Hierarchical Personalized Federated Learning for User Modeling

Jinze Wu[1], Qi Liu[1,*], Zhenya Huang[1], Yuting Ning[1], Hao Wang[1], Enhong Chen[1]
Jinfeng Yi[2], Bowen Zhou[2]
[1]Anhui Province Key Laboratory of Big Data Analysis and Application,
School of Computer Science and Technology, University of Science and Technology of China
[2]JD AI Research
{hxwjz,ningyt,wanghao3}@mail.ustc.edu.cn,{qiliuql,huangzhy,cheneh}@ustc.edu.cn,
{yijinfeng,bowen.zhou}@jd.com

ABSTRACT

User modeling aims to capture the latent characteristics of users from their behaviors, and is widely applied in numerous applications. Usually, centralized user modeling suffers from the risk of privacy leakage. Instead, federated user modeling expects to provide a secure multi-client collaboration for user modeling through federated learning. Existing federated learning methods are mainly designed for consistent clients, which cannot be directly applied to practical scenarios, where different clients usually store inconsistent user data. Therefore, it is a crucial demand to design an appropriate federated solution that can better adapt to user modeling tasks, and however, meets following critical challenges: 1) *Statistical heterogeneity.* The distributions of user data in different clients are not always independently identically distributed which leads to personalized clients; 2) *Privacy heterogeneity.* User data contains both public and private information, which have different levels of privacy. It means we should balance different information to be shared and protected; 3) *Model heterogeneity.* The local user models trained with client records are heterogeneous which need flexible aggregation in the server. In this paper, we propose a novel client-server architecture framework, namely Hierarchical Personalized Federated Learning (HPFL) to serve federated learning in user modeling with inconsistent clients. In the framework, we first define hierarchical information to finely partition the data with privacy heterogeneity. On this basis, the client trains a user model which contains different components designed for hierarchical information. Moreover, client processes a fine-grained personalized update strategy to update personalized user model for statistical heterogeneity. Correspondingly, the server completes a differentiated component aggregation strategy to flexibly aggregate heterogeneous user models in the case of privacy and model heterogeneity. Finally, we conduct extensive experiments on real-world datasets, which demonstrate the effectiveness of the HPFL framework.

CCS CONCEPTS

• **Security and privacy → Privacy protections**; • **Human-centered computing → User models**.

KEYWORDS

User modeling, Federated learning, Privacy heterogeneity, Model personalization

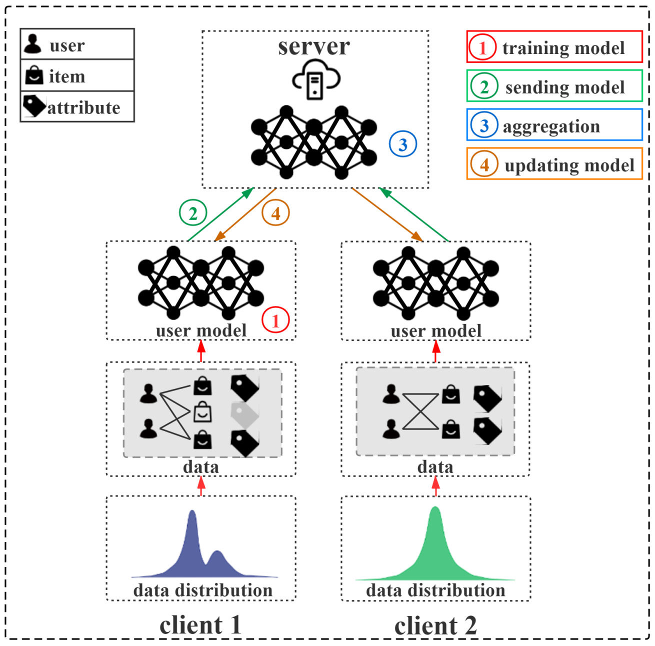
ACM Reference Format:
Jinze Wu, Qi Liu, Zhenya Huang, Yuting Ning, Hao Wang, Enhong Chen, Jinfeng Yi, Bowen Zhou. 2021. Hierarchical Personalized Federated Learning for User Modeling. In *Proceedings of the Web Conference 2021 (WWW '21), April 19-23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3442381.3449926

1 INTRODUCTION

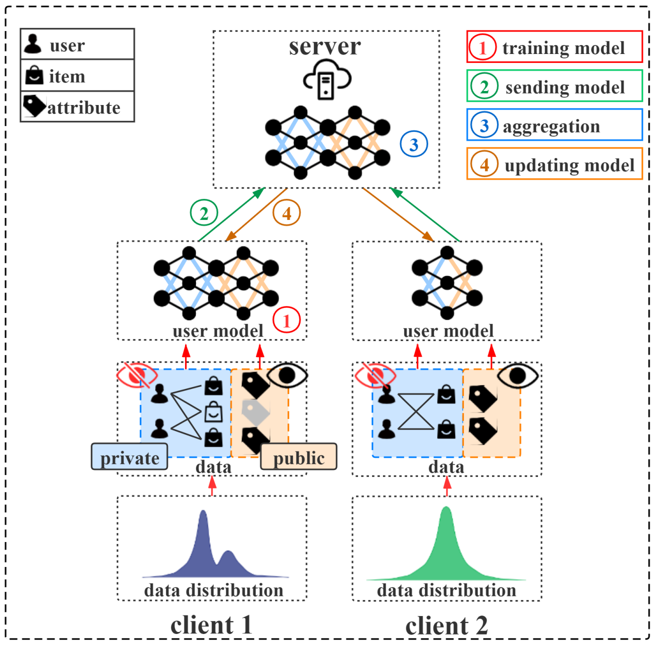
User modeling is an important basis to help researchers capture useful potential characteristics with the reliance on personal data [59]. It has been applied to multiple typical tasks to provide appropriate user models for users with various demands, such as modeling capabilities or preferences from users [10]. For example in intelligent education systems, user modeling assists cognitive diagnosis for modeling student capacities [51], while in recommender systems of e-commerce, user modeling assists for modeling customer preferences [18]. In general, user modeling processes centralized training with data aggregated, which causes privacy leakage. Considering the privacy and sensitivity of personal data, regulations such as General Data Protection Regulation (GDPR) are enacted to restrict the centralized use of private data [4, 49, 50]. Therefore, user data is required to remain local (e.g., personal devices), which results in data isolation scenarios [35]. Focus on this dilemma, federated learning (FL) has received widespread attention for potential of secure distributed user modeling [42, 43]. It refers to building and aggregating user models while leaving private data isolated so that preserves the data privacy [39].

Generally, standard federated learning is an iterative framework of a client-server architecture as shown in Figure 1(a). In particular, there are two core parts in the framework. One is the client, where all clients train local models on user data individually based on the same global model and send them to a central server. The other is the server, which aggregates the homogeneous local models to a global one. During the process, the data of the clients is strictly kept locally, and only the model information are interacted, which guarantees the privacy protection. In the literature, many efforts have been made improving federated learning technically, such as FedSGD, FedAvg [39], FedAtt [23] and FedProx [27]. Though some great performances have been achieved, previous works are "dance in shackles", that is, current federated learning frameworks are designed for consistent clients [16]. In particular, on one hand,

*Corresponding Author.



(a) standard federated learning



(b) hierarchical personalized federated learning

Figure 1: Differences between standard federated learning (left), and our hierarchical personalized federated Learning (right) for user modeling. Standard FL simply aggregates and updates the consistent entire user models indiscriminately, while HPFL partitions and processes the different components of the heterogeneous models independently. The top part shows a server with a global model. The bottom shows clients with Non-IID data and local user models. Each round consists of four steps: training a model locally, sending the model to the server, model aggregation in the server and updating models for clients.

researchers assume that the data in clients are consistent, i.e., independent and identically distributed (IID). On the other hand, researchers simply initialize the local models of various clients with consistent structures. These assumptions limit federated learning to adapt different information and heterogeneous models. Obviously, this view is inappropriate in user modeling tasks, in which there are a variety of user scenarios [25]. In practice, the clients store inconsistent data since users in clients have different habits. Therefore, it is necessary to find a superior federated learning process that better adapts to federated user modeling tasks for isolated scenarios with the inconsistent settings of clients.

Nevertheless, the particularity of user modeling with inconsistent clients leads to three challenges which arise from the bottom up at the level of distribution, data and model: (1) *Statistical heterogeneity*: Different from traditional scenarios where the data is assumed to be IID [3], personal records for user modeling are usually non-independently identically distributed (Non-IID), which results in statistical disparities and personalization across the clients [28]. For example, as shown in Figure 1, the preferences of users in client 1 are focused on the items belong to two different regions, while the users in client 2 prefer the items in a certain region. The methods as mentioned, which train local models for clients based on the consistent global model, inevitably eliminate the personalization of clients and reduce the ability to depict user characteristics [44]. Accordingly, it is necessary to integrate the personalized information of user models to adapt statistical heterogeneity; (2) *Privacy heterogeneity*: As [1, 12] suggested, different information have different levels of privacy. For example, as shown in Figure 1, the attributes information of items (e.g., labels and categories) in clients are relatively public, because they are summarized from the prior domain knowledge and consistent in public [29]. While the information such as representations of users in the user model, are strictly private, since they are generated from preference distributions and proprietary to users. On one hand, in the process of federated learning, rashly sharing representations will bring the risk of exposing privacy [19]. On the other hand, discarding the sensitive information to protect privacy will lead to a loss of information. Therefore, we should flexibly apply specialized federated learning settings to information with privacy heterogeneity so that we can balance the information to be protected or shared across user models; (3) *Model heterogeneity*: The mainstream federated learning methods expect to build a general global model to model all locals. In other words, the local model in client is the copy of the global model in server [25]. However, in practical user modeling applications, due to the different properties of the private data, the user model structures among different clients are often different [26]. As shown in Figure 1, the different amounts of items browsed by users lead to differences in the sizes of the item space so that the user models generated from local data naturally differ in structure. Therefore, the strategy to process heterogeneous user models in federated learning also requires a careful design.

To address the above challenges, we propose a novel Hierarchical Personalized Federated Learning (HPFL) framework for user modeling. HPFL is a client-server architecture as the general flow shown in Figure 1(b). Compared to traditional FL process, there is a two-stage task in client stage. In the first phase, client of HPFL defines the hierarchical information in data by privacy heterogeneity, i.e., public information and private information. Accordingly, we design a user model with hierarchical structure which contains both public component and private component. After local user model training process, client uploads the public component directly, while delivers the drafts of private component to safeguard the data privacy. In the second phase, we propose a fine-grained personalized update strategy to weighted fuse the corresponding components to new local model according to both local user model and global model in server. As for server stage, the server executes differentiated component aggregation strategy for components received from clients. It directly weighted aggregates the public components of the same attribute to obtain the global public components. Correspondingly, for the private component, since the original of the representations is saved in local, the server aggregates the universe of local drafts to generate the private components of global model without alignment operations on representations. With the fine-grained personalized update strategy, we take both the expansion of the user model knowledge at the global perspective and the inheritance of user model personalization at the local perspective into consideration, which accommodates statistical heterogeneity. Moreover, with differentiated component aggregation strategy, we safely aggregate heterogeneous user models by different components so that we address both privacy heterogeneity and model heterogeneity. Finally, we conduct extensive experiments in different scenarios, including student capacities modeling in intelligent education systems and user preferences modeling in recommender systems. The experimental results clearly demonstrate that HPFL outperforms the baselines in user modeling tasks in terms of accuracy performances, ranking effectiveness and modeling rationality. To the best of our knowledge, HPFL is the first framework for federated user modeling tasks, which is specifically designed to take into account both differentiated component aggregation and fine-grained personalized update strategy.

2 RELATED WORK

In this section, we briefly review some related works from two aspects, i.e., user modeling and federated learning.

2.1 User Modeling

User modeling is a fundamental task, which aims to analyze behavioral information to infer the unobservable characteristics, such as capability, preference, habit, tendency and so on [59]. To model the rich characteristics of users, user modeling has been widely used in various applications, for example, based on user capability fitting, researchers employ user modeling to model user vision level [8], lawyer expertise [45] and gamer competitiveness [57]; based on user preference mining, researchers apply user modeling to tasks, such as personalized search [46], restaurant recommendation [58], news recommendation [9], dynamic social network [54] and other broad recommendation tasks [31, 33, 47]. Recently, artificial neural network-based user modeling methods have received widespread attention. Researchers apply the methods into some important personalized user modeling tasks, such as cognitive diagnosis for fitting student cognitive abilities [51] and collaborative filtering recommendation for mining user interest preferences [18], in which each method establishes a set of user modeling process to mine unobservable information of users and builds the hidden relationship between users and items in the particular scenario.

However, most of the existing user modeling methods are centralized training processes, which introduce the risk of revealing the user data privacy, leading to obstacles in practical applications. Therefore, we raise the federated user modeling task, which aims to process user modeling for isolated and inconsistent clients via federated learning technique.

2.2 Federated Learning

Federated learning is a promising machine learning technique in recent years. Federated learning is first proposed to solve the problem of model updating in mobile terminals [39]. With process of training local models independently and aggregating the models centrally, FL ensures data isolation and privacy protection. As people pay more and more attention to privacy protection and regulations such as General Data Protection Regulation (GDPR) limit the collection and use of personal data, FL has received extensive attention. From a technical perspective, existing frameworks can be categorized into three types [55, 56], i.e., horizontal federated learning, vertical federated learning and federated transfer learning. Specifically, in horizontal federated learning, the data in the different clients shares the same feature space of items but users are different; while in vertical federated learning, the data shares the same user space but the feature spaces of items are not quite overlapped [6]; finally, federated transfer learning faces the scenarios where both the feature spaces and the user spaces are inconsistent [24, 34].

Since then, some frameworks for improving the process are proposed. FedSGD and FedAvg [39] train the local model in parallel. The server here simply generates a global model by the weighted average of the local model parameters according to data sizes. FedAtt [23] considers the different importances of local models and aggregates local models by applying a layer-wise soft attention mechanism between local models and the global model. FedProx [27] subjoins a proximal term to close the local model and the global model, which aims to avoid the excessive drift during optimization. However, current works are proposed based on the assumption of consistent clients and provide a uniform model for all the clients, which is out of operation in practical scenarios [38]. Moreover, the methods mentioned still bring the risk of privacy leakage, especially when the models submitted contain sensitive representation information, e.g., the user representations in user models. Unfortunately, the common privacy protection method, differential privacy federated learning [14, 40] faces the dilemma of that the confidentiality and accuracy are not entirely available simultaneously [5, 20, 48]. Therefore, there are still some difficulties for applying federated learning in practical applications.

3 PRELIMINARIES

In this section, we first provide a clear definition of the federated user modeling. Then we consider two scenarios specifications.

3.1 Problem Definition

Before the framework design, we formally provide the issue of federated user modeling. In our scenario, there are $|C|$ clients. In a specific client c, there are $|U_c|$ users and $|V_c|$ items, which can be represented as: $U_c = \{u_1, u_2, \cdots\}$ and $V_c = \{v_1, v_2, \cdots\}$. The attribute dimension of items is K. Moreover, the interactions between users and items generate $|R_c|$ interaction records. Each user u, item v and their interaction result g form a triplet (u, v, g). In the problem tackled in this paper, we aim to train $|C|$ local user models, i.e., $\{\Theta_1, \Theta_2, \cdots\}$ for each clients, where the c-th user model $\boldsymbol{\Theta_c}$ can model the potential characteristics of users in client c and predict the interaction results.

As mentioned earlier, in the federated user modeling scenarios, there is inconsistency between clients. To achieve the inclusion of a variety of client settings for federated user modeling, we define and divide the different information with privacy heterogeneity. As we know, in the real world, all the clients share some public knowledge information, such as attributes of items, which are relatively public, allowing it to be communicated and shared. In addition, clients also hold some strictly private information in personal data that needs to be protected, such as distributions of users and items. In order to reasonably utilize the information with different privacy intensity as much as possible and avoid the risk of privacy disclosure, we define hierarchical information as:

Definition 1. *Public information: it refers to the information that contains the prior domain knowledge so that it can be shared among clients. In this case, the public information is relatively private and incompetent to expose the sensitive user information.*

Definition 2. *Private information: it is the information which is proprietary for clients and represents the unique distributions of users and items among each client. Apparently, it is with strictly privacy and needs to be protected.*

Specially, each local user model $\boldsymbol{\Theta}$ contains two corresponding designed components, i.e., public components as $\boldsymbol{\Theta}_k$ for public information and private components as $\boldsymbol{\Theta}_r$ for private information. Please note that in practical scenarios, the user data in devices is proprietary so that it is difficult for data centers to conduct centralized training in this case, which results in isolated and inconsistent user modeling.

3.2 Scenario specifications

User modeling can be applied in many scenarios, such as education, e-commerce, catering and so on. In this work, we choose two representative issues in real user modeling scenarios. The first one is user capability modeling in areas such as education. This task is specialized as student performance prediction [32]. Correspondingly user u, item v and information of K attributes in our problem are denoted as: student, question and knowledge concepts in question, respectively. The result of interactive behavior g is the student's response to the question, and the target in this scenario is to model student's mastery of questions and predict the student performances. The other is user preference modeling in areas such as recommendation. This task is regarded as customer rating prediction [30]. Similarly user u, item v and K attributes here, that is, customer, product and product categories. The interactive behavior is the user evaluation of the product. We mine customers' interests and complete user rating prediction as the ultima objective.

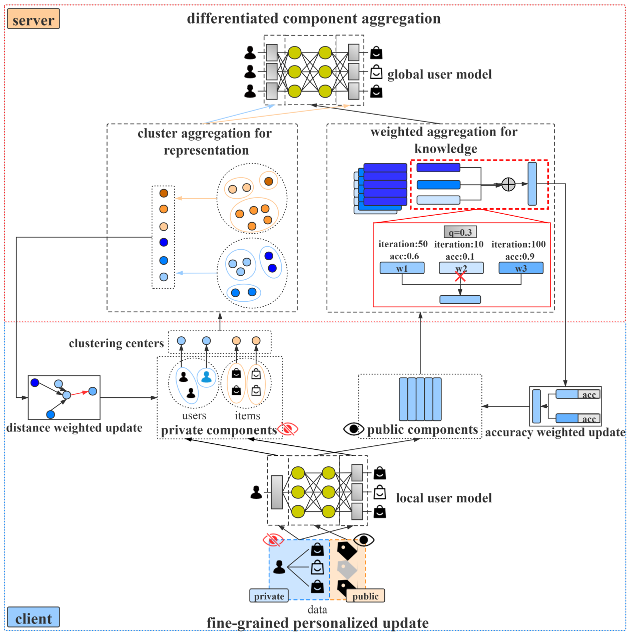


Figure 2: Hierarchical Personalized Federated Learning framework with differentiated component aggregation and fine-grained personalized update strategy.

4 HIERARCHICAL PERSONALIZED FEDERATED LEARNING

In this section, we describe our Hierarchical Personalized Federated Learning (HPFL) framework for user modeling in more details. Specifically, we first introduce an overview of our framework. All of the technical details are described in the following sections, including both client design with fine-grained personalized update strategy and server design with differentiated component aggregation strategy. Then we design a general user model as the local user model for hierarchical information, namely GUM. Finally, we present the whole workflow of HPFL.

4.1 Model Overview

To solve the problems mentioned, we propose a novel Hierarchical Personalized Federated Learning (HPFL) framework as illustrated in Figure 2, which accords to a client-server architecture. Client is the personal device, which is responsible for training a simple while proprietary user model with private records, that is GUM in our framework. Besides, it delivers the different components of user model and updates a personalized user model using the fine-grained personalized update strategy based on the global model received.

Algorithm 1 Fine-grained Personalized Update.

```
Input: The aggregated global public components, $\Theta_k^g$; The aggregated global private components, $\Theta_r^g$; The original local public components, $\Theta_k$ and private components, $\Theta_r$;
Output: The updated local public components, $\Theta_k$ and private components, $\Theta_r$;
1: Update($\Theta_k$,$\Theta_k^g$,$\Theta_r$,$\Theta_r^g$) :
2: compute new local $\Theta_k$ on attributes by Eq.(1)
3: compute new local $\Theta_r$ by Eq.(2)
4: return $\Theta_k$ and $\Theta_r$
```

The server is in charge of fusing heterogeneous local user models to a global one by different components with the differentiated component aggregation strategy. In a nutshell, the client maintains the personalization from Non-IID user data and the server allows to aggregate different components of the heterogeneous user models without compromising privacy. We will introduce the technical details of the two parts in the following subsections, respectively.

4.2 Client Design

The client in our framework is mainly responsible for two phases: one is to upload the trained local user model, and the other is to update the personalized user model based on the global model. Specially, the client first independently initializes and trains a general user model named GUM. The GUM contains both the public and private components, which are designed for hierarchical information (The details of GUM will be introduced in Section 4.4). The user model is trained with only local data and aims to model local user characteristics appropriately.

As for upload phases, client delivers the local user model by different components. In particular, the public component is delivered directly, since it is with public information. While the privacy component is sensitive, and the centralized use of the privacy component can lead to privacy leaks. Therefore, in our framework, client maintains the originals of the private components locally. Instead, it only provides some drafts, which are generated as the rough estimation for user or item representations. Specifically, as shown in Figure 2, client is required to process a clustering task on Θ_r to obtain the local cluster centers as drafts, which are representative for representations in user model, but low sensitive. Then the two components from clients will be aggregated in server, which will be introduced in Section 4.3.

As for update phases, after accepting the aggregated global model, the client is mainly responsible for updating the local GUMs from the global one to provide an appropriate user model for further applications. However, clients in federated user modeling have personalized information due to inconsistency generated from different application scenarios and operation styles. To retain personalized information and customize user models for clients, methods such as model interpolation [38, 41] are utilized in model update process. Unfortunately, since the black-box model interpolation lacks interpretability and may introduce the poor results [2], we regulate a fine-grained personalized update strategy to fuse the local personalized information and global generalized information to update GUMs by different components as Algorithm 1. The fusion process

Algorithm 2 Differentiated Component Aggregation.

```
Input: The set of public components from clients, $S_k$; The set of drafts of private components from clients, $S_r$; The set of local validation results from clients, $S_{acc}$;
Output: The aggregated global public components, $\Theta_k^g$; The aggregated global private components, $\Theta_r^g$;
1: initialize $p$.
2: Aggregation($S_k$,$S_r$,$S_{acc}$) :
3: compute $\Theta_k^g$ on attributes with $S_k$, $S_{acc}$ and $p$ by Eq.(3)
4: $\Theta_r^g \leftarrow$ Cluster($S_r$)
5: return $\Theta_k^g$ and $\Theta_r^g$
```

is in reference to a certain weight, which can reflect the importance of the local model. Noting that the comparison on weights is not our focus. Therefore, we choose local test accuracy as a intuitive dynamic weight in principle.

For public component in GUM, at round t, client i add the local attribute knowledge vector $\mathbf{c}_{k,i}^t$ and the global attribute knowledge $\mathbf{c}_k^{t,g}$ on attribute k via the corresponding accuracy $Acc_{k,i}^t$ to accuracy weighted update the new knowledge vector as:

$$\mathbf{c}_{k,i}^t = \mathbf{c}_{k,i}^t \times Acc_{k,i}^t + \mathbf{c}_k^{t,g} \times (1 - Acc_{k,i}^t). \tag{1}$$

Apparently, the better performances the local attribute knowledge has, the more likely it is to retain its own knowledge. But the poor knowledge vector will be out of the local optima after synthesizing the global information, so as to carry on the further optimization.

Correspondingly, for private component in GUM, the client i distance weighted update new representation with the affects of all global cluster centers, i.e., global private components, via the distances between the local one ($\mathbf{Emb}_{j,i}$) and global centers Θ_r^g. The update process can be denoted as:

$$\begin{aligned}\mathbf{Emb}^g &= \sum_{n=1}^{N} \frac{||\mathbf{Emb}_{j,i} - \Theta_{r,n}^g|| \times \Theta_{r,n}^g}{\sum_{m=1}^{N} ||\mathbf{Emb}_{j,i} - \Theta_{r,m}^g||},\\ \mathbf{Emb}_{j,i} &= \mathbf{Emb}_{j,i} \times \mathbf{Acc}_i + \mathbf{Emb}^g \times (1 - \mathbf{Acc}_i).\end{aligned} \tag{2}$$

In particular, $\mathbf{Acc}_i$ is an accuracy vector, representing the accuracy of the local model on each attribute. Similarly, the more accurate representations are less affected. While the more incompetent representation is updated to a general representation before the next training process on GUM.

4.3 Server Design

The server mainly processes the core task of aggregation of local user models to the global user model so that expands the available information for clients. However, the client inconsistency in federated user modeling causes privacy heterogeneity and model heterogeneity so that existing aggregation strategy which simply processes on a consistent entire model is inappropriate [1]. In order to flexibly implement the aggregation of inconsistent user models in federated learning, we propose the differentiated component aggregation strategy as Algorithm 2. With the strategy, we separately aggregate the different components in GUM, i.e., the public components and private components.

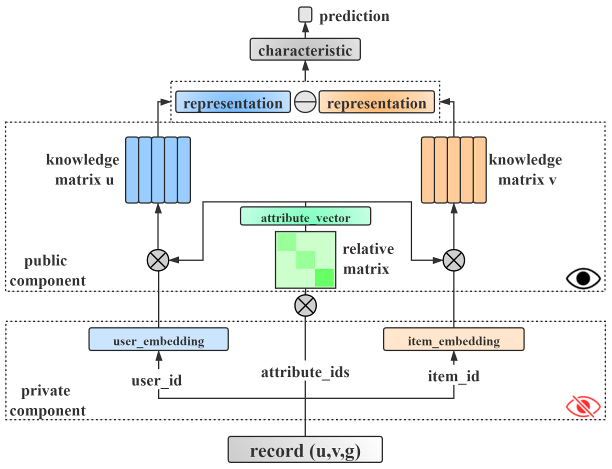


Figure 3: General user model, GUM

For the public component Θ_k in GUM, which is with relative openness and tolerated to be directly fused, the server processes a weighted aggregation of the same attribute to obtain the global public components in Figure 2. In particular, in round t, the local public components are delivered from clients. To obtain the global public components Θ_k^g, the server fuses each knowledge vector ($\mathbf{c}_{k,i}^t$) that represents knowledge information on attribute k from clients i in reference to both the number of iterations on attribute k ($I_{k,i}^t$) as well as the local validation accuracy ($Acc_{k,i}^t$). Finally, the global attribute knowledge, $\mathbf{c}_k^{t,g}$ is denoted as:

$$\mathbf{c}_k^{t,g} = \frac{\sum_{i=1}^{C} \delta(Acc_{k,i}^t, p) \times (I_{k,i}^t \times Acc_{k,i}^t) \times \mathbf{c}_{k,i}^t}{\sum_{i=1}^{C} \delta(Acc_{k,i}^t, p) \times (I_{k,i}^t \times Acc_{k,i}^t)}. \quad (3)$$

In particular, $\delta(x,p)$ is an indicator function, where $\delta(x,p)$=1, if $x > p$; otherwise, $\delta(x,p)$=0, while p is a dynamic threshold. That is, the server only aggregates the knowledge vectors whose local validation accuracy is higher than the threshold to dynamically select vectors in user models that are eligible to partake their knowledge.

For the private component Θ_r in GUM, it is strictly private. Since it is proprietary, unscrupulous alignment and fusion processes will reveal private preference information [19]. Therefore, server receives the cluster centers in clients which are regarded as the drafts of their representations from local user models. After that, server performs a further clustering with all the cluster centers, which is cluster aggregation as shown in Figure 2. The new cluster centers in server, which are defined as global private components Θ_r^g, represent the comprehensive representations of the bunches which involve similar users or items from different clients.

4.4 Local User Model Design

The functional structure in our proposed HPFL framework is the user model Θ, which is expected to model the users properly in various tasks. Indeed, a more complex model (e.g., NCD [51] and NCF [18]) is likely to be suitable for corresponding scenario, while it has weak generalization for other tasks. In addition, both of the above two models focus on low dimension for user and item representations on attributes so that lack the ability to model deep information [52]. As a result, in this section, we propose a General User Model (GUM) as shown in Figure 3, which is flexible, explainable and capable of deep representation.

Our proposed GUM aims to model the potential characteristics of users. It is mainly divided into two parts which are corresponding designed for the hierarchical information, i.e., public component and private component. As shown in Figure 3, we use the triplet record (u, v, g) as the input of GUM. Then we fetch the K-dimensional user embedding ($\mathbf{Emb_u}$) and item embedding ($\mathbf{Emb_v}$) via the user id and item id, respectively. The embeddings reflect the distributions of the inputs on the attributes. As mentioned before, the embeddings contain extreme private information, which can be use to infer local data distributions, so that they are defined as private components. In addition, we convert the attribute ids of the item v into multi-hot vectors, which is used to correspondingly strengthen the information of certain attributes.

Then, we fuse the distributions on the attributes to the multidimensional vector representation, which is treated as a summary of the distributions, via three mapping matrixes, i.e., knowledge matrix u (KM_u), relation matrix (RM) and knowledge matrix v (KM_v) in Figure 3. Where KM_u and KM_v are $K \times N$ matrixes which represent the N-dimensional knowledge vectors of K attributes at user and item perspective, respectively. While RM is a $K \times K$ matrix and indicates the knowledge relation among K attributes. With the multi-hot vector of the attribute information of the current item, we fetch the K-dimensional attribute vector ($\mathbf{Emb_c}$), which is on behalf of the relation of attributes. Obviously, the three matrixes contain the public knowledge information, which represents attributes with multidimensional vectors. Naturally, the knowledge in matrixes is relatively public and difficult to be used to detect private information. Therefore, we define these components of GUM as public components. Finally, we compute user representation ($\mathbf{R_u}$) and item representation ($\mathbf{R_v}$) as:

$$\mathbf{R_u} = \mathbf{Emb_u} \cdot \mathbf{Emb_c}^T \cdot KM_u. \quad (4)$$

$$\mathbf{R_v} = \mathbf{Emb_v} \cdot \mathbf{Emb_c}^T \cdot KM_v. \quad (5)$$

Finally, we simply define the distances between the user and the item as the user reflex on a item as:

$$\mathbf{P_{uv}} = \mathbf{R_u} - \mathbf{R_v}. \quad (6)$$

Noting that, for different tasks, we will use it to conduct different prediction tasks in user modeling as the objective. In particular, we use a simple $N \times 1$ hidden mapping matrix (HM) to map $\mathbf{P_{uv}}$ to a continuous value, representing the prediction as:

$$\mathbf{F_{uv}} = \mathbf{P_{uv}} \cdot HM. \quad (7)$$

In addition, to independently represent the hidden characteristic, e.g., capability or preference of user out of the items based on the multidimensional representations, we denote the hidden characteristic as:

$$\mathbf{h_u} = \mathbf{Emb_u} \cdot (KM_u \cdot HM). \quad (8)$$

With our proposed GUM, we establish both the private components and the public components for hierarchical information. The

Algorithm 3 The HPFL framework.

```
1: Server executes:
2: initialize $\Theta_k^0$.
3: for each round $t$ = 1 ,2,... do
4:     initialize $S_k = \{\}, S_r = \{\}, S_{acc} = \{\}$
5:     for each client index $c \in C$ in parallel do
6:         $\Theta_{k,c}^{t+1}, Lc_c^{t+1}, \mathbf{Acc}_c^{t+1} \leftarrow$ ClientUpdate($c$, $\Theta_k^{t,g}$,$\Theta_r^{t,g}$)
7:         $S_k = S_k \cup \Theta_{k,c}^{t+1}, S_r = S_r \cup Lc_c^{t+1}, S_{acc} = S_{acc} \cup \mathbf{Acc}_c^{t+1}$
8:     $\Theta_k^{g,t+1}$,$\Theta_r^{g,t+1}$ $\leftarrow$ Aggregation($S_k$,$S_r$,$S_{acc}$) according to Algorithm 2

1: ClientUpdate($c$,$\Theta_k^g$,$\Theta_r^g$):
2: $\Theta_k$,$\Theta_r$ $\leftarrow$ Update($\Theta_k$,$\Theta_k^g$,$\Theta_r$,$\Theta_r^g$) according to Algorithm 1
3: $\Theta_k, \Theta_r, \mathbf{Acc} \leftarrow$ LocalTraining($c$,$\Theta_k$,$\Theta_r$)
4: $Lc \leftarrow$ Cluster($\Theta_r$)
5: return $\Theta_k$, $Lc$ and Acc to server
```

private components are superficial private representations of users and items. The public components contain the multidimensional representation of knowledge information on attributes. Furthermore, we map the private representations to the multidimensional representation vectors, i.e. user representations and item representations, with mapping matrixes. Finally, we use a simple and general approach to define the user reflex for a certain item. For different tasks, we will flexibly apply user reflex vector to predict corresponding object.

4.5 HPFL Workflow

As mentioned above, the total workflow of HPFL is presented in Algorithm 3. For each global round, client first processes local training with only local data on GUM and delivers the different components of GUM to server, respectively. Hereafter, the server aggregates all local public components and private components by differentiated component aggregation strategy as Algorithm 2. Then the server distributes the global user model to all the clients. Finally, client receives the different components of global model and personalizes its own user model with the fine-grained personalized update strategy before next-round local training as Algorithm 1. It is worth noting that in our process, the user models are allowed to vary from client to client, so independent initialization is possible.

In summary, our proposed HPFL framework has the following advantages. HPFL achieves personalized model update through the fine-gained model fusion, avoiding the neglect of statistical heterogeneity. Furthermore, through the aggregation process, we specifically aggregate the model components with different privacy intensity to make use of as much information as possible on the premise of protecting the privacy information, which conforms privacy heterogeneity. At the same time, HPFL does not need to align the original representations in user models, so it is adaptive to the heterogeneous representation spaces among user models to suit model heterogeneity. Finally, our framework achieves good performances on multiple user modeling tasks, as covered in the next section.

Table 1: Statistics of the datasets: ASSIST and MovieLens.

Statistics	ASSIST	MovieLens
# of clients	59	10
# of records	327,058	96,538
# of users	3,477	925
# of items	17,561	1,679
# of attributes	122	19
# attributes per item	1.20	1.72
# attributes per record	1.20	2.21

5 EXPERIMENTS

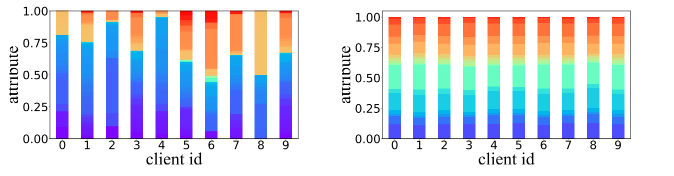
In this section, we first introduce our experimental datasets and framework setups. Then, we conduct experiments to demonstrate the superiority of our HPFL framework in user modeling tasks from the following three aspects: (1) the accuracy performances on user modeling tasks; (2) the ranking effectiveness on predictions; (3) the modeling rationality at the parameter level.

5.1 Experimental Datasets

We conduct our experiments on two typical user modeling tasks as referred in section 3.2. Correspondingly, we use two real-world public datasets to verify our HPFL. For educational task, we employ a public dataset, ASSIST (short for Assistments). ASSIST (2009-2010 "Non-skill builder"[1]) records the mathematics learning logs from an online tutoring program. It has been widely used for user capability modeling. Besides, for recommendation task, we adopt MovieLens (ml-100K [2]) including ratings for movies through the MovieLens web site, which is common for user preference modeling.

We first simulate a real federated learning scenario by dividing the datasets. In ASSIST, we divide the data into clients by teacher ids. Moreover, we filter out the clients whose students and average records are less than 5 to ensure that there is sufficient data for training. Finally, we get over 300K records of 3477 students in 59 clients. There are over 17K questions responded in a two-point scale, which belong to 122 concepts. Similarly, we divide the data into clients according to users' location via national area in zipcode (e.g., 1xxxx–Delaware, New York, Pennsylvania) in MovieLens and we delete the clients with less than 5 customers. We get our MovieLens dataset afterwards, which has over 96K records of 925 customers from 10 clients. Users rate on 1676 products (i.e., movies) belonging to 19 categories with a five-point scale. More statistics of the datasets are presented in Table 1. Specifically, in our datasets, the concepts and categories mentioned are public information as we defined in Sec 3.1. The interaction records of users and items, such as answers to questions of students and users' ratings of movies, contain sensitive information of users, which are the private information. In our scenario, each client only expects to hold its own data during training process, which causes data isolation. We aim to fuse and update local GUMs trained with isolated and inconsistent datasets in a percipient way, so that provide high quality user

[1]https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data/non_skill-builder-data-2009-2010
[2]https://grouplens.org/datasets/movielens/100k



(a) Distribution of attributes in ASSIST. (b) Distribution of attributes in MovieLens.

Figure 4: Distribution of attributes by clients of two datasets: ASSIST (left), MovieLens (right).

models for clients. Consequently, we solve federated user modeling problem for inconsistent local clients.

Then we analyze the data to compare the data distributions on the attributes of different clients. In particular, in order to compare the user modeling effects of HPFL under different distribution scenarios, we conduct experiments on both natural IID and Non-IID datasets. To be specific, we analyze the distribution of all the attributes of both datasets as Figure 4. For better illustration, we choose 10 clients with the largest data volumes. Different colors represent different attributes and the cover areas represent the frequency of occurrence of attributes in data. For example, we can compare the line of client 8 with other lines in Figure 4(a). We can observe that the distributions of some attributes (around orange color) appear more frequently in it than any other clients. It apparently shows that the frequency of occurrence of attributes is inconsistent, which results in Non-IID data across clients in ASSIST. While the distributions of attributes in MovieLens are almost consistent, which demonstrates that the data in MovieLens is closer to being IID. Therefore, ASSIST is a natural Non-IID dataset, whereas MovieLens is used as an approximate IID dataset in our experiments. Although the distribution of attribute in Movielens is IID, there are still inconsistencies in user space and item space that bring inconsistency to the structures of user models.

5.2 Experimental Settings

5.2.1 Data partition. In both datasets, i.e., ASSIST and MovieLens, we execute a random 80%/20% train/test partition of each user records. It is worth noting that we maintain the data in each client isolated in the experiments for federated user modeling methods.

5.2.2 HPFL Setting. We specify the framework setups in HPFL, including the GUM settings and HPFL settings. GUM is a general neural user model, in which we set N, the width of knowledge vector in KM_u and KM_v as 5. Moreover, in HPFL, for both global and local clustering tasks, we simply select the usual method–K-means [37] and the number of centers, K is 1/10 of the numbers of inputs. Finally, before training process, we initialize all parameters with xavier initialization following [15]. To facilitate further research in HPFL, we have published our code[3].

5.2.3 Baselines. First, we demonstrate the generalization and effectiveness of our proposed GUM by comparing two typical user modeling methods with centralized training process, i.e, **NCD**, **NCF**, which focus on cognitive diagnosis and collaborative filtering recommendation tasks, respectively. We verify separately in the special areas of these methods, i.e., NCD for education and NCF for recommendation.

- **NCD** [51] is a state-of-the-art cognitive diagnosis model, which models the complex cognitive relationships of shallow representations of both students and questions.
- **NCF** [18] is also a state-of-the-art collaborative filtering model based on deep neural networks, which models shallow features of both users and items.

Then, we compare some representative federated learning methods, which mainly process in inconsistent scenarios with a general settings. In our experiments, they are applied to both user modeling tasks based on GUM. Besides, we also train the user models in a distributed manner and conduct evaluations to verify the effectiveness of the federated settings in extending the available information, which is denoted as **Distributed**.

- **FedSGD** [39] is a standard federated learning method based on stochastic gradient descent, where the server takes a simple weighted average of all models to obtain a united global model and clients perform one epoch of gradient descent for per training process.
- **FedAvg** [39] also aggregates models to a united global model. However, FedAvg processes more computation steps in gradient descent to accelerate training and convergence.
- **Fednoise** is an extension method based on traditional federated process. It follows local differential privacy [7, 11, 53] which adds some random disturbances, like Laplace noise to local models before transmitted.
- **FedProx** [27] adds a reference loss in local training for each client, that is, the distances between the local model and the global model, so that it constrains the local personalized optimization process not to drift excessively.
- **FedAtt** [23] incorporates soft-attention in aggregation process. The server considers the importance of models and aggregates local models by layers with the distances between the global model and local models. Then it weighted aggregates local models to obtain the final global model.

In addition to verify the different federated process, we compare our complete **HPFL** on the user modeling tasks with two simplified frameworks: **HPFL-K** and **HPFL-R**, which only processes on public components or private components, respectively. For fairness, in our experiments, all methods mentioned are implemented by PyTorch, and trained on a Linux server with two NVIDIA Tesla K80 GPUs and 256G memory to achieve the best performance.

5.2.4 Evaluation metrics. To observe the accuracy performances of our proposed methods in user modeling tasks, we evaluate methods at both classification and regression perspectives. In particular, we use the widely-used ROC Curve (AUC) [13], Prediction Accuracy (ACC), Mean Absolute Error (MAE), and Root Mean Square Error (RMSE) metrics to measure the proximity between prediction and ground truth [22]. In particular, AUC and ACC evaluate the correctness of classification tasks in the range of [0, 1], the larger the values are, the better the results. Moreover, MAE and RMSE evaluate the similarity between prediction and ground truth in

[3]https://github.com/bigdata-ustc/hierarchical-personalized-federated-learning

Table 2: Accuracy performances of user modeling tasks for metrics on both datasets.

Methods	ASSIST			MovieLens		
	ACC	AUC	RMSE	ACC	MAE	RMSE
NCD	0.727	0.749	0.430	-	-	-
NCF	-	-	-	0.385	0.759	0.988
GUM	**0.736**	**0.774**	**0.421**	**0.397**	**0.745**	**0.946**
Distributed	0.699	0.718	0.442	0.389	0.802	1.001
FedSGD	0.704	0.716	0.453	0.341	0.933	1.111
FedAvg	0.703	0.724	0.445	0.397	0.802	0.993
Fednoise	0.701	0.722	0.441	0.387	0.804	1.019
FedProx	0.704	0.725	0.444	0.405	0.798	0.989
FedAtt	0.715	0.727	0.438	0.404	0.796	0.989
HPFL-K	0.715	0.730	0.437	0.403	0.792	0.987
HPFL-R	0.723	0.738	0.433	0.405	0.798	0.991
HPFL	**0.726**	**0.742**	**0.431**	**0.407**	**0.786**	**0.978**

regression tasks whose range is [0, 1], the lower the values are, the better the results.

In the practical user modeling tasks, we not only focus on the accuracy of prediction, but also the partial orders of user preferences for items [42]. Therefore, we adopt the commonly used ranking measurement indicators in user modeling tasks: Degree of Agreement (DOA) [21] and Normalized Discounted Cumulative Gain (NDCG) [17]. The indicators count whether the predicted ranking of the more preferred item is higher, which reflects the ranking effectiveness of the models.

5.3 Experimental Results

5.3.1 Accuracy performances. To evaluate the accuracy performances of all the above methods in isolated user modeling scenarios, we conduct the prediction tasks as mentioned before. Specially, for two typical user modeling tasks, i.e., cognitive diagnosis and collaborative filtering recommendation, we implement the target as student performance prediction and user rating prediction, respectively. We repeat the experiments 5 times and summarize the average of results. Table 2 reports the overall results on both datasets with evaluation metrics mentioned. Noting that, for student performance prediction of two-point scale, we usually focus on AUC and ACC. While for rating prediction, especially non-two-point scale rates, MAE is the more reasonable indicators.

Some key observations as follows: (1) Our proposed GUM model performs better than NCF and NCD on two datasets. It shows our general user model that is capable of deep representation for users and items is general and appropriate for user modeling tasks. (2) In all, federated methods perform better than distributed training processes. It shows federated learning settings that can harness more information from isolated clients, which usually results in better user models. Obviously, our proposed HPFL-based methods have the better performances than any other methods on both datasets. This means that our methods can more effectively accommodate user modeling tasks. (3) Obviously, the improvements in ASSIST from our methods are even more significant, because the data in ASSIST among clients is more inconsistent, that is with the Non-IID characteristic, while the data in MovieLens is basically IID. It indicates that our methods overperform on both datasets, but in data with more Non-IID characteristics, the advantages of our methods are more prominent. (4) HPFL performs the best performances on both tasks. While the performances of simplified methods, HPFL-K and HPFL-R are poorer than HPFL, because both of these methods lack some information of model components, i.e., lack of private component and public component, respectively.

5.3.2 Ranking effectiveness. As we argued earlier, not only the accuracy of prediction, but also the partial orders of user preferences are important in evaluation for user modeling. We adopt some common used indicators to evaluate the ranking effectiveness on both tasks. One is the Degree of Agreement (DOA) for extra ranking, which is used for measuring the consistency of preferences and predictions in group. That is, whether one prefers the same item than another user as user model reflects. Specifically, a DOA result on a specific attribute k is defined as:

$$DOA(k) = \sum_{a=1}^{|U_{c_1}|} \sum_{b=1}^{|U_{c_2}|} I_{abk} \frac{\delta(h_{ak}, h_{bk}) \cap \delta(\bar{g}_{ak}, \bar{g}_{bk})}{\delta(h_{ak}, h_{bk})}. \quad (9)$$

Here, U_{c_1} and U_{c_2} denote the users in clients c_1 and c_2, while h_{ak} indicates the hidden characteristic, e.g., capability or preference of user a on attribute k obtained by the our user models as Eq. 8, and $\bar{g}_{ak}$ is the average respond of user a on attribute k. $\delta(x, y)$ is an indicator function, where $\delta(x, y)$=1, if $x > y$; otherwise, $\delta(x, y)$=0. I_{abk} is another indicator function, where I_{abk}=1 if both user a and user b have interacted on the attribute k before. Furthermore, we average the DOA(k) of all attributes as DOA to measure the extra ranking effectiveness, which is denoted as $DOA = \sum_{k=1}^{K} DOA(k)/K, DOA \in [0.0, 1.0]$, the larger the DOA, the better the performance on the extra ranking.

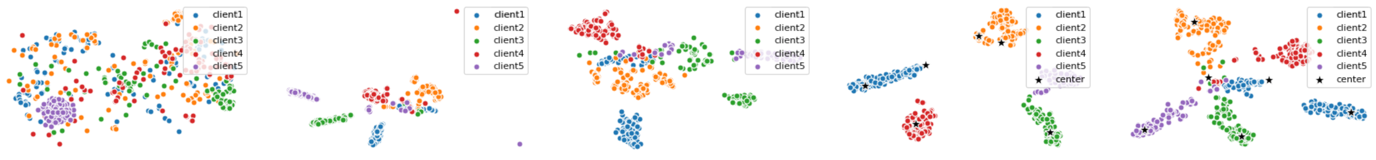
The other is the Normalized Discounted Cumulative Gain (NDCG) for inter ranking, which is used for measuring the consistency of real preferences and predictions for users. That is, whether one prefers an item than another item as the user model reflects. First, we define the DCG of a specific user u is formulated as:

Table 3: Ranking effectiveness of DOA and NDCG on ASSIST.

	NCD	GUM	Distributed	FedSGD	FedAvg	Fednoise	FedProx	FedAtt	HPFL-K	HPFL-R	HPFL
DOA	0.755	**0.773**	0.736	0.736	0.741	0.721	0.743	0.749	**0.756**	0.743	**0.758**
NDCG	0.826	**0.864**	0.837	0.825	0.833	0.831	0.835	0.834	0.834	**0.849**	**0.856**

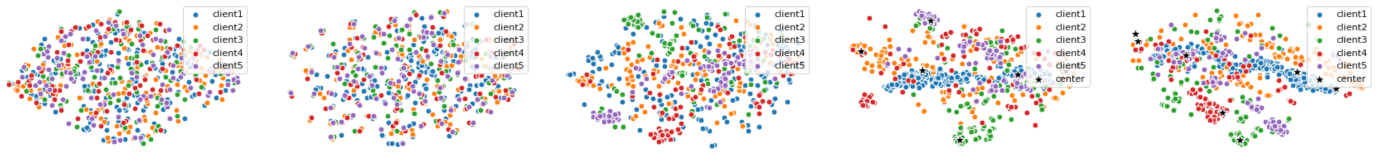
Table 4: Ranking effectiveness of DOA and NDCG on MovieLens.

	NCF	GUM	Distributed	FedSGD	FedAvg	Fednoise	FedProx	FedAtt	HPFL-K	HPFL-R	HPFL
DOA	0.505	**0.590**	0.537	0.519	0.668	0.539	0.668	0.669	**0.678**	0.672	**0.699**
NDCG	0.855	**0.869**	0.893	0.858	0.891	0.864	0.892	0.891	0.896	**0.898**	**0.910**



(a) Centralized (b) Distributed (c) HPFL-K (d) HPFL-R (e) HPFL

Figure 5: The user characteristics of different methods reduced dimension by t-SNE on five clients in ASSIST, where one point corresponds to one user.



(a) Centralized (b) Distributed (c) HPFL-K (d) HPFL-R (e) HPFL

Figure 6: The user characteristics of different methods reduced dimension by t-SNE on five clients in Movielens, where one point corresponds to one user.

$$DCG(u) = \sum_{k=1}^{K} \frac{h_{uk}}{\log_2 k + 1}. \tag{10}$$

Here, K denotes the total attributes and the K attributes are ordered by $\bar{g}_{uk}$ as the recall order. Then we define $NDCG(u) = DCG(u)/IDCG(u)$, where the $IDCG(u)$ is the ideal $DCG(u)$, that is apply $DCG(u)$ to the h_{uk} descending sorted. Furthermore, we average the NDCG(u) of all users as NDCG to measure the inter ranking effectiveness as $NDCG = \sum_{u=1}^{|U|} NDCG(u)/|U|, NDCG \in [0.0, 1.0]$. A larger NDCG means a better inter ranking performance.

Table 3 and Table 4 report the ranking effectiveness on DOA and NDCG. We can conclude the following from the results: (1) GUM performs better than other centralised methods, meaning that our high-dimensional user model adds more comparability for both inter and extra ranking. (2) HPFL performs outstanding results on two aspects on the whole, while HPFL-K gets great results on DOA and HPFL-R has advantages in NDCG. It results from that HPFL-K lacks private components so that it focuses on commonality between models, while HPFL-R ignores public components which add more collaboration between clients. (3) Compared with standard federated learning methods, distributed training method performs a comparable result in NDCG, while performs inferior results in DOA. It demonstrates that standard federated methods bring a coordination among clients so that it is benefit to extra ranking but causes weakness in inter ranking to some extent for user modeling.

5.3.3 *Modeling rationality.*

Furthermore, we deeply analyze the rationality of user models at the parameter level. We expect HPFL to facilitate the creation of more rational user models. As mentioned earlier in section 4.4, there are two components in local GUMs, i.e., public component and private component for hierarchical information. In order to compare the effects of hierarchical information, we deeply analyze methods by different components. In particular, we conduct the similarity analysis of public components and personalization analysis of private components to observe the similarities and differences between clients in federated learning.

Similarity analysis of public components. For the public component, we expect it to represent information collaboration between

Table 5: Similarity of different methods on both datasets.

Methods	ASSIST	MovieLens
Distributed	27.741	1.397
FedAvg	1.628	0.327
HPFL-K	1.945	0.031
HPFL-R	30.292	0.165
HPFL	4.023	0.066

clients. Therefore, we calculate the similarity of public components from clients of different methods. Specifically, we analyze the multi-client methods, such as distributed training process, the standard and clear Fedavg and our methods, then we calculate the cosine similarity of the corresponding public component between different clients. We define a total similarity as:

$$Simi = \sum_{i=1}^{C} \sum_{j=1}^{C} \frac{\sum_{k=1}^{K} cos(\mathbf{c}_{k,i}, \mathbf{c}_{k,j}))}{K}. \tag{11}$$

Where the $cos(x, y)$ is the cosine similarity function and it is applied to the pair-wised knowledge vectors from both clients. The lower value of Simi, the higher the similarity. For better comparison, we choose 10 clients with the largest data volumes on both datasets. Table 5 reports the results of similarity in models from both datasets. According to the results, we obtain the following conclusions: (1) On both datasets, public components across clients in the distributed training method are more different, since there is no federated process that clients communicate on public components. Besides, in our method, HPFL-K has the highest similarity, followed by HPFL. Obviously, aggregation of public components enhances similarity between user models in clients. (2) The similarity on ASSIST, which is Non-IID is much higher than those on MovieLens which is more IID, it shows that models trained on IID data are more likely to learn a similar distribution for parameters that represents the global distribution to some extent to obtain a better local user model, GUM. While models on Non-IID data should have some personalization, because in which case the consistent user models can lead to errors. Just as FedAvg has lower similarity, while performs worse on ASSIST as Table 2.

Personalization analysis of private components. For the private component, we expect to validate the ability of private components to capture personalized information. Specifically, we choose the conventional training methods, i.e., centralized and distributed training process for user modeling with our methods to analyze the rationality of embeddings in user models on clustering impressions. Specifically, we visualize the user characteristics from Eq. 8 after reducing their dimension by t-SNE [36]. For better illustration, we choose 5 clients with the most data. In particular, we annotate the cluster centers of users on figures of HPFL-R and HPFL.

Figure 5 and Figure 6 illustrate the user characteristics on both datasets. Through the visual representation of the figures, we come to the following conclusions: (1) On both datasets, private components in user models are not distinguishable in centralized training process, while distributed training process may enhance the gathering effect. (2) In MovieLens, the aggregation effect, even in the distributed training method, is not noticeable, since the IID distributions weaken personality of the clients. Under such severe case, our HPFL-R and HPFL method that process private components, still capture the personalized information, which shows that on both types of distributions, our methods have advantages to mine peculiarity of clients from user characteristics in user modeling. (3) Though our methods can capture personalized characteristics for users, we also notice the presence of mixed clusters from different clients in HPFL-K and HPFL, while clusters in HPFL-R are purer. It shows that public components share information and promote collaboration among clients while private component tends to capture the uniqueness of users for each client.

6 CONCLUSION

In this paper, we designed a novel federated user modeling framework, called Hierarchical Personalized Federated Learning (HPFL). It enables federated learning to be applied in user modeling tasks with inconsistent clients. Specifically, it is a client-server architecture. In client, we proposed a fine-grained personalized update strategy for personalized user model update, and a differentiated component aggregation strategy was explored in server to flexibly fuse heterogeneous user models. Our results on real-world user modeling tasks showed that HPFL outperforms existing federated learning methods, which demonstrated HPFL is more suitable in wide user modeling scenarios.

In the future, we will consider the data characteristics to improve the federated strategy for more elaborate framework design. We are also willing to design a platform and apply the technical details of HPFL to applications to solve practical problems in user modeling.

ACKNOWLEDGMENTS

This research was partially supported by grants from the National Key Research and Development Program of China (No. 2018YFC0832101), and the National Natural Science Foundation of China (No.s 61922073 and U20A20229). Qi Liu acknowledges the support of the Youth Innovation Promotion Association of CAS (No. 2014299) and the USTC-JD joint lab.

REFERENCES

[1] Hilal Asi, John Duchi, and Omid Javidbakht. 2019. Element Level Differential Privacy: The Right Granularity of Privacy. *arXiv preprint arXiv:1912.04042* (2019).
[2] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. 2019. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*. PMLR, 634–643.
[3] Léon Bottou. 2010. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. Springer, 177–186.
[4] Peter Carey. 2018. *Data protection: a practical guide to UK and EU law.* Oxford University Press, Inc.
[5] Kamalika Chaudhuri and Claire Monteleoni. 2009. Privacy-preserving logistic regression. In *Advances in neural information processing systems (NeurIPS)*. 289–296.
[6] Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, and Qiang Yang. 2019. Secureboost: A lossless federated learning framework. *arXiv preprint arXiv:1901.08755* (2019).
[7] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. 2017. Collecting telemetry data privately. In *Advances in Neural Information Processing Systems (NeurIPS)*. 3571–3580.
[8] Gitta O Domik and Bernd Gutkauf. 1994. User modeling for adaptive visualization systems. In *Proceedings Visualization'94*. IEEE, 217–223.
[9] Fabon Dzogang, Thomas Lansdall-Welfare, Saatviga Sudhahar, and Nello Cristianini. 2015. Scalable preference learning from data streams. In *Proceedings of the 24th International Conference on World Wide Web (WWW)*. 885–890.

[10] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *Proceedings of the 24th International Conference on World Wide Web (WWW)*. 278–288.

[11] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*. 1054–1067.

[12] Adrian Flanagan, Were Oyomno, Alexander Grigorievskiy, Kuan Eeik Tan, Suleiman A Khan, and Muhammad Ammad-Ud-Din. 2020. Federated Multi-view Matrix Factorization for Personalized Recommendations. *arXiv preprint arXiv:2004.04256* (2020).

[13] James Fogarty, Ryan S Baker, and Scott E Hudson. 2005. Case studies in the use of ROC curve analysis for sensor-based estimates in human computer interaction. In *Proceedings of Graphics Interface 2005*. 129–136.

[14] Robin C Geyer, Tassilo Klein, and Moin Nabi. 2017. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557* (2017).

[15] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 249–256.

[16] Filip Hanzely and Peter Richtárik. 2020. Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516* (2020).

[17] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. Trirank: Review-aware explainable recommendation by modelifng aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 1661–1670.

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

[19] Hossein Hosseini, Sungrack Yun, Hyunsin Park, Christos Louizos, Joseph Soriaga, and Max Welling. 2020. Federated Learning of User Authentication Models. *arXiv preprint arXiv:2007.04618* (2020).

[20] Xixi Huang, Ye Ding, Zoe L Jiang, Shuhan Qi, Xuan Wang, and Qing Liao. 2020. DP-FL: a novel differentially private federated learning framework for the unbalanced data. *World Wide Web* (2020), 1–17.

[21] Zhenya Huang, Qi Liu, Enhong Chen, Hongke Zhao, Mingyong Gao, Si Wei, Yu Su, and Guoping Hu. 2017. Question Difficulty Prediction for READING Problems in Standard Tests. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31.

[22] Zhenya Huang, Yu Yin, Enhong Chen, Hui Xiong, Yu Su, Guoping Hu, et al. 2019. Ekt: Exercise-aware knowledge tracing for student performance prediction. *IEEE Transactions on Knowledge and Data Engineering* (2019).

[23] Shaoxiong Ji, Shirui Pan, Guodong Long, Xue Li, Jing Jiang, and Zi Huang. 2019. Learning private neural language modeling with attentive aggregation. In *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[24] Qinghe Jing, Weiyan Wang, Junxue Zhang, Han Tian, and Kai Chen. 2019. Quantifying the performance of federated transfer learning. *ArXiv abs/1912.12795* (2019).

[25] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2019. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977* (2019).

[26] Daliang Li and Junpu Wang. 2019. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581* (2019).

[27] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2018. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127* (2018).

[28] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2019. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189* (2019).

[29] Boyi Liu, Lujia Wang, and Ming Liu. 2019. Lifelong federated reinforcement learning: a learning architecture for navigation in cloud robotic systems. *IEEE Robotics and Automation Letters* 4, 4 (2019), 4555–4562.

[30] Qi Liu, Enhong Chen, Hui Xiong, Chris HQ Ding, and Jian Chen. 2011. Enhancing collaborative filtering by user interest expansion via personalized ranking. *T SYST MAN* 42, 1 (2011), 218–233.

[31] Qi Liu, Yong Ge, Zhongmou Li, Enhong Chen, and Hui Xiong. 2011. Personalized travel package recommendation. In *2011 IEEE 11th International Conference on Data Mining*. IEEE, 407–416.

[32] Qi Liu, Runze Wu, Enhong Chen, Guandong Xu, Yu Su, Zhigang Chen, and Guoping Hu. 2018. Fuzzy cognitive diagnosis for modelling examinee performance. *ACM Transactions on Intelligent Systems and Technology (TIST)* 9, 4 (2018), 1–26.

[33] Qi Liu, Xianyu Zeng, Hengshu Zhu, Enhong Chen, Hui Xiong, Xing Xie, et al. 2015. Mining indecisiveness in customer behaviors. In *2015 IEEE International Conference on Data Mining*. IEEE, 281–290.

[34] Yang Liu, Yan Kang, Chaoping Xing, Tianjian Chen, and Qiang Yang. 2020. A secure federated transfer learning framework. *IEEE Intelligent Systems* 35, 4 (2020), 70–82.

[35] Yang Liu, Yingting Liu, Zhijie Liu, Yuxuan Liang, Chuishi Meng, Junbo Zhang, and Yu Zheng. 2020. Federated forest. *IEEE Transactions on Big Data* (2020).

[36] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[37] James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1. Oakland, CA, USA, 281–297.

[38] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. 2020. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619* (2020).

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

[40] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2017. Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963* (2017).

[41] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. 2019. Agnostic federated learning. *arXiv preprint arXiv:1902.00146* (2019).

[42] 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 *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1234–1242.

[43] Tao Qi, Fangzhao Wu, Chuhan Wu, Yongfeng Huang, and Xing Xie. 2020. Privacy-Preserving News Recommendation Model Training via Federated Learning. *arXiv preprint arXiv:2003.09592* (2020).

[44] Hanchi Ren, Jingjing Deng, and Xianghua Xie. 2020. Privacy Preserving Text Recognition with Gradient-Boosting for Federated Learning. *arXiv preprint arXiv:2007.07296* (2020).

[45] Leonardo Filipe Rodrigues Ribeiro and Daniel Ratton Figueiredo. 2017. Ranking lawyers using a social network induced by legal cases. *Journal of the Brazilian Computer Society* 23, 1 (2017), 6.

[46] Xuehua Shen, Bin Tan, and ChengXiang Zhai. 2005. Implicit user modeling for personalized search. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. 824–831.

[47] Peijie Sun, Le Wu, Kun Zhang, Yanjie Fu, Richang Hong, and Meng Wang. 2020. Dual Learning for Explainable Recommendation: Towards Unifying User Preference Prediction and Review Generation. In *Proceedings of The Web Conference 2020*. 837–847.

[48] Aleksei Triastcyn and Boi Faltings. 2019. Federated learning with bayesian differential privacy. In *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2587–2596.

[49] Jacob M Victor. 2013. The EU general data protection regulation: Toward a property regime for protecting data privacy. *Yale LJ* 123 (2013), 513.

[50] W Gregory Voss. 2016. European union data privacy law reform: General data protection regulation, privacy shield, and the right to delisting. *The Business Lawyer* 72, 1 (2016), 221–234.

[51] Fei Wang, Qi Liu, Enhong Chen, Zhenya Huang, Yuying Chen, Yu Yin, Zai Huang, and Shijin Wang. 2020. Neural Cognitive Diagnosis for Intelligent Education Systems. In *34nd AAAI Conference on Artificial Intelligence, AAAI 2020*. 6153–6161.

[52] Hao Wang, Tong Xu, Qi Liu, Defu Lian, Enhong Chen, Dongfang Du, Han Wu, and Wen Su. 2019. MCNE: An end-to-end framework for learning multiple conditional network representations of social network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1064–1072.

[53] Yansheng Wang, Yongxin Tong, and Dingyuan Shi. 2020. Federated latent Dirichlet allocation: A local differential privacy based framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 6283–6290.

[54] Peizhi Wu, Yi Tu, Zhenglu Yang, Adam Jatowt, and Masato Odagaki. 2018. Deep modeling of the evolution of user preferences and item attributes in dynamic social networks. In *Companion Proceedings of the The Web Conference 2018*. 115–116.

[55] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, 2 (2019), 1–19.

[56] Qiang Yang, Yang Liu, Yong Cheng, Yan Kang, Tianjian Chen, and Han Yu. 2019. Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 13, 3 (2019), 1–207.

[57] Georgios N Yannakakis and Julian Togelius. 2018. *Artificial intelligence and games*. Vol. 2. Springer.

[58] Fuzheng Zhang, Nicholas Jing Yuan, Kai Zheng, Defu Lian, Xing Xie, and Yong Rui. 2016. Exploiting dining preference for restaurant recommendation. In *Proceedings of the 25th International Conference on World Wide Web*. 725–735.

[59] Ingrid Zukerman and David W Albrecht. 2001. Predictive statistical models for user modeling. *User Modeling and User-Adapted Interaction* 11, 1-2 (2001), 5–18.