Personalized Contrastive Federated Learning in Tackling Out of Distribution Unlabeled Data

Dongzhuyuan Lu
School of Science and Engineering
The Chinese University of Hong Kong, Shenzhen
Shenzhen, China
119010216@link.cuhk.edu.cn

***Abstract*—Federated learning (FL) is a paradigm that trains models on data across distributed clients and aggregates the local models on a central server. FL is important because it prevents private data from being shared with others and obviates the need for a large centralized dataset. However, two main difficulties remain for FL to be addressed: 1) non-iid datasets, and 2) label deficiency. Contrastive learning is a self-supervised learning (SSL) paradigm aiming to learn meaningful representations without needing adequately annotated raw data. In this work, we present a personalized contrastive federated learning model to tackle the above challenges. The core of the proposed method is to introduce a regularizer into the loss function of each client to minimize the distance between local model and global model. We employ this personalized federated learning method (*PFL*) to the existing contrastive learning frameworks including *SimCLR*, *BYOL*, and *SimSiam* in the pre-training process. Then, we measure the performance by *linear evaluation*. Experiments on various datasets like *CIFAR-10* and *CIFAR-100* are conducted to reaffirm and validate some previous results without a regularizer to delve deeper into the insights of FL-SSL. Moreover, we compare the results with and without a regularizer, and discover that our *PFL* method can slightly improve the accuracy. We delve into this finding in the discussion section, and we hope this paper will provide useful insights for future research.**

***Index Terms*—Federated learning, contrastive learning, data heterogeneity.**

I. Introduction

FEDERATED learning (FL) is a distributed framework in which datasets are trained locally, and the results on each device are combined together to form a generalized model [1]. In a typical FL model *FedAvg* [1], the local training and aggregation process repeats round-by-round on each decentralized client until convergence. In one of the rounds, the server selects part of the clients, informs them of the latest model, and performs stochastic gradient descent (SGD) to update the model's parameters. While traditional FL shows stronger generalizability and attains better performance in fields with distributed and data privacy setting such as medical applications [11], FL still faces certain serious challenges. An inevitable and practical challenge is the data heterogeneity problem, also known as non-iid (identically and independently distributed) data. The main non-iid data types can be summarized as follows: covariate shift, prior probability shift, concept shift, unbalancedness, etc. Non-iid data is almost ubiquitous in real-life circumstances. For example, a prior probability shift can occur when datasets containing images of flora vary significantly from region to region; large hospitals having more patient data than local clinics can result in unbalanced datasets [5]. Some efforts have been made by current research to tackle the problem of non-iid (out of distribution) data via agent clustering, where *FOCUS* (based on expected-maximization algorithm) is proposed [7]. In the research, however, data used for training was annotated, upon which the state-of-the-art supervised convolutional neural networks (CNN) is based [12]. Unfortunately, labels do not always exist in many practical applications. Street cameras and personal phones, for instance, are generating an enormous amount of unlabeled images with high privacy requirements [8].

Recently developed methods like *FedU* based on self-supervised learning (SSL) can learn meaningful representations from unlabeled data [16]. The current SSL approaches mainly belong to two categories: contrastive learning and non-contrastive (generative) learning. Non-contrastive learning methods, such as masked autoencoder and adversarial learning [17][18], learn representations via generating pixels for mapping. One recent work uses a *generative adversarial network (GAN)* to generate pseudo images and applies FL-SSL where the results are promising [13]. But another generative method–masked autoencoder–seems even more powerful. Transformer has become an important approach in natural language processing (NLP) [21], recently being applied to computer vision tasks [2]. Vision Transformers (ViT) shows its superior performance on image classification problems compared to CNN-based frameworks [19]. Due to the good performance of ViT-based masked autoencoder (MAE), in which partial patches of the input images are masked to train an encoder and decoder among clients collaboratively, researchers have incorporated this framework into federated learning [5][8]. Nevertheless, one existing challenge for this method is its high computational cost, which is not applicable in some circumstances.

This paper mainly focuses on contrastive learning, which is suitable for computation-contrained devices. Contrastive learning frameworks like *SimCLR* learn visual representations by maximizing agreement between positive pairs while minimizing the disagreement among negative pairs [3]. Moreover, *BYOL* and *SimSiam* only contrast among positive pairs [14][15]. Built upon the basic framework of contrastive learning [20], some recent research works combine contrastive SSL with FL to address data heterogeneity on tasks in

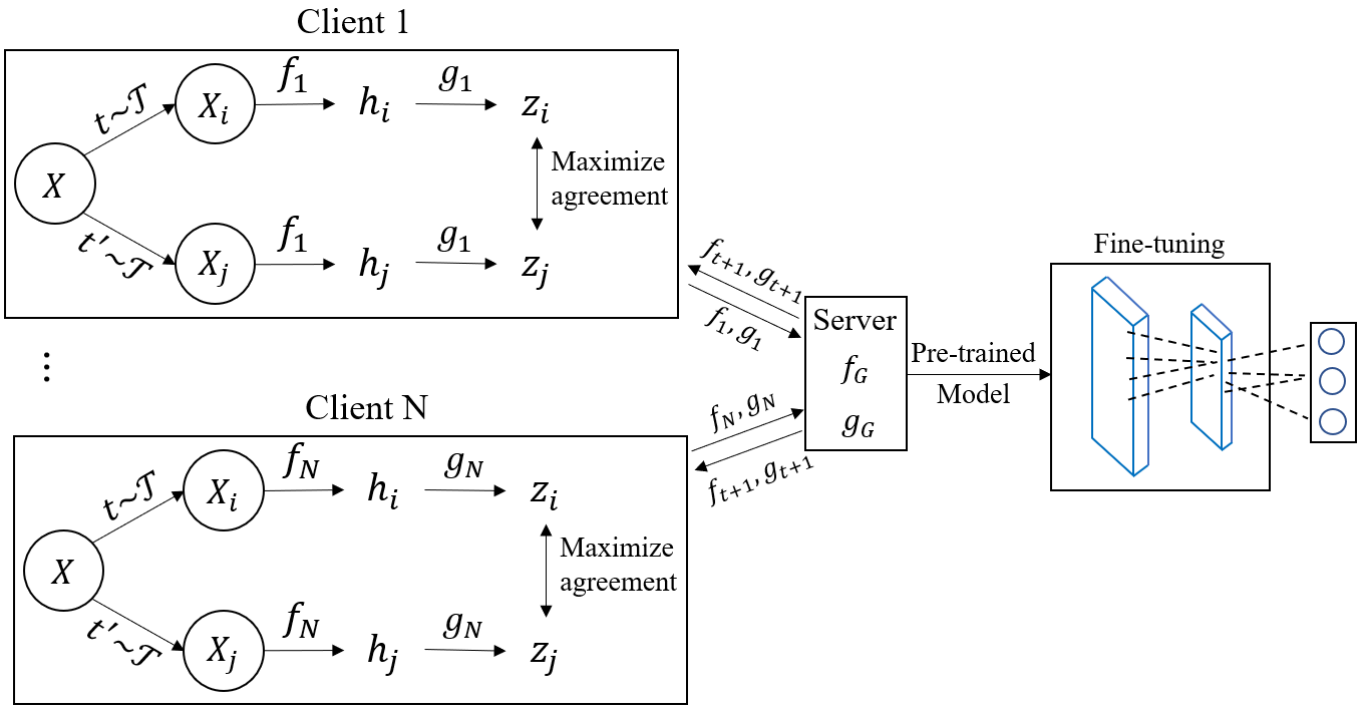

Fig. 1: Federated learning based on the simple framework of contrastive learning (*SimCLR*). In the pre-training stage, two separate data in client k are first augmented randomly. The augmented data will then be used to learn a base encoder network f_k and a projection head g_k that maximizes the agreement between z_i and z_j using a contrastive loss. Upon completion of training, the projection head will be discarded, and only the encoder network will be maintained and applied to the downstream classification tasks.

CIFAR-100 and acoustic events [4][9][10]. Contrastive self-supervised learning has relatively low computational cost [6], but smaller decentralized datasets may significantly impact the performance compared with the larger ones . *FedCLF* further improves unbalancedness through feature-sharing [6]. In this paper, we introduce a regularizer into the loss function of *SimCLR*, *BYOL*, and *SimSiam* in a federated-learning setting, and evaluate its performance on datasets including *CIFAR-10* and *CIFAR-100*.

II. Methodology

A. Problem

This paper proposes a framework that performs collaborative contrastive pre-training on non-iid clients and then fine-tunes the model. Suppose there are in total N clients in which each client $k \in \{1, ..., N\}$ has a local dataset D^k. Our goal is to learn a global SSL model over each dataset D^k, $k \in \{1, ...N\}$. The global loss function can be defined as follows:

$$\mathcal{L}(w) = \sum_{k=1}^{N} \frac{|D^k|}{|D|} \mathcal{L}_k(w), \tag{1}$$

where w denotes the global parameters to be learned. Since we aim to train and test our framework on non-iid data, $\mathcal{D}^m$ and $\mathcal{D}^n$ $(m \neq n)$ follow different distributions $P_m(x, y)$ and $P_n(x, y)$. Furthermore, our focus is on unlabeled data, *i.e.*, data in client *k* can be partitioned into $\mathcal{D}_l^k$ and $\mathcal{D}_u^k$, representing labeled and unlabeled part of the dataset respectively. The local empirical loss function is defined in the following way:

$$\mathcal{L}_k(w) = \mathbb{E}_{x \sim D_k}[l_k(w; x)], \tag{2}$$

in which l_k is the loss function of client k, and w stands for the global model parameters to be learned.

To tackle the challenge of data heterogeneity under the setting of federated learning, we first compare the performance of the current contrastive methods including *SimCLR*, *BYOL*, and *SimSiam*. Then, we explore and propose an improved version of the framework by introducing a regularizer into the loss function. We take *SimCLR* as an example, and a general flow of the framework is shown in Fig 1. The framework consists of two stages: a self-supervised federated pretraining stage and a supervised fine-tuning stage. In the first stage, the local models are individually trained on the distributed clients, and then are aggregated via *FedAvg* to obtain a global model. In each round of local training, different data are augmented and trained through an encoder network and an MLP to maximize the similarity between same data while minimize the dissimilarity among distinct data paris. In the second stage, the knowledge is transferred from the previous stage to the downstream task by fine-tuning the federated models. The flows for *BYOL* and *SimSiam* are similar except that the pretraining is slightly different, which will be covered in the next section.

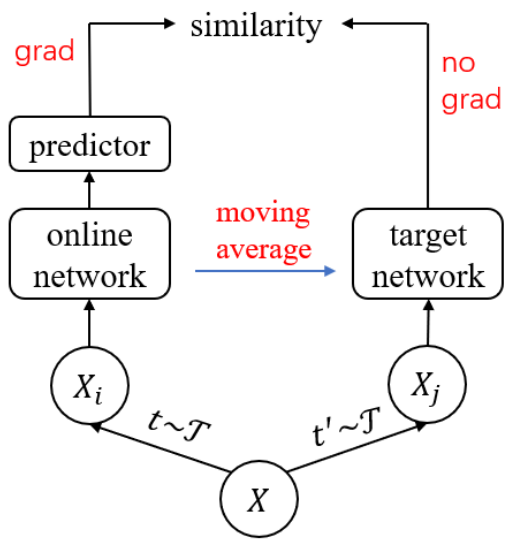

Fig. 2: Framework of *BYOL*

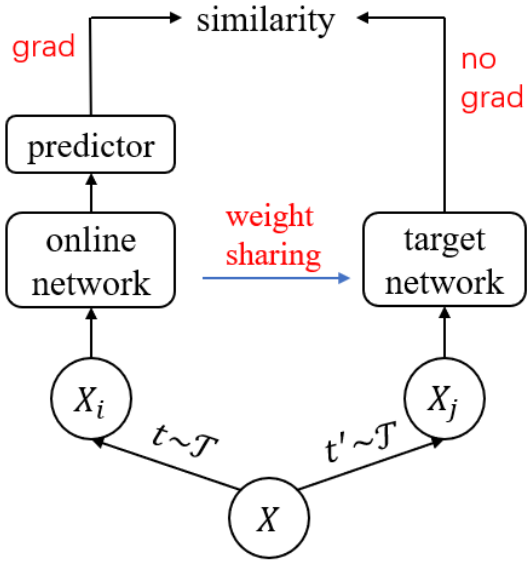

Fig. 3: Framework of *SimSiam*

B. Pre-training

In the pre-training stage, the framework of *SimCLR*, *BYOL*, and *SimSiam* basically comprises primarily five components before fine-tuning.

1) Data Augmentation: For a given data sample x, it is first passed through a stochastic data augmentation operator generating a positive pair, denoted as x_i and x_j. Data augumentation operators inlcude random crop, resize, color jitter, Gaussian noise, and Gaussian blur. The same procedure is applied to other data samples forming different positive pairs. Note that for *SimCLR* particularly, if two augmented data do not form a positive pair, they will together become a negative pair. For *BYOL* and *SimSiam*, we do need negative pairs in the loss function. Although studies have shown that augmentation composition such as random crop and color jitter can improve representation learning of the encoder network [3], we use random augmentation generators here for a more general situation.

2) Encoder Training: The encoder f is based on a neural network that learns representations from augmented data samples. This paper mainly adopts ResNet [23] as the backbone of our encoder network. More specifically, $h_i = f_k(x_i) = ResNet(x_i)$, where $h_i \in \mathbb{R}^d$ is the output of the network.

As for *SimCLR*, we apply the same ResNet to the augmented images X_i and X_j, and maximize the similarity after feeding the learned representations to the MLP layer, as indicated in Fig 1. Then, we perform backpropagation and gradient descent algorithm on both sides of X_i and X_j.

For *BYOL*, the encoder f indirectly shares weights between the two views. From a given target representation, we train a new online representation by predicting the target representation. The target encoder is obtained by exponential average moving of the online network, which is different from the trained target encoder in *SimCLR*, making the architecture asymmetric between the online and target pipeline.

The flow of *SimSiam* is similar to that of *BYOL*, except that its encoder network shares weights directly with the target encoder without an exponential moving average. A simple illustration of *BYOL* and *SimSiam* can be found in Fig. 2. and Fig. 3 respectively.

3) MLP Predictor: The extracted representation is then fed into an MLP projection head, which is a small neural network projection head g that maps to the contrastive loss function. A simple MLP structure can be modeled as $z_i = g(h_i) = W^{(2)}\sigma(W^{(1)}h_i)$ where σ is a non-linear function such as ReLU.

4) Contrastive Loss: For *SimCLR*, given two l_2 normalized representation vector u and v, define the agreement between them as cosine similarity, i.e., $sim(u, v) = u^T v/||u||\ ||v||$. The loss function for a positive pair of examples (i, j) is then defined in the following way,

$$l_{i,j} = -log\frac{exp(sim(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{k\neq i} exp(sim(z_i, z_k)/\tau)} \tag{3}$$

where $\mathbb{1}_{[k\neq i]} \in \{0, 1\}$ is an indicator function evaluating to 1 if and only if $k \neq i$ and τ represents a temperature hyper-parameter. If τ is small, the model will pay more attention to negative pairs [3].

Since we do not incorporate negative pairs in *BYOL*, and a stop-gradient approach is applied to the target branch, the loss function is slightly different from that of *SimCLR*. Suppose that v and v' are two augmented views from the same image. The online network outputs a representation $y_\theta = f_\theta(v)$ and a projection $z_\theta = g_\theta(y)$. The target encoder outputs $y'_\xi = f_\xi(v')$ and the target projection $z'_\xi = g_\xi(y')$ from the second augmented view v'. Here, θ and ξ are parameters of the online network and the target network respectively. The target network has the same architecture as the online network, and its parameters ξ are an exponential moving average of θ. After each training step, the following update is performed:

$$\xi \leftarrow \tau\xi + (1 - \tau)\theta \tag{4}$$

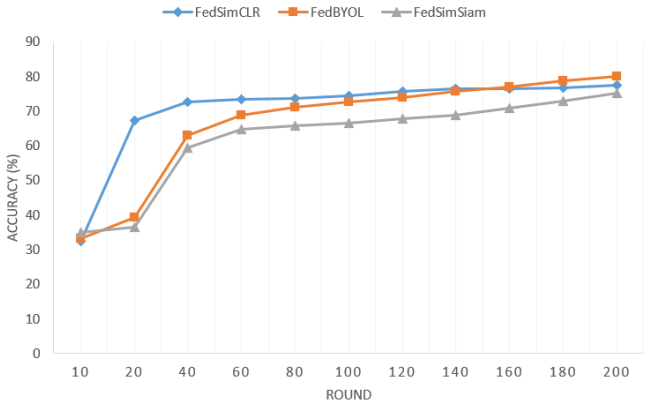

Fig. 4: Experiments on CIFAR-10 without Regularizer

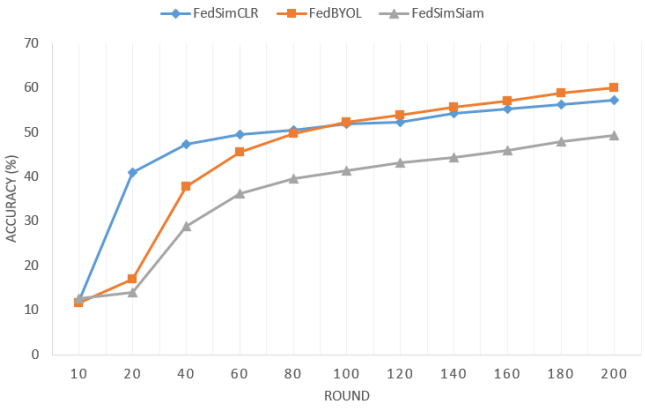

Fig. 5: Experiments on CIFAR-100 without Regularizer

given a target decay rate $\tau \in [0, 1]$.

After that, the representation from the online branch is fed into a predictor whose result is $q_\theta(z_\theta)$. The loss function aims to minimize the mean square error between l_2-normalized $q_\theta(z_\theta)$ and z'_ξ. The formula is shown as follows:

$$\mathcal{L}_{\theta,\xi} = 2 - 2 \cdot \frac{\langle q_\theta(z_\theta)\ ,\ z'_\xi \rangle}{||q_\theta(z_\theta)||_2 \cdot ||z'_\xi||_2} \tag{5}$$

In the framework of *SimSiam*, the encoder f shares weights directly with the two branches. In other words, it does not use a momentum target encoder.

5) Aggregation: Once each local client has learned its parameters at round *T*, the central server will update the global model through *FedAvg*,

$$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{|D^k|}{|\cup D^k|} w_{k+1}^k \tag{6}$$

C. Fine-tuning & Linear Evaluation

In the fine-tuning stage of federated learning, client k with its local encoder initialized as the global encoder pre-trained in the first stage is appended to a classifier. The model is fine-tuned based on the local data D_l^k in a federated setting similar to the previous stage. The representations learned by the model are fed into a linear classifier to minimize the cross entropy loss:

$$J(W) = -\frac{1}{m} \sum_{i,j}^{m,C} [\mathbb{I}(y_i = j) log(f_{w_j,b_j}(x_i))] \tag{7}$$

where f corresponds to the *softmax* function:

$$f_{w_j,b_j}(x) = \frac{exp(w_j^\intercal x + b_j)}{\sum_c exp(w_c^\intercal x + b_c)} \tag{8}$$

in which w, b are parameters to be tuned. Fine-tuning can achieve better performance in transfer learning [14], but in this paper, we mainly adopt *linear evaluation* to measure performance.

D. Preliminary Experiments

1) Datasets: We perform basic federated learning experiments on *SimCLR*, along with other two recently proposed contrastive learning methods which are *BYOL* and *SimSiam*. We partition the datasets to 5 clients according to their labels. This way of partitioning can simulate label-skew non-iid data. For *CIFAR-10*, the number of classes in a client is 2 in our experiments, while the number of classes per client for *CIFAR-100* is set to 20.

2) Setup: The model is trained on NVIDA GeForce RTX 2080. We use ResNet-18 as default encoder network. The projection head used is a two-layer multi-layer perceptron (MLP). During training, we set $R = 200$ rounds with $K = 5$ clients, $E = 5$ local epochs, batch size $B = 128$, and $\eta = 0.032$ with cosine decay.

3) Test in Server: Once we obtain the global model each 10 rounds, we will test the accuracy on a set of centralized data samples. The data samples are not distributed accross the clients, which is different from *Test in Client* as we shall see in the experiments of *PFL*.

4) Preliminary Results: Results of the three models on *CIFAR-10* and *CIFAR-100* are shown in Fig 2, Fig 3, and Table 1. As shown in the figures and table, federated learning based on *SimCLR* and *BYOL* performs better than *SimSiam*. For *CIFAR-10*, *FedSimCLR* and *FedBYOL* achievem 76.30% and 75.26% respectively, about 4% higher than *FedSimSiam*. Nevertheless, *FedSimCLR* and *FedBYOL* only achieve about 54% accuracy on *CIFAR-100*, which contains more classes and data samples, introducing more complexity and potential non-iid property to the data.

TABLE I: Test Accuracy without Regularizer

Method	CIFAR-10 (%)	CIFAR-100 (%)
FedSimCLR	76.30	54.16
FedBYOL	75.26	53.47
FedSimSiam	71.08	43.59

E. Algorithm Proposed

To improve the performance of the three contrastive federated learning schemes and tackle the non-iid data, we will improve the general framework by introducing a regularizer into

each local loss function. More specifically, we will consider each local client separately and optimize each local model with parameters $\{v_0^k\}_{k\in K}$ and local objective $l_k(v^k)$ [22]. The optimization problem in each client can be summarized as follows:

$$min_{v_k} \quad h_k(v^k; w^*) := l_k(v^k) + \frac{\lambda}{2}||v^k - w^*||^2 \qquad (9)$$

where w^* is the global model obtained from last round. The algorithm is illustrated as follows. The K clients are indexed by k; B is the local batch size, E is the number of local epochs, and η is the learning rate. Here, we introduce a parameter v^k for each client, which is the local model in the client.

Algorithm 1 Personalized Contrastive Federated Learning

ServerExecutes:
initialize w_0, m, $\{v_0^k\}_{k\in K}$
for each rount $t = 1, 2...$ **do**
  $S_t \leftarrow$ (random set of m clients)
  for each client $k \in S_t$ **in parallel do**
    $w_{t+1}^k \leftarrow$ ClientUpdate(k,w_t)
    Update v^k for E local iterations: ▷ Run on client k
      $v_{t+1}^k = v_t^k - \eta(\nabla h_k(v_t^k; w_t))$
  $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate(k,w): ▷ Run on client k
Sample batches $\mathcal{B}$ from local data D^k
for each local epoch i from 1 to E **do**
  for batch $b \in \mathcal{B}$ **do**
    $w \leftarrow w - \eta\nabla l(w; b)$
  return w to server

III. Experiment and Results

We assess the performance of our proposed personalized contrastive federated learning algorithm by evaluating the representations learned by *SimCLR*, *BYOL*, and *SimSiam* after self-supervised pretraining on the training set of *CIFAR-10* and *CIFAR-100*. We adopt the same partitioning measure as in the previous section that the dataset is divided by label classes to simulate the relatively worst case of out of distribution data samples. Moreover, we compare the test accuracy of the same contrastive learning framwork but with different regularizer coefficient, namely λ. More precisely, we choose λ to be 0, 0.1, and 0.5, and analyze how the choice of λ affects the performance of our algorithm.

One of the intuitions from our proposed Algorithm 1 is that in each iteration of training, the regularizer aims to minimize the distance between the local model and the global model. If we put more emphasis on the regularizer, the global model will be more affected by the local model. In other words, the model trained at this moment may achieve better performance if we evaluate the performance on this client. Hence, we apply *Test in Client* approach for evaluation as indicated in Fig 6. In Fig 6., each client first trains locally on D_{tr}^k and updates its model to the server. The server then aggregates the local models to obtain the global model. During testing process, rather than testing on a centralized dataset, we assess the performance on client-specific testing data D_{te}^k. In each iteration, we average the test accuracy from the result of each client.

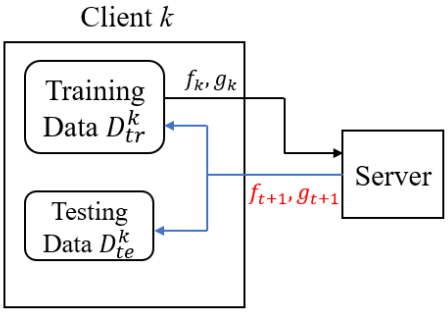

Fig. 6: Illustration of Test in Client for *SimCLR*

Table II shows the result of test accuracy on *CIFAR-10*. We find that regularizer can to some extent improve the performance of *FedSimCLR*, *FedBYOL*, and *FedSimSiam* by approximately 2%. But we do not see much improvement or degredation when changing λ from 0.1 to 0.5. Although the performance of *FedSimCLR* and *FedBYOL* are relatively close to each other, *FedBYOL* in general outperforms the other two contrastive federated learning methods, achieving the accuracy of 79.36% when λ is 0.5.

TABLE II: Test Accuracy (%) of *CIFAR-10* with Regularizer

Method	$\lambda = 0$	$\lambda = 0.1$	$\lambda = 0.5$
FedSimCLR	76.32	78.64	78.58
FedBYOL	76.24	78.42	79.36
FedSimSiam	71.20	72.36	72.94

Table III shows the result of test accuracy on *CIFAR-100*. The introduction of the regularizer also helps to improve the accuracy by 2% to 3%. The slight adjustment of λ from 0.1 to 0.5 does not see much change in the overall performance. Moreover, *FedBYOL* tends to attain better performance than *FedSimCLR* and *FedSimSiam*, and *FedSimSiam* does not achieve a desirable result.

TABLE III: Test Accuracy (%) of *CIFAR-100* with Regularizer

Method	$\lambda = 0$	$\lambda = 0.1$	$\lambda = 0.5$
FedSimCLR	54.16	56.78	56.86
FedBYOL	54.46	57.55	58.02
FedSimSiam	45.57	47.62	48.54

IV. Discussions

Based on the above observations, we further delve into the architecture of our algorithm, attempting to better understand personalized contrastive federated learning. Furthermore, we will point out some issues for future research.

A. Batch Size and Partitioning

When the number of training epochs is small, larger batch size can have a great benefit for *SimCLR* [3]. In our experiment, the local batch size is only 128, which may limit

the encoder network training process. To obtain a better representation from the online encoder, larger batch sizes such as 512 and 1024 should be taken into consideration. But larger batch sizes also indicate longer training procedure, which is not desirable for tasks with time constraints. But for *BYOL*, batch size is not considered to be a major issue [14], because it does not need negative samples.

We partition the data according to their classes, which is considerably label-skew. But in real-life situations, the data samples may not be that label-skew. Hence, it is more practical to use other ways to partition data samples such as Dirichlet sampling.

B. Data Augmentation

Contrastive learning methods like *SimCLR* are sensitive to the combination of image augmentations. If we get rid of color distortion in a *SimCLR*, the performance can drop [3]. So it is important to properly choose the combination of image augmentations to enhance the representation learning. Although *BYOL* and *SimSiam* are less influenced by the augmentations, *BYOL* tends to spend more time training.

C. Regularizer Weight λ

The introduction of the regularizer improves our model accuracy, but empirically, the optimal λ (the most accurate, fair, and robust solution) lies around 5 [22]. Furthermore, the adjustment we make in the experiment may be too small to capture any significant change. In the future research, one can adjust λ to even larger values to visualize the influence of this coefficient.

D. Model Forgetting

In each iteration, we update the testing model with the global model. After many rounds of training, the local model may forget its previous model. Thus, keeping track of the previous models and applying them to the current one are important to undermine the effect of forgetting.

V. Conclusion

This work aims to propose a personalized contrastive federated learning algorithm to tackle non-iid unlabeled data. We incorporate the contrastive learning frameworks, including *SimCLR*, *BYOL*, and *SimSiam* in our pipeline, and evaluate the test accuracy of them. Clients first pre-train a model from distributed unlabeled data, and the server aggregate the distributed models. We use linear evaluation to assess their performance. By introducing a regularizer into the loss function, the overall performance enhances slightly. However, due to the small adjustment to the regularizer coefficient, the results do not vary too much when λ is 0.1 and 0.5. Nevertheless, the discussion section provides future researchers useful insights into the personalized contrastive federated learning.

References

[1] B. McMahan, E. Moore, D. A. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, 2017, pp. 1273-1282.

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[3] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *ICML*, 2020.

[4] W. Huang, M. Ye, and B. Du, "Learn from others and be yourself in heterogeneous federated learning," in *CVPR*, 2022, pp. 10143-10153.

[5] R. Yan, L. Qu, Q. Wei, S. Huang, L. Shen, D. Rubin, L. Xing, and Y. Zhou, "Label-efficient self-supervised federated learning for tackling data heterogeneity in medical imaging," *arXiv*, 2022.

[6] Y. Wu, D. Zeng, Z. Wang, Y. Sheng, L. Yang, A. J. James, Y. Shi, and J. Hu, "Federated contrastive learning and masked autoencoder for dermatological disease diagnosis," *arXiv*, 2022.

[7] W. Chu, C. Xie, B. Wang, L. Li, L. Yin, H. Zhao, and B. Li, "FOCUS: fairness via agent-awareness for federated learning on heterogeneous data," *arXiv*, 2022.

[8] W. Zhuang, Y. Wen, and S. Zhang, "Divergence-aware federated self-supervised learning," in *ICLR*, 2022.

[9] M. Feng, C. Kao, Q. Tang, M. Sun, V. Rozgic, S. Matsoukas, and C. Wang, "Federated self-supervised learning for acoustic event classification," in *ICASSP*, 2022, pp. 481-485.

[10] D. Makhija, N. Ho, and J. Ghosh, "Federated self-supervised learning for heterogeneous clients," *arXiv*, 2022.

[11] M. J. Sheller, B. Edwards, G. A. Reina, J. Martin, S. Pati, A. Kotrotsou, M. Milchenko, W. Xu, D. Marcus, R. R. Colen, "Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data," *Scientific reports*, vol. 10, no. 1, pp. 1-12, 2020.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *NIPS*, 2012.

[13] J. Shi, Y. Zhang, Z. Li, X. Han, S. Ding, J. Wang, and S. Ying, "Pseudo-data based self-supervised federated learning for classification of histopathological images," *Transactions on medical imaging*, 2020.

[14] J. Grill, F. Strub, F. Altche, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. Guo, M. G. Azar, B. Piot, k. kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent - a new approach to self-supervised learning," *Advances in neural information processing systems*, vol. 33, pp. 21271-21284, 2020.

[15] X. Chen and K. He, "Exploring simple siamese representation learning," *arXir*, 2020.

[16] W. Zhuang, X. Gan, Y. Wen, S. Zhang, and S. Yi, "Collaborative unsupervised visual representation learning from decentralized data," in *ICCV*, 2021.

[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[18] K. He, X. Chen, S. Xie, Y. Li, P. Dollar, and R. Girshick, "Masked autoencoders are scalable vision learners," *arXiv preprint arXiv:2111.06377*, 2021.

[19] L. Qu, Y. Zhou, P. P. Liang, Y. Xia, F. Wang, L. Fei-Fei, E. Adeli, and D. Rubin, "Rethinking architecture design for tackling data heterogeneity in federated learning," in *CVPR*, 2022.

[20] A. Oord, Y. Li, and O. Vinyals. "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[22] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federeated learning through personalization," *Proceedings of the 38th International Conference on Machine Learning*, PMLR 139:6357-6368, 2021.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.