant when approximating higher-order gradients, rendering advanced optimization algorithms inapplicable.

B. Computational and Proof Details

B.1. Notations

All notations in this paper are listed in Table.3.

Table 3: Table of Notations

Notation	Description
u, i	A context (user, location, behavior history, etc.) and an item
$\mathcal{U}, \mathcal{I}$	The context set and item set
M, N, K	The number of contexts(users), items and embedded dimensions
$\mathcal{D}$	The interaction set with all interacted pairs $(u, i) \in \mathcal{D}$
R	The interaction matrix
r_{ui}	The (u, i)-th element of interaction matrix
$\mathcal{X}$	The sample space where interaction set drawn from
$\mathcal{P}(\mathcal{X})$	Some unknown distribution on sample space $\mathcal{X}$
$\mathcal{H}$	The hypothesis space
h	A hypothesis function in hypothesis space $\mathcal{H}$
$\hat{\mathcal{R}}_{\mathcal{D}}(h)$	The empirical error of h on dataset $\mathcal{D}$
$\mathcal{R}(h)$	The generalization error of h on $\mathcal{P}(\mathcal{X})$
$o_i^{(u)}$	The predicted preference score of item i in context u
$\boldsymbol{o}^{(u)}$	The vector formed by the scores of context u on all items
$p(o_i^{(u)})$	The normalized probability transformed by softmax function
$\boldsymbol{\theta}_u, \boldsymbol{\theta}_i$	The learnt representations of context(user) u and item i
P, Q	The matrix form of representations by stacking all $\boldsymbol{\theta}_u, \boldsymbol{\theta}_i$ into $M \times K, N \times K$ shape
$P_{u\cdot}, Q_{i\cdot}$	The u-th row and i-th row of P, Q
W	The weight matrix used in Weighted Alternating Least Square Algorithm
S	The matrix used in Weighted Alternating Least Square Algorithm
$\lambda\psi(\theta)$	Some regularization term with coefficient λ
$\pi^{(u)}(i)$	The rank of $o_i^{(u)}$ in $\boldsymbol{o}^{(u)}$
$\mathcal{I}_u$	The set of all items interacted in context u
ϕ	A surrogate loss function
D_ϕ	The Bregman Divergence of function ϕ
Φ	The expectation form of surrogate loss ϕ
Φ^*	The minimum of Φ

B.2. Details of approximated loss

Although the derivation is not complicated, the main procedure of the softmax loss Taylor expansion is provided as follows.

$$\mathcal{L}(\boldsymbol{o}^{(u)}) \overset{Taylor}{=} \mathcal{L}(\mathbf{0}) + \nabla\mathcal{L}(\mathbf{0})^\top \boldsymbol{o}^{(u)} + \frac{1}{2}\boldsymbol{o}^{(u)}{}^\top \nabla^2\mathcal{L}(\mathbf{0})\boldsymbol{o}^{(u)}$$

$$\mathcal{L}(\mathbf{0}) = -\log(\frac{1}{N}) = \log N$$

$$\nabla\mathcal{L} = [\frac{\partial\mathcal{L}}{\partial o_{i_1}^{(u)}}, \cdots, \frac{\partial\mathcal{L}}{\partial o_{i_N}^{(u)}}]^\top = [p(o_{i_1}^{(u)}), p(o_{i_2}^{(u)}), \cdots, p(o_i^{(u)}) - 1, \cdots, p(o_{i_N}^{(u)})]^\top$$

$$\implies \nabla\mathcal{L}(\mathbf{0})^\top \boldsymbol{o}^{(u)} = -o_i^{(u)} + \frac{1}{N}\mathbf{1}_N^\top \boldsymbol{o}^{(u)}$$

As for the second-order parts, we have

$$\frac{\partial^2 \mathcal{L}}{\partial (o_j^{(u)})^2} = p(o_j^{(u)})(1 - p(o_j^{(u)})), \frac{\partial^2 \mathcal{L}}{\partial (o_j^{(u)}) \partial (o_k^{(u)})} = -p(o_j^{(u)}) p(o_k^{(u)})$$
$$\implies \nabla^2 \mathcal{L}(\mathbf{0}) = \frac{1}{N}(\boldsymbol{I} - \frac{1}{N}\mathbf{1}_N \mathbf{1}_N^\top)$$

from which we arrive at the given final form.

Further, to prove that the approximation is an upper bound for the original softmax loss, consider the following theorem:

Theorem B.1. *For any multi-variable function $f : x \to \mathbb{R}, x \in \mathbb{R}^N$ with Jacobian vector ∇f and Hessian matrix $\nabla^2 f$, let M be a symmetric matrix satisfying $M \succeq \nabla^2 f$, then for $\forall x, y \in \mathbb{R}^N$,*

$$f(y) \leq f(x) + \nabla f(x)(y - x) + \frac{1}{2}(y - x)^\top M (y - x) \tag{35}$$

Proof. With the mean value theorem, one can simply expand $f(y)$ with

$$f(y) = f(x) + \nabla f(x)(y - x) + \frac{1}{2}(y - x)^\top \nabla^2 f(\xi)(y - x) \tag{36}$$

Now that $M \succeq \nabla^2 f$, thus for $\forall x \in \mathbb{R}^N, x^\top M x \geq x^\top \nabla^2 f(\xi) x$, hence finishing the proof. □

With the following assumption, the symmetric matrix $M = \frac{1}{N} I_N$ is greater than $\nabla^2 f$.

Proposition B.2. *Suppose that $\frac{1}{N} I_N - diag(a_i)_{i=1}^N + (a_j a_k)_{j,k=1}^{N,N}$ is positive semi-definite, where $a_i = p(o_i^{(u)})$, then the softmax loss $\mathcal{L}_{SM}$ is upper bounded by $\mathcal{L}_{eps}$.*

The proof is a direct application of the above theorem. Note that a_i satisfies $\sum_{i=1}^N a_i = 1$ and $a_i \geq 0, \forall i$. This positive semi-definite assumption holds when the predicted logits have a sufficiently small distance from **0**, either centralized on certain points or uniformly distributed among all instances.

For centralized circumstances, consider one specific item with probability 1 and all other logits equal to 0, i.e. $a_i = 1, a_{j \neq i} = 0$ Then for $\forall [x_1, \cdots, x_N] = \boldsymbol{x} \in \mathbb{R}^N$,

$$\boldsymbol{x}^\top \left(\frac{1}{N} I_N - \text{diag}(a_i)_{i=1}^N + (a_j a_k)_{j,k=1}^{N,N} \right) \boldsymbol{x} = \frac{1}{N} \sum_{j=1}^N x_j^2 - x_i^2 + x_i^2 = \frac{1}{N} \sum_{j=1}^N x_j^2 \geq 0$$

For uniformly distributed instances, all logits have an equal value of $\frac{1}{N}$, then

$$\boldsymbol{x}^\top \left(\frac{1}{N} I_N - \text{diag}(a_i)_{i=1}^N + (a_j a_k)_{j,k=1}^{N,N} \right) \boldsymbol{x} = \frac{1}{N} \sum_{j=1}^N x_j^2 - \frac{1}{N} \sum_{j=1}^N x_j^2 + \frac{1}{N^2} \sum_{j=1}^N x_j^2 = \frac{1}{N^2} \sum_{j=1}^N x_j^2 \geq 0$$

As a result, in the practice of recommendation scenarios, since implicit feedback can be modeled as single-click behaviors, we can safely conclude this semi-positive assumption acceptable.

B.3. Transformation to squared form

Starting from $\mathcal{L}_{\text{eps}}$, we are able to obtain an equivalent $\mathcal{L}_{\text{RG}^2}$ after a simple transformation,

$$
\begin{aligned}
\mathcal{L}_{\text{eps}} &= -\frac{1}{|\mathcal{D}|}\sum_{(u,i)\in\mathcal{D}}\left(o_i^{(u)} - \frac{1}{2N}\|\boldsymbol{o}^{(u)}+\mathbf{1}_N\|^2\right) + \lambda\psi(\theta) \\
&= -\frac{1}{|\mathcal{D}|}\sum_{(u,i)\in\mathcal{D}}\left(o_i^{(u)} - \sum_{j=1}^{N}\frac{1}{2N}(o_j^{(u)}+1)^2\right) + \lambda\psi(\theta) \\
&= -\frac{1}{|\mathcal{D}|}\left(\sum_{(u,i)\in\mathcal{D}} o_i^{(u)} - \sum_{u\in\mathcal{U}}|\mathcal{I}_u|\sum_{j=1}^{N}\frac{1}{2N}(o_j^{(u)}+1)^2\right) + \lambda\psi(\theta) \\
&= -\frac{1}{|\mathcal{D}|}\left(\sum_{(u,i)\in\mathcal{D}}\left(o_i^{(u)} - \frac{|\mathcal{I}_u|}{2N}(o_i^{(u)}+1)^2\right) - \sum_{(u,i)\notin\mathcal{D}}\frac{|\mathcal{I}_u|}{2N}(o_j^{(u)}+1)^2\right) + \lambda\psi(\theta) \\
&= -\frac{1}{|\mathcal{D}|}\left(\sum_{(u,i)\in\mathcal{D}} -\frac{|\mathcal{I}_u|}{2N}\left(o_i^{(u)2} + 2o_i^{(u)} - \frac{2N}{|\mathcal{I}_u|}o_i^{(u)} + 1\right) - \sum_{(u,i)\notin\mathcal{D}}\frac{|\mathcal{I}_u|}{2N}(o_j^{(u)}+1)^2\right) + \lambda\psi(\theta) \\
&= -\frac{1}{|\mathcal{D}|}\left(\sum_{(u,i)\in\mathcal{D}} -\frac{|\mathcal{I}_u|}{2N}\left(\left(o_i^{(u)} + 1 - \frac{N}{|\mathcal{I}_u|}\right)^2 + 1 - \left(1 - \frac{N}{|\mathcal{I}_u|}\right)^2\right) - \sum_{(u,i)\notin\mathcal{D}}\frac{|\mathcal{I}_u|}{2N}(o_j^{(u)}+1)^2\right) + \lambda\psi(\theta) \\
&\propto -\frac{1}{|\mathcal{D}|}\left(\sum_{(u,i)\in\mathcal{D}} -\frac{|\mathcal{I}_u|}{2N}\left(\left(o_i^{(u)} + 1 - \frac{N}{|\mathcal{I}_u|}\right)^2\right) - \sum_{(u,i)\notin\mathcal{D}}\frac{|\mathcal{I}_u|}{2N}(o_j^{(u)}+1)^2\right) + \lambda\psi(\theta) \\
&= -\frac{1}{|\mathcal{D}|}\sum_{u\in\mathcal{U},i\in\mathcal{I}} -\frac{|\mathcal{I}_u|}{2N}\left(o_i^{(u)} + 1 - r_{ui}\frac{N}{|\mathcal{I}_u|}\right)^2 + \lambda\psi(\theta) = \mathcal{L}_{\text{RG}^2}
\end{aligned}
$$

B.4. Proof of Objective Consistency

Inspired by (Cossock & Zhang, 2006), we resort to an intuitive loss as a springboard to complete the proof of alternative loss consistency. For the sake of simplicity, we define $c_i = \frac{1}{\log(1+i)}$ and $f(u,i) = o_i^{(u)}$. Besides, we replace the notation $\pi^{(u)}(i)$ into π_i with some fixed user u, and $\boldsymbol{\pi} = \{\pi_1, \cdots, \pi_N\}$. Given that our task is built on a 0-1 interaction dataset, we have $2^{r_{ui}} - 1 = r_{ui}$, thus rewriting DCG as

$$\text{DCG}(\pi, \mathcal{D}) = \sum_{i=1}^{N} c_{\pi_i} r_{ui}$$

Given any dataset $\mathcal{D}$ drawn from $\mathcal{X}$, the Bayesian scoring function is defined as:

$$f_B(u,i) = \mathbb{E}_{\mathcal{D}} r_{ui}$$

which optimizes $\mathcal{L}_{\text{DCG}}$ and assumed available in the hypothesis space $\mathcal{H}$. Correspondingly, $\boldsymbol{\pi}^* = [\pi_1^*, \cdots, \pi_N^*]$ stands for rank information of $f_B(u,i), i = 1, \cdots, N$. Based on the above discussion, we have the following lemma:

Lemma B.3.

$$\mathcal{L}_{DCG}(\boldsymbol{\pi}) - \mathcal{L}_{DCG}^* \leq \left(2\sum_{i=1}^{N} c_i^2\right)^{\frac{1}{2}} \left(\sum_{i=1}^{N} (f(u,i) - f_B(u,i))^2\right)^{\frac{1}{2}} \tag{37}$$

Proof.

$$\begin{aligned}
\mathcal{L}_{\text{DCG}}(\boldsymbol{\pi}) &= \mathbb{E}_{\mathcal{D}}\left[-\sum_{i=1}^{N} c_{\pi_i} r_{ui}\right] = -\sum_{i=1}^{N} c_{\pi_i} f_B(u,i) \\
&= -\sum_{i=1}^{N} c_{\pi_i} f(u,i) - \sum_{i=1}^{N} c_{\pi_i}\left(f_B(u,i) - f(u,i)\right) \\
&\leq -\sum_{i=1}^{N} c_{\pi_i^*} f(u,i) - \sum_{i=1}^{N} c_{\pi_i}\left(f_B(u,i) - f(u,i)\right) \\
&= -\sum_{i=1}^{N} c_{\pi_i^*} f_B(u,i) - \sum_{i=1}^{N} c_{\pi_i^*}\left(f(u,i) - f_B(u,i)\right) - \sum_{i=1}^{N} c_{\pi_i}\left(f_B(u,i) - f(u,i)\right) \\
&= \mathcal{L}_{\text{DCG}}^* - \sum_{i=1}^{N} c_{\pi_i^*}\left(f(u,i) - f_B(u,i)\right) - \sum_{i=1}^{N} c_{\pi_i}\left(f_B(u,i) - f(u,i)\right) \\
&\leq \mathcal{L}_{\text{DCG}}^* + \left(2\sum_{i=1}^{N} c_i^2\right)^{\frac{1}{2}} \left(\sum_{i=1}^{N}\left(f(u,i) - f_B(u,i)\right)^2\right)^{\frac{1}{2}}
\end{aligned}$$

□

Consider the following surrogate loss function,

$$\phi_{sur}(f,\mathcal{D}) = \sum_{i=1}^{N}\left(f(u,i) - r_{ui}\right)^2 = \sum_{i|(u,i)\in\mathcal{D}}\left(f(u,i) - 1\right)^2 + \sum_{i|(u,i)\notin\mathcal{D}}\left(f(u,i)\right)^2 \tag{38}$$

$$\Phi_{sur}(f) = \mathbb{E}_{\mathcal{D}}[\phi_{sur}(f,\mathcal{D})] = \sum_{i=1}^{N}\left(f(u,i) - f_B(u,i)\right)^2 \tag{39}$$

with the following theorem holding consistency from the above lemma:

Theorem B.4. *The surrogate loss Φ_{sur} is DCG-consistent, i.e.,*

$$\mathcal{L}_{DCG}(\boldsymbol{\pi}) - \mathcal{L}_{DCG}^* \leq \left(2\sum_{i=1}^{N} c_i^2\right)^{\frac{1}{2}} \left(\Phi_{sur}(f) - \Phi_{sur}^*\right)^{\frac{1}{2}} \tag{40}$$

Notably, the ground truth of interaction scores is not strictly limited to $r_{ui} \in \{0,1\}$. Since DCG only cares about the rank information of predicted scores, with any order-preserving mapping $g(\cdot) : \mathbb{R} \to \mathbb{R}$, DCG$(f)$ = DCG$(g(f))$. Thus we are free to replace r_{ui} in Theorem.B.4 and obtain the following corollary,

Corollary B.5. *A squared-form surrogate function with the following form*

$$\phi_{sur}(f,\mathcal{D}) = \sum_{i=1}^{N} w_{ui}(f(u,i) - r_{ui})^2 \tag{41}$$

is DCG-consistent if and only if

1. $w_{ui} > 0, \forall i \in [N]$

2. $\min_{(u,i)\in\mathcal{D}} r_{ui} > \max_{(u,i)\notin\mathcal{D}} r_{ui}$

Based on this corollary, we safely obtain the objective consistency of $\mathcal{L}_{\text{RG}^2}$ with the DCG metric, hence finishing the proof.

C. Experimental Results

C.1. Confidence Intervals

The experiments in Table. 2 are conducted where each algorithm is run for 5 competitions with different random seeds. Results with standard deviation are reported in Table. 4.

Table 4: Confidence Intervals of all baselines and **RG**^{2}.

Dataset	MovieLens		Electronics		Steam	
Metric	MRR@10	NDCG@10	MRR@10	NDCG@10	MRR@10	NDCG@10
BPR	0.3476±0.0038	0.2116±0.0016	0.0124±0.0003	0.0154±0.0004	0.0434±0.0005	0.0521±0.0005
BCE	0.3602±0.0007	0.2228±0.0006	0.0098±0.0002	0.0126±0.0002	0.0435±0.0007	0.0532±0.0005
S-Softmax	0.3785±0.0032	0.2378±0.0014	0.0119±0.0003	0.0152±0.0003	0.0466±0.0005	0.0549±0.0005
Sparsemax	0.3177±0.0054	0.1738±0.0024	0.0078±0.0006	0.0100±0.0004	0.0317±0.0003	0.0359±0.0004
UIB	0.3836±0.0020	0.2045±0.0018	0.0101±0.0004	0.0130±0.0003	0.0410±0.0008	0.0506±0.0006
SML	0.3386±0.0043	0.2045±0.0017	0.0074±0.0002	0.0098±0.0005	0.0307±0.0005	0.0353±0.0008
WRMF	0.4475±0.0014	0.2797±0.0008	0.0146±0.0003	0.0178±0.0002	0.0464±0.0003	0.0544±0.0003
Softmax	0.4487±0.0037	0.2849±0.0026	0.0172±0.0002	0.0212±0.0002	0.0493±0.0001	0.0579±0.0002
RG^2	0.4723±0.0014	0.2963±0.0004	0.0173±0.0001	0.0212±0.0002	0.0492±0.0002	0.0575±0.0002

C.2. RG^2 Loss for LightGCN

To verify the effectiveness of the objective function in **RG**^{2}, we extend the evaluation beyond the traditional MF framework to include the backbone of a typical GNN-based recommender – LightGCN (He et al., 2020). Despite our inclination towards the ALS method, which is inherently suited for MF, we do not intend but have to utilize Stochastic Gradient Descent Methods for fair comparison on LightGCN backbone.

Table 5: Comparisons of recommendation performance on LightGCN backbone. We select well-performed baselines in Table. 2 to validate **RG**^{2}'s consistency in evaluation. **Bold** and underline numbers represent the best and the second-best results respectively.

Dataset	Metric	UIB	SML	Softmax	RG^2
MovieLens	MRR@10	0.3530	0.2482	0.4414	**0.4652**
	NDCG@10	0.2171	0.1535	0.2814	**0.2919**
Electronics	MRR@10	0.0152	0.0088	0.0171	**0.0176**
	NDCG@10	0.0191	0.0115	0.0215	**0.0219**
Steam	MRR@10	**0.0486**	0.0396	0.0476	0.0464
	NDCG@10	**0.0569**	0.0463	0.0564	0.0542

We perform a 3-layer LightGCN with the embedding size set to 64 on each layer, all baselines using the same experimental settings for other parameters in Section 4.1.3 in our manuscript, and select well-performed objective loss functions in Table. 2 for comparison. All experimental results on LightGCN are shown in Table. 5. Our experimental results demonstrate that **RG**^{2} still performs competitively in GNN-based backbones on ML-10M and Electronics, consistent with MF results.

D. Gauss-Newton Method for Optimization

The utilization of ALS to optimize squared-form losses for general linear models has been discussed in the main text, which cannot be applied when complex encoders are applied for user-item representations. Although gradient descent methods are still applicable and widely used, methods leveraging higher-order gradient information, such as Newton's method, still have the potential to improve the efficiency of model convergence. The main obstacle for current recommenders on adapting to higher-order methods is the high complexity of solving second-order gradient information (or Hessian matrix), which is particularly evident for loss functions involving nonlinear operations. NewtonCG (Yuan et al., 2021) proposed an optimization method for extreme similarity learning based on Newton's method, where they took the $O(MN)$ complexity into account in optimizing loss function by transforming all the unobserved interactions ($(u, i) \notin \mathcal{D}$) with a certain squared-form structure. Suppose ℓ is the original loss function, the transformation applies by:

$$\hat{\ell}(o_i^{(u)}) = \begin{cases} \ell(o_i^{(u)}) & (u,i) \in \mathcal{D} \\ \frac{1}{2}\omega a_u b_i (o_i^{(u)} - r_{ui})^2 & (u,i) \notin \mathcal{D} \end{cases}$$

In this way, they were able to replace the $O(MN)$ complexity with a smaller $O(M+N)$. It is worth noting that our proposed RG^2 in Eq.(13) satisfies this form exactly and directly, without introducing any bias due to the approximation to un-interacted pairs. This allows us to directly plug in our proposed loss function to NewtonCG and update the model with second-order gradient information. This opens up the possibility for our RG^2 to handle models beyond the capability of ALS, like deep encoders, demonstrating the flexibility of this loss function and its ability to remain efficient in more scenarios.

For simplicity, we use $\boldsymbol{\theta}_u, \boldsymbol{\theta}_i \in \mathbb{R}^K$ as outputs of user-side and item-side encoders for a two-tower recommender with an inner product scoring. By a direct reformatting of Eq.(13), we have

$$\begin{aligned} \mathcal{L}_{\text{RG}^2} &= -\frac{1}{|\mathcal{D}|} \sum_{(u,i)\in\mathcal{D}} (o_i^{(u)} - \frac{1}{2N}\|\boldsymbol{o}_u + \mathbf{1}\|^2) + \lambda\psi(\boldsymbol{\theta}) \\ &= -\frac{1}{|\mathcal{D}|} \sum_{(u,i)\in\mathcal{D}} (\boldsymbol{\theta}_u^\top \boldsymbol{\theta}_i) + \frac{1}{2N|\mathcal{D}|} \sum_{u\in\mathcal{U}} \sum_{i\in\mathcal{I}} |\mathcal{I}_u| (\boldsymbol{\theta}_u^\top \boldsymbol{\theta}_i + 1)^2 + \lambda\psi(\boldsymbol{\theta}) \end{aligned}$$

Let $\tilde{\boldsymbol{\theta}}_u, \tilde{\boldsymbol{\theta}}_i$ be fixed throughout training with $\tilde{\boldsymbol{\theta}}_u^\top \tilde{\boldsymbol{\theta}}_i = -1$ (with all elements equal to $-\frac{1}{\sqrt{K}}$), thus we have

$$\mathcal{L}_{\text{RG}^2} = -\frac{1}{|\mathcal{D}|} \sum_{(u,i)\in\mathcal{D}} \boldsymbol{\theta}_u^\top \boldsymbol{\theta}_i + \frac{1}{2N|\mathcal{D}|} \sum_{u\in\mathcal{U}} \sum_{i\in\mathcal{I}} |\mathcal{I}_u| (\boldsymbol{\theta}_u^\top \boldsymbol{\theta}_i - \tilde{\boldsymbol{\theta}}_u^\top \tilde{\boldsymbol{\theta}}_i)^2 + \lambda\psi(\boldsymbol{\theta}).$$

Divide the loss form into $\mathcal{L}^+$ and $\mathcal{L}^-$ with

$$\begin{aligned} \mathcal{L}^+ &= -\frac{1}{|\mathcal{D}|} \sum_{(u,i)\in\mathcal{D}} \boldsymbol{\theta}_u^\top \boldsymbol{\theta}_i \\ \mathcal{L}^- &= \frac{1}{2N|\mathcal{D}|} \sum_{u\in\mathcal{U}} \sum_{i\in\mathcal{I}} |\mathcal{I}_u| (\boldsymbol{\theta}_u^\top \boldsymbol{\theta}_i - \tilde{\boldsymbol{\theta}}_u^\top \tilde{\boldsymbol{\theta}}_i)^2. \end{aligned}$$

Let $a_u = |\mathcal{I}_u|$, then

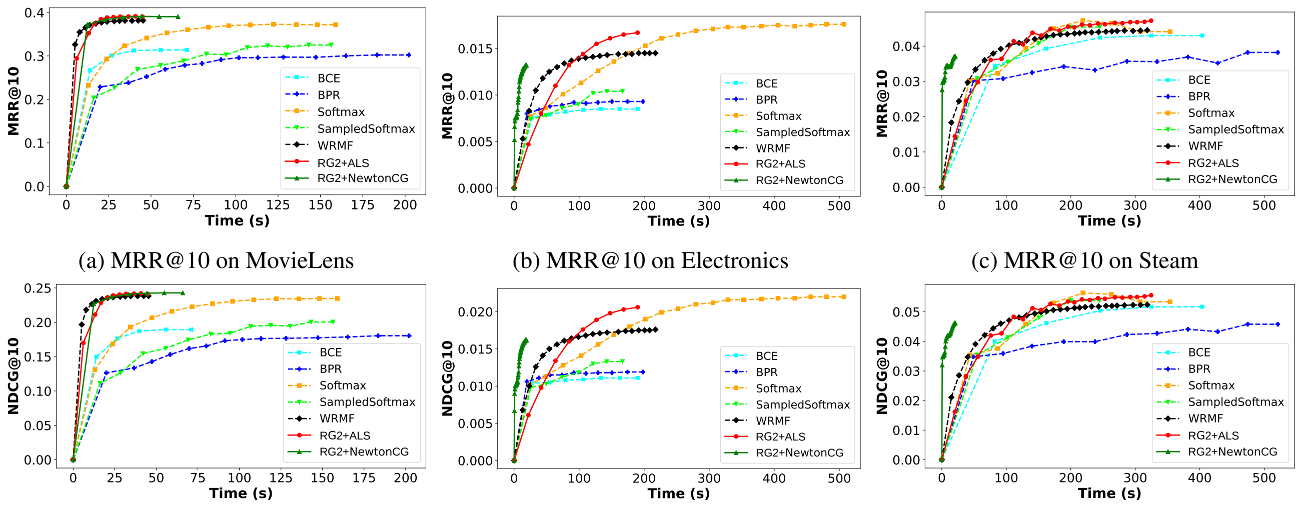
$$\begin{aligned} \mathcal{L}^- &= \frac{1}{2N|\mathcal{D}|} \sum_{u\in\mathcal{U}} \sum_{i\in\mathcal{I}} a_u (\boldsymbol{\theta}_u^\top \boldsymbol{\theta}_i - \tilde{\boldsymbol{\theta}}_u^\top \tilde{\boldsymbol{\theta}}_i)^2 \\ &= \frac{1}{2N|\mathcal{D}|} \sum_{u\in\mathcal{U}} (a_u \tilde{\boldsymbol{\theta}}_u^\top (\sum_{i\in\mathcal{I}} \tilde{\boldsymbol{\theta}}_i \tilde{\boldsymbol{\theta}}_i^\top) \tilde{\boldsymbol{\theta}}_u - 2a_u \tilde{\boldsymbol{\theta}}_u^\top (\sum_{i\in\mathcal{I}} \boldsymbol{\theta}_i \tilde{\boldsymbol{\theta}}_i^\top) \tilde{\boldsymbol{\theta}}_u + a_u \boldsymbol{\theta}_u^\top (\sum_{i\in\mathcal{I}} \boldsymbol{\theta}_i \boldsymbol{\theta}_i^\top) \boldsymbol{\theta}_u) \\ &= \frac{1}{2N|\mathcal{D}|} [\langle \tilde{P}_c, \tilde{Q}_c \rangle_F - 2\langle \hat{P}_c, \hat{Q}_c \rangle_F + \langle P_c, Q_c \rangle_F] \end{aligned}$$

where $A = \text{diag}(a_u)$ and $\langle \cdot, \cdot \rangle_F$ stands for the Frobenius product, with all matrices formalized as,

$$\begin{gathered} \tilde{P}_c = \tilde{P}^\top A \tilde{P}, \hat{P}_c = \tilde{P}^\top A P, P_c = P^\top A P \\ \tilde{Q}_c = \tilde{Q}^\top \tilde{Q}, \hat{Q}_c = \tilde{Q}^\top P, Q_c = Q^\top Q \end{gathered}$$

Furthermore, we handle the gradient information of the loss function, consider

$$\nabla \mathcal{L}^* = \sum_{(u,i)\in\mathcal{D}} \frac{\partial o_i^{(u)}}{\partial \theta} \frac{\partial \mathcal{L}_{ui}^+}{\partial o_i^{(u)}} + \sum_{u\in\mathcal{U}} \sum_{i\in\mathcal{I}} \frac{\partial o_i^{(u)}}{\partial \theta} \frac{\partial \mathcal{L}_{ui}^-}{\partial o_i^{(u)}} + \lambda \nabla \psi(\theta)$$



(a) MRR@10 on MovieLens

(b) MRR@10 on Electronics

(c) MRR@10 on Steam

(d) NDCG@10 on MovieLens

(e) NDCG@10 on Electronics

(f) NDCG@10 on Steam

Figure 2: Comparisons of convergence speed on all datasets.

where $\frac{\partial o_i^{(u)}}{\partial \theta} = [\boldsymbol{\theta}_i^\top \frac{\partial \boldsymbol{\theta}_u(\theta)}{\partial \theta}, \boldsymbol{\theta}_u^\top \frac{\partial \boldsymbol{\theta}_i(\theta)}{\partial \theta}]$, thus

$$\nabla \mathcal{L}^+ = \begin{bmatrix} \sum_{u\in\mathcal{U}} \sum_{i\in\mathcal{I}} (\frac{\partial \boldsymbol{\theta}_u(\theta)}{\partial \theta})^\top \boldsymbol{\theta}_i R_{ui} \\ \sum_{u\in\mathcal{U}} \sum_{i\in\mathcal{I}} (\frac{\partial \boldsymbol{\theta}_i(\theta)}{\partial \theta})^\top \boldsymbol{\theta}_u R_{ui} \end{bmatrix}$$

Suppose $\frac{\partial \boldsymbol{\theta}_u(\theta)}{\partial \theta} \in \mathbb{R}^{k\times D_u}$, $\frac{\partial \boldsymbol{\theta}_i(\theta)}{\partial \theta} \in \mathbb{R}^{k\times D_i}$, let $\boldsymbol{J}^{user} \in \mathbb{R}^{M\times D_u \times k}$, $\boldsymbol{J}^{item} \in \mathbb{R}^{N\times D_i \times K}$ be 3-dimension tensors with slices $\boldsymbol{J}^{user}_{u,:,:} = \frac{\partial \boldsymbol{\theta}_u(\theta)}{\partial \theta}$, $J^{item}_{i,:,:} = \frac{\partial \boldsymbol{\theta}_i(\theta)}{\partial \theta}$, then

$$\nabla \mathcal{L}^+(\theta) = \begin{bmatrix} \langle \boldsymbol{J}^{user}, RQ \rangle \\ \langle \boldsymbol{J}^{item}, R^\top P \rangle \end{bmatrix}$$

$$\nabla \mathcal{L}^-(\theta) = \begin{bmatrix} \langle \boldsymbol{J}^{user}, APQ_c - A\tilde{P}\hat{Q}_c \rangle \\ \langle \boldsymbol{J}^{item}, QP_c - \tilde{Q}\hat{P}_c \rangle \end{bmatrix}$$

$$\Longrightarrow \nabla \mathcal{L}_{\mathrm{RG}^2}(\theta) = -\frac{1}{|\mathcal{D}|} \begin{bmatrix} \langle \boldsymbol{J}^{user}, RQ + \frac{1}{2N}(APQ_c - A\tilde{P}\hat{Q}_c) \rangle \\ \langle \boldsymbol{J}^{item}, R^\top P + \frac{1}{2N}(QP_c - \tilde{Q}\hat{P}_c) \rangle \end{bmatrix}$$

where the inner product operation between a tensor and a matrix is given by

$$\langle \boldsymbol{J}, M \rangle = \sum_{i\in\mathcal{I}} \boldsymbol{J}_{i,:,:}(M_{i,:})^\top$$

Similarly, the second-order information of the Gauss-Newton method has the form of,

$$G\boldsymbol{d} = -\frac{1}{|\mathcal{D}|} \begin{bmatrix} \langle \boldsymbol{J}^{user}, \frac{1}{2N}(AWQ_c + AP_cH) \rangle \\ \langle \boldsymbol{J}^{item}, \frac{1}{2N}(HP_c - Q_cW) \rangle \end{bmatrix} + \lambda \nabla^2 \psi(\theta)\boldsymbol{d}$$

where

$$W_{u\cdot} = \frac{\partial \boldsymbol{\theta}_u(\theta)}{\partial \theta^u} \boldsymbol{d}^u$$

$$H_{i\cdot} = \frac{\partial \boldsymbol{\theta}_i(\theta)}{\partial \theta^i} \boldsymbol{d}^i$$

A significant advantage of our loss function is that the second-order derivative with respect to $\mathcal{L}^+$ is equal to zero, which bypasses a computationally expensive step in NewtonCG. Here we omit the detailed NewtonCG Algorithm.

Although NewtonCG is not restricted on linear models, we still perform its experiments on a MF backbone with RG^2 loss (RG^2+NewtonCG) and plot the comparisions with RG^2+ALS and other baselines. Our experiments on NewtonCG optimization are based on publicly available code by (Yuan et al., 2021) which has been accelerated by some C-language functions. The results are illustrated in Figure 2. We can observe that although NewtonCG sometimes shows competitive to ALS methods (RG^2, WRMF) on certain datasets (MovieLens), it merely maintains a high rate of convergence (which may be highly relevant to the code acceleration) but hard to converge on better performance (Electronics, Steam). This may be explained by the fact that second-order approaches still introduce a large bias to the training process, making it difficult for the loss function converging to the optimum. Another possible reason stems from the complexity of NewtonCG's parameter space, which requires careful parameter tuning compared to the ALS whose search spaces is simple and barely adjusted. This challenge also in effect hinders NewtonCG's potential to achieve higher performance.

Given the extensive variety of nonlinear encoders within the domain of recommender systems, our discussion on the aforementioned NewtonCG with our proposed RG^2 loss function still remains in exploratory stage. We acknowledge its potential and intend to proceed a more thorough exploration in our future work.