Additive Angular Margin Loss for Federated Learning in Image Classification

Phuong-Nam Tran
Dept. of Artificial Intelligence
Kyung Hee University
Yongin, 446-701, Republic of Korea
tpnam0901@khu.ac.kr

Duc Ngoc Minh Dang
AiTA Lab, Dept. of Computing Fundamental
FPT University
Ho Chi Minh City, Vietnam
ducdnm2@fe.edu.vn

Eui-Nam Huh
Dept. of Computer Science and Engineering
Kyung Hee University
Yongin, 446-701, Republic of Korea
johnhuh@khu.ac.kr

Choong Seon Hong*
Dept. of Computer Science and Engineering
Kyung Hee University
Yongin, 446-701, Republic of Korea
cshong@khu.ac.kr

***Abstract*—Federated learning (FL) is a method that leverages data from multiple sources to enhance deep learning models while safeguarding data privacy. This approach finds applications in diverse domains such as healthcare, entertainment, and user experience enhancement. Despite its effectiveness, FL's performance is still low compared to centralized training methods, necessitating further improvements. Particularly in critical areas such as medicine, where accurate model predictions are crucial, conventional FL algorithms fall short compared to centralized training. This study introduces a new training approach to further improve the vanilla FL algorithm. Our proposed method takes advantage of feature-based adjustments using cosine angles by incorporating an angular margin loss function alongside the cross-entropy loss. It notably enhances the accuracy of the aggregated models on datasets such as MNIST, CIFAR10, and CIFAR100 while maintaining FL's privacy-preserving attributes. Moreover, we conduct a comprehensive comparative analysis of the proposed method against existing FL algorithms to evaluate the impact of the angular margin loss function on the learning process. Our experimental results underscore the effectiveness of the proposed algorithm when compared with standard benchmarks, proving its potential to advance the field of FL.**

***Index Terms*—Federated learning (FL), Angular margin loss, Decentralized training, Data heterogeneity, Representation learning, Personalized Federated Learning (PFL)**

I. Introduction

In recent years, the increase of big data has catalyzed significant advancements across academic and industrial, particularly focusing on the artificial intelligence (AI) field. This increase in data has enabled the development of more precise and rapid AI models that find practical applications in various

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2019-0-01287, Evolvable Deep Learning Model Generation Platform for Edge Computing), (No.2021-0-02068, Artificial Intelligence Innovation Hub), and supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2024-RS-2023-00258649) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation) * Dr. CS Hong is the corresponding author.

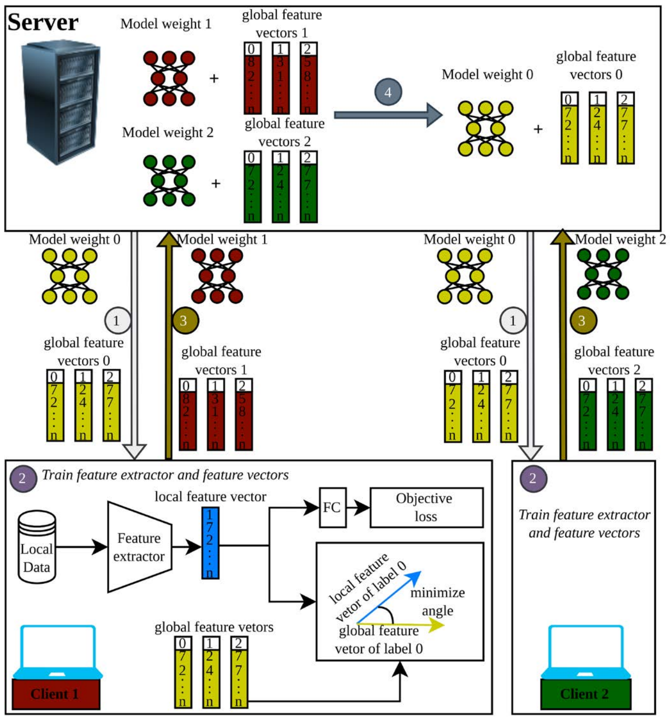

Fig. 1: Our training strategy. (1) The model's weight and feature vectors are initialized on the server and distributed to all clients. (2) Each client loads the received weight and conducts training with its local dataset. This process includes aligning the local feature vector and minimizing the objective loss function (3). After training for a set number of epochs, the client returns the updated model weight and feature vectors to the server. (4) The server executes an aggregation algorithm to combine the received weights and feature vectors from all clients. This iterative process is repeated until an interrupt signal is received or the model achieves the desired performance.

fields of our daily lives. As a result, AI has been applied to numerous domains, such as user experience enhancement, customer service, healthcare, and even military operations [1]–[4]. By taking advantage of vast datasets for training, AI models have moved closer to human-level decision-making capabilities in tackling complex tasks. However, acquiring datasets for training poses a challenge for researchers and businesses. They must guarantee data privacy preservation to safeguard user information while maximizing operational efficiencies to increase profits. The conventional approach of storing large datasets in centralized data centers demands substantial resources and may expose users to potential data leaking.

To tackle this problem, federated learning (FL) [5] emerges as a distributed training strategy. FL removes the need to store data in the data center for training purposes while still delivering outstanding performance comparable to centralized training methodologies. FL thus offers an efficient solution that guarantees data privacy and optimal resource utilization. FL leverages distributed training, taking advantage of the computational capabilities of both servers and clients. As illustrated in Fig 1, the AI model undergoes training solely on the client's resources using its local dataset. Subsequently, only the model weights are transmitted to the server for aggregation. This approach guarantees that users' data remains on their devices, preserving privacy while enabling them to benefit from other datasets through collaboration with other clients. Furthermore, this strategy reduces the demand for central resources while harnessing the diverse computational capabilities of individual clients. Therefore, FL offers a valuable solution that guarantees data privacy preservation and operational efficiency in the fast-growing big data.

Numerous recent FL algorithms [5]–[13] have been developed to address these challenges and enhance aggregation techniques for improving performance. However, applying these algorithms to real-world scenarios often results in a significant drop in model efficiency, as indicated in [7]. One challenge of real-world application lies in the presence of heterogeneous data, which poses a difficulty in effectively preserving the valuable features of the client models. Due to the non-IID (Non-independent identifies distributed) data distribution of different clients, each client may lack complete representation of all classes within its local dataset. Consequently, this scenario makes each client converge solely toward the local minimum of its dataset, disregarding the broader context of balancing client and server model minimums across all shared datasets.

To tackle this challenge, various strategies are applied that consider the similarities among clients' model weights before executing aggregation processes. This methodology, utilized by FeSem [8], underscores the significance of clustering clients' model weights within the aggregation algorithm. In the context of FeSem, clients with similar model weights are consolidated into groups to generate new composite weights. Subsequently, the final model is aggregated based on these new weights, thereby mitigating dissimilarities among the models and improving the model performance. This clustering approach is also utilized in DemLearn [10], wherein nodes (clients) are updated based on their position within a tree graph structure. In DemLearn, each node exclusively communicates with its nearest higher or lower node in the tree, minimizing discrepancies among node weights. The construction of the tree graph in DemLearn is based on the similarity of node weights, facilitating the formation of a weight aggregation graph. This methodology enables flexibility among clients within the graph, augmenting the final model's performance by aligning the distribution of the global model with that of the individual node models and vice versa.

Another approach to this challenge is to train a feature-based model by fine-tuning the feature vectors during the training phase. This approach trains the model to represent samples in a high-dimensional space. Acting as an encoder, the model transforms each sample into a feature vector that encapsulates information specific to that sample. Subsequently, these feature vectors can be adjusted based on Euclidean distance, Cosine angles, or feature-based loss function. One of the first algorithms that adopt this approach in FL is MOON [9], which leverages contrastive learning to refine the feature vectors of local and global models. By integrating the contrastive loss function with the conventional cross-entropy loss, MOON has demonstrated a notable enhancement in FL performance compared to FedAvg [5]. Similarly, FedSeg [7] is another innovative approach that harnesses the feature-based methodology to improve FedAvg's performance. FedSeg achieves this by aligning the local pixel embeddings with the global pixel embeddings through the contrastive loss function in the context of semantic segmentation tasks. FedSeg has shown significant improvement in model accuracy and efficiency through experiments with its pixel embedding representation in high-dimensional space.

Based on the potential of feature-based learning, this paper proposes a new FL methodology for classification tasks illustrated in Fig. 1 to address the challenges outlined earlier. Our proposed method integrates the angular margin loss function with the conventional classification loss function to balance the model's classification accuracy and feature representation, as shown in [14]. In detail, it leverages ArcFace [15] to fine-tune the feature vectors while concurrently harnessing the cross-entropy loss function to enhance prediction performance. By applying feature-based learning within the FL framework, our method enables local models to achieve a better local minimum, aligning with the global minimum during optimization. This strategic fusion of techniques in our method underscores its potential to optimize classification tasks within the FL domain.

To summarize, the contributions of this paper are as follows:

- We propose a new FL method that adjusts feature vectors based on the angular margin loss function to improve the performance of the global model.
- We provide experiments and analysis on three classification datasets to evaluate our proposed method for the classification FL problem.

The remainder of this paper is structured as follows. Section II describes our proposed method and loss functions. Subsequently, insights from experimental results and our analysis are presented in Section III. Finally, Section IV summarizes the study's discoveries.

II. Methodology

A. Federated Learning

As depicted in Fig. 1, the FL algorithm includes four steps for training and aggregating a global model. Suppose there are K clients, each with a local dataset $\mathcal{D}_k$. To optimize the performance of the global model over the dataset $\mathcal{D} := \cup_{k\in[K]}\mathcal{D}_k$, the training strategy aims to identify a weight w that minimizes the following function:

$$\underset{w}{\text{argmin}}\ f(w) = \sum_{k}^{K} \frac{|\mathcal{D}_k|}{|\mathcal{D}|} F_k(w_k) \tag{1}$$

where $f(w)$ is the global objective function, $F_k(w_k)$ is the local objective function which is define as $F_k(w_k) := \sum_{i=1}^{n_k} \mathcal{L}(x_i, y_i, w_k)$ where $\mathcal{L}(.)$ is the loss of prediction on example (x_i, y_i) made with model parameters w of k^{th} client and total client sample $n_k := |\mathcal{D}_k|$.

Equation 1 outlines how the global model is updated, considering the proportion of a client's sample size relative to total samples across all clients' datasets. Despite overlooking dissimilarities among the model weights, this updating process consistently shows considerable performance enhancements for the global model. In fact, FedAvg has underscored the efficiency of this methodology through comprehensive experimental validations. Our study uses FedAvg as the baseline for our development and comparative analysis.

B. Angular Margin Loss function

In deep learning, a model makes predictions based on the feature vector, which is a structured set of numerical or categorical values representing the essential characteristics of a data point extracted by a feature extractor network. The accurate representation of these features is crucial for the model's performance, as it influences the model's ability to learn from the data and generalize effectively. Well-crafted feature vectors effectively capture relevant information and relationships within the dataset, enabling models to make accurate predictions or classifications. On the other hand, poorly defined features can significantly impact the model's performance, resulting in suboptimal outcomes.

To improve the representation of feature vectors, a straightforward approach is bringing similar vectors closer and pushing dissimilar vectors farther apart. This objective can be achieved by minimizing the Euclidean distance or maximizing the Cosine similarity score between two similar vectors as utilized in TripLet [16]. An alternative method for representation learning is fine-tuning feature vectors according to cosine angles. Instead of aligning feature vectors solely based on Euclidean distances, ArcFace [15] introduces a loss function that integrates cross-entropy loss, softmax function, and the final fully connected layer. The fundamental concept of this algorithm is to factorize the separability between samples and parameters into components of amplitude and angular variations using cosine similarity. The formula below provides the computation process for the last linear layer:

$$\text{logits} = \sum_{i}^{n_k} W^T z(x_i, y_i, w_k) + b \tag{2}$$

where logits are the output of the final classifier layer with the weight matrix $W \in \mathbb{R}^{d\times c}$ and bias vector $b \in \mathbb{R}^c$, d is the dimension of the preceding linear layer and c is the number of classes in the dataset, $z(x_i, y_i, w_k)$ denotes the feature vector, which serves as the output from the layer just before the final layer. Based on the cosine similarity formula between two vectors, the Equation 2 can be rewritten as follows:

$$\text{logits} = \sum_{i}^{n_k} ||W^T||||z(x_i, y_i, w_k)|| \cos(\theta) + b \tag{3}$$

where θ is the angle between each vector in the weight matrix and the feature vector. To reduce the dependent variable of Equation 3, ArcFace [15] sets $b = 0$, and $||W^T|| = 1$, $||z(x_i, y_i, w_k)|| = 1$ by normalizing the them before calculating the logits. Consequently, Equation 3 will only rely on the θ value. A margin 'm' is added to θ to establish a distinct boundary between disparate vectors. By incorporating this margin, a clear separation is enforced, enhancing the model's discriminative ability. Finally, this loss function leverages the cross-entropy loss to quantify the alignment between predicted and actual classes. The final angular margin loss function [15] ($\mathcal{L}_{arc}$) can be presented as follows:

$$\mathcal{L}_{arc} = \frac{1}{n_k}\sum_{i}^{n_k} -log\left(\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))} + \sum_{j,j\neq y_i}^{c} e^{s(\cos(\theta_j))}}\right) \tag{4}$$

where s is a scale value, which is a hyper-parameter used to rescale the logits, θ_{y_i} is the angle between the feature vector and the ground truth vector in the weight matrix.

C. Additive Angular Margin Loss for Federated Learning

Based on the above techniques, we introduce a new framework that enhances FL performance by integrating $\mathcal{L}_{arc}$. Our method takes advantage of representation learning while still maintaining the vanilla cross-entropy loss ($\mathcal{L}_{ce}$) to improve the performance of models. In detail, the $\mathcal{L}_{arc}$ will be added as another loss function head in the training process. Thus, the local objective function of each client is as follows:

$$\mathcal{L}(x, y, w) = \mathcal{L}_{ce}(x, y, w) + \lambda\mathcal{L}_{arc}(x, y, w) \tag{5}$$

where λ is a hyper-parameter that controls the effect of representation learning on the final loss function. This function encapsulates the customized objective tailored for individual clients within the FL setting, combining the primary $\mathcal{L}_{ce}$ with the $\mathcal{L}_{arc}$ to optimize model parameters effectively. Our method presents a promising avenue for enhancing FL outcomes by leveraging representation learning and specialized loss functions. It replaces the previous *LocalUpdate* in the original FedAvg algorithm [5] with *ClientUpdate*, showing in

Algorithm 1 to combine representation learning with specialized loss functions. Our algorithm focuses on local objective updates to improve the performance of the global model compared to the vanilla algorithm.

Algorithm 1 Total number of clients K, local batch size B, number of client epochs E, number of communication rounds R, learning rate γ, random fraction C for selecting the clients participating in training each round, feature vector F.

```
ServerExecute
 1: Procedure SERVEREXECUTE($K$, $C$, $E$, $R$, $B$, $\gamma$)
 2: initialize $w_0, F_0$
 3: $t \leftarrow t = 0$
 4: while $t < R$ do
 5:     $m \leftarrow max(C \times K, 1)$
 6:     $S_t \leftarrow$ (random set of $m$ clients)
 7:     for each client $k \in S_t$ do
 8:         $w^k_{t+1}, F^k_{t+1} \leftarrow ClientUpdate(k, E, B, \gamma, w_t, F_t)$
 9:     end for
10:     $w_{t+1} \leftarrow \sum_k^{|S_t|} \frac{n_k}{n} w^k_{t+1}$
11:     $F_{t+1} \leftarrow \sum_k^{|S_t|} \frac{n_k}{n} F^k_{t+1}$
12:     $t \leftarrow t + 1$
13: end while
14: End Procedure
ClientUpdate
 1: Procedure CLIENTUPDATE($k$, $E$, $B$, $\gamma$, $w, F$)
 2: $\mathcal{B} \leftarrow$ split $\mathcal{D}_k$ into batches of size $B$
 3: $e \leftarrow 0$
 4: while $e < E$ do
 5:     for batch $b \in \mathcal{B}$ do
 6:         $x, y \leftarrow b$
 7:         $\mathcal{L}(x, y, w) \leftarrow \mathcal{L}_{ce}(x, y, w) + \lambda \times \mathcal{L}_{arc}(x, y, w, F)$
 8:         $w \leftarrow w - \gamma \frac{\partial \mathcal{L}(x,y,w)}{\partial w}$
 9:         $F \leftarrow F - \gamma \frac{\partial \mathcal{L}(x,y,w)}{\partial F}$
10:     end for
11:     $e \leftarrow e + 1$
12: end while
13: return $w, F$
14: End Procedure
```

As depicted in Fig. 1 and Algorithm 1, the model is trained to generate feature vectors similar to the global feature vectors within the same class. This learning approach facilitates local models in converging towards the global optimal point during optimization, thereby enhancing the performance of the global model. Moreover, the local models help refine the global feature vectors to identify the most suitable ones that effectively accommodate all clients' datasets.

III. EXPERIMENT RESULTS

A. Dataset

The experiments explore the performance of our algorithm across three benchmark datasets: MNIST [17], CIFAR10 [18], and CIFAR100 [18]. The experimental design involves an approach to data partitioning. CIFAR10 follows an IID split approach, while MNIST and CIFAR100 follow a non-IID partitioning approach. This partitioning approach leverages the unique characteristics of each dataset to investigate the impact of varying data distributions.

1) IID partitioning approach: In the context of IID, the experiments are solely performed on the CIFAR10 [18] dataset. The implementation utilizes the StratifiedKFold functionality from the Scikit-learn [19] library to construct the IID dataset. The number of folds in StratifiedKFold is the number of clients within the system. Under this methodology, each client is mapped with the test set from each fold, serving as a local training dataset splitting from the CIFAR10 training set. This approach guarantees that every client has an equal number of total samples and maintains an identical distribution of samples across all classes in the dataset.

2) Non-IID partitioning approach: In the non-IID context, our experiments utilize the MNIST [17] and CIFAR100 [18] datasets to evaluate the efficiency of the proposed methodology. Unlike the IID scenario, each client within our non-IID partitioning has a maximum of three classes represented in its local dataset. The initial step in constructing this non-IID dataset involves sorting the dataset based on sample labels. Subsequently, leveraging parameter $shards\ per\ clients$, we partition the training dataset into $shards\ per\ clients \times number\ of\ clients$ continuous shards, which each shard may contain from 1 to 3 classes. These shards are then shuffled, and $shards\ per\ clients$ shards are assigned to each client. In this configuration, the distribution ensures that each client's dataset contains an equivalent number of samples but limited classes, thereby establishing a non-IID setup. For this setup, the $number\ of\ clients$ is set to 100 for MNIST and 125 for CIFAR100. The same $shards\ per\ clients$ is utilized in both datasets, which is 2.

B. Hyper-parameters setup

The experiments were executed on an Ubuntu server equipped with an NVIDIA 3080ti GPU and a Python environment. For the MNIST dataset, a simple convolutional neural network architecture was employed. The network architecture comprises two convolutional layers with kernel sizes of 5x5 and output channels of 32 and 64, sequentially followed by a max-pooling layer with a kernel size of 2x2. A simple embedding block after these two blocks was incorporated to align the model with the $\mathcal{L}_{arc}$. This block consists of a convolutional layer with a 1x1 kernel and 32 output channels and a linear layer comprising 512 units. Finally, a classifier layer is implemented to create predictions. Notably, the $\mathcal{L}_{arc}$ is integrated after the simple embedding block, and each layer in this embedding block does not use bias in its calculation.

For the CIFAR10 and CIFAR100 datasets, a modified version of the ResNet18 architecture [20] is employed. In this adaptation, the traditional fully connected layer is replaced with an embedding block inspired by the settings of ArcFace [15]. This block comprises a convolutional layer with an output channel of 512 and a kernel size of 1x1, followed by a batch normalization layer and a linear layer with 512 units. Bias terms are removed from the calculations within these layers. Following the embedding block, a classifier layer is utilized to predict based on the extracted features.

The selection of additional hyperparameters is guided by FedAvg [5] and ArcFace [15] settings. Each client undergoes training for five epochs across 10,000 communication rounds. The batch size is 50 for the MNIST dataset and 32 for CIFAR10 and CIFAR100. The stochastic gradient descent optimizer is employed with a learning rate of 0.001. The loss function computation involves a λ value of 0.5. The margin parameter for the ArcFace loss function is set at 0.4. For centralized training, the model will undergo 10,000 epochs using the same batch size for each dataset. After each epoch, the best accuracy will be assessed to determine the model's optimal performance on the validation dataset.

C. Results and analysis

Table I illustrates the performance comparison of the proposed algorithm, showcasing its effectiveness over other methodologies. The proposed method outperforms other approaches across the MNIST, CIFAR10, and CIFAR100 datasets. Through our non-IID problem creation, our method demonstrates robust performance that is closely aligned with the centralized training approach over 10,000 rounds. In contrast, other methods struggle to reach optimal values and necessitate additional communication rounds to enhance performance on the MNIST dataset. However, when examining FL methods on the CIFAR100 non-IID dataset, all approaches failed to achieve similar centralized training performance. While MNIST comprises only 10 handwritten digits, CIFAR100 encompasses 100 classes representing diverse common objects, making it a more complex dataset. Centralized training on CIFAR100 achieves only 53,78% and 54,40% accuracy for the vanilla loss function and our proposed loss function, respectively, highlighting its complexity. Our non-IID partitioning strategy further amplifies the challenge of CIFAR100. The performance of all FL methods demonstrates a significant 65% drop in accuracy compared to centralized training.

TABLE I: Performance evaluation of our algorithm against other methods under non-IID and IID partitioning scheme

Method	**non-IID**		**IID**
	MNIST	***CIFAR100***	***CIFAR10***
$\mathcal{L}_{ce}$[a]	**0.9903**	0.5378	0.8039
$\mathcal{L}_{ce}$ + $\mathcal{L}_{arc}$[a] (Ours)	0.9900	**0.5440**	**0.8080**
FedAvg [5]	0.8873	0.1639	0.7966
FedProx [6]	0.9095	0.1613	0.7967
MOON [9]	0.9097	0.1553	0.7994
Ours	**0.9866**	**0.1904**	**0.8383**

[a] *Centralized training*

However, our method continues to demonstrate its potential by achieving top performance on the CIFAR10 IID dataset. Interestingly, our performance surpassed that of centralized training, which has unrestricted access to the complete dataset. This result creates our assumption that having full access to the dataset may potentially make the model easy to overfit on the training dataset. Conversely, FL algorithms can only access limited datasets with sufficient samples for each class to train models tailored to each subset, leading to superior global minimum points for the global model compared to centralized training. Still, this assumption needs further investigation for a comprehensive understanding of the algorithm, so this in-depth analysis is left to future research due to the out-of-scope of the study.

Table II illustrates the communication rounds needed for each algorithm to attain specific performance milestones on the MNIST dataset. In centralized training, our proposed loss function achieves a comparable performance level to the vanilla loss function. Nevertheless, our proposed loss function necessitates additional communication rounds to achieve peak accuracy. This is the trade-off when integrating our novel loss function into centralized training to enhance performance, as evidenced in the CIFAR10 and CIFAR100 experiments. On the other hand, our method achieves better performance in the context of FL, facilitating faster model convergence while maintaining superior performance compared to other methods. While other approaches encounter challenges in reaching peak performance, our method achieves higher performance with fewer communication rounds. By incorporating the $\mathcal{L}_{arc}$ to align embedding vectors, our method effectively enhances the aggregation of client weights in each round. However, while the $\mathcal{L}_{arc}$ enhances the global model's performance, it also restricts the algorithm from being used only in classification tasks or tasks tailored to the cross entropy loss function. This limitation reduces the flexibility of the framework to diverse tasks, highlighting the trade-off between task-specific optimization and generalizability within our algorithm.

TABLE II: Comparison of required rounds/epochs for achieving the desired performance of our method against other methods on MNIST Dataset

Method	**Accuracy**				
	85%	***88%***	***90%***	***98%***	***99%***
$\mathcal{L}_{ce}$[a]	7	10	14	81	181
$\mathcal{L}_{ce}$ + $\mathcal{L}_{arc}$[a] (Ours)	7	10	14	81	189
FedAvg [5]	2273	4795	-	-	-
FedProx [6]	2273	5186	7697	-	-
MOON [9]	2273	5412	7684	-	-
Ours	651	675	713	2568	-

[a] *Centralized training*

We utilize t-SNE to visualize the embedding vectors of individual classes extracted from the MNIST test dataset, as depicted in Fig. 2. This visualization facilitates the evaluation of the impact of the $\mathcal{L}_{arc}$ on the feature extracted by the model. The visualization reveals dense overlap among the training classes when employing FedAvg [5] with the standard $\mathcal{L}_{ce}$, as illustrated in Fig. 2a. In contrast, the embedding vectors generated by our method, showcased in Fig. 2, display clear separations between each class, establishing distinct boundaries. This clarity enables the model to effectively differentiate between classes, thereby enhancing the overall performance

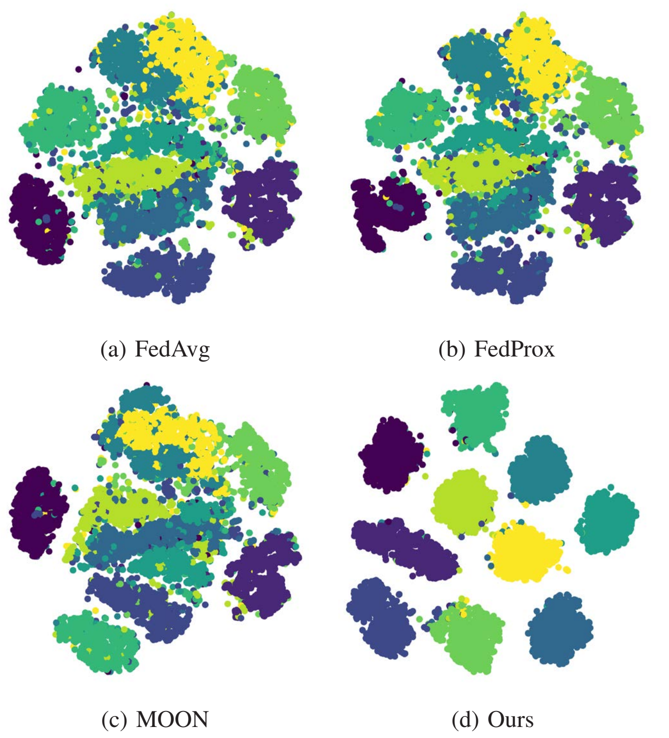
(a) FedAvg (b) FedProx (c) MOON (d) Ours

Fig. 2: Visualization of the embedding vectors for different federated learning algorithms on the MNIST dataset

of the global model. Conversely, FedProx [6] and MOON [9] exhibit high overlap similar to the FedAvg algorithm, as shown in Figs. 2b and 2c, leading to diminished performance in their models. Once again, these results underscore the efficiency of feature representation learning within the FL framework, particularly through the utilization of our algorithm.

IV. Conclusion

This study explores the impact of non-IID and IID scenarios in FL using the MNIST, CIFAR10, and CIFAR100 datasets. To enhance the performance of existing FL techniques, we introduce a baseline approach, which combines the angular margin loss function inspired by ArcFace [15] with standard $\mathcal{L}_{ce}$ function. Through our experiments, our proposed method demonstrates impressive accuracy across all datasets. The angular margin loss function effectively enhances feature representation, empowering the proposed method to tackle both non-IID and IID challenges. However, a limitation of our method lies in its specialization for tasks requiring cross-entropy loss due to the design of the angular margin loss function, which is a significant drawback. In future research, we aim to address these constraints to develop our method to a more general framework for FL applications.

References

[1] D. T. Tran, N. D. H. Nguyen, T. T. Pham, P.-N. Tran, T.-D. T. Vu, C. T. Nguyen, H. Dang-Ngoc, and D. N. M. Dang, "Swintexco: Exemplar-based video colorization using swin transformer," Expert Systems with Applications, vol. 260, p. 125437, 2025.

[2] Q. B. Le, K. Tuan Trinh, N. D. Hung Son, P.-N. Tran, C. T. Nguyen, and D. Ngoc Minh Dang, "Mersa: Multimodal emotion recognition with self-align embedding," in 2024 International Conference on Information Networking (ICOIN), 2024, pp. 500–505.

[3] L. X. Nguyen, P. Sone Aung, H. Q. Le, S.-B. Park, and C. S. Hong, "A new chapter for medical image generation: The stable diffusion method," in 2023 International Conference on Information Networking (ICOIN), 2023, pp. 483–486.

[4] D. K. Phan, P.-N. Tran, N. T. Pham, T. H. T. Le, and D. N. M. Dang, "Innovative multi-modal control for surveillance spider robot: An integration of voice and hand gesture recognition," in Proceedings of the 2024 9th International Conference on Intelligent Information Technology, ser. ICIIT '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 141–148.

[5] Y. Zhou, Q. Ye, and J. Lv, "Communication-efficient federated learning with compensated overlap-fedavg," IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 1, pp. 192–205, 2022.

[6] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," Proceedings of Machine learning and systems, vol. 2, pp. 429–450, 2020.

[7] J. Miao, Z. Yang, L. Fan, and Y. Yang, "Fedseg: Class-heterogeneous federated learning for semantic segmentation," in 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 8042–8052.

[8] G. Long, M. Xie, T. Shen, T. Zhou, X. Wang, and J. Jiang, "Multi-center federated learning: clients clustering for better personalization," World Wide Web, vol. 26, no. 1, pp. 481–500, 2023.

[9] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 10 713–10 722.

[10] M. N. H. Nguyen, S. R. Pandey, T. N. Dang, E.-N. Huh, N. H. Tran, W. Saad, and C. S. Hong, "Self-organizing democratized learning: Toward large-scale distributed learning systems," IEEE Transactions on Neural Networks and Learning Systems, vol. 34, no. 12, pp. 10698-10710, 2023.

[11] H. Q. Le, L. X. Nguyen, S.-B. Park, and C. S. Hong, "Layer-wise knowledge distillation for cross-device federated learning," in 2023 International Conference on Information Networking (ICOIN), 2023, pp. 526–529.

[12] H. Q. Le, Y. Qiao, L. X. Nguyen, L. Zou, and C. S. Hong, "Federated multimodal learning for iot applications: A contrastive learning approach," in 2023 24st Asia-Pacific Network Operations and Management Symposium (APNOMS), 2023, pp. 201–206.

[13] L. X. Nguyen, H. Q. Le, Y. L. Tun, P. S. Aung, Y. K. Tun, Z. Han, and C. S. Hong, "An efficient federated learning framework for training semantic communication systems," IEEE Transactions on Vehicular Technology, vol. 73, no. 10, pp. 15 872–15 877, 2024.

[14] P.-N. Tran, T.-D. T. Vu, N. T. Pham, H. Dang-Ngoc, and D. N. M. Dang, "Comparative analysis of multi-loss functions for enhanced multi-modal speech emotion recognition," in 2023 14th International Conference on Information and Communication Technology Convergence (ICTC), 2023, pp. 425–429.

[15] J. Deng, J. Guo, J. Yang, N. Xue, I. Kotsia, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 10, pp. 5962–5979, 2022.

[16] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815–823.

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

[18] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," J. Mach. Learn. Res., vol. 12, no. null, p. 2825–2830, nov 2011.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.