EASTER: Embedding Aggregation-based Heterogeneous Models Training in Vertical Federated Learning

Shuo Wang, Keke Gai, *Senior Member, IEEE*, Jing Yu, *Member, IEEE*, Liehuang Zhu, *Senior Member, IEEE*, Weizhi Meng, *Senior Member, IEEE*, Bin Xiao, *Fellow, IEEE*

Abstract—Vertical Federated Learning (VFL) allows collaborative machine learning without sharing local data. However, existing VFL methods face challenges when dealing with heterogeneous local models among participants, which affects optimization convergence and generalization of participants' local knowledge aggregation. To address this challenge, this paper proposes a novel approach called *Embedding Aggregation-based Heterogeneous Models Training in Vertical Federated Learning* (EASTER). EASTER focuses on aggregating the local embeddings of each participant's knowledge during forward propagation. We propose an embedding protection method based on lightweight blinding factors, which injects the blinding factors into the local embedding of the passive party. However, the passive party does not own the sample labels, so the local model's gradient cannot be calculated locally. To overcome this limitation, we propose a new method in which the active party assists the passive party in computing its local heterogeneous model gradients. Theoretical Analysis and extensive experiments demonstrate that EASTER can simultaneously train multiple heterogeneous models with heterogeneous optimization and outperform some recent methods in model performance. For example, compared with the state-of-the-art method, the model accuracy of EASTER was improved by 4% under the FMNIST and LetNet networks.

Index Terms—Vertical federated learning, embedding aggregates, blinding factor, heterogeneous models

1 Introduction

As the amount of data generated by mobile clients and Web-of-Things (WoT) devices grows, it powers a wide range of machine-learning applications. Mobile clients and WoT devices have always had to share their data with the cloud to help with machine model training. Mobile clients and WoT devices are wary about sharing raw data due to privacy concerns. As a new distributed machine learning paradigm, federated learning (FL) [1]–[3] offers a solution for locally training ML models with data from mobile and WoT devices. Particularly, FL implements a multi-participant collaborative training methodology that maintains data locally, ensuring data security. Two major types of FL that have been paid wide attention in recent years include *Horizontal Federated Learning* (HFL) [4], [5] and *Vertical Federated Learning* (VFL) [6], [7]. Specifically, HFL supports collaborative learning for those participants with the same feature sets even though the space IDs are varied; VFL addresses situations when the training takes place on distinct feature subsets and the same sample space [8]. The feature of the same sample space determines that VFL is a primary option for constructing cross-organization cooperation compared to HFL [9]. We focus on VFL in this work.

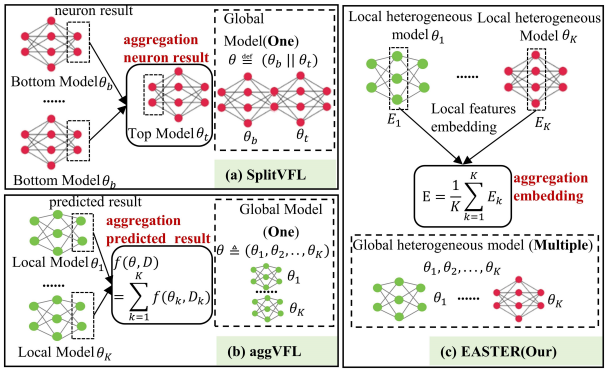


Fig. 1. Typical architectures of VFL. (a): Current SplitVFL architectures; (b): Current aggVFL architectures; (c): The proposed VFL architecture (EASTER).

According to prior research [10], [11], we categorize VFL participants into active and passive parties. The active party is the one who owns the labels and features, while the passive party only owns the features. Current research primarily divides VFL into SplitVFL [12], [13] and aggVFL [14], [15] based on the trainability of the top model. The key

- *Shuo Wang, Keke Gai, and Liehuang Zhu are with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China (e-mails: {3220215214, gaikeke, liehuangz}@bit.edu.cn). Keke Gai is also with the Yangtze Delta Region Academy of Beijing Institute of Technology, Jiaxing, Zhejiang, China (e-mail: gaikeke@bit.edu.cn).*
- *Jing Yu is with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China (E-mail: yujing02@iie.ac.cn).*
- *Weizhi Meng is with the Department of Applied Mathematics and Computer Science, Technical University of Denmark, 2800 Lyngby, Denmark (e-mail: weme@dtu.dk).*
- *Bin Xiao is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China (e-mail: b.xiao@polyu.edu.hk).*

Manuscript received Month XX, 202X; revised Month XX, 20XX.

difference between SplitVFL and aggVFL is that SplitVFL's aggregated top-level model is trainable, but aggVFL's aggregated model is not. Specifically, SplitVFL [16], [17] as shown in Figure 1(a) splits a neural network model into a top and a bottom network. The active party aggregates the outputs of the bottom-level networks to acquire all parties' local knowledge. Meanwhile, the active party utilizes the aggregated neuron results for top-model training. An aggVFL divides a network model into several parts, with each participant holding one part [18]. The active party aggregates the local prediction results of each participant. It implies that most existing VFL research depends on an assumption that participating parties hold consistent local models with the same optimizer [19]. As each participant has different resources and computing power, parties can choose varied local model architectures in most practical cases. For example, some major businesses or companies may have high-performance servers, GPU accelerators, or dedicated deep-learning accelerator cards, while others may just have basic computer resources. For example, multiple medical institutions in the same area may have distinct medical image data that can be used to train a disease prediction model collaboratively. Hospitals in counties might only have basic computing resources, whereas hospitals in cities might have more advanced computing resources. Therefore, in VFL, heterogeneity in computer resources is common. However, the heterogeneous model architecture of participants has been rarely addressed by prior studies [20], [21].

Our investigations find that existing work has rarely addressed heterogeneous VFL issues. Prior work has evidenced that training the same model on varied datasets will output distinct performance in accuracy [22], [23]. Similarly, training different models on one dataset will receive varied accuracy performance [24], [25]. This phenomenon implies that the convergence of the model is influenced by both model heterogeneity and training datasets, even though current existing methods mainly focus on the scenarios with heterogeneous local models rather than covering heterogeneity and the local knowledge information from the bottom-level networks or forward propagation results. Incorporating additional model information during VFL training may degrade the performance of heterogeneous models.

One of the keys to achieving VFL with heterogeneous local models is to reduce additional model information during aggregation. Inspired by prototype learning, heterogeneous neural networks can be divided into representation and decision layers [22], [26], [27]. The representation layer embeds similar features into the same embedding space (class prototype), while the decision layer generates predictions from the embedded information and heterogeneous networks. Therefore, we observe that the embedding results obtained from the representation layer only contain the local knowledge of parties. Prototype learning provides a new perspective for achieving VFL with heterogeneous local models.

To tackle the model heterogeneous challenge in VFL, this paper proposes a method entitled *Embedding Aggregation-based Heterogeneous Models Training in Vertical Federated Learning* (EASTER). Our method aggregates all parties' local embeddings to obtain global embeddings that contain the knowledge from the local training datasets of all parties. These global embeddings are used for training heterogeneous local models by all participant parties. Considering the context of VFL, typically only active parties have access to labeled data and passive parties send intermediate forward propagation results to active parties. Then, the active party assists passive parties in computing the gradient for updating heterogeneous local models, such that it enables the completion of the training of the local model.

The main contributions of our work are summarized as

- We proposed a novel approach (e.g. EASTER) that aggregates local embeddings from multiple heterogeneous parties into a comprehensive global embedding. This method successfully optimizes the training of models across various architectures, achieving higher accuracy regardless of whether the local models are homogeneous or heterogeneous. Moreover, EASTER is capable of simultaneously optimizing multiple heterogeneous models, resulting in several independently optimized models that maintain their distinct characteristics while benefiting from shared, aggregated insights. This capability significantly enhances the applicability and flexibility of VFL systems across diverse computational and data environments.
- Our work develops a secure embedding aggregation scheme that protects the privacy of the local embedding information of participating parties. Our work develops a secure embedding aggregation scheme within the EASTER framework. This scheme meticulously safeguards the privacy of local embedding information from all participating entities. By effectively protecting these embeddings during the aggregation process, our approach ensures that the integrity and confidentiality of participant data are maintained, thereby enhancing the overall security of the VFL system.
- We conduct theoretical analysis and extensive experiments on MNIST, FashionMNIST, and CIFAR-10 datasets to evaluate the EASTER performance. Compared with existing FL methods, EASTER has superior learning performance under heterogeneous local models. For example, compared with the state-of-the-art method, the model accuracy of EASTER was improved by 4% under the FMNIST dataset and LetNet network.

The rest of this paper is organized in the following order. Section 2 presents the preliminaries of our method, essential for understanding the subsequent sections. The system design and our proposed methodology are presented in Section 3 and Section 4, respectively. The experience evaluations in Sections 5. In Section 6, we briefly review the related literature. Finally, we summarize this work in Section 7.

2 Preliminaries

In this section, we provide some preliminary understanding of EASTER. To simplify reading, we present descriptions for some notations in Table 1.

TABLE 1
Notations Table.

Notation	Explanations
θ	Global model parameters
θ_b	Bottom model parameters
θ_t	Top model parameters
θ_k	l_kth party's heterogeneous model parameters
E_{l_k}	Local embedding value generated by party l_k
$[E_{l_k}]$	Blinded local embedding value generated by party l_k
E	Global embedding
l_k	The kth participant, and the special l_0 represents the active party
N	The number of sample space **ID**
K	the number of all passive parties
C	the number of all parties
$\theta_{l_k}^t$	Heterogeneous model parameters obtained by l_kth party in tth epoch
$f_{l_k}(\cdot)$	The l_kth loss function
D	Data sets participating in training
R_{l_k}	Local prediction value generated by passive party l_k
$\mathbb{G}$	The cyclic group of prime order p
SK_{l_k}	The privacy key generated by the l_kth passive party
PK_{l_k}	The public key generated by the l_kth passive party
$H(\cdot)$	The collusion-resistant secure hash function
r_{l_k}	Blinding factor
θ_k^*	The converged model parameters

2.1 Vertical Federated Learning

VFL is a distributed machine-learning architecture, which implements multi-clients collaboratively training a model with sample alignment and features union. In VFL, a single party cannot train a complete model since labels and features are stored in multiple parties. The goal of VFL is to train a global model θ with multiple participants to minimize loss values without revealing local data.

The training process of VFL is as follows.

1) Each passive party uses local data features to perform local model forward training to obtain intermediate results, and uploads the intermediate results to the active party.
2) The active party aggregates the intermediate results and calculates the global model loss.
3) Each passive party downloads corresponding gradients from the active party to update their local model.

The training problem of VFL can be formulated as follows.

$$\min_{\theta \in \mathbb{R}^d} \ell(\theta, D) :\triangleq \frac{1}{N} \sum_{j=1}^{N} f\left(\theta; \mathbf{x}_j, y_j\right) \tag{1}$$

In Eq. (1), N is the total number of data samples. $\mathbf{x}_j$ denote the union of the data features of jth row for all passive parties. y_j is the jth labels of sample. $\theta \in \mathbb{R}^d$ denotes the global parameters. $f(\cdot)$ is the loss function.

2.2 Diffe-Hellman Key Exchange

In this paper, we make utilization of the *Decisional Diffie-Hellman* (DDH) key exchange to generate blinding factors that further protect the passive party's local embedding value. Let $\mathbb{G}$ be a cyclic group of prime order p, and g is a generator of $\mathbb{G}$. We assume the DDH problem is hard [28]. Passive parties can securely share a secret as follows: The one passive party obtains a secret a and sets its public value to $g^a \in \mathbb{G}$. The other passive party obtains his secret b and sets his public value to $g^b \in \mathbb{G}$. The one passive party and the other passive party exchange the public values and raise the other party's value to their secret, i.e., $g^{ab} = (g^a)^b = (g^b)^a$. If DDH is hard, only the two passive parties know the shared secret.

3 Model Design

3.1 Design Goals

To address the heterogeneous model challenge in VFL, we aim to design a VFL with multiple heterogeneous models. The main design goals of EASTER are as follows:

- **Heterogeneous Parties Collaboration**. EASTER should be able to achieve collaborative training of participants with heterogeneous models. That is, EASTER can aggregate heterogeneous model information as little as possible during the training process to achieve collaborative training of participants with heterogeneous models.
- **Multiple Models Training**. EASTER should be able to support training multiple heterogeneous models simultaneously. In one training, we should be able to obtain multiple optimized heterogeneous models. If there are currently three participants with heterogeneous models participating in VFL training, then at the end of one collaborative training, EASTER should be able to obtain three optimized heterogeneous models.
- **Privacy Protection**. EASTER should safeguard passive parties' local privacy and security. Participants cannot infer the original features from the obtained embeddings.

3.2 Problem Statement

Under the EASTER setting, we consider K passive parties and one active party. We define C as the number of all parties, and $C = K + 1$. This paper focuses on the training process of VFL so we assume that both the active party and K passive parties have the same sample space **ID**. We give the training set $\{\mathbf{x_i}, y_i\}_{i=1}^N$, where N denotes the number of sample space **ID**, $\mathbf{x_i}$ denotes all features of ith sample and y_i denote the lable of $i$the sample. For a VFL system, each $\mathbf{x_i}$ is vertically distributed C shares and stored in each party, i.e., $\mathbf{x_i} = \{\{x_i\}_a, \{x_i\}_{l_1}, \{x_i\}_{l_2}, ..., \{x_i\}_{l_K}\}$. In VFL, the active party owns the privacy dataset $D_a = \left\{\{\{x_i\}_a\}_{i=1}^N, Y = \{y_i\}_{i=1}^N\right\}$, and each passive party owns private data $D_{l_k} = \{\{x_i\}_{l_k}\}_{i=1}^N$, where l_k denote kth passive party. For convenience, we use l_0 to stand for active party a.

In VFL, the goal is to collaboratively train models θ among the active party and all passive parties, who have the same data **ID** and differential features. In addition, to protect data privacy, parties keep their data locally during training. For VFL, only the active party has the label space, and passive parties only have some features space $\{x_i\}_{(l_k)}{}_{i=1}^N$, not have label space. Labels and features are stored in different parties in VFL. A single party cannot train a complete model. This paper assumes that each party can individually choose a locally trained model θ. The goal of EASTER is to train multi-models $(\theta_a, \theta_{l_1}, ..., \theta_{l_K})$ jointly

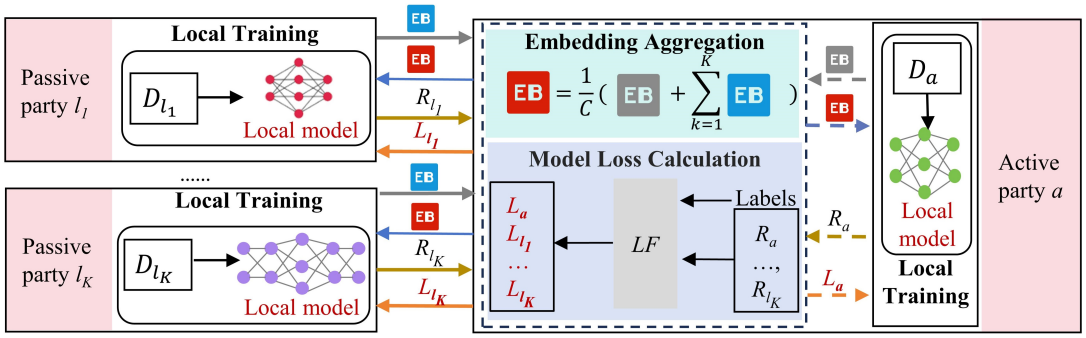


Fig. 2. An overview of the EASTER training in the heterogeneous models setting. Each participant is the owner of the heterogeneous local model and incomplete features. The active party also has labels, partial features, and heterogeneous local models. K passive parties and one active party are the suppositions made by EASTER. The dashed box is merely being used as an example. The solid box depicts the entity or module.

with all parties without revealing local samples, which aims to minimize the loss of all training samples. The problem of EASTER can be formulated as

$$\min_{\theta_a,...,\theta_{l_K}\in\mathbb{R}^d} \ell(\theta_a,...,\theta_{l_K};D) :\triangleq \frac{1}{N}\sum_{i=1}^{N} f\left(\theta_a,...,\theta_{l_K};\mathbf{x}_i,y_i\right) \tag{2}$$

In Eq. (2), N is the total number of samples, $\mathbf{x}_i = \bigcup_{k=0}^{K}\{x_i\}_{l_k}$ denote ith sample features, y_i is the label of ith sample labels. $\theta_a \in \mathbb{R}^d$ denotes the model parameters owned by the active party and $\theta_{l_k} \in \mathbb{R}^d$ denotes the model parameters owned by l_kth passive party . $f(\cdot)$ is the loss function.

To meet the real application requirements, this paper considers that the active party and the passive parties have independent local models, namely $(\theta_a,...,\theta_{l_K})$. Each local model can be expressed as Eq. (2), that is, it has C loss functions $(f_a(\cdot),...,f_{l_K}(\cdot))$.

We focus on solving the issue of the active and passive parties working together to train the C heterogeneous local model safely. That is, the active party and all passive parties collaborate to minimize all loss functions $(f_a(\cdot),...,f_{l_K}(\cdot))$.

3.3 Threat Model

In EASTER, all participants aim to collaboratively train optimal multiple models without compromising the privacy of their local data. In VFL, participants collaborate to train an optimal model without compromising local data privacy. We assume all participants of VFL are honest but curious. The passive party is capable of accurately performing local pre-training and embedding calculations. However, there is a possibility that they may attempt to infer the label information of the active party from the training process. On the other hand, while the active party can correctly execute the global model training process, it may attempt to infer the private information of the passive party from their embeddings. Additionally, external attackers and malicious message tampering are also ignored in our method. A secure communication channel exists between the active party and the passive party, ensuring that external attackers cannot access or detect the communication model information.

4 Proposed Methodology

4.1 Overview

To solve the low accuracy problem caused by the passive party's local heterogeneous model, we proposed a new method EASTER. An overview of the proposed EASTER is shown in Fig. 2. Our scheme comprises C participants, consisting of one active party a and K passive parties. Each participant owns private datasets and heterogeneous models locally. The main goal of EASTER is to aggregate the private datasets from all parties and utilize them collaboratively to train C heterogeneous models.

The EASTER includes two entities: the active and passive parties. In the training phase, an active party is mainly responsible for local heterogeneous model training, local embedding aggregation, and loss value calculation. Specifically, the active party collaborates with the passive party to train local heterogeneous models. The active party needs to aggregate the local embedding values of all passive parties and obtain the global embedding value to train heterogeneous models. The passive party is mainly responsible for initializing blinding factors and training local heterogeneous models. Specifically, the passive party generates a pair of public and private keys and sends the public key to the active party. The passive party generates a blinding factor based on its private key and the public keys of other passive parties to protect the local embedding value. In addition, the passive party completes the training of local heterogeneous models with the assistance of the active party.

In addition, the EASTER consists of three main modules.

- **Local Training**. All participants' collaborative training of multiple local heterogeneous models occurs in this module. In detail, each participating party utilizes local features and models to generate local embedding values. Furthermore, each participant employs the global embeddings alongside their local models to derive local predictive outcomes Finally, each participant updates their local model in reverse according to the loss values and the optimization function.
- **Embedding Aggregation**. This module combines all participants' local embeddings to generate the global embedding.

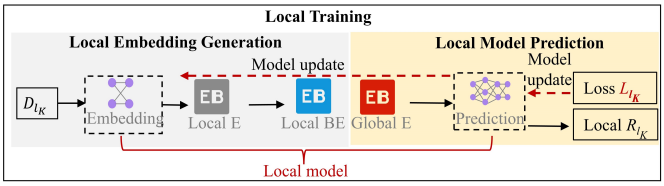


Fig. 3. The local training process of EASTER."Local E" denotes the local embedding of the party. "Local BE" denotes the blinding local embedding of the passive party.

- **Model Loss Calculation**. In this module, the active party aims to utilize local labels to support the passive party in calculating the heterogeneous model's loss value.

4.2 Local Model Training

To address the issue of a limited exchange of model information among participants, we adopted a strategy inspired by prior studies [22], [29]. When performing local training, the local heterogeneous network is partitioned into two distinct parts: the embedding network (e.g., embedding layer) and the prediction network (e.g., decision layers). **The embedding layer** embeds some features owned by participants in the same embedding space. The embedding layer of the l_kth participant is $h(\theta_{l_k})$. We denote $E_{l_k} = h(\theta_{l_k}, D_{l_k})$ as the embedding of D_{l_k} for l_kth participant. **The decision layers** are the result of predicting sample x based on the supervised learning task. We represent the prediction result by using $R_{l_k} = p(\theta_{l_k}, x)$.

Local model training consists of local embedding generation, local model prediction, and model update, as shown in Fig.3. Local embedding generation refers to the method by which participants acquire the local embedding result E from the input layer to the embedding layer using the local heterogeneous network. To project the local embedding result, we inject the blinding factors into the local embedding of the passive party. Local model prediction refers to the process where each participant uses the global embedding values as inputs to the decision layer to obtain sample prediction values. Model update is the process of optimizing model parameters from the output layer to the input layer using an optimization algorithm. For example, when the l_kth participant gets the stochastic gradient descent (SGD) [30] algorithms during backpropagation, the local heterogeneous model will be updated based on the following:

$$\theta_{l_k}^{t+1} \leftarrow \theta_{l_k}^{t} - \eta_{l_k} \nabla_{\theta_{l_k}^{t}} L_{l_k} \tag{3}$$

where $\theta_{l_k}^{t+1}$ stands for the model of l_kth participant in epoch t. η_{l_k} denotes learning rate and $\nabla_{\theta_{l_k}^{t}} L_{l_k}$ denotes the gradient. Below we introduce the process of local embedding blinding in detail.

Local Embedding Blinding. In this paper, the global embedding aggregation is implemented in the active party, it implies that we only need to protect the local feature of the passive party. We prevent the direct leakage of local features from computing the local embedding $\{E_{l_k}\}_{k=1}^{K}$ at each participant locally. To safeguard the original features further, we add a blinding factor (r_{l_k}) [31] to the local embedding $(\{E_{l_k}\}_{k=1}^{K})$ of the passive party to strengthen the security of aggregation. Each passive party initializes a public-private key pair and then generates the blinding factor.

- **Key Generation.** Let $\mathbb{G}$ be a cyclic group of prime order p, with discrete logarithmic relation unknown generator g. Each l_k generates a private key $SK_{l_k} = s_{l_k} \in \mathbb{Z}_p$ and a public key $PK_{l_k} = g^{s_{l_k}} \in \mathbb{G}$, where $k \in [1, K]$. Each passive party sends its public key to the active party.
- **Blinding Factor Generation.** After the key initialization, each passive party l_k downloads other passive parties' public key from the active party and computes the shared key $CK_{k,j} = H((PK_{l_j})^{SK_{l_k}})$, where $k, j \in [1, K], j \neq k$ and $H(\cdot)$ is a collusion-resistant secure hash function capable of converting strings of any length into elements in $\mathbb{Z}_p$. The mathematical expression of the connection between shared keys is given in Eq. (4).

$$\begin{aligned} CK_{k,j} &= H((PK_{l_j})^{SK_{l_k}}) = H((g^{s_{l_j}})^{s_{l_k}}) \\ &= H((g^{s_{l_k}})^{s_{l_j}}) = H((PK_{l_k})^{SK_{l_j}}) = CK_{j,k} \end{aligned} \tag{4}$$

Each passive party l_k computes the blinding factor to generate r_{l_k} (see Eq. (5)).

$$r_{l_k} = \sum_{j \in [1,k], j \neq k} (-1)^{k>j} H(CK_{k,j}) \tag{5}$$

Where $(-1)^{k>j} = -1$ if $k > j$. Particularly, the sum of all blinding factors $(\sum_{k \in [1,K]} r_{l_k})$ is 0.

The local embedding value with privacy protection can be obtained by adding the above-mentioned blinding factor to the passive party's local embedding as follows.

$$\{[E_{l_k}]\}_{k=1}^{K} = \{E_{l_k} + r_{l_k}\}_{k=1}^{K} \tag{6}$$

The l_kth passive party sends the blinded embedding $[E_{l_k}]$ to the active party.

4.3 Secure Embedding Aggregation

Given the heterogeneous model in participants, the optimal model parameters required by each participant are different. Existing aggregation-based VFL cannot provide enough effective information for each participant. Fortunately, with the same label features, the same embedding space can be obtained through the embedding layer of the heterogeneous network. We effectively exchange the local features owned by aggregating the local embedding of each participant. The active party obtains the blinding local embedding, refer to $\{[E_{l_k}]\}_{k=1}^{K} = \{E_{l_k} + r_{l_k}\}_{k=1}^{K}$ A global embedding E is generated after the average aggregating method (see Eq. (7)).

$$E = \frac{1}{C}(E_a + \sum_{k=1}^{K}[E_{l_k}]) = \frac{1}{C}(E_a + \sum_{k=1}^{K} E_{l_k} + \sum_{k=1}^{K} r_{l_k}). \tag{7}$$

In Eq. (7), C denotes the total number of parties; K denotes the total number of passive parties. We know from blinding factor generation that $\sum_{k=1}^{K} r_{l_k} = 0$, so we can obtain the global embedding.

Algorithm 1 EASTER Training

Require: Datasets D_{l_k}, parties $k = 0, 1, ..., K$, Epoch T
Ensure: Excellent $\theta_{l_0}, \theta_{l_1}, ..., \theta_{l_K}$

1: **for** $t = 1, ..., T$ **do**
2: **for** each party l_k in parallel **do**
3: $[E_k] \leftarrow h(\theta_{l_k}, D_{l_k}) + r_{l_k}$
4: **end for**
5: Perform the global embedding E update by Eq. (7)
6: The active party sent E to passive parties
7: **for** each party l_k in parallel **do**
8: Perform local prediction $R_{l_k} = p(\theta_{l_k}, E)$
9: Sent prediction R_{l_k} to active party
10: **end for**
11: $L_{l_k} = LF(R_{l_k}, Y)$ // Calculate the loss of each party
12: The active party sent L_{l_k} to l_kth passive party
13: **for** each party l_k in parallel **do**
14: $\theta_{l_k}^{t+1} \leftarrow \theta_{l_k}^{t} - \eta_{l_k} \nabla_{l_k} L_{l_k}$ // Party update local heterogeneous model
15: **end for**
16: **end for**

4.4 Model Loss Calculation

In EASTER, the passive party lacks labels locally, making it unable to calculate the loss value independently. Only the active party possesses the labels and has a heterogeneous model loss value calculation module to aid the passive party in computing the loss value. The active party determines the optimal loss function LF following the supervised learning task's requirements. When the active party selects the cross-entropy loss function, for instance, the loss function of the l_kth participant's heterogeneous model is calculated by Eq. (8).

$$LF\left(R_{l_k}, Y\right) = -\frac{1}{N} \sum_{i=1}^{i=N} \left((Y_i) \log_2 \left((R_{l_k})_i \right) + (1 - Y_i) \log_2 \left(1 - (R_{l_k})_i \right) \right) \tag{8}$$

In Eq. (8), R_{l_k} represents the predicted probability; Y represents the actual label value; N represents the total number of the training sample.

4.5 Multiple Heterogeneous models Training

Algorithm 1 presents the pseudo-codes of EASTER for implementing multiple heterogeneous models training process. The party refers to the general terms of active party and passive parties. l_0 represents the active party. Participants only have the features subsets and need the assistance of other participants during the training process. In addition, the local model of the passive party does not have labels and cannot calculate the loss value of the heterogeneous local model, which requires the cooperation of the active party.

The training process of one round of EASTER mainly includes the following steps. *Step 1* (line 3 to line 5 of the Algorithm 1) Each participant uses local data feature D_{l_k} and local heterogeneous embedding network $h(\theta_{l_k}, D_{l_k})$ to obtain the local blinded embedding $[E_{l_k}] = h(\theta_{l_k}, D_{l_k}) + r_{l_k}$. *Step 2* (line 6) The active party realizes the safe aggregation of global embedding by Eq. (7) and obtains the local data characteristics of all participants. *Step 3* (lines 8 to 9) The participants calculate the prediction result R_{l_k} from the global embedding E and the decision layer $p(\theta_{l_k}, E)$ of the heterogeneous network in parallel. *Step 4* (lines 11 to 13) The active party calculates the loss value of participant l_k from the loss function $LF(R_{l_k}, Y)$, the prediction R_{l_k}, and local labels. $LF(R_{l_k}, Y)$ chooses different loss functions by the task requirements. For example, the cross-entropy or the logistic regression loss function can be chosen for classification tasks. *Step 5* (lines 13 to 15) Each participant updates and optimizes the parameters of the local heterogeneous model according to the loss and the optimization method (e.g., SGD, SGD with momentum, Adagrad, and Adam). After T training rounds, we will obtain C local heterogeneous models. Therefore, the EASTER model trains multi-heterogeneous models and safeguards the passive side's local embedding. EASTER enables the generation of multiple optimized heterogeneous models from a single training.

4.6 Convergence Analysis

We provide the convergence analysis of EASTER below, which relies on the fact that the local gradients in the training process are unbiased. In our analysis, we use $f_k(\theta)$ to denote the loss function of kth participant. Additionally, we present a set of assumptions required to perform the convergence analysis, similar to existing general frameworks [7], [22].

Assumption 1 (L - Lipschitz Continuous)**.** *Suppose there exists a constant $L > 0$ and the gradient of the local objective function is L - Lipschitz Continuous for $\forall\ \theta_k, \theta_k'$ and $k \in \{0, 1, ..., K\}$, there is*

$$\left\| \bigtriangledown f_k(\theta_k) - \bigtriangledown f_k(\theta_k') \right\| \leqslant L \left\| \theta_k - \theta_k' \right\| \tag{9}$$

Assumption 2 (Unbiased Gradient and Bounded Variance)**.** *The local gradient $g_k^t = \bigtriangledown f_k(\theta^t, E)$ for each participant is an unbiased estimator, where k denotes l_kth participant. We assume that the expectation of the local heterogeneous model gradient g_k^t satisfies $\mathbb{E}\left[g_k^t\right] = \nabla f_k\left(\theta_k^t\right) = \nabla f_k^t, \forall k \in \{0, 1, 2, \ldots, K\}$. And the variance of the local gradient g_k^t is bounded by σ_k^2: $\mathbb{E}\left[\left\|g_k^t - \nabla f_k\left(\theta_k^t\right)\right\|\right] \leqslant \sigma_k^2, \forall k \in \{0, 1, 2, \ldots, K\}, \sigma_k^2 \geqslant 0$.*

Assumption 3 (Bounded Gradient)**.** *We assume that G bounds the expectation of the local heterogeneous model gradient. We have $\mathbb{E}\left[\left\|g_k^t\right\|\right] \leqslant G, \forall k \in \{0, 1, 2, \ldots, K\}$.*

Assumption 4 (u - convex)**.** *Suppose the f_k is u - convex, which means that there exists u such that $\nabla^2 f_k(\theta) \geqslant u$*

We aim to produce convergent outcomes for each participant's local heterogeneous model. When $\min_\theta$ is the smallest, we anticipate that each participant's objective function $f(\theta, D)$ converges. Since the proof of convergence for each participant's objective function is consistent, the convergence result of the statement for kth participant is displayed below.

Theorem 1 (Convergence of EASTER)**.** *Under assumptions 1 - 4, we prove that the convergence of EASTER is:*

$$\begin{aligned} &\mathbb{E}\left[f_k\left(\theta_k^{t+1}\right) - f_k\left(\theta_k^*\right)\right] \leqslant \\ &\left(1 - \mu\eta_k\sigma_k^2\right) \mathbb{E}\left[f_k\left(\theta_k^t\right) - f_k\left(\theta_k^*\right)\right] + \frac{1}{2}\eta_k^2 LG \end{aligned} \tag{10}$$

Where the θ_k^t represents the kth participant's local model obtained in the t epoch and θ_k^* denotes the convergent model. η_k denotes the learning rates in tth epoch. As the number of epoch t increases, the upper bound of the distance between the current model parameter θ_k^t and the convergent model θ_k^* gradually decreases, which shows the convergence of the EASTER model.

Based on the existing assumptions, we provide complete proof of the convergence of EASTER.

Proof. We assume that Assumption 1 - 4 hold. We analyze the convergence of EASTER according to [30].

At step t, we suppose the kth participant optimizes the model θ_k^t. So the current step's stochastic gradient is g_k^t (denotes g)and learning rate η_k. We have the following formula:

$$\theta_k^{t+1} = \theta_k^t - \eta_k g\left(\theta_k^t\right) \tag{11}$$

According to the Assumption 1, the following inferences 12 will hold.

$$f_k(\theta) - \left[f_k\left(\acute{\theta}\right) + \nabla f_k\left(\acute{\theta}\right)^T\left(\theta - \acute{\theta}\right)\right] \leqslant \frac{1}{2}L\left\|\theta - \acute{\theta}\right\|_2^2 \tag{12}$$

Eq. (11) can be expressed as follows based on Assumption 1 and Inferences 12.

$$f_k\left(\theta_k^{t+1}\right) - f_k\left(\theta_k^t\right) = -\eta_k \nabla f_k\left(\theta_k^t\right)^T g\left(\theta_k^t\right) + \frac{1}{2}L\left\|g\left(\theta_k^t\right)\right\|_2^2 \tag{13}$$

By taking the expectation of Eq. (13), the following equation is obtained:

$$\begin{aligned}\mathbb{E}\left[f_k\left(\theta_k^{t+1}\right)\right] - f_k\left(\theta_k^t\right) \leqslant & -\eta_k \nabla f_k\left(\theta_k^t\right)^T \mathbb{E}\left[g\left(\theta_k^t\right)\right] \\ & + \frac{1}{2}\eta_k^2 L \mathbb{E}\left[\left\|g\left(\theta_k^t\right)\right\|_2^2\right]\end{aligned} \tag{14}$$

Based on Assumption 2 and Assumption 3, Eq. (14) can be represented by the following equation.

$$\begin{aligned}A &= -\eta_k \nabla f_k\left(\theta_k^t\right)^T \mathbb{E}\left[g\left(\theta_k^t\right)\right] \leqslant -\eta_k \sigma_k^2 \left\|\nabla f_k\left(\theta_k^t\right)\right\|_2^2 \\ B &= \frac{1}{2}\eta_k^2 L \mathbb{E}\left[\left\|g\left(\theta_k^t\right)\right\|_2^2\right] \leqslant \frac{1}{2}\eta_k^2 L(G\|\nabla f_k\left(\theta_k^t\right)\|^2 + G)\end{aligned} \tag{15}$$

According to Eqs. (15) - (14), an equivalent equation is obtained.

$$\mathbb{E}\left(f_k\left(\theta_k^{t+1}\right)\right) - f_k\left(\theta_k^t\right) \leqslant -\left(\eta_k \sigma_k^2 - \tfrac{1}{2}\eta_k^2 LG\right)\left\|\nabla f_k\left(\theta_k^t\right)\right\|^2 + \tfrac{1}{2}\eta_k^2 LG \tag{16}$$

The objective function should decrease at each step, which requires ensuring that the coefficient of $\|\nabla f_k\left(\theta_k^t\right)\|_2^2$ is less than 0. Therefore we need to assume $(\eta_k \sigma_k^2 - \frac{1}{2}\eta_k^2 LG) > 0$. For convenience, we assume $\sigma_k^2 \leqslant \eta_k LG$. Upon repeated iteration, the ensuing equation shall be derived.

$$\mathbb{E}\left[f_k\left(\theta_k^{t+1}\right)\right] - f_k\left(\theta_k^t\right) \leqslant -\tfrac{1}{2}\eta_k \sigma_k^2 \left\|\nabla f_k\left(\theta_k^t\right)\right\|_2^2 + \tfrac{1}{2}\eta_k^2 LG \tag{17}$$

The following formula could be obtained based on Assumption 4.

$$\mathbb{E}\left[f_k\left(\theta_k^{t+1}\right)\right] - f_k\left(\theta_k^t\right) \leqslant -\mu\eta_k \sigma_k^2\left(f_k\left(\theta_k^t\right) - f\left(\theta^*\right)\right) + \frac{1}{2}\eta_k^2 LG \tag{18}$$

The goal of our optimization is to minimize $f_k\left(\theta_k^t\right) - f\left(\theta^*\right)$, and the Eq. (18) can be derived iteratively as follows.

$$\begin{aligned}&\mathbb{E}\left[f_k\left(\theta_k^{t+1}\right) - f_k\left(\theta_k^*\right)\right] - \mathbb{E}\left[f_k\left(\theta_k^t\right) - f_k\left(\theta_k^*\right)\right] \leqslant \\ &-\mu\eta_k \sigma_k^2 \mathbb{E}\left[f_k\left(\theta_k^t\right) - f_k\left(\theta^*\right)\right] + \frac{1}{2}\eta_k^2 LG\end{aligned} \tag{19}$$

Eq. (19) can be rewritten as

$$\mathbb{E}\left[f_k\left(\theta_k^{t+1}\right) - f_k\left(\theta_k^*\right)\right] \leqslant \left(1 - \mu\eta_k \sigma_k^2\right)\mathbb{E}\left[f_k\left(\theta_k^t\right) - f_k\left(\theta_k^*\right)\right] + \tfrac{1}{2}\eta_k^2 LG \tag{20}$$

This completes the proof of Theorem 1. □

4.7 Security Analysis

In EASTER, we use a blinding factor to hide the local embedding of passive parties. If the blinding factor is security, we can ensure that other passive parties cannot infer the local features of passive parties. Next, we illustrate the security of the blinding factor.

In our model, the active party stores the public key PK of all passive parties and local embedding with blinding factors submitted by passive parties during the model training process. The local embedding submitted by each passive party l_k is appended with a blinding factor generated based on the shared key $CK_{k,j}$, where $j \in [1, K], j \neq k$. Our method ensures that the passive party cannot obtain the shared key between the other passive parties, so the true local embedding cannot be obtained. The Computational Diffie–Hellman (CDH) problem ensures that given g^a and g^b, the probability of computing the g^{ab} is negligible. Therefore, given access to all public key $PK_{l_k} = g^{s_{l_k}}$, the semi-honest passive party l_k is not able to infer the shared key $CK_{k,j} = H((g^{s_{l_j}})^{s_{l_k}})$, where $\forall k, j \in [1, K], j \neq k \neq K$. The semi-honest passive party cannot obtain the real local embedding of the remaining passive parties. The passive party cannot infer the raw features of the remaining passive parties. Each passive party only has its private key, and even in the case of collusion between multiple semi-honest passive parties, the shared key between the other honest passive parties cannot be obtained, and the real local embedding cannot be obtained.

Therefore, the EASTER method ensures that the honest-but-curious passive party cannot obtain real local embedding with blinding factors from global embedding and thus cannot infer the raw features. The EASTER method effectively protects the raw features of the passive parties.

5 PERFORMANCE EVALUATION

5.1 Experimental Setup

To evaluate the performance of the EASTER method, we chose three classical image datasets for model training and referenced the training data sets of existing VFL methods [16], [18]. We selected six typical neural networks and discussed the tasks utilized to train them.

5.1.1 Datasets

The three datasets are MNIST [32], *FashionMNIST* (FMNIST) [33] and CIFAR10 [34]. MNIST was a picture data set of handwritten digits, including 60,000 training data sets

TABLE 2
Comparison of VFL methods with baseline methods on three benchmark datasets. "Homo" refers to homogeneous, and "heter" refers to heterogeneous.

Datasets	Methods	Types	Models Testing Accuracy (%)					
			CNN	LeNet	MLP	ResNet18	ResNet50	ResNet101
MNIST	Local	-	23.47 ± 2.36	35.57 ± 3.14	31.89 ± 3.10	95.06 ± 0.53	94.72 ± 0.69	93.24 ± 0.74
	Pyvertical [16]	Homo	92.68 ± 1.14	94.18 ± 1.16	91.77 ± 0.70	97.01 ± 0.24	94.82 ± 0.85	95.29 ± 0.46
	C_VFL [18]	Homo	89.71 ± 1.65	94.72 ± 1.74	95.33 ± 4.6	97.84 ± 0.89	97.27 ± 1.49	95.23 ± 1.19
	Agg_VFL [21]	Homo	91.72 ± 2.08	86.03 ± 1.41	95.71 ± 2.39	98.28 ± 2.39	97.81 ± 1.98	97.73 ± 0.83
	EASTER(our)	Heter	**97.58 ± 1.71**	**97.87 ± 1.47**	**97.73 ± 1.48**	**98.54 ± 1.14**	**97.84 ± 1.37**	**97.89 ± 1.64**
FMNIST	Local	-	65.55 ± 3.41	76.24 ± 1.04	70.40 ± 2.37	84.47 ± 0.49	84.25 ± 0.35	82.37 ± 1.06
	Pyvertical [16]	Homo	72.62 ± 1.63	76.34 ± 2.35	73.86 ± 2.12	84.86 ± 1.45	84.55 ± 0.88	83.79 ± 0.85
	C_VFL [18]	Homo	83.92 ± 1.68	84.32 ± 1.84	84.53 ± 1.05	87.50 ± 2.51	87.41 ± 1.81	87.31 ± 1.60
	Agg_VFL [21]	Homo	84.37 ± 1.88	84.68 ± 1.41	84.37 ± 2.08	87.50 ± 1.64	86.54 ± 1.17	84.37 ± 1.73
	EASTER(our)	Heter	**87.78 ± 1.07**	**88.33 ± 0.99**	**88.08 ± 1.13**	**88.01 ± 1.20**	**87.84 ± 1.45**	**87.60 ± 1.38**
CIFAR10	Local	-	37.35 ± 0.49	53.04 ± 0.65	32.33 ± 0.48	59.05 ± 0.95	55.16 ± 0.93	49.98 ± 0.36
	Pyvertical [16]	Homo	37.95 ± 1.71	53.71 ± 2.35	33.85 ± 1.82	62.94 ± 1.96	57.62 ± 2.08	52.80 ± 1.79
	C_VFL [18]	Homo	48.85 ± 3.56	61.42 ± 3.51	46.16 ± 1.33	63.75 ± 2.02	58.24 ± 1.22	56.19 ± 2.05
	Agg_VFL [21]	Homo	55.04 ± 2.82	60.11 ± 1.28	42.67 ± 1.51	74.93 ± 1.25	71.99 ± 1.02	64.22 ± 1.18
	EASTER(our)	Heter	**62.38 ± 2.46**	**64.72 ± 1.77**	**64.64 ± 2.62**	**71.74 ± 2.36**	**72.81 ± 2.18**	**73.76 ± 1.94**

and 10,000 test data sets. FMNIST was a frontal image of commodities covering 70,000 grey levels from 10 categories, including 60,000 training datasets and 10,000 test data sets. The CIFAR-10 dataset was a collection of color pictures that was divided into 10 distinct categories. It included 50,000 images for training and 10,000 images for testing. In our experimental setup, the training dataset was utilized for training the performance of EASTER, while the test dataset was employed to evaluate the accuracy of EASTER.

5.1.2 Models

We considered three neural networks to simulate local heterogeneous networks: *Multi-Layer Perceptron neural network* (MLP) [35], *Convolutional Neural Network* (CNN) [36], and LeNet [37]. Among them, MLP was a three-layer fully connected network. CNN included 2 convolutional layers and 2 fully connected layers. LeNet consisted of three convolutional layers, one pooling layer, and three fully connected layers. Moreover, we evaluated our method's performance in three heterogeneous ResNet networks (e.g. ResNet18, ResNet50, and ResNet101) to demonstrate the scalability of our method. During the training process, participants randomly selected a network based on local resources, and multiple participants formed heterogeneous models for collaborative training.

5.1.3 Training Tasks

Each participant in the scenario learned the task of heterogeneous supervised learning using their own set of local features. The simulation involved C participants who independently selected their local heterogeneous networks based on their requirements. All C participants collaboratively trained multiple heterogeneous networks. The active party worked with K passive parties to calculate the loss value of the local model. In the end, all C participants complete the local heterogeneous model learning task.

5.1.4 Baselines

To evaluate the performance of EASTER under heterogeneous models, we conducted a comparative analysis with baseline methodologies. The comparison included local training, where the active party trains a model separately using their local features, along with three recent VFL studies as follows.

- Pyvertical [16] implemented basic SplitVFL model training and divided the entire model into a top model and a bottom model. The active party owned the top-level model, and the passive party owned the bottom-level model. The active party and the passive party collaboratively trained the global model.
- C_VFL [18] reduced the communication overhead of VFL by compressing the amount of transmitted data based on traditional SplitVFL.
- Agg_VFL [21] was an AggVFL method used to solve the imbalanced features of each participant.

Moreover, all methods were under the same experimental configuration. Furthermore, aiming to evaluate the performance of local embedding aggregation in EASTER, we conducted a comparative analysis with baseline methodologies.

- Local involved independent model training by the active party using their local features.
- Heter_aggVFL involved collaborative training among participants with heterogeneous local models, achieved by aggregating local prediction results.
- Homo_aggVFL involved collaborative training among participants with homogeneous local models by aggregating local prediction results.
- Homo_EASTER enabled collaborative training among participants with heterogeneous local models, achieved by aggregating local embedding.

The local prediction results of active part aggregation in the two methods Homo_aggVFL and Heter_aggVFL, i.e., no further training was required after active part aggregation. In the Homo_EASTER and EASTER methods, it was necessary to further train the locally embedded data aggregated by the active party. Finally, we evaluated the performance of EASTER on local heterogeneous optimizers.

5.1.5 Implementation Details

The EASTER and baseline methods were implemented using the PyTorch [38] framework. In VFL, four participators

were employed, with one party assuming an active role and the remaining three parties adopting passive roles. The data set comprising all samples was partitioned into four distinct portions vertically. Each participant possessed a subset of features that was representative of all the samples. The active party owns ownership of the labels assigned to all samples. During the training process, all participants engaged in collaborative training of local heterogeneous models while ensuring that the samples used for training were aligned. Furthermore, in our training process, we set the batch size to 128, and the learning rate for all networks to 0.05.

5.2 Performance in Local Heterogeneous Models

We conducted a comparative analysis between EASTER and the baseline scheme of the standard neural network-based VFL methods. All approaches were adjusted to similar configurations. For instance, all methods configured the same model network to be trained on the same dataset. To evaluate the effectiveness of EASTER on local heterogeneous models, our approach incorporated three small neural networks and three ResNet networks. Each participant included in the study randomly gets a local model. All local models were trained through collaboration among the four participants. However, the current baseline scheme exclusively facilitated each participant's cooperative training of a global model. Therefore, the settings for the baseline scheme were as follows. Local refers to the active party independently training the local model using the local partial features and labels. Pyvertical [16] and C_VFL [18] were two VFLs that belong to the SplitVFL category. Agg_VFL [21] was belong to one of the aggVFL. In both methods, all participants jointly trained multiple heterogeneous network models. The experimental findings were presented in Table 2, indicating that EASTER achieves the highest level of accuracy.

5.2.1 Models Accuracy

We evaluated the accuracy of EASTER on classic data sets and heterogeneous models, and the experimental results are shown in Table 2. The baseline method's training type was homogenous, meaning that all passive local models were homogenous. Our EASTER's training type was heterogeneous, which means that the local model of the passive party was also heterogeneous. From Table 2, we found that compared with *Local* method, the model accuracy of EASTER had been significantly improved. This was because the *Local* method only used some of the local features of the active party to train the model, while EASTER collaborated with all the features of the four participants to train the model. Therefore, as the features of the training data set increased, the training accuracy of the model continued to improve. Compared with the better method C_VFL, the model accuracy of EASTER was improved by 4% under the FMNIST dataset and LetNet network. This was because the intermediate results or prediction results aggregated by the existing VFL method had more local model information, while the embedding aggregated by EASTER had less local model information. Local model information had a certain negative impact on the training results. Therefore, compared with existing methods, the accuracy of the heterogeneous model trained by EASTER was better. This finding demonstrated EASTER's ability to simultaneously train locally heterogeneous models with improved accuracy.

TABLE 3
Comparison performance in embedding aggregation of EASTER on three benchmark datasets and three heterogeneous models.

Datasets	Methods	Models Accuracy (%)		
		CNN	LeNet	MLP
MNIST	Local	23.47	35.57	31.89
	Heter_aggVFL	36.29	45.15	68.78
	Homo_aggVFL	91.72	86.03	95.71
	Homo_EASTER	96.05	**98.61**	96.57
	EASTER(our)	**97.58**	97.87	**97.73**
FMNIST	Local	65.55	76.24	70.40
	Heter_aggVFL	45.45	70.04	42.82
	Homo_aggVFL	84.37	84.68	84.37
	Homo_EASTER	86.42	**90.45**	86.57
	EASTER(our)	**87.78**	88.33	**88.08**
CIFAR10	Local	37.35	53.04	32.33
	Heter_aggVFL	51.72	60.70	40.65
	Homo_aggVFL	55.04	60.11	42.67
	Homo_EASTER	59.55	**66.41**	61.86
	EASTER(our)	**62.38**	64.72	**64.64**

5.2.2 Communication Rounds

We evaluated the number of communication rounds for EASTER and the baseline methods. This evaluation was conducted during the training of three heterogeneous networks. Our approach necessitated four communication rounds between passive and active participants in each epoch. In specially, the passive party sent the local embedding with the blinding factor to the active party in the first communication round. The active party delivered the global embedding to the passive party in the second communication round. The prediction output of the local heterogeneous model was sent by the passive party to the active party in the third communication round. The active party communicated with each passive party of the loss value in the fourth communication round. In comparison, the baseline approach only required two communication rounds. Therefore, the number of communication rounds each epoch in our technique was twice that of the baseline method. The first round of communication occurred when the passive party sent local intermediate results to the active party. In the second communication round, each passive party obtained the gradient value from the active party. However, EASTER realized training multiple models simultaneously. When the number of heterogeneous models trained simultaneously is greater than or equal to 3, it has been observed that EASTER provides a significant decrease in the number of communication rounds required compared to the baseline method. Additionally, EASTER and the baseline method could converge at the same global epoch. This finding showed that EASTER could simultaneously train multiple models that were different in the environment while needing fewer total communication rounds of training.s

5.3 Performance in Embedding Aggregation

Embedding aggregation was a core component of the EASTER. To assess the efficacy of embedding aggregation,

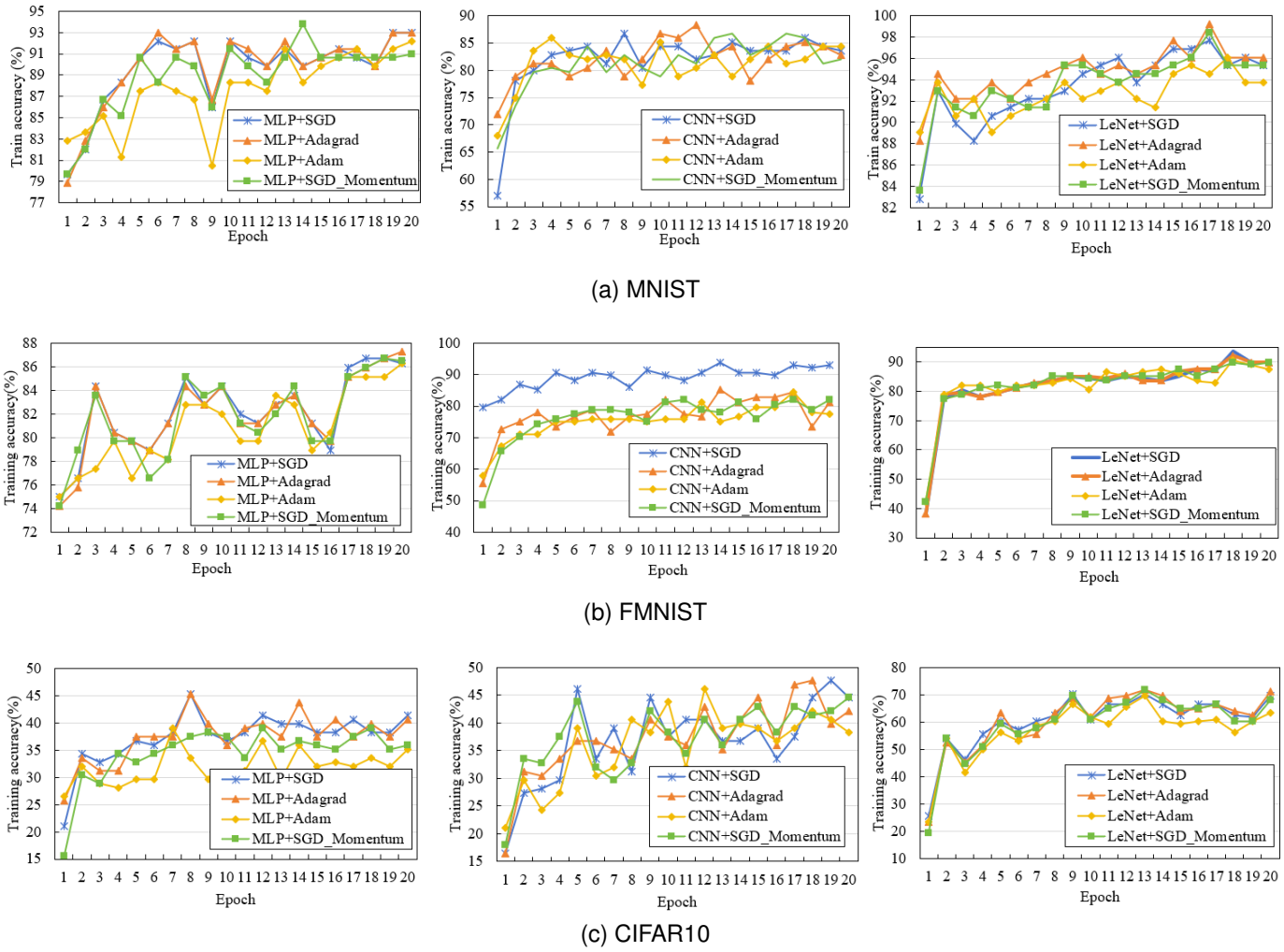


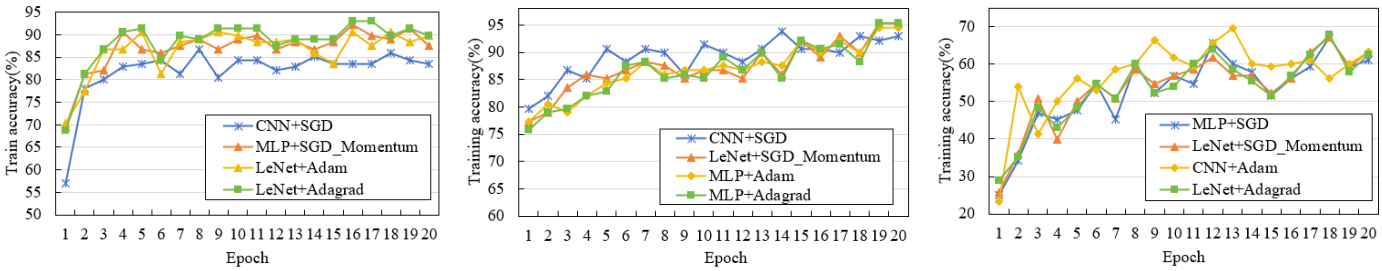
Fig. 4. Comparison performance in local heterogeneous optimation of EASTER on three datasets and three models. (a) MNIST datasets; (b) FMNIST datasets; (c) CIFAR10 datasets.

we conducted a comparative analysis of the performance of homogenous and heterogeneous embedding aggregation techniques (Homo_EASTER, EASTER) with homogeneous and heterogeneous prediction aggregation techniques (Heter_aggVFL, Homo_aggVFL). The performance comparison results of each method were presented in Table 3. From Table 3, it was evident that both homogenous and heterogeneous embedding aggregation methods exhibit superior accuracy compared to the prediction aggregation methods. The reason for this was that embedding solely encompasses the local features of the participants, whereas the prediction results encompass both feature and model information. The additional model information could potentially influence the model's correctness.

We specifically discussed the performance of Homo_EASTER and Homo_aggVFL. Even though each participant had a homogeneous network, homogeneous network training model parameters were varied. While Homo_aggVFL aggregated local predicted values, Homo_EASTER aggregates local embedding values. Local predicted values had more model information that harms model accuracy. However, local embedding contained less model information which harmed the heterogeneous model training. Thus, Homo_EASTER was more accurate than Homo_aggVFL. Moreover, EASTER attained comparable levels of accuracy when compared to the Homo_EASTER approach. The findings demonstrated that EASTER was appropriate for training heterogeneous models as local embedding has less information.

5.4 Performance in Local Heterogeneous Optimization

Extensive experiments were done to evaluate the effect of local heterogeneous models and heterogeneous optimizers on the efficiency of EASTER. We had selected four widely used optimization algorithms, namely SGD [30], SGD with Momentum [39], Adagrad and Adam [40], to implement local heterogeneous optimizers. The heterogeneous optimizer of EASTER was validated using the same model and datasets including MNIST, FMNIST, and CIFAR10. The experimental findings were shown in Fig.s 4(a)-4(c). For example, as shown in Fig. 4(a), it could be found that, with the same local model and different optimizers, the local models of the four participants exhibit convergence. Moreover, Fig.s 4(b),(c) depicted the training accuracy of the EASTER for the three heterogeneous networks CNN, MLP, and LeNet on the FMNIST dataset and the CIFAR10



(a) MNIST (b) FMNIST (c) CIFAR10

Fig. 5. Comparison performance in local heterogeneous optimation of EASTER on three heterogeneous models and four optimization functions.(a) MNIST datasets; (b) FMNIST datasets; (c) CIFAR10 datasets.

TABLE 4
Comparison test accuracy of EASTER on three benchmark datasets, three heterogeneous models, and four heterogeneous optimizations.

Datasets	Optimization	Testing Accuracy (%)		
		CNN	LeNet	MLP
MNIST	SGD	91.50	96.48	95.96
	SGD_Momentum	91.65	96.67	95.92
	Adagrad	91.34	96.68	95.93
	Adam	91.25	96.67	95.31
FMNIST	SGD	85.86	84.14	86.45
	SGD_Momentum	85.76	83.88	86.19
	Adagrad	85.48	84.23	86.44
	Adam	83.91	84.06	86.02
CIFAR10	SGD	51.01	67.03	46.64
	SGD_Momentum	51.05	64.99	45.09
	Adagrad	51.01	63.17	46.94
	Adam	48.92	67.95	43.08

dataset, respectively. The results showed that each network converges under heterogeneous optimizers.

In addition, Fig.s 5(a)-5(c) demonstrated that the four participants used three heterogeneous local networks and four optimization algorithms to train their respective local models using the MNIST, FMNIST, and CIFAR10 datasets. The results showed that each participant's local heterogeneous model could converge to the convergence value. Therefore, the EASTER aggregated local knowledge, and heterogeneous networks and optimizers had little impact on the performance of EASTER. Each participant could autonomously train a local heterogeneous network with a heterogeneous optimizer.

The results of the test accuracy of EASTER under the heterogeneous model and heterogeneous optimizer were shown in Table 4. Under heterogeneous optimizers, the same model could achieve convergent test accuracy. This demonstrated that EASTER could achieve greater model accuracy using the same model but with a different optimizer for each participant. Moreover, heterogeneous models could obtain convergence accurately using the same optimizer. The study's findings indicated that every participant could accurately train a local heterogeneous model. Thus, EASTER enabled the incorporation of both local model heterogeneity and optimizer heterogeneity within the participants.

6 Related Work

6.1 Heterogeneous Model in Federated Learning

The heterogeneity model in FL refers to varied structures of participants' local model and heterogeneity is deemed to be one of the key challenges in FL [41], [42]. Recent research mostly can be grouped into two types, including Knowledge Distillation (KD)-based methods [43]–[46] and Partial Training (PT)-based methods [47]. A typical KD-based method extracts knowledge from a teacher model into student models with varied architectures. FedGKT [46] is a group knowledge transfer that regularly transfers client knowledge to the server's global model through knowledge distillation. However, the current methods [48] for KD need each client to share a set of basic samples to make distilled soft predictions. To remedy the data leakage caused by clients sharing input samples for KD. The work [45] proposed a federated distillation framework. The framework enables data-agnostic knowledge transmission between servers and clients via distributed *Generative Adversarial Networks* (GAN). A PT-based method extracts sub-models from the global server model [49]. FedRolex [47] is a partial training-based approach that achieves model heterogeneity in FL, such that it enables a larger global training model compared to the local model. We find that current methods for solving FL heterogeneous models are suitable for HFL, meaning that each participant owns sample labels and can independently carry out local model training. However, the passive party of VFL typically lacks labels and cannot independently train a local model. These methods may not always apply to solving the heterogeneous model problems in VFL.

6.2 Embedding Learning

A typical embedding learning [50], [51] mappings samples to a feature space and utilizes sample embeddings for classification or clustering tasks. An embedding represents a class by computing the average of feature vectors within each class, which enables effective classification or intra-cluster compactness by maximizing the distance between embeddings [52]–[54]. In recent years, embeddings has been extensively explored in various domains. In classification tasks [52], [55] and clustering tasks [56], embeddings represent a class by computing the average of feature vectors within each class, enabling effective classification or intra-cluster compactness by maximizing the distance between

embeddings. In clustering tasks, ProPos [56] aims to enhance representation uniformity and intra-cluster compactness by maximizing the distance between embeddings. In FL embeddings are used to address data heterogeneity issues. FedNH [57] utilizes uniform and semantic class embeddings to tackle class imbalance and improve local models' personalization and generalization. The work [22] and [29], [51] adopt embeddings to represent classes and use average federated embedding aggregation to improve the efficiency of model training. MPFed [58] uses multiple embeddings to represent a class and performs model inference by testing the distance between the target embedding and multiple embeddings. In our work, we focus on VFL in which data from all parties are aligned. We use a single-class embedding to represent each participant's local knowledge. By aggregating the local embeddings (e.g., embedding), we obtain knowledge from all participants that do not include model-specific details.

7 Conclusions

This paper proposed a novel method, EASTER, to address the challenge of poor model performance due to the party-local model heterogeneity. The key idea of EASTER was to leverage the aggregation of local embeddings rather than intermediate results to capture the local knowledge of all participants. To ensure data privacy, we employed secure aggregation techniques to obtain global embeddings while preserving the confidentiality of the participants' original data. In this paper, we conducted a comprehensive analysis of the effectiveness of EASTER, considering both theoretical aspects and extensive experimental evaluations. The research results demonstrated that EASTER can provide simultaneous training for multiple local heterogeneous models that exhibit good performance. In addition, EASTER assisted participants in improving their local heterogeneous models by utilizing local heterogeneous optimizers. Our method's practical applicability and advantages contribute to the development and broader adoption of VFL in real-world scenarios.

References

[1] Z. Pan, S. Wang, C. Li, H. Wang, X. Tang, and J. Zhao, "FedMDFG: Federated learning with multi-gradient descent and fair guidance," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, pp. 9364–9371.

[2] Z. Xiong, W. Li, and Z. Cai, "Federated generative model on multi-source heterogeneous data in IoT," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, pp. 10 537–10 545.

[3] X. Tang, C. Guo, K. R. Choo, and Y. Liu, "An efficient and dynamic privacy-preserving federated learning system for edge computing," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 207–220, 2024.

[4] B. McMahan, E. Moore, D. Ramage, and et al, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, 2017, pp. 1273–1282.

[5] J. So, R. E. Ali, B. Güler, and et al, "Securing secure aggregation: Mitigating multi-round privacy leakage in federated learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, pp. 9864–9873.

[6] M. Gong, Y. Zhang, Y. Gao, A. K. Qin, Y. Wu, S. Wang, and Y. Zhang, "A multi-modal vertical federated learning framework based on homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 1826–1839, 2024.

[7] Q. Zhang, B. Gu, C. Deng, and H. Huang, "Secure bilevel asynchronous vertical federated learning with backward updating," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, pp. 10 896–10 904.

[8] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, "Federated learning: A survey on enabling technologies, protocols, and applications," *IEEE Access*, vol. 8, pp. 140 699–140 725, 2020.

[9] Q. Li, Z. Wen, Z. Wu, and et al, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 3347–3366, 2023.

[10] W. Xia, Y. Li, L. Zhang, Z. Wu, and X. Yuan, "Cascade vertical federated learning," in *2022 IEEE International Conference on Multimedia and Expo (ICME)*, 2022, pp. 1–6.

[11] Y. Wu, N. Xing, G. Chen, and et al, "Falcon: A privacy-preserving and interpretable vertical federated learning system," *Proceedings of the VLDB Endowment*, vol. 16, no. 10, pp. 2471–2484, 2023.

[12] X. Jin, P.-Y. Chen, C.-Y. Hsu, and et al, "Catastrophic data leakage in vertical federated learning," in *Advances in Neural Information Processing Systems*, 2021, pp. 994–1006.

[13] I. Ceballos, V. Sharma, E. Mugica, and et al, "SplitNN-driven vertical partitioning," *arXiv preprint*, vol. PP, no. 99, p. 1, 2020.

[14] P. Wei, H. Dou, S. Liu, R. Tang, L. Liu, L. Wang, and B. Zheng, "FedAds: A benchmark for privacy-preserving CVR estimation with vertical federated learning," in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023, pp. 3037–3046.

[15] C. Fu, X. Zhang, S. Ji, and et al, "Label inference attacks against vertical federated learning," in *31st USENIX security symposium (USENIX Security 22)*, 2022, pp. 1397–1414.

[16] D. Romanini, A. J. Hall, P. Papadopoulos, and et al, "Pyvertical: A vertical federated learning framework for multi-headed splitNN," *arXiv*, vol. PP, no. 99, p. 1, 2021.

[17] S. Li, D. Yao, and J. Liu, "FedVS: Straggler-resilient and privacy-preserving vertical federated learning for split models," in *International Conference on Machine Learning*, 2023, pp. 20 296–20 311.

[18] T. J. Castiglia, A. Das, S. Wang, and S. Patterson, "Compressed-VFL: Communication-efficient learning with vertically partitioned data," in *International Conference on Machine Learning*, 2022, pp. 2738–2766.

[19] R. Xu, N. Baracaldo, Y. Zhou, and et al, "Fedv: Privacy-preserving federated learning over vertically partitioned data," in *Proceedings of the 14th ACM workshop on artificial intelligence and security*, 2021, pp. 181–192.

[20] T. Castiglia, S. Wang, and S. Patterson, "Flexible vertical federated learning with heterogeneous parties," *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, no. 99, pp. 1–15, 2023.

[21] J. Zhang, S. Guo, Z. Qu, D. Zeng, H. Wang, Q. Liu, and A. Y. Zomaya, "Adaptive vertical federated learning on unbalanced features," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 4006–4018, 2022.

[22] Y. Tan, G. Long, L. Liu, T. Zhou, Q. Lu, J. Jiang, and C. Zhang, "Fedproto: Federated prototype learning across heterogeneous clients," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, pp. 8432–8440.

[23] Q. Zhang, B. Gu, C. Deng, and et al, "Asysqn: Faster vertical federated learning algorithms with better computation resource utilization," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 3917–3927.

[24] X. Fang and M. Ye, "Robust federated learning with noisy and heterogeneous clients," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 072–10 081.

[25] K. Wang, Q. He, F. Chen, and et al, "FlexiFed: Personalized federated learning for edge clients with heterogeneous model architectures," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 2979–2990.

[26] L. Liu, W. L. Hamilton, G. Long, J. Jiang, and H. Larochelle, "A universal representation transformer layer for few-shot image classification," in *9th International Conference on Learning Representations, ICLR 2021*. Virtual Event: OpenReview.net, 2021, pp. 1–12.

[27] Y. Qiao, S.-B. Park, S. M. Kang, and et al, "Prototype helps federated learning: Towards faster convergence," *arXiv*, vol. PP, no. 99, p. 1, 2023.

[28] Y. Ma, J. Woods, S. Angel, and et al, "Flamingo: Multi-round single-server secure aggregation with applications to private federated learning," in *2023 IEEE Symposium on Security and Privacy (SP)*, 2023, pp. 477–496.

[29] Y. Tan, G. Long, J. Ma, L. LIU, T. Zhou, and J. Jiang, "Federated Learning from Pre-Trained Models: A Contrastive Learning Approach," in *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2022, pp. 19 332–19 344.

[30] H. Gao, A. Xu, and H. Huang, "On the convergence of communication-efficient local SGD for federated learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, pp. 7510–7518.

[31] Y. Zheng, S. Lai, Y. Liu, and et al, "Aggregation service for federated learning: An efficient, secure, and more resilient realization," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 2, pp. 988–1001, 2023.

[32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[33] X. Han, R. Kashif, and V. Roland, "Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. PP, no. 99, p. 1, 2017.

[34] A. Krizhevsky, *Learning multiple layers of features from tiny images*. ON, Canada: Toronto, 2009.

[35] P. Moallem and S. A. Ayoughi, "Removing potential flat spots on error surface of multilayer perceptron (MLP) neural networks," *International Journal of Computer Mathematics*, vol. 88, no. 1, pp. 21–36, 2011.

[36] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology*, 2017, pp. 1–6.

[37] Y. LeCun *et al.*, "LeNet-5, convolutional neural networks," *URL: http://yann.lecun.com/exdb/lenet*, vol. 20, no. 5, p. 14, 2015.

[38] P. Adam, G. Sam, M. Francisco, L. Adam, B. James, C. Gregory, K. Trevor *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019, pp. 8024–8035.

[39] W. Liu, L. Chen, Y. Chen, and W. Zhang, "Accelerating federated learning via momentum gradient descent," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 8, pp. 1754–1766, 2020.

[40] A. Défossez, L. Bottou, F. R. Bach, and N. Usunier, "A simple convergence proof of Adam and Adagrad," *Transactions on Machine Learning Research*, vol. PP, no. 99, p. 1, 2022.

[41] L. Hu, H. Yan, L. Li, Z. Pan, X. Liu, and Z. Zhang, "MHAT: An efficient model-heterogenous aggregation training scheme for federated learning," *Information Sciences*, vol. 560, no. 99, pp. 493–503, 2021.

[42] W. Huang, M. Ye, and B. Du, "Learn from others and be yourself in heterogeneous federated learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 143–10 153.

[43] Z. Zhu, J. Hong, and J. Zhou, "Data-free knowledge distillation for heterogeneous federated learning," in *International conference on machine learning*, 2021, pp. 12 878–12 889.

[44] Y. J. Cho, J. Wang, T. Chirvolu, and G. Joshi, "Communication-efficient and model-heterogeneous personalized federated learning via clustered knowledge transfer," *IEEE Journal of Selected Topics in Signal Processing*, vol. 17, no. 1, pp. 234–247, 2023.

[45] J. Zhang, S. Guo, J. Guo, D. Zeng, J. Zhou, and A. Zomaya, "Towards data-independent knowledge transfer in model-heterogeneous federated learning," *IEEE Transactions on Computers*, vol. 72, no. 10, pp. 2888–2901, 2023.

[46] C. He, M. Annavaram, and S. Avestimehr, "Group knowledge transfer: Federated learning of large cnns at the edge," in *Advances in Neural Information Processing Systems*, 2020, pp. 14 068–14 080.

[47] S. Alam, L. Liu, M. Yan, and M. Zhang, "FedRolex: Model-heterogeneous federated learning with rolling sub-model extraction," in *Advances in Neural Information Processing Systems*, 2022, pp. 29 677–29 690.

[48] T. Gao, X. Jin, and Y. Lai, "Model heterogeneous federated learning for intrusion detection based on knowledge distillation," in *Proceedings of the 2022 12th International Conference on Communication and Network Security*. Beijing, China: ACM, 2022, pp. 30–35.

[49] H. Wu, P. Wang, and A. C. Narayan, "Model-heterogeneous federated learning with partial model training," in *2023 IEEE/CIC International Conference on Communications in China (ICCC)*, 2023, pp. 1–6.

[50] H. Yang, X. Zhang, F. Yin, and C. Liu, "Robust classification with convolutional prototype learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3474–3482.

[51] G. Li, V. Jampani, L. Sevilla-Lara, D. Sun, J. Kim, and J. Kim, "Adaptive prototype learning and allocation for few-shot segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8334–8343.

[52] U. Michieli and P. Zanuttigh, "Continual semantic segmentation via repulsion-attraction of sparse and disentangled latent representations," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 1114–1124.

[53] H. Wu, B. Zhang, C. Chen, and J. Qin, "Federated semi-supervised medical image segmentation via prototype-based pseudo-labeling and contrastive learning," *IEEE Transactions on Medical Imaging*, vol. PP, no. 99, p. 1, 2023.

[54] A. Wang, L. Yang, H. Wu, and Y. Iwahori, "Heterogeneous defect prediction based on federated prototype learning," *IEEE Access*, vol. 11, pp. 98 618–98 632, 2023.

[55] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 1–11.

[56] Z. Huang, J. Chen, J. Zhang, and H. Shan, "Learning representation for clustering via prototype scattering and positive sampling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 6, pp. 7509–7524, 2022.

[57] Y. Dai, Z. Chen, J. Li, S. Heinecke, L. Sun, and R. Xu, "Tackling data heterogeneity in federated learning with class prototypes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, pp. 7314–7322.

[58] Y. Qiao, M. S. Munir, A. Adhikary, A. D. Raha, S. H. Hong, and C. S. Hong, "A framework for multi-prototype based federated learning: Towards the edge intelligence," in *International Conference on Information Processing in Sensor Networks*, 2023, pp. 134–139.