A Federated Multi-View Deep Learning Framework for Privacy-Preserving Recommendations

Hao Li[*], **Mingkai Huang**, **Bing Bai**, **Chang Wang**, **Kun Bai**, **Fei Wang**[+], **Xinghua Zhu**, **Binwen Zhao**, **Ganggang Liu**, **Chao Qi**

Tencent Incorporation, [+]Cornell University

{leehaoli, mingkhuang, icebai, coracwang, kunbai, xinghuazhu, bainzhao, ganggangliu, chaoqi}@tencent.com, [+]few2001@med.cornell.edu

Abstract

Privacy protection in recommender system (RS) is gaining momentum recently due to rising concerns over user privacy and data security. Federated recommendation (FedRec) algorithms have been proposed to realize personalized privacy-preserving recommendations. However, existing FedRec algorithms extended from traditional collaborative filtering (CF) suffer from *cold-start* problems. In addition, the performance degradation from federated learning (FL) compared to centralized RS is often non-negligible. This paper presents a federated multi-view framework for privacy-preserving RS. The proposed framework could not only address the *cold-start* problem in RS, but also significantly boost the recommendation performance, by learning a federated model from multiple data sources that capture rich user-level features. The novel federated multi-view setting in the proposed framework potentially opens up new application models and brings about new security challenges to federated RS. We prove the security guarantees, and demonstrate the efficacy of our approach using a public benchmark datasets in comparison to state-of-the-art FedRec algorithms.

1 Introduction

Privacy protection in recommender systems (RS) has received heated public attention due to increasing concerns over user privacy, data security, and strict government regulations such as Europe's General Data Privacy Regulations (GDPR). Privacy-preserving recommendation is thus gaining momentum recently. Federated Learning (FL) has been recognized as one of the effective privacy-preserving machine learning paradigms for bridging data repositories while respecting data privacy. FL systems accomplish decentralized collaborative machine learning process without exposing local raw data of any FL participant. The combination of recommendation and FL has led to many federated recommendation (FedRec) algorithms in the literature.

Existing FedRec algorithms are mostly derived from collaborative filtering (CF) [3] that predict the most relevant items for users based on their interaction history. The performance overhead of a CF-based FedRec model [2, 6] is observable but acceptable comparing to conventional CF. Inherited with the CF algorithm, CF-based FedRec also suffers from cold-start problem [18]. The drawbacks of CF-based FedRec motivated an initial attempt on content-based FedRec, FedNewsRec [15]. FedNewsRec is a simple application of the vanilla FedAvg [13] on a deep neural model designed specifically for news recommendation. It is hard to be generalized to other federated RS scenarios.

In this work we propose *FL-MV-DSSM*, a federated multi-view framework for RS. The proposed framework is based on Deep Structured Semantic Models (DSSM [10]), which maps users and items to a shared semantic space for content-based recommendation. By transforming Deep Structured Semantic Models (DSSM [10]) into a horizontal federated setting [13], *FL-MV-DSSM* handles the cold-start problem in existing FedRec algorithms by utilizing the inherent features of users and items. Then, *FL-MV-DSSM* boosts the recommendation performance by learning a federated model from multiple data sources that capture richer user-level features. Moreover, *FL-MV-DSSM* presents a new federated multi-view setting, which potentially opens up new application models, e.g. jointly learning a federated model using data from different mobile phone Apps (e.g. gaming Apps and mobile App markets for accurate mobile game recommendations). But also, the proposed framework inevitably brings in new challenges, e.g. preventing data leakage among different phone Apps.

The contribution of this paper is threefold. First, to the best of our knowledge, we present the first generic content-based federated multi-view framework and several algorithms that address the cold-start problem and recommendation performance simultaneously. Second, we extend the vanilla FedAvg [13] from conventional federated setting to a new federated multi-view setting, and correspondingly present a novel approach for securely learning and aggregating multiple local models. Third, we carefully study the challenges of the new federated multi-view setting, and present a solution to guarantee its security. Empirical evaluations on *FL-MV-DSSM* and its variations with public datasets demonstrate the effectiveness of our framework.

[*]Contact Author

2 Related Work

Recommender Systems

Conventional RS can be generally divided into CF-based [3, 17] and content-based recommendation [11, 12, 10]. CF relies on user's history data to predict the most relevant items for each user. It suffers from the problem known as cold-start [18], when no history data is available for new users and items. Content-based recommendation [11, 12, 10], on the other hand, retrieves relevant items or users based on the user and items' inherent information. Thus it handles the cold-start problem better than CF, but may fail to deliver high quality recommendation when it is difficult to acquire sufficient user and item information [7].

Cross-domain RS, e.g. MV-DSSM [7] and PeterRec [19], handles cold-start problem by leveraging information from multiple data sources. MV-DSSM extends DSSM by utilizing information from different Apps simultaneously and trains a shared user sub-model, leading to better performance on item recommendation. PeterRec first learns user behavior patterns in a source domain, e.g. Tencent Kandian, and then deploys the learned patterns to a target domain, e.g. Tencent WeSee, where there exist common users.

Federated Recommender Systems

Most existing recommender systems rely on centralized data for training and prediction. FL, on the other hand, is a decentralized privacy-preserving machine learning paradigm to bridge data repositories without compromising data security and privacy. Therefore, FedRec algorithms have been proposed in federated RS to hande the increasing concerns over data privacy leakage. Similarly with conventional RS, existing FedRec can be mainly divided into CF-based (e.g. FCF [2], FedMF [6], FED-MVMF [8], etc.), and content-based (e.g. FedNewsRec [15]). Both FCF and FedMF are direct extensions of the conventional CF algorithm, with a federated training protocol protected by homomorphic encryption(HE) [16]. FED-MVMF extends FCF to matrix factorization with multiple data sources, and thus outperforms FCF, which only uses a single data source. CF-based FedRec algorithms are inherently subject to the cold start problem [18]. FedNewsRec is the first content-based FedRec system. The recommendation quality of FedNewsRec is heavily dependant on its model design, which is carefully tailored for news recommendation only. The proposed *FL-MV-DSSM* and its variations are content-based FedRec algorithms based on the general DSSM architecture. The flexibility of the proposed framework potentially enables it to be optimized for various recommendation tasks.

3 Preliminaries

Problem Definition. The proposed federated multi-view RS aims at learning a model over data residing on m distributed nodes and n isolated "views" on each distributed node. A view is often the user behavior on a distinct App, such as Fortnite (games), Google Play (app market), etc. User behavior data is critical to create user personas that help understand user needs, experiences, etc. However, collecting user behavior data gets more and more difficult as the law gets stricter, without technology like FL. We thus denote the decentralized federated multi-view datasets on a distributed node, or a FL client, as $D^n = (\mathbf{U}_1, \mathbf{I}), \ldots, (\mathbf{U}_i, \mathbf{I}), \ldots, (\mathbf{U}_n, \mathbf{I})$, where all user view datasets $\mathbf{U} \in \mathbb{R}^{n \times d_{\mathbf{U}}}$ are generated by different views, and item dataset $\mathbf{I} \in \mathbb{R}^{d_{\mathbf{I}}}$ is downloaded from server, e.g. a mobile App's cloud services. The semantic vectors, or embeddings, can be extracted from each view-level user dataset $\mathbf{U}_i \in \mathbb{R}^{d_{\mathbf{U}_i}}$ and item dataset $\mathbf{I}$ respectively by using deep models like DSSM. Our goal is to find a non-linear mapping $f(\cdot)$ for each view such that the sum of similarities, in the semantic space between mapping of all user view datasets $\mathbf{U}$ and item dataset $\mathbf{I}$, is maximized on each client. Our objective function on each FL client is defined as follows:

$$\underset{\mathbf{W}_{\mathbf{I}}, \mathbf{W}_{\mathbf{U}_1}, \ldots, \mathbf{W}_{\mathbf{U}_n}}{\operatorname{argmax}} \sum_{j=1}^{S} \frac{\exp(\gamma \cos(\mathbf{y}_{\mathbf{I}}, \mathbf{y}_{i,j}))}{\sum_{\mathbf{X}' \in \mathbf{U}_i} \exp(\gamma \cos(\mathbf{y}_{\mathbf{I}}, f_i(\mathbf{X}', \mathbf{W}_i)))}, \tag{1}$$

where S denote the number of positive user-item pair $(\mathbf{X}_{\mathbf{U}_i,j}, \mathbf{X}_{\mathbf{I}_j})$, i is the index of the view $\mathbf{U}_i$ in sample j, $\mathbf{y}$ denote the mapping result of $f(\cdot)$, and γ is the temperature parameter.

Security Definition. In addition to the security requirements from vanilla FL, *FL-MV-DSSM* requires extra security guarantees. In our federated multi-view setting, although all views collaboratively train a model with datasets $\mathbf{U}$ and $\mathbf{I}$ on each FL client, there should be no raw data interaction between views since each dataset $\mathbf{U}_i$ contains private view-specific information that should be protected. Moreover, each view's contribution to the item sub-model, learned from shared local dataset $\mathbf{I}$, should be protected as well since a malicious view could otherwise infer an innocent view's raw data from her gradients [20] by monitoring her changes to the shared local item sub-model.

Threat Models. We consider the following threat models in our federated multi-view setting:

- **[*Vanilla FL*]**: FL clients and/or FL server are active adversaries who deviate from the FL protocol, e.g., sending incorrect and/or arbitrarily chosen messages to honest users, aborting, omitting messages, and sharing their entire view of the protocol with each other, and also with the server if server is an active adversary.
- **[*Federated Multi-View*]**: Certain view can be fully malicious, which means as an App, it would act arbitrarily, e.g. monitoring network interface to observe an innocent view's network traffic, making null updates to shared local item sub-model to infer an innocent view's update, or monitoring changes of item sub-model, etc., in order to infer an innocent view's data information.

In addition, we make the following assumption:

- **[*View-Level Isolation*]**: Each view's dataset $\mathbf{U}_i$ and model $\mathbf{W}_{\mathbf{U}_i}$ are only accessible to the i-th view, such that malicious view cannot access $\mathbf{U}_i$ and $\mathbf{W}_{\mathbf{U}_i}$. The isolation can be achieved through encryption or TEE [14].

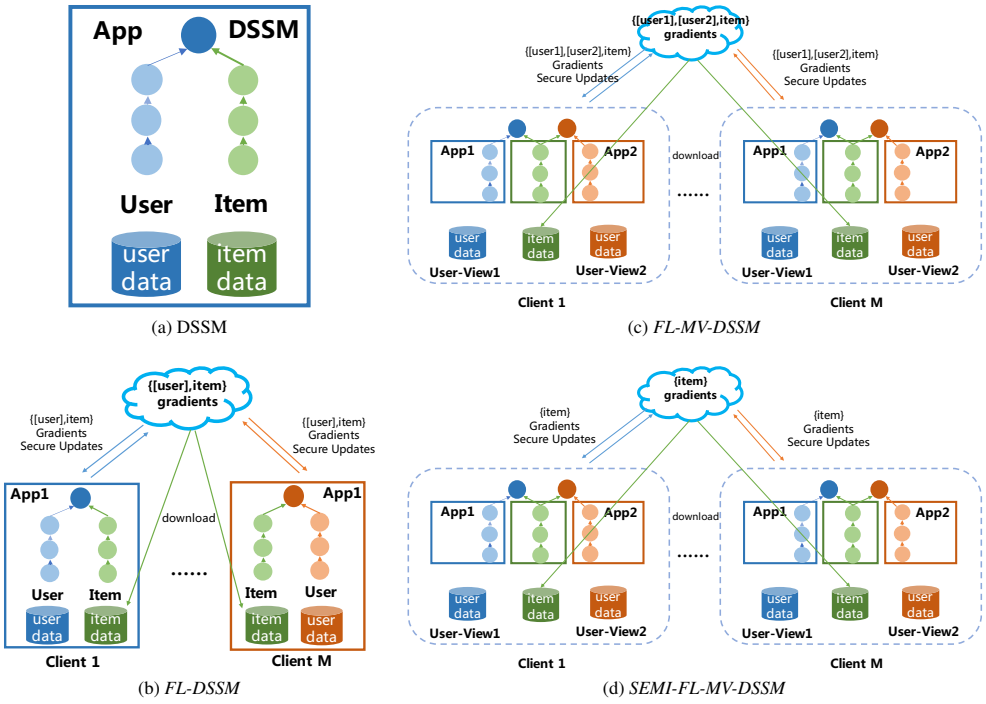

Figure 1: *FL-MV-DSSM* and its variations.

4 *FL-MV-DSSM*: Generic Multi-View Federated Recommendation Framework

This section presents our generic multi-view federated learning framework. We first give a brief review of the DSSM architecture. Then, regarding recommendation scenario, we introduce *FL-MV-DSSM*'s training and prediction algorithms respectively, and show how *FL-MV-DSSM* guarantees data privacy and user privacy between different views in federated multi-view setting. Variations of *FL-MV-DSSM*, including *FL-DSSM* and *SEMI-FL-MV-DSSM*, as shown in Figure 1, are also introduced in this section.

4.1 Deep Structured Semantic Models (DSSM).

DSSM [10], originally designed for web search, extracts semantic vectors from users' queries and candidate documents, by multi-layer neural networks. Cosine similarity between semantic vectors is employed to measure the relevance between a query and document pair. In the proposed generic federated multi-view recommendation setting, we adopt DSSM as the basic model (see Figure 1a), and extend it into *FL-MV-DSSM*. Specifically, in *FL-MV-DSSM*, DSSM's user query is equivalent to *FL-MV-DSSM*'s user feature of i-th view $\mathbf{U}_i$, and document is equivalent to item $\mathbf{I}$.

More formally, if we denote $\mathbf{x}$ as the original feature vector of query words or documents, $\mathbf{y}$ as the semantic vector, $l_i, i = 1, \ldots, N-1$, as the intermediate hidden layers, $\mathbf{W}_i$ as the i-th weight matrix, and b_i as the i-th bias term, we have DSSM's forward propagation process defined as:

$$\begin{aligned} l_1 &= \mathbf{W}_1 \mathbf{x}, \\ l_i &= f(\mathbf{W}_i l_{i-1} + b_i), i = 2, \ldots, N-1, \\ \mathbf{y} &= f(\mathbf{W}_N l_{N-1} + b_N). \end{aligned} \tag{2}$$

The semantic relevance score between a query Q and a document D is measured as $R(Q, D) = cosine(\mathbf{y}_Q, \mathbf{y}_D) = \mathbf{y}_Q^\top \mathbf{y}_D / (\|\mathbf{y}_Q\| \cdot \|\mathbf{y}_D\|)$, where $\mathbf{y}_Q$ and $\mathbf{y}_D$ are semantic vectors of query and document, respectively.

We assume that a query is relevant to the documents that are clicked on for that query, and the parameters of the DSSM, i.e., the weight matrix $\mathbf{W}$ are optimized to maximize the conditional likelihood of the clicked documents given queries. The posterior probability of a document given a query is calculated from

$$\mathbb{P}(D|Q) = \frac{\exp(\gamma R(Q, D))}{\sum_{D' \in \mathbf{D}} \exp(\gamma R(Q, D'))}, \tag{3}$$

where γ is the temperature parameter in the softmax function. $\mathbf{D}$ denotes the set of candidate documents to be ranked. In practice, for each pair of query and the clicked-document, denoted by (Q, D^+), we approximate $\mathbf{D}$ by the union of D^+ and S randomly selected unclicked documents, denote by $\{D_j^-; j = 1, ..., S\}$. The target function to be minimized is then defined as

$$L(\Lambda) = -\log \prod_{(Q, D^+)} \mathbb{P}(D^+|Q), \tag{4}$$

where Λ denotes the parameter set of the neural networks.

4.2 *FL-MV-DSSM* Training Algorithm

Algorithm 1 elaborates *FL-MV-DSSM*'s training algorithm, $FedMvDssmTrain$. Clients with user behavior data w.r.t. item dataset $\mathbf{I}$ are selected to participate into the training phase. Within each view i, gradients of the user sub-model and item sub-model are calculated based on the ith view's

Algorithm 1: The $FedMvDssmTrain$ algorithm

FL Client:

Input: Dataset $\mathbf{D} = \{(\mathbf{X}_i, \mathbf{y}), i \in \{1, \ldots, N\}\}$. Number of FL training round T. Learning rate η. Initial user sub-models $\{\mathbf{W}^0_{\mathbf{U}_1}, \ldots, \mathbf{W}^0_{\mathbf{U}_N}\}$. Initial item sub-model weights $\mathbf{W}^0_{\mathbf{I}}$.

Output: User sub-model weights $\{\mathbf{W}^T_{\mathbf{U}_1}, \ldots, \mathbf{W}^T_{\mathbf{U}_N}\}$. Item sub-model weights $\mathbf{W}^T_{\mathbf{I}}$.

1 **for** $k = 1 : T$ **do**

2 **for** *each view* $i = 1 : N$ **do**

3 $(\mathbf{g}^k_{\mathbf{I}})_i = \frac{\partial L(\mathbf{W}^k_{\mathbf{I}}, \mathbf{W}^k_{\mathbf{U}_i}, \mathbf{X}_i, \mathbf{y})}{\partial \mathbf{W}^k_{\mathbf{I}}}$,

4 $(\mathbf{g}^k_{\mathbf{U}_i})_i = \frac{\partial L(\mathbf{W}^k_{\mathbf{I}}, \mathbf{W}^k_{\mathbf{U}_i}, \mathbf{X}_i, \mathbf{y})}{\partial \mathbf{W}^k_{\mathbf{U}_i}}$.

5 **end**

6 $\mathbf{g}^k_{\mathbf{I}} =$ **local_secure_aggregate**$(\{(\mathbf{g}^k_{\mathbf{I}})_i\}_{i=1}^N)$.

7 $\mathbf{G}^k_{\mathbf{I}} =$ **remote_secure_aggregate**$(\mathbf{g}^k_{\mathbf{I}})$.

8 **if** *aggregate_user_sub-model* **then**

9 **for** *each view* $i = 1 : N$ **do**

10 $(\mathbf{G}^k_{\mathbf{U}_i})_i =$ **remote_secure_aggregate**$((\mathbf{g}^k_{\mathbf{U}_i})_i)$.

11 **end**

12 **end**

13 $\mathbf{W}^{k+1}_{\mathbf{I}} = \mathbf{W}^k_{\mathbf{I}} - \eta \mathbf{G}^k_{\mathbf{I}}$.

14 **for** *each view* $i = 1 : N$ **do**

15 $\mathbf{W}^{k+1}_{\mathbf{U}_i} = \mathbf{W}^k_{\mathbf{U}_i} - \eta \mathbf{G}^k_{\mathbf{U}_i}$.

16 **end**

17 **end**

FL Server:

Input: Number of FL training round T. Client updates $\mathbf{g}^k_j$ in FL round k.

Output: Securely aggregate FL client's summative information, e.g. gradients.

18 **for** $k = 1 : T$ **do**

19 **server_secure_aggregate**$(\{\mathbf{g}^k_j\}_{j \in \mathbf{S} \subseteq \{1, \ldots, M\}})$

20 **end**

private user data $\mathbf{U}_i$ and locally shared item data $\mathbf{I}$. Although *FL-MV-DSSM* is a content-based FedRec, we empirically found that aggregating gradients of the item sub-model leads to better recommendation performance, comparing to only aggregating gradients of user sub-model. This finding is also coherent with the results of CF-based FedRec [2, 6]. Thus in *FL-MV-DSSM*, gradients of item sub-model will be aggregated in a federated manner, while the aggregation of user gradients is configurable, by the *aggregate_user_sub-model* flag in line 8 of Algorithm 1. Setting *aggregate_user_sub-model* to false leads to a variation of *FL-MV-DSSM*, *SEMI-FL-MV-DSSM*. After each FL training round, both user and item sub-models are updated according to the new global gradients distributed by FL server.

The gradients for both user and item sub-models contain view-specific information that must be protected. *FL-MV-DSSM* introduces two secure aggregation primitives, **local_secure_aggregate()** and **remote_secure_aggregate()**, to secure both local and remote gradients aggregation. Details of both secure aggregation primitives will be specified in Section 4.4.

4.3 *FL-MV-DSSM* Prediction Algorithm

Algorithm 2 elaborates *FL-MV-DSSM*'s prediction algorithm, $FedMvDssmPredict$. For each item $\mathbf{x}_{\mathbf{I}_j}$, old or new, item sub-model output its semantic vector, or embedding, $\mathbf{y}_{\mathbf{I}_j}$. Meanwhile for user's output, $\mathbf{y}_{\mathbf{U}}$, which is locally secure aggregated from user sub-models in multiple views, is used to compare against all $\mathbf{y}_{\mathbf{I}_j}$s to determine their similarities. Based on the similarity scores, *FL-MV-DSSM* will output the top-K items for the user, old or new.

Algorithm 2: The $FedMvDssmPredict$ algorithm

FL Client:

Input: User sub-models $\{\mathbf{W}_{\mathbf{U}_i}\}, i \in \{1, \ldots, N\}$. Item sub-model $\mathbf{W}_{\mathbf{I}}$. User features $\mathbf{x}_{\mathbf{U}_i}$. Item features $\mathbf{x}_{\mathbf{I}_j}$.

Output: List of top-K items for recommendation.

1 **for** *each item* $j = 1 : M$ **do**

2 Compute $\{\mathbf{y}_{\mathbf{I}_j} = f(\mathbf{W}_{\mathbf{I}}, \mathbf{x}_{\mathbf{I}_j})\}$ according to Eq.(2).

3 **end**

4 **for** *each item* $j = 1 : M$ **do**

5 **for** *each view* $i = 1 : N$ **do**

6 Compute $\mathbf{y}_{\mathbf{U}_i} = f(\mathbf{W}_{\mathbf{U}_i}, \mathbf{x}_{\mathbf{U}_i})$ according to Eq.(2).

7 $\mathbb{P}(\mathbf{x}_{\mathbf{I}_j} | \mathbf{x}_{\mathbf{U}_i}) = \frac{\exp(\gamma \cos(\mathbf{y}_{\mathbf{U}_i}, \mathbf{y}_{\mathbf{I}_j}))}{\sum_{\mathbf{y}'_{\mathbf{I}} \in \{\mathbf{y}_{\mathbf{I}_j}\}} \exp(\gamma \cos(\mathbf{y}_{\mathbf{U}_i}, \mathbf{y}'_{\mathbf{I}}))}$.

8 **end**

9 $\mathbb{P}(\mathbf{x}_{\mathbf{I}_j} | \mathbf{x}_{\mathbf{U}}) =$ **local_secure_aggregate**$(\{\mathbb{P}(\mathbf{x}_{\mathbf{I}_j} | \mathbf{x}_{\mathbf{U}_i})\}, i \in \{1, \ldots, N\})$.

10 **end**

11 **return** *top-K items sorted by* $\{\mathbb{P}(\mathbf{x}_{\mathbf{I}_j} | \mathbf{x}_{\mathbf{U}})\}_{j=1}^M$.

4.4 Secure Primitives for Privacy Protection

To defend against the threat models defined in Section 3, *FL-MV-DSSM* adopts two secure primitives, **local_secure_aggregate()** and **remote_secure_aggregate()** during training and prediction. The purpose of the two primitives is to securely aggregate vectors, locally or remotely, without exposing raw data to other participants or the curator. However, different execution environment leads to different implementations for both primitives.

remote_secure_aggregate()

The **remote_secure_aggregate()** primitive is used in $FedMvDssmTrain$ to securely aggregate $(\mathbf{g}^k_{\mathbf{U}_i})_i$ and $\mathbf{g}^k_{\mathbf{I}}$ over all FL clients. The secure aggregation across large number of clients is well studied in conventional FL [4]. We follow the secure aggregation methods and proof of the proposed "Secure Aggregation Protocol" (SAP) in [4] to realize **remote_secure_aggregate()** in *FL-MV-DSSM*. Thus, we ensure that the FL server only observers the aggregated result without knowing each FL client's update. The SAP also gracefully handles the client drop-out problem.

local_secure_aggregate()

The **local_secure_aggregate()** is deployed on FL client devices. It is used to

1. securely aggregate n gradients of item sub-models in n different views during training; and
2. securely aggregate n probabilities, each of which is the possibility that an item is interesting to a user according to a user view $\mathbf{U}_i$, during prediction.

The SAP used for **remote_secure_aggregate()** is only secure when the number of aggregation participants is large. In practice, n is often small due to limited source of views. When $n = 2$, MPC-based aggregation protocols such as SAP fail to counteract the inference attacks from opposing participant. In addition, the computation cost of MPC-based solution is quadratic in FL client [4]. Therefore we leverage differential privacy (DP) to realize **local_secure_aggregate()**.

Specifically, we apply Gaussian Mechanism [1] in DP to line 6 in $FedMvDssmTrain$, by adding noise to each gradients of item sub-model $(\mathbf{g}_\mathbf{I}^k)_i$, before aggregation. For $FedMvDssmPredict$, we add Gaussian noise to each probability before aggregation. Following the moments accountant algorithm [1], the concrete steps are:

Step 1. **Random sub-sampling.** For n views in each client, a random subset B ($|B|_1 \leq n$) is sampled in each FL round.

Step 2. **Gradient clipping.** Clip each gradient by its ℓ_2 norm, i.e., the gradient $(\mathbf{g}_\mathbf{I}^k)_i$ is replaced by $(\mathbf{g}_\mathbf{I}^k)_i / \max(1, \frac{\|(\mathbf{g}_\mathbf{I}^k)_i\|_2}{C})$, for a clipping threshold C.

Step 3. **Distorting.** A Gaussian Mechanism is used to distort the sum of all updates.

$$\widetilde{(\mathbf{g}_\mathbf{I}^k)} = \frac{1}{|B|}(\sum_{i \in B}(\mathbf{g}_\mathbf{I}^k)_i + \mathcal{N}(0, \sigma^2 C^2)), \qquad (5)$$

where the value of σ satisfies the Theorem 1 in [1].

From the Eq. 5, the average distortion is governed by the value of σ and C, and from the Theorem 1 in [1], we can know that the value of σ is inversely proportional to the value of privacy budget ϵ. For example, with $|B|_1 = n$, $\sigma = 4$, $\delta = 10^{-3}$, and $T = 100$, we have $\epsilon \approx 4.33$ using the moments accountant theory [1]. We can reduce the noise scale by reducing the value of $|B|_1$ and training rounds T while ensuring our model performance.

4.5 *FL-MV-DSSM* Variation and Extension

The proposed *FL-MV-DSSM* framework is indeed a general protocol that directs the FL of user- and item- representations from multiple views and devices. The embeddings of the users and items are extracted by DSSM in this paper, but can be extended to any representation learning algorithms, such as sequence-based user profiling, graph embedding, etc. The aggregation itself can take also forms such as weighted average, concatenation, or even a neural feature selector.

In the following section, we demonstrate the performance of *FL-MV-DSSM* using DSSM and the simple average aggregation. Two variations of *FL-MV-DSSM* are compared in the experiments. *FL-DSSM* is a degraded version of *FL-MV-DSSM* when the number of views is 1, and conducts secure aggregation of global user and item sub-models respectively. By configuring the *aggregate_user_sub-model* flag to $false$ in $FedMvDssmTrain$, we have the semi-global variation of the proposed framework, *SEMI-FL-MV-DSSM*, where global aggregation is only conducted on the item representations.

Table 1: Structures of item and user sub-models.

	Item	User (View-1)	User (View-2)
Input	Input (4739)	Input (23)	Input (30)
Layer1	Dense (64)	Dense (64)	Dense (64)
Layer2	Dense (32)	Dense (32)	Dense (32)
Layer3	Dense (16)	Dense (16)	Dense (16)

5 Experiments

This section evaluates *FL-MV-DSSM* and its variations. We have proved in Section 4.4 that *FL-MV-DSSM* is able to protect data privacy among different views. Now we empirically address the following questions: (**Q1**) Is *FL-MV-DSSM* able to alleviate the cold-start problem? (**Q2**) How is recommendation performance of *FL-MV-DSSM*, and its variations?

5.1 Environmental Setup

We implement *FL-MV-DSSM* framework and its variations on Google's TensorFlow Federated (TFF) simulation framework. For multi-view scenarios, we take two user views as an example. Table 1 shows the DSSM model architectures in our implementation.

Data Pre-Processing. Similar to existing FedRec algorithms, we use the popular public dataset, MovieLens-100K [9], for our evaluations. The MovieLens-100K not only consists of 100K ratings from 943 users on 1682 items (movies), but also contains user and item information, e.g. user's age and movie's title. For label pre-processing, we create implicit feedbacks as 1 for all ⟨user, item⟩ pairs where a user explicitly interacted with an item in the dataset, and 0 for the rest. For user features pre-processing, we randomly sample a portion of MovieLens data and select age (normalized to less than or equal to 1), gender (binary feature), and occupation (one-hot vector) as user stataic features for one view (**View-1**); meanwhile we use user embeddings learned by singular value decomposition (SVD) from MovieLens' interaction matrix, orthogonal to the data of View-1, as user dynamic features for another view (**View-2**). For item feature pre-processing, we select title and genre, coded with 3-gram representation and a series of bits, respectively.

Evaluation Metrics. Precision@10, Recall@10, NDCG@10, and AUC are used in our experiments for recommendation performance evaluation. Among these metrics, Precision@10, Recall@10 and NDCG@10 only concentrate on the very top of recommendation list, while AUC evaluates the overall accuracy of recommendations.

Hyper-parameter setting. Adam optimizer with learning rate 0.001 is used in centralized training and in server-side federated training. SGD optimizer with learning rate 0.2 is used in client-side federated training. Batch size is set to 20. The dimension is set to 30 for matrix factorization. For DP parameters, $C = 0.5, \sigma = 1, \delta = 0.001$, and $|B| = 2$.

5.2 Cold-Start Recommendations

To evaluate cold-start recommendations, we mainly focus on *FL-MV-DSSM*, and examine three cold-start scenarios: 1) cold-start users (CS-users), 2) cold-start items (CS-items), and 3) cold-start users-items (CS-users-items). For CS-users, a random subset of 10% users and their interaction data are excluded during the model training. Model parameters are learned with the remaining 90% of the users and their interaction data. For CS-items, a random subset of 10% items are left-out during model training. For CS-users-items, a random subset of 10% users and items are excluded from the model training and model parameters are learned with the rest of users, interaction data, and items. We use the 10% held-out datasets in all three scenarios as our testing datasets.

Table 2 shows the results of three cold-start recommendations. The results support that *FL-MV-DSSM* can be used for cold-start recommendation reliably. Particularly, *FL-MV-DSSM* achieves good cold-start prediction performance for new users, which is valuable for privacy-preserving recommendations since new users are continuously enrolled in the recommendation service. However, the performance of cold-start items and users-items are lower than that of cold-start users. The reason behind this might be that in our datasets, the difference between users (considering age, gender, occupation, etc.) is less significant than that of items (movie title, genres, etc.), and *FL-MV-DSSM* could learn this difference correctly and recommend with higher precision.

5.3 Recommendation Performance

To evaluate recommendation performance, we examine *FL-MV-DSSM*, *FL-DSSM*, *SEMI-FL-MV-DSSM*, centralized DSSM, and existing FedRec algorithms including FCF[2] and FED-MVMF[8]. Specifically, for *FL-MV-DSSM* related evaluations, we use two generated user views, View-1 and View-2 as described in Section 5.1, to train two user sub-models. For *FL-MV-DSSM* and *SEMI-FL-MV-DSSM*, we randomly select 100 users within each FL training round, and for each user, the two user sub-models share a single item sub-model, as introduced in Figure 1c and Figure 1d, to form a *FL-MV-DSSM* task. For *FL-DSSM*, we launch two FL training tasks, each of which randomly selects 100 users within each FL training round, and each user sub-model is paired with a item sub-model. For centralized DSSM, the datasets for all users are centralized and trained at the single place. For both FCF and FED-MVMF, we follow the experimental setups in their papers. All the datasets, centralized or decentralized, are randomly divided into an 80% training set and 20% testing set.

Figure 2 illustrates the training performance, precision and recall, of centralized DSSM, decentralized *FL-MV-DSSM*, and its variations. Table 3 lists the recommendation results of existing FedRec algorithms, *FL-MV-DSSM*, and its variations. From the results we can see that among *FL-MV-DSSM* variations, *FL-MV-DSSM* achieves better performance than *FL-DSSM*, since *FL-MV-DSSM* can incorporate more user features from multiple views, e.g. from multiple user Apps, to jointly train a better model. Among FedRec algorithms including FCF and FED-MVMF, interestingly, we find *SEMI-FL-MV-DSSM*, which aggregates only the shared item sub-model rather than user sub-models, achieves the best performance. This is understandable in that for all FedRec algorithms, their performance data are collected through "federated evaluation [5]", and the performance of user sub-model will fit to user local data fast if not aggregating the contributions from other FL participants.

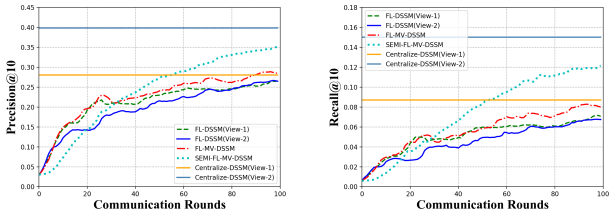

Figure 2: Recommendation precision (left) and recall (right) of *FL-MV-DSSM* and its variations.

6 Conclusions

This paper presents *FL-MV-DSSM*, the first generic content-based federated multi-view framework that could address cold-start problem in existing FedRec algorithms, and meanwhile improve recommendation performance. Besides, this paper extends the vanilla federated setting into a new federated multi-view setting, which might potentially enable new usage models of FL in recommendation scenarios and bring in new security challenges. By carefully studying the challenges, this paper presents a novel solution addressing the security requirements. Empirical evaluations on *FL-MV-DSSM* and its variations with public datasets demonstrate the efficacy of our approach.

References

[1] M Abadi, Chu A, I Goodfellow, and et al. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[2] M Ammad-ud-din, E Ivannikova, S.A Khan, and et al. Federated collaborative filtering for privacy-preserving personalized recommendation system. *CoRR*, abs/1901.09888, 2019.

[3] R.M Bell and Y Koren. Improved neighborhood-based collaborative filtering. 2007.

[4] K Bonawitz, V Ivanov, B Kreuter, and et al. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 1175–1191, New York, NY, USA, 2017. Association for Computing Machinery.

[5] S Caldas, P Wu, T Li, and et al. LEAF: A benchmark for federated settings. *CoRR*, abs/1812.01097, 2018.

[6] D Chai, L Wang, K Chen, and Q Yang. Secure federated matrix factorization. *CoRR*, abs/1906.05108, 2019.

[7] A.M Elkahky, Y Song, and X He. A multi-view deep learning approach for cross domain user modeling in

Table 2: Cold-Start recommendation performance of *FL-MV-DSSM* on MovieLens dataset with three scenarios. The values denote the mean $\pm$ standard deviation across 3 different model builds.

	Precision@10	Recall@10	NDCG@10	AUC
CS-Users	0.4574 ± 0.0085	0.0696 ± 0.0100	0.3697 ± 0.0031	0.7793 ± 0.0617
CS-Items	0.1586 ± 0.0002	0.1086 ± 0.0015	0.1318 ± 0.0008	0.5809 ± 0.0721
CS-Users-Items	0.1408 ± 0.0033	0.1335 ± 0.0315	0.1232 ± 0.0024	0.5672 ± 0.0043

Table 3: Recommendation performance on MovieLens dataset, after 100 FL training rounds. The values denote the mean $\pm$ standard deviation of different model builds and their performance comparison with the FCF in the brackets.

	Precision@10	Recall@10	NDCG@10	AUC
Centralize-DSSM(View-1)	0.2804 ± 0.0011	0.0870 ± 0.0036	0.2389 ± 0.0013	0.8479 ± 0.0009
Centralize-DSSM(View-2)	0.3980 ± 0.0025	0.1501 ± 0.0009	0.3401 ± 0.0035	0.9202 ± 0.0011
FCF [2]	0.3010 ± 0.0032	0.1017 ± 0.0019	0.2606 ± 0.0008	0.8627 ± 0.0003
FED-MVMF [8]	0.3223 ± 0.0017 (+7.08%)	0.1193 ± 0.0041 (+17.31%)	0.2797 ± 0.0038 (+7.33%)	0.8833 ± 0.0017(-2.39%)
FL-DSSM(View-1)	0.2691 ± 0.0015 (−10.60%)	0.0721 ± 0.0027 (−29.11%)	0.2292 ± 0.0018 (−12.05%)	0.8445 ± 0.0523 (−2.11%)
FL-DSSM(View-2)	0.2656 ± 0.0021 (−11.76%)	0.0676 ± 0.0137 (−33.53%)	0.2186 ± 0.0023 (−16.12%)	0.8398 ± 0.0314 (−2.65%)
FL-MV-DSSM	0.2845 ± 0.0034 (−5.48%)	0.0805 ± 0.0113 (−20.85%)	0.2369 ± 0.0104 (−9.09%)	0.8490 ± 0.0033 (−1.59%)
SEMI-FL-MV-DSSM	$\mathbf{0.3512 \pm 0.0070}$ (+16.68%)	$\mathbf{0.1217 \pm 0.0214}$ (−19.67%)	$\mathbf{0.3136 \pm 0.0038}$ (+20.34%)	$\mathbf{0.8986 \pm 0.0051}$ (+4.16%)

recommendation systems. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, page 278–288, Republic and Canton of Geneva, CHE, 2015. International World Wide Web Conferences Steering Committee.

[8] A Flanagan, W Oyomno, A Grigorievskiy, and et al. Federated multi-view matrix factorization for personalized recommendations. *arXiv preprint arXiv:2004.04256*, 2020.

[9] F.M Harper and J.A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

[10] P.S Huang, X He, J Gao, and et al. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, page 2333–2338, New York, NY, USA, 2013.

[11] G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[12] J Liu, P Dolan, and E.R Pedersen. Personalized news recommendation based on click behavior. In *Proceedings of the 15th International Conference on Intelligent User Interfaces*, IUI '10, page 31–40, New York, NY, USA, 2010. Association for Computing Machinery.

[13] B McMahan, E Moore, D Ramage, and et al. Communication-efficient learning of deep networks from decentralized data. In Aarti Singh and Xiaojin (Jerry) Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54, pages 1273–1282. PMLR, 2017.

[14] S Pinto and N Santos. Demystifying arm trustzone: A comprehensive survey. *ACM Comput. Surv.*, 51(6), January 2019.

[15] T Qi, F Wu, C Wu, and et al. Fedrec: Privacy-preserving news recommendation with federated learning. *arXiv preprint arXiv:2003.09592*, 2020.

[16] R.L Rivest, L Adleman, and M.L Dertouzos. On data banks and privacy homomorphisms. *Foundations of Secure Computation, Academia Press*, pages 169–179, 1978.

[17] B Sarwar, G Karypis, J Konstan, and J Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, page 285–295, New York, NY, USA, 2001. Association for Computing Machinery.

[18] A.I. Schein, A Popescul, L.H. Ungar, and D.M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '02, page 253–260, New York, NY, USA, 2002. Association for Computing Machinery.

[19] Fajie Yuan, Xiangnan He, Alexandros Karatzoglou, and Liguang Zhang. Parameter-efficient transfer from sequential behaviors for user modeling and recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1469–1478, New York, NY, USA, 2020. Association for Computing Machinery.

[20] L Zhu, Z Liu, and S Han. Deep leakage from gradients. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 14774–14784. Curran Associates, Inc., 2019.