# SUPPLEMENTARY MATERIAL TO THE PAPER ENTITLED "PREVENTING THE POPULAR ITEM EMBEDDING BASED ATTACK IN FEDERATED RECOMMENDATIONS"

## A. Details of the Experiment Settings

*1) Datasets:* We use three datasets for evaluation, namely MovieLens-100K (ML-100K) [12], MovieLens-1M (ML-1M) [12], and Amazon Digital Music (AZ) [13]. The statistics are shown in Table IX, where AZ features a lower interaction rate compared to the other two while all datasets from real-world scenarios exhibit high sparsity. Such sparsity typically makes it challenging for FRS to accurately learn users' preferences and achieve effective recommendations. Following a previous study [17], for each user, we adopt the leave-one-out method to create the training and test sets.

TABLE IX: Dataset statistics ('Rate' equals #(Interactions)/ #(Users) and 'Sparsity' equals $1 - \#(\text{Interactions})/(\#(\text{Users}) \times \#(\text{Items}))$).

| Dataset | #(Users) | #(Items) | #(Interactions) | Rate | Sparsity |
|---|---|---|---|---|---|
| ML-100K | 943 | 1,682 | 100,000 | 106 | 93.70% |
| ML-1M | 6,040 | 3,706 | 1,000,209 | 166 | 95.53% |
| AZ | 16,566 | 11,797 | 169,781 | 10 | 99.91% |

*2) System Settings:* To confirm the model-agnostic nature of PIECK, we adopt Matrix Factorization [30] for MF-FRS and Neural Collaborative Filtering (NCF) [16] for DL-FRS as the underlying base models. Despite the existence of various dedicated recommender models with intricate components, the above two models have gained widespread adoption in practice and have been extended to the federated setting [31], [32], [42]. Their appeal lies in their ability to ensure high generalization and low communication overhead, making them highly representative choices for our evaluation. In our setting, each user in the dataset is regarded as a 'client' in the federation. The batch size of randomly selected users per round for MF-FRS is 256 on ML-100K and ML-1M datasets and 1024 on AZ. For DL-FRS, the batch size remains 256 for all datasets. We train MF-FRS and DL-FRS with learning rates $\eta = 1.0$ and $\eta = 0.005$, where corresponding model parameters are set the same as previous studies [31], [32]. Other parameter settings, related to the popular item mining algorithm and the defense loss can be found in our instruction [1].

*3) Attack Baselines:* We compare PIECK with the following state-of-the-art model poisoning attacks:

- FEDRECA [32] accesses a *portion* of benign users' historical interactions to approximate their embeddings.
- PIPA [42] involves explicit promotion and popularity enhancement for target items.
- A-RA [31] randomly initializes users' embeddings at malicious users, specifically designed for DL-FRS.
- A-HUM [31] extends A-RA by enhancing the attack based on mining hard users for increased effectiveness.

For a fair comparison without utilizing prior knowledge, we mask the historical interactions for FEDRECA and popularity levels of items for PIPA in the implementation. Moreover, we set null parameters for A-RA when applying it to MF-FRS.

*4) Defense Methods:* To evaluate PIECK and compare our proposed defense method, we apply different defense methods to the aggregation function $\text{Agg}(\cdot)$ on the server and tune them *optimal*. They process the uploaded gradients as follows.

- NORMBOUND [33]: A thresholding approach is employed to bound the $L_2$ Norm of all gradients uploaded by users.
- MEDIAN [40]: The median of received gradients for each dimension is computed.
- TRIMMEDMEAN [40]: The $\tilde{p}$ largest and smallest values for each dimension are removed, and the rest are averaged.
- KRUM [5]: The most similar gradient from received gradients in the squared Euclidean norm space is selected.
- MULTIKRUM [5]: The $(2\tilde{p})$ least similar gradients produced by KRUM are removed iteratively with the rest averaged.
- BULYAN [25]: Gradients are selected by MULTIKRUM and then averaged using TRIMMEDMEAN.

*5) Evaluation Metrics:* In attacking and defending an FRS, both the **effectiveness of attack** and the **recommendation performance** are crucial considerations. An effective attack should successfully promote target items without significantly degrading the performance of the system to avoid detection.

To assess the attack effectiveness in promoting target items, we utilize the *Exposure Ratio at rank K* (ER@$K$) as defined in Eq. (3). Attacks like PIPA and A-HUM specifically focus on the least popular items, which allows for more significant promotion opportunities since the server receives fewer benign gradients. However, for fairness, we follow FEDRECA [32] and randomly select target items to $T$ from the set of uninteracted items. To measure the recommendation performance, we employ the *Hit Ratio at rank K* (HR@$K$) following the NCF approach [16]. A higher HR@$K$ indicates more accurate recommendations of the FRS.

## B. Effect of Sampling Ratio $q$

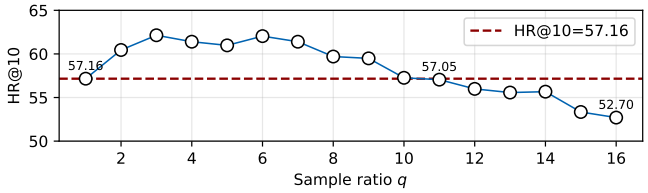


Fig. 7: The effect of sample ratio $q$ of MF-FRS on ML-100K.

Here, we discuss the effect of varying the sampling ratio $q$ on the recommendation performance. A case study is conducted on the MF-FRS using the ML-100K dataset, in line with the primary experiments. Our analysis reveals that the sampling ratio, $q$, critically influences the recommendation performance, with HR@10 peaking at intermediate values of $q$. When $q$ is increased from its lowest value, we observe an improvement in recommendation quality. This improvement suggests that a slight increase in $q$ enhances the algorithm's ability to predict by providing a more comprehensive capture of user-item interactions. However, this upward trend in performance plateaus and subsequently inverts beyond a

TABLE X: Effect of $|T|$ on proposed attacks and defense on MF-FRS and ML-100K.

| Attacks | Defenses | Train Together | | | | | | | | Train One Then Copy | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $\|T\|=2$ | | $\|T\|=3$ | | $\|T\|=4$ | | $\|T\|=5$ | | $\|T\|=3$ | | $\|T\|=5$ | |
| | | ER@10 | HR@10 | ER@10 | HR@10 | ER@10 | HR@10 | ER@10 | HR@10 | ER@10 | HR@10 | ER@10 | HR@10 |
| NoAttack | NoDefense | 0.00 | 57.26 | 0.23 | 57.16 | 0.23 | 57.16 | 0.23 | 57.16 | 0.23 | 57.16 | 0.23 | 57.16 |
| PieckIpe | NoDefense | 75.61 | **57.58** | 57.09 | 57.48 | 39.55 | 57.16 | 32.70 | 56.84 | 59.16 | 57.26 | 54.76 | 56.73 |
| PieckIpe | **ours** | 0.80 | 57.48 | 0.46 | **58.01** | 0.05 | 57.90 | 0.02 | **58.22** | 0.21 | 57.58 | 2.33 | 58.01 |
| PieckUea | NoDefense | **92.72** | 57.05 | **87.33** | 57.05 | **85.24** | 56.73 | **81.84** | 56.31 | **93.93** | 57.90 | **97.65** | 57.69 |
| PieckUea | **ours** | 0.00 | 56.2 | 0.00 | 57.69 | 0.00 | **58.11** | 0.13 | **58.22** | 0.50 | **58.43** | 0.34 | **58.32** |

TABLE XI: Effect of inconsistent learning rates on our attacks.

| $\eta_i$ | NoAttack | | PieckIpe | | PieckUea | |
|---|---|---|---|---|---|---|
| | ER@10 | HR@10 | ER@10 | HR@10 | ER@10 | HR@10 |
| 1e-0 | 0.23 | 57.16 | 87.47 | 57.69 | 93.39 | 57.69 |
| 1e-2 | 0.00 | 49.31 | 98.06 | 48.99 | 95.56 | 48.36 |
| 1e-2∼1e-0 | 0.00 | 21.74 | 55.13 | 22.16 | 55.69 | 21.95 |

TABLE XII: Effect of loss function on proposed attacks and defense.

| Attacks | Defenses | BCE | | BPR | |
|---|---|---|---|---|---|
| | | ER@10 | HR@10 | ER@10 | HR@10 |
| NoAttack | NoDefense | 0.23 | 57.16 | 0.00 | 57.26 |
| PieckIpe | NoDefense | 87.47 | **57.69** | 83.14 | **57.48** |
| PieckIpe | **ours** | 1.25 | 56.31 | 4.67 | 54.08 |
| PieckUea | NoDefense | **93.39** | 57.69 | **90.21** | 56.95 |
| PieckUea | **ours** | 0.00 | 55.89 | 0.00 | 53.45 |

critical threshold of $q$, Specifically, when $q$ is equal to or exceeds 11, there is a marked deterioration in HR@10. It is noteworthy that the decline in recommendation quality at high $q$ values is not just suboptimal; it actually falls below the quality at the lowest sampling ratio tested (i.e., $q = 1$, where HR@10 equals 57.16%). This degradation at elevated $q$ values renders the system less appealing to attackers, as the utility of the recommendations is compromised to the extent that the value derived from potential manipulation is outweighed by the intrinsic inefficacy of the system.

In conclusion, our research highlights the importance of finely tuning the sampling ratio $q$ to an optimal, smaller value. This balance is crucial to avoid overfitting while maintaining representativeness, ensuring that the system can provide high-quality recommendations for real-world usage.

### C. Effect of Target Item Number $|T|$

It is interesting to know how the attacks as well as the defense perform when the attacker wants to promote multiple item numbers. Therefore, we vary the number of target items $|T|$ from 2 to 5 and evaluate the system performance on an MF-FRS using the ML-100K dataset. We maintain a constant proportion $\tilde{p}$ of malicious clients at 5% to ensure a fair comparison. The results are presented in Table X, covering two different training strategies in the attack.

We first consider a `Train Together` strategy, where multiple malicious clients are involved ($N = 5$ for PieckIpe and $N = 50$ for PieckUea for optimal performance) and produce poisonous gradients to promote the set of $T$ of target items jointly. For this strategy, the results in Table X shows that our attack achieves relatively high ER@$K$ and our defense is very robust. However, as the $|T|$ increases, the efficacy of the attack diminishes. This decline is attributed to potential interference in updates between different target items, posing challenges for malicious users attempting to optimize multiple poisonous gradients simultaneously. To counteract this, we experimented with segregating the attackers into groups with each focusing on a single item. This required an increase in the percentage of attackers to preserve attack strength, which is less efficient.

Another feasible strategy is to train only one target item while uploading $|T|$ copies of its poisonous gradient to the server, referred to as `Train One Then Copy`. This is a more straightforward and cost-efficient strategy as it eliminates the need for extensive training or a large group of attackers. Experiments were conducted with $|T| = 3, 5$, and with slight adjustments to $N$ for optimal performance ($N = 15$ for PieckIpe and $N = 250$ for PieckUea), while other settings remained consistent with the `Train Together` strategy. The corresponding columns in Table X illustrate that the approach significantly enhances the attack's effectiveness, e.g., the ER@10 for $|T| = 5$ increasing from 81.84% to 97.65%.

Therefore, we decided to apply the `Train One Then Copy` strategy for the primary experiment in Section VI-G.

### D. Discussion on Inconsistent Learning Rates in FRS

We explore the impact of learning rate inconsistency between client and server on proposed attacks using the ML-100K dataset on the MF-FRS. Specifically, we test two different scenarios: 1) clients using a static learning rate $\eta_i = 1e-2$ unlike the server's ($\eta = 1e-0$); 2) clients with dynamic $\eta_i$ values ($1e-2$ to $1e-0$). Their results are reported in rows 2–3 of Table XI. Compared to our standard scenario (Row 1), these two scenarios incur a significant drop in recommendation quality (HR@10), showing the drawbacks of mismatched learning rates. For example, under 'NoAttack', the HR@10 decreases by 7.85% for the fixed inconsistency scenario (Row 2), while the dynamic inconsistency (Row 3) dropped even further by 35.42%. In contrast, the attack effectiveness (ER@10) of PieckIpe* remains robust except in the dynamic rate scenario (Row 3) where performance notably decreased (HR@10 fell to 21.74% and ER@10 of PieckIpe* is about 55%). These results confirm the robustness of our attack in a stable FRS environment (no poorly configured learning rates). Therefore, as long as the recommendation system is well-functioning, its popularity bias will exist (without our defense employed). By leveraging the aggregation operation of FRS, malicious users can exploit the three properties of popular items (cf. Section IV) based on the gradient changes, thereby achieving effective PieckIpe to poison the FRS model.

### E. Discussion on Other Loss Functions in FRS

We further discuss the generalizability of our proposed attacks and defenses to other popular loss functions. We let the client utilize BPR loss [30] to train MF-FRS based on ML-100K and test the performance of attack and defense. As shown in Table XII, our attacks and defenses remain effective under the BPR loss, highlighting the universality of our attack and defense strategies across different loss functions. This universality can be attributed to the underlying optimization objectives shared by various loss functions in collaborative filtering-based recommendation systems. Regardless of whether it is BPR, BCE loss, or other prevalent loss functions, they all fundamentally strive to prompt the recommendation system to allocate higher scores to items with user interactions and lower scores to those without.

Popular items, being accessed by a larger user, offer a distinct gradient change, allowing malicious users to distinguish them from unpopular ones within the FRS. In a similar fashion, our customized defense strategy effectively combats these attacks.