FedFR: Evaluation and Selection of Loss Functions for Federated Face Recognition

Ertong Shang, Zhuo Yang, Hui Liu(✉), Junzhao Du, and Xingyu Wang

Xidian University, Xi'an 710126, Shaanxi, China
{etshang,zhuo_yang}@stu.xidian.edu.cn, {liuhui,dujz}@xidian.edu.cn

Abstract. With growing concerns about data privacy and the boom in mobile and ubiquitous computing, federated learning, as an emerging privacy-preserving collaborative computing approach, has been receiving widespread attention recently. In this context, many clients collaboratively train a shared global model under the orchestration of a remote server, while keeping the training data localized. To achieve better federated learning performance, the majority of existing works have focused on designing advanced learning algorithms, such as server-side parameter aggregation policies. However, the local optimization on client devices, especially selecting an appropriate loss function for local training, has not been well studied. To fill this gap, we construct a federated face recognition prototype system and test five classical metric learning methods(i.e. loss functions) in this system, comparing their practical performance in terms of the global model accuracy, communication cost, convergence rate, and resource occupancy. Extensive empirical studies demonstrate that the relative performance between these approaches varies greatly in different federated scenarios. Specifically, when the number of categories to recognize on each client is large, using the classification-based loss function can make a better global model faster with less communication cost; while when there are only a few classes on each client, using the pair-based method can be more communication-efficient and obtain higher accuracy. Finally, we interpret this phenomenon from the perspective of similarity optimization and offer some suggestions on making suitable choices amongst various loss functions.

Keywords: Federated learning · Face recognition · Loss function · Metric learning

1 Introduction

In the last few years, fueled by advances in big data, processing power, and algorithms, Artificial Intelligence (AI), especially Deep Learning (DL), has achieved great breakthroughs in a wide range of applications. However, conventional single-machine model training requires the great amount of data to be centralized in a cloud server or a data center to produce effective inference models, which is facing unprecedented challenges for the following reasons. On the one

H. Gao et al. (Eds.): CollaborateCom 2022, LNICST 460, pp. 95–114, 2022.
https://doi.org/10.1007/978-3-031-24383-7_6

hand, nations across the world are strengthening laws to protect users' privacy and data security, posing new challenges to the data-transaction procedures commonly used today in AI [38]. On the other hand, the popularity of networked devices has led to an exponential increase in data generated at the edge of networks. As a consequence, existing cloud-based AI is gradually unable to manage such massively distributed computing power and analyze these data.

Thanks to the rapid advancement of computational and storage capabilities of edge smart devices and wireless communication technologies such as 5G, federated learning provides a potential solution to this dilemma. Federated Learning (FL), first proposed by Google [19], is a privacy-preserving distributed machine learning setting where many clients (e.g. resource-constrained edge devices or whole organizations) collaboratively train a model under the orchestration of a central server(e.g. a service provider) while keeping the training data decentralized [12]. Since the data never leave the data owners' devices, FL not only mitigates potential privacy leakage risks but also relieves computation and storage burden on servers. Other advantages of FL, as described in [40], include: reducing communication overhead by avoiding massive data uploads; enabling a global model that applies to different scenarios.

Because of the aforementioned great benefits, FL has attracted widespread attention recently. Among the existing researches, a large number of works focused on improving the performance and generalization capabilities of federated learning models by designing advanced learning algorithms [11,35,36], reducing communication costs [27] and enhancing privacy security of federated learning [22]. However, only a few works dealt with its implementation to real-world large-scale applications since the majority of existing works are carried out on small-scale datasets, e.g. MNIST, CIFAR, and their variants [8]. More importantly, the local optimization on client devices, especially selecting an appropriate loss function for local training, which determines the fundamental performance of the FL system, remain unexplored.

On the other hand, although a range of loss functions have been compared and studied on different tasks [3,23,28], they are all conducted in the traditional centralized training scenario, whose data characteristics are different from those in FL. For example, millions of images composed of tens of thousands of individuals are gathered together in the traditional face recognition task which thus can be regarded as a very large-scale classification problem. But in federated face recognition, each client may only contain images of dozens of people. The performance of these loss functions on such small-scale data is still unknown, which is also the motivation of this work.

Accordingly, in this paper, we focus on the implementation and evaluation of FL in a real large-scale application, face recognition. Meanwhile, we introduce the latest progress of metric learning into FL and explore the system performance in detail when different loss functions are used for local training. Interestingly, we observed large differences in relative performance between these loss functions in different FL scenarios. In other words, the loss function should be carefully selected according to the data characteristics of application scenarios when using

FL. To the best of our knowledge, this is the first time to study and evaluate the performance of different loss functions in FL settings. The main contributions of this research are as follows.

- Firstly, we construct a large-scale FL prototype system for face recognition, Federated Face Recognition (FedFR), and consider various application scenarios for it.
- Further, we evaluate and compare the performance of five classical loss functions in our federated face recognition system. The abundant experiments not only prove that the advances in metric learning are still effective in FL but also show that the performance of these methods varies greatly in different federated settings.
- Finally, we provide a similarity optimization-based interpretation for the observed results and draw important conclusions that would help the researchers in making suitable choices amongst various loss functions for federated face recognition.

2 Related Works

2.1 Face Recognition

Face recognition technology is a biometric technology, which is based on the identification of facial features of a person and is also one of the most important topics in computer vision and pattern recognition [14]. It can be categorized into closed-set or open-set settings. For the closed-set setting, all testing identities are predefined in the training set, thus can be addressed as a classification problem. For the open-set setting, the testing identities are not seen during training. Therefore it is usually seen as a metric learning problem in which faces must be mapped to a discriminative embedding space. In the past, traditional machine learning algorithms were mainly used for face recognition, such as the Eigenfaces [31], Bayesian face [20], support vector machine based [6], etc. Recently, with the development of deep neural networks, there have been significant advances in deep learning-based face recognition technology [25,29,34]. However, deep learning-based approaches need to collect a large number of high-quality face image centrally for training, which is very difficult now due to the emphasis on data privacy and security. Fortunately, FL is suitable to cope with this dilemma.

Besides the advances in deep model architectures and the appearance of some public face datasets, the design of powerful loss functions [4,17,25,29,32–34] is another major factor for the great success of face recognition and is also a research hotspot in recent years. In general, loss functions in metric learning can be divided into classification-based and pair-based types. Classification-based methods train an efficient feature extractor by correctly classifying samples with class-level labels, while pair-based ones learn embeddings directly via optimizing the similarity between samples with pair-wise labels. Although both the types of approaches have been extensively studied in metric learning, their performance in federated settings is still unknown, which is the focus of this work.

2.2 Federated Learning

Federated Learning is a distributed machine learning approach for training models from decentralized data residing on remote devices. In recent years, many efforts have been made to address various challenges in it, such as statistical heterogeneity, expensive communication, and poisoning attacks [12,16]. For statistical heterogeneity, FedProx proposed adding a proximal term to the local objective to help ensure convergence in statistically heterogeneous settings [24], while MOON introduced model-contrastive federated learning, correcting the local updates by maximizing the similarity of representation learned by the local model and the representation learned by the global model [15]. For efficient communication, McMahan et al. [18] proposed the federated averaging algorithm (FedAvg) to reduce the number of communication rounds, which is the most commonly used method for FL now. On the other hand, Konecny et al. [13] focused on communication compression techniques to lower the amount of data traffic in a single round. Finally, for privacy protection, Bonawitz et al. introduced a secure multiparty computation protocol to protect individual model updates [1]. Geyer et al. proposed an algorithm for client-sided differential privacy-preserving federated optimization to tackle differential attacks [5]. However, the core of this paper is the local optimization on client devices, especially the selection of loss functions in different federated settings, which is compatible with and complementary to these techniques.

3 Classical Loss Functions

As shown in Sect. 2.1, many excellent loss functions have been proposed, which made great advances in recognition accuracy. However, the performance of these methods in federated settings is still unknown, so we decide to study them in detail in FedFR. Next, we will introduce the five loss functions studied in this paper, namely, Softmax Loss [29], Large Margin Cosine Loss(CosFace Loss) [34], ArcFace Loss [4], Triplet Loss [25] and Multi-Similarity Loss(MS Loss) [37]. Softmax Loss and its latest variants, ArcFace and CosFace Loss, are classification-based methods. Triplet Loss is usually regarded as the baseline of pair-based methods, and MS Loss is one of the latest developments in this category.

3.1 Classification-Based Loss Functions

Softmax Loss. Softmax Loss is the most widely used classification loss function in deep learning. In particular, it consists of a classifier layer followed by a multi-class cross-entropy loss. It is presented as follows:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T x_i + b_j}} \tag{1}$$

where W and b are the weight matrix and bias vector of the last layer, that is, the classifier layer. x_i and y_i denote the deep feature and the ground-truth label

of the i-th training sample respectively. N is the batch size, n is the total class number. W_j and b_j are the j-th column of W and the j-th item of b respectively, which correspond to the j-th class.

By simply fixing the bias $b = 0$ and further normalizing the deep feature as well as the columns of the weight matrix, NormFace [33] reformulate Eq. (1) as:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{s\,cos(\theta_{y_i,i})}}{\sum_{j=1}^{n} e^{s\,cos(\theta_{j,i})}} \tag{2}$$

where $\theta_{j,i}$ is the angle between W_j and x_i, and s is a large constant to prevent the gradient from getting too small in the training phase.

CosFace Loss. However, the embedding features learned by Softmax Loss and its normalized version are not sufficiently discriminative because they only penalize classification errors. To address this issue, CosFace Loss introduces an additive cosine margin $m(m \geq 0)$ to the classification boundary, described by:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{s\,(cos(\theta_{y_i,i})-m)}}{e^{s\,(cos(\theta_{y_i,i})-m)} + \sum_{j\neq y_i} e^{s\,cos(\theta_{j,i})}} \tag{3}$$

ArcFace Loss. Similarly, ArcFace Loss adds an additive angular margin penalty $m(m \geq 0)$ between x_i and W_{y_i} to simultaneously enhance the intra-class compactness and inter-class discrepancy. It is presented as follows:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{s\,(cos(\theta_{y_i,i}+m))}}{e^{s\,(cos(\theta_{y_i,i}+m))} + \sum_{j\neq y_i} e^{s\,cos(\theta_{j,i})}} \tag{4}$$

3.2 Pair-Based Loss Functions

Triplet Loss. Triplet Loss aims to learn discriminative feature embeddings using embedding triplets. A triplet consists of an anchor, a positive sample, and a negative one, where the positive shares the same label as the anchor while the negative comes from other classes. Each triplet tries to tries to enforce the anchor-positive distance to be smaller than the anchor-negative distance by a predefined margin m. It can be described as follows:

$$L = \sum_{i=1}^{N}\left[\|x_i^a - x_i^p\|_2^2 - \|x_i^a - x_i^n\|_2^2 + m\right]_+ \tag{5}$$

where N is the number of triplets constructed in a mini-batch, x_i^a, x_i^p and x_i^n denote the feature embedding of the anchor, positive and negative sample in the i-th triplet, respectively.

MS Loss. How to improve sampling schemes to construct pairs with more information is the key to embedding-based methods. To meet this challenge, [37] casts the sampling problem into a general pair weighting formulation and considers

three similarities, defined as self-similarity, positive relative similarity, and negative relative similarity, for pair mining and weighting. Where the positive relative similarity is used for pair mining, while the self-similarity and negative relative similarity are jointly used for weighting the selected pairs. On this basis, they proposes MS loss:

$$L = \frac{1}{N}\sum_{i=1}^{N}\left\{\frac{1}{\alpha}log\Big[1+\sum_{k\in\mathcal{P}_i}e^{-\alpha(S_{ik}-\lambda)}\Big]+\frac{1}{\beta}log\Big[1+\sum_{k\in\mathcal{N}_i}e^{\beta(S_{ik}-\lambda)}\Big]\right\} \quad (6)$$

where N is the number of samples in a mini-batch, S_{ik} denotes the similarity between x_i and x_k. $\mathcal{P}_i$ and $\mathcal{N}_i$ denote the index set of selected positive pairs and negative pairs for an anchor x_i. λ, α, β are fixed hyper-parameters.

4 System Design and Performance Metrics

In this section, we will introduce our federated face recognition system, FedFR. Specifically, we start with the architecture and learning protocol of FedFR before introducing its application scenarios. Next, we show the performance metrics we defined to evaluate the performance of different loss functions, including effectiveness, communication efficiency, convergence rate, and memory usage.

4.1 System Overview

As shown in Fig. 1, the federated face recognition prototype system, FedFR, is designed based on the Client-Server architecture implementing the FedAvg [18] algorithm. In this system, K participants, called clients, collaboratively learn a global face recognition model under the coordination of a central server. Clients are data owners and they train local models using their own data and computation resources. The server maintains a global model and updates it by aggregating the local models periodically. It is worth noting that each client's raw data is stored locally and not exchanged or transferred. Correspondingly, there is no data stored and no training performed in the central server.

We use FedAvg as the learning protocol of FedFR. Its main process is as follows:

(1) Global Parameter Broadcasting: Server selects some clients and broadcasts the global model parameters to them.
(2) Local Model Training: Each selected client independently trains its local model using the local data.
(3) Local Parameter Backing: Clients send back their updated local parameters to the server.
(4) Model Aggregation: Server aggregates the received parameters using a certain aggregation algorithm and updates the global model.

The four main steps are iterated until the global model achieves the desired performance or completes a specified number of iterations.

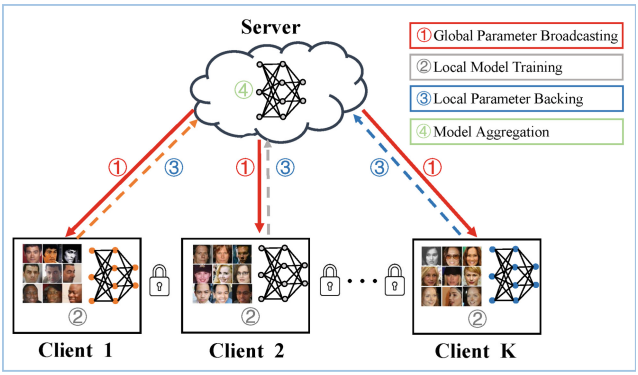


Fig. 1. The architecture of FedFR.

4.2 Application Scenarios

We consider two typical application scenarios for FedFR: Federated-by-organizations and Federated-by-devices. The main differences between them lie in the number of clients, the hardware capacity of a single client, and the amount of data stored on each client.

Federated-by-Organizations Scenario. In this scenario, clients are companies or organizations with massive data. The number of clients is usually relatively small and each client has great computational and communication powers. For example, several companies in a building can use this system to jointly learn a face recognition model for the building's access control.

Federated-by-Devices Scenario. In this scenario, clients are usually defined as mobile devices or embedded devices. Contrary to the Federated-by-organizations scenario, each device usually has a relatively small amount of data as well as limited computational and communication powers in this setting, so the devices cannot afford to perform much computation to train a huge model. Moreover, due to a large number of clients, only a small fraction of devices perform computation during each round of training. For example, a large number of cameras in different locations can be viewed as clients to build an intelligent safe-guard system that uses face recognition technology, without uploading their privacy sensitive images.

4.3 Performance Metrics

Test Accuracy is used to evaluate the effectiveness of each method. By following the standard face recognition evaluation protocol, we test the face verification accuracy of the global model at intervals and record all results during training. And then, we report the highest test accuracy.

Communication Rounds. Communication cost can be calculated by multiplying the number of communication rounds and the amount of data traffic in a

single round. While in this paper, we use the number of communication rounds when the global model reaches the desired accuracy (**A**) for the first time (**R@A**) to measure communication efficiency because the data traffic in each round is the same.

Local Training Time of each global iteration refers to the time required for all clients selected in this round to complete their local training. It, together with the number of global iterations determines the convergence rate of the system. In a real-world FL system, Local Training Time depends on the slowest one of all clients selected in that iteration.

Memory Usage refers to the amount of memory used during local training.

5 Experimental Setup

In this section, we describe the experimental setup in detail, including the simulation environment (Sect. 5.1), datasets and data preprocessing (Sect. 5.2), network architectures we used (Sect. 5.3), and some implementation details (Sect. 5.4).

5.1 Simulation Environment

We simulate the FL setup (one server and K clients) on a commodity machine with Intel Xeon Silver 4214 CPU @ 2.2 GHz and 3 NVIDIA GeForce RTX 2080Ti GPUs. All experiments are implemented using Pytorch with Python 3.6. We select a number of parameters to control the federated settings in our experiments. These parameters are: 1) K - number of devices, 2) C - clients' participation percentage in each round, 3) S - number of online clients in each round (i.e., $S = K \times C$), 4) R - number of communication rounds, 5) E - number of local training epochs per round, 6) N - number of individuals (training classes) on each client.

5.2 Datasets and Preprocessing

Training Data. CASIA-WebFace [39] and MS-Celeb-1M (MS1M) [7] are the most widely used public datasets for face recognition, containing 0.49M face images from 10,575 different subjects and about 10M images from 100K celebrities, respectively. So we use the two datasets as our training sets. In this work, we take 10K individuals from CASIA-WebFace and 50K individuals from MS1M clean version respectively, and divide them into many small subsets according to their identities, where each subset represents the private dataset of one client in FedFR. According to the characteristics of the Federated-by-organizations scenario and Federated-by-devices scenario described in Sect. 3.2, we partition these individuals into multiple subsets in two manners, which is shown in Table 1. To eliminate the impact of unbalanced data distribution, each client simply has the same number of identities(i.e., classes) in this paper, and there is no identity overlap among clients.

Table 1. Characteristics of two different federated face recognition scenarios. **Ids/client** and **Imgs/client** represents the number of identities and the number of image samples per client, respectively. **Clients** indicates the number of clients in the corresponding scenario.

Dataset	Scenarios	Ids/client	Imgs/client	Clients
CASIA-WebFace	Federated-by-organizations	1,000	∼50K	10
	Federated-by-devices	20	∼1K	500
MS1M	Federated-by-organizations	1,000	∼53K	50
	Federated-by-devices	20	∼1.1K	2500

Test Data. During training, we use an efficient face verification dataset, Labeled Faces in the Wild (LFW) [10], to test the accuracy and to analyze the convergence of the global model on the server. The LFW dataset consists of 13,233 face images from 5749 persons from uncontrolled conditions. We use the standard verification protocol mentioned in [10] to evaluate 6,000 image pairs.

In addition, we also use CFP-FP [26] as well as AgeDB-30 [21], two more difficult test protocols, to evaluate the model accuracy. Celebrities in Frontal-Profile data set (CFP) [26] contains 12,557 face images of 500 celebrities in front and profile views. It contains two verification protocols: one comparing only frontal faces (CFP-FF), the other comparing frontal and profile faces (CFP-FP). Each protocol consists of 7000 comparisons, and we consider the more challenging verification protocols, CFP-FP, in our experiments. AgeDB [21] is a manually collected in-the-wild age database, containing 16,488 images of 568 various famous people. It is a more challenging face verification dataset since it with large range of ages for each subject. Generally, the original AgeDB contains four verification schemes, where the compared faces have an age difference of 5, 10, 20 and 30 years, respectively. In our experiments, we select the most challenging scheme(AgeDB-30) for evaluation, which also contains 6,000 comparisons.

For data prepossessing, we follow the most common treatment to generate the face crops (112×112) by utilizing five facial points. Each pixel ($[0, 255]$) in the RGB images is then normalized by subtracting 127.5 and divided by 128, without any other data augmentation.

5.3 Network Architectures

For the trunk network, we employ two widely used CNN architectures, ResNet and MobileFaceNet. It is worth noting that the classification-based method requires an additional classifier layer, which is always maintained in clients and does not need to be uploaded for aggregation because each column of the parameter matrix of the classifier layer represents the proxy of its corresponding class.

ResNet. The ResNet architecture was first proposed by He et al. in 2015 [9]. It has proven to be an effective network architecture for a wide variety of vision

tasks. In this paper, we adopt the ResNet50 model used in ArcFace to get a 512-D feature embedding for each image. This model contains 43 million parameters and has a computational cost of 6,309 million MAdds(the number of operations measured by multiply-adds) in total. Given limited computation resources, we only train ResNet-50 model on CASIA-WebFace.

MobileFaceNet. Chen et al. [2] presented a light face feature embedding CNN, MobileFaceNet, which aims to improve the efficiency for real-time face verification on mobile and embedded devices. It learns a more compact face embedding of 128 dimensions, using only about 1 million parameters and 227 million MAdds.

5.4 Implementation Details

We conduct abundant experiments in different federated settings to comprehensively compare the system performance with different loss functions. For the sake of fair comparison, we use the same hyper-parameter settings as their authors for all loss functions except ArcFace Loss, because the original setting(s is 64, m is 0.5) for ArcFace Loss can not converge well in our federated settings. After several attempts, we eventually set the feature scale s to 30 and choose the angular margin m of 0.3 for ArcFace Loss. For classification-based methods, the batch size is set to 64. For pair-based methods, we first randomly choose 16 classes and then randomly sample 4 instances from each class selected to form a mini-batch. As a comparison, we also train a model in the traditional centralized way using the loss functions described above. For all settings, the stochastic gradient descent(SGD) algorithm with momentum set to 0.9 and weight decay of 5e-4 is used as an optimizer. Starting from 0.1, the learning rate decreases by 10% every 5 rounds for all federated settings.

For centralized training, the number of training epochs is set to 20 for both datasets. For Federated-by-organizations scenario, we set $R = 80, E = 1, S = 10$ (i.e., $K = 10, C = 1.0$ for CASIA-WebFace; $K = 50, C = 0.2$ for MS1M). For Federated-by-devices scenario, we set $R = 200, E = 10$ for both dataset, $S = 25$ (i.e., $K = 500, C = 0.05$) for CASIA-WebFace; $S = 50$ (i.e., $K = 2500, C = 0.02$) for MS1M.

6 Performance Evaluation and Summary

In this section, we conduct a series of experiments in different federated settings to compare the loss functions mentioned above according to the performance metrics we defined. By analyzing the abundant results, we gain some meaningful conclusions. More importantly, some suggestions are given for selecting appropriate loss functions in federated face recognition.

6.1 Comparison of Test Accuracy

Table 2 and Table 3 show the test accuracy of the models trained with CASIA-WebFace and MS1M dataset, respectively. We can see that for each loss function,

the accuracy achieved in federated settings is very close to that in the centralized setting. That is to say, the federated training protocol we adopted can obtain satisfactory results, which proves the effectiveness of FL for large-scale face recognition tasks.

Table 2. Highest test accuracy for both models trained with CASIA-Webface dataset. **Cent**, **Fed_O** and **Fed_D** represents centralized training, Federated-by-organizations scenario and Federated-by-devices scenario, respectively.

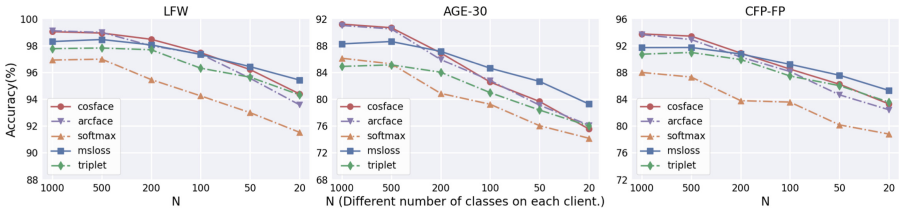
Model	Loss	LFW			CFP-FP			AgeDB-30		
		Cent	Fed_O	Fed_D	Cent	Fed_O	Fed_D	Cent	Fed_O	Fed_D
ResNet	CosFace	99.25	99.05	94.40	94.51	93.77	83.30	92.73	91.23	75.55
	ArcFace	99.17	99.12	93.58	94.34	93.67	82.43	92.63	91.03	76.12
	Softmax	98.82	96.93	91.53	92.51	88.00	78.80	89.65	86.13	74.17
	MS	98.40	98.31	95.40	92.24	91.73	85.31	89.08	88.28	79.28
	Triplet	97.98	97.78	94.33	91.20	90.74	83.57	86.25	84.90	76.00
MobileFaceNet	CosFace	99.00	98.70	92.57	92.73	92.00	80.03	91.43	89.87	74.00
	ArcFace	98.88	98.68	92.38	93.07	91.94	80.31	90.97	89.82	74.27
	Softmax	97.90	96.37	91.83	90.04	87.29	79.27	87.00	84.08	74.75
	MS	98.28	97.38	94.40	90.33	89.60	82.84	87.15	85.37	77.18
	Triplet	97.65	97.02	92.57	90.59	89.25	81.79	84.57	82.88	73.65

Table 3. Highest test accuracy for MobileFaceNet trained with MS1M dataset.

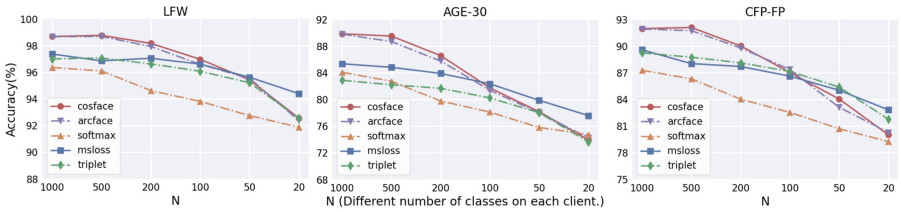
Model	Loss	LFW			AgeDB-30			CFP-FP		
		Cent	Fed_O	Fed_D	Cent	Fed_O	Fed_D	Cent	Fed_O	Fed_D
MobileFaceNet	CosFace	99.25	98.85	95.05	95.18	92.87	79.57	87.20	85.10	74.89
	ArcFace	99.28	98.77	94.57	95.45	91.70	79.68	88.39	84.37	74.30
	Softmax	98.97	96.30	93.65	92.93	85.42	78.55	85.56	77.11	73.46
	MS	98.82	97.93	96.43	92.20	88.97	83.85	87.17	83.36	78.75
	Triplet	98.57	97.78	95.97	90.67	87.20	82.67	86.73	83.35	78.20

We also compare the performance of different loss functions in FedFR. First of all, it can be seen that for classification-based loss functions, ArcFace and CosFace loss consistently outperform Softmax loss. Similarly, for pair-based ones, MS loss is always better than Triplet loss too. This is entirely in accord with the observation in the centralized setting. In other words, it demonstrates that the progress made in centralized metric learning is also applicable in FL.

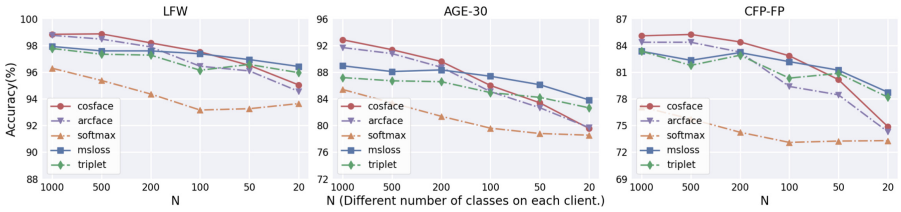
Another phenomenon is that the model trained in Federated_by_organizations scenario outperform its counterpart in Federated_by_devices scenario, which is true for all loss functions. However, what's surprising is that the performance of classification-based methods and pair-based ones varies greatly in these two scenarios. Specifically, in the Federated-by-organizations scenario, ArcFace and



(a) ResNet-50 trained with CASIA-WebFace.



(b) MobileFaceNet trained with CASIA-WebFace.



(c) MobileFaceNet trained with MS1M.

Fig. 2. Comparison of the highest test accuracies for both model using different loss functions in different federated settings. N on the horizontal axis indicates that the number of individuals on each client is N.

CosFace loss completely beat MS loss, while in the Federated-by-devices scenario, the opposite is true. Similarly, Triplet Loss, the baseline of pair-based methods, is also superior to its rival, Softmax loss, in the Federated-by-devices scenario. To further explore these observations, we conduct additional experiments, where the training set is divided into different number of subsets. Specifically, the training dataset is more exhaustively divided according to the number of classes (individuals) on each client which varies within the range of {1000, 500, 200, 100, 50, 20}. The results in all federated settings are shown in Fig. 2.

From Fig. 2, we can clearly observe a trend that as the amount of data on each client decreases, the test accuracy of the global model decreases, which is consistent on all three test sets. The reason is that the decrease in the amount of local data on each client increases the over-fitting degree of local models as well as the variance between local models, making model aggregation difficult. It is worth mentioning that the negative impact on the performance can be alleviated by some federated optimization techniques, such as using a more robust

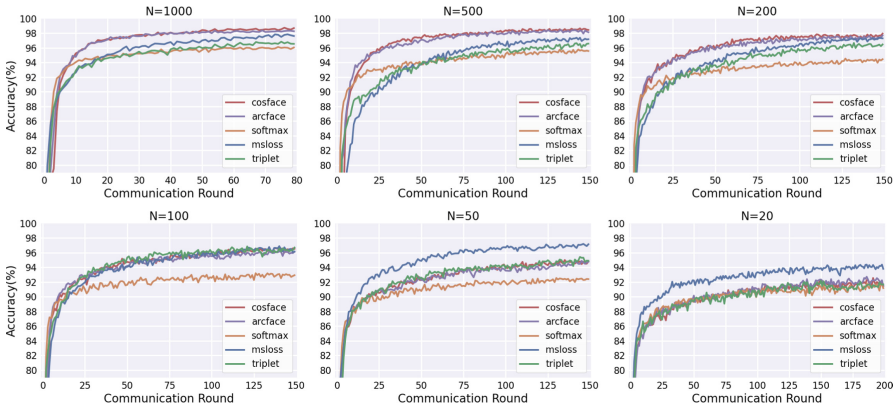


Fig. 3. Test accuracy of MobiFaceNet on LFW test set vs. number of communication rounds for the above five loss functions in different federated settings.

aggregation mechanism [36], introducing a proximal term to the local objective to reduce local bias [24], reducing the number of local epochs and etc., while it's not the point of this work. At the same time, we see that in the case of more data on each client, the performance of the classification-based loss function is better than that of the pair-based one, while the situation is just on the contrary in the case of less local data. To be specific, when each client contains 1000 or 500 individuals, the model accuracy using CosFace Loss and ArcFace Loss is completely superior to MS Loss. As the number of individuals decreased to 200 or 100, the three methods attain close results. Nevertheless, Ms Loss has a great advantage over CosFace and ArcFace Loss when the number of individuals is further reduced to 20. That is to say, with the decrease of the amount of local data on each client, the performance degradation of classification-based methods is more severe than that of pair-based ones.

6.2 Comparison of Communication Efficiency

As mentioned above, we measure communication efficiency in terms of **R@A**, i.e., the number of communication rounds when the global model reaches the desired accuracy (**A**) for the first time. For each federated setting, we set the values for (**A**) based on the lowest accuracy achieved among these loss functions. Table 4 reports the test results on LFW for the MobileFaceNet and ResNet-50 trained with CASIA-WebFace dataset. Moreover, Fig. 3 visualizes the training dynamics of MobileFaceNet using the five methods in different federated settings.

From the recorded results, we observe a similar trend in the communication cost comparison among these methods with the test accuracy comparison among them. First, in almost all settings, Cosface and ArcFace Loss have similar communication cost, which is less than Softmax, and the communication cost of MS Loss is less than that of Triplet Loss. Morever, it can be seen from the test accuracy curves with varing epochs in Fig. 3, in Cls_1000 (i.e., a thousand

Table 4. The number of communication rounds for first-time achievement of a certain test accuracy in different federated settings, where Cls_N represents there are N training classes on each client. The value in () denotes the deceleration computed against CosFace Loss.

Method	Cls_1000	Cls_500	Cls_200	Cls_100	Cls_50	Cls_20
	R@95%	R@95%	R@90%	R@90%	R@90%	R@90%
(1) ResNet-50						
CosFace	**10 (1.0×)**	**17 (1.0×)**	**10 (1.0×)**	**10 (1.0×)**	**17 (1.0×)**	**33 (1.0×)**
ArcFace	9 (0.9×)	19 (1.1×)	10 (1.0×)	11 (1.1×)	24 (1.4×)	43 (1.3×)
Softmax	23 (2.3×)	56 (3.3×)	15 (1.5×)	12 (1.2×)	31 (1.8×)	70 (2.3×)
Triplet	24 (2.4×)	48 (2.8×)	19 (1.9×)	16 (1.6×)	21 (1.2×)	41 (1.2×)
MS	**16 (1.6×)**	**32 (1.9×)**	**15 (1.5×)**	**12 (1.2×)**	**13 (0.8×)**	**19 (0.6×)**
(2) MobileFaceNet						
CosFace	**11 (1.0×)**	**21 (1.0×)**	**10 (1.0×)**	**11 (1.0×)**	**20 (1.0×)**	**60 (1.0×)**
ArcFace	12 (1.1×)	19 (0.9×)	9 (0.9×)	11 (1.0×)	20 (1.0×)	51 (0.9×)
Softmax	22 (2.0×)	61 (2.9×)	10 (1.0×)	15 (1.4×)	21 (1.0×)	66 (1.1×)
Triplet	26 (2.4×)	76 (3.6×)	17 (1.7×)	16 (1.5×)	19 (1.0×)	61 (1.0×)
MS	**21 (1.9×)**	**82 (4.3×)**	**19 (1.9×)**	**17 (1.5×)**	**13 (0.6×)**	**26 (0.4×)**

training classes per client), Cls_500, Cls_200 and Cls_100 federated settings, the classification-based approaches required fewer communication rounds (i.e. have more efficient communication) than the pair-based approaches to reach the same accuracy, whereas the reverse is true as the number of training classes per client decreased. For example, in the Cls_1000 setting, the communication cost of MS Loss is 1.9× compared to that of CosFace Loss. But in the Cls_20 setting, the communication cost of MS Loss is only 0.4× compared to that of CosFace Loss.

6.3 Comparison of Local Training Time

For all the loss functions, we also record the time required for all clients to complete their local training in each global iteration, called Local Training Time. In a real-world FL system, it is the time consumed by the slowest one of the all clients selected in each iteration. In this work, all the clients are simulated using the same desktop computer, thus we report the average time of all the clients completing a global iteration as the Local Training Time. Specifically, we compare the Local Training Time of MobileFaceNet and ResNet-50 model trained on different computing platforms (i.e., GPU or CPU).

Firstly, Fig. 4 demonstrates that for the same model, Local Training Time for classification-based loss functions is usually less than that for pair-based ones because the latter requires extra time to construct effective sample pairs or triplets. For example, when MobileFaceNet is used as the backbone network and trained on GPUs, Softmax, ArcFace, and CosFace loss functions have almost

the same Local Training Time, while in all federated settings, Local Training Time for Triplet and MS Loss are roughly its 2X and 4X, respectively. However, by comparing the results in Fig. 4(a) and Fig. 4(b), we find that as the network architecture changes, so do the exact ratio between the Local Training Time for these loss functions. More precisely, Local Training Time for Triplet and MS Loss is only about 1.2X and 1.7X what it is for classification-based loss functions when using ResNet50 and trained on GPUs. In addition, similar phenomenon can be observed when the same model is trained on different computing platforms. Specially, for both models, the Local Training Time required for these five loss functions is relatively close when trained on the CPU. In other words, Local Training Time is not only related to the selected loss function but also closely related to the model architecture and the hardware resources used.

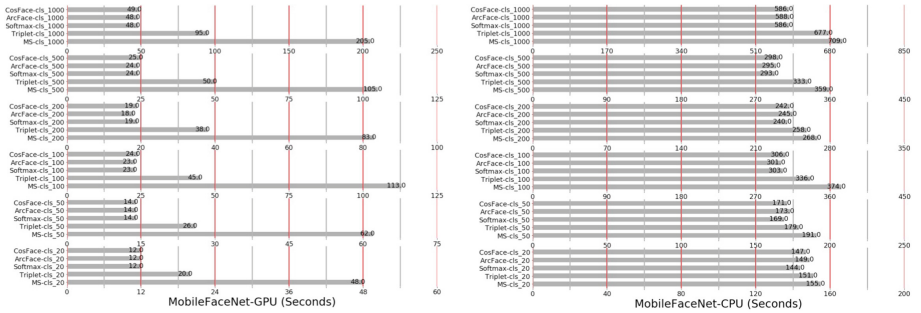
6.4 Comparison of Memory Usage

As described in Sect. 5.1, the classification-based method requires an additional classifier layer whose number of parameters is dependent on the size of the embedding and the number of classes contained on the client. For the Federated-by-devices scenario, each client contains only 20 different subjects, so there are only $20 * 128 = 2560$ and $20 * 512 = 10240$ additional parameters when using MobileFaceNet and ResNet50 respectively. Compared with the number of parameters of the backbone network(1M for MobileFaceNet and 43M for ResNet50), these additional parameters are completely negligible. Similarly, even for the Federated-by-organizations scenario that each client contains more individuals, the number of additional parameters introduced by the classifier layer is still insignificant. That is to say, for the two application scenarios we described, these loss functions are almost identical in terms of memory usage.

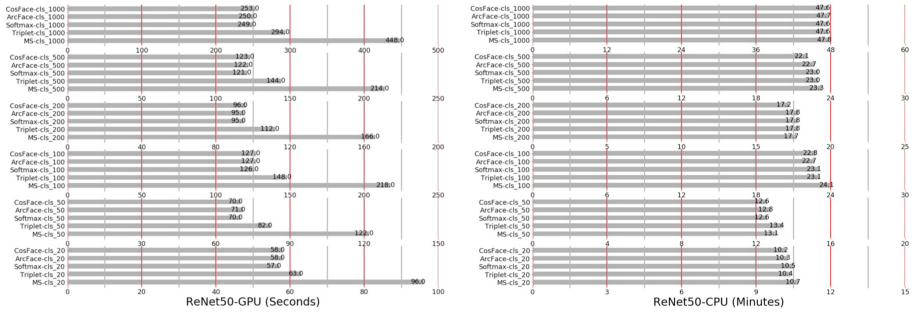
6.5 Summary and Suggestion

To sum up, based on the above observations, we first confirm the effectiveness of FedFR. Besides, we find a valuable rule that when the amount of data, especially the number of classes, is large on each client, using the classification-based loss function results in a more accurate global model faster with less communication cost. But when there is only a small number of classes on each client, using the pair-based one yields higher global accuracy with less communication cost instead, while the local training time of each iteration is longer than that of their rivals. In particular, in our federated face recognition experiment, this change occurs when the number of subjects on each client is about 100.

On this basis, some suggestions are given here for selecting appropriate loss functions in federated face recognition. When each client participating in federated training can collect large amounts of training data, the classification-based loss function, especially its latest developments, is the best choice. On the contrary, when there is only limited training data on each client, we should make choices based on the actual performance requirements. If the focus is on the final



(a) MobileFaceNet trained with CASIA-WebFace.



(b) ResNet-50 trained with CASIA-WebFace.

Fig. 4. Comparison of Local Training Time for models trained on CPU and GPU using different loss functions. The naming convention is as follows: Loss function-federated setting. **Cls_1000**: Each client performs E = 1 local epochs; **Cls_500**: Each client performs E = 1 local epochs; **Cls_200**: Each client performs E = 2 local epochs; **Cls_100**: Each client performs E = 5 local epochs; **Cls_50**: Each client performs E = 5 local epoch; **Cls_20**: Each client performs E = 10 local epoch.

accuracy and communication cost during training, the latest pair-based methods are preferred. If the model owner wants to get a usable model faster, which loss function to choose should be considered in conjunction with the computing power of the client and the network model they used.

7 Similarity Optimization Based Interpretation

Our evaluation reported above shows that the classification-based method and the pair-based method vary greatly in performance in different federated settings. This section attempts to provide a theoretical interpretation for this meaningful phenomenon.

According to [30], there is no intrinsic difference between the two methods from the perspective of similarity optimization. Specifically, they both aim to minimize between-class similarity S_n as well as to maximize within-class similarity S_p by embedding S_n and S_p into similarity pairs and seeking to reduce

$(S_n - S_p)$. But during the optimization process, the way they construct positive pairs and negative pairs which are used to calculate S_p and S_n respectively is completely different. For the classification-based method, similarity scores are calculated between the deep feature x and each weight vector of the classifier layer, w_i(also called the proxy for class i). Thus, 1 positive pair and $N-1$(N is the number of training classes) negative pairs can be built for each sample regardless of the size of mini-batches. While for the pair-based method, similarity scores between x and the other features in the same mini-batch are calculated, so the number of positive and negative pairs depend entirely on the number of samples contained in a mini-batch. For instance, if a mini-batch consists of M classes in which each class contains K samples, then $K-1$ positive pairs and $(M-1)K$ negative pairs can be formed for each sample. Therefore, for both the optimization approaches, the number of $(S_n - S_p)$ pairs in mini-batches of the same size may vary greatly.

More importantly, the classification-based method can obtain more global information in each optimization iteration because the similarity scores between x and all classes' proxies are calculated. But for the pair-based one, only limited local information can be seen since only similarity scores between x and other samples contained in the same mini-batch are calculated, which is a serious drawback for large-scale classification tasks. In general, when the total number of training classes is far greater than the number of classes contained in a mini-batch, using the classification-based loss function can not only yield more global information but also construct more $(S_n - S_p)$ pairs than the pair-based one. That is why it can converge faster and achieve higher accuracy in this case. However, when the number of training classes is not much larger than the number of classes contained in a mini-batch, the pair-based approach may outperform its counterpart because it can build more diverse sample pairs in a mini-batch, which becomes the decisive factor of performance.

8 Conclusion

In this paper, we construct a federated face recognition prototype system to evaluate and compare five commonly used metric learning methods in different federated settings. The abundant results not only demonstrate the effectiveness of FL for large-scale face recognition task but also show that the performance of these loss functions varies greatly in different federated settings. Finally, we explain this phenomenon from the perspective of similarity optimization and give some suggestions on the selection of loss functions in federated face recognition task. This work can serve as a useful reference for the researchers and practitioners working in federated metric learning.

For future work, we will deploy our FedFR in real-world environments that involve more challenges to be addressed, such as stricter privacy protection and system heterogeneity. In addition, a broader range of metric learning tasks and methods in FL can be further investigated in the future.

Acknowledgment. This work is partially supported by a grant from the National Natural Science Foundation of China (No. 62032017), the Fundamental Research Funds for the Central Universities, the Innovation Fund of Xidian University, the Key Industrial Innovation Chain Project in Industrial Domain of Shaanxi Province (No. 2021ZDLGY03-09, No. 2021ZDLGY07-02, No. 2021ZDLGY07-03) and The Youth Innovation Team of Shaanxi Universities.

References

1. Bonawitz, K., et al.: Practical secure aggregation for federated learning on user-held data. arXiv preprint arXiv:1611.04482 (2016)
2. Chen, S., Liu, Y., Gao, X., Han, Z.: MobileFaceNets: efficient CNNs for accurate real-time face verification on mobile devices. In: Zhou, J., et al. (eds.) CCBR 2018. LNCS, vol. 10996, pp. 428–438. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-97909-0_46
3. Chung, J.S., et al.: In defence of metric learning for speaker recognition. arXiv preprint arXiv:2003.11982 (2020)
4. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: Arcface: additive angular margin loss for deep face recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4690–4699 (2019)
5. Geyer, R.C., Klein, T., Nabi, M.: Differentially private federated learning: a client level perspective. arXiv preprint arXiv:1712.07557 (2017)
6. Guo, G., Li, S.Z., Chan, K.: Face recognition by support vector machines. In: Proceedings fourth IEEE International Conference on Automatic Face and Gesture Recognition (cat. no. PR00580), pp. 196–201. IEEE (2000)
7. Guo, Y., Zhang, L., Hu, Y., He, X., Gao, J.: MS-Celeb-1M: a dataset and benchmark for large-scale face recognition. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 87–102. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_6
8. He, C., et al.: FedML: a research library and benchmark for federated machine learning. arXiv preprint arXiv:2007.13518 (2020)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
10. Huang, G.B., Mattar, M., Berg, T., Learned-Miller, E.: Labeled faces in the wild: a database for studying face recognition in unconstrained environments. In: Workshop on faces in Real-Life Images: Detection, Alignment, and Recognition (2008)
11. Jiang, J., Ji, S., Long, G.: Decentralized knowledge acquisition for mobile internet applications. World Wide Web **23**(5), 2653–2669 (2020)
12. Kairouz, P., et al.: Advances and open problems in federated learning. Found. Trends® Mach. Learn. **14**(1–2), 1–210 (2021)
13. Konečnỳ, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T., Bacon, D.: Federated learning: strategies for improving communication efficiency. arXiv preprint arXiv:1610.05492 (2016)
14. Li, L., Mu, X., Li, S., Peng, H.: A review of face recognition technology. IEEE Access **8**, 139110–139120 (2020)
15. Li, Q., He, B., Song, D.: Model-contrastive federated learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10713–10722 (2021)

16. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: challenges, methods, and future directions. IEEE Signal Process. Mag. **37**(3), 50–60 (2020)
17. Liu, W., Wen, Y., Yu, Z., Li, M., Raj, B., Song, L.: Sphereface: deep hypersphere embedding for face recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 212–220 (2017)
18. McMahan, B., Moore, E., Ramage, D., Hampson, S., Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282. PMLR (2017)
19. McMahan, H.B., Moore, E., Ramage, D., Arcas, B.A.: Federated learning of deep networks using model averaging (2016)
20. Moghaddam, B., Jebara, T., Pentland, A.: Bayesian face recognition. Pattern Recogn. **33**(11), 1771–1782 (2000)
21. Moschoglou, S., Papaioannou, A., Sagonas, C., Deng, J., Kotsia, I., Zafeiriou, S.: AgeDB: the first manually collected, in-the-wild age database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 51–59 (2017)
22. Mothukuri, V., Parizi, R.M., Pouriyeh, S., Huang, Y., Dehghantanha, A., Srivastava, G.: A survey on security and privacy of federated learning. Futur. Gener. Comput. Syst. **115**, 619–640 (2021)
23. Musgrave, K., Belongie, S., Lim, S.-N.: A metric learning reality check. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12370, pp. 681–699. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58595-2_41
24. Sahu, A.K., Li, T., Sanjabi, M., Zaheer, M., Talwalkar, A., Smith, V.: On the convergence of federated optimization in heterogeneous networks. arXiv preprint arXiv:1812.06127 (2018)
25. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 815–823 (2015)
26. Sengupta, S., Chen, J.C., Castillo, C., Patel, V.M., Chellappa, R., Jacobs, D.W.: Frontal to profile face verification in the wild. In: 2016 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1–9. IEEE (2016)
27. Shahid, O., Pouriyeh, S., Parizi, R.M., Sheng, Q.Z., Srivastava, G., Zhao, L.: Communication efficiency in federated learning: achievements and challenges. arXiv preprint arXiv:2107.10996 (2021)
28. Srivastava, Y., Murali, V., Dubey, S.R.: A performance evaluation of loss functions for deep face recognition. In: Babu, R.V., Prasanna, M., Namboodiri, V.P. (eds.) NCVPRIPG 2019. CCIS, vol. 1249, pp. 322–332. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-8697-2_30
29. Sun, Y., Chen, Y., Wang, X., Tang, X.: Deep learning face representation by joint identification-verification. Adv. Neural. Inf. Process. Syst. **27**, 1988–1996 (2014)
30. Sun, Y., et al.: Circle loss: a unified perspective of pair similarity optimization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6398–6407 (2020)
31. Turk, M., Pentland, A.: Eigenfaces for recognition. J. Cogn. Neurosci. **3**(1), 71–86 (1991)
32. Wang, F., Cheng, J., Liu, W., Liu, H.: Additive margin softmax for face verification. IEEE Signal Process. Lett. **25**(7), 926–930 (2018)
33. Wang, F., Xiang, X., Cheng, J., Yuille, A.L.: Normface: L2 hypersphere embedding for face verification. In: Proceedings of the 25th ACM International Conference on Multimedia, pp. 1041–1049 (2017)

34. Wang, H., et al.: Cosface: large margin cosine loss for deep face recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5265–5274 (2018)
35. Wang, H., Yurochkin, M., Sun, Y., Papailiopoulos, D., Khazaeni, Y.: Federated learning with matched averaging. In: International Conference on Learning Representations (ICLR) (2020)
36. Wang, J., Liu, Q., Liang, H., Joshi, G., Poor, H.V.: Tackling the objective inconsistency problem in heterogeneous federated optimization. Adv. Neural. Inf. Process. Syst. **33**, 7611–7623 (2020)
37. Wang, X., Han, X., Huang, W., Dong, D., Scott, M.R.: Multi-similarity loss with general pair weighting for deep metric learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5022–5030 (2019)
38. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: concept and applications. ACM Trans. Intell. Syst. Technol. (TIST) **10**(2), 1–19 (2019)
39. Yi, D., Lei, Z., Liao, S., Li, S.Z.: Learning face representation from scratch. arXiv preprint arXiv:1411.7923 (2014)
40. Zhuang, W., et al.: Performance optimization of federated person re-identification via benchmark analysis. In: Proceedings of the 28th ACM International Conference on Multimedia, pp. 955–963 (2020)