VHFed: A Two-Tier Vertical and Horizontal Federated Learning Framework for Enhanced Model Performance

Weiliang Ke[1], Zechen Liu[1], Shuyao Guo [3], Wei Song[1,2,4(✉)], and Yuan Feng[5]

[1] School of Computer Science, Wuhan University, Wuhan, China
{keweiliang,2017301500039,songwei}@whu.edu.cn
[2] Xinjiang Production and Construction Corps Key Laboratory of Computing Intelligence and Network Information Security, Shihezi University
[3] Economics and Management School, Wuhan University, Wuhan, China
2022301051276@whu.edu.cn
[4] Intellectual Computing Laboratory for Cultural Heritage, Wuhan University, China
[5] Wuhan Dameng Database Company Limited
fengyuan@dameng.com

Abstract. Federated Learning (FL) enables collaborative model training while preserving data privacy, making it a critical solution for sensitive data scenarios. However, Vertical Federated Learning (VFL) faces challenges such as inefficient data utilization, high communication costs, and limited personalization in heterogeneous data environments. This paper proposes VHFed, a two-tier hybrid federated learning framework combining horizontal and vertical FL to address these challenges. VHFed incorporates a global local neural network (GLNN) to support personalized learning while preserving participant privacy. Theoretical analysis and experiments on multiple datasets demonstrate that VHFed significantly improves accuracy, personalization, and privacy protection, showcasing its advantages in handling heterogeneous data and multi-party collaborations.

Keywords: vertical federated learning · personalized federated learning · heterogeneous data

1 Introduction

Federated learning (FL) is a distributed machine learning paradigm enabling multiple parties to collaboratively train models while preserving data privacy by keeping local data undisclosed. Amid increasing privacy concerns and regulatory requirements, FL has garnered significant attention in academia and industry. FL is broadly categorized into horizontal federated learning (HFL) and vertical federated learning (VFL), each tailored to specific data partitioning scenarios. However, VFL applications face critical challenges, especially in large-scale institutional collaborations.

In VFL, participants possess distinct feature spaces with overlapping sample IDs, leading to inefficiencies in data utilization and high communication costs.

These challenges are exacerbated in multi-party collaborations, such as those in financial institutions, where assumptions about small-scale participants and labeled data availability are often unrealistic. Such scenarios result in limited sample and feature overlap, suboptimal data utilization, and increased communication overhead.

Although federated transfer learning (FTL) methods [1,2] address low sample overlap, they introduce additional privacy and communication costs. Personalization in multi-party VFL systems remains underexplored, despite its importance in addressing heterogeneity in data distributions, sample sizes, and resource contributions. Unlike HFL, where personalization mechanisms are well-studied, the lack of such mechanisms in VFL further limits its practical applicability.

To tackle these issues, we propose **VHFed**, a two-tier federated learning framework that integrates horizontal and vertical FL to optimize data utilization and enhance model performance. VHFed incorporates a Global Local Neural Network (**GLNN**) model, enabling participant-specific personalization through a dual structure of global and local modules. The local module preserves participant-specific features, enhancing personalization while maintaining data privacy. VHFed is evaluated using accuracy and AUC, standard benchmarks that measure performance across diverse data distributions. The contributions of this work are as follows:

- We propose a novel two-tier federated learning framework, VHFed, that incorporates all client features and instances into model construction, combining the strengths of horizontal and vertical FL.
- We introduce the GLNN model within VHFed, which enhances adaptability to diverse scenarios while protecting client-specific features for personalization. Concrete analyses on convergence and privacy are also provided.
- Extensive experiments on multiple datasets validate the effectiveness of VHFed and demonstrate the significant role of GLNN in improving performance. Results highlight superior accuracy, robustness, and adaptability to non-IID data distributions through client-specific feature utilization.

2 Related Work

2.1 Vertical Federated Learning

Federated learning (FL) has gained significant traction in recent years. Hardy et al. [2] introduced the concept of Vertical Federated Learning (VFL), involving a third party and two participants. Yang et al. [3] later proposed a third-party-free VFL framework. Despite these advancements, VFL faces challenges, particularly in scenarios with passive parties lacking labeled data. Most existing studies focus on two-party setups, overlooking scenarios with multiple participants. Feng et al. [4] addressed this by proposing a multi-participant VFL framework that considers one active participant. However, as the number of users increases, the aligned samples decrease, exacerbating heterogeneity issues in VFL.

Currently, four primary approaches address these challenges [1]: self-supervised learning [5,6,7], semi-supervised learning [8], knowledge distillation [9], and transfer learning [4,10]. While these approaches aim to improve model performance by leveraging unlabeled data or additional guidance, better results can still be achieved with greater participant overlap. To address this, we propose a FL framework that enhances participant sample utilization and maintains compatibility with these methods.

2.2 Personalized Federated Learning

Personalized federated learning focuses on tackling data heterogeneity and distribution differences. A common approach is partial personalization, which personalizes specific model layers and excludes them from aggregation [11,12]. Another approach, the decoupling method, splits the weights of the same layer into private and global weights [13,14,15]. Additionally, transfer learning has been explored as a solution to enhance personalization in federated learning [16,17,18]. Methods like FedProx [19] address personalization by balancing global and local model parameters, limiting the distance between the server and local models. Additionally, methods like FedFTG [20] optimize aggregation using data-free knowledge distillation.

Beyond mainstream approaches, notable methods include pFedAtt [21] and CCVR [22]. In pFedAtt, instead of aggregating all local updates into a single global model, the server maintains multiple personalized cloud models, capturing common knowledge from the top-k similar clients. CCVR refines decision boundaries in deep networks after federated training.

Existing personalization studies often rely on computer vision datasets, which complicate variable control and personalization effect quantification. Our approach integrates the decision boundary refinement of CCVR with the advantages of partial personalized layers and decoupling, achieving personalization and privacy protection while supporting heterogeneous local modules.

3 Methodology

3.1 Problem Definition

We define a FL setting involving k clients $\{c_1, c_2, \ldots, c_k\}$ collaborating to train a model on their respective local datasets $D_i = \{x_i, y_i\}$, while maintaining data privacy. In this setting, each client c_i can be clustered into a group g_j based on their feature characteristics. Within each group, clients share the same global features f_g^j, while maintaining distinct local features f_l^i. This clustering facilitates more efficient federated learning by leveraging commonalities in global features within groups while allowing for differentiation through local features. c_i has a feature space $\mathcal{F}_{c_i}$, which includes three types of features: global features f_g^j, local features f_l^i, and potential features f_p^i. Here, f_g^j denotes features shared across all clients, f_l^i are features unique to client c_i.

Table 1: Notations

Notations	Meanings
$\mathcal{F}_{ci}$	feature space of the i-th client
c_i	i-th client
g_j	j-th group
f_g^j	global features of the j-th group
f_l^i, f_p^i	local, potential features of the i-th client
P_G, P_L^i, P_P^i	distribution of global,local and potential features
P_i	distribution of features in i-th client dataset
$D(P_1, P_2)$	distance between P_1 and P_2
x_i, x_i'	samples with global, local features only of i-th client
$\mathcal{M}_G^i$, $\mathcal{M}_L^i$, $\mathcal{M}_P^i$	global, local and potential feature model of GLNN of i-th client
o	the output of $\mathcal{M}_G^i$
$O_{i,j}$	the overlapping samples between i-th client and j-th client
C_i	the historical contribution of i-th client

Assumption 1 *For all clients within the same group g_j, they share an identical global feature space, represented by f_g^j.*

Assumption 2 *For two different groups g_i and g_j, the union of samples from all participants within each group $\cup_{c \in g_i} x_c$ and $\cup_{c \in g_j} x_c$ has a significant overlap.*

Definition 1 (Potential Feature) *In a heterogeneous data environment, f_p^i is defined as a latent attribute that, while not explicitly represented in the shared feature space across different parties, significantly impacts model predictions due to differences in data distributions or organization-specific contexts. These features arise from institution-specific business practices or local data characteristics and influence outcomes in ways not directly observable to other participants.*

The core objective of our research is to leverage as many samples and features as possible in a heterogeneous data environment while ensuring that federated learning remains personalized so that each participant obtains a model with high performance tailored to their specific data. For instance, two financial participants A and B may have different data distributions due to various factors.

We use P_i to denote the data distribution of A and B respectively. And $D(P_A, P_B)$ denotes the distance between P_A and P_B. Traditional federated learning models generally focus on a globally optimal result. For example, models like FedAvg learn the global distribution P_G from P_A and P_B. However, this type of model often fails to meet the personalized needs of clients. Some models, such as FedBN and GCAE, recognize that client data may be non-IID, leading them to make adjustments summarized by the following equations:

$$\arg \min_{\hat{\theta}, \{\theta_i\}_{i=0}^n} D\left(F\left(f_i(x_i;\theta_i);\hat{\theta}\right), P_i(x_i)\right) \tag{1}$$

or

$$\arg \min_{\hat{\theta}, \{\theta_i\}_{i=0}^n} D\left(f_i\left(F(x_i;\hat{\theta});\theta_i\right), P_i(x_i)\right) \tag{2}$$

where f_i is a local model, θ_i denotes local parameters, $\hat{\theta}$ denotes global parameters, and x_i denotes clients' training samples. According to [23], we assume $F(x_i; \hat{\theta}) \approx P_G$.

This kind of model achieves higher accuracy in local datasets but still cannot make full use of clients' features. In the proposed VHFed, we enable all features to participate in the model training process, but the challenge remains in how to effectively combine them.

In the whole system, there are four kinds of data distribution: the distribution of client's own features P_i, the distribution of all clients' global features P_G, the distribution of client's potential features P_P^i, and the distribution of a single client's local features P_L^i. Among them, the distribution of the client's own features is the target distribution we need to fit, and the other three distributions are the assistants we use to help fit the target distribution. If we find the relationship between the above four distributions, we can achieve VHFed.

Our task can be summarized as finding the function $F_i(\cdot)$ for c_i which can fit the P_i.

$$P_i \approx F_i\left(P_G, P_P^i, P_L^i\right) \tag{3}$$

In the VFL scenario, the feature space across clients is partitioned, resulting in each client c_i holding only a subset of the feature space. This can be represented as a joint feature space $x_i = [x_i^1, x_i^2, \ldots, x_i^k]$, where x_i^j denotes the subset of features held by the j-th client for the i-th sample.

Definition 2 (VHFed) *We call the model VHFed model while the model can fit the function $F(\cdot)$ and make Eq.(3) hold.*

3.2 Cross-Group Matching Scheme in Vertical Federated Learning

Firstly, the purpose of training in VFL is to combine the features held by participants to train a model with higher accuracy, and VFL requires the use of samples with consistent IDs among participants. Therefore, our grouping strategy should maximize the overlap of IDs and group participants with complementary features. As shown in Fig.2, after clustering based on $\mathcal{F}_c$, the parties in the k-th group have a similar feature space and the same global feature:

$$D(\mathcal{F}_{c_i}, \mathcal{F}_{c_j}) < \epsilon, \quad \forall c_i, c_j \in g_k \tag{4}$$

$$f_g^k = f_g^j, \quad \forall j \in g_k \tag{5}$$

where $D(\mathcal{F}_{c_i}, \mathcal{F}_{c_j})$ represents the distance between $\mathcal{F}_{c_i}$ and $\mathcal{F}_{c_j}$, and ϵ is a small threshold indicating that they are similar but not identical. In VFL, an increase in the number of users obviously leads to a decrease in sample overlap and a significant increase in communication costs. Additionally, in VFL, at least one party needs to have labeled data. To satisfy fairness, our strategy is to allow the active party to bid on the passive party in each round of VFL. The bidding rule is that both parties use the PSI algorithm based on the KKRT16 protocol[24] as shown in Algorithm 1 to calculate a trust score. The protocol uses Cuckoo

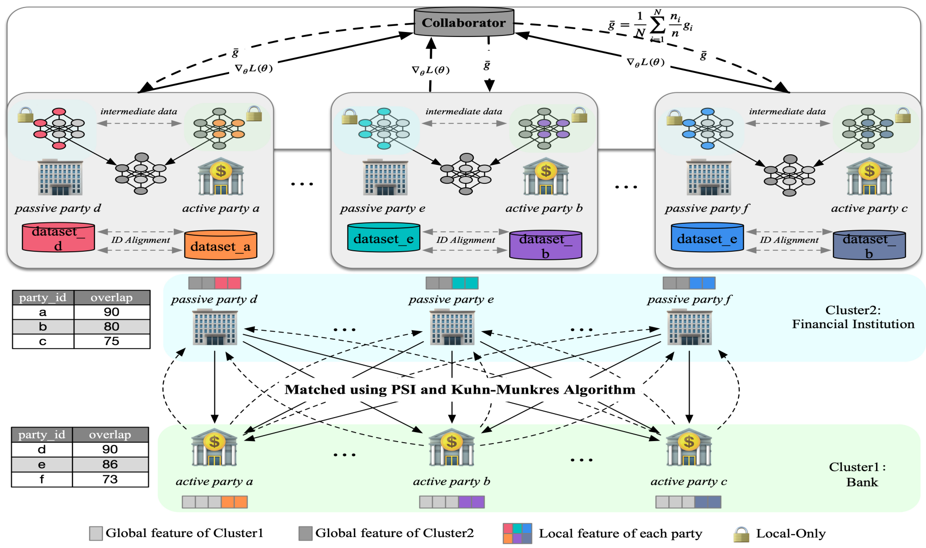


Fig. 1: Overview of VHFed. The diagram shows how the two clusters at the bottom layer are matched using PSI and the Kuhn-Munkres algorithm. After the matching, VFL is performed using shared f_g. The top layer illustrates how the GLNN leverages local, global and potential features to conduct HFL, ensuring comprehensive utilization of data for enhanced training performance.

active party a

ID	age	income	deposit_a	loan(label)	credit_card_limit
1	23	3000	3000	yes	5000
2	34	8000	7000	no	20000
3	27	9000	2000	yes	4000
4	33	7000	9000	no	10000

passive party d

ID	job	education	household_size
1	loan officer	master	1
2	teacher	doctorial	5
5	chef	bachelor	3
6	doctor	docrorial	2

active party b

ID	age	income	deposit_b	loan(label)	loan_overdue
2	34	8000	2000	no	4
3	27	9000	3000	yes	2
4	33	7000	5000	no	0
5	30	5000	8000	yes	3

passive party e

ID	job	education	experience_year
3	student	high school	0
4	lawyer	no formal	10
6	doctor	docrorial	20
8	scientist	docrorial	40

active party c

ID	age	income	deposit_c	loan(label)	investment_risk
3	27	9000	4000	yes	low
4	33	7000	3000	no	high
5	30	5000	1000	yes	medium
6	41	9000	8000	yes	low

passive party f

ID	job	education	academic_score
2	teacher	bachelor	5000
5	chef	high school	20000
7	actor	bachelor	4000
8	scientist	doctorial	10000

Global feature of each party
Local feature of each party

Fig. 2: An example of bottom-layer matching in the federated learning framework. The figure illustrates the data structure of two groups involved in the matching process: active parties a, b, c belong to a group and passive parties d, e, f form another. Each party possesses shared global features f_g (shown in gray) and unique private features f_p (highlighted in color).

hashing with 3 hash functions. Items are assigned to bins by checking if any of the bins corresponding to the hash values of the item are empty. If not, the item currently in one of those bins is evicted and replaced with a new item. If the process fails to terminate, the final evicted item is placed in a special bin called the stash. This protocol is very efficient and is applied in the psi of both parties.

After two parties completed the PSI operation, we use $t_{a,b}$ and $t_{b,a}$ to indicate the level of trust between a and b.

$$t_{a,b} = \frac{O_{a,b}}{\min(n_1, n_2)} - \tilde{D}(P_a, P_b) + C_b \tag{6}$$

$$\tilde{D}(P_a, P_b) = \frac{D(P_a, P_b) - \min(D)}{\max(D) - \min(D)} \tag{7}$$

where $O_{a,b}$ represents the number of overlapping samples, while $\tilde{D}(P_a, P_b)$ represents the distance between the distribution of a and b. To account for the potential impact of variables with large value ranges, as shown in Eq.(7), we utilized the min-max normalization method to even out their influence.

Algorithm 1: Trust Calculation Algorithm

Input: Active party **a** with IDs $\mathcal{X}$; passive party **b** with IDs $\mathcal{Y}$; maximum size $n = \max(|\mathcal{X}|, |\mathcal{Y}|)$; historical contributions H_a and H_b; stash size limit s.

Output: Trust level $t_{a,b}$ between **a** and **b**.

Step 1: Cuckoo Hashing. Partition $\mathcal{X}$ into $1.2n$ bins and a stash using Cuckoo hashing. For each $x \in \mathcal{X}$, calculate $z(x)$, where $z(x) = \perp$ if x is in the stash, and otherwise, $z(x)$ indicates the bin index. Fill unused spaces with virtual elements.

Step 2: Prepare Inputs for OPRF. Generate inputs r_i for $i \in [1, 1.2n + s]$:

$$r_i = \begin{cases} x \parallel z(x) & \text{if } x \text{ exists in the bin or stash} \\ \text{dummy value} & \text{otherwise} \end{cases}$$

Step 3: OPRF Execution.

- **a** sends r_i to **b** and receives $F(k_i, r_i)$.
- **b** computes hash sets H_i and S_i for bins and stash, then sends them to **a**.

Step 4: Match Verification. For each $x \in \mathcal{X}$:

- If $z(x) = \perp$ (stash) and $F(k_{1.2n+j}, x) \in S_j$: increment $O_{a,b}$.
- Else if $z(x) \neq \perp$ and $F(k_{h_{z(x)}}, x \parallel z(x)) \in H_{z(x)}$: increment $O_{a,b}$.

Step 5: Trust Calculation.

$$t_{a,b} = \frac{O_{a,b}}{\min(|\mathcal{X}|, |\mathcal{Y}|)} - \tilde{D}(P_a, P_b) + C_a$$

a computes and sends $t_{a,b}$ to **b**. **b** computes $t_{b,a}$ similarly.

After all the active and passive parties initialize their trust tables using Algorithm 1, they are sorted in descending order. Then, we transform the problem into an assignment problem and solve it.

Definition 3 (Assignment Problem in Matching) *After constructing the trust table for $c_a \in g_1, \quad \forall a$ with respect to $c_b \in g_2, \quad \forall b$ using Algorithm 1, we need to solve the assignment problem to determine which pairs of participants will collaborate for VFL. We can represent the problem as an assignment problem by creating a bipartite graph $G = (U, V, E)$, where U represents the* active *parties in g_1, V represents the* passive *parties in g_2, and E represents the set of edges between active parties and passive parties. Each edge $(u, v) \in E$ has a weight $t_{u,v}$ representing the trust $\{t_{u,v} | u \in g_1, v \in g_2\}$ between active party u and passive party v. The problem is to find a perfect matching M in G such that the total weight of the matching, $w(M) = \sum_{(u,v)\in M} w_{u,v}$, is maximized.*

$$\max_{M \subseteq E} \sum_{(u,v)\in M} t_{u,v} \tag{8}$$

We address this problem using an optimized Kuhn-Munkres algorithm, as detailed in Algorithm 2. To enhance efficiency, we introduce a pre-check to skip passive parties already assigned to higher-trust active parties, reducing the search space. Additionally, we streamline the augmenting path process by directly updating groupings, eliminating the need for a matching matrix as in the standard algorithm.

Algorithm 2: Optimized Kuhn-Munkres Algorithm for VFL Matching

Input: Active parties set U, passive parties set V, trustworthiness matrix T

Output: Optimal matching G

```
Initialize G as empty while there exists an unmatched active party i ∈ U do
    Initialize an alternating tree T with i, and label arrays U', V' as 0
    while no augmenting path is found do
        Select an unmarked passive party j ∈ V if j is unmatched then
            Add j to the augmenting path, update G, and break
        else
            Update labels U', V' to adjust trustworthiness for j Add j
            and its matched active party to the alternating tree
        If no augmenting path, increase labels by Δ, the minimum slack
        of unmatched nodes
```

return G

3.3 Global Local Neural Network (GLNN)

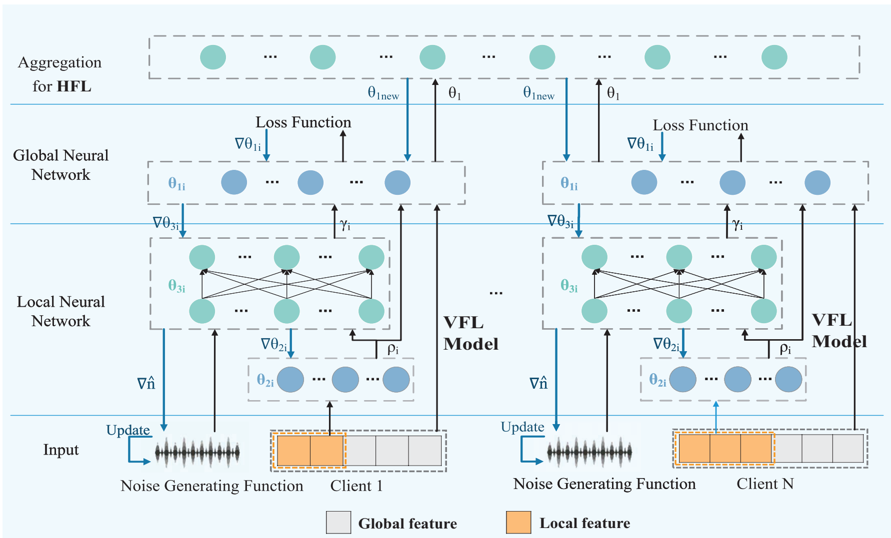


Fig. 3: An illustration of GLNN. Separating the network into local part and global part allows clients to input all their features into the model.

To build an effective personalized model using data from multiple owners while leveraging local private features f_l of each party, we propose the VHFed model, which includes four components: estimators for P_G, P_P^i, P_L^i, and a function $F(\cdot)$. We categorize these components into global and local modules. The global module participates in aggregation, while the local module does not. The global components estimate P_G, P_i, while the local components estimate P_P^i, P_L^i. The challenge lies in integrating the influence of local and potential features on the overall result. Our task is to estimate the parameters $\boldsymbol{\rho_i}, \boldsymbol{\gamma_i}, \theta_i$ such that:

$$P_i(x_i) \approx F(x_i; \boldsymbol{\rho_i}, \boldsymbol{\gamma_i}, \boldsymbol{\theta_1}) \tag{9}$$

$$\boldsymbol{\rho_i} \approx f_L^i(\boldsymbol{x_i'}; \boldsymbol{\theta_{2i}}) \tag{10}$$

$$\boldsymbol{\gamma_i} \approx f_p^i(\boldsymbol{\hat{n}}; \boldsymbol{\theta_{3i}}) \tag{11}$$

where $\boldsymbol{\hat{n}}$ denotes noise following a specific distribution, $\boldsymbol{\theta_1}$ are global parameters, and θ_{ji} are local parameters for client i. We view P_L^i, P_P^i as components of $P_G(x_i)$. By determining the appropriate $\boldsymbol{\rho_i}, \boldsymbol{\gamma_i}$, Eq.(9) is satisfied. Different $\boldsymbol{\rho_i}$ and $\boldsymbol{\gamma_i}$ for the same x_i can lead to varying outputs from $F(\cdot)$. The global neural network focuses on finding $\boldsymbol{\theta_1}$, while the local neural network determines $\boldsymbol{\theta_{2i}}, \boldsymbol{\theta_{3i}}$.

Local Neural Network The local neural network estimates P_L^i and P_P^i through two main components: $\mathcal{M}_L^i$ for local features and $\mathcal{M}_P^i$ for potential features.

For the local feature component $\mathcal{M}_L^i$, the input $\mathbf{x}_i'$ yields $\boldsymbol{\rho_i}$:

$$\mathcal{M}_L^i = \boldsymbol{\rho_i} = \sigma_L\left(\boldsymbol{\theta}_2^i \cdot \mathbf{x}_i'\right) \tag{12}$$

where $\sigma_L(\cdot)$ is the sigmoid activation function.

For the potential feature component $\mathcal{M}_P^i$, we consider two scenarios. In the first, the client knows the distribution $\mathcal{D}(\alpha)$ of potential features. For example, a young soldier may assume good health, whereas an older person may assume otherwise. In the second case, the distribution is unknown or specific data generation is needed. The model uses $\boldsymbol{\rho_i}$ and noise $\hat{\mathbf{n}}$ as inputs to generate $\boldsymbol{\gamma_i}$:

$$\mathcal{M}_P^i = \boldsymbol{\gamma_i} = \sigma_P^2\left(\ddot{\boldsymbol{\theta}}_{3i} \cdot \sigma_P^1\left(\boldsymbol{\theta}_{3i} \cdot [\rho_i, \hat{\mathbf{n}}]^T\right)\right) \tag{13}$$

After one round of training, we get $\nabla\hat{\mathbf{n}}$ by back propagation, and get $E(\nabla\hat{\mathbf{n}})$ by finding the mean of the gradient. If $|E(\nabla\hat{\mathbf{n}})| > T$, then the update algorithm is $\alpha = \alpha + \eta(\alpha' - \alpha)$, $\beta = \beta + \eta(\beta' - \beta)$ where η is learning rate, T is pre-set threshold. On the contrary, α and β remain unchanged.

Including $\boldsymbol{\rho_i}$ in $\boldsymbol{\gamma_i}$'s input enables feature distribution sharing across clients through the global network. Clients with known distributions can substitute $\mathcal{D}(\alpha)$ for better accuracy.

VFL Results Integration In this step, VFL leverages the shared global features among participants to collaboratively train and produce an initial model parameter set, denoted as w^{VFL}. This parameter set serves as the baseline for further training within the global neural network.

The VFL process among participants c_i within a group G_i can be expressed as:

$$\boldsymbol{w_i^{VFL}} = \mathrm{VFL}(\mathbf{x}_i^G, \mathbf{y}_i) \tag{14}$$

where $\mathbf{x}_i^G$ represents the global feature set, and $\mathbf{y}_i$ denotes the labels held by active participants.

The output $\boldsymbol{w_i^{VFL}}$ from each group is combined with the outputs of local neural networks $\mathcal{M}_L^i$ and $\mathcal{M}_P^i$ to form the input for the global neural network.

Global Neural Network The global neural network integrates the VFL outputs $\boldsymbol{w_i^{VFL}}$ and local outputs $\boldsymbol{\rho_i}$ and $\boldsymbol{\gamma_i}$ to estimate P_i and P_G. The global network's output o for participant c_i is defined as:

$$\mathcal{M}_G^i = o = \sigma_G\left(\boldsymbol{\theta}_1\left[\boldsymbol{w_i^{VFL}}, \boldsymbol{\rho_i}, \boldsymbol{\gamma_i}, \mathbf{x}_i\right]^\top\right) \tag{15}$$

where $\sigma_G(\cdot)$ is the activation function, and $\boldsymbol{\theta}_1$ are the parameters of the global model.

Complete GLNN To protect the clients' privacy, the parameters $\boldsymbol{\theta}_{2i}$ and $\boldsymbol{\theta}_{3i}$ will not participate in aggregation and the dimension of $\hat{\mathbf{n}}$ can vary from client to client. Because each client owns different features and different number of features, it is extremely costly for attacker to reconstruct or infer the sensitive data of client without knowing the number of features. The complete GLNN structure shown in Fig.3 includes:

$$o = \sigma_G\left(\boldsymbol{\theta}_1\left[\boldsymbol{w_i^{VFL}}, \boldsymbol{\rho_i}, \boldsymbol{\gamma_i}, \mathbf{x}_i\right]^T\right) \tag{16}$$

$$\gamma_i = \sigma_P^2\left(\ddot{\boldsymbol{\theta}}_{3i} \cdot \sigma_P^1\left(\dot{\boldsymbol{\theta}}_{3i} \cdot [\boldsymbol{\rho_i}, \hat{\boldsymbol{n}}]^T\right)\right) \tag{17}$$

$$\rho_i = \sigma_L\left(\boldsymbol{\theta}_{2i} \cdot \mathbf{x}_i'\right) \tag{18}$$

In each training round, the global model parameters $\boldsymbol{\theta}_1$ are updated using:

$$\boldsymbol{\theta}_1^{t+1} = \sum_{i=1}^{l} \frac{n_i}{n} \boldsymbol{\theta}_1^t + \eta \nabla \ell(o, \mathbf{y}) \tag{19}$$

where n_i is the number of overlapping samples in group G_i, n is the total number of samples, and η is the learning rate.

This integration of VFL outputs and local feature information ensures that the global neural network benefits from both global and local learning, enhancing model performance and personalization.

Multi-Layer GLNN To enhance fitting, GLNN can be extended to multiple layers:

$$\boldsymbol{o^j} = GLNN^j(x_i', \hat{\boldsymbol{n}}^j, x_i) \tag{20}$$

$$result = GLNN^n(x_i', \hat{\boldsymbol{n}}^n, \boldsymbol{o^{n-1}}) \tag{21}$$

Parameters are updated via SGD across all GLNN layers.

4 Experiments

4.1 VHFed without GLNN

To evaluate the impact of the GLNN module in the VHFed framework, we introduced a baseline variant, VHFed without GLNN, which excludes the GLNN component while retaining all other elements, such as the dual-layer federated learning architecture and matching strategy. This serves as a control to isolate the contribution of GLNN to model performance. Comparing VHFed with and without GLNN, we observe that the complete model exhibits superior accuracy and stability, particularly in handling data heterogeneity and label imbalance.

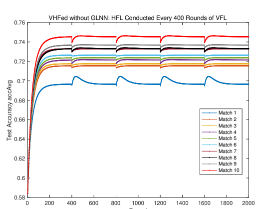
These results highlight the critical role of GLNN in enabling robust federated learning.

In this configuration, VHFed without GLNN directly uses the VFL output as the global neural network for HFL aggregation, omitting local neural networks. Consequently, participants cannot fine-tune the model with their local features, leading to identical models for clients within each match. This design allows us to clearly assess GLNN's role in facilitating personalization and improving performance by leveraging local features.

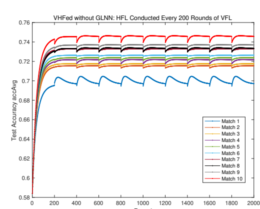
Setup: We divided 20 participants into two groups: g_a, consisting of active parties $\{c_1, c_2, \ldots, c_{10}\}$, and g_b, consisting of passive parties $\{c_{11}, c_{12}, \ldots, c_{20}\}$. Active parties hold labeled data, while passive parties hold unlabeled data. Matching between active and passive parties is performed using VFL, enabling each pair (c_i, c_j) to collaborate. Overlapping samples are used for training, while non-overlapping samples are reserved for testing. The overlap increases incrementally for matches 1 to 10, with an additional $50 \times i$ samples per match, where i is the match index. The Churn Modelling dataset [1], containing customer demographic and account data, was preprocessed to ensure clarity and consistency.

Results and Analysis: We evaluated VHFed performance under varying HFL aggregation frequencies and assessed the influence of GLNN on accuracy. Fig. 4(a) and Fig. 4(b) illustrate test accuracy across matches when HFL was conducted every 400 and 200 rounds of VFL, respectively. More frequent HFL updates (every 200 rounds) yielded faster convergence and higher accuracy, demonstrating that increased horizontal aggregation enhances inter-client learning and improves generalization on heterogeneous data.

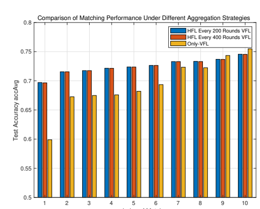
Fig. 4(c) highlights the effect of different aggregation strategies. Even without GLNN, VHFed achieves competitive results, but incorporating GLNN further enhances performance, especially in addressing client-specific data distribution challenges. Table 2 compares VHFed with and without GLNN under 200-round VFL aggregation, underscoring the advantages of VHFed's dual-layer architecture and grouping strategy in addressing data imbalance and heterogeneity.



(a) Weighted average accuracy in each match



(b) Weighted average accuracy in each match



(c) Impact of aggregation frequency on final accuracy

Fig. 4: Performance of matches based on different aggregation

[1] https://www.kaggle.com/shrutimechlearn/churn-modelling

Table 2: Comparison between VHFed(without GLNN) and VHFed

Match	match1		match2		match3		match4		match5	
VHFed(without GLNN)	0.697100		0.715484		0.717565		0.721653		0.723778	
VHFed	client1	client2	client1	client2	client1	client2	client1	client2	client1	client2
Accuracy	0.8056	0.7492	0.8513	0.7975	0.7752	0.8273	0.7689	0.8032	0.7850	0.7763
Match	match6		match7		match8		match9		match10	
VHFed(without GLNN)	0.726439		0.7330574		0.733371		0.736942		0.745695	
VHFed	client1	client2	client1	client2	client1	client2	client1	client2	client1	client2
Accuracy	0.8354	0.8431	0.7394	0.7512	0.8793	0.8617	0.8246	0.8112	0.7890	0.7748

4.2 VHFed with GLNN

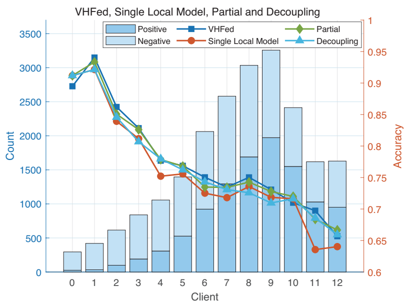
To demonstrate the advantages of our model in addressing non-IID and data heterogeneity issues, we selected a diabetes health indicators dataset from Kaggle [2] for testing. The non-IID nature of medical data in this dataset is prominent. We compare our model with the state-of-the-art federated learning models, such as FedBN, as well as other personalized federated learning models.

Table 3: The accuracies of models on each client. We can find that VHFed achieves highest weighted average accuracy in most clients. And high weighted average ROC-AUC indicates that VHFed is able to resist label distribution skew.

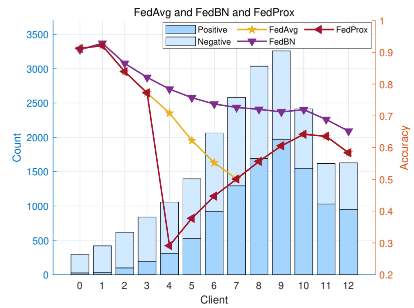
Client	Age	Total	FedProx	FedAvg	**GLNN**	FedBN	Decoupling	Partial	Single Local
client0	0-9	295	0.911864	**0.911864**	0.894915	0.908475	0.911864	**0.911864**	**0.911864**
client1	10-19	420	0.921429	0.921429	**0.940476**	0.928571	0.921429	0.933333	0.921429
client2	20-29	616	0.839286	0.839286	0.862013	**0.86526**	0.845779	0.850649	0.839286
client3	30-39	839	0.772348	0.772348	**0.828367**	0.821216	0.806913	0.825983	0.811681
client4	40-49	1057	0.291390	0.708609	0.776727	**0.784295**	0.779565	0.778619	0.752129
client5	50-59	1396	0.377507	0.622493	**0.7688625**	0.757163	0.762178	0.767908	0.755731
client6	60-69	2063	0.447407	0.552593	**0.750364**	0.737276	0.743093	0.734852	0.725158
client7	70-79	2582	0.501162	0.501162	**0.734702**	0.726569	0.730829	0.734314	0.718048
client8	80-89	3035	0.556837	0.556837	**0.750247**	0.720593	0.725865	0.742669	0.73575
client9	90-99	3258	0.605586	0.605586	**0.730816**	0.7124	0.709638	0.727747	0.718539
client10	100-109	2414	0.642088	0.642088	0.710025	0.719967	0.716239	**0.710025**	0.707692
client11	110-119	1619	0.635578	0.635578	**0.697344**	0.688697	0.685608	0.683755	0.635578
client12	120-129	1629	0.583794	0.583794	**0.656845**	0.651934	0.6593	0.666667	0.64027
accAvg	-	21223	0.568534	0.615653	**0.745936**	0.734345	0.72346	0.741695	0.72346
ROC-AUC	-	21223	0.701442	0.731308	**0.782874**	0.776014	0.76628	0.782604	0.691555

Setup: The dataset, originally designed for general machine learning tasks, lacked client grouping and feature-type variability. To simulate non-IID scenarios, we divided the data into 13 groups based on patients' age, as diabetes incidence strongly correlates with age. This resulted in significant heterogeneity across groups, including label distribution skew in some cases. Randomly selected features were designated as global features. Due to network limitations,

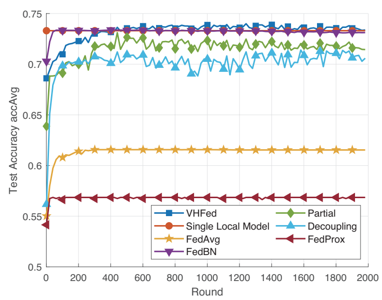
[2] https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset



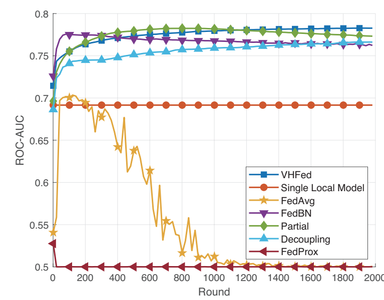
(a) Best performance of models



(b) Best performance of models



(c) Weighted average accuracy



(d) Weighted average ROC-AUC

Fig. 5: (a) and (b) represent the best performance of models for each client. (c) illustrates the test accuracy curves with aggregation rounds. (d) shows ROC-AUC curves with aggregation rounds on the same dataset.

FedAvg, FedProx, FedBN, and Decoupling models utilized only global features. To introduce feature diversity, we randomly removed specific features for each group. Multi-layer GLNN, Partial model, and Single Local Model utilized all client-specific features. The models were configured as follows: FedBN, FedProx, FedAvg, Decoupling model, and Single Local Model shared the same architecture with 120 input neurons, 120 hidden neurons, and a single output neuron. Partial model employed 120 input neurons, 16 hidden neurons, and a single output neuron. For Decoupling, 50% of each layer's weights were aggregated. Multi-layer GLNN utilized a local module with 2-dimensional random input (following $U(0,1)$), potential output, and local module output per layer. The global module had 24 neurons in the first layer, 12 in the second, and 1 in the third. All models except FedBN were optimized using Binary Cross Entropy loss and SGD (learning rate 0.01, batch size 30). FedBN used the entire dataset as a batch under similar settings.

Result and Analysis: In the experiments on the real dataset, we analyzed the weighted average ROC-AUC and accuracy, along with per-client ROC-AUC and accuracy. As shown in Fig. 5, our model consistently outperformed all other models in both metrics. The Single Local Model, affected by class imbalance, struggled to learn effectively for some clients, resulting in the lowest ROC-AUC. The Multi-layer GLNN mitigated the impact of class imbalance by aggregating local and global features, achieving superior ROC-AUC and demonstrating the effectiveness of the aggregation strategy. Models like FedBN, FedProx, Decoupling, and FedAvg performed suboptimally due to their reliance on limited features. The Partial model, while underperforming compared to GLNN, showed some mitigation of inter-client imbalance by reducing the dominance of clients with larger sample sizes over those with fewer samples, despite lacking well-organized aggregation due to hierarchical feature processing.

5 Conclusion

This paper presented **VHFed**, a two-tier FL framework combining horizontal and vertical FL to tackle challenges in heterogeneous data environments. By integrating the **GLNN**, VHFed enhances personalization while ensuring privacy. The dual-layer architecture improves feature utilization, addressing data heterogeneity and label skew. Theoretical analysis and experiments on multiple datasets confirmed VHFed's effectiveness, showing improvements in accuracy, AUC, and adaptability to non-IID data. The **GLNN** further enhances model performance by leveraging client-specific features, offering a practical solution for privacy-preserving collaborations in sectors like finance and healthcare.

Acknowledgements. This work is supported by the National Natural Science Foundation of China Nos. 62372340 and 62072349, the Major Technical Research Project of Hubei Province No. 2023BAA018, Key Research and Development Program Projects of the Corps Nos. 2024AA010, 2024AB080, Ali cooperation research and development project No. 24566207, Xinjiang Cyber Information Technology Innovation Research Project No. 12421607, and Xinjiang Production and Construction Corps Key Laboratory of Computing Intelligence and Network Information Security Open Fund No. CZ002702-02.

References

1. Liu, Y., Kang, Y., Zou, T., Pu, Y., He, Y., Ye, X., Ouyang, Y., Zhang, Y., Yang, Q.: Vertical Federated Learning. arXiv. https://arxiv.org/abs/2211.12814v2 (2022)
2. Hardy, S., Henecka, W., Ivey-Law, H., Nock, R., Patrini, G., Smith, G., Thorne, B.: Private Federated Learning on Vertically Partitioned Data via Entity Resolution and Additively Homomorphic Encryption. arXiv preprint. https://arxiv.org/abs/1711.10677 (2017)
3. Yang, S., Ren, B., Zhou, X., Liu, L.: Parallel Distributed Logistic Regression for Vertical Federated Learning without Third-party Coordinator. arXiv preprint. https://arxiv.org/abs/1911.09824 (2019)

4. Feng, S., Yu, H.: Multi-participant multi-class vertical federated learning. arXiv preprint. https://arxiv.org/abs/2001.11154 (2020)
5. Yang, Z., Zhou, M., Yu, H., Sinnott, R.O., Liu, H.: Efficient and Secure Federated Learning with Verifiable Weighted Average Aggregation. IEEE Trans. Netw. Sci. Eng. **10**(1), 205–222 (2023)
6. Tan, L., Zhang, X., Zhou, Y., Che, X., Hu, M., Chen, X., Wu, D.: AdaFed: Optimizing Participation-Aware Federated Learning with Adaptive Aggregation Weights. IEEE Trans. Netw. Sci. Eng. **9**(4), 2708–2720 (2022)
7. Wang, H., Yurochkin, M., Sun, Y., Papailiopoulos, D.S., Khazaeni, Y.: Federated Learning with Matched Averaging. *ICLR 2020*
8. Wang, Z., Fan, X., Qi, J., Wen, C., Wang, C., Yu, R.: Federated Learning with Fair Averaging. *IJCAI 2021*, pp. 1615–1623.
9. Wang, J., Zhang, L., Li, A., You, X., Cheng, H.: Efficient Participant Contribution Evaluation for Horizontal and Vertical Federated Learning. *ICDE 2022*.
10. Liu, Z., Chen, Y., Zhao, Y., Yu, H., Liu, Y., Bao, R., Jiang, J., Nie, Z., Xu, Q., Yang, Q.: Contribution-Aware Federated Learning for Smart Healthcare. *AAAI 2022*, pp. 12396–12404.
11. Pillutla, K., Malik, K., Mohamed, A., Rabbat, M., Sanjabi, M., Xiao, L.: Federated Learning with Partial Model Personalization. *ICML 2022*, pp. 17716–17758.
12. Wu, J., Liu, Q., Huang, Z., Ning, Y., Wang, H., Chen, E., Yi, J., Zhou, B.: Hierarchical Personalized Federated Learning for User Modeling. *WWW 2021*.
13. Chen, H.-Y., Chao, W.-L.: On Bridging Generic and Personalized Federated Learning for Image Classification. *ICLR 2022*.
14. Gao, L., Fu, H., Li, L., Chen, Y., Xu, M., Xu, C.-Z.: FedDC: Federated Learning with Non-IID Data via Local Drift Decoupling and Correction. *CVPR 2022*, pp. 10102–10111.
15. Shen, Y., Zhou, Y., Yu, L.: CD^2-pFed: Cyclic Distillation-guided Channel Decoupling for Model Personalization in Federated Learning. *CVPR 2022*.
16. Yi, L., Yu, H., Shi, Z., Wang, G., Liu, X., Cui, L., Li, X.: FedSSA: Semantic Similarity-based Aggregation for Efficient Model-Heterogeneous Personalized Federated Learning. *IJCAI 2024*, pp. 5371–5379.
17. Xie, L., Lin, M., Luan, T., Li, C., Fang, Y., Shen, Q., Wu, Z.: MH-pFLID: Model Heterogeneous Personalized Federated Learning via Injection and Distillation for Medical Data Analysis. *ICML 2024*.
18. Xu, H., Shen, D., Wang, M., Wang, B.: Adaptive Group Personalization for Federated Mutual Transfer Learning. *ICML 2024*.
19. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated Optimization in Heterogeneous Networks. *MLSys 2020*.
20. Zhang, L., Shen, L., Ding, L., Tao, D., Duan, L.-Y.: Fine-tuning Global Model via Data-Free Knowledge Distillation for Non-IID Federated Learning. *CVPR 2022*, pp. 10164–10173.
21. Ma, Z., Lu, Y., Li, W., Yi, J., Cui, S.: PFedAtt: Attention-based Personalized Federated Learning on Heterogeneous Clients. *ACML 2021*, pp. 1253–1268.
22. Luo, M., Chen, F., Hu, D., Zhang, Y., Liang, J., Feng, J.: No Fear of Heterogeneity: Classifier Calibration for Federated Learning with Non-IID Data. *NeurIPS 2021*, vol. 34, pp. 5972–5984.
23. McMahan, B., Moore, E., Ramage, D., Hampson, S., Agüera y Arcas, B.: Communication-Efficient Learning of Deep Networks from Decentralized Data. *AISTATS 2017*, vol. 54, pp. 1273–1282.
24. Kolesnikov, V., Kumaresan, R., Rosulek, M., Trieu, N.: Efficient Batched Oblivious PRF with Applications to Private Set Intersection. *CCS 2016*, pp. 818–829.