RESEARCH-ARTICLE

Toward Few-Label Vertical Federated Learning

LEI ZHANG, Sun Yat-Sen University, Guangzhou, Guangdong, China

LELE FU, Sun Yat-Sen University, Guangzhou, Guangdong, China

CHEN LIU, Shenzhen Transsion Holdings Co Ltd, Shenzhen, Guangdong, China

ZHAO YANG, Shenzhen Transsion Holdings Co Ltd, Shenzhen, Guangdong, China

JINGHUA YANG, Southwest Jiaotong University, Chengdu, Sichuan, China

ZIBIN ZHENG, Sun Yat-Sen University, Guangzhou, Guangdong, China

View all

Published: 19 June 2024
Online AM: 09 April 2024
Accepted: 31 March 2024
Revised: 24 January 2024
Received: 01 April 2023

Toward Few-Label Vertical Federated Learning

LEI ZHANG, Sun Yat-sen University, Guangzhou, China
LELE FU, School of Systems Science and Engineering, Sun Yat-sen University, Guangzhou, China
CHEN LIU, Shenzhen Transsion Holdings Co. Ltd., Shenzhen, China
ZHAO YANG, Shenzhen Transsion Holdings Co. Ltd., Shenzhen, China
JINGHUA YANG, Southwest Jiaotong University, Chengdu, China
ZIBIN ZHENG, Sun Yat-sen University, Guangzhou, China
CHUAN CHEN, School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China

Federated Learning (FL) provides a novel paradigm for privacy-preserving machine learning, enabling multiple clients to collaborate on model training without sharing private data. To handle multi-source heterogeneous data, Vertical Federated Learning (VFL) has been extensively investigated. However, in the context of VFL, the label information tends to be kept in one authoritative client and is very limited. This poses two challenges for model training in the VFL scenario. On the one hand, a small number of labels cannot guarantee to train a well VFL model with informative network parameters, resulting in unclear boundaries for classification decisions. On the other hand, the large amount of unlabeled data is dominant and should not be discounted, and it is worthwhile to focus on how to leverage them to improve representation modeling capabilities. To address the preceding two challenges, we first introduce supervised contrastive loss to enhance the intra-class aggregation and inter-class estrangement, which is to deeply explore label information and improve the effectiveness of downstream classification tasks. Then, for unlabeled data, we introduce a pseudo-label-guided consistency mechanism to induce the classification results coherent across clients, which allows the representations learned by local networks to absorb the knowledge from other clients, and alleviates the disagreement between different clients for classification tasks. We conduct sufficient experiments on four commonly used datasets, and the experimental results demonstrate that our method is superior to the state-of-the-art methods, especially in the low-label rate scenario, and the improvement becomes more significant.

CCS Concepts: • **Computing methodologies** → **Neural networks**; Classification and regression trees; • **Information systems** → *Data mining;*

Additional Key Words and Phrases: Vertical federated learning, semi-supervised learning, contrastive learning

The research was supported by the National Key Research and Development Program of China (2023YFB2703700), the National Natural Science Foundation of China (62176269), the Guangzhou Science and Technology Program (2023A04J0314), the Natural Science Foundation of Sichuan Province under grant 2024NSFSC7075, and the Postdoctoral Fellowship Program of CPSF under grant GZC20232198.

Authors' addresses: L. Zhang, School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, Guangdong, China; e-mail: zhanglei73@mail2.sysu.edu.cn; L. Fu, School of Systems Science and Engineering, Sun Yat-sen University, Guangzhou, Guangdong, China; e-mail: fulle@mail2. sysu.edu.cn; C. Liu and Z. Yang, Shenzhen Transsion Holdings Co. Ltd., Shenzhen, Guangdong, China; e-mails: chen.liu3@transsion.com, zhao.yang3@transsion.com; J. Yang, School of Information Science and Technology, Southwest Jiaotong University, Chengdu, Sichuan, China; e-mail: yangjinghua110@126.com; Z. Zheng, School of Software and Engineering, Sun Yat-sen University, Zhuhai, Guangdong, China; e-mail: zhzibin@mail.sysu.edu.cn; C. Chen (Corresponding author), School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, Guangdong, China; e-mail: chenchuan@mail.sysu.edu.cn.

ACM 1556-4681/2024/06-ART176
https://doi.org/10.1145/3656344

ACM Reference Format:
Lei Zhang, Lele Fu, Chen Liu, Zhao Yang, Jinghua Yang, Zibin Zheng, and Chuan Chen. 2024. Toward Few-Label Vertical Federated Learning. *ACM Trans. Knowl. Discov. Data.* 18, 7, Article 176 (June 2024), 21 pages. https://doi.org/10.1145/3656344

1 INTRODUCTION

With the gradual proliferation of smart devices, data tends to be generated in a decentralized manner, and each device stores only a portion of data. Due to increasing privacy concerns and data protection regulations [36], it is not possible to centrally collect data scattered on multifarious devices for model training. In this case, the effectiveness of traditional machine learning models can be greatly affected because of limited available data. Accordingly, how to utilize decentralized storage of data to improve the performance of client models has become an essential research topic.

To address the challenges, **Federated Learning (FL)** is proposed to jointly train multiple independent client models to improve respective performance. In the FL settings, each device first trains the model with local privacy data and uploads the generated gradient information to the server. Then, the server adopt a vanilla aggregation or an advanced processing strategy to handle these gradient information, and distributes the processed gradient to the participating clients. Each client uses the refined gradient information to update the local model. One of the earliest FL implementations is FedAvg [4], which averages the model parameters uploaded by the selected clients and uses the averaged parameters to update the local client. It achieves encouraging performance under the **independent identical distribution (IID)** of data. However, in practical scenarios, the data distribution of clients does not always satisfy the assumption of IID. When it comes to non-IID of data, the effectiveness of FedAvg drops dramatically. Much research has been conducted to address this challenge, and these approaches are mainly implemented by limiting local model updating. FedProx [24] introduces a proximal regularization term that constrains local model updates to deviate from the global model, thus mitigating the effects of heterogeneous data. MOON [23] utilizes the idea of contrastive learning to constrain the local model output representations, aiming at keeping the update direction of local model from deviating excessively from that of the global model. Besides, there are related studies that consider heterogeneity as a blessing; clustered FL [12] divides the clients with different distributions into different clusters, and the clients in the same cluster are trained using FedAvg. FedPer [7] considers that the model mainly consists of personalization layers and non-personalization layers, and only aggregates the non-personalization layers on the server side, whereas the personalization layers are kept local due to its client-side personalization knowledge. All of the preceding methods can solve the inconsistent distribution of client data to some extent. Today, a large amount of data is generated by different kinds of devices, which produce various data with diverse features. For example, a camera generates picture data, whereas a recording device can yield audio data. For such multi-source heterogeneous data, these methods based on horizontal FL cannot be applied, as each device requires different type of model.

To conjointly train machine learning models from multiple sources of heterogeneous data, **Vertical Federated Learning (VFL)** is proposed. VAFL [6] provides an asynchronous VFL framework and avoids privacy leakage by adding noise to the transmitted data. PyVertical [37] provides a VFL framework with SplitNN (split neural networks). FedOnce [49] uses an unsupervised training method to learn features and then train a predictive model with the learned features. MMVFL [10] takes advantage of the consistency of prediction results across client data to ensure the model training effect. AsySQN [60] provides a more resource-efficient algorithm for model training.

Although VFL can combine multiple sources of heterogeneous data for collaborative model training, to ensure consistent data labeling, VFL methods usually assume that the data labeling

information is held by the server or some authoritative organizations. Furthermore, since labeling data is always a time-consuming and labor-intensive task, data labels are often scarce. Under this situation, the previous VFL methods are applied to the classification task using only the cross-entropy loss to learn the decision margin. Nevertheless, it is not reasonable, because cross-entropy loss alone cannot extract enough valid information when the number of labels is very small. Besides, most VFL methods argue that the unlabeled data is useless and leave them untreated [6, 10, 59, 60] because unlabeled data cannot compute the objective function due to missing labels, inducing the inefficient utilization of data. In conclusion, the preceding two reasons together lead to the problem of unsatisfactory results of most VFL methods in few-label scenarios. In VFL settings, the features of row data should be extracted first, then the features are used for the downstream task. Generally speaking, the extracted features with a clear decision margin tend to yield premium prediction results, whereas the clear decision margin demands sufficient labeled data to support. Therefore, the few-label VFL scenario should be further studied to cope with the widespread emergence of few-label scenarios and the challenges they generate.

To solve the problem caused by insufficient labels in the vertical federated scenario, the following challenges need to be tackled:

— It is challenging to obtain valid information for making predictions from multiple types of data.
— Considering that few-label scenarios can have a large negative impact on classification results, unlabeled data should be exploited effectively.
— During the training process, the representations transmitted to the server are susceptible to attacks that can lead to privacy leakage, and therefore necessary measures need to be taken to prevent privacy leakage.

Therefore, in the few-label VFL scenario, we introduce the supervised contrastive loss with the labeled data, ensuring a close distance between features of the same category and a large distance between features of different categories. With the help of supervised contrastive, cluster-like properties among features can be enhanced, then the decision margin can be clearly depicted. Besides, considering that the data belonging to each client is generated from the same entity, there is a strong consistency between different client data— for instance, a picture of a dog is semantically consistent with a textual description of a dog. Actually, the consistency is widely used in various tasks of machine learning [5, 26, 40, 58]. However, too much consistency may have side effects for downstream tasks [39], which makes it necessary to properly consider the manner of consistency information to be extracted. Taking into account that our downstream task is a classification task, the consistency associated with labels becomes particularly important. Thus, we achieve the extraction of label-related consistency information by imposing constraints on the classification results of different client representations so that their classification results are as similar as possible. In this step, the unlabeled data is exploited, making full use of all data, not just labeled data. Finally, since the data features are extracted locally on the client side with a neural network, traditional optimization methods cannot directly optimize the parameters for privacy issues. Hence, we develop a new network parameter optimization method, which can optimize both client-side and server-side network parameters, and safeguard the client-side private data from leakage.

In summary, we propose a novel VFL method—*SSVFL*—in this article, which is capable of federating V clients with different features for model training, where V is the number of clients participating in the training. Furthermore, considering the sparse label information, the supervised contrastive term and the pseudo-label-guided consistency mechanism are simultaneously introduced to help learn well-predictive network, which endows the SSVFL with the capability of

achieving encouraging performance as well as convergence even in the situation of a low labeling rate. The contributions of this work are summarized as follows:

— We adopt supervised contrastive learning to strengthen intra-class cohesion and inter-class dispersion, thus helping the local model learn a clearer decision margin.
— We propose a pseudo-label-guided consistency loss, which is able to extract consistent information favorable for prediction against unlabeled data, improving the utilization efficiency of the whole dataset and contributing to the effectiveness of downstream tasks.
— We conduct extensive experiments on four commonly used datasets, and experimental results demonstrate that the proposed method outperforms the state-of-the-art methods. In particular, the improvement achieved by SSVFL over other methods is more significant in scenarios of few labeled data.

The rest of the article is organized as follows. The related work is described in Section 2. The details of the proposed SSVFL are presented in Section 3. To illustrate the superiority of the proposed SSVFL, the experiments on four datasets are conducted in Section 4. The article concludes in Section 5.

2 RELATED WORK

In this section, we review the existing research progress in FL, VFL, and few-label learning.

2.1 Federated Learning

FL is a new paradigm of machine learning that unites various participants for model training while protecting privacy, without each participant sharing local private data. The most typical implementation is FL is FedAvg [4]. In FedAvg, each participant trains a local model with the private local data, and this process is called *local updates*. After several rounds of local updates, the updated local model is uploaded to the server to perform global averaging. During the whole training process, private data is not shared. FedAvg is a simple but effective method and can achieve good performance under IID settings. However, under the non-IID settings, such as the distributions of the different clients being inconsistent, the performance of FedAvg will decrease significantly. According to the analysis of Li et al. [25], different data distributions will lead to different optimization directions. When performing global averaging, FedAvg will make the aggregated model deviate from the optimal solution, then lead to a decrease in the performance. Therefore, heterogeneity in FL is an urgent issue.

To mitigate the effects of heterogeneity, many works have been explored. These works focus on two perspectives: client-side training and server-side aggregation approaches. For the research on client-side training [20, 23, 24, 43], they argued that data heterogeneity makes the local update direction of the client vary widely and is not consistent with the global optimal update direction. FedProx [24] provided a new regular term, constraining the discrepancy between the local model parameters and the global model parameters. MOON [23] introduced contrastive learning into FL, and the global model is regarded as a *positive sample* and the previous global models as a *negative sample*. SCAFFOLD [20] introduced a client control variate to control the update of the local model. Wang et al. [43] proposed deep reinforcement learning for client selection, and with selected several related clients, the effects of heterogeneity can be mitigated. For the research on server-side aggregation [27, 44], novel aggregation methods were provided for aggregating the global model with generalizability. FedNova [44] aggregated a normalized gradient, and this mitigated the effects of heterogeneity but decreased the convergence rate. FedDF [27] provided a robust model fusion method with ensemble distillation.

The preceding methods perform well in light of heterogeneity scenarios, but in heavy heterogeneity scenarios, for example, where the client distribution is extremely different, the

effectiveness of all of these methods is severely degraded. Hence, a few studies [12, 55] considered heterogeneity as a blessing rather than a challenge. IFCA [12] clustered the received local models into several clusters and performed FedAvg within the same clusters. To further improve the clustering accuracy and training effect, G-FML [55] learned the representations of the entire dataset with auto-encoders to improve clustering accuracy and introduced meta-learning within the clusters to improve training results. Further, parameter decomposition [1, 30, 35] is a manner of solving heterogeneity—for example, FedPer [1] provided a *generic layer* + *personalization layer* model to achieve personalization, sharing the parameters of the *generic layer* and keeping the parameters of the *personalization layer* locally. Recently, FedOT [9] introduced optimal transportation [42] into FL systems, mapping the distributions of each client into a universal space, then training the model in the new space. Heterogeneity is not only reflected in data but also may be reflected in devices. ODE [13] explored the impact of on-device storage on the performance of FL, and defined a new data valuation metric for data evaluation and selection in FL with theoretical guarantees. SHIELD [57] proposed a novel incentivized FL with **differential privacy (DP)** in an MCS system to motivate clients to actively participate in training. Besides, to prevent privacy leakage, DP techniques are used as well. A_{FL} [34] designed an effective incentive mechanism to incentivize the participation of heterogeneous clients.

The introduced methods could solve the problem of heterogeneity with the assumption that the data feature space of each client is consistent. However, when each client only stores partial features of the data entity, the models trained by each client are different. Therefore, these methods cannot handle multi-source heterogeneous data.

2.2 Vertical Federated Learning

VFL [28] enables model training by associating clients with heterogeneous features. The first VFL method was provided by Hardy et al. [15]; it trained a logistic regression model with encrypted data divided vertically into two parts. For adapting VFL to multi-client scenarios, MMVFL [10] introduced a privacy label-sharing mechanism that enabled individual clients to use the received representations as well as private labels for model training. Pivot [48] provided a method for training tree-based models in VFL scenarios and resisting attacks from semi-honest clients. To improve the efficiency of VFL training, VAFL [6] proposed an asynchronous vertical federated training approach implementing a tradeoff between training accuracy and efficiency. PyVertical [37] introduced SplitNN and provided a universal network training framework. In VFL settings, model training consumes a large amount of bandwidth, since the uploaded representations and the received gradients may be large. To address the communication overhead, AsySQN [60] extended the gradient descent step by approximating the computation of Hessian information and speeded up the convergence rate, decreasing the communication rounds. However, AsySQN assumes that each client holds labels, which is inconsistent with the assumption made in this work.

The most current methods assume that the majority of data is labeled. However, in the few-label scenarios, these methods usually suffer significantly. FedMVT [19] gave a solution for handling missing labels but can only handle the case of two clients. Therefore, we propose an optimization algorithm in the few-label scenario, which is able to combine multiple clients for training and improve the model effect of VFL in the few-label scenario.

2.3 Few-Label Learning

Traditional machine learning also suffers from missing label scenarios, and there are numerous works proposing solutions. For instance, semi-supervised learning [41] exploits additional data points with unknown labels to augment model training. A typical semi-supervised method is *co-training* [3], which first inferred the pseudo-label and classification confidence of unlabeled data

and then added the unlabeled data and pseudo-label that exceeded the threshold to the training set so that the classifier can be trained jointly with the labeled and unlabeled data. To improve the performance, active learning [33] is introduced into the co-training framework. For instance, active learning was introduced to minimize the labeling cost by selecting the high-value data that can best improve model performance [11].

Besides, the artificially constructed information can be introduced into the training process to enhance performance, such as contrastive learning [14, 16, 22, 50, 56, 61]. CoMatch [22] jointly learned embeddings and class probabilities, and regularized the embeddings with graph-based contrastive learning. GCL [16] provided a framework unifying supervised metric learning and unsupervised contrastive learning. SsCL [61] combined contrastive loss and cross-entropy loss, and optimized these two objective functions in an end-to-end manner. ICL-SSL [56] designed a novel contrastive loss to guide the training of the network and eventually improved the discriminative capability.

Learning the core information of the data is very important for few-label learning, and several studies focus on studying how to extract features that contain information about the essential nature of the data, and constrained the extracted features to benefit downstream tasks [51–54]. TTLRR [52] attempted to find the latent low-rank tensor structure from the corrupted data with singular value decomposition for recovering the clean data. Yang et al. [53] took into account scenarios where noise is present in the data, and introduced tensor dictionary learning to remove both structure noise and Gaussian noise.

These methods require centralized processing of data from individual clients, which raises privacy issues and is therefore not suitable for application in VFL settings.

3 PROPOSED METHOD: SSVFL

In this section, we first define the few-label problem in the VFL scenario and then introduce the framework. Specifically, to explore the label information more deeply and to make the classification decision margin clearer, we propose supervised contrastive loss. Besides, we introduce a pseudo-label-guided consistency loss, which improves the utilization of the whole dataset.

3.1 Problem Formulation

Consider a set of V clients: $\mathcal{V} := \{1, \ldots, V\}$, where V is the number of clients participating in training. A dataset of N samples with V different types of features, the whole dataset D, defined by Equation (1), are maintained by V local clients with M labeled data and $N - M$ unlabeled data. Each client v is also associated with a specific type of feature. For example, client v maintains feature $x_n^{(v)} \in \mathbb{R}^{d_v}$, where $n = 1, \ldots, N$. Since the label information is usually maintained by an authority, it is reasonable to suppose that the label information y_n is held by the server.

$$D := \{\{x_n, y_n\}_{n=1}^{M}, \{x_n\}_{n=M+1}^{N}\} \tag{1}$$

To preserve privacy, $x_n^{(v)} \in \mathbb{R}^{d_v}$ are not permitted to be shared with other clients and the server. However, with the help of a neural network, $x_n^{(v)}$ could be mapped into a low-dimensional vector $h_n^{(v)}$, and sharing $h_n^{(v)}$ does not raise the privacy issue. Therefore, the objective function can be given as

$$F(\theta, w) = \frac{1}{N} \sum_{n=1}^{N} L(\theta, h_n^{(1)}, \ldots, h_n^{(V)}; y_n), \quad s.t.\ h_n^{(v)} = f(x_n^{(v)}; w_v), v = 1, \ldots, V, \tag{2}$$

where θ represents the parameters of the global model learned by the server, $w = [w_1, \ldots, w_V]$ denotes the set of network parameters of local models, L is the loss function, and f is the feature extraction function.

In the actual situation, not all samples have label information, and L is not computable for the unlabeled data. Therefore, a separate design of unlabeled loss L_{semi} is necessary to ensure that unlabeled data contribute to the model. Finally, the objective function should be modified as

$$\begin{aligned} F(\theta, w) &= \frac{1}{|I_l|} \sum_{n \in I_l} L_{sup}(\theta, h_n^{(1)}, \ldots, h_n^{(V)}; y_n) + \lambda * L_{semi}(I_u), \\ s.t.\ h_n^{(v)} &= f(x_n^{(v)}; w_v), v = 1, \ldots, V, \end{aligned} \tag{3}$$

where I_l are the indexes of the samples with label information, I_u are the indexes of the samples without the label, and L_{sup} is the supervised loss like L in Equation (2). Usually, for classification tasks, L_{sup} is given as the cross-entropy loss. Hence, designing a suitable L_{semi} for the unlabeled samples is the key to solve the semi-supervised problem. Previous VFL methods do not focus on the samples without a label and have a limited performance. Therefore, we propose SSVFL and design a novel form of L_{semi}, which boost the classification performance in the few-label scenario. Besides, we improve the supervised loss L_{sup}, allowing the model to explore the label information more deeply.

3.2 Details of SSVFL

In this section, we present an overview of the proposed SSVFL, including the framework and network architecture. Additionally, we introduce the two predominant modules: the supervised contrastive learning module and the pseudo-label-guided consistency information learning module.

3.2.1 Overview of SSVFL. Without loss of generality, we suppose that there exists V clients in the VFL system, and the dataset owned by the v-th client is $X^{(v)} \in \mathbb{R}^{N \times d^{(v)}}$, with a total of N samples. In this dataset, each sample is represented as a vector of $d^{(v)}$ dimensions. The network structure consists of three parts: the client-specific feature encoder, the global classifier, and the client-specific classifier. The overall framework of the proposed method SSVFL is illustrated in Figure 1. Next, each part of the network structure is described in detail:

(1) *Client-specific feature encoder:* Since the feature dimension of each client is different, we set an encoder consisting of a cascaded linear layer for each client to extract compact features, respectively. For the v-th client, the encoder is noted as f_v with parameters of w_v. Consequently, the extracted feature is given as $H^{(v)} = f_v(X^{(v)}; w_v)$, and the dimension of $H^{(v)}$ is set to d for each client-specific encoder.
(2) *Global classifier:* To finish the classification task, a global classification network is needed to make a prediction for classification. Considering that the global classifier should consider information from each client, the input of the global classifier should be the aggregation of the uploaded representations from each client, and the parameter is noted as θ.
(3) *Client-specific classifier:* To ensure that the learned representations are label consistent, the pseudo-label of the representations should be calculated. Therefore, a classifier is set up on the server side for each client, whose input is the uploaded representations from each client, and the parameter for the v-th client is noted as $\theta^{(v)}$.

For the labeled data, we propose supervised contrastive learning to enhance the discriminative nature of the representations, which in turn makes the classification decision margin clearer. This is introduced in Section 3.2.2. For the unlabeled data, we propose pseudo-label-guided consistency learning to enforce the client-specific feature encoders extract information benefits to the classification tasks, which is introduced in Section 3.2.3.

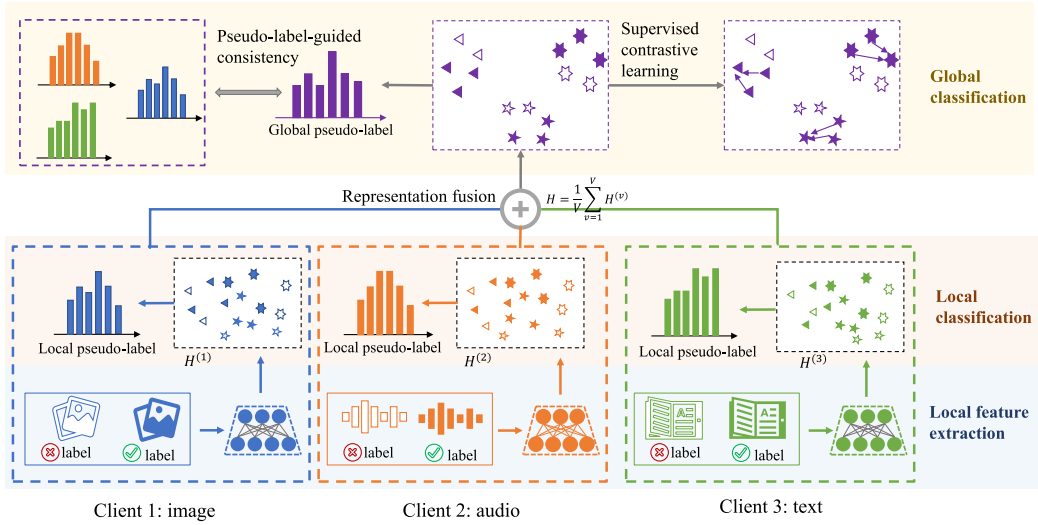


Fig. 1. The overall framework of SSVFL, wherein each client uses the local client-specific encoder to extract the representations $H^{(v)}$ of both labeled and unlabeled data, and then sends $H^{(v)}$ to the server. The server aggregates the received representations for finishing the classification tasks. For improving the classification performance and to make the classification decision margin clearer, supervised contrastive loss is proposed. Further, pseudo-label-guided consistency loss is proposed for extracting beneficial information from unlabeled data for downstream classification tasks.

3.2.2 Supervised Contrastive Representation Learning. The decision margin is defined as the distance between two different classes, which can make the classification result of the sample uncertain [32]. Thus, a clear decision margin not only enhances the robustness of the classification network but also its generalization, which is important in the few-label scenario.

For the classification tasks, the input of the classification network is the extracted representations. According to the definition of the decision margin, it is affected by the distribution of the representations—that is, a compact distribution will lead to a clearer decision margin. Therefore, the extracted representations should have good cluster structure—that is, representations with the same labeled data should be close to each other in the representation space, whereas representations with differently labeled data should be far from each other. Considering that in semi-supervised scenarios the insufficient labeled data will result in not being able to extract enough information using only cross-entropy loss, supervised contrastive loss needs to be imposed on the labeled data to be able to use the labeled information more effectively.

Contrastive learning constrains the representations of positive example pairs to be close to each other and representations of negative example pairs to be far from each other. In the supervised scenario, positive example pairs can be viewed as samples with the same label, and negative example pairs can be viewed as samples with different labels.

Inspired by this, we propose supervised contrastive loss to increase the utilization of the labeled data. The specific expression of the loss is given as follows:

$$L_{con} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\mathcal{P}_i}\frac{\exp(g(H[i],H[j]))}{\sum_{k\in\mathcal{N}_i}\exp(g(H[i],H[k]))}, \tag{4}$$

$$g(X,Y) = \frac{XY^T}{||X|| * ||Y||}, \tag{5}$$

where $g(x, y)$ measures the distance between two representations x and y, and cosine similarity is used in this equation. $\mathcal{P}_i$ denotes the set of positive samples consisting of samples with the same label as sample i, and $\mathcal{N}_i$ denotes the set of negative samples consisting of samples with the different label as sample i. H is the averaged representations, and $H[i]$ is the averaged representation of sample i.

First, to extract meaningful information from the raw data, each client trains an encoder to obtain the representations, noted as $H^{(v)}$ and calculated as Equation (6). For the consideration of performance, the representations used for the downstream classification tasks should contain information about each client. Therefore, the aggregation method of $H^{(v)}$ is crucial. Furthermore, we adopt a simple but effective approach to aggregation by calculating the mean value of all $H^{(v)}$, which is represented as Equation (7). Finally, we obtain the representations H for the downstream classification tasks.

$$H^{(v)} = f_v(X^{(v)}; w_v), \quad s.t.\ v = 1, \cdots, V \tag{6}$$

$$H = \frac{1}{V} \sum_{v=1}^{V} H^{(v)} \tag{7}$$

Since H is used for classification, and cross-entropy loss is frequently used in classification tasks to place constraints on representations [17, 31], the cross-entropy loss is employed to constrain the representations H. To implement the classification task, a global classifier cls, parameterized by θ, is employed. Finally, the loss function is given as follows:

$$L_{CE} = -\sum_{i=1}^{K-1} y_i \log(prob_i), \tag{8}$$

$$prob = cls(H; \theta), \tag{9}$$

where $prob$ is the classification results, and $prob_i$ is the i-th value of vector $prob$.

In summary, L_{CE} enforces the client-specific feature encoders to extract label-related information of the labeled data, whereas L_{con} minimizes the distances between the features with the same label and enhances the discrimination of the features. Combining these two loss items, label-related information is deeply exploited and the classification decision margin is clearer. Finally, the classification accuracy is improved.

3.2.3 Pseudo-Label-Guided Consistency Learning. Since label information may be very sparse, traditional supervised learning methods converge slowly. To address this challenge, we employed a semi-supervised learning approach.

In the VFL setting, the samples corresponding to each client-owned dataset come from the same entity. Therefore, there is a strong consistency between different client data. However, the consistency information is not always beneficial for downstream tasks and may even have side effects [39]. Considering the classification as downstream tasks, the label-related consistency information should be extracted to assist the classification task.

To extract the label-related consistency information, the classification result using each $H^{(v)}$ should be similar. Therefore, KL divergence is utilized to measure the degree of similarity of the classification results. Considering that the aggregated representations H contain information from multiple clients, it will work better in the classification task. To make the learned consistency information contribute to the classification task, we force to minimize the KL divergence between the classification results of each $H^{(v)}$ and the aggregated representation H. Finally, the problem is

formulated as

$$L_{lgc} = \sum_{v=1}^{V} D_{KL}(cls(H^{(v)}; \theta^{(v)}), cls(H; \theta)). \tag{10}$$

Due to the unique nature of data owned by each client, it is not reasonable to use a classifier to obtain the classification results for each $H^{(v)}$ and reduce the KL divergence. Here, we set a classifier for each client, whose parameters are represented by $\theta^{(v)}$.

Obviously, L_{lgc} is not related to the label information, and therefore it is possible to calculate L_{lgc} on unlabeled data. During the training process, the prediction of client-specific classifiers could be similar to the result of the global classifier. Additionally, this enforces the client-specific feature encoders to extract the consistent information which benefits the classification. Consequently, L_{lgc} could introduce the unlabeled data into the training process and improve the performance of classification.

3.3 Federated Optimization

To enforce the local encoders to encode better representations, and the global classifier to make a better prediction, the overall objective function is given as

$$L = L_{CE} + \lambda * L_{con} + \beta * L_{lgc}. \tag{11}$$

In Equation (11), L_{con} could clarify the classification decision margin, and L_{lgc} could boost dataset utilization and improve the performance of classification. To balance the role of the two terms, two hyperparameters λ and β are set. It is not difficult to find that the objective is a convex function and can be optimized with gradient descent. However, in VFL settings, the transmission of raw data is forbidden, which leads to the fact that the gradients of parameters cannot be calculated directly.

Fortunately, the gradient can be calculated indirectly according to the chain rule. For the local feature extractor, the gradient of parameter w_v can be calculated with

$$grad_{w_v} = \frac{\partial L}{\partial w_v} = \frac{\partial L}{\partial H^{(v)}} \frac{\partial H^{(v)}}{\partial w_v}. \tag{12}$$

Therefore, gradient descent can be used for updating the parameters of the *client-specific feature encoder*, and the detailed function is given as Equation (13). For the parameter of the global classifier and the client-specific classifier, $\theta, \theta^{(v)}$, their gradients can be calculated directly, and the detailed updating functions are given as Equations (14) and (15), respectively. In the updating equations, η is the learning rate.

$$w_v = w_v + \eta * g_{w_v} = w_v + \eta * \frac{\partial L}{\partial H^{(v)}} \frac{\partial H^{(v)}}{\partial w_v} \tag{13}$$

$$\theta = \theta - \eta * \frac{\partial L}{\partial \theta} \tag{14}$$

$$\theta^{(v)} = \theta^{(v)} - \eta * \frac{\partial L}{\partial \theta^{(v)}} \tag{15}$$

During the training, a client needs to transmit $H^{(v)}$ to the server, and after the server receives all representations $\{H^{(v)}\}_{v=1}^{V}$, the gradient $\frac{\partial L}{\partial H^{(v)}}$ can be calculated and transmitted to the corresponding client. The details of the proposed algorithm are given in Algorithm 1. Considering $\{H^{(v)}\}_{v=1}^{V}$ is equivalent to the encrypted data with a non-linear transformation, there is no privacy leakage.

3.4 Enforcing DP

During the training process illustrated by Algorithm 1, the uploaded information by the client is representations $\{H^{(v)}\}_{v=1}^{V}$. Considering that the representations are equivalent to the encrypted data, this does not raise privacy issues.

Recently, several works [18, 29, 46, 47] focused on the feature inference attack on VFL systems and provided several threat models to extract raw data from the uploaded features. Since DP [8] is an effective method to prevent inference attack, we introduce DP into the proposed method.

For the client-specific encoders, each of them extracts the embeddings of local data with a neural network, and the procedure for extracting embeddings can be presented as follows:

$$h_0 = x^{(v)}, \tag{16}$$

$$h_l = \sigma_l(w_l h_{l-1} + b_l), \quad l = 1, 2, \ldots, L, \tag{17}$$

$$h = h_L, \tag{18}$$

where σ_l is an activation function, which introduces non-linear transformation into the process of extracting features, and w_l, b_l are the optimizable parameters of composition w_v—for example, $w_v = [w_1, b_1, \ldots, w_L, b_L]$. To introduce DP into the procedure of feature extracting, we perturb the embeddings by adding a random noise at the last layer, and the perturbed embeddings are given as

$$h_L = \sigma(w_L h_{L-1} + b_L) + Z, \tag{19}$$

where Z is a random variable. To ensure that h is smooth and enables DP, we enforce Z to satisfy the Gaussian distribution with zero means, and Equation (19) can be reformulated as Equation (20), where c^2 is the variance. For the proposed method, we set a flag *use_DP* for enabling DP.

$$h_L = \sigma(w_L h_{L-1} + b_L) + \mathcal{N}(0, c^2) \tag{20}$$

4 EXPERIMENTS

4.1 Datasets

For verifying the effectiveness of the proposed SSVFL, we collect four public datasets and conduct experiments on them. Detailed information on the dataset employed is as follows:

- *UCI* [2] contains 2,000 handwritten numeric images, ranging from 0 to 9. Additionally, there are three types of features: PIX feature, FOU feature, and MOR feature.
- *Caltech101* [21] contains 9,144 object images, and six different types of features are extracted: Gabor feature, wavelet moments feature, CENTRIST feature, HOG feature, GIST feature, and LBP feature. Additionally, there are 102 classes in total.
- *MNIST10k*[1] contains 100,000 digit images, and three types of features are extracted: 30-dimension IsoProjection features, 9-dimension Linear Discriminant Analysis (LDA) features, and 9-dimension Neighborhood Preserving Embedding (NPE) features.
- *Reuters*[2] contains 18,758 documents with five languages, and each sample is represented as a bag of words. Considering that this dataset is stored in the form of a sparse matrix, we transformed it into the form of a dense matrix before training.

Detailed information about the given four datasets is presented in Table 1.

[1] http://yann.lecun.com/exdb/mnist/

[2] http://archive.ics.uci.edu/ml/machine-learning-databases/00259/

Table 1. Detailed Information about the Four Datasets

Dataset ID	Dataset	Samples	Types of Feature	Feature Dimensions
1	UCI	2,000	3	240/76/6
2	Caltech101	9,144	6	48/40/254/1,984/512/928
3	MNIST10k	10,000	3	30/9/30
4	Reuters	18,758	5	21,531/24,892/34,251/16,606/11,547

ALGORITHM 1: The main steps of SSVFL.

Input: Local dataset $X^{(v)}$
Output: Trained models $\{e_v\}_{v=1}^{V}$,
Initialize the clients' model parameters.
procedure CLIENTFORWARD
$H^v = e_v(X^{(v)}; w_v)$
if use_DP **then**
N = sample_noise()
$H^{(v)} = H^{(v)} + N$
end if
return $H^{(v)}$
procedure CLIENTBACKWARD($\frac{\partial L}{\partial H^{(v)}}$)
Update parameters of local feature extractor with $w_v = w_v - \eta * \frac{\partial L}{\partial H^{(v)}} * \frac{\partial H^{(v)}}{\partial w_v}$
procedure SERVER EXECUTION
for all each communication rount $t \in \{1, \ldots, T\}$ **do**
for all $v \in [1, 2, \ldots, V]$ **do**
$H^{(v)} = ClientForward(v)$
end for
$H = \frac{1}{V} \sum_{v=1}^{V} H^v$
Calculate the cross-entropy loss L_{CE}
Calculate the supervised contrastive loss L_{con}
Calculate the pseudo-label-guided consistency loss L_{lgc}
Calculate the complete objective loss L by Equation (11)
Update θ with Equation (14)
update $\theta^{(i)}$ with Equation (15)
Calculate the gradient $\{\frac{\partial L}{\partial H^{(v)}}\}_{v=1}^{V}\}$
for all $v \in [1, 2, \ldots, V]$ **do**
$LocalBackward(\frac{\partial L}{\partial H^{(v)}})$
end for
end for

4.2 Baseline Methods

To illustrate that the information from other clients benefits the semi-supervised classification task, we train a classifier with the first type of feature, which is denoted as *Local Training*. Besides, we collect three state-of-the-art VFL methods and compare them with the proposed SSVFL, and descriptions about them are listed as follows:

— *Local Training*: This method feeds the data of the labeled client into a network and trains the classification network with only the labeled data.

— *VAFL* [6]: This method is proposed for combining information from other clients to enhance the classification performance. To improve the efficiency of training, VAFL adopts an asynchronous approach for model training.
— *MMVFL* [10]: This method provides a privacy-preserving label sharing mechanism, which benefits the classification tasks.
— *PyVertical* [37]: This method introduces split neural networks into VFL settings and trains a Vertically Federated Machine Learning algorithm on data distributed across the premises of multiple data owners with the PySyft library [38].
— *FedOnce* [49]: This method learns representations locally with the unsupervised representation learning method NAT, then sends the representations to the server for training the aggregated classifier.

4.3 Implementation Details

First, we construct a VFL scenario by distributing one type of feature to one client. For the UCI dataset, we set up three clients and allow client 1 to hold the PIX features, client 2 to hold the FOU features, and client 3 to hold the MOR features. For the Caltech101 dataset, six clients are set, and each client maintains one type of feature. The labels are maintained on the server, and we randomly select a portion of the data as labeled data, ranging from 1% to 20%, whereas the remaining unlabeled data are used for model testing. For calculating accuracy on the test dataset, we use client-specific encoders to obtain representations of the test data, $H_{test}^{(v)}$, and then send them to the server. At the server side, the global representations H_{test} are obtained by averaging each $H_{test}^{(v)}$. Finally, the classification results are obtained by feeding $H_{test}^{(v)}$ to the global classifier.

For the proposed SSVFL, we vary λ in $\{0.001, 0.01, 0.1\}$ and β in $\{0.001, 0.003, 0.03\}$. The dimensions of the encoder are set as $d^{(v)} - 200 - 100 - 64$, and the dimensions of classifiers are set as $64 - c$, where c is the number of categories. For large datasets like Caltech101, the dimensions of the encoder are set as $d^{(v)} - 500 - 200 - 128$, and the dimensions of classifiers are set as $128 - c$. All of these networks are optimized by the Adam optimizer with a learning rate of 0.001. To ensure that the raw data is not inferred by the uploaded representations, we add noise $n \sim \mathcal{N}(0, 1)$ to the representations.

For a fair comparison, with respect to Local Training, we feed the data on the client with label information into the classifier and train the classifier with gradient descent. For MMVFL, we feed the labeled data into the network, and set η and ζ_k to 1,000 according to the settings in the original paper. For the consideration of performance, we implement VAFL without *local perturbation* and *enforcing DP*. For FedOnce,[3] we implement FedOnce-L0 and train the local model until convergence, then send the extracted features to the server for classification. For PyVertical,[4] we feed data into the provided SplitNN and aggregate the features for classification in a form of averaging.

4.4 Experimental Results

We illustrate the experimental results of the compared methods with the proposed SSVFL in Table 2. The highest accuracies are noted in bold font, and the second best results are rendered with an underline.

From this table, it is no surprise that Local Training with only the data of one client performs less effectively on most datasets. The reason is that in this method, it is not possible to use information from other clients, which is crucial in the case of insufficient label information. In contrast, VFL methods, such as VAFL, MMVFL, and FedOnce-L0, could utilize useful information from each

[3] https://github.com/JerryLife/FedOnce
[4] https://github.com/OpenMined/PyVertical

Table 2. Classification Accuracy on Different Label Rates and on the Given UCI, Caltech101, MNIST10k, and Reuters Datasets

Dataset	Methods	1%	3%	5%	10%	20%
UCI	Local Training	0.5005	0.8675	0.9068	0.9378	0.9518
	FedOnce-L0	0.5121	0.8031	0.8211	0.8761	0.8963
	MMVFL	0.4611	0.7619	0.8658	0.9078	0.9281
	VAFL	0.5843	0.8711	0.9084	0.9456	0.9550
	PyVertical	0.5061	0.8479	0.8916	0.9383	0.9563
	SSVFL-DP	0.6313	0.8994	0.9247	0.9556	0.9644
	SSVFL	**0.6465**$^{\uparrow 10.64\%}$	**0.9010**$^{\uparrow 3.43\%}$	**0.9279**$^{\uparrow 2.15\%}$	**0.9567**$^{\uparrow 1.17\%}$	**0.9650**$^{\uparrow 0.91\%}$
Caltech101	Local Training	0.1743	0.2051	0.2455	0.2894	0.3203
	FedOnce-L0	0.2592	0.2685	0.3479	0.3633	0.4222
	MMVFL	0.1882	0.1894	0.2107	0.2174	0.2239
	VAFL	0.2191	0.2720	0.3042	0.3491	0.3897
	PyVertical	0.2617	0.2985	0.3554	0.4181	0.5078
	SSVFL-DP	0.2850	0.3440	0.4034	0.4600	0.5262
	SSVFL	**0.3013**$^{\uparrow 15.14\%}$	**0.3638**$^{\uparrow 21.86\%}$	**0.4049**$^{\uparrow 13.94\%}$	**0.4603**$^{\uparrow 10.08\%}$	**0.5291**$^{\uparrow 4.19\%}$
MNIST10k	Local Training	0.7346	0.8321	0.8654	0.8958	0.9035
	FedOnce-L0	0.7257	0.7454	0.7682	0.7730	0.7811
	MMVFL	0.6558	0.7847	0.8104	0.8270	0.8358
	VAFL	0.716	0.836	0.878	0.901	0.918
	PyVertical	0.7707	0.8762	0.8965	0.9117	0.9263
	SSVFL-DP	**0.8437**$^{\uparrow 9.48\%}$	0.9019	0.9197	0.9312	**0.9424**$^{\uparrow 1.73\%}$
	SSVFL	0.8246	**0.9036**$^{\uparrow 3.13\%}$	**0.9201**$^{\uparrow 2.63\%}$	**0.9343**$^{\uparrow 2.48\%}$	0.9407
Reuters	Local Training	0.5705	0.5813	0.6178	0.6850	0.6878
	FedOnce-L0	0.6402	0.6867	0.6713	0.4499	0.5219
	MMVFL	0.5910	0.6442	0.7077	0.7897	0.8828
	VAFL	0.6144	0.6557	0.6708	0.7785	0.8074
	PyVertical	0.6343	0.6587	0.6631	0.7679	0.7905
	SSVFL-DP	0.7648	0.8121	0.8211	0.8371	0.8561
	SSVFL	**0.7656**$^{\uparrow 19.59\%}$	**0.8188**$^{\uparrow 19.23\%}$	**0.8329**$^{\uparrow 17.69\%}$	**0.8517**$^{\uparrow 7.85\%}$	**0.8643**$^{\uparrow 4.29\%}$

client and hence improves the classification accuracy. Recent VFL methods, such as FedOnce-L0, MMVFL, PyVertical, and VAFL, only consider the labeled data, and as the label rate decreases, the performance drops dramatically. Furthermore, it is concluded that for the traditional VFL methods, VAFL and MMVFL optimize the local neural networks with cross-entropy loss, which does not extract enough label-related information, and hence achieve limited performance. Nonetheless, SSVFL introduces supervised contrastive learning, by which the classification decision margin is clarified. Moreover, the unsupervised method FedOnce-L0 is sensitive to the initial parameters, such as C_i, and therefore less robust, as evidenced by the fact that the classification performance does not improve with label rates on the Reuters dataset. The proposed SSVFL and SSVFL-DP are more robust to the initial parameters, since the performance improves with label rate on each dataset. By comparing the results of SSVFL and SSVFL-DP, the effect of noise on the classification results is not significant, with the effect being reduced by no more than 2%. Surprisingly, SSVFL-DP may outperform SSVFL—for example, on the Caltech101 dataset with 10% labeled data, this may be caused by the introduction of noise that enhances the generalization of the model. Therefore, it is concluded that the proposed method is able to protect the privacy and does not affect performance. Thus, we do not add the DP module in the following experiments. For the proposed SSVFL, the improvement of classification performance increases with the decrease in labeling rate—for instance, SSVFL achieves 10.64% improvement with 1% labeled data but 0.91% with 20% labeled data for the UCI dataset.

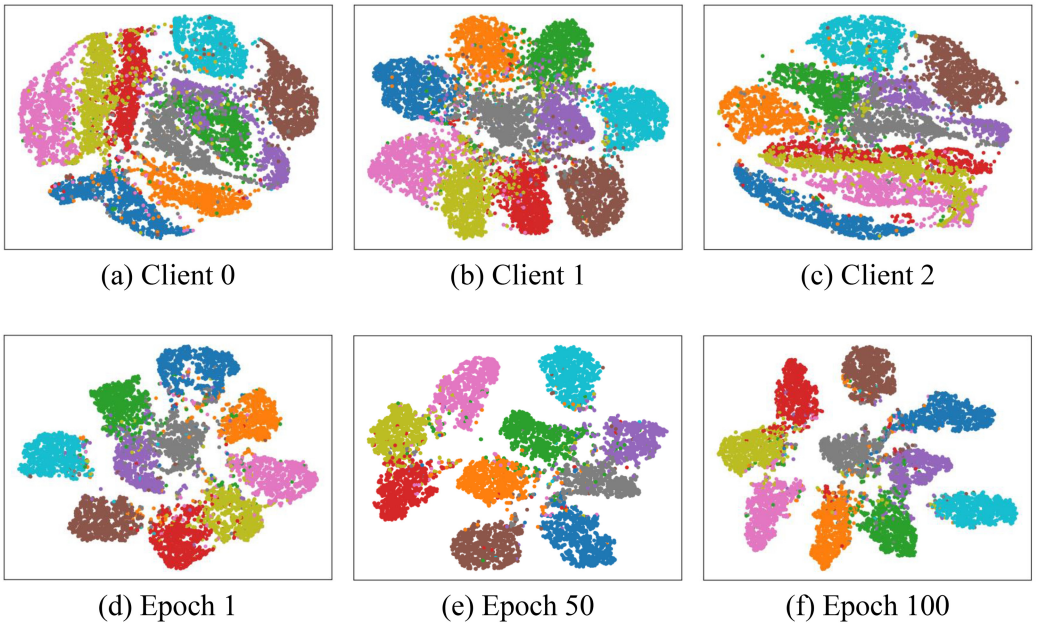


Fig. 2. A scatter plot of classification using the raw data maintained by each client and the fused representation on the MNIST10k dataset as the training epochs increase.

Figure 2 shows the gradual separation process of different classes as the number of training epochs increases. As a common tool for the dimensionality reduction of data, T-SNE is employed to project high-dimensional representations onto a two-dimensional plane and to visualize the representations. It is observed that for each client, the sample points without processing are cluttered. As the training process progresses, the shapes of various classes gradually become clear and the distances between the different classes increase. Therefore, it is concluded that with the increasing training epochs, the decision margin becomes clearer, which guarantees better classification performance.

4.5 Ablation Study

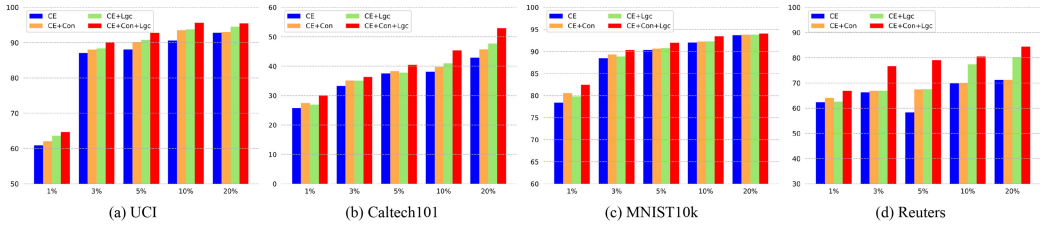
The proposed SSVFL consists of three loss terms, including cross-entropy loss L_{CE}, supervised contrastive loss L_{con}, and pseudo-label-guided consistency L_{lgc}. To illustrate their effects on the classification task separately, the results are given in Table 3 and Figure 3.

In Table 3 and Figure 3, CE means using L_{CE} to train the networks, CE+Con notes $L_{CE} + L_{con}$, CE+Lgc means $L_{CE}+L_{lgc}$, and CE+Con+Lgc means $L_{CE}+L_{con}+L_{lgc}$. From this table, it is concluded that L_{con} and L_{lgc} both can have a performance improvement, and the combination of the two will improve the effect more obviously. It is worth noting that for the different datasets, the two losses work differently. For example, on the UCI dataset, L_{lgc} contributes more to the final result, and L_{lgc} results in a 1% to 2% improvement in accuracy, whereas L_{con} only improved by up to 1%. However, on the Caltech101 dataset, L_{con} contributes more. This may be caused by the nature of the dataset. For the UCI dataset, the consistency between different kinds of features is more strongly expressed, leading to a better classification result.

Besides, the effect of cross-entropy is investigated. Considering the classification task as a downstream task, we chose MSE as an alternative to cross-entropy. The definition of MSE is

Table 3. Ablation Results (%) on UCI, Caltech101, MNIST10k, and Reuters Datasets

Loss	UCI					Caltech101				
	1%	3%	5%	10%	20%	1%	3%	5%	10%	20%
CE	60.91 -	87.11 -	88.11 -	90.61 -	92.81 -	25.72 -	33.27 -	37.50 -	38.12 -	42.93 -
CE+Con	62.07 ↑	88.04 ↑	90.16 ↑	93.56 ↑	93.00 ↑	27.50 ↑	35.16 ↑	38.31 ↑	39.76 ↑	45.75 ↑
CE+Lgc	63.59 ↑	88.40 ↑	90.79 ↑	93.78 ↑	94.56 ↑	26.90 ↑	35.12 ↑	37.84 ↑	40.96 ↑	47.66 ↑
MSE+Con+Lgc	56.11 ↓	89.58 ↑	92.37 ↑	95.06 ↑	94.81 ↑	21.35 ↑	29.12 ↓	31.02 ↓	33.10 ↓	33.62 ↓
CE+Con+Lgc	**64.65** ↑	**90.10** ↑	**92.79** ↑	**95.67** ↑	**96.50** ↑	**30.13** ↑	**36.38** ↑	**40.49** ↑	**45.43** ↑	**52.91** ↑
Loss	MNIST10k					Reuters				
	1%	3%	5%	10%	20%	1%	3%	5%	10%	20%
CE	78.41 -	88.45 -	90.37 -	92.05 -	93.72 -	62.33 -	66.28 -	58.36 -	69.86 -	71.80 -
CE+Con	80.56 ↑	89.32 ↑	90.65 ↑	92.26 ↑	93.84 ↑	64.13 ↑	66.85 ↑	67.47 ↑	70.04 ↑	71.21 ↑
CE+Lgc	79.81 ↑	88.87 ↑	90.77 ↑	92.30 ↑	93.85 ↑	62.64 ↑	66.87 ↑	67.55 ↓	77.45 ↑	80.23 ↑
MSE+Con+Lgc	74.07 ↓	85.87 ↓	88.60 ↓	92.03 ↓	92.80 ↓	61.67 ↓	76.51 ↑	78.78 ↑	79.01 ↑	84.23 ↑
CE+Con+Lgc	**82.46** ↑	**90.36** ↑	**92.01** ↑	**93.43** ↑	**94.07** ↑	**66.83** ↑	**76.69** ↑	**79.09** ↑	**80.57** ↑	**84.40** ↑



(a) UCI (b) Caltech101 (c) MNIST10k (d) Reuters

Fig. 3. Ablation details on the given datasets.

given as

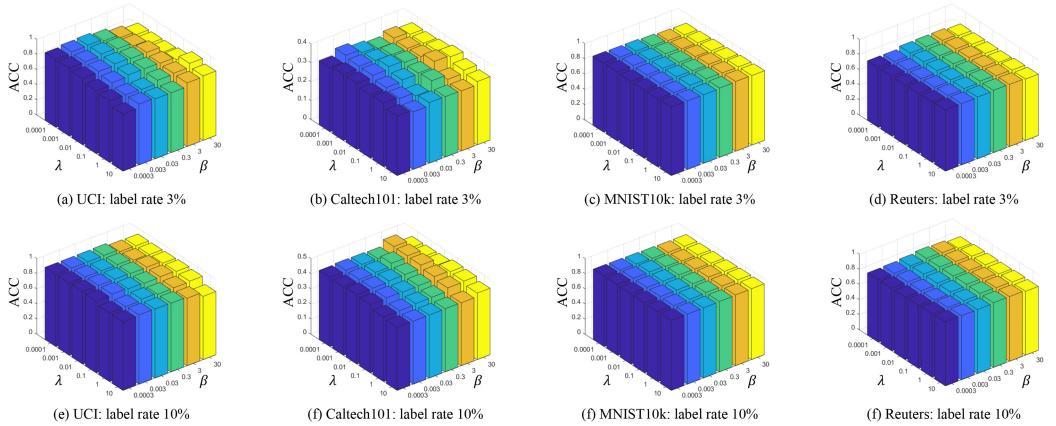
$$L_{MSE} = \frac{1}{N} \sum ||prob - one\_hot(y)||_2^2, \tag{21}$$

where $prob$ is the output of the model, y is the label, and $one\_hot(y)$ converts y into one hot vector. The results are given in Table 3, and MSE+Con+Lgc means $L_{MSE} + L_{con} + L_{lgc}$. From the table, it is concluded that when the number of labeled samples is insufficient—for example, the label rate is 1% for the UCI dataset—MSE cannot fit the data well, resulting in a serious decline in classification accuracy. When labeled samples are sufficient, the effect of MSE will be slightly worse than that of cross-entropy. This is mainly because MSE is less conducive to gradient update than cross-entropy.

4.6 Parameter Sensitivity Investigation

In the objective function (11), there are two tradeoff parameters λ and β. To explore their effects on the classification performance of the proposed SSVFL, we vary λ in $\{0.0001, 0.001, 0.01, 0.1, 1, 10\}$ and β in $\{0.0003, 0.003, 0.03, 0.3, 3, 30\}$, and the label rates are set as 0.03 and 0.1.

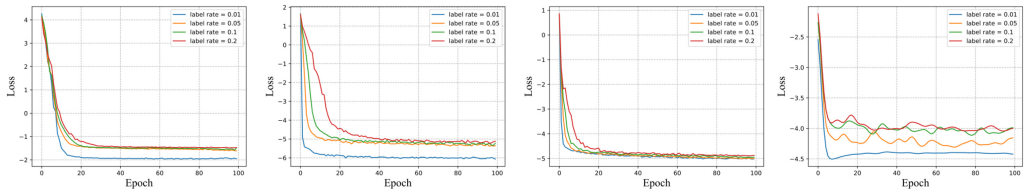
Figure 4 is used for comparison. In the figure, it can be seen that the classification performance is inferior when β is set to a large value on the UCI dataset, especially when the label rate is low. This is attributed to excessive penalty for the contrastive loss L_{lgc}, resulting in extracting too much consistent information including background or noise, which is irrelevant to the downstream classification tasks, and in return reduces the representational power of the representation and degrades the classification performance. Interestingly, when β is tuned to be relatively large with a large λ, the performance becomes a little better. On the Caltech101 and MNIST10k datasets, the performance is relatively stable in general. However, they have a common phenomenon in which the classification accuracies are worse when β is set to the biggest value, which inspires us to determine that β should not be set too big.



(a) UCI: label rate 3% (b) Caltech101: label rate 3% (c) MNIST10k: label rate 3% (d) Reuters: label rate 3%

(e) UCI: label rate 10% (f) Caltech101: label rate 10% (f) MNIST10k: label rate 10% (f) Reuters: label rate 10%

Fig. 4. Parameter sensitivity investigation with respect to λ and β on UCI, Caltech101, and MNIST10k datasets.



(a) UCI (b) Caltech101 (c) MNIST10k (d) Reuters

Fig. 5. Convergence curves of objective values of the proposed SSVFL on UCI, Caltech101, MNIST10k, and Reuters datasets

4.7 Convergence Verification

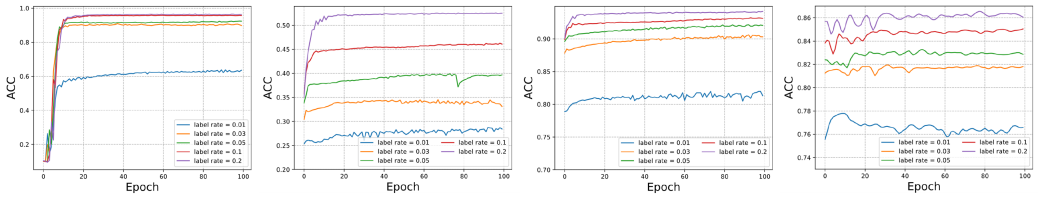
To illustrate the convergence of the proposed SSVFL, we record the value of the objective function after each training epoch on the four datasets with the label rate varying at [1%, 5%, 10%, 20%].

The results are present the convergence curves of the proposed SSVFL in Figure 5. To make it more clear, the objective values are processed with the *log* function. From this figure, it can be concluded that the proposed SSVFL has good convergence even though the amount of labeled samples is insufficient. However, as the label rate increases, the convergence rate does not accelerate as intuitively. As can be seen in Figure 5(b), on the Caltech101 dataset, SSVFL converges at nearly 20 rounds when the label rate is 1% and at nearly 50 rounds when the label rate is 20%. This is because the labeled data increase and SSVFL needs to spend more time to extract meaningful information from the labeled data, thus making the convergence slower.

In Figure 6, it is concluded that as the training procedure goes on, the classification accuracy is improved, and after several epochs, the accuracy does not change significantly as the proposed method converges. For an extremely large dataset, such as the Reuters dataset, the proposed method converges rapidly. Thus, the classification accuracies do not improve too much, as shown in Figure 6(d).

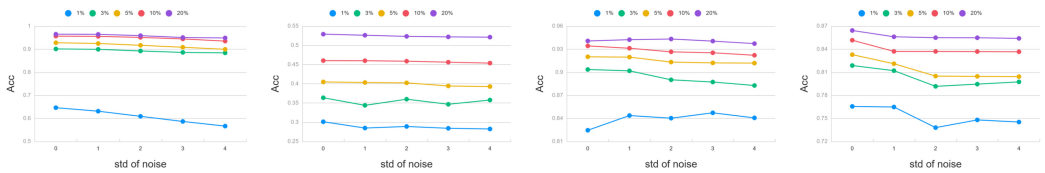
4.8 The Impact of DP

DP is used to protect the privacy. Due to the addition of Gaussian noise, the attacker cannot obtain private data from the uploaded embeddings. However, noise may affect the training of the model, thereby affecting the classification accuracy [45].



(a) UCI (b) Caltech 101 (c) MNIST10k (d) Reuters

Fig. 6. The curves of classification accuracies in different numbers of epoch and different label rates.



(a) UCI (b) Caltech101 (c) MNIST10k (d) Reuters

Fig. 7. The results of different datasets under different standard deviations of noise.

To reveal the impact of DP on the accuracy, we conducted multiple experiments, adding noise with different c, standard deviations, in the training phase for different label rates in different datasets, and the standard deviation range is {1, 2, 3, 4}. The detailed results are given in Figure 7.

From Figure 7, it is concluded that classification accuracy decreases as the standard deviation of the added noise increases, illustrated in Figure 7(a), (c), and (d). However, DP does not negatively affect model effects in all situations. In some special scenarios, such as in Figure 7(c), where the labeling rate is 1% for the MNIST10k dataset, the effect improves with increasing noise. Overall, the introduction of DP still causes some degradation in accuracy.

5 CONCLUSION

In this article, we proposed a novel VFL method, SSVFL, which can be applied to semi-supervised scenarios. Specifically, we considered improving classification performance from two aspects: one to fully explore the label information and the other to make full use of the unlabeled data. Besides, we proposed a privacy-preserving optimization method for the proposed SSVFL, transmitting the embeddings rather than the raw data. We found that SSVFL outperforms baselines significantly with empirical evidence, showing its ability to address the difficult challenges in semi-supervised VFL. However, during the training process, the embeddings are required to be transferred to the server, which is bandwidth intensive. Therefore, future work needs to be done to deal with communication costs for SSVFL.

REFERENCES

[1] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. 2019. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818* (2019).

[2] Arthur Asuncion and David Newman. 2007. UCI Machine Learning Repository. Retrieved April 15, 2024 from https://archive.ics.uci.edu

[3] Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory*. 92–100.

[4] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. 2019. Towards federated learning at scale: System design. *Proceedings of Machine Learning and Systems* 1 (2019), 374–388.

[5] Akin Caliskan, Armin Mustafa, Evren Imre, and Adrian Hilton. 2020. Multi-view consistency loss for improved single-image 3D reconstruction of clothed people. In *Proceedings of the Asian Conference on Computer Vision*.
[6] Tianyi Chen, Xiao Jin, Yuejiao Sun, and Wotao Yin. 2020. VAFL: A method of vertical asynchronous federated learning. *arXiv preprint arXiv:2007.06081* (2020).
[7] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. 2021. Exploiting shared representations for personalized federated learning. In *Proceedings of the International Conference on Machine Learning*. 2089–2099.
[8] Cynthia Dwork. 2006. Differential Privacy. In *Automata, Langages and Programming*. Lecture Notes in Computer Science, Vol. 4052. Springer, 1–12.
[9] Farzan Farnia, Amirhossein Reisizadeh, Ramtin Pedarsani, and Ali Jadbabaie. 2022. An optimal transport approach to personalized federated learning. *IEEE Journal on Selected Areas in Information Theory* 3, 2 (2022), 162–171.
[10] Siwei Feng and Han Yu. 2020. Multi-participant multi-class vertical federated learning. *arXiv preprint arXiv:2001.11154* (2020).
[11] Mingfei Gao, Zizhao Zhang, Guo Yu, Sercan Ö Arık, Larry S. Davis, and Tomas Pfister. 2020. Consistency-based semi-supervised active learning: Towards minimizing labeling cost. In *Proceedings of the 16th European Conference on Computer vision (ECCV'20)*. 510–526.
[12] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. 2020. An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems* 33 (2020), 19586–19597.
[13] Chen Gong, Zhenzhe Zheng, Fan Wu, Yunfeng Shao, Bingshuai Li, and Guihai Chen. 2023. To store or not? Online data selection for federated learning with limited storage. In *Proceedings of the ACM Web Conference 2023*. 3044–3055.
[14] Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*. 297–304.
[15] Stephen Hardy, Wilko Henecka, Hamish Ivey-Law, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne. 2017. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv preprint arXiv:1711.10677* (2017).
[16] Nakamasa Inoue and Keita Goto. 2020. Semi-supervised contrastive learning with generalized contrastive loss and its application to speaker recognition. In *Proceedings of the 2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC'20)*. 1641–1646.
[17] Antoine Jamin and Anne Humeau-Heurtier. 2019. (Multiscale) cross-entropy methods: A review. *Entropy* 22, 1 (2019), 45.
[18] Xiao Jin, Pin-Yu Chen, Chia-Yi Hsu, Chia-Mu Yu, and Tianyi Chen. 2021. CAFE: Catastrophic data leakage in vertical federated learning. *Advances in Neural Information Processing Systems* 34 (2021), 994–1006.
[19] Yan Kang, Yang Liu, and Tianjian Chen. 2020. FedMVT: Semi-supervised vertical federated learning with multiview training. *arXiv preprint arXiv:2008.10838* (2020).
[20] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proceedings of the 37th International Conference on Machine Learning*, Hal Daumé III and Aarti Singh (Eds.). Proceedings of Machine Learning Research, Vol. 119, Hal Daumé III and Aarti Singh (Eds.). PMLR, 5132–5143.
[21] Fei-Fei Li, Marco Andreeto, Marc'Aurelio Ranzato, and Pietro Perona. 2022. *Caltech 101 [Data Set]*. CaltechDATA. https://doi.org/10.22002/D1.20086
[22] Junnan Li, Caiming Xiong, and Steven C. H. Hoi. 2021. CoMatch: Semi-supervised learning with contrastive graph regularization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 9475–9484.
[23] Qinbin Li, Bingsheng He, and Dawn Song. 2021. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'21)*. 10713–10722.
[24] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems* 2 (2020), 429–450.
[25] 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).
[26] Youwei Liang, Dong Huang, and Chang-Dong Wang. 2019. Consistency meets inconsistency: A unified graph learning framework for multi-view clustering. In *Proceedings of the 2019 IEEE International Conference on Data Mining (ICDM'19)*. IEEE, 1204–1209.
[27] Tao Lin, Lingjing Kong, Sebastian U. Stich, and Martin Jaggi. 2020. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems* 33 (2020), 2351–2363.
[28] Yang Liu, Yan Kang, Tianyuan Zou, Yanhong Pu, Yuanqin He, Xiaozhou Ye, Ye Ouyang, Ya-Qin Zhang, and Qiang Yang. 2022. Vertical federated learning. *arXiv preprint arXiv:2211.12814* (2022).
[29] Xinjian Luo, Yuncheng Wu, Xiaokui Xiao, and Beng Chin Ooi. 2021. Feature inference attack on model predictions in vertical federated learning. In *Proceedings of the 2021 IEEE 37th International Conference on Data Engineering (ICDE'21)*. IEEE, 181–192.

[30] Xiaosong Ma, Jie Zhang, Song Guo, and Wenchao Xu. 2022. Layer-wised model aggregation for personalized federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10092–10101.

[31] Shie Mannor, Dori Peleg, and Reuven Rubinstein. 2005. The cross entropy method for classification. In *Proceedings of the 22nd International Conference on Machine Learning*. 561–568.

[32] David Mickisch, Felix Assion, Florens Greßner, Wiebke Günther, and Mariele Motta. 2020. Understanding the decision boundary of deep neural networks: An empirical study. *arXiv preprint arXiv:2002.01810* (2020).

[33] Ion Muslea, Steven Minton, and Craig A. Knoblock. 2006. Active learning with multiple views. *Journal of Artificial Intelligence Research* 27 (2006), 203–233.

[34] Jinlong Pang, Jieling Yu, Ruiting Zhou, and John C. S. Lui. 2022. An incentive auction for heterogeneous client selection in federated learning. *IEEE Transactions on Mobile Computing*. Published Online, June 14, 2022.

[35] Krishna Pillutla, Kshitiz Malik, Abdel-Rahman Mohamed, Mike Rabbat, Maziar Sanjabi, and Lin Xiao. 2022. Federated learning with partial model personalization. In *Proceedings of the International Conference on Machine Learning*. 17716–17758.

[36] Protection Regulation. 2016. Regulation (EU) 2016/679 of the European Parliament and of the Council. *Regulation (EU)* 679 (2016), 2016.

[37] Daniele Romanini, Adam James Hall, Pavlos Papadopoulos, Tom Titcombe, Abbas Ismail, Tudor Cebere, Robert Sandmann, Robin Roehm, and Michael A. Hoeh. 2021. PyVertical: A vertical federated learning framework for multi-headed SplitNN. *arXiv preprint arXiv:2104.00489* (2021).

[38] Theo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. 2018. A generic framework for privacy preserving deep learning. *arXiv preprint arXiv:1811.04017* (2018).

[39] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. 2020. What makes for good views for contrastive learning? *Advances in Neural Information Processing Systems* 33 (2020), 6827–6839.

[40] Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. 2018. Multi-view consistency as supervisory signal for learning shape and pose prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2897–2905.

[41] Jesper E. Van Engelen and Holger H. Hoos. 2020. A survey on semi-supervised learning. *Machine Learning* 109, 2 (2020), 373–440.

[42] Cédric Villani. 2009. *Optimal Transport: Old and New*. Grundlehren der mathematischen Wissenschaften, Vol. 338. Springer.

[43] Hao Wang, Zakhary Kaplan, Di Niu, and Baochun Li. 2020. Optimizing federated learning on non-IID data with reinforcement learning. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM'20)*. IEEE, 1698–1707.

[44] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. 2020. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in Neural Information Processing Systems* 33 (2020), 7611–7623.

[45] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H. Yang, Farhad Farokhi, Shi Jin, Tony Q. S. Quek, and H. Vincent Poor. 2020. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security* 15 (2020), 3454–3469.

[46] Kang Wei, Jun Li, Chuan Ma, Ming Ding, Sha Wei, Fan Wu, Guihai Chen, and Thilina Ranbaduge. 2022. Vertical federated learning: Challenges, methodologies and experiments. *arXiv preprint arXiv:2202.04309* (2022).

[47] Haiqin Weng, Juntao Zhang, Feng Xue, Tao Wei, Shouling Ji, and Zhiyuan Zong. 2020. Privacy leakage of real-world vertical federated learning. *arXiv preprint arXiv:2011.09290* (2020).

[48] Yuncheng Wu, Shaofeng Cai, Xiaokui Xiao, Gang Chen, and Beng Chin Ooi. 2020. Privacy preserving vertical federated learning for tree-based models. *arXiv preprint arXiv:2008.06170* (2020).

[49] Zhaomin Wu, Qinbin Li, and Bingsheng He. 2022. Practical vertical federated learning with unsupervised representation learning. *IEEE Transactions on Big Data*. Published Online, June 6, 2022.

[50] Fan Yang, Kai Wu, Shuyi Zhang, Guannan Jiang, Yong Liu, Feng Zheng, Wei Zhang, Chengjie Wang, and Long Zeng. 2022. Class-aware contrastive semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14421–14430.

[51] J. H. Yang, C. Chen, H. N. Dai, M. Ding, L. L. Fu, and Z. B. Zheng. 2022. Hierarchical representation for multi-view clustering: From intra-sample to intra-view to inter-view. In *Proceedings of the Conference on Information and Knowledge Management*.

[52] J. H. Yang, C. Chen, H. N. Dai, M. Ding, Z. B. Wu, and Z. B. Zheng. 2022. Robust corrupted data recovery and clustering via generalized transformed tensor low-rank representation. *IEEE Transactions on Neural Networks and Learning Systems*. Early Access, November 3, 2022. DOI: 10.1109/TNNLS.2022.3215983.

[53] J. H. Yang, C. Chen, H. N. Dai, L. L. Fu, and Z. B. Zheng. 2022. A structure noise-aware tensor dictionary learning method for high-dimensional data clustering. *Information Sciences* 612 (2022), 87–106.

[54] J. H. Yang, L. L. Fu, C. Chen, H. N. Dai, and Z. B. Zheng. 2023. Cross-view graph matching for incomplete multi-view clustering. *Neurocomputing* 515 (2023), 79–88.
[55] Lei Yang, Jiaming Huang, Wanyu Lin, and Jiannong Cao. 2023. Personalized federated learning on non-IID data via group-based meta-learning. *ACM Transactions on Knowledge Discovery from Data* 17, 4 (2023), Article 49, 20 pages.
[56] Xihong Yang, Xiaochang Hu, Sihang Zhou, Xinwang Liu, and En Zhu. 2022. Interpolation-based contrastive learning for few-label semi-supervised learning. *IEEE Transactions on Neural Networks and Learning Systems*. Published Online, July 7, 2022.
[57] Chenhao Ying, Haiming Jin, Xudong Wang, and Yuan Luo. 2020. Double insurance: Incentivized federated learning with differential privacy in mobile crowdsensing. In *Proceedings of the 2020 International Symposium on Reliable Distributed Systems (SRDS'20)*. IEEE, 81–90.
[58] Chunjie Zhang, Jian Cheng, and Qi Tian. 2019. Multi-view image classification with visual, semantic and view consistency. *IEEE Transactions on Image Processing* 29 (2019), 617–627.
[59] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. 2021. A survey on federated learning. *Knowledge-Based Systems* 216 (2021), 106775.
[60] Qingsong Zhang, Bin Gu, Cheng Deng, Songxiang Gu, Liefeng Bo, Jian Pei, and Heng Huang. 2021. AsySQN: Faster vertical federated learning algorithms with better computation resource utilization. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3917–3927.
[61] Yuhang Zhang, Xiaopeng Zhang, Jie Li, Robert Qiu, Haohang Xu, and Qi Tian. 2022. Semi-supervised contrastive learning with similarity co-calibration. *IEEE Transactions on Multimedia* 25 (2022), 1749–1759. https://doi.org/10.1109/TMM.2022.3158069

Received 1 April 2023; revised 24 January 2024; accepted 31 March 2024