# Vertical Federated Principal Component Analysis on Feature-Wise Distributed Data

Yiu-ming Cheung[1](✉), Jian Lou[2], and Feng Yu[1]

[1] Department of Computer Science, Hong Kong Baptist University, Kowloon Tong, Hong Kong SAR, China
ymc@comp.hkbu.edu.hk

[2] Guangzhou Institute of Technology, Xidian University, Guangzhou, China
jlou@xidian.edu.cn

**Abstract.** Despite the wide attention to federated learning (FL) in the literature, the existing studies mostly focus on supervised federated learning under the horizontally partitioned local dataset setting. This paper will study the unsupervised FL under the vertically partitioned dataset setting. Accordingly, we propose the vertically dataset partitioned federated principal component analysis (VFedPCA) method, which reduces the dimensionality across the joint datasets over all the clients and extracts the principal component feature information for downstream data analysis. VFedPCA features efficient local computation, communication efficiency, and privacy-preserving. Further, we study two communication topologies. The first is a server-client topology where a semi-trusted server coordinates the federated training, while the second is the fully-decentralized topology which eliminates the requirement of the server by allowing clients to communicate with their neighbors. Extensive experiments conducted on real-world datasets justify the efficacy of VFedPCA under vertical partitioned FL setting.

**Keywords:** Principal component analysis · Federated learning · Vertical distributed data

## 1 Introduction

Federated learning (FL) [15] has been receiving increasing attention in the literature, which enables collaborative machine learning in a communication-efficient and privacy-preserving way. FL provides a general-purpose collaborative learning framework, which consists of a coordinating central server and multiple participating clients with their local datasets, e.g., organizations (cross-silo setting) or devices (cross-device setting). More recent FL methods also propose fully-decentralized learning, where clients directly communicate with their neighbors. It eliminates the need of the coordinating server which can sometimes expose security and privacy risks to the collaborative training [26]. During the FL training, the raw local datasets of all clients are kept locally and restricted to be exchanged or transferred over network for privacy-preserving purpose. Instead, it suffices to communicate only intermediate training variables (e.g., local gradients or local model parameters). Due to its generality and superiority in the privacy-preserving aspect, FL paves its way to a growing number of application areas. Nevertheless, most existing FL methods focus almost exclusively on the supervised learning problems under the horizontally distributed local dataset setting. As far as we know, unsupervised FL learning under the settings of vertically distributed datasets has yet to be well explored.

In fact, the vertically partitioned local dataset setting is common in many practical applications [16]. Under this setting, different clients hold a local partition of features, instead of samples as in the horizontally partitioned setting. It naturally arises when the features/attributes describing the same group of entities are collected and stored among multiple clients. For example, the financial features of a person can split among multiple financial companies s/he has dealt with, e.g., banks, credit card companies, stock market. Similar to the horizontal FL setting, it is important to collaboratively train the model based on all clients' data partition to maximize the global model performance. For example, when a bank refers to a machine learning (ML) model for deciding whether to grant a loan application of a customer, it is ideal for the model training to take account into all the financial records of the customer, not only the transactions history held by this bank. It is apparent from the example that raw local datasets should not be exchanged because the vertically partitioned datasets can contain sensitive information, i.e., the financial status of the customer. Furthermore, unsupervised learning is practically appealing because it need not the labels for model training. The labels can be expensive and time-consuming to mark, and even require domain expertise [13,19]. The label can also contain sensitive information, which will incur privacy leakage if not handled properly, especially under the vertically partitioned setting where the labels need to be shared among all clients [5,8].

Under the circumstances, this paper will concentrate on principal component analysis (PCA), which is one of the most fundamental and useful unsupervised learning task with multiple clients holding the vertically partitioned datasets under the FL framework. PCA plays a pivotal role in high-dimensional data analysis by extracting principal components, which are uncorrelated and ordered, with the leading ones retaining most of the data variations [18]. Very recently, federated PCA is studied on the sample-wise distributed (also known as horizontally distributed) data [9,10]. However, to the best of our knowledge, there is no federated PCA tailored to the feature-wise distributed (also known as vertically distributed) data. The two data distribution settings, though seemed subtle, has fundamental difference as summarized in the review [23]. To this end, we propose a vertically partitioned federated PCA method, abbreviated VFedPCA, which features computational efficiency, communication efficiency and privacy-preserving. In VFedPCA, clients keep their datasets local and only requires the communication of model parameters. Such model exchange suffices to occur periodically to reduce communication overhead. Within each communication round, clients run the computationally efficient local power iteration [1,11,17], and the warm-start power iteration method can further improve performance. Furthermore, we consider the relationship between each independent subset and the full set based on the weight ratio of eigenvalue. Subsequently, we supplement the weight scaling method to further improve the accuracy and performance of the algorithm. In addition, we consider both of two settings: (1) a centralized server coordinates the learning, and (2) the fully decentralized setting which eliminates the need of the server. In summary, the main contributions of this paper are below:

1. We first attempt to study PCA under the vertically-partitioned FL setting.
2. We propose VFedPCA, featuring computational efficiency, communication efficiency, and privacy-preserving. VFedPCA comes with two variants: one requires a central server for coordination and the other performs fully decentralized learning where a client communicates directly with its neighbors.
3. Extensive experiments on real datasets justify the favorable features of VFedPCA under the vertically-partitioned FL setting, while maintaining accuracy similar to unseparated counterpart dataset.

## 2 Notations and Background

### 2.1 Notation

We use boldfaced lower-case letters, e.g., $\mathbf{x}$, to denote vectors and boldfaced upper-case letters, e.g., $\mathbf{X}$, to denote matrix. The superscript is associated with the number of iterations, e.g., $\mathbf{X}^t$ denotes the decision variable at iteration $t$, while the subscript is associated with indices of different parties, e.g., the data matrix $\mathbf{X}_i \in \Re^{n \times p_i}$ denotes the i-th party that has $\mathbf{p}_i$ variables (i.e., features) and $\mathbf{n}$ samples (i.e., users). We use $\mathbf{X}^*$ to denote a dual space and $\|\mathbf{x}\|$ denotes the standard Euclidean norm for the vector $\|\mathbf{x}\|$. For image dataset, the data matrix $\mathbf{X}_i \in \Re^{n \times m \times m}$ is the i-th party's input dimension, where $m$ denotes the pixel size. The summary of the used symbols in this paper is shown in Table 1.

**Table 1.** Summary of frequently used notation

| Notation | Description | Notation | Description |
|---|---|---|---|
| $n$ | Number of samples | $m$ | Number of features |
| $p$ | Partitions | $l$ | Local iterations |
| $t$ | Federated communications | $f_i$ | The features of the i-th party |
| $s_i$ | The samples of the i-th party | $\boldsymbol{\alpha}_i$ | The eigenvalue of the i-th party |
| $\mathbf{a}_i$ | The eigenvector of the i-th party | $\boldsymbol{\omega}_i$ | The weight of the i-th party |
| $\mathbf{u}$ | The federated eigenvector | $\mathbf{u}_G$ | The gobal eigenvector |

### 2.2 PCA and Power Iteration

In this subsection, we describe the basic PCA setup and present the centralized power iteration. Let $\mathbf{X} \in \Re^{m\times n}$ be the data matrix. The goal of PCA is to find the top eigenvectors of the symmetric positive semidefinite (PSD) matrix $\mathbf{A} = \frac{1}{n}\mathbf{X}\mathbf{X}^\top \in \Re^{m\times m}$, which is the covariance matrix. We assume that the target matrix A has eigenvalues $1 \geq \boldsymbol{\alpha}_1 \geq \boldsymbol{\alpha}_2 \cdots \geq \boldsymbol{\alpha}_m \geq 0$ with corresponding normalized eigenvectors $\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_m$, then $\mathbf{A}\mathbf{a} = \boldsymbol{\alpha}\mathbf{a}$. In general, $\mathbf{A}^k\mathbf{a} = \boldsymbol{\alpha}^k\mathbf{a}$ for all $k = 1, ..., m$, which is the basis of the power iteration method. Let $\boldsymbol{\alpha}^{(0)}$ be an approximation to an eigenvector of $\mathbf{A}$, and $\mathbf{a}^{(0)}$ as an initial vector. The power method [21] estimates the top eigenvector by repeatedly applying the update step below:

$$\boldsymbol{\omega} = \mathbf{A}\mathbf{a}^{(k-1)}, \mathbf{a}^{(k)} = \frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|}, \tag{1}$$

where at each iteration, $\mathbf{a}^{(k)}$ will get closer to the top principal eigenvector $\mathbf{a}_1$. When $\mathbf{k} \to \infty$, $\mathbf{a}^{(k)}$ will converge to $\mathbf{a}_1$.

Recently, [17] has proposed a distributed SVD algorithm based on the local power iteration, which incorporates the periodic communication for communication-efficiency. The LocalPower method can save communication by $l$ times without much impact on the accuracy. To guarantee the convergence, they partition the data matrix $\boldsymbol{X}$ randomly. They assume the local correlation matrix $\mathbf{A}_i = \frac{1}{s_i}\mathbf{X}_i\mathbf{X}_i^\top$ is a good approximation to the global correlation matrix $\mathbf{A} = \frac{1}{n}\mathbf{X}\mathbf{X}^\top$, which is equivalent to $\|\mathbf{A} - \mathbf{A}_i\| \leq \rho\|\mathbf{A}\|$, where $\mathbf{A}_i \in \Re^{s_i\times m}$ is the i-th partition, $\rho$ bounds the difference between $\mathbf{A}_i$ and $\mathbf{A}$, which is assumed as small as $\epsilon$ . However, our setting allows all parties to independently calculate the principal component vector based on the basic PCA method, and derive the federated principal component vector in the global center according to the weights of the parties, which can reduce the complexity of communication calculations and ensure the privacy of participants.

### 2.3 Federated Learning

FL is data-private collaborative learning, where multiple clients jointly train a global ML model with the help of a central coordination server. The goal of FL is achieved through the aggregation of intermediate learning variables, while the raw data of each customer is stored locally without being exchanged or transferred. According to different data partition structures, the distribution can be generally categorized into two regimes, namely horizontally and vertically partitioned data [7]. Currently, there is relatively less attention paid to the vertically partitioned data. Most of the existing cross-silo FL work is based on the supervised datasets, including trees [4], linear and logistic regression [20,25], and neural networks [15]. These supervised FL works rely on the labels, which are expensive to acquire and require domain expertise. For example, diabetes patients may wish to contribute to the FL with their everyday health monitor data like glucose level and blood press. Since patients lack advanced medical expertise, their local data are often unlabeled without a medical expert's evaluation. As far as we know, the current work on the unlabelled data and vertical learning has yet to be explored. The only existing related papers focus on the semi-supervised FL [12] and weakly-supervised FL [14], respectively.

## 3 Federated-PCA on Privacy-Preserving Vertical-Partitioned Data

In this section, we first formulate the problem of federated PCA on the vertically partitioned dataset. Then, we propose the VFedPCA algorithm to collaboratively extract principal components with the participation of all clients, while ensuring privacy by avoiding sharing of raw local datasets. Our method devises the local power iteration for efficient local computation, along with periodic communication for better communication efficiency.

In particular, we consider two types of communication topologies. The first is the server-clients topology, which follows the most existing FL methods by introducing a semi-trusted server to coordinate the training. The second is the fully-decentralized topology, where the clients communicate in a peer-to-peer manner with their neighbors. It eliminates the need of the server, which itself can be malicious and sometimes considered unpractical provided that such a semi-trusted server exists.

### 3.1 Problem Formulation

Let $\mathbf{X} \in \Re^{m \times n}$ be the data matrix, which have $m$ features and $n$ samples, we partition $\mathbf{X}$ into $p$ clients as $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \cdots \mathbf{X}_p]^\top$, where $\mathbf{X}_i \in \Re^{f_i \times n}$ contains $f_i$ features of $\mathbf{X}$ and $\sum_{i=1}^{p} f_i = m$. Let $\mathbf{S} = \frac{1}{m}\mathbf{X}^\top \mathbf{X} \in \Re^{n \times n}$. Let $\mathbf{Z} = \mathbf{X}\mathbf{U}^\top \in \Re^{m}$, which can be considered as the coordinate of the projection along the direction given by $\mathbf{U}$. Note that $\mathbf{Var}(\mathbf{Z}) = \mathbf{U}^\top \mathbf{S} \mathbf{U}$, where $\mathbf{S}$ is the covariance matrix. Our purpose is to find the leading k-dimensional subspace such that the projection of the data onto the subspace has the largest variance. The problem could be formulated as follows:

$$\max_{\mathbf{U} \in \mathbb{R}^{m \times k}, \mathbf{U}^\top \mathbf{U} = \mathbf{I}} \|\mathbf{U}^\top \mathbf{S} \mathbf{U}\| = \boldsymbol{\alpha}_G \mathbf{U}_G^\top \mathbf{U}_G = \boldsymbol{\alpha}_G, \tag{2}$$

where $\boldsymbol{\alpha}_G$ are the leading eigenvalues of $\mathbf{S}$ and $\mathbf{U}_G$ are the eigenvectors corresponding to $\boldsymbol{\alpha}_G$. Our aim is to minimize the distance between the global eigenvector $\mathbf{U}_G$ that is computed as if all features were centralized together, and each client's $\mathbf{U}_i$, as follows

$$\min_{\mathbf{U}_G, \mathbf{U}_i \in \Re^{m \times k}} dist(\mathbf{U}_G, \mathbf{U}_i), \text{ for all } i = 1, ..., p. \tag{3}$$

We define the squared $\boldsymbol{\chi}$-distance [3] as the distance between $\mathbf{U}_G$ and $\mathbf{U}_i$ by:

$$dist(\mathbf{U}_G, \mathbf{U}_i) := \sum_{j=1}^{m} \sum_{h=1}^{k} \frac{((\mathbf{U}_G)_{j,h} - (\mathbf{U}_i)_{j,h})^2}{(\mathbf{U}_G)_{j,h} + (\mathbf{U}_i)_{j,h}}, \tag{4}$$

where $\mathbf{U}_G$ is the leading eigenvectors of the global covariance matrix $\mathbf{S}$, and $\mathbf{U}_i$ is the leading eigenvectors of each client's (say i-th) local covariance matrix.

## 3.2 Local Power Method

Let us consider the data partition among $p$ clients. In each client, we use power iteration algorithm (also known as the power method) to produce the greatest (in absolute value) eigenvalue of $\mathbf{A}_i = \frac{1}{f_i}\mathbf{x}_i^\top \mathbf{x}_i$, and a nonzero vector $\mathbf{a}_i$, which is a corresponding eigenvector of $\boldsymbol{\alpha}_i$, i.e. $\mathbf{A}_i\mathbf{a}_i = \boldsymbol{\alpha}_i\mathbf{a}_i$. For $l = 1, 2 \cdots L$, each client will compute locally until convergence by

$$\mathbf{a}_i^{l+1} = \frac{\mathbf{A}_i\mathbf{a}_i^l}{\|\mathbf{A}_i\mathbf{a}_i^l\|}, \tag{5}$$

where the vector $\mathbf{a}_i^l$ is multiplied by the matrix $\mathbf{A}_i$ and normalized at every iteration. Throughout the paper, we use $l \in [L]$ to denote local iterations and reserve $t \in [T]$ to denote global rounds.

If $\mathbf{a}_i^l$ is an eigenvector of $\mathbf{A}_i$, its corresponding eigenvalue is given by

$$\boldsymbol{\alpha}_i^l = \frac{\mathbf{A}_i(\mathbf{a}_i^l)^\top \mathbf{a}_i^l}{(\mathbf{a}_i^l)^\top \mathbf{a}_i^l}, \tag{6}$$

where $\mathbf{a}_i^l$ and $\boldsymbol{\alpha}_i^l$ represent the largest eigenvector and eigenvalue of the i-th client, respectively.

## 3.3 Federated Communication

Each client uploads the eigenvector $\mathbf{a}_i^t$ and the eigenvalue $\boldsymbol{\alpha}_i^t$ to the central sever. First, the server computes the weight $\boldsymbol{\omega}_i$ of each client. Then, the server merges all clients' results and computes the federated eigenvector $\mathbf{u}^t$

$$\boldsymbol{\omega}_i^t = \frac{\boldsymbol{\alpha}_i^t}{\sum_{i=1}^p \boldsymbol{\alpha}_i^t}, \mathbf{u}^t = \boldsymbol{\omega}_1^t\mathbf{a}_1^t + \boldsymbol{\omega}_2^t\mathbf{a}_2^t + \cdots \boldsymbol{\omega}_p^t\mathbf{a}_p^t, \tag{7}$$

where $\boldsymbol{\omega}_i^t$ is the weight of the i-th client and $\mathbf{u}^t$ is the shared projection feature vector by merging all clients' $\mathbf{a}_i^t$'s.

## 3.4 Sever-Clients Architecture

All clients share parameters with the help of the central coordination server. In our setting, this third-client coordinator can be trusted, which means that it will honestly conduct the designated functionality and will not attempt to breach the raw data of any clients. After adjusting the new parameters, this federated-parameter is returned to all clients. Then, each client calculates locally based on this new federated-parameter. Moreover, this "communication-and-local-computation" cycle is iterative. The framework of this model is shown in

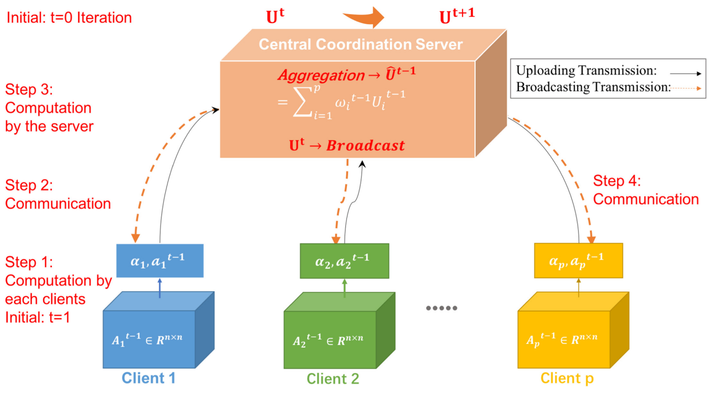
Initial: t=0 Iteration
$U^t$
$U^{t+1}$
Central Coordination Server
Aggregation → $\hat{U}^{t-1} = \sum_{i=1}^{p} \omega_i^{t-1} U_i^{t-1}$
$U^t$ → Broadcast
Uploading Transmission:
Broadcasting Transmission:
Step 3: Computation by the server
Step 2: Communication
Step 4: Communication
Step 1: Computation by each clients
Initial: t=1
$\alpha_1, a_1^{t-1}$
$\alpha_2, a_2^{t-1}$
$\alpha_p, a_p^{t-1}$
$A_1^{t-1} \in R^{n\times n}$
$A_2^{t-1} \in R^{n\times n}$
$A_p^{t-1} \in R^{n\times n}$
Client 1
Client 2
Client p

**Fig. 1.** The framework of VFedPCA with central server.

Fig. 1. The algorithm is summarized in Algorithm 1. The key steps of this method are as follows:

***Local Training:*** All clients perform local power method independently. Each client first calculates the covariance matrix $\mathbf{A}$. Subsequently, the largest eigenvector $\mathbf{a}$ and eigenvalue $\boldsymbol{\alpha}$ are calculated (see steps 4–10 of Algorithm 1);

***Model Integration:*** The central server calculates the weight $\boldsymbol{\omega}_i$ occupied by each client in the federated model by receiving the eigenvalue $\boldsymbol{\alpha}_i$ of each client, and then combines the received eigenvector $\mathbf{a}_i, i \in \{1, \cdots p\}$, to derive a federated feature vector $\mathbf{u}^t$, where the communication is cyclical (see steps 12–16 of Algorithm 1);

***Parameters Broadcasting and Updating:*** The central coordination server broadcasts the aggregated parameters $\mathbf{u}^t$ to the $p$ clients. Each client updates local eigenvector with the new returned federated feature vector $\mathbf{u}^t$ (see step 17 of Algorithm 1). The advantage of this model is relatively more efficient, although it relies on the help of the server which can be potentially malicious.

## 3.5 Local Power Iteration with Warm Start

The general power iteration algorithm starts with an initialization vector $\mathbf{a}^{(0)}$, which may be an approximation to the dominant eigenvector or a random unit vector. Here, we initialize the algorithm from a "warm-start" vector $\mathbf{a}^{(0)}$, which is based on the federated vector from the previous communication round. It is natural to reach convergence faster after consecutive iterations, which will reduce the actual running time and improve the performance of the local power iteration algorithm in Sect. 3.2. The algorithm is summarized in Algorithm 2. The main step is as follows:

***Local Training:*** Each node uses the federated feature vector $\mathbf{u}$ as the initialize value to perform local power method, then calculate the k + 1 to 2k eigenvector $\mathbf{a}$ and eigenvalue $\boldsymbol{\alpha}$ and send to the server (see steps 2–8 of Algorithm 2).

**Algorithm 1:** VFedPCA Learning with Central Server

```
1  ⇒ Run on the i-th node
2  Input: Data $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{f_i}\} \in \Re^{n \times f_i}$ belongs to client $i$
3  Output: Federated principal eigenvector $\mathbf{u}$
4  Initial: Let $\overline{\mathbf{x}} = 0$, $\mathbf{a}_i^{(0)} \in \Re^n$ randomly
5  for $l$=1 to $L$ do
6      while each client $i, i = \{1, 2 \cdots p\}$ do
7          $\mathbf{a}_i^{l+1} = \frac{\mathbf{A}_i \mathbf{a}_i^l}{\|\mathbf{A}_i \mathbf{a}_i^l\|}$, $\boldsymbol{\alpha}_i^l = \frac{\mathbf{A}_i (\mathbf{a}_i^l)^\top \mathbf{a}_i^l}{(\mathbf{a}_i^l)^\top \mathbf{a}_i^l}$
8      end
9      Send $(\boldsymbol{\alpha}_i^l, \mathbf{a}_i^l)$ to the central server.
10 end
11 ⇒ Run on central coordination server
12 for $t$=1 to $T$ do
13     Receive $(\boldsymbol{\alpha}_i^t, \mathbf{a}_i^t) = (\boldsymbol{\alpha}_i^l, \mathbf{a}_i^l)$ from n clients
14     for $i = 1$ to $p$ do
15         $\boldsymbol{\omega}_i^t \leftarrow \boldsymbol{\alpha}_i^t$ and $\mathbf{u}^t \leftarrow Merge(\boldsymbol{\omega}_i^t, \mathbf{a}_i^t)$ by eq.(7)
16     end
17     Broadcast and update eigenvector $\mathbf{u}^t \leftarrow \mathbf{u}^{t+1}$
18 end
```

This method considers the use of an adaptive loop idea which allows communication between clients to reduce the error to a certain extent, especially for the case where the error between the general federated result $\mathbf{u}$ and the global result $\mathbf{u}_G$ is large.

## 3.6 Weight Scaling Method

The federated result may be sometimes not beneficial to all clients. In general, the client with the larger eigenvalue has a greater influence on the federated result, which hints a natural intuition that following the direction of the clients with larger eigenvalues tends to reach the global consensus faster and costs less communication rounds. Inspired by this intuition, we introduce a weight scaling factor to further improve the federation communication. The improved weight scaled federated average is formulated by

$$\begin{aligned}\mathbf{u}^t &= (1+\eta_1^t)\boldsymbol{\omega}_1^t\mathbf{a}_1^t + \cdots + (1+\eta_{\lceil p/2\rceil}^t)\boldsymbol{\omega}_1^t\mathbf{a}_{\lceil p/2\rceil}^t \\ &\quad + (1-\eta_{\lceil p/2\rceil+1}^t)\boldsymbol{\omega}_1^t\mathbf{a}_{\lceil p/2\rceil+1}^t \cdots (1-\eta_p^t)\boldsymbol{\omega}_p^t\mathbf{a}_p^t,\end{aligned} \tag{8}$$

where $\sum_{i=1}^{\lceil p/2\rceil} \eta_i^t = \sum_{i=\lceil p/2\rceil+1}^{p} \eta_i^t$. In brief, we gradually increase the impact of the first half of the clients with larger eigenvalues, while further decrease the impact of the clients of the second half with smaller eigenvalues. The algorithm is summarized in Algorithm 3. The main step is as follows:

***Model Integration:*** The central server calculates the weight $\boldsymbol{\omega}_i$ occupied by each client in the federated model based on the received eigenvalue $\boldsymbol{\alpha}_i$ of each

**Algorithm 2:** Local Power Iteration with Warm Start

```
1 ⇒ Run on the i-th node
2 Initial: Let $\mathbf{a}_i^{(0)} = \mathbf{u}$ -warm-start
3 for l=1 to L do
4     while each client $i, i = \{1, 2 \cdots p\}$ do
5         $\mathbf{a}_i^{l+1} = \frac{\mathbf{A}_i \mathbf{a}_i^l}{\|\mathbf{A}_i \mathbf{a}_i^l\|}$ $\boldsymbol{\alpha}_i^l = \frac{\mathbf{A}_i (\mathbf{a}_i^l)^\top \mathbf{a}_i^l}{(\mathbf{a}_i^l)^\top \mathbf{a}_i^l}$
6     end
7     Send $(\boldsymbol{\alpha}_i^l, \mathbf{a}_i^l)$ to the central server.
8 end
```

**Algorithm 3:** Weight Scaling Method

```
1 ⇒ Run on central collaborative server
2 for t=1 to T do
3     Receive $\alpha_i^t, \mathbf{a}_i^t$ from n clients
4     for $i = 1$ to $p$ do
5         $\boldsymbol{\omega}_i^t \leftarrow \boldsymbol{\alpha}_i^t$ by eq.(7), $\eta_i^t = \eta$
6         $\mathbf{u}^t \leftarrow Merge(\eta_i^t, \boldsymbol{\omega}_i^t, \mathbf{a}_i^t)$ by eq. (8)
7     end
8 end
```

client, and then adds $\eta$ parameter to further adjust the weight scale of each client. Then, it combines the received eigenvector $\mathbf{a}_i, i \in \{1, \cdots p\}$, to derive a federated feature vector $\mathbf{u}^t$ (i.e., steps 2–8 of Algorithm 3). This method is especially suitable for situations where some clients have a larger weight in the federated process.

## 3.7 Fully Decentralized Architecture

In this variant, we eliminate the need of the central server, where all clients only share principal component parameters with each other, and compute the results locally. The framework of this model is shown in Fig. 2. The algorithm is summarized in Algorithm 4. The main step of this method is as follows:

***Parameters Communication and Updating:*** Communication among clients is connected and share intermediate results of the parameters $\boldsymbol{\alpha}_i$ and $\mathbf{a}_i$. Each client obtains the extracted final data based on calculating the weights $\boldsymbol{\omega}_i$'s occupied by all clients and updates locally (see steps 3–5 of Algorithm 4).

This model circumvents the need for third-client agencies to help participants further reduce some external risks, especially for two-client scenarios. However, the principal components of all clients need to be calculated independently, and the computing efficiency will be greatly affected.

### 3.8 Complexity Analysis

In each client, let $L$ be the total number of local iterations, which only depends on the eigen-gap $\Delta$ after full passes over the data between the top two eigenvalues of $\mathbf{A}_i$. In Algorithm 1, let $T$ be the total number of federated aggregations executed by the server.

**Theorem 1** ***(Local Iteration Complexity).*** *After $O(\frac{1}{\Delta}\log\frac{n}{\epsilon})$ steps, the local power method can achieve $\epsilon$-accuracy.*

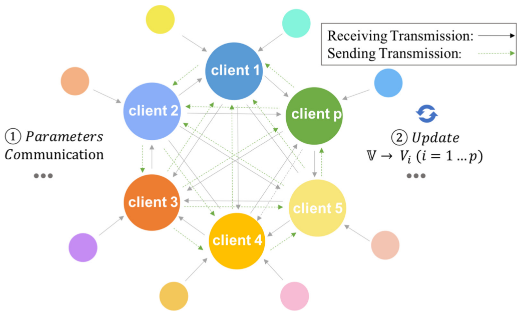

**Fig. 2.** The framework of fully decentralized (*peer-to-peer*) VFPCA learning

**Algorithm 4:** Fully decentralized VFedPCA learning

```
1 ⇒ Run on the i-th node
2 Parameters communication between connected clients
3 for $i = 1$ to $p$ do
4     $\omega_i^l \leftarrow \alpha_i^l$, $\mathbf{a}^l \leftarrow Merge(\omega_i^l, \mathbf{a}_i^l)$
5     Update eigenvector $\mathbf{a}_i^l \leftarrow \mathbf{a}^l$
6 end
```

*Remark 1.* We show the proof in the appendix[1]. According to Theorem 1, the normalized iterate $\frac{\mathbf{A}_i\mathbf{a}_i^l}{\|\mathbf{A}_i\mathbf{a}_i^l\|}$ is an $\epsilon$-accurate estimate of the top principal component. Hence, the total local power iterations complexity is $O(\frac{1}{\Delta}\log\frac{n}{\epsilon})$.

## 4 Experimental Results

This section will empirically evaluate the proposed method. In the experiments, we utilized four groups of real-world datasets: 1) structured datasets from different domains [6]; 2) medical image dataset [24]; 3) Face image dataset [2]; 4) Gait

[1] Link to Appendix: https://www.comp.hkbu.edu.hk/~ymc/papers/conference/wise21/appendix.pdf.

image dataset [22]. The feature and sample size information of all datasets are summarized in Table 2. In addition, illustrative images of the image dataset are shown in Figs. 3, 4 and 5, respectively. In all experiments, we measure the communication cost by calculating the bits of all the variables transmitted throughout the algorithm execution. As a preliminary study, we set T = 1, i.e., one-shot communication between the clients and the server, in our experiments, unless otherwise stated.

## 4.1 Experiment on Structured Dataset

**Semi-synthetic Datasets.** We use 8 structured datasets from different domains, which are the real data that are publicly available at the UCI Machine Learning Repository [6]. We compare the communication cost and estimation error by the different settings of $p$ and $l$ based on the feature-wise setting. We change the number of partitions $p = 3,5,10$, $p = 10,50,100$, and the local iterations $l = 5,10,20$, $l = 50,100,200$, and we vary the number of communication period to

**Table 2.** Summary of datasets

| Dataset | # of features | # of samples |
|---|---|---|
| College | 9 | 990 |
| GlaucomaM | 54 | 196 |
| PimaIndiansDiabetes | 33 | 351 |
| Musk | 166 | 476 |
| Vehicle | 18 | 846 |
| Sonar | 60 | 208 |
| Swarm | 2400 | 2000 |
| TCGA | 20500 | 800 |
| DeepLesion | 262144 | 100 |
| Face | 10000 | 225 |
| CASIA | 65536 | 240 |

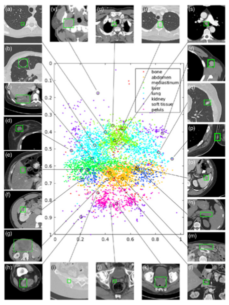

**Fig. 3.** The DeepLesion dataset.

**Fig. 4.** The YaleFace dataset

**Fig. 5.** The CASIAGait database.

$t = 5$, $t = 10$. The number of each client's features is split and configured according to the different settings of $p$.

**Communication Cost and Estimation Error** In this part, we simulate the communication between the clients and the server on a single machine, where the CPU time are wall-clock time.

– **The effect of local feature size.** In Fig. 6, we fix $l = 10$ and $l = 100$ but change the number of involved clients $p = 3, 5, 10$ and $p = 10, 50, 100$. Then, we plot the distance error and the running time cost between the global eigenvector and the federated eigenvector with the different partitions $p$. In all experiments, compared with the un-communicated situation, the distance error after the communication has been significantly reduced and convergence has been achieved. It can be observed that the smaller $p$ will lead to less communication cost.

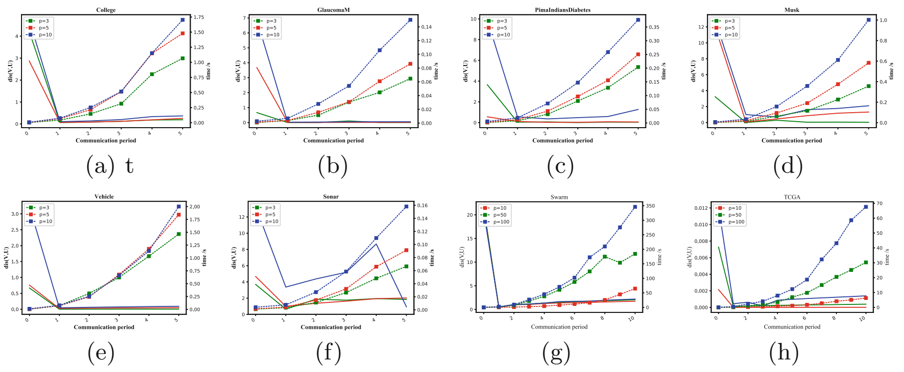

**Fig. 6.** The distance error (left vertical axis) and the running time (right vertical axis) of Algorithm 1 with respect to the number of parties on structured datasets: (a) College, (b) GlaucomaM, (c) PimaIndiansDiabetes, (d) Musk, (e) Vehicle, (f) Sonar, (g) Swarm, and (h) TCGA, where $l = 10$, $p = 3, 5, 10$ for (a)–(f), and $l = 100$, $p = 10, 50, 100$ for (g)–(h).

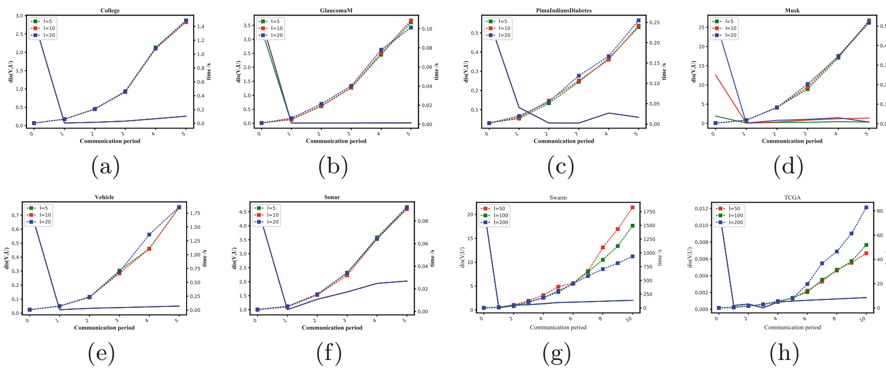

**Fig. 7.** The distance error (left vertical axis) and the running time (right vertical axis) of Algorithm 1 with respect to the number of local iterations on structured datasets: (a) College, (b) GlaucomaM, (c) PimaIndiansDiabetes, (d) Musk, (e) Vehicle, (f) Sonar, and (g) Swarm (h) TCGA; For (a)–(f), $p = 5$, $l = 5$, 10, 20; for (g)–(h), $p = 50$, 100, $l = 50$, 100, 200.

- **The effect of local iterations.** In Fig. 7, we fix $p = 5$ and $p = 100$, but change the number of local power iterations to $l = 5$, 10, 20 and $l = 50$, 100, 200. Then, we plot the distance error and the running time cost between the global eigenvector and the federated eigenvector with the different local iterations $l$. All experimental results show that, after the communication, the error eventually decreases and the algorithm converges. It can also be seen that the result will remain stable when $l$ is up to 20.
- **The effect of warm-start power iterations.** In Fig. 8, we fix $p = 5$ and local power iterations $l = 100$. Then, we plot the distance error between the global eigenvector and the federated eigenvector after communications $t = 5$. Experimental results show the trend that the error further decreases and converges.
- **The effect of $\eta$.** In Fig. 8, we fix $p = 5$ and local power iterations $l = 100$. Then, we plot the distance error between the global eigenvector and the federated eigenvector after iterations $t = 5$, 10. Experimental results show the trend that the error further decreases and converges compared with the one without using the adjustment factor $\eta$.

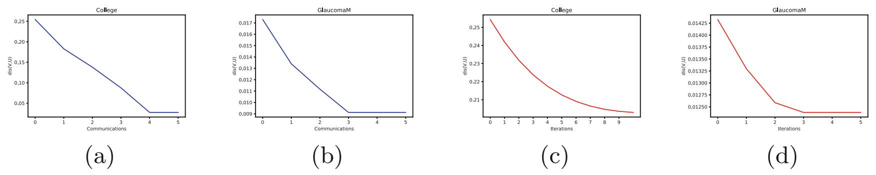

**Fig. 8.** The distance error of Algorithms 2 and 3 with respect to the number of local iterations on structured datasets: (a) College, (b) GlaucomaM, where $p = 5$, $l = 100$.

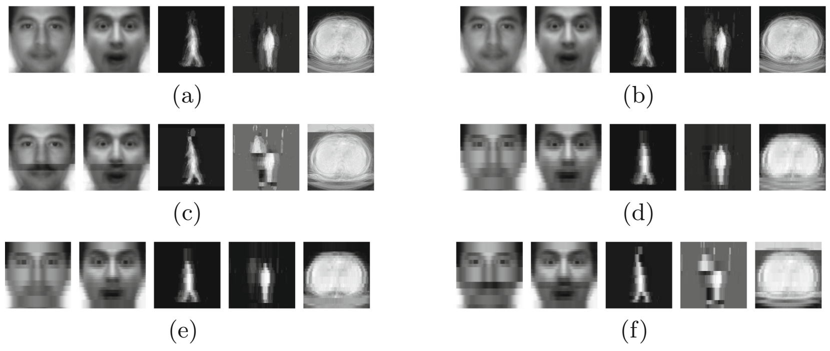

**Fig. 9.** The final results of comparative experiment on image datasets: YaleFace (center-light and surprised), CasiaGait (sequence 1 and 10) and DeepLesion, where (a) PCA on the un-split data, (b) VFedPCA on the split data, (c) PCA on the isolated data. Image segmentation (k = 10) results: (d) PCA on the un-split data, (e) VFedPCA on the split data, (f) PCA on the isolated data.

### 4.2 Case Studies

**Medical Image Dataset.** The DeepLesion dataset [24] is a CT slices collection from 4427 unique patients, which contains a variety of lesions (e.g., lung nodules, liver lesions). We selected 100 samples from the dataset and each image is normalized to an $512 \times 512$ gray image. We set $p = 8$ clients, re-split the features, and assign them equally to each client. We used the commonly clustering method: k-means, with the number $k = 20$ of clusters.

**Yale Face Dataset.** We use the Yale Face Dataset [2], which contains 165 grayscale images of 15 subjects. Each subject configures 11 different facial expressions and $n = 15$ samples for each facial expression, where each face image is normalized to a $100 \times 100$ gray image and we set $p = 10$ clients. We use the k-means with $k = 10$.

**Gait Estimation.** The CASIA is a gait database [22] for gait recognition, including 20 persons. We have image sequences, 4 sequences for each of the three directions and $n = 20$ samples for each direction, where each image is resized to the $256 \times 256$ scale. We set $p = 16$ clients. We used the k-means with $k = 10$.

**Comparative Results.** We first perform PCA on image datasets. Figure 9 (a)(b)(c) show that the final images after federating is almost the same as the final images after un-split image data. Then, to further verify that the final image results extract useful features, we also perform the clustering experiment on the image dataset. Figure 9 (d)(e)(f) show the clustering result (also known as image segmentation) after using the federated PCA method, which also shows improvement over the isolated PCA (i.e., independently run PCA on splitted local datasets). It will help each client to further perform local training tasks.

## 5 Concluding Remarks

In this paper, we have proposed VFedPCA algorithm, which can obtain a collaborative model that improves over the local models learned separately by each client. Through extensive comparative studies on various tasks, we have verified that the collaborative model achieves comparative accuracy with the centralized model as if the dataset were un-splitted. In addition, we propose two strategies, i.e., the local power iteration warm start method and the weight scaling method, to further improve the performance and accuracy of the model. In terms of the federated communication, the full decentralized model has lower communication cost and more flexible application value.

From the practical point of view, it is not uncommon that there exists a non-linear relationship between various data, especially for the image data, most of which possess a non-linear relationship. Under the circumstances, the PCA method which is essentially linear is inapplicable to find an appropriate representative direction. In the future, we will therefore study the vertical federated kernel PCA learning on datasets with the different types of non-linearity.

**Acknowledgment.** This work was supported by HKBU AI-Info Communication Study (AIS) Scheme with Project No. AIS 21-22/03.

## References

1. Balcan, M.-F., Du, S.S., Wang, Y., Yu, A.W.: An improved gap-dependency analysis of the noisy power method. In: Conference on Learning Theory, pp. 284–309. PMLR (2016)
2. Belhumeur, P.N., Hespanha, J.P., Kriegman, D.J.: Eigenfaces vs. fisherfaces: recognition using class specific linear projection. IEEE Trans. Pattern Anal. Mach. Intell. **19**(7), 711–720 (1997)
3. Cha, S.: Comprehensive survey on distance/similarity measures between probability density functions (2007)
4. Cheng, K., Fan, T., Jin, Y., Liu, Y., Chen, T., Yang, Q.: SecureBoost: a lossless federated learning framework. arXiv:1901.08755 (2019)
5. Du, W., Han, Y.S., Chen, S.: Privacy-preserving multivariate statistical analysis: linear regression and classification. In: SIAM International Conference on Data Mining. SIAM (2004)
6. Dua, D., Graff, C.: UCI machine learning repository (2017)
7. Fan, J., Wang, D., Wang, K., Zhu, Z.: Distributed estimation of principal eigenspaces. Ann. Stat. **47**(6), 3009 (2019)
8. Gascón, A., et al.: Secure linear regression on vertically partitioned datasets. IACR Cryptol. ePrint Arch. 2016:892 (2016)
9. Grammenos, A., Smith, R.M., Crowcroft, J., Mascolo, C.: Federated principal component analysis. In: NeurIPS (2020)

10. Guo, X., Li, X., Chang, X., Wang, S., Zhang, Z.: Privacy-preserving distributed SVD via federated power. arXiv:2103.00704 (2021)
11. Hardt, M., Price, E.: The noisy power method: a meta algorithm with applications. In: NeurIPS (2014)
12. Jeong, W., Yoon, J., Yang, E., Hwang, S.J.: Federated semi-supervised learning with inter-client consistency. arXiv:2006.12097 (2020)
13. Ji, Z., Zou, X., Huang, T., Wu, S.: Unsupervised few-shot learning via self-supervised training. arXiv:1912.12178 (2019)
14. Jin, Y., Wei, X., Liu, Y., Yang, Q.: Towards utilizing unlabeled data in federated learning: a survey and prospective. arXiv:Learning (2020)
15. Kairouz, P., McMahan, H.B., Avent, B., et al.: Advances and open problems in federated learning. arXiv:1912.04977 (2019)
16. Kargupta, H., Huang, W., Sivakumar, K., Johnson, E.: Distributed clustering using collective principal component analysis. Knowl. Inf. Syst. **3**(4), 422–448 (2001)
17. Li, X., Wang, S., Chen, K., Zhang, Z.: Communication-efficient distributed SVD via local power iterations. arXiv:2002.08014 (2020)
18. Mishra, S.P., et al.: Multivariate statistical data analysis-principal component analysis (PCA). Int. J. Liv. Res. **7**(5), 60–78 (2017)
19. Nian, K., Zhang, H., Tayal, A., Coleman, T., Li, Y.: Auto insurance fraud detection using unsupervised spectral ranking for anomaly. J. Finance Data Sci. **2**(1), 58–75 (2016)
20. Nock, R., et al.: Entity resolution and federated learning get a federated resolution. arXiv:1803.04035 (2018)
21. Saad, Y.: Numerical methods for large eigenvalue problems: revised edition. SIAM (2011)
22. Wang, L., Tan, T., Ning, H., Weiming, H.: Silhouette analysis-based gait recognition for human identification. IEEE Trans. Pattern Anal. Mach. Intell. **25**(12), 1505–1518 (2003)
23. Wu, S.X., Wai, H.-T., Li, L., Scaglione, A.: A review of distributed algorithms for principal component analysis. Proc. IEEE **106**(8), 1321–1340 (2018)
24. Yan, K., Wang, X., Lu, L., Summers, R.M.: Deeplesion: automated mining of large-scale lesion annotations and universal lesion detection with deep learning. J. Med. Imaging **5**(3), 036501 (2018)
25. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Silhouette analysis-based gait recognition for human identification. ACM Trans. Intell. Syst. Technol. (TIST) **10**(2), 1–19 (2019)
26. Zantedeschi, V., Bellet, A., Tommasi, M.: Fully decentralized joint learning of personalized models and collaboration graphs. In: International Conference on Artificial Intelligence and Statistics, pp. 864–874 (2020)