arXiv:2202.06701v1 [cs.IR] 14 Feb 2022

UA-FedRec: Untargeted Attack on Federated News Recommendation

Jingwei Yi[1], Fangzhao Wu[2], Bin Zhu[2], Yang Yu[1], Chao Zhang[1], Guangzhong Sun[1], Xing Xie[2]
[1]University of Science and Technology of China, Hefei 230027, China
[2]Microsoft Research Asia, Beijing 100080, China
{yjw1029,yflyl613}@mail.ustc.edu.cn,gzsun@ustc.edu.cn,{fangzwu,binzhu,xingx}@microsoft.com,zclfe00@gmail.com

ABSTRACT

News recommendation is critical for personalized news distribution. Federated news recommendation enables collaborative model learning from many clients without sharing their raw data. It is promising for privacy-preserving news recommendation. However, the security of federated news recommendation is still unclear. In this paper, we study this problem by proposing an untargeted attack called UA-FedRec. By exploiting the prior knowledge of news recommendation and federated learning, UA-FedRec can effectively degrade the model performance with a small percentage of malicious clients. First, the effectiveness of news recommendation highly depends on user modeling and news modeling. We design a news similarity perturbation method to make representations of similar news farther and those of dissimilar news closer to interrupt news modeling, and propose a user model perturbation method to make malicious user updates in opposite directions of benign updates to interrupt user modeling. Second, updates from different clients are typically aggregated by weighted-averaging based on their sample sizes. We propose a quantity perturbation method to enlarge sample sizes of malicious clients in a reasonable range to amplify the impact of malicious updates. Extensive experiments on two real-world datasets show that UA-FedRec can effectively degrade the accuracy of existing federated news recommendation methods, even when defense is applied. Our study reveals a critical security issue in existing federated news recommendation systems and calls for research efforts to address the issue.

CCS CONCEPTS

• **Information systems → Collaborative filtering**.

KEYWORDS

Untargeted Attack, Federated Learning, News Recommendation

ACM Reference Format:
Jingwei Yi, Fangzhao Wu, Bin Zhu, Guangzhong Sun, Xing Xie. 2018. UA-FedRec: Untargeted Attack on Federated News Recommendation. In *Proceedings of the 28th International ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'21), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD '22, August 14–18, 2022, Washington, DC, USA
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Nowadays, a large amount of news is generated every day, making users overwhelmed. To tackle the information overload problem, personalized news recommendation is proposed, aiming to recommend news according to user interests [2, 27, 30, 40, 42, 45]. Most personalized news recommendation approaches have three components: news model, user model, and click prediction module. The news model learns news representations from news textual information. The user model learns user representations from users' historical clicked news. The click prediction module predicts click scores for each user-and-news-representation pair. However, most news recommendation methods rely on centralized storage, which raises concerns about user privacy. Moreover, some privacy regulations, such as GDPR[1] and CCPA[2], are proposed to protect user privacy. It may not be able to train models with centralized user data in the future.

Federated learning (FL) is a technology that enables multiple clients to collaboratively train a model without sharing their train data [23]. Several federated news recommendation methods are proposed for privacy-preserving news recommendation [31, 32, 50]. Qi et al. [31] propose a privacy-preserving news recommendation method, called *FedRec*, based on federated learning. In FedRec, a central server keeps a global news recommendation model and distributes it to a group of randomly sampled clients in each round. Selected clients train their local models and upload model updates to the server. The server updates the global news recommendation model by aggregating received model updates. Yi et al. [50] propose an efficient federated learning framework, called *Efficient-FedRec*, for privacy-preserving news recommendation. In Efficient-FedRec, the news recommendation model is decomposed into a large news model maintained in the server and a light-weight user model shared among both server and clients, where news representations and the user model are communicated between server and clients. Qi et al. [32] propose a unified news recommendation framework, which contains recall and ranking stages, and can train models and serve users in a privacy-preserving way.

Although these federated news recommendation methods can protect user privacy, the security of federated news recommendation systems is not clear. Since clients need to submit model updates to the central server in federated news recommendation systems, it is possible that an attacker controls multiple malicious clients to submit poisoned updates to attack the global news recommendation model, resulting in degraded performance or preventing convergence of the global news recommendation model. Such attacks are known as untargeted attacks. An untargeted attack on

[1] https://gdpr-info.eu/
[2] https://oag.ca.gov/privacy/ccpa

federated news recommendation can impact a large number of benign clients/users and severely deteriorate the user experience. Therefore, it is necessary to study potential attacks on and effective defenses for federated news recommendation systems.

In this paper, we propose an untargeted attack, called *UA-FedRec*[3], on federated news recommendation systems. By fully exploiting the prior knowledge of news recommendation and federated learning, UA-FedRec can effectively degrade the global model performance with a small percentage of malicious clients. Since the performance of news recommendation models highly depends on the accuracy of user modeling and news modeling [2, 28, 43, 44], we design a news similarity perturbation method to make representations of similar news farther and those of dissimilar news closer and propose a user model perturbation method to make malicious updates neutralize benign updates. Additionally, since updates from different clients are aggregated in vanilla federated learning with weighted-averaging based on their sample sizes, we amplify the impact of malicious updates by proposing a quantity perturbation method that enlarges sample sizes of malicious clients in a reasonable range. The main contributions of this paper can be summarized as follows:

- We present the first study, to the best of our knowledge, on untargeted attacks against federated news recommendation.
- We propose UA-FedRec, an effective untargeted attack on federated news recommendation systems. It requires a small percentage of malicious clients and is thus more practical.
- Extensive experiments on two real-world datasets prove UA-FedRec's effectiveness, even under defenses. Our study reveals a critical security issue in existing federated news recommendation systems, which should draw the attention of researchers in the field.

2 RELATED WORK

2.1 Personalized News Recommendation

Personalized news recommendation is a critical way to personalize news distribution and alleviate the information overload problem. Multiple news recommendation methods have been proposed recently [27, 29, 30, 40, 42, 43, 46]. Generally, there are three core components in news recommendation methods: news model, user model, and click prediction module. The news model is used to learn news representations from news textual information. For example, Wang et al. [41] propose to learn news representations with a knowledge-aware convolutional network (KCNN) and a max-pooling layer. Wu et al. [45] use the combination of multi-head self-attention and additive attention to learn news representations. Wu et al. [46] apply pre-trained language model in the news model to empower its semantic understanding ability. The user model is used to learn user representations from users' historical clicked news representations. For example, Wu et al. [43] apply user embeddings as the query of an additive attention layer to learn user representations. An et al. [2] use a GRU network to capture short-term user interests, and use user embeddings to capture long-term user interests. Qi et al. [29] apply a candidate-aware additive attention network to learn user representations. Click prediction model computes the click score given a pair of user and candidate news representation, which can be implemented by dot product [46], cosine similarity [14], or MLP network [41].

2.2 Federated Recommendation System

Federated learning is a technique that multiple clients collaboratively train a global model without sharing their private data [23]. It performs the following three steps in each round. First, the central server distributes the current global model to a group of randomly sampled clients. Second, each selected client trains the local model with local private data and sends the model update and the number of training samples to the central server. Third, the server aggregates the model updates received from clients to update the global model according to a specific aggregation rule. In FedAvg [23], updates are weighted-averaged based on sample sizes of clients.

Federated learning has been applied to build privacy-preserving recommendation systems [17–20, 26, 37]. Ammad et al. [1] propose federated collaborative filtering (FCF). In FCF, clients use their local private data to compute updates of user embeddings and item embeddings in the CF model. User ID embeddings are directly updated locally. Updates of item embeddings are submitted to the central server, which are further aggregated to update the global item embeddings. Shin et al. [35] propose secure federated matrix factorization (FMF). FMF is similar to FCF, but clients compute local updates according to the matrix factorization algorithm. Qi et al. [31] propose FedRec, a privacy-preserving method for news recommendation model training. In FedRec, clients utilize their local data to compute local updates of the news recommendations and upload the updates to the central server. The central server further aggregates the updates to update the global model.

2.3 Poisoning Attacks

Poisoning attacks interfere with model training via manipulating input samples or model parameters to achieve a certain malicious goal. They can be divided into three categories according to the goal to achieve: targeted attacks, backdoor attacks, and untargeted attacks. Targeted attacks [6] aim to cause misprediction on a specific set of target samples while maintaining the same prediction on the rest of samples. Backdoor attacks [4, 21, 39, 47] aim to cause misprediction only when the backdoor trigger is applied. Untargeted attacks [5, 10] aim to degrade the performance on arbitrary input samples. Poisoning attacks can also be divided into two categories according to the attack method: data poisoning attacks and model poisoning attacks. Data poisoning attacks [7, 9, 12, 22] manipulate input samples, while model poisoning attacks [5, 6, 10] directly manipulate model parameters.

Several data poisoning attack methods on recommendation systems have been proposed [11, 16, 25, 49]. These attacks usually inject fake user item interactions into the training dataset to prompt the exposure rate of the target item. For example, Fang et al. [11] propose to attack graph-based recommendation systems and formulate the attack problem as an optimization problem. Tang et al. [38] formulate the poisoning attack on recommendation as a bi-level optimization problem and solve it with a gradient-based approach. Zhang et al. [52] simulate the recommendation system with an ensemble model and train a deep Q-network [24] to generate adversarial samples. These methods assume that the adversary can

[3] Our code is released at https://github.com/yjw1029/UA-FedRec.

access the full history of the recommendation system, which might not be feasible in practice. To tackle this problem, Zhang et al. [53] design an attack based on incomplete data. All the above attacks are for centralized recommendation systems. Recently, Zhang et al. [54] propose PipAttack, a poisoning attack on federated recommendation systems, which trains a popularity classifier and generates perturbed updates to prompt the target item by raising its popularity. All existing attacks are designed to prompt one or more target items. To the best of our knowledge, untargeted attacks have not been studied yet for the news recommendation scenario.

Recently, several untargeted attacks on federated learning have been proposed [5, 10]. Label flipping [10] is an untargeted data poisoning attack on federated learning by flipping labels of training samples at malicious clients. Some model poisoning attacks on federated learning have been proposed to directly manipulate model updates, which can usually achieve better performance. LIE [5] adds a small mount of noise on each dimension of the average of benign updates, with the noise being small enough to circumvent defense methods. Fang et al. [10] propose to add noise in the opposite direction from the average of benign updates. Besides, they tailor the attack algorithm to evade defenses. However, these untargeted attacks are usually based on a large percentage of malicious clients, which is not practical for federated recommendation systems.

3 METHODOLOGY

In this section, we first introduce the problem formulation and the threat model of federated news recommendation. Then we introduce the basic news recommendation model structure used in our experiments. Finally, we describe the detail of UA-FedRec.

3.1 Problem Formulation

Denote the news set as $\mathcal{N} = \{n_1, n_2, ...n_L\}$, where L is the number of pieces of news. Each piece of news n_i is presented by its title t_i. Denote $\mathcal{U} = \{u_1, u_2, ...u_N\}$ as the total clients participating in federated model training, where N is the number of clients. Given a user u_j, his private click data $\mathcal{B}_j$ is stored in his local device. In federated news recommendation, these N clients collaboratively train a global news recommendation model Θ. In each round, the central server randomly selects k clients. Each selected client trains his local news recommendation model with his local dataset. The difference of the updated model and the global model received from the server is denoted as the model update $\mathbf{g}$. Model updates are uploaded by selected clients and further aggregated by the central server. Among the N clients, we assume there are m malicious clients controlled by an attacker. The malicious clients are denoted as $\mathcal{U}_m = \{u_1, u_2, ...u_m\} \subseteq \mathcal{U}$. The attacker aims to degrade the resulting global model's performance on any input samples by uploading malicious model updates $\mathbf{g}^m$ from selected malicious clients.

3.2 Threat Model

Attacker's Objective. The attacker's objective is to degrade the performance of the federated news recommendation system on arbitrary input samples, i.e., it is an untargeted attack on a federated news recommendation system.

Attacker's Capability. As mentioned in Section 3.1, there are m malicious clients, controlled by an attacker, among N clients participating in model training. Since a recommendation system generally has millions of users in practice, we believe that a reasonable percentage of malicious clients should be up to 1%. The attacker can manipulate model updates of malicious clients to degrade the performance of the global model.

Attacker's Knowledge. We assume that the attacker has full access to the code, local model, and benign datasets on devices of malicious clients. Additionally, we assume the attacker has the information of all pieces of news, such as news titles. Since clients in federated news recommendation do not share their local data, we assume that the attacker has only partial knowledge of the data distribution. Since the server might not release its aggregation code, we assume the attacker does not know the aggregation rule used by the server. Meanwhile, we assume the malicious clients can communicate and collaborate to attack the global recommendation model.

3.3 Basic News Recommendation Model

FedRec [31] is compatible with the majority of news recommendation models. For generality, our UA-FedRec is agnostic of the news recommendation model structure. A news recommendation model is generally composed of three core components: a news model, a user model, and a click prediction model. Given a piece of news n, the news model generates the news representation $\mathbf{n}$ from the news title. We will conduct experiments on two models, NRMS [45] and LSTUR [2]. In NRMS, the news model is implemented with the combination of a multi-head self-attention network and an additive attention layer. In LSTUR, the news model is composed of a convolutional network and an additive attention layer. Given the historical news representations $[\mathbf{n}_1, \mathbf{n}_2...\mathbf{n}_s]$ of a user u, the user encoder learns the user representation $\mathbf{u}$. NRMS applies the combination of a user-level multi-head self-attention network and an additive attention network to learn user representations. LSTUR uses user ID embeddings to capture users' short-term interests and uses a GRU network to capture users' long-term interests. The click prediction model computes click score s for each pair of user and candidate news representation, which is implemented by dot product in both NRMS and LSTUR.

Both NRMS [45] and LSTUR [2] apply negative sampling strategy to compute loss. For each clicked piece of news, P unclicked pieces of news are sampled in the same impression. For the $P + 1$ samples, we denote their click scores as $\{c_1, c_2...c_{P+1}\}$ and their click labels as $\{y_1, y_2...y_{P+1}\}$. The click probability of the i-th piece of news is computed as $p_i = exp(c_i)/\sum_{j=1}^{P+1} exp(c_j))$, and the loss of this sample is computed as $\mathcal{L} = -\sum_{i=1}^{P+1} y_i \times log(p_i)$. For a benign client u_j, the summation of all samples in his local dataset is computed, which is defined as $L_j = \sum_{s_j^i \in \mathcal{B}_j} L_j^i$. Loss L_j is used to compute a update from client u_j, which is denoted as $\mathbf{g}_j$.

3.4 Framework of UA-FedRec

In this subsection, we introduce our UA-FedRec on federated news recommendation. The overall framework is shown in Figure 1. It is composed of three core components: user model perturbation, news

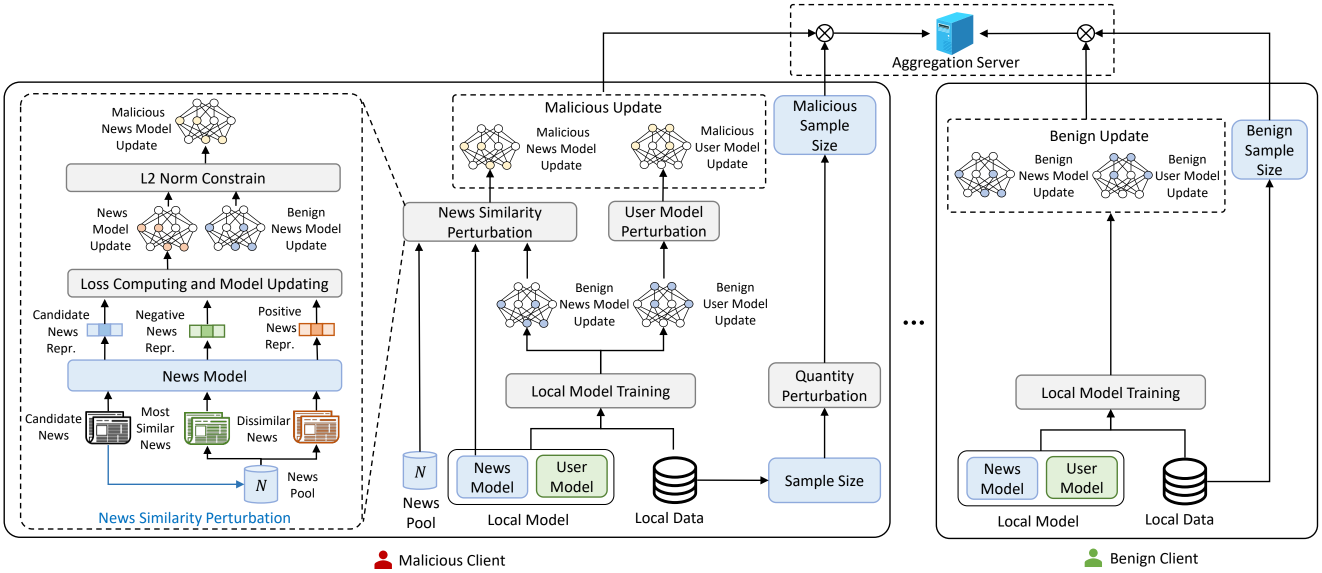


Figure 1: The framework of our UA-FedRec method.

similarity perturbation, and quantity perturbation. Their details are described as follows.

3.4.1 User model perturbation. The user model perturbation is used to generate malicious updates for the user model. In UA-FedRec, we leverage the prior knowledge in news recommendation that the performance of news recommendation highly depends on user modeling and perturb updates of the user model in opposite directions of benign updates. First, we estimate benign updates from benign datasets in the devices of malicious clients. Specifically, for each malicious client $u_i \in \mathcal{U}_m$, we compute a benign update following the steps described in Section 3.3. The benign user model update of client u_i is denoted as $\mathbf{g}_i^u$. Then we average the benign user model updates of all malicious clients to estimate a benign user model update: $\boldsymbol{\mu}_u = \sum_{1 \le i \le m} \mathbf{g}_i^u / m$. Second, we compute the direction of the estimated benign user model update, $\mathbf{s}_u = sign(\boldsymbol{\mu}_u)$. We also compute the standard deviation of the benign user model updates of all malicious clients, which is denoted as $\boldsymbol{\sigma}_u$. To circumvent defenses, a malicious user update should not be too far way from a benign update. To meet this requirement, the malicious update from a malicious client is computed as $\mathbf{g}_u^m = \boldsymbol{\mu}_u - \lambda_1 \mathbf{s}_u \odot \boldsymbol{\sigma}_u$, where $\lambda_1 \le [3, 4]$ and $\odot$ stands for the element-wise product operation.

3.4.2 News similarity perturbation. News similarity perturbation is used to generate malicious updates for the news model. It is motivated from the prior knowledge in news recommendation that news modeling is critical for news recommendation. For example, a user who read "Best PS5 games: top PlayStation 5 titles to look forward to" likely also read "Marvel's Avengers game release date, news, trailers and first impressions", but is less likely to click "The Cost of Trump's Aid Freeze in the Trenches of Ukraine's War". For a good news recommendation model, the second news's representation should be close to the first news's representation in the vector space, while the third news's representation should be far away from the first news's representation in the vector space. Therefore, we design our news similarity perturbation to make representations of similar news farther and those of dissimilar news closer.

First, we inference news representations and search the closest and farthest pieces of news for each piece of news. For the i-th piece of news n_i, its closest and farthest pieces of news, denoted as n_i^n and n_i^f, respectively, can be computed as follows:

$$\begin{aligned} n_i^f &= \min_{n_j \in \mathcal{N}, j \neq i} \mathbf{n}_i^T \mathbf{n}_j, \\ n_i^n &= \max_{n_j \in \mathcal{N}, j \neq i} \mathbf{n}_i^T \mathbf{n}_j, \end{aligned} \tag{1}$$

where $\mathbf{n}_i$ and $\mathbf{n}_j$ are news representations for the i-th and the j-th pieces of news, respectively. Computing all news representations and selecting neighbors in each round are time-consuming. To reduce complexity, we assume that distances between news representations do not change significantly in K rounds, and thus update the selected news neighbors once every K rounds. Moreover, we apply the approximate nearest neighbor (ANN) [3, 15] to search the nearest and farthest news more efficiently.

Second, we enlarge the MSE loss between $\mathbf{n}_i$ and $\mathbf{n}_i^n$ and reduce the MSE loss between $\mathbf{n}_i$ and $\mathbf{n}_i^f$. The news similarity perturbation loss is computed as follows:

$$\mathcal{L}_n = \sum_{n_i \in \mathcal{N}} (\mathbf{n}_i - \mathbf{n}_i^f)^T (\mathbf{n}_i - \mathbf{n}_i^f) - (\mathbf{n}_i - \mathbf{n}_i^n)^T (\mathbf{n}_i - \mathbf{n}_i^n). \tag{2}$$

The local model is updated using the loss in Eq. 2 with the backpropagation algorithm to get news model update $\mathbf{g}_n$. To evade detection, we constrain the L_2 norm of a malicious news model update not too far away from the L_2 norm of benign news model updates. We estimate benign updates in the following way. For each malicious client $u_i \in \mathcal{U}_m$, we compute its benign news model update $\mathbf{g}_i^n$ using its local benign dataset according to the steps described in Section 3.3.

We then compute the average and the standard deviation of the L_2 norm of benign updates from all malicious clients, denoted as μ_n and σ_n, respectively. Assuming the L_2 norm of benign updates follow the Gaussian distribution, we set a reasonable maximum L_2 norm of malicious news model updates as $\mu_n + \lambda_2\sigma_n$. The final malicious news model update is thus computed as:

$$\mathbf{g}_n^m = \frac{\mathbf{g}_n}{max(1, ||\mathbf{g}_n||_2/(\mu_n + \lambda_2\sigma_n))}. \tag{3}$$

3.4.3 Quantity perturbation. In most federated learning methods, updates from different clients are aggregated with weighted-averaging based on their sample sizes. To exploit this prior knowledge, we enlarge sample sizes of malicious clients in sending to the server to magnify the impact of malicious updates. Generated malicious sample sizes should be sufficiently large to enhance the influence of malicious updates, but should also be small enough to evade detection. Unlike some other federated learning scenarios, sample sizes vary across clients in the recommendation scenario [34, 48]. We leverage this characteristic to enlarge sample sizes of malicious clients in the following way. Denote benign sample sizes of malicious clients as $\{s_1, s_2, ...s_m\}$. We compute their average and standard deviation, denoted as μ_s and σ_s, respectively. The final sample size submitted to the central server by a malicious client is $\mu_s + \lambda_3\sigma_s$, where $0 \le \lambda_3 \le 4$.

4 EXPERIMENTAL EVALUATION

In this section, we conduct several experiments on two datasets to answer the following research questions:

- **RQ1:** How does our UA-FedRec perform comparing with baseline attack methods?
- **RQ2:** Can our UA-FedRec circumvent defense methods while preserving its attack performance?
- **RQ3:** Are the proposed news similarity perturbing, user model perturbing, and quantity perturbing all effective?
- **RQ4:** How does the percentage of malicious clients influence the attack performance?

4.1 Dataset and Experimental Settings

We conduct experiments on two real-world datasets, i.e. MIND[4] and Feeds. MIND is a public dataset collected from anonymized behavior logs of Microsoft News website, which contains user behaviors in six weeks. We collect the Feeds dataset from Microsoft news App from August 1st, 2020 to September 1st, 2020. For MIND, we directly use the provided training, validation, and test datasets. For Feeds, we use the impressions in the first three weeks as the training dataset, the impressions in the later two days as the validation dataset, and the rest in the last week for testing. The detailed dataset statistics are summarized in Table 1. Following previous news recommendation works [2, 30, 45], we use AUC, MRR, nDCG@5 and nDCG@10 as the evaluation metrics. We note that the experimental results reported here are all on benign datasets. Even though news recommendation is a personalized system, our results reflect the impact of our attack on benign clients by using a small percentage of malicious clients.

We evaluate our UA-FedRec against two news recommendation models: NRMS [45] and LSTUR [2]. We apply the non-uniform weighted averaging FedAdam [33] to train the news recommendation models. We use the ANN algorithms implemented by Johnson et al. [13]. We set λ_1, λ_2 and λ_3 to 1.5, 1.5, 3, respectively, on Feeds with LSTUR. In other experiments, λ_1, λ_2 and λ_3 are set to 3.0. The dimension of news representations is 400. To mitigate overfitting, we apply dropout with dropout rate 0.2 in news recommendation models. The learning rate is 0.0001. The number of negative samples associated with each positive sample is 4. The number of users randomly sampled per round is 50 for both MIND and Feeds. The percentage of malicious clients is set to 1% unless stated otherwise. All hyper-parameters are selected according to results on the validation set. We repeat each experiment 5 times independently, and report the average results with standard deviations.

Table 1: Detailed statistics of MIND and Feeds.

	MIND	Feeds
#news	65,238	643,177
#users	94,057	10,000
#impressions	230,117	320,578
#positive samples	347,727	437,072
#negative samples	8,236,715	6,621,187

4.2 Performance Comparison (RQ1)

We select existing untargeted attacks as baseline methods and compare our UA-FedRec with them. The baseline methods include the following data poisoning attack methods:

- **No Attack**, where no attack is applied. It is the upper bound of model performance;
- **Label Flipping (LF)** [10], an attack that flips click labels of training input samples;
- **Popularity Perturbation (Pop)** [54], an untargeted version of the explicit boosting in PipAttack, where malicious clients click only cold news without clicking popular news;

and the following model poisoning attack methods:

- **Gaussian** [10], where the attacker first estimates the Gaussian distribution of benign model updates using benign data on devices of malicious clients, and then samples updates from the distribution for malicious clients;
- **Little is Enough (LIE)** [5], adding a small amount of noise to each dimension of the average of the benign updates. The noise is large enough to adversely impact the global model yet sufficiently small to evade detection of the defense methods;
- **Fang** [10], where noise is added in the opposite direction from the average of benign model updates. The attacker solves an optimization problem to get sub-optimal noise scale that is large enough yet is able to circumvent the target defense.

The experimental results are shown in Table 2. We have the following observations from the table. First, our UA-FedRec outperforms data poisoning attack methods (LF and Pop). This is because UA-FedRec directly manipulates model updates, while data poisoning attacks perturb only input samples. Second, our UA-FedRec outperforms other model poisoning attack methods (Gaussian, LIE,

[4] https://msnews.github.io/. We use the small version of MIND for fast experiments.

Table 2: Attack performance of different attack methods with no defense.

Base Model	Methods	MIND				Feeds			
		AUC	MRR	nDCG@5	nDCG@10	AUC	MRR	nDCG@5	nDCG@10
NRMS	No Attack	66.73±0.13	32.34±0.15	35.05±0.14	40.75±0.12	65.05±0.09	31.92±0.10	34.39±0.12	42.15±0.10
	LF [10]	66.69±0.15	32.26±0.10	34.97±0.10	40.69±0.09	64.90±0.11	31.78±0.10	34.20±0.13	42.00±0.13
	Pop [54]	66.72±0.23	32.34±0.12	35.05±0.12	40.74±0.12	64.99±0.18	31.87±0.13	34.33±0.16	32.11±0.16
	Gaussian [10]	66.64±0.17	32.33±0.13	35.02±0.15	40.71±0.12	64.87±0.17	31.82±0.11	34.27±0.15	42.04±0.12
	LIE [5]	59.52±0.43	27.69±0.26	29.43±0.27	35.03±0.27	61.63±0.25	29.19±0.15	30.85±0.19	38.85±0.21
	Fang [10]	62.92±0.71	29.64±0.48	31.83±0.57	37.52±0.58	61.04±0.26	28.74±0.16	30.33±0.19	38.31±0.19
	UA-FedRec	**55.81**±0.34	**25.08**±0.37	**26.19**±0.37	**31.79**±0.35	**58.96**±0.61	**27.13**±0.52	**28.30**±0.63	**36.39**±0.58
LSTUR	No Attack	66.67±0.09	32.30±0.12	34.97±0.11	40.67±0.11	65.17±0.04	31.91±0.08	34.39±0.13	42.19±0.08
	LF [10]	66.63±0.09	32.24±0.08	34.87±0.10	40.58±0.10	65.12±0.13	31.80±0.14	34.27±0.17	42.09±0.15
	Pop [54]	66.81±0.14	32.40±0.11	35.07±0.14	40.76±0.13	65.30±0.05	32.01±0.04	34.50±0.05	42.32±0.04
	Gaussian [10]	66.69±0.14	32.26±0.11	34.90±0.13	40.62±0.13	65.15±0.03	31.91±0.02	34.42±0.03	42.18±0.02
	LIE [5]	63.56±0.20	29.99±0.25	32.20±0.24	37.91±0.23	63.93±0.57	30.78±0.41	32.99±0.54	40.91±0.50
	Fang [10]	63.87±0.35	30.33±0.26	32.57±0.33	38.25±0.30	61.81±0.67	29.17±0.47	30.93±0.57	38.92±0.58
	UA-FedRec	**54.33**±0.69	**24.37**±0.70	**25.30**±0.66	**30.96**±0.56	**59.36**±0.39	**27.25**±0.31	**28.52**±0.36	**36.64**±0.35

Table 3: Performance of different defense methods without any attack.

Base Model	Methods	MIND				Feeds			
		AUC	MRR	nDCG@5	nDCG@10	AUC	MRR	nDCG@5	nDCG@10
NRMS	No Defense	66.72±0.13	32.34±0.15	35.05±0.14	40.75±0.12	65.05±0.09	31.92±0.10	34.39±0.12	42.15±0.10
	Median [51]	56.05±0.24	25.45±0.07	26.50±0.07	32.06±0.10	60.56±0.12	28.32±0.18	29.82±0.22	37.85±0.17
	Trimmed-Mean [51]	63.64±0.25	30.00±0.22	32.12±0.24	37.85±0.23	61.31±0.25	28.84±0.18	30.46±0.23	38.52±0.21
	Krum [8]	56.97±0.03	25.84±0.18	27.15±0.19	32.82±0.12	62.15±0.29	29.49±0.32	31.37±0.37	39.35±0.34
	Multi-Krum [8]	65.80±0.17	31.66±0.10	34.23±0.11	39.93±0.12	62.51±0.08	29.73±0.06	31.62±0.07	39.63±0.08
	Norm-Bounding [36]	66.92±0.19	32.44±0.13	35.18±0.14	40.88±0.14	64.97±0.05	31.84±0.09	34.31±0.10	42.08±0.09
LSTUR	No Defense	66.67±0.09	32.30±0.12	34.97±0.11	40.67±0.11	65.17±0.04	31.91±0.08	34.39±0.13	42.19±0.08
	Median [51]	56.26±0.18	25.65±0.19	26.77±0.19	32.35±0.16	60.22±0.12	27.93±0.13	29.35±0.13	37.45±0.12
	Trimmed-Mean [51]	63.19±0.10	29.58±0.07	31.66±0.07	37.41±0.07	61.48±0.29	29.02±0.06	30.69±0.06	38.68±0.12
	Krum [8]	56.62±0.41	25.69±0.48	26.97±0.59	32.55±0.54	62.71±0.16	29.95±0.20	31.99±0.19	39.97±0.13
	Multi-Krum [8]	65.94±0.19	31.68±0.15	34.19±0.15	39.92±0.14	62.86±0.11	29.97±0.08	31.90±0.09	39.96±0.09
	Norm-Bounding [36]	66.75±0.16	32.30±0.18	34.96±0.20	40.66±0.18	65.22±0.14	31.98±0.09	34.49±0.09	42.27±0.10

and Fang). This is because UA-FedRec has fully exploited the prior knowledge in news recommendation and federated learning: it applies both user model perturbation and news similarity perturbation since user modeling and news modeling are critical for news recommendation. The user model perturbation makes updates of user model less accurate. The news similarity perturbation makes similar news farther and dissimilar news closer, which can effectively interfere with news modeling. Moreover, UA-FedRec applies quantity perturbation to amplify the impact of malicious updates. Third, the well-designed model poisoning attacks (LIE, Fang and UA-FedRec) perform better than the data poisoning attacks (LF and Pop). This is because perturbing model updates is more effective than manipulating input samples. A model poisoning attack is generally more flexible and performs better than a data poisoning attack. Finally, our UA-FedRec significantly reduces the performance of news recommendation models with only 1% of malicious clients, making the attack more practical for the federated news recommendation.

4.3 Circumventing Defenses (RQ2)

To evaluate the effectiveness of existing defenses against our UA-FedRec, we consider several defenses, including:

- **Median** [51], a coordinate-wise aggregation algorithm that aggregates updates by computing the median of each dimension of the updates.
- **Trimmed-Mean** [51], another coordinate-wise aggregation algorithm that aggregates updates by computing the trimmed-mean of each dimension of the updates.
- **Krum** [8], selecting the update from the set of received updates that is closest to its subset of neighboring updates.
- **Multi-Krum** [8], a variant of Krum that selects multiple updates from the set of received updates instead of one, and averages the selected updates.
- **Norm-Bounding** [36], bounding the L_2 norm of received updates with a fixed threshold and computing the weighted average of all the updates.

A defense method should not incur any significant adverse impact on the performance of a model. To evaluate the impact of these defenses on the performance of federated news recommendation systems, we first evaluate them with NRMS and LSTUR news recommendation models on both datasets. The experimental results are shown in Table 3. The table shows that some defenses (Krum, Median, Trimmed-Mean) severely degrade the performance

Table 4: Attack performance of different methods with Norm-Bounding defense.

Base Model	Methods	MIND				Feeds			
		AUC	MRR	nDCG@5	nDCG@10	AUC	MRR	nDCG@5	nDCG@10
NRMS	No Attack	66.92±0.19	32.44±0.13	35.18±0.14	40.88±0.14	64.97±0.05	31.84±0.09	34.31±0.10	42.08±0.09
	LF [10]	66.70±0.11	32.27±0.12	34.99±0.14	40.69±0.12	65.02±0.10	31.87±0.13	34.34±0.15	42.11±0.13
	Pop [54]	66.68±0.25	32.30±0.16	35.00±0.19	40.68±0.18	65.15±0.10	31.97±0.12	34.43±0.12	42.22±0.11
	Gaussian [10]	66.66±0.07	32.28±0.08	34.98±0.10	40.69±0.09	65.02±0.06	31.90±0.04	34.34±0.08	42.13±0.04
	LIE [5]	63.46±0.26	30.07±0.19	32.32±0.23	38.03±0.22	61.21±0.34	28.83±0.25	30.47±0.30	38.46±0.31
	Fang [10]	66.25±0.19	32.03±0.18	34.65±0.22	40.35±0.21	63.08±0.70	30.26±0.59	32.24±0.76	40.16±0.69
	UA-FedRec	**57.00**±0.26	**25.61**±0.21	**26.91**±0.24	**32.57**±0.25	**59.73**±0.26	**27.62**±0.24	**28.98**±0.30	**37.08**±0.27
LSTUR	No Attack	66.75±0.16	32.30±0.18	34.96±0.20	40.66±0.18	65.22±0.14	31.98±0.09	34.49±0.09	42.27±0.10
	LF [10]	66.62±0.14	32.24±0.11	34.90±0.11	40.61±0.10	65.02±0.07	31.85±0.07	34.29±0.08	42.11±0.05
	Pop [54]	66.75±0.09	35.04±0.09	35.05±0.12	40.74±0.10	65.25±0.06	32.02±0.02	34.52±0.03	42.31±0.04
	Gaussian [10]	66.69±0.25	32.29±0.18	34.94±0.21	40.64±0.20	65.18±0.05	31.93±0.05	34.39±0.05	42.19±0.06
	LIE [5]	64.97±0.12	31.11±0.04	33.60±0.03	39.30±0.05	64.95±0.28	31.78±0.25	34.23±0.26	42.06±0.21
	Fang [10]	66.36±0.12	32.09±0.10	34.70±0.09	40.40±0.09	64.83±0.13	31.63±0.15	34.04±0.17	41.86±0.15
	UA-FedRec	**55.24**±0.85	**24.89**±0.51	**25.99**±0.54	**31.56**±0.56	**61.83**±0.87	**29.10**±0.78	**30.90**±0.95	**38.92**±0.90

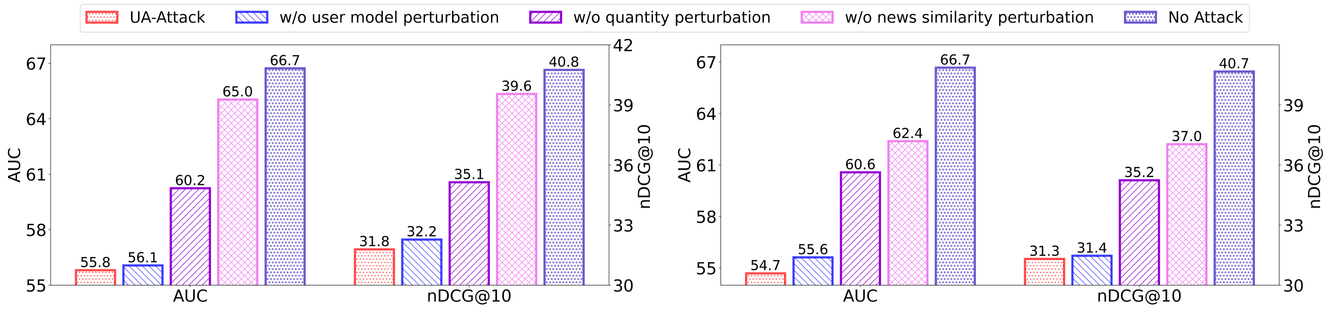
Table 5: Attack performance of different methods with Multi-Krum defense.

Base Model	Methods	MIND				Feeds			
		AUC	MRR	nDCG@5	nDCG@10	AUC	MRR	nDCG@5	nDCG@10
NRMS	No Attack	65.80±0.17	31.66±0.10	34.23±0.11	39.93±0.12	62.51±0.08	29.73±0.06	31.62±0.07	39.63±0.08
	LF [10]	65.63±0.28	31.54±0.21	34.06±0.24	39.77±0.22	62.44±0.07	29.74±0.07	31.64±0.07	39.64±0.07
	Pop [54]	65.73±0.19	31.62±0.16	34.14±0.23	39.83±0.21	62.28±0.11	29.57±0.13	31.39±0.17	39.44±0.14
	Gaussian [10]	65.75±0.18	31.66±0.19	34.19±0.17	39.90±0.16	62.28±0.16	29.59±0.14	31.42±0.19	39.46±0.13
	LIE [5]	62.93±0.23	29.64±0.09	31.77±0.10	37.49±0.10	61.77±0.06	29.20±0.06	30.89±0.07	38.96±0.06
	Fang [10]	65.45±0.16	31.37±0.12	33.83±0.13	39.53±0.15	61.84±0.28	29.26±0.28	30.99±0.36	39.02±0.29
	UA-FedRec	**60.30**±0.80	**27.97**±0.47	**29.78**±0.51	**35.39**±0.52	**61.02**±0.10	**28.65**±0.11	**30.20**±0.15	**38.26**±0.13
LSTUR	No Attack	65.94±0.19	31.68±0.15	34.19±0.15	39.92±0.14	62.86±0.11	29.97±0.08	31.90±0.09	39.96±0.09
	LF [10]	66.06±0.08	31.82±0.07	34.33±0.08	40.06±0.08	62.54±0.07	29.74±0.07	31.62±0.10	39.69±0.08
	Pop [54]	65.97±0.18	31.79±0.15	34.30±0.17	40.02±0.16	62.40±0.07	29.55±0.23	31.37±0.29	39.49±0.30
	Gaussian [10]	65.99±0.18	31.76±0.13	34.26±0.15	39.99±0.13	62.81±0.09	29.91±0.01	31.84±0.04	39.90±0.02
	LIE [5]	65.92±0.18	31.23±0.17	33.60±0.17	39.45±0.12	62.51±0.16	29.78±0.17	31.70±0.17	39.71±0.21
	Fang [10]	65.69±0.26	31.58±0.16	34.06±0.17	39.78±0.18	62.11±0.01	29.40±0.02	31.19±0.01	39.27±0.04
	UA-FedRec	**59.87**±0.62	**27.24**±0.31	**28.89**±0.32	**34.63**±0.32	**61.70**±0.31	**29.19**±0.03	**30.92**±0.09	**38.93**±0.17

of both news recommendation models. As a result, we select only the defenses, i.e., Norm-Bounding and Multi-Krum, that have small performance degradation to evaluate our UA-FedRec and the baseline methods.

The experimental results of attacking federated new recommendation systems are shown in Table 4 when the Norm-Bounding defense is applied and in Table 5 when the Multi-Krum defense is applied. From both Table 4 and Table 5, we have several observations. First, data poisoning attacks (LF and Pop) are ineffective when Norm-Bounding or Multi-Krum is applied. These attacks perform poorly without any defense, as Table 2 shows, since they require more than 1% malicious clients, let alone with defense. Second, our UA-FedRec outperforms model poisoning attacks (LIE and Fang) with both Norm-Bounding and Multi-Krum defenses. Our news similarity perturbation and user model perturbation can still effectively impact news recommendation models even when these defenses are applied. Third, well-designed model poisoning attacks (LIE, Fang, and UA-FedRec) perform better than data poisoning attacks (Lf and Pop). This is because these model poisoning attack methods optimize the perturbation degree directly on model updates while adding constraints to circumvent defenses, resulting in a better capability to evade defenses. Forth, comparing with the performance without any defense, both Norm-Bounding and Multi-Krum improve the performance when facing the tested attacks, except for Multi-Krum on Feeds. This is because the defenses can contain the impact of malicious updates or directly detect malicious updates and filter them out.

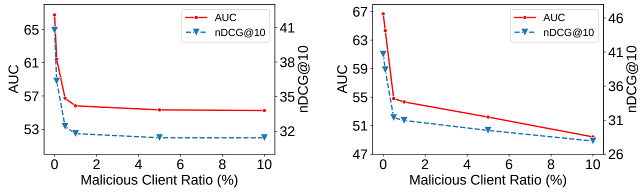
Our experimental results indicate that existing robust aggregation rules either significantly degrade the performance of news recommendation models or cannot effectively thwart UA-FedRec. As a future work, we plan to study effective defense methods on federated news recommendation systems to defend against UA-FedRec. Specifically, first, we plan to detect malicious news similarity updates to defend against the news similarity perturbation. Since the news information is public for both server and clients, the server can estimate news similarity scores with self-supervised or unsupervised training methods. Second, we plan to take sample sizes into robust aggregation rules to restrict the impact of updates with larger sample sizes to defend against quantity perturbation. Third,



(a) Performance of NRMS on MIND

(b) Performance of LSTUR on MIND

Figure 2: Effectiveness of each core component in UA-FedRec.



(a) on NRMS.

(b) on LSTUR.

Figure 3: Impact of malicious client ratio on MIND.

we plan to detect malicious user modeling updates to defend against the user perturbation.

4.4 Ablation Study (RQ3)

In this subsection, we study the impact of the three core components of our UA-FedRec, i.e., user model perturbation, news similarity perturbation, and quantity perturbation. The experimental results on MIND are shown in Figure 2 while the results on Feeds are in Appendix. We can make the following observations. First, the attack performance of our UA-FedRec degrades without the news similarity perturbation. This is because news similarity modeling is critical to news recommendation and our news similarity perturbation can effectively interfere with model's learning news similarity. Second, the attack performance of our UA-FedRec degrades without the quantity perturbation. This is because model updates are aggregated based on sample sizes in FedAvg. Our quantity perturbation amplifies the impact of malicious updates. Third, the attack performance of our UA-FedRec degrades a little without the user perturbation. Our user perturbation manipulates a user model update in the opposite direction of the average of benign updates. Since news representations are polluted by the news similarity perturbation, the user model is unable to capture user interests even without the user model perturbation, resulting in a small drop of performance without the user perturbation.

4.5 Impact of Malicious Client Ratio (RQ4)

In this subsection, we study the impact of the percentage of malicious clients. We conduct experiments with 0.1%, 0.5%, 1%, 5% and 10% of malicious clients. The experimental results on MIND dataset are shown in Figure 3 and those on Feeds are shown in Appendix. We can see that the attack performance improves with a larger percentage of malicious clients. This is expected since more malicious updates are uploaded with a higher percentage of malicious clients, resulting in a more heavily affected global news recommendation model. Second, our UA-FedRec can effectively attack the global news recommendation model with a percentage of malicious clients as low as 0.1%. By exploiting the prior knowledge in news recommendation and federated learning, UA-FedRec effectively perturbs news similarity modeling and user modeling and amplifies the impact of malicious updates with the quantity perturbation. These perturbations can effectively reduce the percentage of malicious clients in launching an effective untargeted attack.

5 CONCLUSION

In this paper, we propose an untargeted attack, called UA-FedRec, on federated news recommendation systems. By exploiting the prior knowledge in news recommendation and federated learning, we have designed three perturbation methods in UA-FedRec, i.e., news similarity perturbation, user model perturbation and quantity perturbation, to interfere with news similarity modeling, user modeling, and amplify the impact of malicious updates. The user model perturbation makes news representations of similar news farther and those of dissimilar news closer, which can effectively interfere with news similarity modeling in news recommendation. The user model perturbation perturbs user model updates in opposite directions of benign updates to interfere with user modeling. The quantity perturbation enlarges sample sizes of malicious clients in a reasonable range to amplify the impact of malicious updates. Extensive experiments on two real-world datasets indicate that our UA-FedRec can effectively degrade the performance of federated news recommendation systems while circumventing defenses with a percentage of malicious clients as low as 1%. It outperforms existing untargeted attacks using data poisoning or model poisoning. Our study reveals a critical security issue in existing federated news recommendation systems and calls for more research efforts to address this issue. In the future, we plan to study effective defense methods to thwart UA-FedRec and other potential attacks against news recommendation systems. In addition, we also plan to extend our UA-Attack to other content-based recommendation scenarios.

REFERENCES

[1] Muhammad Ammad, E. Ivannikova, S. Khan, Were Oyomno, Qiang Fu, Kuan Eeik Tan, and A. Flanagan. 2019. Federated Collaborative Filtering for Privacy-Preserving Personalized Recommendation System. *ArXiv* abs/1901.09888 (2019).

[2] Mingxiao An, Fangzhao Wu, Chuhan Wu, Kun Zhang, Zheng Liu, and Xing Xie. 2019. Neural News Recommendation with Long- and Short-term User Representations. In *ACL*. 336–345.

[3] Sunil Arya, David M Mount, Nathan S Netanyahu, Ruth Silverman, and Angela Y Wu. 1998. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM (JACM)* 45, 6 (1998), 891–923.

[4] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. 2020. How to backdoor federated learning. In *AISTATS*. 2938–2948.

[5] Gilad Baruch, Moran Baruch, and Yoav Goldberg. 2019. A Little Is Enough: Circumventing Defenses For Distributed Learning. In *NIPS*, Vol. 32.

[6] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. 2019. Analyzing federated learning through an adversarial lens. In *ICML*. 634–643.

[7] Battista Biggio, Blaine Nelson, and Pavel Laskov. 2011. Support vector machines under adversarial label noise. In *Asian conference on machine learning*. 97–112.

[8] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. 2017. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In *Advances in Neural Information Processing Systems*, Vol. 30.

[9] Di Cao, Shan Chang, Zhijian Lin, Guohua Liu, and Donghong Sun. 2019. Understanding Distributed Poisoning Attack in Federated Learning. In *ICPADS*. 233–239.

[10] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 2020. Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. In *USENIX*.

[11] Minghong Fang, Guolei Yang, Neil Zhenqiang Gong, and Jia Liu. 2018. Poisoning Attacks to Graph-Based Recommender Systems. In *ACSAC*. 381–392.

[12] Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. 2020. The Limitations of Federated Learning in Sybil Settings. In *RAID*. 301–316.

[13] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734* (2017).

[14] Vaibhav Kumar, Dhruv Khattar, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep Neural Architecture for News Recommendation. In *CLEF (Working Notes)*.

[15] Eyal Kushilevitz, Rafail Ostrovsky, and Yuval Rabani. 2000. Efficient search for approximate nearest neighbor in high dimensional spaces. *SIAM J. Comput.* 30, 2 (2000), 457–474.

[16] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. 2016. Data Poisoning Attacks on Factorization-Based Collaborative Filtering. In *NIPS*. 1893–1901.

[17] Tan Li, Linqi Song, and Christina Fragouli. 2020. Federated Recommendation System via Differential Privacy. In *ISIT*. 2592–2597.

[18] Feng Liang, Weike Pan, and Zhong Ming. 2021. FedRec++: Lossless Federated Recommendation with Explicit Feedback. *AAAI* 35 (2021), 4224–4231.

[19] Guanyu Lin, Feng Liang, Weike Pan, and Zhong Ming. 2021. FedRec: Federated Recommendation With Explicit Feedback. *IEEE Intelligent Systems* 36, 5 (2021), 21–30.

[20] Yujie Lin, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Dongxiao Yu, Jun Ma, Maarten de Rijke, and Xiuzhen Cheng. 2020. Meta Matrix Factorization for Federated Rating Predictions. In *SIGIR*. 981–990.

[21] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2017. Trojaning attack on neural networks. In *NDSS*.

[22] Saeed Mahloujifar, Mohammad Mahmoody, and Ameer Mohammed. 2019. Universal Multi-Party Poisoning Attacks. In *ICML*, Vol. 97. 4274–4283.

[23] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*. 1273–1282.

[24] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[25] Bamshad Mobasher, Robin Burke, Runa Bhaumik, and Chad Williams. 2007. Toward Trustworthy Recommender Systems: An Analysis of Attack Models and Algorithm Robustness. *TOIT* 7, 4 (oct 2007), 23–es.

[26] Khalil Muhammad, Qinqin Wang, Diarmuid O'Reilly-Morgan, Elias Tragos, Barry Smyth, Neil Hurley, James Geraci, and Aonghus Lawlor. 2020. FedFast: Going Beyond Average for Faster Training of Federated Recommender Systems. In *KDD*. 1234–1242.

[27] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based news recommendation for millions of users. In *KDD*. 1933–1942.

[28] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-Based News Recommendation for Millions of Users. In *KDD*. 1933–1942.

[29] Tao Qi, Fangzhao Wu, Chuhan Wu, and Yongfeng Huang. 2021. Personalized news recommendation with knowledge-aware interactive matching. In *SIGIR*. 61–70.

[30] Tao Qi, Fangzhao Wu, Chuhan Wu, and Yongfeng Huang. 2021. PP-Rec: News Recommendation with Personalized User Interest and Time-aware News Popularity. In *ACL*. Online, 5457–5467.

[31] Tao Qi, Fangzhao Wu, Chuhan Wu, Yongfeng Huang, and Xing Xie. 2020. Privacy-Preserving News Recommendation Model Learning. In *EMNLP Findings*. 1423–1432.

[32] Tao Qi, Fangzhao Wu, Chuhan Wu, Yongfeng Huang, and Xing Xie. 2021. Uni-FedRec: A Unified Privacy-Preserving News Recommendation Framework for Model Training and Online Serving. In *Findings of EMNLP*. 1438–1448.

[33] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. 2021. Adaptive Federated Optimization. In *ICLR*.

[34] Shaoyun Shi, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Attention-Based Adaptive Model to Unify Warm and Cold Starts Recommendation. In *CIKM*. 127–136.

[35] Hyejin Shin, Sungwook Kim, Junbum Shin, and Xiaokui Xiao. 2018. Privacy enhanced matrix factorization for recommendation with local differential privacy. *TKDE* 30, 9 (2018), 1770–1782.

[36] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. 2019. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963* (2019).

[37] Ben Tan, Bo Liu, Vincent Zheng, and Qiang Yang. 2020. A Federated Recommender System for Online Services. In *RecSys*. 579–581.

[38] Jiaxi Tang, Hongyi Wen, and Ke Wang. 2020. Revisiting Adversarially Learned Injection Attacks Against Recommender Systems. In *RecSys*. 318–327.

[39] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy yong Sohn, Kangwook Lee, and Dimitris S. Papailiopoulos. 2020. Attack of the Tails: Yes, You Really Can Backdoor Federated Learning. In *NIPS*.

[40] Heyuan Wang, Fangzhao Wu, Zheng Liu, and Xing Xie. 2020. Fine-grained Interest Matching for Neural News Recommendation. In *ACL*. 836–845.

[41] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep Knowledge-Aware Network for News Recommendation. In *WWW*. 1835–1844.

[42] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Neural News Recommendation with Attentive Multi-View Learning. In *IJCAI*. 3863–3869.

[43] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. NPA: Neural News Recommendation with Personalized Attention. In *KDD*. 2576–2584.

[44] Chuhan Wu, Fangzhao Wu, Mingxiao An, Yongfeng Huang, and Xing Xie. 2019. Neural news recommendation with topic-aware news representation. In *ACL*. 1154–1159.

[45] Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019. Neural News Recommendation with Multi-Head Self-Attention. In *EMNLP*. 6389–6394.

[46] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Empowering News Recommendation with Pre-Trained Language Models. In *SIGIR*. 1652–1656.

[47] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. 2020. DBA: Distributed Backdoor Attacks against Federated Learning. In *ICLR*.

[48] Jingwei Xu, Yuan Yao, Hanghang Tong, XianPing Tao, and Jian Lu. 2015. Ice-Breaking: Mitigating Cold-Start Recommendation Problem by Rating Comparison. In *IJCAI*. 3981–3987.

[49] Guolei Yang, Neil Zhenqiang Gong, and Ying Cai. 2017. Fake Co-visitation Injection Attacks to Recommender Systems.. In *NDSS*.

[50] Jingwei Yi, Fangzhao Wu, Chuhan Wu, Ruixuan Liu, Guangzhong Sun, and Xing Xie. 2021. Efficient-FedRec: Efficient Federated Learning Framework for Privacy-Preserving News Recommendation. In *EMNLP*. 2814–2824.

[51] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. 2018. Byzantine-robust distributed learning: Towards optimal statistical rates. In *ICML*. 5650–5659.

[52] Hengtong Zhang, Yaliang Li, Bolin Ding, and Jing Gao. 2020. Practical Data Poisoning Attack against Next-Item Recommendation. In *WWW*. 2458–2464.

[53] Hengtong Zhang, Changxin Tian, Yaliang Li, Lu Su, Nan Yang, Wayne Xin Zhao, and Jing Gao. 2021. Data Poisoning Attack against Recommender System Using Incomplete and Perturbed Data. In *KDD*. 2154–2164.

[54] Shijie Zhang, Hongzhi Yin, Tong Chen, Zi Huang, Quoc Viet Hung Nguyen, and Lizhen Cui. 2021. PipAttack: Poisoning Federated Recommender Systems forManipulating Item Promotion. *arXiv preprint arXiv:2110.10926* (2021).

APPENDIX

Experimental Environment

There are 8 Tesla V100-SXM2-32GB in the server with CUDA 11.1. The CPU is Intel(R) Xeon(R) Platinum 8168 CPU @ 2.70GHz. We use python 3.7.11, pytorch 1.10.0. Each experiment is run on a single GPU and a single CPU core.

Ablation Study on Feeds

In this subsection, we study the impact of the three core components in UA-FedRec on Feeds, i.e., the news similarity perturbation, the user model perturbation, and the quantity perturbation. The experimental results of NRMS on Feeds are shown in Figure 4, and the experimental results of LSTUR on Feeds are shown in Figure 5. The observations we can make from Figure 4 and Figure 5 are similar to those on the results presented in Section 4.4. First, whichever component is removed, the attack performance degrades. Second, the performance degrades less when the user model perturbation is removed.

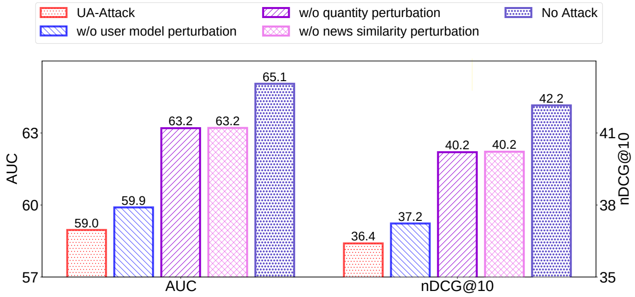


Figure 4: Ablation study of NRMS on Feeds.

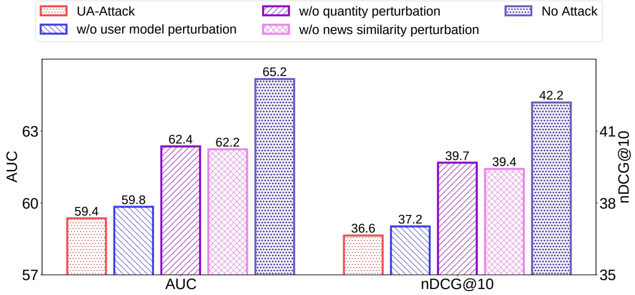
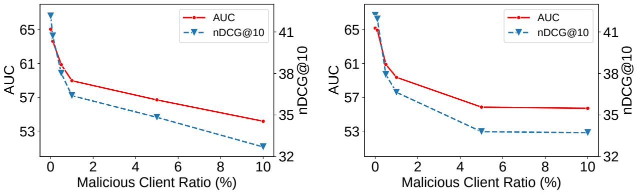


Figure 5: Ablation study of LSTUR on Feeds.

Impact of Malicious Clients Ratio on Feeds

In this subsection, we study the impact of the percentage of malicious clients on Feeds. We conduct experiments with 0.1%, 0.5%, 1%, 5% and 10% of malicious clients. The experimental results are shown in Figure 6. The observation we can make from Figure 6 is similar to that on the results presented in Section 4.5. The attack performance improves with a larger percentage of malicious clients. This is because more malicious clients will be sampled per round with a larger percentage of malicious clients, resulting in more malicious updates being uploaded, in turn making the global news recommendation model more heavily affected.



(a) on NRMS. (b) on LSTUR.

Figure 6: Impact of malicious clients ratio on Feeds.

Hyper-parameter Settings

The complete hyper-parameter settings on MIND are listed in Table 6, and the complete hyper-parameter settings on Feeds are listed in Table 7.

Table 6: Hyper-parameter settings on MIND.

Hyperparameters	NRMS	LSTUR
learning rate	0.0001	0.0001
number of negative samples P	4	4
sampled user per round k	50	50
number of rounds to update news neighbors K	100	100
malicious clients number m	500	500
dimention of news representations	400	400
dropout rate	0.2	0.2
λ_1	3.0	1.5
λ_2	3.0	1.5
λ_3	3.0	3.0
Adam β_1	0.9	0.9
Adam β_2	0.99	0.99
Adam τ	10^{-8}	10^{-8}

Table 7: Hyper-parameter settings on Feeds.

Hyperparameters	NRMS	LSTUR
learning rate	0.0001	0.0001
number of negative samples P	4	4
sampled user per round k	50	50
number of rounds to update news neighbors K	100	100
malicious clients number m	100	100
dimention of news representations	400	400
dropout rate	0.2	0.2
λ_1	3.0	3.0
λ_2	3.0	3.0
λ_3	3.0	3.0
Adam β_1	0.9	0.9
Adam β_2	0.99	0.99
Adam τ	10^{-8}	10^{-8}