2023 IEEE 33rd International Workshop on Machine Learning for Signal Processing (MLSP) | 979-8-3503-2411-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/MLSP55844.2023.10285898

FEDERATED REPRESENTATION LEARNING THROUGH CLUSTERING

Runxuan Miao, Erdem Koyuncu

University of Illinois Chicago, Electrical and Computer Engineering, Chicago, IL

ABSTRACT

Federated self-supervised learning (FedSSL) methods have proven to be very useful in learning unlabeled data that is distributed to multiple clients, possibly heterogeneously. However, there is still a lot of room for improvement for FedSSL methods, especially for the case of highly heterogeneous data and a large number of classes. In this paper, we introduce a new way of thinking to approach the FedSSL problems. Specifically, we propose optimizing the representations through the more difficult task of clustering. The resulting federated representation learning through clustering (FedRLC) scheme utilizes i) a crossed KL divergence loss with a data selection strategy during local training and ii) a dynamic upload on local cluster centers during communication updates. Experimental results show that FedRLC achieves state-of-the-art results on widely used benchmarks even with highly heterogeneous settings and datasets with a large number of classes such as CIFAR-100.

Index Terms— Federated learning, self-supervised representation learning, clustering, KL divergence.

1. INTRODUCTION

By considering information security and accommodating low-resource computing devices, federated learning (FL) provides a means to train a neural network model over distributed data across multiple machines. However, most existing FL methods [1–4] rely on labeled data for supervised learning. Recently, self-supervised learning (SSL) methods have been proposed for learning representations on unlabeled data. Unfortunately, most SSL paradigms [5–11] assume that the data is centralized.

Recently, Federated SSL (FedSSL) [12–16] methods have been developed to learn representations of unlabeled data that is distributed to several local machines. For example, FedU [13] and FedEMA [12] directly adapt a fundamental centralized SSL model that is referred to as "Bootstrap Your Own Latent (BYOL) [6]" to federated learning. The key idea of BYOL is utilizing two views of the same data under augmentation and training an online network to predict the features extracted from a target network. However, the existing approaches [12, 13] measure the difference between model parameters and emphasize on managing the model upload during communication. Gradient-based local training steps are still performed using the BYOL loss functions. Besides, combining BYOL and FL directly can raise challenges: When the number of classes are large, the learned representations from BYOL are typically non-uniformly distributed over the representation space, leading to sub-optimal performance.

BYOL-like methods are typically referred to as non-contrastive methods as learning is accomplished through two augmented versions, or positive samples, of the same input. Contrastive methods [5], which rely on comparing the input with many negative samples have also been utilized for FedSSL. A recent notable example is [16], which also utilizes the novel idea of clustering clients for "FL in groups." We note that there have been several works on clustering centralized data [17–20], which are not immediately applicable to our distributed setting.

In this paper, we aim to address the challenges of FedSSL via our proposed Federated Representation Learning through Clustering (FedRLC) framework. A key idea of FedRLC is to solve the clustering task to aid in finding accurate representations. We do this by introducing a novel crossed KL divergence loss with a data selection strategy to optimize the cluster centers and the BYOL neural networks simultaneously. Intuitively, well-learned cluster centers are beneficial to extract more distinct information between different classes. Experimental results show that FedRLC improves the performance of existing FedSSL methods by a considerable margin and achieves state-of-the-art results on benchmark datasets such as CIFAR-100.

The rest of this paper is organized as follows: We introduce the FedSSL problem and the BYOL approach in Section 2. In Section 3, we introduce our proposed FedRLC scheme. Numerical results are provided in Section 4. We draw our main conclusions in Section 5.

2. PRELIMINARIES

The general goal of FedSSL is to learn a machine model for unlabeled data distributed over multiple clients. Typ-

This work was supported in part by the Army Research Lab (ARL) under Grant W911NF-21-2-0272, and in part by the National Science Foundation (NSF) under Grant CNS-2148182.

979-8-3503-2411-2/23/$31.00 ©2023 IEEE

ically, a central server is utilized to aggregate the client local models, each of which is trained via local data. Contrastive learning and non-contrastive learning are two main directions in SSL learning. In this work, we focus on a non-contrastive approach based on the BYOL scheme. In fact, the existence of a large number of negative samples in contrastive learning causes class collision issues. The non-contrastive nature of BYOL circumvents this problem and typically provides a better performance; see also FedEMA [12] that follows a similar approach.

In the following, we provide an overview of BYOL [6], and its straightforward federated generalization that we shall refer to as FedBYOL. Let D_k denote the local unlabeled dataset on Client k. Given some data $x_i \in D_k$, where i represents the data index, two samples $x_i^a \sim t^a(x_i)$ and $x_i^b \sim t^b(x_i)$ are generated through the augmentations t^a and t^b, respectively. The augmented data $t^\alpha(x_i)$, $\alpha \in \{a, b\}$ are then processed by the so-called online and target networks. The online network consists of an online encoder f^O and an online predictor g^O, which are trained by gradient descent.[1] The target network only consists of a target encoder f^T. The weights of f^T are updated via the exponential moving average (EMA) of the online encoder f^O, as will be explained in the following. Now, let $z_i^{\alpha,O} = g^O(f^O(t^\alpha(x_i)))$, $\alpha \in \{a, b\}$ and $z_i^{\alpha,T} = f^T(t^\alpha(x_i))$, $\alpha \in \{a, b\}$ denote the d-dimensional representations that one would obtain from the online and the target networks, respectively. Defining the scaled cosine similarity loss function as $\delta(x, y) = 2 - 2\frac{x^T y}{\|x\| \cdot \|y\|}$, BYOL uses the symmetrized loss $x_i \to L(x_i) = \delta(z_i^{a,O}, z_i^{b,T}) + \delta(z_i^{b,O}, z_i^{a,T})$.

The local objective of user k is then given by

$$(f^O, g^O) \to L_{INS} = \sum_{x_i \in D_k} L(x_i), \tag{1}$$

which signifies that only the online networks f^O, g^O are updated via gradient descent. The subscript "INS" means that we consider the instance-level loss without any consideration about the clusters. The target network parameters are instead updated through the EMA

$$f^T \leftarrow \sigma f^T + (1 - \sigma) f^O, \tag{2}$$

where $\sigma \in [0, 1]$.

As we have mentioned, FedSSL aims to learn a global model over the dataset $D = \bigcup_{k=1}^K D_k$. For example, one can aggregate the BYOL loss functions of all clients in order to attain the objective

$$\min \sum_{k=1}^K \frac{|D_k|}{|D|} \sum_{x_i \in D_k} L(x_i). \tag{3}$$

The objective (3) can be solved by using numerous FL algorithms [1–3] such as the classic federated averaging (FedAvg) [1]. However, simply extending the loss (1) to FL leads to suboptimal performance as the learned features from BYOL are not uniform between distinct classes, especially for heterogeneous data. In this paper, we propose FedRLC as a new alternative for FedSSL. FedRLC learns accurate representations through updating and keeping track of the centroids of each class.

[1] We refer to the composition of the encoder and the projector in the original BYOL paper as simply the "encoder" in this paper.

3. THE FEDRLC FRAMEWORK

In this section, we will introduce the proposed FedRLC framework. Our scheme relies on clustering to achieve good representations. The clustering is center-based. Hence, at each client, we keep track of M cluster centers, where the number of clusters M is assumed to be known a-priori. We start by introducing a novel crossed KL divergence loss with data selection for optimizing cluster centers to improve the quality of learned representations during local training. We will then present a dynamic rule to update the local cluster centers as well as the local neural networks during training.

The block diagram of the FedRLC framework is illustrated in Fig. 1 for local training at a certain Client k. In the following, we shall describe each stage in the figure in detail. The first stages to obtain the instance representations (until L^{INS}) apply verbatim from the BYOL scheme. Namely, samples x_i pass through the online and target networks to provide representations $z_i^{\alpha,v}$. Note that we have similarly omitted to indicate the dependence of the representations on the client index for brevity. We now describe the next steps.

3.1. Crossed KL divergence loss

In FedRLC, we define a novel crossed KL divergence loss (CKL) to learn a well-separated representation. CKL aims at optimizing M cluster centers by a crossed divergence between probabilities calculated from the online network and the target distribution from the target network. Specifically, let $\mu_1, \ldots, \mu_M \in \mathbb{R}^d$ denote cluster centers at a certain Client k. In practice, the cluster centers are initialized randomly. Given $\alpha \in \{a, b\}$ and $v \in \{O, T\}$, let $q_{i,m}^{\alpha,v}$ denote the probability that the representation $z_i^{\alpha,v}$ belongs to cluster m with center μ_m. Following DEC [17], we model these cluster assignment probabilities with a student t-distribution with one degree of freedom

$$\Delta_m(z, \{\mu_n\}_{n=1}^M) = \frac{(1 + \|z - \mu_m\|^2)^{-\frac{1}{2}}}{\sum_n (1 + \|z - \mu_n\|^2)^{-\frac{1}{2}}}. \tag{4}$$

Specifically, we set

$$q_{i,m}^{\alpha,v} = \Delta_m(z_i^{\alpha,v}, \{\mu_n\}_{n=1}^M), \; m \in \{1, \ldots, M\}, \; \alpha \in \{a, b\}, \; v \in \{O, T\}, \; \forall i. \tag{5}$$

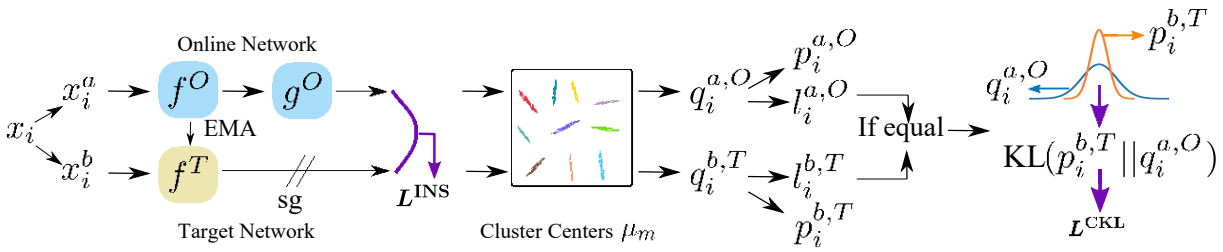


Fig. 1. The FedRLC framework during local training. sg means stop gradient. In the figure, we illustrate the construction of the first terms of the symmetric loss function in (9); the second terms are similar.

Effectively, each representation is assigned a probability distribution. According to (4), the closer the representation to a cluster center with index (say) m, the higher the belief/probability that the corresponding sample should belong to Cluster m.

To facilitate SSL, we now define a target distribution of the probabilities $q_m^{\alpha,T}$ that originate from the target networks described in Section 2. Following [17], we set

$$p_{i,m}^{\alpha,T} = \frac{(q_{i,m}^{a,T})^2 / \sum_i q_{i,m}^{a,T}}{\sum_n (q_{i,n}^{a,T})^2 / \sum_i q_{i,n}^{a,T}}. \quad (6)$$

The target distribution is computed by squaring the probability and normalizing it by the frequency of each class. The motivation of squaring is to "harden" the soft assignments, while frequency normalization penalizes imbalanced clusters.

We can now compare the probabilities $q_{i,m}^{\alpha,O}$ induced by the online networks with the probabilities $p_{i,m}^{\alpha,T}$ of the target networks. In this work, we utilize the KL divergence to compare the probability distributions. Letting

$$KL(p||q) = \sum_m p_m \log \frac{p_m}{q_m}, \quad (7)$$

the crossed KL divergence objective can be defined as

$$L^{CKL} = \frac{1}{N} \sum_i KL(p_i^{b,T}||q_i^{a,O}) + KL(p_i^{a,T}||q_i^{b,O}), \quad (8)$$

where N represents the batch size. The crossed KL objective (8) intends to optimize the local cluster centers by incorporating information from both augmented views of the input. The two augmented samples are supposed to share similar probabilities because they are created from the same data under different transformations.

3.2. Data Selection

Another novelty that we incorporate in FedRLC is to make sure that the augmentations that are involved in the crossed KL objective in (8) are not too far. Indeed, intuitively, completely irrelevant augmentations would harm, instead of benefit the overall performance. This is why we only incorporate pairs whose hard decisions match in the KL divergence losses. Let $l_i^{\alpha,v} = \text{argmax}_m (q_{i,m}^{\alpha,v})$ denote the hard clustering decisions of the online and target networks with different augmentations. Ties are broken in favor of the smallest index.

The data is chosen to contribute to the crossed KL divergence loss only when the predicted label from the online and the target networks are the same. We thus modify the loss in (8) to work with

$$L^{CKL} = \frac{1}{N} \sum_i KL(p_i^{b,T}||q_i^{a,O}) \cdot \mathbb{1}_{l_i^{b,T} = l_i^{a,O}} + \frac{1}{N} \sum_i KL(p_i^{a,T}||q_i^{b,O}) \cdot \mathbb{1}_{l_i^{a,T} = l_i^{b,O}}. \quad (9)$$

3.3. Local Training

As shown in Fig. 1, we jointly optimize the cluster centers and the online/target networks during local training. Therefore, the overall loss function is given by

$$L_k = L^{CKL} + L^{INS}, \quad (10)$$

where L^{INS} recalls the classical instance-level non-contrastive loss defined in (1). Usually, a hyperparameter can be incorporated to the loss function to control the relative weight of the losses L^{CKL} and L^{INS}. In our experiments, equal weights on the losses already provided a good performance. We thus leave a detailed study on hyperparameter tuning as future work.

3.4. Updates After Server-to-Client Communications

In this part, we describe the cluster center and online network update mechanisms during the server-to-client communications. We use subscript $\square$ to denote the global models, the subscript k to be the local model, and the superscript O to be the online networks. Let r represent the current training round. During the communication update, only the cluster centers and the online network are updated. We introduce a novel rule to update the centers and apply the specific EMA rule in [12] to update the online networks.

Specifically, given centers $\{\mu_{m,k}^{r-1} \in \mathbb{R}^d\}_{m=1}^M$ in local user k with local data D_k at round $r-1$, global centers $\mu_{m,\square}^r$ at round r, the centers of Client k at round r are updated according to

$$\mu_{m,k} = \frac{1}{|\epsilon|} \mu_{m,k} + \left(1 - \frac{1}{|\epsilon|}\right) \mu_{m,\square}, \quad (11)$$

where ϵ is updated progressively by the KL divergence between the probability generated from the local and global centers. Specifically, letting f_{θ}^{r} and $f_k^{O,r-1}$ denote the global encoder in round r and the local encoder in round $r-1$ at Client k, respectively, we define

$$z_{\theta,i,k} \triangleq \frac{1}{2}(f_{\theta}^{r}(x_{i,k}^{a}) + f_{\theta}^{r}(x_{i,k}^{b})), \tag{12}$$

$$z_{i,k} \triangleq \frac{1}{2}(f_k^{O,r-1}(x_{i,k}^{a}) + f_k^{O,r-1}(x_{i,k}^{b})) \tag{13}$$

as the mean representations of data $x_{i,k}$ under different augmentations and with global and local networks. We now evaluate the soft class probabilities for the data of Client k according to the global model at Round r as

$$q_{\theta,i,m,k} \triangleq \Delta_m(z_{\theta,i,k}, \{\mu_{n,\theta}^{r}\}_{n=1}^{M}). \tag{14}$$

Likewise, we can evaluate the class probabilities according the local model at Round $r-1$ as

$$q_{i,m,k} \triangleq \Delta_m(z_{i,k}, \{\mu_{n,k}^{r-1}\}_{n=1}^{M}). \tag{15}$$

We can now compute the momentum parameter ϵ via

$$\epsilon = \frac{1}{|D_k|}\sum_{i=1}^{|D_k|} KL\ \{q_{\theta,i,m,k}\}_{m=1}^{M} \| \{q_{i,m,k}\}_{m=1}^{M}. \tag{16}$$

When ϵ is large, the divergence between probabilities generated from global and local networks is large, so that the cluster centers inherit more local knowledge. Otherwise, a smaller ϵ gathers more information from global cluster centers. Finally, we discuss how to update the online networks of the client. For this purpose, we follow the EMA scheme [12]. Specifically, the online networks at Round r are updated as

$$(f_k^{O,r}, g_k^{O,r}) \leftarrow \gamma(f_k^{O,r-1}, g_k^{O,r-1}) + (1-\gamma)(f_{\theta}^{O,r}, g_{\theta}^{O,r}). \tag{17}$$

In (17), the parameter γ is used to control the weight between the global model and the local model. An explicit formula for γ is given by [12] $\gamma = \min(\lambda_k \| f_{\theta}^{r} - f_k^{O,r-1} \|, 1)$ where $\lambda_k = \frac{\tau}{\| f_{\theta}^{r-1} - f_k^{O} \|}$ is a customized magnitude, τ is a tuned hyperparameter, and f is the encoder. In EMA [12], λ_k is only measured once at the first round.

Algorithm 1 shows the overall FedRLC scheme.

4. EXPERIMENTS

Baselines and Datasets: We evaluate FedRLC on linear evaluation and semi-supervised learning tasks. Our baselines include FedU [13] and FedEMA [12], which are current state-of-the-art FedSSL methods. We also evaluate FedBYOL, which refers to combining BYOL [6] with federated averaging as in (3). Single-Training refers to training each client independently, and the accuracy is calculated by the average of all clients.

Algorithm 1 FedRLC

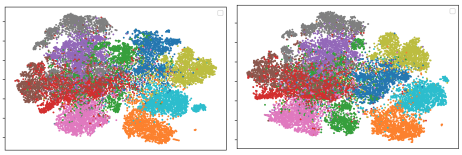
Input: Number of communication rounds R, Number of clients K, Number of local epochs E.
Output: Global encoder f_{θ} and predictor g_{θ}.

Server executes: Initialize server's network parameters f_{θ}, g_{θ}, and $\mu_{\theta,m}$. Have the clients initialize local parameters f_k^O, g_k^O, and $\mu_{k,m}$
for $r = 1, \ldots, R$ do
  for $k = 1, 2, \ldots, K$ in parallel do
    Send global encoder f_{θ}, predictor g_{θ}, and cluster centers $\mu_{\theta,m}$ to client k.
    $f_k^O, g_k^O, \mu_{m,k} \leftarrow$ ClientTraining($f_{\theta}, g_{\theta}, \mu_{\theta,m}$).
  end for
  FedAvg: $(f_{\theta}^O, g_{\theta}^O, \mu_{\theta,m}) \leftarrow \sum_k \frac{|D_k|}{|D|}(f_k^O, g_k^O, \mu_{m,k})$.
end for
Return global encoder f_{θ} and predictor g_{θ}.
ClientTraining($f_k^O, g_k^O, \mu_{k,m}$)
Update the online networks and cluster centers via global parameters by (17) and (11), respectively.
for epochs $= 1, \ldots, E$ and size-N batch learning within each epoch over dataset D_k do
  Update online networks and cluster centers via global parameters by descending the gradient of the local cost function in (10).
  Update the target network parameters f_k^T via (2).
end for
Return the online networks f_k^O and g_k^O.

Data Heterogeneity: We follow the exact settings of FedU [13] and FedEMA [12] for a fair comparison. Namely, to simulate data heterogeneity in federated learning, each user only consists of samples from M/K classes, where M is the number of classes, and K is the number of clients. This is referred to as the data-split scenario. For independent and identically distributed (IID) data, each user has the same number of samples from M classes. In addition to the data-split non-IID scenario, to evaluate on different non-IID scenarios, we sample a specific proportion of the data from class m to client k, where the proportion is followed by the Dirichlet distribution with parameter β, which is also a widely-used method to simulate non-IID data distribution. A smaller β indicates a more heterogeneous distribution.

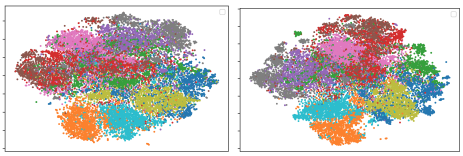
Implementation Details: For federated training, we adopt the SGD optimizer with a 0.032 initial learning rate. The learning rate is decayed by cosine annealing. The batch size is 128, and the input size is 32×32. We use ResNet18 to be the encoder, and the predictor is a two-layer multiplayer perceptron (MLP) with the output dimension 2048. The σ of EMA is 0.99, and the $\tau = 0.7$ is directly followed by [12] without tuning. We set both the number of local clients and the number of local epochs to 5. The total communication rounds are

100 in federated learning, which are the same as recent FedSSL approaches [12, 13] for a fair comparison.



(a) IID FedBYOL (b) IID FedRLC

Fig. 2. t-SNE data visualization on CIFAR-10.



(a) Non-IID FedBYOL (b) Non-IID FedRLC

Fig. 3. t-SNE data visualization on CIFAR-10.

Visualization of Representations: To analyze the data features visually, we plot the t-SNE visualization of the CIFAR-10 learned from FedBYOL and FedRLC in Fig 2 and Fig 3, where different colors indicate different classes. From the comparison between FedBYOL and FedRLC, we observe that the data representations obtained from FedRLC are separated more clearly. The following linear and semi-supervised evaluations further verify the effectiveness of FedRLC.

Linear Evaluation: To validate the quality of learned representations, a linear classifier is trained on top of the frozen representations learned from different FedSSL methods. For linear evaluation training, the AdamW optimizer is adopted with a learning rate of 0.022. The results are shown in Table 1 and 2. FedRLC constantly outperforms other methods, especially for CIFAR-100 with a large number of classes, where it improves by 2.77% and 1.62% on IID and non-IID data, respectively. FeatARC, which was announced recently and developed independently of the current study, provides experiments in the same setup as in Table 1 for the CIFAR-10 dataset. It achieves an accuracy of 86.74% and 84.63% for the IID and non-IID settings, respectively. FedRLC outperforms FeatARC in the IID case, while it is worse in the non-IID case. We note that FeatARC relies on the significantly more memory and computationally-intensive contrastive-learning methods, and do not provide numerical results for the CIFAR-100 dataset with a large number of classes. Also, the data clustering methodology of FedRLC can be combined with the client clustering method of FeatARC to potentially improve the performance of either method, as will be discussed in a future work.

Semi-supervised Learning: We compare our model with state-of-the-art works on semi-supervised learning tasks. A new MLP is added on the top of the encoder in semi-supervised learning, and we fine-tune the entire model with 10% labeled data. We compare different federated representation learning methods under IID and Non-IID setting for CIFAR-10 and CIFAR-100 datasets. Tables 3 and 4 demonstrate that our scheme achieves the best results in all cases. In particular, FedRLC improves the performance of CIFAR-100 by 1.68% under a highly heterogeneous scenario.

Table 1. Linear Evaluation: IID & Data-Split Non-IID.

Dataset	CIFAR-10		CIFAR-100	
Method	IID	Non-IID	IID	Non-IID
Single-Training	82.42	74.95	53.88	52.37
FedBYOL	84.29	79.44	54.24	57.51
FedU	83.96	80.52	54.82	57.21
FedEMA	86.26	83.34	58.55	61.78
FedRLC	87.06	84.08	61.32	63.40
BYOL (Centralized)	90.46		65.54	

Table 2. Linear Evaluation: Dirichlet Non-IID.

Dataset	CIFAR-10		CIFAR-100	
β	0.5	0.1	0.5	0.1
Single-Training	83.42	83.08	58.45	57.20
FedBYOL	85.44	84.69	59.14	59.93
FedU	85.62	85.33	59.10	58.06
FedEMA	86.12	86.00	60.26	61.46
FedRLC	86.89	86.69	62.39	63.21

Table 3. Semi-Supervised Learning: IID & Data-Split Non-IID.

Dataset	CIFAR-10		CIFAR-100	
Method	IID	Non-IID	IID	Non-IID
Single-Training	78.08	69.06	43.50	39.99
FedBYOL	83.24	76.95	49.20	47.07
FedU	82.61	77.06	47.64	46.67
FedEMA	83.38	79.49	49.26	50.48
FedRLC	83.99	79.52	49.67	52.16

Table 4. Semi-Supervised Learning: Dirichlet Non-IID.

Dataset	CIFAR-10		CIFAR-100	
β	0.5	0.1	0.5	0.1
Single-Training	81.72	79.89	48.53	49.41
FedBYOL	82.84	82.20	50.00	50.12
FedU	81.33	81.66	49.25	49.31
FedEMA	83.18	82.06	50.11	51.07
FedRLC	83.41	82.73	50.41	51.19

5. CONCLUSIONS

We have proposed FedRLC, a federated self-supervised representation learning scheme. A key idea of FedRLC is to achieve good representations through the aid of clustering. In particular, FedRLC optimized a crossed KL divergence loss between two augmented data with a data selection mechanism and updated several cluster centers dynamically during communication. Evaluation on the learned image features demonstrated that our approach learned better semantic knowledge of the data compared with other existing FedSSL methods. Moreover, FedRLC has achieved state-of-the-art results on benchmark downstream tasks including linear evaluation and semi-supervised learning.

6. REFERENCES

[1] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in Artificial intelligence and statistics. PMLR, 2017.

[2] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith, "Federated optimization in heterogeneous networks," Proceedings of Machine learning and systems, vol. 2, pp. 429–450, 2020.

[3] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in Proceedings of the 37th ICML. 2020, PMLR.

[4] Qinbin Li, Bingsheng He, and Dawn Song, "Model-contrastive federated learning," in CVPR, 2021.

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton, "A simple framework for contrastive learning of visual representations," in ICML. PMLR, 2020, pp. 1597–1607.

[6] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al., "Bootstrap your own latent-a new approach to self-supervised learning," NeurIPS, 2020.

[7] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick, "Momentum contrast for unsupervised visual representation learning," in CVPR, 2020.

[8] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," Neurips, 2020.

[9] Xinlei Chen and Kaiming He, "Exploring simple siamese representation learning," in CVPR, June 2021, pp. 15750–15758.

[10] Phuc H. Le-Khac, Graham Healy, and Alan F. Smeaton, "Contrastive representation learning: A framework and review," IEEE Access, 2020.

[11] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stephane Deny, "Barlow twins: Self-supervised learning via redundancy reduction," in ICML, 2021.

[12] Weiming Zhuang, Yonggang Wen, and Shuai Zhang, "Divergence-aware federated self-supervised learning," in ICLR, 2022.

[13] Weiming Zhuang, Xin Gan, Yonggang Wen, Shuai Zhang, and Shuai Yi, "Collaborative unsupervised visual representation learning from decentralized data," in ICCV, 2021.

[14] Runxuan Miao and Erdem Koyuncu, "Federated momentum contrastive clustering," arXiv preprint arXiv:2206.05093, 2022.

[15] Fengda Zhang, Kun Kuang, Zhaoyang You, Tao Shen, Jun Xiao, Yin Zhang, Chao Wu, Yueting Zhuang, and Xiaolin Li, "Federated unsupervised representation learning," arXiv preprint arXiv:2010.08982, 2020.

[16] Lirui Wang, Kaiqing Zhang, Yunzhu Li, Yonglong Tian, and Russ Tedrake, "Does learning from decentralized non-iid unlabeled data benefit from self supervision?," in ICLR, 2023.

[17] Junyuan Xie, Ross Girshick, and Ali Farhadi, "Unsupervised deep embedding for clustering analysis," in ICML. PMLR, 2016, pp. 478–487.

[18] Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng, "Contrastive clustering," in AAAI, 2021.

[19] Zhizhong Huang, Jie Chen, Junping Zhang, and Hongming Shan, "Learning representation for clustering via prototype scattering and positive sampling," IEEE TPAMI, 2022.

[20] Erdem Koyuncu, "Centroidal clustering of noisy observations by using rth power distortion measures," IEEE TNNLS, 2022.