RECOMMENDATION WITH USER ACTIVE DISCLOSING WILLINGNESS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recommender system has been deployed in a large amount of real-world applications, profoundly influencing people's daily life and production. Traditional recommender models mostly collect as comprehensive as possible user behaviors for accurate preference estimation. However, considering the privacy, preference shaping and other issues, the users may not want to disclose all their behaviors for training the model. In this paper, we study a novel recommendation paradigm, where the users are allowed to indicate their "willingness" on disclosing different behaviors, and the models are optimized by trading-off the recommendation quality as well as the violation of the user "willingness". More specifically, we formulate the recommendation problem as a multiplayer game, where the action is a selection vector representing whether the items are involved into the model training. For efficiently solving this game, we design a tailored algorithm based on influence function to lower the time cost for recommendation quality exploration, and also extend it with multiple anchor selection vectors. We conduct extensive experiments to demonstrate the effectiveness of our model on balancing the recommendation quality and user disclosing willingness.

1 INTRODUCTION

For a long time, the recommender models are developed based on the key assumption of collaborative filtering (CF), where similar users in the past may also behave similarly in the future. Usually, the similarities between different users are inferred from their behaviors based on either explicit heuristics (Ricci et al., 2015) or latent embeddings (Koren et al., 2009). Thus, more comprehensive user behaviors are the basis for computing more accurate user similarities and obtaining better recommendation quality.

However, from the perspective of the users, they may not want to disclose all their behaviors for training the model due to the privacy, preference shaping and other issues. As exampled in Figure 1, in the first case, if an item X with privacy information of user A is leveraged to train the model, then the recommendation list may contain items similar to X, and people who have seen this recommendation list may easily infer the user's privacy. In the second case, the users may want to actively shape their profiles on the platform by editing their historical interactions, such that the learned model can provide more tailored recommendations. Both of the above examples suggest that while more comprehensive user behaviors may promise better user similarity estimation, in real-world scenarios, the users may not want to disclose all of them.

To consider both the recommendation quality and user disclosing willingness, we study a novel recommendation paradigm, where each user can actively specify a willingness vector to show how much she would not like her behaviors to be leveraged for training the recommender model. We formulate our problem as a multiplayer game. Each player corresponds to a user, whose action is a selection vector indicating a subset of the user interacted items[1]. The recommender model is optimized based on all the items selected by different players. The reward is designed to improve the recommendation quality and simultaneously follow the users' willingness. To solve this game, the major challenge lies in how to efficiently derive the recommendation quality for different selection vectors.

[1]The interaction with an item can be the user behavior such as click and purchase.

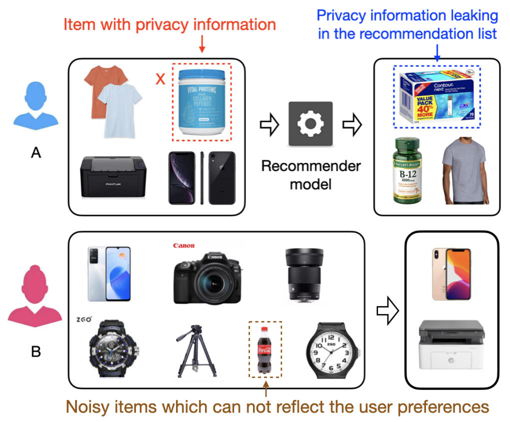
Item with privacy information
Privacy information leaking in the recommendation list
A
Recommender model
B
Noisy items which can not reflect the user preferences

Figure 1: Motivating examples. In the first case, the user does not want to disclose her interaction with the medicine. In the second case, the user would like to remove the items which cannot reflect her preference on the digital items.

To overcome this challenge, we firstly set an anchor selection vector, based on which we train the recommender model. Then, for different actions, we compute the recommendation quality based on the influence function (Koh & Liang, 2017) without retraining the model. We name our method as **i**nfluence **f**unction based **r**ecommendation **q**uality **e**xploration (called IFRQE for short). To achieve more accurate quality approximation, we further extend the above method with multiple anchor selection vectors. We provide theoretical analysis on the designed model in terms of the convergence rate and the benefit of increasing the number of anchor selection vectors. Extensive experiments based on both synthetic and real-world datasets are conducted to demonstrate the effectiveness of our model.

Notably, there are a few previous works on federated recommendation (Yang et al., 2020), recommendation unlearning (Chen et al., 2022a) and controllable recommendation. While these studies share some similarities with our work, they neither allow user active disclosing behaviors nor involve user willingness into the optimization target, which makes them fundamentally differ from our idea. Chen et al. (2022b) is a recently proposed model for trading-off the user privacy and recommendation quality. Comparing with this work, we improve it by allowing more flexible user willingness, and use influence function to enhance the optimization efficiency. In a summary, the main contributions of this paper can be concluded as follows: (1) We propose a novel recommendation paradigm, where the users can explicitly indicate their disclosing willingness, and the model needs to trade off the recommendation quality and user willingness. (2) To solve the above problem, we formulate the recommendation task as a multiplayer game, and design two influence function based algorithms to solve the game efficiently. (3) Extensive experiments are conducted to verify the superiorities of our model based on both synthetic and real-world datasets.

2 Problem Formulation

Suppose we have a user set $\mathcal{U}$ and an item set $\mathcal{V}$. Let $\mathcal{O}^u$ be the set of items interacted by user u, and we separate it into a training set $\mathcal{S}^u$, a validation set $\mathcal{T}^u$ and a testing set $\mathcal{D}^u$. In our problem, each user u can specify a disclosing willingness vector $\boldsymbol{\beta}^u = \{\beta^u_1, \beta^u_2, ..., \beta^u_{|\mathcal{S}^u|}\}$, where $\beta^u_k \in [0, 1]$, $\forall k \in [1, |\mathcal{S}^u|]$. Suppose $\mathcal{S}^u = \{s^u_1, s^u_2, ..., s^u_{|\mathcal{S}^u|}\}$, then the larger β^u_k is, the more the user do not want s^u_k to join into the model training. Our task is to select a subset from $\mathcal{S} = \{\mathcal{S}^u | u \in \mathcal{U}\}$ to train the recommender model, such that the learned model can not only provide acceptable recommendation quality, but also can follow the user willingness. Intuitively, if we select more items to train the model, then the user can be better understood, and receive higher recommendation quality. However, the user willingness is more likely to be violated, since the items with larger β^u_k may be selected. Conversely, if we let a small amount of items to train the model, although the user willingness can be better satisfied, the recommendation quality can be lowered. The key of our task is to learn an optimal strategy, which can well balance the recommendation quality and user willingness.

To solve this task, we regard each user as a player, and formulate the recommendation problem as a multiplayer game. For each user u, the action is a binary selection vector $\boldsymbol{o}^u = \{o^u_1, o^u_2, ..., o^u_{|\mathcal{S}^u|}\}$, where $o^u_k = 1$ means s^u_k is selected to train the model, otherwise $o^u_k = 0$. The reward for user u is designed as follows:

$$\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u}) = -L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o}^u, \boldsymbol{o}^{-u})) - \lambda \sum_{k=1}^{|\mathcal{S}^u|} \boldsymbol{o}^u_k \beta^u_k \quad (1)$$

where $\boldsymbol{o}^{-u} = \{\boldsymbol{o}^1, ..., \boldsymbol{o}^{u-1}, \boldsymbol{o}^{u+1}, ..., \boldsymbol{o}^N\}$ is the joint selection vectors of all the user except u. f is a recommender model. $\hat{\boldsymbol{\theta}}(\boldsymbol{o}^u, \boldsymbol{o}^{-u})$ are the parameters of f learned based on the training samples selected by $\{\boldsymbol{o}^u, \boldsymbol{o}^{-u}\}$. $-L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o}^u, \boldsymbol{o}^{-u}))$ is negative validation loss based on $\hat{\boldsymbol{\theta}}(\boldsymbol{o}^u, \boldsymbol{o}^{-u})$, which is leveraged to measure the recommendation quality. The second term evaluates the violation of the user willingness. If the items that the user does not want to disclose (*e.g.*, β_k^u is large) are selected to train the model, then $o_k^u \beta_k^u$ is large, which lowers the reward. λ is a pre-defined balancing parameter. Let the strategy of each player be $\boldsymbol{\alpha}^u = [\boldsymbol{\alpha}^u_{\boldsymbol{o}^u}] \in \triangle(2^{|\mathcal{S}^u|})$, which is a discrete distribution, and $\boldsymbol{\alpha}^u_{\boldsymbol{o}^u}$ is the probability of leveraging the items indicated by $\boldsymbol{o}^u$ to train the model. For example, suppose the training set is $\mathcal{S}^u = \{0, 1, 2\}$, then $\boldsymbol{\alpha}^u$ is a distribution defined on $\{\{0,0,0\}, \{0,0,1\}, \{0,1,0\}, \{1,0,0\}, \{0,1,1\}, \{1,0,1\}, \{1,1,0\}, \{1,1,1\}\}$, and $\boldsymbol{\alpha}^u_{\{0,1,1\}}$ is the probability of using items $\{1, 2\}$ to train the model.

We aim to learn the optimal joint strategy $\boldsymbol{\alpha}^* = \{\boldsymbol{\alpha}^{1*}, \boldsymbol{\alpha}^{2*}, ..., \boldsymbol{\alpha}^{N*}\}$, such that the corresponding expected reward of each user u is the largest when the other user strategies are fixed, that is:

$$z_u(\boldsymbol{\alpha}^{u*}, \boldsymbol{\alpha}^{-u*}) \geq z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u*}), \ \forall u \in [1, 2, ..., N], \ \forall \boldsymbol{\alpha}^u \in \triangle(2^{|\mathcal{S}^u|}) \tag{2}$$

where $\boldsymbol{\alpha}^{-u} = \{\boldsymbol{\alpha}^1, ..., \boldsymbol{\alpha}^{u-1}, \boldsymbol{\alpha}^{u+1}, ..., \boldsymbol{\alpha}^N\}$ is the joint strategy of all the users except u. $z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u})$ is the expected reward for user u under $\{\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u}\}$, that is, $z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u}) = E_{\boldsymbol{o}^u \sim \boldsymbol{\alpha}^u, \ \boldsymbol{o}^{-u} \sim \boldsymbol{\alpha}^{-u}}[\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u})]$. After obtaining the optimal strategy $\boldsymbol{\alpha}^*$, we firstly sample the selection vectors $\boldsymbol{o} = \{\boldsymbol{o}^u, \boldsymbol{o}^{-u}\}$ from $\boldsymbol{\alpha}^*$. Then, the training samples are generated by filtering $\mathcal{S}$ with $\boldsymbol{o}$, which are leveraged to optimize the final recommender model.

Remark. (i) The above formulation is from two aspects: on the one hand, β_k^u can be defined based on the privacy concerns, the purpose of reducing the storage/computation burden or any other reasons that the users do not want their behaviors to be disclosed. On the other hand, we do not impose constraints on f and L, thus they can be any recommender model and loss function, which are compatible with most of the previous work. (ii) In our formulation, we do not assume that the system is adverse. The recommender model is developed to better serve the users by following their disclosing willingness and achieving acceptable recommendation quality. (iii) While the action space looks extremely large, which may lower the training efficiency, there are many strategies to alleviate this problem. For example, one can assign the same o_k^u for the items in the same category, or only allow the users to indicate their willingness on the most important parts of their interactions. In addition, we can also initialize $\boldsymbol{\alpha}$ with an informative prior, and search the optimal solution around this prior to speed up the training process.

3 The IFRQE Model

3.1 The basic model

To solve the game defined based on (1) and (2), one has to derive the validation loss $L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o}))$ for different $\boldsymbol{o}$'s. A straightforward method is firstly training f for each $\boldsymbol{o}$ to obtain the corresponding parameter $\hat{\boldsymbol{\theta}}(\boldsymbol{o})$, and then the loss is computed based on the validation set $\mathcal{T}^u$ and $\hat{\boldsymbol{\theta}}(\boldsymbol{o})$. However, such method is infeasible, since repeatedly training the recommender model is quite time-consuming.

Fortunately, the previous studies on influence function (Koh & Liang, 2017) may shed some lights on approximating the validation loss without retraining the model. In specific, we first define an anchor selection vector $\widetilde{\boldsymbol{o}} = \{\widetilde{\boldsymbol{o}}^1, ..., \widetilde{\boldsymbol{o}}^N\}$, and then train the recommender model f based on $\widetilde{\boldsymbol{o}}$ to obtain the parameters $\widetilde{\boldsymbol{\theta}}$. At last, for a candidate selection vector $\boldsymbol{o}$, we approximate $L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o}))$ via the following theory[2].

Theorem 1. *Given the model parameters $\boldsymbol{\theta}$, we define the validation loss by $L_f(\mathcal{T}^u, \boldsymbol{\theta}) = \sum_{y \in \mathcal{T}^u} l_f(y, \boldsymbol{\theta})$, where $l_f(y, \boldsymbol{\theta})$ is the loss of sample y based on $\boldsymbol{\theta}$. Suppose $| \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})| \leq B$ and $\sum_v \sum_{k=1}^{|\mathcal{S}^v|} (\widetilde{o}_k^v - o_k^v) B$ is a small value, then the validation loss for $\boldsymbol{o}$ is:*

$$L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o})) \approx L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}) - \frac{1}{Z} \sum_{y \in \mathcal{T}^u} \sum_{v \in \mathcal{U}} \sum_{k=1}^{|\mathcal{S}^v|} o_k^v \bigtriangledown l_f(y, \widetilde{\boldsymbol{\theta}}) H_{\widetilde{\boldsymbol{\theta}}}^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) \tag{3}$$

where $\widetilde{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \frac{1}{Z} \sum_{v \in \mathcal{U}} \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v l_f(s_k^v, \boldsymbol{\theta})$ are the parameters learned based on the anchor selection vector. Z is the total number of training samples. $\bigtriangledown l_f(\cdot)$ is the gradient of a sample loss. $H_{\widetilde{\boldsymbol{\theta}}} = \frac{1}{Z} \sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})$ is the Hessian matrix of the training loss.

[2]We present the proofs of all the theories throughout this paper in the Appendix.

Algorithm 1: Learning algorithm for $\boldsymbol{\alpha}_m^u$

```
1 Indicate the learning rate $\gamma$.
2 Let $\boldsymbol{\alpha}^{-u} = \boldsymbol{\alpha}_{m-1}^{-u}$ and $\boldsymbol{\alpha}_1^u = \boldsymbol{\alpha}_{m-1}^u$.
3 for l in [1, L] do
4     Sample $\boldsymbol{o}^u, \boldsymbol{o}^{-u}$ according to $\boldsymbol{\alpha}_l^u, \boldsymbol{\alpha}^{-u}$.
5     Compute the gradient $\hat{g}_{\boldsymbol{o}^u}$ based on (9).
6     Let $[\hat{\boldsymbol{g}}]_s = 1(s = \boldsymbol{o}^u)\hat{g}_{\boldsymbol{o}^u}$.
7     Update $\boldsymbol{\alpha}_{l+1}^u = \Pi_\triangle[\boldsymbol{\alpha}_l^u + \gamma\hat{\boldsymbol{g}}]$, where $\Pi_\triangle$ means projecting a vector into a simplex.
8 end
9 Return $\boldsymbol{\alpha}^u = \frac{1}{L}\sum_{l=1}^{L}\boldsymbol{\alpha}_l^u$.
```

Based on this theory, we can compute $L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o}))$ without retraining the model via $\hat{\boldsymbol{\theta}}(\boldsymbol{o}) = \arg\min_{\boldsymbol{\theta}} \frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} o_k^{t,v} l_f(s_k^v, \boldsymbol{\theta})$. By bringing (3) into (1), $\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u})$ can be written as:

$$\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u}) = -L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}) + \frac{1}{Z}\sum_{y\in\mathcal{T}^u}\sum_{v\in\mathcal{U}}\sum_{k=1}^{|\mathcal{S}^v|} o_k^v \bigtriangledown l_f(y, \widetilde{\boldsymbol{\theta}}) H_{\widetilde{\boldsymbol{\theta}}}^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) - \lambda\sum_{k=1}^{|\mathcal{S}^u|} o_k^u \beta_k^u. \tag{4}$$

Accordingly, the expected reward $z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u})$ is computed via the follow theory.

Theorem 2. *Let $\boldsymbol{g}_y^v = [g(s_1^v, y), g(s_2^v, y), ..., g(s_{|\mathcal{S}^v|}^v, y)]$, where $g(s_k^v, y) = \bigtriangledown_\theta l_f(y, \widetilde{\boldsymbol{\theta}})^T H_{\widetilde{\boldsymbol{\theta}}}^{-1} \bigtriangledown_\theta l_f(s_k^v, \widetilde{\boldsymbol{\theta}})$, then: $z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u}) = E_{\boldsymbol{o}}[\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u})] = \sum_{\boldsymbol{o}^u} \alpha_{\boldsymbol{o}^u}^u [(\sum_{y\in\mathcal{T}^u} \frac{\boldsymbol{g}_y^u}{Z} - \lambda\boldsymbol{\beta}^u)^T \boldsymbol{o}^u] + C$, where $C = -L(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}) + \sum_{v\neq u} E_{\boldsymbol{o}}[(\boldsymbol{o}^v)^T \sum_{y\in\mathcal{T}^u} \frac{\boldsymbol{g}_y^v}{Z}]$ is irrelevant with $\boldsymbol{\alpha}^u$.*

To solve objective (2), we need to find an optimal $\boldsymbol{\alpha}^u$ for the following optimization problem:

$$\max_{\boldsymbol{\alpha}^u} \sum_{\boldsymbol{o}^u} \alpha_{\boldsymbol{o}^u}^u A(\boldsymbol{o}^u) \;\; s.t. \; \sum_{\boldsymbol{o}^u} \alpha_{\boldsymbol{o}^u}^u = 1, \;\; \alpha_{\boldsymbol{o}^u}^u \geq 0 \tag{5}$$

where we denote $(\sum_{y\in\mathcal{T}^u} \frac{\boldsymbol{g}_y^u}{N} - \lambda\boldsymbol{\beta}^u)^T \boldsymbol{o}^u$ by $A(\boldsymbol{o}^u)$. We present the complete training process of this model in the appendix.

3.2 The improved model with multiple anchor selection vectors

The key of the above method lies in the accurate approximation of $L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o}))$. However, if the current selection vector $\boldsymbol{o}$ is too much different from the anchor vector $\widetilde{\boldsymbol{o}}$, then the assumption "$\sum_v \sum_{k=1}^{|\mathcal{S}^v|} (\widetilde{o}_k^v - o_k^v) B$ is a small value in theory 1 may not hold, which can lead to larger approximation errors. To alleviate this problem, we propose to set multiple anchor vectors $\{\widetilde{\boldsymbol{o}}^1, \widetilde{\boldsymbol{o}}^2, ...\widetilde{\boldsymbol{o}}^T\}$, where $\widetilde{\boldsymbol{o}}^t = \{\widetilde{\boldsymbol{o}}^{t,1}, ..., \widetilde{\boldsymbol{o}}^{t,N}\}$ is the tth anchor vector and $\widetilde{o}_k^{t,v}$ is the kth element of $\widetilde{\boldsymbol{o}}^{t,v}$. For approximating $L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o}))$, we select the anchor vector nearest to $\boldsymbol{o}$, where we have the following theory.

Theorem 3. *For a candidate selection vector $\boldsymbol{o}$, suppose $t = \arg\min_{i\in[1,T]} \sum_{v=1}^{N} D(\widetilde{\boldsymbol{o}}^{i,v}, \boldsymbol{o}^v)$, where D is the hamming distance counting the number of different bits between two vectors, and $\widetilde{\boldsymbol{\theta}}^t = \arg\min_{\boldsymbol{\theta}} \frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^{t,v} l_f(s_k^v, \boldsymbol{\theta})$, then*

$$L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o})) \approx L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}^t) - \frac{1}{Z}\sum_{y\in\mathcal{T}^u}\sum_{v\in\mathcal{U}}\sum_{k=1}^{|\mathcal{S}^v|} o_k^v \bigtriangledown l_f(y, \widetilde{\boldsymbol{\theta}}^t) H_{\widetilde{\boldsymbol{\theta}}^t}^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}^t) \tag{6}$$

where $H_{\widetilde{\boldsymbol{\theta}}^t} = \frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^{t,v} \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}}^t)$.

Based on (6), the expected reward can be derived based on the following theory.

Theorem 4. *Let $A_t = \{\boldsymbol{o} | \sum_{v=1}^{N} D(\widetilde{\boldsymbol{o}}^{t,v}, \boldsymbol{o}^v) \leq \sum_{v=1}^{N} D(\widetilde{\boldsymbol{o}}^{t',v}, \boldsymbol{o}^v), \;\; \forall t' \neq t\}$. $g(s_k^v, y, t) = \bigtriangledown l_f(y, \widetilde{\boldsymbol{\theta}}^t) H_{\widetilde{\boldsymbol{\theta}}^t}^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}^t)$. $\boldsymbol{g}_y^{t,v} = [g(s_1^v, y, t), g(s_2^v, y, t), ...g(s_{|\mathcal{S}^v|}^v, y, t)]$ and $\boldsymbol{g}^{t,v} = \sum_{y\in\mathcal{T}^u} \boldsymbol{g}_y^{t,v}$. Suppose we define $\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u}, t) = -L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}^t) + \frac{1}{Z}\sum_{y\in\mathcal{T}^u}\sum_{v\in\mathcal{U}}\sum_{k=1}^{|\mathcal{S}^v|} o_k^v \bigtriangledown l_f(y, \widetilde{\boldsymbol{\theta}}^t) H_{\widetilde{\boldsymbol{\theta}}^t}^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}^t) - \lambda\sum_{k=1}^{|\mathcal{S}^u|} o_k^u \beta_k^u$, Then*

Algorithm 2: Learning Algorithm with Multiple θ_t

```
1  Initialize $\{\boldsymbol{\alpha}^1, \boldsymbol{\alpha}^2, ..., \boldsymbol{\alpha}^N\}$ and let $\boldsymbol{\alpha}_0^u = \boldsymbol{\alpha}^u$ $(u \in [1, N])$.
2  Indicate the max iteration number M and threshold $\kappa$.
3  Indicate the anchor vectors $\{\widetilde{\boldsymbol{o}}^1, \widetilde{\boldsymbol{o}}^2, ...\widetilde{\boldsymbol{o}}^T\}$.
4  Obtain the model parameters $\widetilde{\boldsymbol{\theta}}^t$ for each $\widetilde{\boldsymbol{o}}^t$.
5  for m in [1, M] do
6      for u in [1, N] do
7          Let $\boldsymbol{\alpha}_{m-1}^{-u} = \{\boldsymbol{\alpha}_{m-1}^1, ..., \boldsymbol{\alpha}_{m-1}^{u-1}, \boldsymbol{\alpha}_{m-1}^{u+1}, ..., \boldsymbol{\alpha}_{m-1}^N\}$.
8          Obtain $\boldsymbol{\alpha}_m^u$ by inputting $\boldsymbol{\alpha}_{m-1}^{-u}$ and $\boldsymbol{\alpha}_{m-1}^u$ into Algorithm 1.
9      end
10     if $|\boldsymbol{\alpha}_m^u - \boldsymbol{\alpha}_{m-1}^u| < \kappa$ , $\forall u \in [1, N]$ then
11         Break.
12     end
13 end
14 Output $\boldsymbol{\alpha}^{u*} = \boldsymbol{\alpha}_m^u$ $(u \in [1, N])$.
```

$$z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u}) = E_{\boldsymbol{o}}[\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u})] = E_{\boldsymbol{o}}[\sum_{t=1}^{T} 1(\boldsymbol{o} \in A_t)\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u}, t)]$$
$$= \sum_{t=1}^{T}\sum_{\boldsymbol{o}} 1(\boldsymbol{o} \in A_t)\alpha_{\boldsymbol{o}^u}^u \alpha_{\boldsymbol{o}^{-u}}^{-u}\{-L(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}^t) + \frac{1}{Z}\sum_{v \neq u}(\boldsymbol{o}^v)^T \boldsymbol{g}^{t,v} + \frac{1}{Z}(\boldsymbol{o}^u)^T \boldsymbol{g}^{t,u} - \lambda(\boldsymbol{o}^u)^T \boldsymbol{\beta}^u\} \tag{7}$$

where $1(c) = 1$ *if the condition c is true, otherwise* $1(c) = 0$. $\alpha_{\boldsymbol{o}^{-u}}^{-u} = \alpha_{\boldsymbol{o}^1}^1 ... \alpha_{\boldsymbol{o}^{u-1}}^{(u-1)} \alpha_{\boldsymbol{o}^{u+1}}^{(u+1)} ... \alpha_{\boldsymbol{o}^N}^N$.

Then we derive $\boldsymbol{\alpha}^u$ by solving the following optimization problem:

$$\max_{\alpha^u} \sum_{t=1}^{T}\sum_{\boldsymbol{o}} 1(\boldsymbol{o} \in A_t)\alpha_{\boldsymbol{o}^u}^u \alpha_{\boldsymbol{o}^{-u}}^{-u} B(\boldsymbol{o}^u, \boldsymbol{o}^{-u}, t) \quad s.t. \sum_{\boldsymbol{o}^u} \alpha^u(\boldsymbol{o}^u) = 1, \;\; \alpha^u(\boldsymbol{o}^u) > 0 \tag{8}$$

where $B(\boldsymbol{o}^u, \boldsymbol{o}^{-u}, t) = -L(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}^t) + \frac{1}{Z}\sum_{v \neq u}(\boldsymbol{o}^v)^T \boldsymbol{g}^{t,v} + \frac{1}{Z}(\boldsymbol{o}^u)^T \boldsymbol{g}^{t,u} - \lambda(\boldsymbol{o}^u)^T \boldsymbol{\beta}^u$.

Since it is hard to efficiently obtain a closed-form solution for $\boldsymbol{\alpha}^u$, we learn it based on the projected gradient descent method (Calamai & Moré, 1987). More specifically, the gradient of $z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u})$ w.r.t $\alpha_{\boldsymbol{o}^u}^u$ is:

$$\frac{\partial z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u})}{\partial \alpha_{\boldsymbol{o}^u}^u} \overset{\text{def}}{=} g_{\boldsymbol{o}^u} = E_{\boldsymbol{o}^{-u}}[\sum_{t=1}^{T} B(\boldsymbol{o}^u, \boldsymbol{o}^{-u}, t)\mathbf{1}(\boldsymbol{o}^{-u} \in S_t(\boldsymbol{o}^u))]. \tag{9}$$

where $S_t(\boldsymbol{o}^u) = \{\boldsymbol{o}^{-u} | [\boldsymbol{o}^u, \boldsymbol{o}^{-u}] \in A_t\}$. Let $\boldsymbol{g} = [g_{\boldsymbol{o}^u}]_{\boldsymbol{o}^u}$ be the gradient of $z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u})$ w.r.t $\boldsymbol{\alpha}^u$, which is a $2^{|\mathcal{S}^u|}$ dimensional vector. $\hat{g}_{\boldsymbol{o}^u}$ is the stochastic gradient by sampling $\boldsymbol{o}^{-u}$ from $\boldsymbol{\alpha}^{-u}$. Then the complete training process can be seen in Algorithm 2.

3.3 Theoretical analysis

In this section, we provide theoretical analysis on the convergence of our algorithm and also present the advantage of multiple anchor selection vectors comparing with the single one in terms of the validation loss approximation. For the convergence analysis, we have the following theory:

Theorem 5. *Based on the definition of* $\boldsymbol{g}$, *we have* $z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u}) = (\boldsymbol{\alpha}^{\boldsymbol{u}})^T \boldsymbol{g}$. *Suppose* $\hat{\boldsymbol{g}}$ *is an unbiased estimation of* $\boldsymbol{g}$, *and* $||\hat{\boldsymbol{g}}(\boldsymbol{\alpha}^u)||_2^2 \leq G$, *then the solution obtained from Algorithm 1 is larger than the optimal solution minus a bounded value, that is:* $E[z_u(\hat{\boldsymbol{\alpha}}^u, \boldsymbol{\alpha}^{-u})] \geq \max_{\boldsymbol{\alpha}^u} E[z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u})] - (\frac{1}{L\gamma} + \gamma^2 G^2)$, *where* $\hat{\boldsymbol{\alpha}}^u$ *is obtained based on Algorithm 1.*

This theory provides foundations for Algorithm 1, which ensures that the learned strategies are nearly optimal. For the advantage of using multiple anchor selection vectors, we have the following theory:

Theorem 6. *For two sets of anchor selection vectors* $\boldsymbol{P} = \{\widetilde{\boldsymbol{o}}_P^1, \widetilde{\boldsymbol{o}}_P^2, ...\widetilde{\boldsymbol{o}}_P^{T_P}\}$ *and* $\boldsymbol{Q} = \{\widetilde{\boldsymbol{o}}_Q^1, \widetilde{\boldsymbol{o}}_Q^2, ...\widetilde{\boldsymbol{o}}_Q^{T_Q}\}$*, suppose* $A_t^P = \{\boldsymbol{o} | \sum_{v=1}^N D(\widetilde{\boldsymbol{o}}^{t,v}, \boldsymbol{o}^v) \leq \sum_{v=1}^N D(\widetilde{\boldsymbol{o}}^{t',v}, \boldsymbol{o}^v), \forall \widetilde{\boldsymbol{o}}^{t'} \in \boldsymbol{P}\}$ *and* $A_t^Q = \{\boldsymbol{o} | \sum_{v=1}^N D(\widetilde{\boldsymbol{o}}^{t,v}, \boldsymbol{o}^v) \leq \sum_{v=1}^N D(\widetilde{\boldsymbol{o}}^{t',v}, \boldsymbol{o}^v), \forall \widetilde{\boldsymbol{o}}^{t'} \in \boldsymbol{Q}\}$*. Based on theory 1, we consider the following upper bounds of the approximation error for the validation loss:*

$$err(\boldsymbol{P}) = \sum_{t=1}^{T_P} \sum_{\boldsymbol{o} \in A_t^P} [\sum_v D(\widetilde{\boldsymbol{o}}^{t,v}, \boldsymbol{o}^v)]B \;\; and \;\; err(\boldsymbol{Q}) = \sum_{t=1}^{T_Q} \sum_{\boldsymbol{o} \in A_t^Q} [\sum_v D(\widetilde{\boldsymbol{o}}^{t,v}, \boldsymbol{o}^v)]B, \tag{10}$$

We have if $\boldsymbol{P} \subseteq \boldsymbol{Q}$*, then* $err(\boldsymbol{P}) \geq err(\boldsymbol{Q})$*.*

This theory suggests if the multiple anchor vectors in section 3.2 are selected to include the single one in section 3.1, then the approximation error upper bound of the validation loss can be reduced. However, we should note that introducing more anchor vectors means more times for retraining the recommender model, which lowers the efficiency. Actually, the improved model provides us with an opportunity to balance the approximation accuracy and model efficiency.

4 RELATED WORK

Recommender system is a rapidly developing research field, with a large amount of models developed each year (Ricci et al., 2015). Early recommender models are mostly based on the matrix factorization (Koren et al., 2009). And then, with the prospering of the deep learning technique, a large amount of deep recommender models (Zhang et al., 2019) have been proposed, such as the sequential (Kang & McAuley, 2018; Tang & Wang, 2018; Sun et al., 2019) or graph (Wang et al., 2019; He et al., 2020; Fan et al., 2019) based algorithms. These models mostly leverage all the user behaviors to train the model, no matter whether the users would like to disclose them. To protect the user privacy, many federated recommender models (Lin et al., 2020; Wu et al., 2021; Yang et al., 2020; Chen et al., 2020) have been proposed. These studies aim to move the training of the model from the server to the clients, so that the system cannot access the raw user behaviors, and the user privacy is protected. While our model can also be used to protect the privacy, we do not assume that the server is adverse, but aim to ensure that the privacy information does not leak from the recommendation results. Another topic related to our work is the recommendation unlearning (Chen et al., 2022a; Li et al., 2022), which aims to remove the influences of many user-item interactions from the trained recommender model. However, these models do not capture the user active disclosing behaviors, and the user willingness is also not incorporated into the optimization targets. Many recent studies propose to design controllable recommendation (Wang et al., 2022; Parra & Brusilovsky, 2015), but they mainly aim to break the filter bubbles instead of following the indicated user willingness. Game theory based recommender models (Ben-Porat & Tennenholtz, 2018; Xu et al., 2018; Halkidi & Koutsopoulos, 2011) have also been investigated before. For example, Ben-Porat & Tennenholtz (2018) formulates the recommendation problem via a multiplayer game from the content provider perspective. Xu et al. (2018); Halkidi & Koutsopoulos (2011) study the incentive mechanisms to encourage user rating behaviors and save the user cost simultaneously. While these models share some similarities with our work, we build the game from the user side and do not limit our framework to the rating behaviors. In addition, we leverage the influence function to improve the efficiency of recommendation quality estimation, which has not been explored before.

5 EXPERIMENTS

5.1 EXPERIMENT SETUP

Datasets. We base our experiments on both synthetic and real-world datasets. For the synthetic dataset, we generated it based on a well known recommendation simulator called RecSim[3]. In specific, we generate 1000 users and 1000 items, where four features are simulated to profile each user or item. For a user-item pair, suppose the user and item features are $\boldsymbol{e}_u \in \mathbb{R}^4$ and $\boldsymbol{e}_v \in \mathbb{R}^4$, respectively, then the interaction is generated according to $\mathbf{1}(\sigma(\boldsymbol{e}_u^T \boldsymbol{e}_v) \geq \eta)$, where $\mathbf{1}(\cdot)$ is the indicator function and η is a threshold for setting the hardness of generating the interaction. For the disclosing willingness of

[3]https://github.com/google-research/recsim

user u on item v, we simulate it by $\beta_v^u = \sigma(s_u g(\boldsymbol{e}_u, \boldsymbol{e}_v) + b_u)$, where s_u is sampled from a uniform distribution in the range of $[a_1, a_2]$, controlling the sensitivity of the user disclosing willingness. Smaller s_u means the user has similar willingness on disclosing different behaviors, while larger s_u means the user have more diverse willingness. b_u is sampled from a Gaussian distribution $\mathcal{N}(0, a_3)$, indicating the average level of the user disclosing willingness. g is a one-layer fully connected neural network with randomly assigned parameters. For the real-world experiments, we evaluate different models based three public available datasets from different domains, including Diginetica[4], Steam[5] and Amazon Video[6]. More detailed statistics of these datasets are presented in the Appendix.

Baselines. To demonstrate the generality of our idea, we implement the base model f with the following recommender algorithms[7]: **MF** Koren et al. (2009) is the traditional matrix factorization method, which has been used as a baseline in a large amount of previous work. **NeuMF** He et al. (2017) is a well known neural collaborative filtering model, which aims to capture the non-linear correlations between the users and items. **LightGCN** He et al. (2020) is a graph-based recommender model, which highlights the importance of the structure information in collaborative filtering. **DIN** Zhou et al. (2018) is a sequential recommender model, and **CDAE** Wu et al. (2016) is an auto-encoder based recommender model. For each base model, we compare our framework with the following methods, for the first one, the user behaviors are randomly disclosed (*i.e.*, the selection vector $\boldsymbol{o}$ is randomly sampled). We call this method as **Random**. For the second one, we directly set $o_u^k = 0$, if β_u^k is larger than 0.5. We call this method as **Threshold**. We also compare our model with **Proactive** Chen et al. (2022b), which is a model considering both of the recommendation accuracy and user privacy, and the objective is the same as our frameworks'. The model proposed in section 3.1 is named as **IFRQE**, and the improved version in section 3.2 is called **IFRQE++**.

Implementation Details. For each user, we leverage 20% and 10% of all her interactions as the testing and validation sets, and the others are left for model training. For all the models, we use binary cross entropy as the loss function. To evaluate the recommendation quality, we use $\boldsymbol{F_1}$ (Chicco & Jurman, 2020) as the metric, and five items are recommended to compare with the ground truth. To evaluate whether the model can follow the user willingness, we compute the overall willingness violation by $wv = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\sum_{k=1}^{|\mathcal{S}^u|} o_k^u \beta_k^u$, and a model is more satisfied if its wv is smaller. To evaluate the capability of balancing the recommendation quality and user willingness, we compare different models based on the **reward** of all the users according to equation (1). It should be noted that our framework and the baselines are optimized with the same reward, and we experiment with different λ's to demonstrate the superiority of our framework. The anchor selection vectors in IFRQE and IFRQE++ are both randomly selected[8]. The model hyper parameters are determined by grid search. For the real world datasets, we specify the disclosing willingness vector of each user in a random manner. To reduce the randomness, we repeat each experiment for ten times, and report the average performance as well as the standard error. More settings can be found in the Appendix.

5.2 Overall Comparison

The overall comparison results can be seen in Table 1, where we can see: from the perspective of following user willingness, our model can usually perform better than the base model and heuristic methods, which is not surprising since the base model leverages all the items for training, and the heuristic methods consider the user willingness in too coarse manners. From the recommendation quality perspective, our models do not sacrifice the performance too much. On average, the performance is dropped by about 8.02% from the base model. Interestingly, in some cases, like on the Diginetica and Amazon Video datasets with MF as the base model, the performances are even improved. We argue that such observation is because many items with higher β_k^u are exactly not important for the prediction of the items in the testing set. Thus removing them can not only lead to higher reward via following more user willingness, but also can enhance the performance by focusing on more informative training samples. From the reward perspective, our model can achieve the best performances on most datasets and base models, which demonstrate the effectiveness and generality of our idea on balancing the recommendation quality and user willingness. On average,

[4]https://darel13712.github.io/rs_datasets/Datasets/diginetica/

[5]https://steam.internet.byu.edu/

[6]https://jmcauley.ucsd.edu/data/amazon/

[7]Experiments with more other base models can be found in the appendix.

[8]We ensure that the anchor selection vectors of IFRQE++ include the one of IFRQE

Table 1: Overall comparison between different models. We use bold fonts to label the best performance for each dataset, evaluation metric and base model. "()" indicates the standard error. The results of F_1 are percentage values with "%" omitted. For the metrics, ↑ means the larger the better, while ↓ means the lower the better. The performance improvements of our model against the baselines are significant under paired t-test.

Dataset	Simulation			Diginetica			Steam			Amazon Video		
Metric	$F_1\uparrow$	$wv\downarrow$	reward↑	$F_1\uparrow$	$wv\downarrow$	reward↑	$F_1\uparrow$	$wv\downarrow$	reward↑	$F_1\uparrow$	$wv\downarrow$	reward↑
MF	**1.75**$_{(.022)}$	2.12$_{(.007)}$	-2.25$_{(.032)}$	4.72$_{(.044)}$	2.01$_{(.053)}$	-2.65$_{(.072)}$	**10.4**$_{(.043)}$	2.62$_{(.014)}$	-2.99$_{(.093)}$	1.70$_{(.012)}$	2.36$_{(.021)}$	-2.43$_{(.022)}$
w/ Random	1.73$_{(.031)}$	2.01$_{(.006)}$	-2.13$_{(.014)}$	4.18$_{(.014)}$	1.96$_{(.013)}$	-2.04$_{(.006)}$	9.40$_{(.009)}$	2.61$_{(.017)}$	-2.82$_{(.015)}$	1.53$_{(.011)}$	2.24$_{(.022)}$	-2.30$_{(.023)}$
w/ Threshold	1.70$_{(.036)}$	2.01$_{(.032)}$	-2.20$_{(.021)}$	4.70$_{(.012)}$	1.97$_{(.011)}$	-2.06$_{(.040)}$	9.25$_{(.048)}$	2.35$_{(.023)}$	-2.57$_{(.017)}$	1.50$_{(.026)}$	2.12$_{(.027)}$	-2.20$_{(.021)}$
w/ Proactive	1.23$_{(.025)}$	**1.56**$_{(.013)}$	-2.25$_{(.007)}$	2.21$_{(.018)}$	**1.30**$_{(.024)}$	-1.99$_{(.016)}$	9.93$_{(.019)}$	**2.08**$_{(.022)}$	-2.43$_{(.025)}$	1.55$_{(.035)}$	1.94$_{(.026)}$	-2.07$_{(.005)}$
w/ IFRQE	1.50$_{(.015)}$	1.97$_{(.003)}$	-2.09$_{(.017)}$	**5.23**$_{(.008)}$	2.01$_{(.014)}$	-2.18$_{(.006)}$	**10.4**$_{(.012)}$	2.14$_{(.012)}$	-2.43$_{(.015)}$	1.57$_{(.015)}$	**1.70**$_{(.016)}$	-2.10$_{(.005)}$
w/ IFRQE++	1.63$_{(.007)}$	1.82$_{(.003)}$	**-1.92**$_{(.012)}$	4.02$_{(.011)}$	1.79$_{(.019)}$	**-1.92**$_{(.022)}$	9.83$_{(.017)}$	2.13$_{(.032)}$	**-2.42**$_{(.014)}$	**1.72**$_{(.012)}$	1.87$_{(.032)}$	**-1.98**$_{(.056)}$
NeuMF	0.87$_{(.017)}$	2.13$_{(.013)}$	-2.32$_{(.011)}$	**4.22**$_{(.013)}$	2.01$_{(.015)}$	-2.11$_{(.003)}$	**10.2**$_{(.007)}$	2.62$_{(.012)}$	-2.76$_{(.007)}$	1.65$_{(.014)}$	2.36$_{(.013)}$	-2.47$_{(.010)}$
w/ Random	0.89$_{(.009)}$	2.02$_{(.005)}$	-2.19$_{(.011)}$	2.38$_{(.018)}$	1.91$_{(.011)}$	-2.11$_{(.016)}$	7.73$_{(.005)}$	2.47$_{(.019)}$	-2.59$_{(.011)}$	1.80$_{(.014)}$	2.31$_{(.006)}$	-2.41$_{(.010)}$
w/ Threshold	1.50$_{(.032)}$	2.01$_{(.048)}$	-2.18$_{(.031)}$	4.01$_{(.012)}$	1.97$_{(.044)}$	-2.13$_{(.006)}$	7.50$_{(.019)}$	2.35$_{(.020)}$	-2.86$_{(.010)}$	1.55$_{(.002)}$	2.12$_{(.018)}$	-2.21$_{(.017)}$
w/ Proactive	**1.83**$_{(.005)}$	2.01$_{(.023)}$	-2.18$_{(.017)}$	1.30$_{(.008)}$	**1.69**$_{(.011)}$	-2.14$_{(.012)}$	9.55$_{(.025)}$	2.22$_{(.032)}$	-2.42$_{(.005)}$	1.43$_{(.015)}$	**1.96**$_{(.006)}$	-2.41$_{(.015)}$
w/ IFRQE	1.40$_{(.014)}$	1.91$_{(.013)}$	-2.48$_{(.013)}$	2.92$_{(.137)}$	2.01$_{(.017)}$	-2.17$_{(.019)}$	8.17$_{(.033)}$	**2.16**$_{(.019)}$	-2.45$_{(.027)}$	**1.83**$_{(.015)}$	1.99$_{(.014)}$	-2.24$_{(.012)}$
w/ IFRQE++	0.70$_{(.015)}$	**1.80**$_{(.003)}$	**-2.11**$_{(.011)}$	3.00$_{(.017)}$	1.91$_{(.013)}$	**-2.08**$_{(.023)}$	8.87$_{(.021)}$	2.21$_{(.014)}$	**-2.37**$_{(.027)}$	1.22$_{(.031)}$	2.00$_{(.014)}$	**-2.17**$_{(.011)}$
LightGCN	0.90$_{(.005)}$	2.13$_{(.008)}$	-2.82$_{(.012)}$	**7.50**$_{(.009)}$	2.01$_{(.012)}$	-2.71$_{(.009)}$	10.1$_{(.010)}$	2.62$_{(.009)}$	-3.13$_{(.014)}$	**2.15**$_{(.021)}$	2.36$_{(.008)}$	-3.05$_{(.007)}$
w/ Random	0.82$_{(.013)}$	2.02$_{(.013)}$	-2.71$_{(.022)}$	7.48$_{(.011)}$	1.95$_{(.005)}$	-2.64$_{(.007)}$	10.0$_{(.027)}$	2.49$_{(.026)}$	-3.07$_{(.019)}$	1.75$_{(.003)}$	2.24$_{(.009)}$	-2.93$_{(.015)}$
w/ Threshold	0.96$_{(.038)}$	2.01$_{(.008)}$	-2.70$_{(.003)}$	7.13$_{(.017)}$	1.97$_{(.040)}$	-2.60$_{(.011)}$	9.40$_{(.048)}$	2.35$_{(.027)}$	-2.82$_{(.046)}$	1.53$_{(.033)}$	2.12$_{(.022)}$	-2.30$_{(.048)}$
w/ Proactive	**1.01**$_{(.013)}$	1.78$_{(.011)}$	-2.48$_{(.017)}$	5.51$_{(.038)}$	**1.42**$_{(.007)}$	-2.11$_{(.018)}$	**10.3**$_{(.021)}$	**2.12**$_{(.022)}$	-2.81$_{(.015)}$	1.28$_{(.013)}$	1.95$_{(.016)}$	-2.64$_{(.018)}$
w/ IFRQE	0.97$_{(.008)}$	2.12$_{(.010)}$	-2.82$_{(.014)}$	4.88$_{(.019)}$	1.47$_{(.011)}$	-2.16$_{(.013)}$	9.02$_{(.016)}$	2.49$_{(.025)}$	-2.82$_{(.008)}$	1.90$_{(.007)}$	**1.75**$_{(.018)}$	-2.13$_{(.013)}$
w/ IFRQE++	0.75$_{(.022)}$	**1.75**$_{(.008)}$	**-2.44**$_{(.016)}$	5.26$_{(.012)}$	1.32$_{(.014)}$	**-2.01**$_{(.009)}$	8.90$_{(.018)}$	2.48$_{(.018)}$	**-2.80**$_{(.005)}$	1.62$_{(.013)}$	**1.75**$_{(.009)}$	**-2.03**$_{(.023)}$
DIN	**1.43**$_{(.047)}$	2.13$_{(.001)}$	-2.56$_{(.044)}$	4.13$_{(.023)}$	2.01$_{(.005)}$	-2.22$_{(.031)}$	11.6$_{(.019)}$	2.62$_{(.048)}$	-2.91$_{(.005)}$	5.05$_{(.038)}$	2.36$_{(.012)}$	-2.48$_{(.026)}$
w/ Random	1.23$_{(.024)}$	2.02$_{(.037)}$	-2.42$_{(.037)}$	**4.33**$_{(.016)}$	1.90$_{(.042)}$	-2.10$_{(.024)}$	10.8$_{(.030)}$	2.48$_{(.042)}$	-2.78$_{(.014)}$	4.48$_{(.008)}$	2.24$_{(.023)}$	-2.39$_{(.015)}$
w/ Threshold	1.37$_{(.004)}$	2.01$_{(.022)}$	-2.13$_{(.029)}$	4.32$_{(.010)}$	1.63$_{(.032)}$	-1.70$_{(.036)}$	10.0$_{(.036)}$	2.35$_{(.045)}$	-2.86$_{(.029)}$	4.25$_{(.009)}$	2.12$_{(.006)}$	-2.24$_{(.022)}$
w/ Proactive	1.21$_{(.015)}$	1.67$_{(.012)}$	-2.06$_{(.007)}$	2.73$_{(.008)}$	1.63$_{(.024)}$	-2.19$_{(.006)}$	12.6$_{(.012)}$	2.12$_{(.016)}$	-2.43$_{(.015)}$	**5.33**$_{(.015)}$	1.84$_{(.016)}$	-1.97$_{(.005)}$
w/ IFRQE	1.36$_{(.030)}$	1.10$_{(.007)}$	-1.36$_{(.016)}$	3.92$_{(.019)}$	1.46$_{(.047)}$	-1.62$_{(.012)}$	**13.2**$_{(.021)}$	2.57$_{(.030)}$	-2.89$_{(.041)}$	5.26$_{(.007)}$	1.61$_{(.019)}$	-1.75$_{(.043)}$
w/ IFRQE++	1.03$_{(.016)}$	**1.09**$_{(.020)}$	**-1.33**$_{(.045)}$	3.50$_{(.014)}$	**1.24**$_{(.042)}$	**-1.38**$_{(.031)}$	9.53$_{(.030)}$	**2.02**$_{(.016)}$	**-2.17**$_{(.045)}$	5.18$_{(.021)}$	**1.57**$_{(.047)}$	**-1.70**$_{(.038)}$
CDAE	**1.23**$_{(.010)}$	2.13$_{(.015)}$	-2.30$_{(.031)}$	**1.48**$_{(.009)}$	2.01$_{(.036)}$	-2.08$_{(.012)}$	11.5$_{(.008)}$	2.62$_{(.029)}$	-2.68$_{(.014)}$	1.11$_{(.011)}$	2.36$_{(.027)}$	-2.44$_{(.045)}$
w/ Random	1.22$_{(.013)}$	1.90$_{(.013)}$	-2.71$_{(.022)}$	1.38$_{(.044)}$	1.99$_{(.022)}$	-2.06$_{(.031)}$	11.6$_{(.021)}$	2.49$_{(.003)}$	-2.55$_{(.041)}$	1.21$_{(.001)}$	2.24$_{(.015)}$	-2.31$_{(.031)}$
w/ Threshold	0.60$_{(.021)}$	2.01$_{(.003)}$	-2.08$_{(.041)}$	1.38$_{(.030)}$	1.81$_{(.026)}$	-1.88$_{(.016)}$	9.01$_{(.045)}$	2.35$_{(.044)}$	-2.54$_{(.010)}$	1.22$_{(.016)}$	2.12$_{(.040)}$	-2.19$_{(.035)}$
w/ Proactive	1.06$_{(.015)}$	1.97$_{(.017)}$	-2.06$_{(.027)}$	1.18$_{(.008)}$	1.64$_{(.011)}$	-1.76$_{(.016)}$	**16.3**$_{(.012)}$	1.79$_{(.022)}$	-1.85$_{(.025)}$	1.63$_{(.015)}$	1.79$_{(.016)}$	-1.85$_{(.005)}$
w/ IFRQE	0.92$_{(.011)}$	1.65$_{(.015)}$	-2.07$_{(.027)}$	**1.48**$_{(.011)}$	1.61$_{(.032)}$	-1.68$_{(.013)}$	11.3$_{(.008)}$	2.23$_{(.025)}$	-2.30$_{(.023)}$	1.80$_{(.027)}$	1.64$_{(.029)}$	-1.73$_{(.046)}$
w/ IFRQE++	0.86$_{(.037)}$	**1.45**$_{(.005)}$	**-1.88**$_{(.012)}$	1.43$_{(.007)}$	**1.45**$_{(.019)}$	**-1.52**$_{(.042)}$	11.4$_{(.018)}$	**1.64**$_{(.046)}$	**-1.70**$_{(.032)}$	**1.83**$_{(.049)}$	**1.52**$_{(.009)}$	**-1.61**$_{(.020)}$

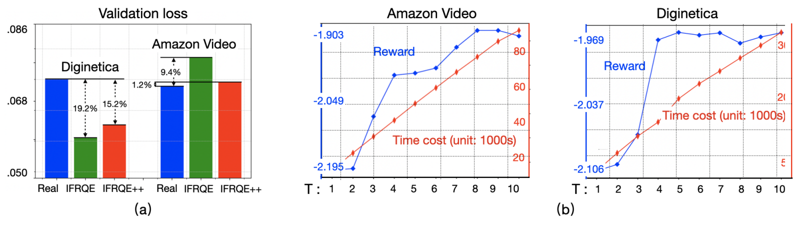

Figure 2: (a) Approximation error on the validation loss. (b) Influence of T.

IFRQE++ can improve the base model by about 12.4%, 11.1%, 14.6% and 21.3% on the datasets of simulation, Diginetica, Steam and Amazon Video, respectively. The improved performance of our models against the random method manifests that the studied problem is non-trivial, where blindly sampling the selection vectors does not work. Comparing with Proactive, our model do not improve the performance too much. However, training our framework is, on average, about 13.5 times faster, since Proactive needs to retrain the recommender model frequently. Between the proposed two methods, the improved version (*i.e.*, IFRQE++) is better in more cases. We speculate that the multiple anchor vectors may lead to more accurate approximation of the validation loss, which may propagate better signals to learn the optimal strategies, and thus result in better rewards.

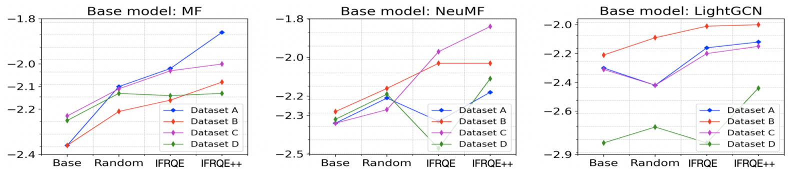

Figure 3: Model comparisons on the datasets with differently characterized user willingness.

5.3 Approximation of the Validation Loss

The key of our model lies in the accurate approximation of the validation loss. We investigate the approximation error of our models. We focus on the datasets of Diginetica and Amazon Video, and MF is selected as the base model. We randomly sample ten selection vectors and train the base model to obtain the parameters and real validation loss. Then, we approximate the validation loss by equation (3) and (6), where we use two anchor vectors in **IFRQE++**. At last, the (approximated) validation losses are reported by averaging over all the selection vectors, and we label the approximation error by $\frac{|\text{real validation loss} - \text{approximated validation loss}|}{\text{real validation loss}}$. The results are presented in Figure 2(a), and we can see the approximation accuracy of different models is satisfied. Especially, on Amazon Video, the approximation error of **IFRQE++** is only 1.2%. By leveraging more anchor vectors, **IFRQE++** is more accurate than **IFRQE** on both datasets. In specific, the approximation error is reduced from 19.2% to 15.2% and 9.4% to 1.2% on the datasets of Diginetica and Amazon Video, respectively.

5.4 Influence of the Number of Anchor Vectors

In this section, we study the influence of the number of anchor vectors in **IFRQE++**. More specifically, we tune T in the range of $[1, 10]$, where $T = 1$ is exactly the method proposed in section 3.1. We base the experiments on the same datasets and base model as in the above experiment. We report the reward as well as the time cost for each T in Figure 2(b). We can see: on both datasets, the performances are not satisfied when $T = 1$, and more anchor vectors can indeed improve the performance. With the increasing of T, the reward continues to rise up, and tends to be stable at last, where we speculate that, at this time, the performance is dominated by some key anchor vectors, and introducing the other ones do not make much help. From the efficiency perspective, as T becomes larger, our model costs more time, which is as expected, since larger T means more times to retrain the base model. In practice, one can properly set T to achieve better trade-offs between the effectiveness and efficiency.

5.5 Influence of Different Characters of the User Willingness

Here we study whether our model is always effective on differently characterized user willingness. We base the experiments on the simulation datasets, where the disclosing willingness of a user is controlled by the sensitive parameter s_u and average disclosing level b_u. Considering that the distributions of s_u and b_u are determined by a_1, a_2 and a_3, we build four datasets by specifying (a_1, a_2, a_3) with $A = (0.6, 2, 1.2)$, $B = (0.4, 2, 0.8)$ and $C = (0.6, 1, 1.5)$ and $D = (0.5, 1, 1)$ to simulate different user willingness characters. We present the comparisons of different models on these datasets in Figure 3, where we can see: the random method is less competitive, and sometimes, it is even worse than the base model (*e.g.*, Datasets A and B with LightGCN as the base model). For different datasets and base models, IFRQE and IFRQE++ can usually obtain the second best and best performances. This result agrees with the observations in section 5.2 and manifests that our methods are generally effective for different user willingness characters.

6 Conclusion and Future Work

This paper improves traditional system-centered recommendation paradigm by allowing the users to participate into the model optimization process via specifying their disclosing willingness. However, there are some limitations can be studied in the future. We do not consider the dynamic nature of the recommender system, therefore, an interesting direction is to study the temporal user disclosing willingness, for example, under on-line settings. In addition, we currently assume that the user overall utility is a linear combination between the recommendation quality and user willingness. One can extend it to the non-linear case, where more efforts are needed to solve the multiplayer game.

REPRODUCIBILITY STATEMENT

For the experimental results presented in the paper, we include the code and simulation dataset in the supplemental material, and specify all the training details in Appendix. For the datasets used in the paper, we also give a clear explanation in Section 5.1. More details are available at https://ifrqe.github.io/IFRQE/.

REFERENCES

Omer Ben-Porat and Moshe Tennenholtz. A game-theoretic approach to recommendation systems with strategic content providers. *Advances in Neural Information Processing Systems*, 31, 2018.

Paul H Calamai and Jorge J Moré. Projected gradient methods for linearly constrained problems. *Mathematical programming*, 39(1):93–116, 1987.

Chen Chen, Jingfeng Zhang, Anthony KH Tung, Mohan Kankanhalli, and Gang Chen. Robust federated recommendation system. *arXiv preprint arXiv:2006.08259*, 2020.

Chong Chen, Fei Sun, Min Zhang, and Bolin Ding. Recommendation unlearning. *arXiv preprint arXiv:2201.06820*, 2022a.

Ziqian Chen, Fei Sun, Yifan Tang, Haokun Chen, Jinyang Gao, and Bolin Ding. Proactively control privacy in recommender systems. *arXiv preprint arXiv:2204.00279*, 2022b.

Davide Chicco and Giuseppe Jurman. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):1–13, 2020.

Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In *The world wide web conference*, pp. 417–426, 2019.

Maria Halkidi and Iordanis Koutsopoulos. A game theoretic framework for data privacy preservation in recommender systems. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 629–644. Springer, 2011.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pp. 173–182, 2017.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 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.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206. IEEE, 2018.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pp. 1885–1894. PMLR, 2017.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

Yuyuan Li, Xiaolin Zheng, Chaochao Chen, and Junlin Liu. Making recommender systems forget: Learning and unlearning for erasable recommendation. *arXiv preprint arXiv:2203.11491*, 2022.

Guanyu Lin, Feng Liang, Weike Pan, and Zhong Ming. Fedrec: Federated recommendation with explicit feedback. *IEEE Intelligent Systems*, 36(5):21–30, 2020.

Denis Parra and Peter Brusilovsky. User-controllable personalization: A case study with setfusion. *International Journal of Human-Computer Studies*, 78:43–67, 2015.

Francesco Ricci, Lior Rokach, and Bracha Shapira. Recommender systems: introduction and challenges. In *Recommender systems handbook*, pp. 1–34. Springer, 2015.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 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.

Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pp. 565–573, 2018.

Wenjie Wang, Fuli Feng, Liqiang Nie, and Tat-Seng Chua. User-controllable recommendation against filter bubbles. *arXiv preprint arXiv:2204.13844*, 2022.

Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 950–958, 2019.

Chuhan Wu, Fangzhao Wu, Yang Cao, Yongfeng Huang, and Xing Xie. Fedgnn: Federated graph neural network for privacy-preserving recommendation. *arXiv preprint arXiv:2102.04925*, 2021.

Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the ninth ACM international conference on web search and data mining*, pp. 153–162, 2016.

Lei Xu, Chunxiao Jiang, Yan Chen, Yong Ren, and KJ Ray Liu. User participation in collaborative filtering-based recommendation systems: a game theoretic approach. *IEEE transactions on cybernetics*, 49(4):1339–1352, 2018.

Liu Yang, Ben Tan, Vincent W Zheng, Kai Chen, and Qiang Yang. Federated recommendation systems. In *Federated Learning*, pp. 225–239. Springer, 2020.

Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)*, 52(1):1–38, 2019.

Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 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.

A Appendix

A.1 Proof of theory 1

Proof. For a candidate selection vector $\boldsymbol{o}$, we have $\hat{\boldsymbol{\theta}}(\boldsymbol{o}) = \arg\min_{\boldsymbol{\theta}} \frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} o_k^v l_f(s_k^v, \boldsymbol{\theta})$, where if $o_k^v = 0$, then s_k^v is not leveraged to train the model. Suppose we define:

$$\hat{\boldsymbol{\theta}}(\boldsymbol{o}, \epsilon) = \arg\min_{\boldsymbol{\theta}}[\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v l_f(s_k^v, \boldsymbol{\theta}) + \epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|} (\widetilde{o}_k^v - o_k^v) l_f(s_k^v, \boldsymbol{\theta})]. \tag{11}$$

Then we have $\hat{\boldsymbol{\theta}}(\boldsymbol{o}) = \hat{\boldsymbol{\theta}}(\boldsymbol{o}, -\frac{1}{Z})$ and $\widetilde{\boldsymbol{\theta}} = \hat{\boldsymbol{\theta}}(\boldsymbol{o}, 0)$.

Since $\hat{\boldsymbol{\theta}}(\boldsymbol{o}, \epsilon)$ is the optimal solution of (11), then

$$\begin{aligned} 0 &\approx \bigtriangledown[\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v l_f(s_k^v, \hat{\boldsymbol{\theta}}(\boldsymbol{o}^u, \epsilon)) + \epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|} (\widetilde{o}_k^v - o_k^v) l_f(s_k^v, \hat{\boldsymbol{\theta}}(\boldsymbol{o}^u, \epsilon))] \\ &= \frac{1}{Z}\sum_u \sum_{k=1}^{|\mathcal{S}^u|} \widetilde{o}_k^v \bigtriangledown l_f(s_k^v, \hat{\boldsymbol{\theta}}(\boldsymbol{o}^u, \epsilon)) + \epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|} (\widetilde{o}_k^v - o_k^v) \bigtriangledown l_f(s_k^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o}^u, \epsilon))]. \end{aligned} \tag{12}$$

We regard $\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown l_f(s_k^v, \hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon)) + \epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^v - o_k^v) \bigtriangledown l_f(s_k^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon))]$ as a function of $\hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon)$. If $\epsilon \to 0$, then $\hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon) \to \widetilde{\boldsymbol{\theta}}$. According to the Taylor expansion, we have:

$$\begin{aligned}
0 \approx& \frac{1}{Z}\sum_u \sum_{k=1}^{|\mathcal{S}^u|} \widetilde{o}_k^v \bigtriangledown l_f(s_k^u, \widetilde{\boldsymbol{\theta}}) + \epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^v - o_k^v) \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) \\
&+ [\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) + \epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^v - o_k^v) \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})](\hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon) - \widetilde{\boldsymbol{\theta}})
\end{aligned} \tag{13}$$

Let $\triangle_\epsilon = \hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon) - \widetilde{\boldsymbol{\theta}}$, then

$$\begin{aligned}
0 \approx& \frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) + \epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^v - o_k^v) \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) \\
&+ [\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) + \epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^v - o_k^v) \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})]\triangle_\epsilon \\
\triangle_\epsilon \approx& -[\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) + \epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^v - o_k^v) \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})]^{-1} \\
&[\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) + \epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^v - o_k^v) \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}})]
\end{aligned} \tag{14}$$

Since $| \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})| \leq B$ and $\sum_v \sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^v - o_k^v)B$ is a small value, we can ignore the term $\epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^v - o_k^v) \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})$, which lead to the following equation:

$$\triangle_\epsilon \approx -[\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})]^{-1}[\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) + \epsilon \sum_v \sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^v - o_k^v) \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}})] \tag{15}$$

If Z is a large value, then $\frac{d\hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon)}{d\epsilon}|_{\epsilon\to 0} \approx \frac{\triangle_{-\frac{1}{Z}}}{-\frac{1}{Z}}$. According to (15),

$$\begin{aligned}
\triangle_{-\frac{1}{Z}} \approx& -[\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})]^{-1}[\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) - \frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^v - o_k^v) \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}})] \\
=& -[\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})]^{-1}[\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} o_k^v \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}})]
\end{aligned} \tag{16}$$

Thus, $\frac{d\hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon)}{d\epsilon}|_{\epsilon\to 0} \approx \frac{\triangle_{-\frac{1}{Z}}}{-\frac{1}{Z}} = [\frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})]^{-1}[\sum_v \sum_{k=1}^{|\mathcal{S}^v|} o_k^v \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}})]$

Because

$$\begin{aligned}
\frac{L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o})) - L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}})}{-\frac{1}{Z}} =& \frac{L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o}, -\frac{1}{Z})) - L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}})}{-\frac{1}{Z}} \approx \frac{dL_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon))}{d\epsilon}|_{\epsilon\to 0} \\
=& \sum_{y\in\mathcal{T}^u} \bigtriangledown l_f(y, \widetilde{\boldsymbol{\theta}}) \times \frac{d\hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon)}{d\epsilon}|_{\epsilon\to 0} \\
\approx& \sum_{y\in\mathcal{T}^u} \bigtriangledown l_f(y, \widetilde{\boldsymbol{\theta}}) H_{\widetilde{\boldsymbol{\theta}}}^{-1}[\sum_v \sum_{k=1}^{|\mathcal{S}^v|} o_k^v \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}})]
\end{aligned} \tag{17}$$

where $H_{\widetilde{\boldsymbol{\theta}}} = \frac{1}{Z}\sum_v \sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^v \bigtriangledown^2 l_f(s_k^v, \widetilde{\boldsymbol{\theta}})$, then we have:

$$\begin{aligned} L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o})) - L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}) &\approx -\frac{1}{Z}\sum_{y\in\mathcal{T}^u} \bigtriangledown l_f(y, \widetilde{\boldsymbol{\theta}}) H_{\widetilde{\boldsymbol{\theta}}}^{-1}[\sum_v \sum_{k=1}^{|\mathcal{S}^v|} o_k^v \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}})] \\ L_f(\mathcal{T}^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o})) &\approx L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}) - \frac{1}{Z}\sum_{y\in\mathcal{T}^u}\sum_v\sum_{k=1}^{|\mathcal{S}^v|} o_k^v \bigtriangledown l_f(y, \widetilde{\boldsymbol{\theta}}) H_{\widetilde{\boldsymbol{\theta}}}^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) \end{aligned} \tag{18}$$

□

A.2 Proof of theory 2

Proof. By bringing the result of theory 1 into $\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u})$, we have

$$\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u}) = -L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}) + \frac{1}{Z}\sum_{y\in\mathcal{T}^u}\sum_{v\in\mathcal{U}}\sum_{k=1}^{|\mathcal{S}^v|} o_k^v \bigtriangledown l_f(y, \widetilde{\boldsymbol{\theta}}) H_{\widetilde{\boldsymbol{\theta}}}^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) - \lambda \sum_{k=1}^{|\mathcal{S}^u|} o_k^u \beta_k^u. \tag{19}$$

Recall that $\boldsymbol{g}_y^v = [g(s_1^v, y), g(s_2^v, y), ..., g(s_{|\mathcal{S}^v|}^v, y)]$, where $g(s_k^v, y) = \bigtriangledown_\theta l_f(y, \widetilde{\boldsymbol{\theta}})^T H_{\widetilde{\boldsymbol{\theta}}}^{-1} \bigtriangledown_\theta l_f(s_k^v, \widetilde{\boldsymbol{\theta}})$, then

$$\begin{aligned} & z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u}) = E_{\boldsymbol{o}}[\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u})] \\ =& -L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}) + E_{\boldsymbol{o}}[\frac{1}{Z}\sum_{y\in\mathcal{T}^u}\sum_{v\in\mathcal{U}}\sum_{k=1}^{|\mathcal{S}^v|} o_k^v g(s_k^{u'}, y) - \lambda\sum_{k=1}^{|\mathcal{S}^u|} o_k^u \beta_k^u] \\ =& -L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}) + E_{\boldsymbol{o}}[\frac{1}{Z}\sum_{y\in\mathcal{T}^u}\sum_{v\in\mathcal{U}} (\boldsymbol{o}^v)^T \boldsymbol{g}_y^v - \lambda(\boldsymbol{o}^u)^T\boldsymbol{\beta}^u] \\ =& -L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}) + E_{\boldsymbol{o}}[\sum_{v\in\mathcal{U}} (\boldsymbol{o}^v)^T \sum_{y\in\mathcal{T}^u}\frac{\boldsymbol{g}_y^v}{Z} - \lambda(\boldsymbol{o}^u)^T\boldsymbol{\beta}^u] \\ =& -L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}) + \sum_{v\neq u} E_{\boldsymbol{o}}[(\boldsymbol{o}^v)^T \sum_{y\in\mathcal{T}^u}\frac{\boldsymbol{g}_y^v}{Z}] + E_{\boldsymbol{o}}[(\boldsymbol{o}^u)^T\sum_{y\in\mathcal{T}^u}\frac{\boldsymbol{g}_y^u}{Z}] - \lambda E_{\boldsymbol{o}}[(\boldsymbol{o}^u)^T\boldsymbol{\beta}^u] \\ =& -L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}) + \sum_{v\neq u} E_{\boldsymbol{o}}[(\boldsymbol{o}^v)^T \sum_{y\in\mathcal{T}^u}\frac{\boldsymbol{g}_y^v}{Z}] + E_{\boldsymbol{o}^u}[(\boldsymbol{o}^u)^T(\sum_{y\in\mathcal{T}^u}\frac{\boldsymbol{g}_y^u}{Z} - \lambda\boldsymbol{\beta}^u)] \end{aligned} \tag{20}$$

□

A.3 Proof of theory 3

Proof. The proof of this theory is similar to that of theory 1. We define:

$$\hat{\boldsymbol{\theta}}(\boldsymbol{o}, \epsilon) = \arg\min_{\boldsymbol{\theta}}[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|} \widetilde{\boldsymbol{o}}_k^{t,v} l_f(s_k^v, \boldsymbol{\theta}) + \epsilon\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{\boldsymbol{o}}_k^{t,v} - o_k^v) l_f(s_k^v, \boldsymbol{\theta})]. \tag{21}$$

Then,

$$\begin{aligned} 0 &\approx \bigtriangledown[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|} \widetilde{o}_k^{t,v} l_f(s_k^v, \hat{\boldsymbol{\theta}}(\boldsymbol{o}, \epsilon)) + \epsilon\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^{t,v} - o_k^v) l_f(s_k^v, \hat{\boldsymbol{\theta}}(\boldsymbol{o}, \epsilon))] \\ &= \frac{1}{Z}\sum_u\sum_{k=1}^{|\mathcal{S}^u|} \widetilde{o}_k^{t,v} \bigtriangledown l_f(s_k^v, \hat{\boldsymbol{\theta}}(\boldsymbol{o}, \epsilon)) + \epsilon\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^{t,v} - o_k^v) \bigtriangledown l_f(s_k^u, \hat{\boldsymbol{\theta}}(\boldsymbol{o}, \epsilon))]. \end{aligned} \tag{22}$$

According to Taylor expansion at point $\widetilde{\boldsymbol{\theta}}^t$, we have:

$$\begin{aligned} 0 &\approx \frac{1}{Z}\sum_u\sum_{k=1}^{|\mathcal{S}^u|}\widetilde{o}_k^{t,v}\bigtriangledown l_f(s_k^u,\widetilde{\boldsymbol{\theta}}^t)+\epsilon\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^{t,v}-o_k^v)\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t) \\ &+[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)+\epsilon\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^{t,v}-o_k^v)\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)](\hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon)-\widetilde{\boldsymbol{\theta}}^t) \end{aligned} \tag{23}$$

Let $\triangle_\epsilon=\hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon)-\widetilde{\boldsymbol{\theta}}^t$, then:

$$\begin{aligned} 0 &\approx \frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)+\epsilon\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^{t,v}-o_k^v)\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t) \\ &+[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)+\epsilon\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^{t,v}-o_k^v)\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)]\triangle_\epsilon \\ \triangle_\epsilon &\approx -[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)+\epsilon\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^{t,v}-o_k^v)\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)]^{-1} \\ &[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)+\epsilon\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^{t,v}-o_k^v)\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)] \end{aligned} \tag{24}$$

By ignoring $\epsilon\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^{t,v}-o_k^v)\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}})$, we have:

$$\triangle_\epsilon\approx-[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)]^{-1}[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)+\epsilon\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^{t,v}-o_k^v)\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)] \tag{25}$$

Then

$$\begin{aligned} \triangle_{-\frac{1}{Z}} &\approx -[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)]^{-1}[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)-\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}(\widetilde{o}_k^{t,v}-o_k^v)\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)] \\ &= -[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)]^{-1}[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}o_k^v\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)] \end{aligned} \tag{26}$$

Thus, $\frac{d\hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon)}{d\epsilon}|_{\epsilon\to0}\approx\frac{\triangle_{-\frac{1}{Z}}}{-\frac{1}{Z}}=[\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)]^{-1}[\sum_v\sum_{k=1}^{|\mathcal{S}^v|}o_k^v\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)]$, and we have

$$\begin{aligned} \frac{L_f(\mathcal{T}^u,\hat{\boldsymbol{\theta}}(\boldsymbol{o}))-L_f(\mathcal{T}^u,\widetilde{\boldsymbol{\theta}}^t)}{-\frac{1}{Z}} &= \frac{L_f(\mathcal{T}^u,\hat{\boldsymbol{\theta}}(\boldsymbol{o},-\frac{1}{Z}))-L_f(\mathcal{T}^u,\widetilde{\boldsymbol{\theta}}^t)}{-\frac{1}{Z}}\approx\frac{dL_f(\mathcal{T}^u,\hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon))}{d\epsilon}|_{\epsilon\to0} \\ &= \sum_{y\in\mathcal{T}^u}\bigtriangledown l_f(y,\widetilde{\boldsymbol{\theta}}^t)\times\frac{d\hat{\boldsymbol{\theta}}(\boldsymbol{o},\epsilon)}{d\epsilon}|_{\epsilon\to0} \\ &\approx \sum_{y\in\mathcal{T}^u}\bigtriangledown l_f(y,\widetilde{\boldsymbol{\theta}}^t)H_{\widetilde{\boldsymbol{\theta}}^t}^{-1}[\sum_v\sum_{k=1}^{|\mathcal{S}^v|}o_k^v\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)] \end{aligned} \tag{27}$$

where $H_{\widetilde{\boldsymbol{\theta}}^t}=\frac{1}{Z}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}\widetilde{o}_k^{t,v}\bigtriangledown^2 l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)$. At last,

$$\begin{aligned} L_f(\mathcal{T}^u,\hat{\boldsymbol{\theta}}(\boldsymbol{o}))-L_f(\mathcal{T}^u,\widetilde{\boldsymbol{\theta}}^t) &\approx -\frac{1}{Z}\sum_{y\in\mathcal{T}^u}\bigtriangledown l_f(y,\widetilde{\boldsymbol{\theta}}^t)H_{\widetilde{\boldsymbol{\theta}}^t}^{-1}[\sum_v\sum_{k=1}^{|\mathcal{S}^v|}o_k^v\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t)] \\ L_f(\mathcal{T}^u,\hat{\boldsymbol{\theta}}(\boldsymbol{o})) &\approx L_f(\mathcal{T}^u,\widetilde{\boldsymbol{\theta}}^t)-\frac{1}{Z}\sum_{y\in\mathcal{T}^u}\sum_v\sum_{k=1}^{|\mathcal{S}^v|}o_k^v\bigtriangledown l_f(y,\widetilde{\boldsymbol{\theta}}^t)H_{\widetilde{\boldsymbol{\theta}}^t}^{-1}\bigtriangledown l_f(s_k^v,\widetilde{\boldsymbol{\theta}}^t) \end{aligned} \tag{28}$$

□

A.4 Proof of theory 4

Proof.

$$\begin{aligned}
&z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u}) = E_{\boldsymbol{o}}[\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u})] = E_{\boldsymbol{o}}[\sum_{t=1}^{T} \mathbf{1}(\boldsymbol{o} \in A_t)\overline{z}_u(\boldsymbol{o}^u, \boldsymbol{o}^{-u}, t)] \\
&= \sum_{t=1}^{T}\sum_{\boldsymbol{o}} \mathbf{1}(\boldsymbol{o} \in A_t)\alpha^u(\boldsymbol{o}^u)\boldsymbol{\alpha}^{-u}(\boldsymbol{o}^{-u})\{-L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}^t) + \frac{1}{Z}\sum_{y\in\mathcal{T}^u}\sum_{v\in\mathcal{U}}\sum_{k=1}^{|\mathcal{S}^v|} o_k^v g(s_k^v, y, t) - \lambda\sum_{k=1}^{|\mathcal{S}^u|} o_k^u \beta_k^u\} \\
&= \sum_{t=1}^{T}\sum_{\boldsymbol{o}} \mathbf{1}(\boldsymbol{o} \in A_t)\alpha^u(\boldsymbol{o}^u)\boldsymbol{\alpha}^{-u}(\boldsymbol{o}^{-u})\{-L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}^t) + \frac{1}{Z}\sum_{y\in\mathcal{T}^u}\sum_{v\in\mathcal{U}}(\boldsymbol{o}^v)^T \boldsymbol{g}_y^{t,v} - \lambda(\boldsymbol{o}^u)^T\boldsymbol{\beta}^u\} \\
&= \sum_{t=1}^{T}\sum_{\boldsymbol{o}} \mathbf{1}(\boldsymbol{o} \in A_t)\alpha^u(\boldsymbol{o}^u)\boldsymbol{\alpha}^{-u}(\boldsymbol{o}^{-u})\{-L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}^t) + \frac{1}{Z}\sum_{v\in\mathcal{U}}(\boldsymbol{o}^v)^T \boldsymbol{g}^{t,v} - \lambda(\boldsymbol{o}^u)^T\boldsymbol{\beta}^u\} \\
&= \sum_{t=1}^{T}\sum_{\boldsymbol{o}} \mathbf{1}(\boldsymbol{o} \in A_t)\alpha^u(\boldsymbol{o}^u)\boldsymbol{\alpha}^{-u}(\boldsymbol{o}^{-u})\{-L_f(\mathcal{T}^u, \widetilde{\boldsymbol{\theta}}^t) + \frac{1}{Z}\sum_{v\neq u}(\boldsymbol{o}^v)^T \boldsymbol{g}^{t,v} + \frac{1}{Z}(\boldsymbol{o}^u)^T\boldsymbol{g}^{t,u} - \lambda(\boldsymbol{o}^u)^T\boldsymbol{\beta}^u\}
\end{aligned} \tag{29}$$

□

A.5 Proof of theory 5

We rewrite the objective as follows:

$$\max_{\boldsymbol{\alpha}^u\in\triangle}\sum_{\boldsymbol{o}^u}\alpha^u_{\boldsymbol{o}^u}[\sum_{\boldsymbol{o}^{-u}}\alpha^{-u}_{\boldsymbol{o}^{-u}}\sum_{t=1}^{T}\mathbf{1}(\boldsymbol{o}\in A_t)B(\boldsymbol{o}^u, \boldsymbol{o}^{-u}, t)] \tag{30}$$

Suppose the optimal solution for (30) is $\boldsymbol{\alpha}^u$, and the output of the lth iteration is $\boldsymbol{\alpha}^u_l$. Recall that $\boldsymbol{g} = \sum_{\boldsymbol{o}^{-u}}\alpha^{-u}_{\boldsymbol{o}^{-u}}\sum_{t=1}^{T}\mathbf{1}(\boldsymbol{o}\in A_t)B(\boldsymbol{o}^u, \boldsymbol{o}^{-u}, t)$, then we have:

$$\begin{aligned}
E[\boldsymbol{\alpha}^u\boldsymbol{g} - \boldsymbol{\alpha}^u_l\boldsymbol{g}] &= E[(\boldsymbol{\alpha}^u - \boldsymbol{\alpha}^u_l)\hat{\boldsymbol{g}}] \\
&= E[\frac{1}{2\gamma}(||\boldsymbol{\alpha}^u - \boldsymbol{\alpha}^u_l||_2^2 + \gamma^2||\hat{\boldsymbol{g}}||_2^2 - ||\boldsymbol{\alpha}^u - (\boldsymbol{\alpha}^u_l + \gamma\hat{\boldsymbol{g}}||_2^2)] \\
&\leq E[\frac{1}{2\gamma}(||\boldsymbol{\alpha}^u - \boldsymbol{\alpha}^u_l||_2^2 + \gamma^2||\hat{\boldsymbol{g}}||_2^2 - ||\boldsymbol{\alpha}^u - \boldsymbol{\alpha}^u_{l+1}||_2^2)] \\
&\leq E[\frac{1}{2\gamma}(||\boldsymbol{\alpha}^u - \boldsymbol{\alpha}^u_l||_2^2 - ||\boldsymbol{\alpha}^u - \boldsymbol{\alpha}^u_{l+1}||_2^2 + \gamma^2G^2)]
\end{aligned} \tag{31}$$

where the third line hold because $\boldsymbol{\alpha}^u_{l+1} = \Pi_\triangle[\boldsymbol{\alpha}^u_l + \gamma\hat{\boldsymbol{g}}(\alpha^u)]$.

In the next,

$$\begin{aligned}
\sum_{l=1}^{L}E[\boldsymbol{\alpha}^u\boldsymbol{y} - \boldsymbol{\alpha}^u_l\boldsymbol{y}] &\leq E[\frac{1}{2\gamma}(||\boldsymbol{\alpha}^u - \boldsymbol{\alpha}^u_1||_2^2 - ||\boldsymbol{\alpha}^u - \boldsymbol{\alpha}^u_{L+1}||_2^2 + L\gamma^2G^2)] \\
&\leq E[\frac{1}{2\gamma}(||\boldsymbol{\alpha}^u - \boldsymbol{\alpha}^u_1||_2^2 + L\gamma^2G^2)] \\
&\leq \frac{1}{\gamma} + L\gamma^2G^2
\end{aligned} \tag{32}$$

where the third line hold because $\boldsymbol{\alpha}^u$ and $\boldsymbol{\alpha}^u_1$ are both simplex.

Table 2: Statistics of the datasets

Dataset	# User	# Item	# Interaction	Sparsity
Simulation	1000	1000	6148	99.39%
Diginetica	2852	10739	17073	99.94%
Steam	11942	6955	86595	99.89%
Amazon Video	2790	12435	18703	99.95%

Table 3: Statistics of the simulation datasets with different η's, where the number of users and items are both 1000.

η	0.1	0.2	0.3	0.4	0.5
#Interaction	11296	10068	8828	7507	6184
Sparsity	98.87%	99.00%	99.12%	99.25%	99.39%

At last,

$$\begin{aligned}
\frac{1}{L}\sum_{l=1}^{L} E[\boldsymbol{\alpha}^u \boldsymbol{g} - \boldsymbol{\alpha}^u_l \boldsymbol{g}] &\leq \frac{1}{L\gamma} + \gamma^2 G^2 \\
E[\boldsymbol{\alpha}^u \boldsymbol{g}] - \frac{1}{L}\sum_{l=1}^{L} E[\boldsymbol{\alpha}^u_l \boldsymbol{g}] &\leq \frac{1}{L\gamma} + \gamma^2 G^2 \\
\frac{1}{L}\sum_{l=1}^{L} E[\boldsymbol{\alpha}^u_l \boldsymbol{g}] &\geq E[\boldsymbol{\alpha}^u \boldsymbol{g}] - (\frac{1}{L\gamma} + \gamma^2 G^2) \\
\frac{1}{L}\sum_{l=1}^{L} E[\boldsymbol{\alpha}^u_l \boldsymbol{g}] &\geq \max_{\boldsymbol{\alpha}^u} E[\boldsymbol{\alpha}^u \boldsymbol{g}] - (\frac{1}{L\gamma} + \gamma^2 G^2) \\
E[z_u(\hat{\boldsymbol{\alpha}}^u, \boldsymbol{\alpha}^{-u})] = E[\hat{\boldsymbol{\alpha}}^u \boldsymbol{g}] &\geq \max_{\boldsymbol{\alpha}^u} E[\boldsymbol{\alpha}^u \boldsymbol{g}] - (\frac{1}{L\gamma} + \gamma^2 G^2) \\
&= \max_{\boldsymbol{\alpha}^u} E[z_u(\boldsymbol{\alpha}^u, \boldsymbol{\alpha}^{-u})] - (\frac{1}{L\gamma} + \gamma^2 G^2)
\end{aligned} \tag{33}$$

A.6 Proof of theory 6

In equation (14), the approximation error comes from ignoring the term $\sum_v \sum_{k=1}^{|\mathcal{S}^v|} (\widetilde{o}^v_k - o^v_k) \bigtriangledown^2 l_f(s^v_k, \widetilde{\boldsymbol{\theta}})$, where $\widetilde{o}^v = \{\widetilde{o}^v_1, ...\widetilde{o}^v_{|\mathcal{S}^v|}\}$ is the anchor selection vector of user v. Let D be the hamming distance counting the number of different bits between two vectors, then we consider the value $\sum_v \sum_{k=1}^{|\mathcal{S}^v|} |\widetilde{o}^v_k - o^v_k| B = \sum_v D(\widetilde{o}^v, \boldsymbol{o}^v) B$, which upper bounds the approximation error, and are interrested in whether the multi-anchor proposal can reduce this value.

In specific, for a given selection vector $\boldsymbol{o}$, suppose $\widetilde{\boldsymbol{o}}^{t_1}$ is the nearest anchor vector to $\boldsymbol{o}$ in $\boldsymbol{P}$. Since $\boldsymbol{P} \subseteq \boldsymbol{Q}$, we have $\widetilde{\boldsymbol{o}}^{t_1} \in \boldsymbol{Q}$. Suppose $\widetilde{\boldsymbol{o}}^{t_2}$ is the nearest anchor vector to $\boldsymbol{o}$ in $\boldsymbol{Q}$, then according the definition of $\boldsymbol{Q}$, we have $\sum_{v=1}^{N} D(\widetilde{\boldsymbol{o}}^{t_2,v}, \boldsymbol{o}^v) \leq \sum_{v=1}^{N} D(\widetilde{\boldsymbol{o}}^{t_1,v}, \boldsymbol{o}^v)$. Since $B > 0$, we have $\sum_{v=1}^{N} D(\widetilde{\boldsymbol{o}}^{t_2,v}, \boldsymbol{o}^v) B \leq \sum_{v=1}^{N} D(\widetilde{\boldsymbol{o}}^{t_1,v}, \boldsymbol{o}^v) B$. By summing all the candidate selection vectors, and grouping them according to A^P_t and A^Q_t, respectively, we have:

$$\sum_{t=1}^{T_P} \sum_{\boldsymbol{o} \in A^P_t} [\sum_v D(\widetilde{\boldsymbol{o}}^{t,v}, \boldsymbol{o}^v)] B \geq \sum_{t=1}^{T_Q} \sum_{\boldsymbol{o} \in A^Q_t} [\sum_v D(\widetilde{\boldsymbol{o}}^{t,v}, \boldsymbol{o}^v)] B, \tag{34}$$

A.7 Learning Algorithm for the Basic Model

Here, we present the learning algorithms for the basic model in algorithm 3.

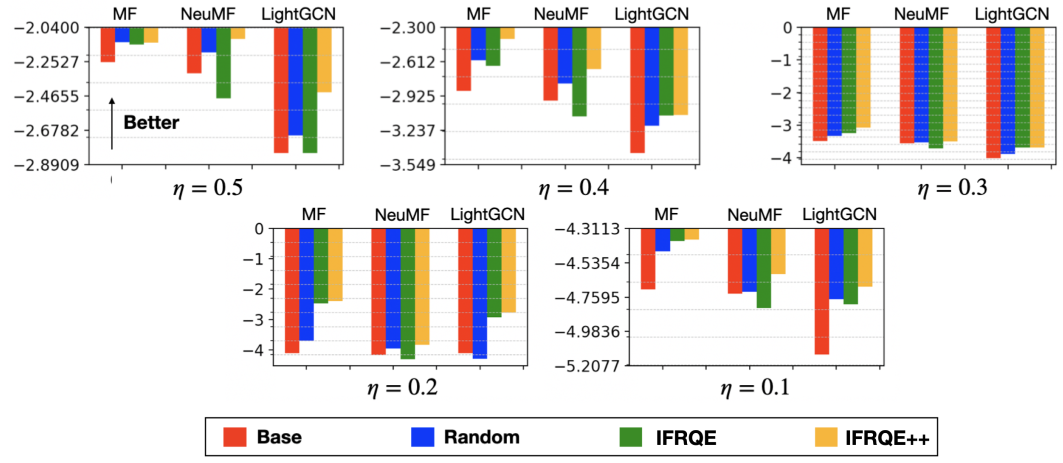

Figure 4: Performance comparison on the dataset with different sparsities.

Algorithm 3: Training process of the based model

```
1  Initialize $\{\boldsymbol{\alpha}^1, \boldsymbol{\alpha}^2, ..., \boldsymbol{\alpha}^N\}$ and let $\boldsymbol{\alpha}_0^u = \boldsymbol{\alpha}^u \ (u \in [1, N])$.
2  Indicate the max iteration number M and threshold $\kappa$.
3  for m in [1, M] do
4      for u in [1, N] do
5          Let $\boldsymbol{\alpha}_{m-1}^{-u} = \{\boldsymbol{\alpha}_{m-1}^1, ..., \boldsymbol{\alpha}_{m-1}^{u-1}, \boldsymbol{\alpha}_{m-1}^{u+1}, ..., \boldsymbol{\alpha}_{m-1}^N\}$.
6          Learning $\boldsymbol{\alpha}_m^u$ based on (5) by fixing $\boldsymbol{\alpha}_{m-1}^{-u}$.
7      end
8      if $|\boldsymbol{\alpha}_m^u - \boldsymbol{\alpha}_{m-1}^u| < \kappa$ , $\forall u \in [1, N]$ then
9          Break.
10     end
11 end
12 Output $\boldsymbol{\alpha}^{u*} = \boldsymbol{\alpha}_m^u \ (u \in [1, N])$.
```

A.8 More Implementation Details

For the simulation dataset, the threshold η and (a_1, a_2, a_3) are initially set as 0.5 and $(0.5, 1, 1)$, respectively. And then, we tune them in the experiments to study the influence of different dataset sparsities and user willingness characters. For the real world datasets, **Diginetica** and **Amazon Video** are e-commerce datasets, where we are provided with the user-item purchasing records. **Steam** is a game dataset, which includes the interactions (*e.g.*, reviewing behaviors) between the users and games. Since we do not know the real user disclosing willingness, we simulate it by randomly assigning the willingness vector for each user, and repeat the experiments for ten times to make sure that the experiment results are not from the randomness. The statistics of the above datasets are concluded in Table 2.

In IFRQE++, considering that the space of $\boldsymbol{\alpha}$ can be extremely large, it is less efficient to initialize $\boldsymbol{\alpha}$ completely at random, and blindly learn it in the optimization process. To solve this problem, we initialize $\boldsymbol{\alpha}$ with a prior, assuming that most of the items should be leveraged to train the model for achieving acceptable recommendation performance. In specific, for each $\boldsymbol{\alpha}^u \in \boldsymbol{\alpha}$, we initialize it with a Binomial distribution $p(k, n) = C_n^k s^k (1-s)^{n-k}$, where k is the number of disclosed items, and n is the total number of items in the training set (*i.e.*, $|\mathcal{S}^u|$). Notably, we do not discriminate the item differences in the initialization process of $\boldsymbol{\alpha}^u$. For example, suppose there are three items, then $\boldsymbol{\alpha}_{\{1,1,0\}}^u = \boldsymbol{\alpha}_{\{1,0,1\}}^u = \boldsymbol{\alpha}_{\{0,1,1\}}^u$. In the experiment, we set $s = 0.9$, which means, in the beginning, about $0.9 * |\mathcal{S}^u|$ items will be involved into the model training process.

In order to efficiently compute the inverse of the Hessian matrix, we use the stochastic estimation method discussed in Koh & Liang (2017). In specific, according to the Taylor expansion, we can

Table 4: Parameter settings in the experiments.

Parameter	Tuning range	Simulation	Diginetica	Steam	Amazon Video
Learning rate	$[0.001, 0.01, 0.05]$	0.01	0.01	0.01	0.01
Batch size	$[1024, 2048, 4096]$	2048	2048	2048	2048
Embedding size	$[64, 128, 256]$	64	64	64	64
Drop ratio	$[0.01, 0.1, 0.2]$	0.1	0.1	0.1	0.1
λ	$[0.1, 0.5, 1]$	1	1	1	1
Iteration number M	$[1, 3, 5, 10]$	10	10	10	10
Training epochs	$[50, 100, 150]$	50	50	150	100
L	$[500, 1000, 2000]$	1000	1000	500	1000
T	$[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$	2	8	4	6
N_J for computing H^{-1}	$[10, 20, 30]$	30	10	20	20

express H^{-1} by $\sum_{i=0}^{\infty}(I-H)^i$. Let $H_j^{-1} = \sum_{i=0}^{j}(I-H)^i$, then we have $H_j^{-1} = I+(I-H)H_{j-1}^{-1}$. To compute $H_{\widetilde{\boldsymbol{\theta}}}^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}})$, we uniformly sample a training data s, and approximate H by $\bigtriangledown^2 l_f(s, \widetilde{\boldsymbol{\theta}})$. Then we have the following recursive equation:

$$H_j^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) = \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) + (I - \bigtriangledown^2 l_f(s, \widetilde{\boldsymbol{\theta}}))H_{j-1}^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) \tag{35}$$

Obviously, when $j \to \infty$, we have $H_j^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}}) \to H_{\widetilde{\boldsymbol{\theta}}}^{-1} \bigtriangledown l_f(s_k^v, \widetilde{\boldsymbol{\theta}})$. In the experiment, we resample s for each iteration, and the total number of iterations N_J is tuned to better effectiveness-efficiency trade-off.

For the model parameters, we determine them by grid search. For example, the number of anchor selection vectors is searched in $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$. The learning rate and batch size are determined in the ranges of $[0.001, 0.01, 0.05]$ and $[1024, 2048, 4096]$, respectively. The anchor selection vectors are sampled from the Binomial distribution, where, similar to $\boldsymbol{\alpha}^u$, we set the mean as 0.9. The final parameters used in our experiments are concluded in Table 4. Our project has been released at https://ifrqe.github.io/IFRQE/.

A.9 More Experiments

In this section, we present more experiments to evaluate and analyze our proposed frameworks.

A.9.1 Comparison between our framework and the method of "training from scratch"

In this section, we compare our framework with the method of "training from scratch" (we call it as **SCR**), where we drop the influence function, and for each action exploration, we retrain the recommender model. We remain the other model components of this method the same as our framework. The comparison results are presented in Table 5. We can see, the performances of our framework do not surpass SCR in most cases. This is understandable, since SCR uses the true loss, and our framework only leverages the approximated values. However, we find that the reward gap is not large, which may suggest that our designed influence function can well approximate the true loss, and help to achieve satisfied reward. An important superiority of our framework is the efficiency. As can be seen in the last column of Table 5, we can improve the training efficiency by about 20.3 times. This superiority is very important for the recommender system, which is an on-line service, and has to make quick responses the user feedback.

A.9.2 Influence of the Data Sparsity

In real-world scenarios, recommender systems can be applied in different applications, where the dataset sparsity may vary a lot. In this section, we would like to study whether our methods are consistently competitive for the datasets with different sparsities. In order to flexibly control the sparsity, we conduct this experiment based on the simulation dataset. Since the threshold η controls the hardness of generating the user-item interactions, we tune η in the range of $\{0.1, 0.2, 0.3, 0.4, 0.5\}$

Table 5: Comparison between between our framework and the method of training from scratch (SCR). Results based on base models MF, NeuMF, LightGCN, DIN and CDAE. "()" indicates the standard error. The metrics for evaluating the recommendation performance are percentage values with "%" omitted. For the metrics, ↑ means the larger the better, while ↓ means the lower the better.

Dataset	Diginetica						
Metric	precision↑	NDCG↑	MRR↑	F_1 ↑	wv↓	rewarde↑	time↓
w/ SCR	3.26$_{(.014)}$	13.5$_{(.017)}$	12.6$_{(.023)}$	5.43$_{(.021)}$	2.00$_{(.013)}$	-2.08$_{(.031)}$	69126$_{(.023)}$
w/ IFRQE++	3.14$_{(.018)}$	12.5$_{(.010)}$	11.5$_{(.012)}$	5.23$_{(.028)}$	2.01$_{(.036)}$	-2.08$_{(.032)}$	739$_{(.026)}$
w/ SCR	2.09$_{(.013)}$	7.38$_{(.012)}$	6.37$_{(.013)}$	3.48$_{(.011)}$	2.00$_{(.023)}$	-2.10$_{(.011)}$	21342$_{(.033)}$
w/ IFRQE++	1.82$_{(.028)}$	6.04$_{(.018)}$	5.04$_{(.022)}$	3.03$_{(.009)}$	2.00$_{(.016)}$	-2.09$_{(.032)}$	1443$_{(.026)}$
w/ SCR	1.19$_{(.004)}$	3.77$_{(.027)}$	3.05$_{(.013)}$	1.98$_{(.012)}$	1.99$_{(.023)}$	-2.21$_{(.011)}$	21555$_{(.023)}$
w/ IFRQE++	0.84$_{(.014)}$	2.60$_{(.012)}$	2.07$_{(.032)}$	1.40$_{(.021)}$	2.01$_{(.032)}$	-2.37$_{(.032)}$	1331$_{(.025)}$
w/ SCR	1.56$_{(.014)}$	4.70$_{(.029)}$	3.68$_{(.013)}$	2.60$_{(.019)}$	1.97$_{(.017)}$	-2.22$_{(.037)}$	137494$_{(.033)}$
w/ IFRQE++	1.11$_{(.016)}$	3.28$_{(.015)}$	2.54$_{(.013)}$	1.85$_{(.008)}$	2.00$_{(.026)}$	-2.23$_{(.013)}$	2328$_{(.023)}$
w/ SCR	0.86$_{(.013)}$	2.51$_{(.019)}$	1.94$_{(.025)}$	1.43$_{(.021)}$	2.00$_{(.017)}$	-2.06$_{(.010)}$	7320$_{(.032)}$
w/ IFRQE++	0.85$_{(.018)}$	2.54$_{(.020)}$	1.99$_{(.022)}$	1.41$_{(.023)}$	2.01$_{(.016)}$	-2.07$_{(.032)}$	879$_{(.033)}$
Dataset	Amazon Video Games						
Metric	precision↑	NDCG↑	MRR↑	F_1 ↑	wv↓	rewarde↑	time↓
w/ SCR	1.25$_{(.022)}$	4.50$_{(.027)}$	3.92$_{(.009)}$	2.08$_{(.011)}$	2.32$_{(.023)}$	-2.40$_{(.031)}$	9415$_{(.023)}$
w/ IFRQE++	1.09$_{(.018)}$	3.79$_{(.010)}$	3.24$_{(.021)}$	1.82$_{(.023)}$	2.35$_{(.016)}$	-2.42$_{(.012)}$	1021$_{(.036)}$
w/ SCR	1.09$_{(.007)}$	3.34$_{(.017)}$	2.65$_{(.013)}$	1.81$_{(.021)}$	2.32$_{(.023)}$	-2.52$_{(.011)}$	14943$_{(.033)}$
w/ IFRQE++	0.86$_{(.028)}$	2.64$_{(.011)}$	2.09$_{(.012)}$	1.43$_{(.038)}$	2.35$_{(.016)}$	-2.50$_{(.023)}$	3192$_{(.026)}$
w/ SCR	1.23$_{(.0245}$	4.43$_{(.027)}$	3.87$_{(.015)}$	2.04$_{(.022)}$	2.31$_{(.023)}$	-2.68$_{(.031)}$	18492$_{(.023)}$
w/ IFRQE++	1.19$_{(.019)}$	4.29$_{(.014)}$	3.74$_{(.019)}$	1.98$_{(.018)}$	2.34$_{(.031)}$	-2.63$_{(.019)}$	2148$_{(.026)}$
w/ SCR	1.57$_{(.029)}$	5.71$_{(.017)}$	5.31$_{(.015)}$	3.07$_{(.022)}$	2.00$_{(.023)}$	-2.07$_{(.031)}$	161501$_{(.033)}$
w/ IFRQE++	1.59$_{(.019)}$	5.94$_{(.011)}$	5.59$_{(.022)}$	3.11$_{(.021)}$	2.01$_{(.016)}$	-2.06$_{(.023)}$	2743$_{(.013)}$
w/ SCR	0.97$_{(.025)}$	2.97$_{(.019)}$	2.35$_{(.029)}$	1.61$_{(.009)}$	2.31$_{(.013)}$	-2.52$_{(.021)}$	1045387$_{(.023)}$
w/ IFRQE++	1.13$_{(.010)}$	3.64$_{(.017)}$	2.99$_{(.011)}$	1.88$_{(.021)}$	2.34$_{(.016)}$	-2.44$_{(.027)}$	579$_{(.013)}$

to build the datasets with different densities, where larger η can lead more sparse dataset. The statistics of the generated datasets are presented in Table 3. In Figure 4, we report the performance of different models based on the reward, where we can see: the performance of the base model is not satisfied in most cases. IFRQE usually outperforms the random method, although there are a few exceptions. IFRQE++ can always achieves the best performance, which is consistent on all the base models and datasets with different sparsities. These results demonstrate the robustness of our model, and suggests that it can be potentially applied to a wide range of real-world applications.

A.9.3 INFLUENCE OF THE BALANCING PARAMETER λ

In the reward function, λ balances the importances of the recommendation quality and user disclosing willingness. To study whether our model can adaptively trade-off the above two aspects, we specify λ with different values, and observe whether our model can always achieve better performance than the baselines. In specific, we set λ as 0.1, 0.5, 1.0 and 2.0 respectively, and the results of comparing our models with the baselines are presented in Table 6. We can see: on different datasets, because the base model completely ignores the user disclosing willingness, the overall reward is the worst comparing with the other methods. Blindly integrating the user disclosing willingness is also sub-optimal, which is evidenced by the lower performance of the random method. By designing a principled model to optimize the overall reward, IFRQE can achieve better performance than the base and random models in most cases. As expected, by leveraging more anchor selection vectors to simulate the validation loss, the final model IFRQE++ achieves the best performance. The above observations are consistent for different λ's, which manifests that our model is robust to the predefined relative importance between the recommendation quality and user disclosing willingness.

A.9.4 COMPLETE RESULTS FOR SECTION 5.3 AND 5.4 IN THE MAIN PAPER

To begin with, we present the complete results of the experiments in section 5.3 of the main paper. From the results shown in Figure 5, we can see: similar to the results in the main paper, the validation loss can be in general well approximated in most cases. IFRQE++ can achieve better

Table 6: Comparison between different models with different λ's. We use bold fonts to label the best performance for each dataset, evaluation metric and base model. "()" indicates the standard error.

Dataset	Simulation	Diginetica	Steam	Amazon Video
$\lambda = 0.1$				
MF	-0.34$_{(.002)}$	-0.29$_{(.009)}$	-0.49$_{(.005)}$	-0.32$_{(.007)}$
w/ Random	-0.46$_{(.005)}$	-0.28$_{(.017)}$	-0.48$_{(.019)}$	-0.31$_{(.011)}$
w/ Threshold	-0.48$_{(.015)}$	-0.48$_{(.022)}$	-0.50$_{(.017)}$	-0.30$_{(.018)}$
w/ Proactive	-0.33$_{(.034)}$	-0.28$_{(.025)}$	-0.31$_{(.007)}$	-0.36$_{(.019)}$
w/ IFRQE	-0.34$_{(.007)}$	-0.23$_{(.011)}$	-0.49$_{(.013)}$	-0.30$_{(.008)}$
w/ IFRQE++	**-0.32**$_{(.006)}$	**-0.22**$_{(.005)}$	**-0.47**$_{(.006)}$	**-0.28**$_{(.002)}$
NeuMF	-0.40$_{(.001)}$	-0.30$_{(.002)}$	-0.37$_{(.009)}$	-0.33$_{(.007)}$
w/ Random	-0.39$_{(.012)}$	-0.31$_{(.010)}$	-0.36$_{(.015)}$	-0.34$_{(.006)}$
w/ Threshold	-0.38$_{(.019)}$	-0.38$_{(.012)}$	-0.35$_{(.007)}$	-0.41$_{(.014)}$
w/ Proactive	-0.49$_{(.027)}$	-0.35$_{(.012)}$	-0.37$_{(.007)}$	-0.35$_{(.019)}$
w/ IFRQE	-0.37$_{(.005)}$	-0.34$_{(.002)}$	-0.25$_{(.014)}$	-0.37$_{(.012)}$
w/ IFRQE++	**-0.34**$_{(.007)}$	**-0.28**$_{(.003)}$	**-0.24**$_{(.005)}$	**-0.32**$_{(.002)}$
LightGCN	-0.32$_{(.011)}$	-0.28$_{(.016)}$	-0.36$_{(.013)}$	-0.49$_{(.011)}$
w/ Random	-0.30$_{(.013)}$	-0.27$_{(.022)}$	-0.39$_{(.019)}$	-0.47$_{(.016)}$
w/ Threshold	-0.36$_{(.021)}$	-0.33$_{(.030)}$	-0.35$_{(.026)}$	-0.34$_{(.011)}$
w/ Proactive	-0.29$_{(.034)}$	-0.51$_{(.025)}$	-0.34$_{(.007)}$	-0.25$_{(.019)}$
w/ IFRQE	-0.26$_{(.005)}$	-0.27$_{(.014)}$	-0.34$_{(.011)}$	-0.25$_{(.003)}$
w/ IFRQE++	**-0.25**$_{(.006)}$	**-0.25**$_{(.016)}$	**-0.33**$_{(.008)}$	**-0.24**$_{(.004)}$
$\lambda = 0.5$				
MF	-1.19$_{(.006)}$	-1.10$_{(.012)}$	-1.54$_{(.011)}$	-1.27$_{(.016)}$
w/ Random	-1.28$_{(.016)}$	-1.08$_{(.007)}$	-1.52$_{(.019)}$	-1.21$_{(.018)}$
w/ Threshold	-1.32$_{(.021)}$	-1.44$_{(.030)}$	-1.37$_{(.026)}$	-1.21$_{(.011)}$
w/ Proactive	-1.18$_{(.028)}$	-1.08$_{(.025)}$	-1.44$_{(.007)}$	-0.36$_{(.019)}$
w/ IFRQE	-0.97$_{(.003)}$	-0.85$_{(.007)}$	-0.77$_{(.006)}$	-1.04$_{(.008)}$
w/ IFRQE++	**-0.95**$_{(.001)}$	**-0.81**$_{(.006)}$	**-0.76**$_{(.001)}$	**-1.03**$_{(.004)}$
NeuMF	-1.11$_{(.009)}$	-1.11$_{(.002)}$	-1.42$_{(.006)}$	-1.27$_{(.009)}$
w/ Random	-1.22$_{(.006)}$	-1.10$_{(.010)}$	-1.40$_{(.013)}$	-1.22$_{(.017)}$
w/ Threshold	-1.32$_{(.021)}$	-1.44$_{(.030)}$	-1.37$_{(.026)}$	-1.23$_{(.011)}$
w/ Proactive	-1.21$_{(.017)}$	-1.29$_{(.025)}$	-1.30$_{(.007)}$	-0.36$_{(.019)}$
w/ IFRQE	-0.93$_{(.011)}$	-1.01$_{(.002)}$	-1.30$_{(.005)}$	-1.26$_{(.004)}$
w/ IFRQE++	**-0.91**$_{(.006)}$	**-0.98**$_{(.003)}$	**-1.28**$_{(.011)}$	**-1.20**$_{(.007)}$
LightGCN	-1.17$_{(.011)}$	-1.09$_{(.016)}$	-1.41$_{(.009)}$	-1.48$_{(.006)}$
w/ Random	-1.11$_{(.016)}$	-1.06$_{(.022)}$	-1.43$_{(.012)}$	-1.38$_{(.019)}$
w/ Threshold	-1.32$_{(.021)}$	-1.44$_{(.030)}$	-1.74$_{(.026)}$	-1.75$_{(.011)}$
w/ Proactive	-1.20$_{(.034)}$	-1.54$_{(.033)}$	-1.19$_{(.007)}$	-1.70$_{(.019)}$
w/ IFRQE	-1.04$_{(.007)}$	-0.97$_{(.014)}$	-1.41$_{(.003)}$	-1.33$_{(.006)}$
w/ IFRQE++	**-1.02**$_{(.001)}$	**-0.96**$_{(.016)}$	**-1.40**$_{(.008)}$	**-1.29**$_{(.005)}$
$\lambda = 1.0$				
MF	-2.25$_{(.032)}$	-2.65$_{(.072)}$	-2.99$_{(.093)}$	-2.43$_{(.022)}$
w/ Random	-2.13$_{(.014)}$	-2.04$_{(.006)}$	-2.82$_{(.015)}$	-2.30$_{(.023)}$
w/ Threshold	-2.20$_{(.011)}$	-2.06$_{(.013)}$	-2.57$_{(.023)}$	-2.20$_{(.015)}$
w/ Proactive	-2.25$_{(.034)}$	-2.14$_{(.012)}$	-2.43$_{(.007)}$	-2.08$_{(.019)}$
w/ IFRQE	-2.09$_{(.017)}$	-2.18$_{(.006)}$	-2.43$_{(.015)}$	-2.10$_{(.005)}$
w/ IFRQE++	**-1.92**$_{(.012)}$	**-1.92**$_{(.022)}$	**-2.42**$_{(.014)}$	**-1.98**$_{(.056)}$
NeuMF	-2.32$_{(.011)}$	-2.11$_{(.003)}$	-2.76$_{(.007)}$	-2.47$_{(.010)}$
w/ Random	-2.19$_{(.011)}$	-2.11$_{(.016)}$	-2.59$_{(.011)}$	-2.41$_{(.010)}$
w/ Threshold	-2.18$_{(.012)}$	-2.08$_{(.013)}$	-2.86$_{(.011)}$	-2.21$_{(.025)}$
w/ Proactive	-2.18$_{(.034)}$	-2.14$_{(.015)}$	-2.42$_{(.022)}$	-2.41$_{(.019)}$
w/ IFRQE	-2.48$_{(.013)}$	-2.17$_{(.019)}$	-2.45$_{(.027)}$	-2.24$_{(.012)}$
w/ IFRQE++	**-2.11**$_{(.011)}$	**-2.08**$_{(.023)}$	**-2.37**$_{(.027)}$	**-2.17**$_{(.011)}$
LightGCN	-2.82$_{(.012)}$	-2.71$_{(.009)}$	-3.13$_{(.014)}$	-3.05$_{(.007)}$
w/ Random	-2.71$_{(.022)}$	-2.64$_{(.007)}$	-3.07$_{(.019)}$	-2.93$_{(.015)}$
w/ Threshold	-2.70$_{(.008)}$	-2.60$_{(.031)}$	-2.82$_{(.022)}$	-2.30$_{(.021)}$
w/ Proactive	-2.47$_{(.034)}$	-2.11$_{(.013)}$	-2.81$_{(.027)}$	-2.64$_{(.039)}$
w/ IFRQE	-2.82$_{(.014)}$	-2.65$_{(.013)}$	-2.82$_{(.008)}$	-2.13$_{(.013)}$
w/ IFRQE++	**-2.44**$_{(.016)}$	**-2.55**$_{(.009)}$	**-2.80**$_{(.005)}$	**-2.03**$_{(.023)}$
$\lambda = 2.0$				
MF	-4.53$_{(.011)}$	-4.23$_{(.052)}$	-5.46$_{(.013)}$	-4.56$_{(.012)}$
w/ Random	-4.30$_{(.024)}$	-4.01$_{(.016)}$	-5.18$_{(.025)}$	-4.64$_{(.023)}$
w/ Threshold	-4.34$_{(.009)}$	-3.72$_{(.027)}$	-4.92$_{(.023)}$	-4.32$_{(.025)}$
w/ Proactive	-4.37$_{(.014)}$	-4.10$_{(.021)}$	-4.55$_{(.027)}$	-4.09$_{(.029)}$
w/ IFRQE	-4.12$_{(.017)}$	-3.24$_{(.026)}$	-4.05$_{(.015)}$	-3.71$_{(.043)}$
w/ IFRQE++	**-4.06**$_{(.010)}$	**-3.16**$_{(.032)}$	**-4.02**$_{(.014)}$	**-3.37**$_{(.056)}$
NeuMF	-4.44$_{(.012)}$	-4.12$_{(.043)}$	-5.34$_{(.027)}$	-4.81$_{(.019)}$
w/ Random	-4.21$_{(.012)}$	-4.05$_{(.035)}$	-5.07$_{(.021)}$	-4.58$_{(.031)}$
w/ Threshold	-4.38$_{(.012)}$	-3.74$_{(.013)}$	-4.81$_{(.011)}$	-4.33$_{(.035)}$
w/ Proactive	-4.65$_{(.024)}$	-3.77$_{(.025)}$	-4.62$_{(.007)}$	-4.18$_{(.009)}$
w/ IFRQE	-4.56$_{(.013)}$	-4.00$_{(.019)}$	-3.90$_{(.028)}$	-3.06$_{(.022)}$
w/ IFRQE++	**-4.00**$_{(.012)}$	**-3.67**$_{(.021)}$	**-3.40**$_{(.019)}$	**-2.74**$_{(.012)}$
LightGCN	-4.94$_{(.021)}$	-4.10$_{(.009)}$	-5.75$_{(.024)}$	-5.20$_{(.057)}$
w/ Random	-4.70$_{(.032)}$	-3.90$_{(.007)}$	-5.07$_{(.029)}$	-4.73$_{(.033)}$
w/ Threshold	-4.36$_{(.017)}$	-4.32$_{(.031)}$	-5.21$_{(.022)}$	-4.93$_{(.041)}$
w/ Proactive	-3.63$_{(.035)}$	-4.72$_{(.039)}$	-4.54$_{(.007)}$	-4.71$_{(.011)}$
w/ IFRQE	-3.89$_{(.013)}$	-2.54$_{(.013)}$	-3.90$_{(.018)}$	-3.71$_{(.023)}$
w/ IFRQE++	**-3.55**$_{(.026)}$	**-2.52**$_{(.009)}$	**-3.40**$_{(.025)}$	**-3.69**$_{(.033)}$

approximation accuracy than IFRQE, which demonstrate the effectiveness of using more anchor vectors for computing the validation loss. In Figure 6, we show the complete results of the experiments in section 5.4. We can see: the reward changing patterns seem to be quite diverse as more anchor

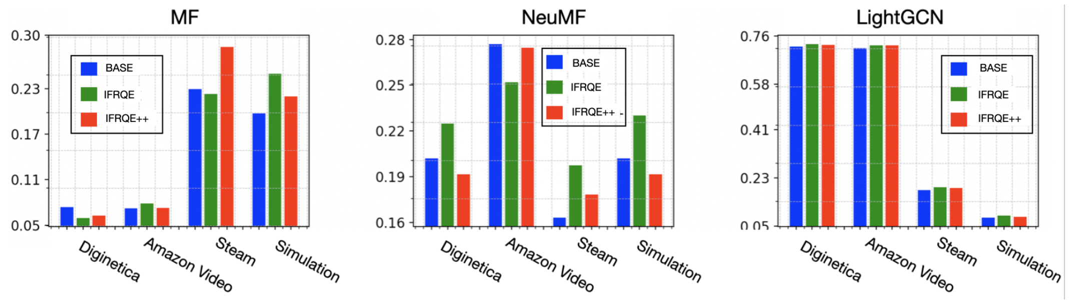

Figure 5: Approximation error on the validation loss for all the datasets and base models.

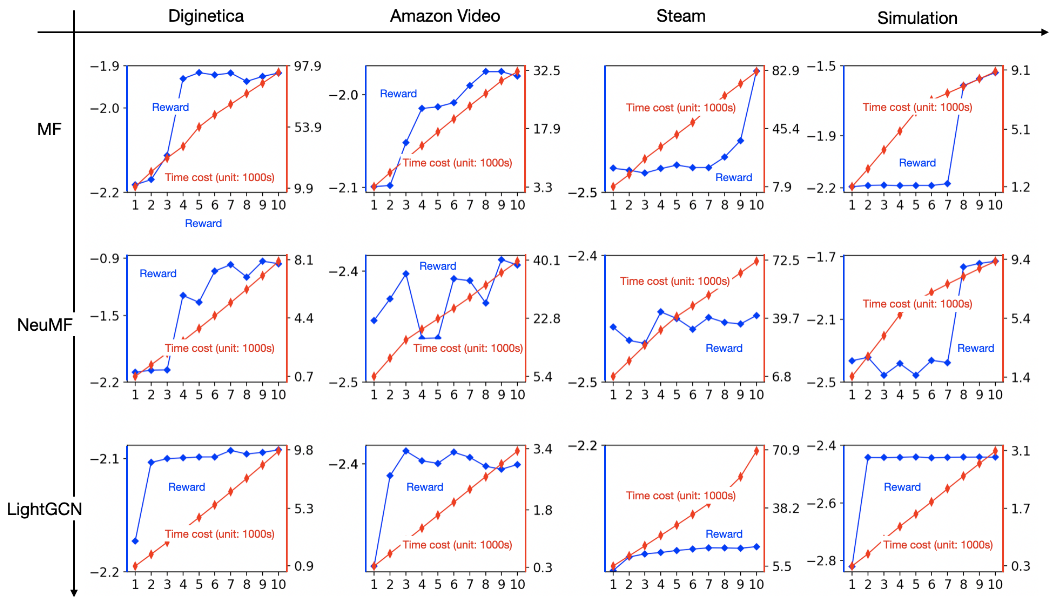

Figure 6: Influence of T for all the datasets and base models.

selection vectors are leveraged in our model. For example, in the case of MF + Diginetica, the reward has a performance jump from $T = 3$ to $T = 4$. Similar performance jumping patterns can also be observed in the settings of NeuMF + Diginetica, LightGCN + Diginetica, LightGCN + Amazon Video and the simulation dataset. However, in the case of NeuMF + Amazon Video, the reward changes irregularly as T becomes larger. While different combinations between the base model and dataset may lead to various performance change patterns, a common phenomenon is that, in most cases, the performance tends to be better as more anchor selection vectors. Simultaneously, the time cost is increased almost linearly as more anchor vectors are deployed to achieve better performance. These observations are aligned with the conclusions in the main paper.

A.9.5 Overall comparison with more performance evaluation

To begin with, we augment Table 1 in the main paper by reporting the recommendation performance based on Precision, NDCG and MRR. From the results shown in Table 7, we can draw similar conclusions as Table 1, that is, there are many cases that, although we have removed some items due to the user willingness, the recommendation performances are not lowered.

Table 7: Additional metrics for evaluating the recommendation performance. We use "P" to represent the precision. All the results are percentage values with "%" omitted.

Dataset	Simulation			Diginetica			Steam			Amazon Video		
Metric	P↑	NDCG↑	MRR↑	P↑	NDCG↑	MRR↑	P↑	NDCG↑	MRR↑	P↑	NDCG↑	MRR↑
MF	0.39$_{(.022)}$	1.02$_{(.017)}$	0.72$_{(.023)}$	2.83$_{(.012)}$	11.3$_{(.017)}$	10.3$_{(.023)}$	6.21$_{(.043)}$	19.5$_{(.014)}$	15.8$_{(.093)}$	**1.03**$_{(.012)}$	**3.52**$_{(.021)}$	**2.99**$_{(.022)}$
w/ Random	**0.41**$_{(.031)}$	**1.11**$_{(.056)}$	0.80$_{(.024)}$	2.51$_{(.031)}$	9.88$_{(.006)}$	9.00$_{(.014)}$	5.64$_{(.019)}$	17.7$_{(.017)}$	14.3$_{(.015)}$	0.92$_{(.011)}$	3.10$_{(.022)}$	2.61$_{(.023)}$
w/ Threshold	0.34$_{(.036)}$	1.07$_{(.022)}$	**1.01**$_{(.011)}$	2.80$_{(.022)}$	11.0$_{(.013)}$	9.98$_{(.042)}$	5.55$_{(.028)}$	17.7$_{(.033)}$	14.4$_{(.027)}$	0.90$_{(.026)}$	2.86$_{(.037)}$	2.33$_{(.041)}$
w/ Proactive	0.37$_{(.015)}$	1.01$_{(.021)}$	0.93$_{(.013)}$	1.33$_{(.005)}$	4.73$_{(.023)}$	4.10$_{(.042)}$	5.96$_{(.011)}$	19.4$_{(.013)}$	16.0$_{(.009)}$	0.93$_{(.015)}$	3.15$_{(.017)}$	2.66$_{(.031)}$
w/ IFRQE	0.37$_{(.025)}$	**1.11**$_{(.033)}$	0.87$_{(.017)}$	**3.14**$_{(.025)}$	**12.5**$_{(.033)}$	**11.5**$_{(.017)}$	**6.23**$_{(.012)}$	**19.8**$_{(.012)}$	**16.1**$_{(.015)}$	0.34$_{(.015)}$	1.28$_{(.016)}$	1.14$_{(.005)}$
w/ IFRQE++	0.33$_{(.027)}$	1.02$_{(.013)}$	0.82$_{(.012)}$	2.41$_{(.007)}$	9.46$_{(.013)}$	8.21$_{(.012)}$	5.90$_{(.017)}$	18.9$_{(.032)}$	15.4$_{(.014)}$	**1.03**$_{(.012)}$	3.37$_{(.032)}$	2.78$_{(.056)}$
NeuMF	0.52$_{(.037)}$	1.40$_{(.003)}$	1.02$_{(.011)}$	**2.53**$_{(.037)}$	**10.5**$_{(.003)}$	**9.83**$_{(.011)}$	**6.12**$_{(.007)}$	**19.5**$_{(.012)}$	**15.9**$_{(.007)}$	0.99$_{(.014)}$	3.38$_{(.013)}$	2.87$_{(.010)}$
w/ Random	0.54$_{(.019)}$	1.71$_{(.015)}$	1.39$_{(.012)}$	1.43$_{(.019)}$	5.04$_{(.015)}$	4.34$_{(.012)}$	4.64$_{(.005)}$	14.6$_{(.019)}$	11.5$_{(.011)}$	1.08$_{(.014)}$	3.63$_{(.006)}$	3.04$_{(.010)}$
w/ Threshold	0.90$_{(.022)}$	4.02$_{(.028)}$	3.86$_{(.011)}$	2.43$_{(.011)}$	9.74$_{(.044)}$	8.94$_{(.016)}$	4.50$_{(.029)}$	14.2$_{(.022)}$	11.5$_{(.021)}$	0.93$_{(.012)}$	3.03$_{(.028)}$	2.49$_{(.017)}$
w/ Proactive	1.10$_{(.033)}$	4.47$_{(.022)}$	**4.13**$_{(.012)}$	0.78$_{(.017)}$	2.31$_{(.043)}$	1.79$_{(.011)}$	5.73$_{(.028)}$	17.8$_{(.032)}$	14.3$_{(.012)}$	0.86$_{(.006)}$	2.82$_{(.021)}$	2.33$_{(.039)}$
w/ IFRQE	0.84$_{(.014)}$	2.73$_{(.023)}$	2.26$_{(.033)}$	1.75$_{(.004)}$	7.27$_{(.023)}$	2.99$_{(.033)}$	4.90$_{(.023)}$	14.6$_{(.019)}$	11.4$_{(.027)}$	**1.10**$_{(.015)}$	**3.77**$_{(.014)}$	**3.39**$_{(.012)}$
w/ IFRQE++	**1.80**$_{(.015)}$	**6.34**$_{(.003)}$	3.77$_{(.021)}$	1.80$_{(.015)}$	6.34$_{(.003)}$	3.77$_{(.021)}$	5.32$_{(.021)}$	16.7$_{(.014)}$	13.4$_{(.027)}$	0.72$_{(.031)}$	2.65$_{(.014)}$	2.48$_{(.011)}$
LightGCN	0.54$_{(.025)}$	1.41$_{(.008)}$	1.01$_{(.042)}$	**4.51**$_{(.019)}$	**20.2**$_{(.010)}$	**19.5** $_{(.009)}$	6.04$_{(.010)}$	18.8$_{(.009)}$	15.1$_{(.014)}$	**1.29**$_{(.021)}$	**4.90**$_{(.008)}$	**4.40**$_{(.007)}$
w/ Random	0.49$_{(.013)}$	1.27$_{(.013)}$	0.89$_{(.022)}$	4.50$_{(.013)}$	20.0$_{(.013)}$	19.2$_{(.022)}$	6.03$_{(.027)}$	18.8$_{(.026)}$	15.1$_{(.019)}$	1.05$_{(.003)}$	3.84$_{(.009)}$	3.37$_{(.015)}$
w/ Threshold	0.58$_{(.038)}$	2.25$_{(.018)}$	2.04$_{(.018)}$	2.43$_{(.027)}$	9.74$_{(.041)}$	8.94$_{(.021)}$	5.79$_{(.049)}$	18.1$_{(.027)}$	14.6$_{(.036)}$	0.97$_{(.013)}$	3.28$_{(.023)}$	2.78$_{(.048)}$
w/ Proactive	**0.61**$_{(.011)}$	**2.42**$_{(.023)}$	**2.22**$_{(.013)}$	3.31$_{(.026)}$	14.1$_{(.014)}$	13.3$_{(.011)}$	6.21$_{(.019)}$	**20.6**$_{(.017)}$	**17.2**$_{(.028)}$	0.77$_{(.015)}$	2.48$_{(.021)}$	2.04$_{(.038)}$
w/ IFRQE	0.58$_{(.008)}$	1.60$_{(.010)}$	1.18$_{(.014)}$	2.93$_{(.008)}$	10.9$_{(.010)}$	9.74$_{(.014)}$	**6.21**$_{(.016)}$	19.8$_{(.025)}$	**16.1**$_{(.008)}$	1.14$_{(.007)}$	3.98$_{(.018)}$	3.42$_{(.013)}$
w/ IFRQE++	0.45$_{(.022)}$	1.21$_{(.008)}$	0.87$_{(.016)}$	3.16$_{(.022)}$	12.1$_{(.008)}$	10.9$_{(.016)}$	6.15$_{(.018)}$	19.7$_{(.018)}$	16.0$_{(.005)}$	0.97$_{(.013)}$	3.44$_{(.009)}$	2.98$_{(.023)}$
DIN	**0.86**$_{(.022)}$	**2.76**$_{(.017)}$	**2.25**$_{(.023)}$	2.48$_{(.023)}$	8.30$_{(.005)}$	6.95$_{(.031)}$	7.01$_{(.019)}$	23.5$_{(.048)}$	19.7$_{(.005)}$	**1.38**$_{(.038)}$	**5.38**$_{(.012)}$	**4.39**$_{(.026)}$
w/ Random	0.74$_{(.031)}$	2.30$_{(.016)}$	1.84$_{(.024)}$	**2.60**$_{(.016)}$	**8.65**$_{(.042)}$	**7.22**$_{(.024)}$	6.54$_{(.030)}$	21.4$_{(.042)}$	17.7$_{(.014)}$	1.27$_{(.018)}$	3.78$_{(.023)}$	2.95$_{(.015)}$
w/ Threshold	0.82$_{(.004)}$	2.30$_{(.022)}$	1.73$_{(.029)}$	2.24$_{(.010)}$	7.45$_{(.032)}$	6.22$_{(.036)}$	5.79$_{(.036)}$	24.9$_{(.045)}$	20.8$_{(.029)}$	1.26$_{(.009)}$	3.84$_{(.036)}$	3.03$_{(.022)}$
w/ Proactive	0.73$_{(.012)}$	2.14$_{(.024)}$	1.65$_{(.018)}$	1.64$_{(.015)}$	5.05$_{(.017)}$	4.03$_{(.019)}$	7.60$_{(.029)}$	25.2$_{(.035)}$	21.1$_{(.025)}$	0.98$_{(.020)}$	3.03$_{(.021)}$	2.43$_{(.027)}$
w/ IFRQE	0.72$_{(.027)}$	2.48$_{(.013)}$	2.11$_{(.012)}$	2.35$_{(.017)}$	8.04$_{(.047)}$	6.82$_{(.022)}$	**7.93**$_{(.021)}$	**26.6**$_{(.030)}$	**22.3**$_{(.041)}$	0.74$_{(.034)}$	2.47$_{(.019)}$	2.04$_{(.043)}$
w/ IFRQE++	0.62$_{(.027)}$	1.73$_{(.013)}$	1.27$_{(.012)}$	2.10$_{(.014)}$	6.78$_{(.042)}$	5.56$_{(.030)}$	5.72$_{(.030)}$	17.5$_{(.016)}$	13.9$_{(.045)}$	1.09$_{(.021)}$	3.47$_{(.047)}$	2.83$_{(.038)}$
CDAE	**0.74**$_{(.010)}$	2.73$_{(.015)}$	2.41$_{(.031)}$	**0.89**$_{(.009)}$	2.65$_{(.016)}$	2.06$_{(.042)}$	6.91$_{(.008)}$	21.8$_{(.029)}$	17.6$_{(.014)}$	0.67$_{(.011)}$	2.01$_{(.027)}$	1.58$_{(.045)}$
w/ Random	**0.74**$_{(.013)}$	**2.87**$_{(.013)}$	**2.60**$_{(.022)}$	0.83$_{(.014)}$	2.63$_{(.042)}$	**2.13**$_{(.030)}$	**6.96**$_{(.021)}$	**22.0**$_{(.003)}$	**17.8**$_{(.041)}$	0.73$_{(.011)}$	2.27$_{(.015)}$	1.82$_{(.031)}$
w/ Threshold	0.36$_{(.021)}$	1.05$_{(.003)}$	0.81$_{(.041)}$	0.83$_{(.030)}$	2.61$_{(.026)}$	2.11$_{(.006)}$	5.34$_{(.045)}$	16.9$_{(.044)}$	13.8$_{(.010)}$	0.73$_{(.016)}$	2.27$_{(.040)}$	1.82$_{(.015)}$
w/ Proactive	0.98$_{(.014)}$	3.03$_{(.024)}$	2.43$_{(.028)}$	0.71$_{(.017)}$	2.11$_{(.039)}$	1.64$_{(.007)}$	6.76$_{(.026)}$	21.2$_{(.033)}$	17.1$_{(.014)}$	0.98$_{(.012)}$	3.03$_{(.031)}$	2.43$_{(.022)}$
w/ IFRQE	0.56$_{(.011)}$	0.87$_{(.015)}$	0.69$_{(.027)}$	**0.89**$_{(.011)}$	**2.68**$_{(.032)}$	2.11$_{(.013)}$	6.77$_{(.008)}$	21.2$_{(.025)}$	17.1$_{(.023)}$	1.08$_{(.027)}$	3.62$_{(.029)}$	**3.03**$_{(.046)}$
w/ IFRQE++	0.52$_{(.037)}$	0.91$_{(.005)}$	0.79$_{(.012)}$	0.86$_{(.007)}$	2.58$_{(.019)}$	2.03$_{(.042)}$	6.85$_{(.018)}$	21.4$_{(.046)}$	17.2$_{(.032)}$	**1.10**$_{(.027)}$	**3.63**$_{(.029)}$	3.00$_{(.046)}$