Dynamic Gradient Tailor for Non-I.I.D. Federated Learning

Jiachen Li
Beijing University of Posts and Telecommunications

Yuchao Zhang*
Beijing University of Posts and Telecommunications

Yiping Li
University of Washington

Xiangyang Gong
Beijing University of Posts and Telecommunications

Wendong Wang
Beijing University of Posts and Telecommunications

ABSTRACT

Federated learning (FL) has yielded impressive results in recent years. But its effectiveness on non-independently and identically distributed (non-i.i.d) data remains challenging and the reasons are also not fully studied. In this work, we disclosed the detrimental gradient interference phenomenon on non-i.i.d FL, and proposed the gradient similarity among clients is an key index which links both non-i.i.d status and the global model performance. Building on this observation, we proposed the Dynamic Gradient Tailor **(DGT)** plugin to improve the FL convergence speed and performance on non-i.i.d data by alleviate the harmful gradient interaction among clients. DGT is a server side gradient optimization plugin that encourages the uploaded gradients to get close to each others by gradient rotation before federated aggregation. Experiments shows the DGT plugin significantly improves the inference accuracy by 5% and achieves 6 times convergence speedup. Further experiment proves that DGT can be easily combined with various mainstream frameworks to further improve the performance on non-i.i.d data.

CCS CONCEPTS

• **Computer systems organization** → *Client-server architectures.*

KEYWORDS

Federated Learning, Non-i.i.d. Data, Gradient Calibration

1 INTRODUCTION

Deep neural networks (DNNs) with millions of parameters have been proven to be outperformed by centralized training on millions of data aggregated from IoT devices, including sensors, mobile devices, and smart vehicles [5, 6]. However, the growing demand for privacy protection making it difficult to transfer IoT devices' private data to the centralized server[9], a mass of data islands hinder the development of high-performance models that driven by big data. Federated learning (FL) [12] is a promising solution that enables collaborative learning across devices without sharing private data [15]. FL is a decentralized framework, client models are collaboratively trained by communicating models or gradients with the aid of a centralized federated server. By offloading model upgrading processes to client devices, FL protects private data locally, eliminating the need of data uploading [9]. With the trend of privacy protection, FL provides a promising approach for achieving high-performance model service while offering private data protection.

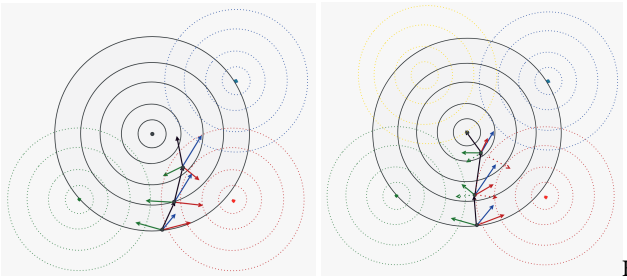
But there is no free lunch. Centralized training paradigm benefit from having access to the entire dataset to ensures the optimization objective's consistency with the true distribution of global data naturally. However, FL faces an unique challenge when dealing with non-i.i.d data due to population or geopolitical factors. The local data samples' statistical properties may differ or correlated across clients, causing the model trained on one client's data to perform poorly on other clients' data. Traditional FL aggregate clients knowledge by sum gradients or parameters from all clients uniformly may caused slow convergence speed and performance loss[24].

Thus, improve the FL effeciency on non-i.i.d data is a hot research point in recent years, many research has demonstrated the impact of non-i.i.d on convergence and proposed algorithms and proposed many methods to improve the performance of non-i.i.d FL [3]. But the root cause of non-i.i.d challenges is still a black-box and correspondingly targeted optimization techniques are also absent. In this paper, we attempt to peek into the black-box of the non-i.i.d challenge. We first links the FL task and the multi-objective learning task, where the federated global model aims to optimize the loss across all sub-task formulated by clients with non-i.i.d private data and corresponding optimization planes. From this perspective, the FL performance is vitally effected by the correlation of sub-task optimization objects. Different, even conflicting optimization objects may cause detrimentally interacts among gradients to be aggregated, further influence the global model upgrading. As shown in 1, individually clients' gradients represented by different colors are aggregated to form the federated global model gradient. The different distribution among clients leading to gradients detrimentally interference with each others hinders the convergence process. Based on insights in MTL Researches that gradient similarity gradients similarity enjoys more similar loss geometries [22]. We further delve into the FL process and found that gradient similarity among gradients to be aggregated also positive correlated to the FL performance and the the degree of variation in client data distribution, and evolves across the FL processes.

Naturally, an interesting question arises : **can we enhance the FL performance by deliberately encourage the similarity of gradients to be aggregated on non-i.i.d setting?**

According to this insight, we design a general plugin to enhance the convergence speed and performance of non-iid FL by leverages clients relatedness to set gradient similarity objectives and adaptively align gradients throughout the FL process, named **DGT**. The overall FL framework with DGT plugin is shown in Figure 2 and the procedure is illustrated in Algorithm 1. Similar to conventional FL framework, to begin with global model parameters initialization

*Correspondence to Yuchao Zhang



(a) (b)

Figure 1: The figure(a) illustrate the FL convergence process on the non-i.i.d data, detrimental interaction among clients leads to biased optimization directions and affects the step-size unintentionally. The figure(b) shows the optimization process of FL with gradient calibration, a desired gradient calibration strategy alleviates detrimental gradient pairs and diminishes contradictions gradient pairs, thereby accelerating convergence and improving performance.

and broadcasting to all clients. Next, clients update their private model based on the fresh global model using their respective data. After client model upgrade process, the accumulated gradients are then sent to the federated server, and the global model parameters are then updated accordingly for the following client broadcasting.

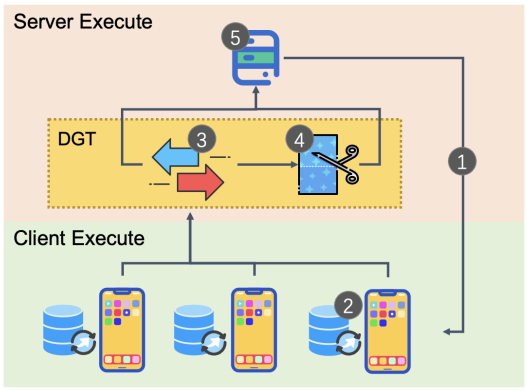


Figure 2: High-level view of non-i.i.d FL with FGT plugin

But unlike the conventional FL frameworks that directly aggregate client gradients into the global model. DGT plugin will calibrate gradients to be upgraded on the federated server, and the calibrated gradient is then used to update the federated global model. DGT aims to minimize detrimental interactions in non-i.i.d FL by rotating the direction of uploaded gradients at the federated server. The rotation of DGT is achieved by replaces the target gradient with a vector consisting of the original gradient and the calibration target. In DGT module the pre-aggregated optimization plan (POP) is proposed as the calibration target which calculate by pre-aggregating all other uploaded gradients. By introducing POP, the computation overhead is restricted to $O(n)$. Also to avoiding unacceptable time overhead, DGT employs a parallel calibration mechanism, where the calibration process relies only on the original gradient. Take the advantages of both low communication and computation cost, the DGT plugin can be easily deployed on the FL system with large scale participator. Additionally, the DGT can be easily deploy on different tasks. The calibration conditions, direction and the rotation step size of calibration step are dynamically decided according to the gradient similarity matrix during federated learning without any handcraft hyper-parameter.

Algorithm 1 Federated Learning

Initialize global model parameters θ
for each round $t = 1$ to T **do**
  Select K client devices
  for each client i in selected clients **do**
    Receive current global model parameters θ_t from the server
    Receive local training data D_i
    Initialize client model with θ_t
    for each local epoch $j = 1$ to E **do**
      Update client model using local optimizer O_i
    end for
    Compute local model updates $\Delta\theta_i = \theta_t - \theta_{t-1}$
    Send $\Delta\theta_i$ to the server
  end for
  replace $\Delta\theta_i$ with $DGT(\Delta\theta_i)$ by applying the DGT module
  Aggregate received model updates: $\Delta\theta_{\text{agg}} = \frac{1}{K}\sum_{i=1}^{K}\Delta\theta_i$
  Update global model parameters: $\theta_{t+1} = \theta_t + \Delta\theta_{\text{agg}}$
end for
Output: Final global model parameters θ

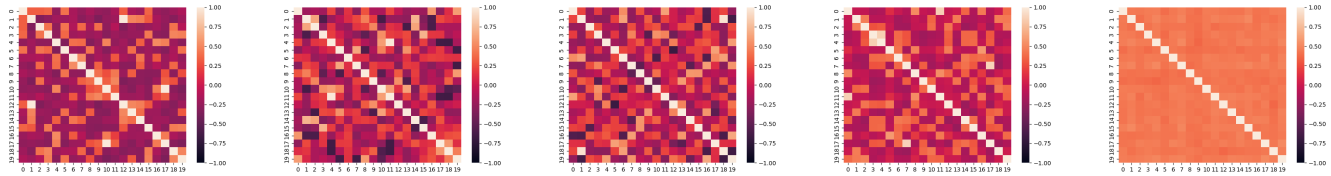
The main contributions of our work are as follows:

- We proposed Dynamic Gradient Transformer(DGT) to improve the FL convergence speed and performance on non-i.i.d data by deliberately encourage the similarity of gradients to be aggregated on the federated server.
- The DGT plugin is a general plug-in that can be easily integrated with mainstream FL methods to further improve the FL efficiency. Also, the DGT module is communication and computation resource-friendly that can be deployed on large-scale FL system.
- Experiments show that the DGT plugin achieves better prediction performance and faster convergence speed than baseline method on non-i.i.d data. Further, we confirmed introduce GDT module in mainstream FL frameworks can further improves the convergence speed by 2-3 times with 10%-20% inference performance improvement, illustrated the generality of the DGT module.

2 OBSERVATIONS

To answer the aforementioned questions, we conducted a comprehensive study to link the optimization trajectory similarity and the performance of global model, and make the following observations.

(1) **clients with more similariy private data distribution enjoy similar loss geometries** We control non-i.i.d setting by manipulating the number of categories of private data

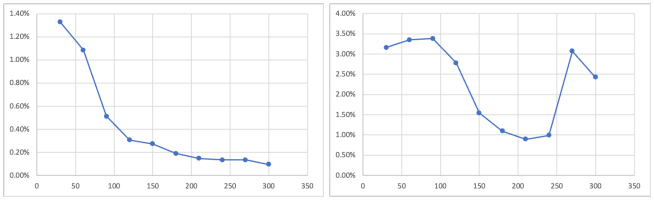


(a) NOC=2 (b) NOC=4 (c) NOC=6 (d) NOC=8 (e) NOC=10

Figure 3: Gradient similarity matrix on different non-i.i.d setting, which control by the number of class (NOC) each client aggregate data. It can be seen that gradient similarity among clients increases with client class.

in each client denoted as $k \in \{2, 4, 6, 8, 10\}$. Figure 3 depicts the optimization trajectory similarity among clients increase with more categories share across clients, also as illustrated in many researches that the number of shared categories correlate positively with the global model performance.

(2) **Gradient similarities evolve across training steps and different across tasks**. By analysis the gradient similarity throughout the FL process. As shown in Figure4, we set a 300 round FL task with 20 clients, each client has data from two categories and mark the gradient similarity in each 30 steps, result shows the gradient similarities evolves throughout the FL process on both MNIST and CIFAR-10 dataset, and the variation of gradient similarity on different tasks is also different.



(a) (b)

Figure 4: (a) and (b) shows the similarity of gradients to be aggregated throughout the FL process on MNIST and CIFAR-10 dataset. The gradient similarity of each round is measured by averaging the cosine similarity of all pairs of gradients during each round of aggregation.

Our analysis highlights the important role of loss geometries similarity in non-i.i.d FL. With these points in mind, we next turn to the design of how to adaptive encourage gradients to be aggregated close to each others thereby enhence their similarity to improve the FL performance.

3 BACKGROUND

Federated Learning (FL) enable models to be trained on decentralized data while preserving privacy, making it ideal for the trend of private data protection. However, previous studies have shown that FL encounters challenges with non-i.i.d data, leading to slower convergence speeds and poorer performance. In this section, we introduce the basic concepts of FL and analyze the challenges presented by non-i.i.d data.

We outline a problem in which N categories are allocated in a feature space X and label space Y in a federated system with K participants. The global model aims to minimizes the weighted sum loss across all clients while accounting for uneven data distribution among them. Consequently, the objective function of the federated global model is defined as :

$$F(w) = \frac{1}{N} \sum_{i=1}^{N} \frac{n_i}{N} f(w; S_i) \tag{1}$$

ω represents the global model weights, n_i denotes the number of private data that client i contributes to S_i, and $f(w; S_i)$ is the loss function of the global model on the private dataset of the i-th client. In client model training process, clients optimize their private model based on the global model on private data. Thus, the objective function of each client is defined as

$$f_i(w_i) = \frac{1}{n_i} \sum_{j=1}^{n_i} f(w_i; x_j, y_j) \tag{2}$$

w_i indicates the private model parameter of client i, and $f(w_i; x_j, y_j)$ is the loss function of the i-th client's private model on the j-th data point. Further, clients perform private model updates based on their own private datasets using optimization methods like stochastic gradient descent (SGD) algorithm as follows:

$$w_i^{t+1} = w_i^t - \eta \nabla_i^t \tag{3}$$

where η represents the learning rate, and ∇_i^t denotes i-th client's gradient at iteration t.

After the local update progress, clients uploads private model or gradient to the federated server, and the federated server aggregates the gradients to update the global model parameters as

$$w_{t+1} = w_t - \eta \nabla_{t+1}$$

$$\nabla_{t+1} = \sum_{i=1}^{N} \frac{n_i}{n} \nabla_i^{t+1}$$

w_t denotes the current global model parameters, and ∇^t aggregation of client gradients.

4 RELATED WORK

FL aims to solve the problem of joint training among locally stored datasets due to data privacy and security concerns. FedAvg [14] is the pioneering work in the field of FL, using SGD in local model fine-tune and weighted aggregation according to the size of client private dataset. However, the non-iid data among different clients

hinders the FL performance. To address this challenge, researchers have proposed many methods to improve the FL algorithm. Starting from the optimization process on the client side, FedProx [4] adds a new L_2 regularization term to the loss function of the local model to control the difference between the local and global models. FedNova [20] utilizes different optimizers and hyper-parameters for local models to control the extent of updates during training and reduce the diversity between devices. FedDFIA [21] adds a constraint of the distribution feature to the loss function of the global model to constrain the distance between local and global models. Fedmatch [18] adds a low-noise regularization term to the global model so that it can adapt to the changes of local data to some extent. Starting from the server-side aggregation direction. FEDPNS [16] introduces a new network architecture that excludes client models conflicting with the global model during federated aggregation to control the distance between local and global models and reduce the impact of non-iid data on FL. MOCHA [7] integrates multiple objectives into a unified framework to optimize multiple objectives in FL models such as model accuracy, communication efficiency, and edge device energy consumption.

As for the client selection and weighted aggregation mechanism. Two new client selection criteria and a greedy client selection algorithm is proposed to improve the federated performance on non-IID clients [11]. FedRANK [2] introduces a global ranking mechanism that sorts all users participating in FL according to some specific rules, and adjusts the gradient updating contribution of each user based on the user's ranking information to eliminate the data distribution bias among users. HeteroFL [10] proposed heterogeneous grouping, dynamic communication, and adaptive learning module to improve the FL performance of non-IID data in heterogeneous edge network environment. FedOpt [19] proposes a client selection method based on optimization objectives to find a suitable subset of clients to minimize the global loss. FedAvg-MSP [23] proposed minimum contamination subset (MSP) and select clients that are similar to the global distribution and have a significant impact on the global objective function to participate in training.

From the perspective of personalized FL, ArFL [8] regards FL as a multi-task learning problem. In the FL process, ArFL selects a specific auxiliary task for each client that is closely related to the client's data, and obtains more information through this auxiliary task to improve the model's performance. FedRep [1] views multiple rounds of FL as a sequential decision-making process. By repeating the training on a portion of the data at each round, it reduces the uncertainty of client data and improve inference accuracy, and improves model robustness and generalization ability. Hermes[17] introduces personalized pruning and adaptive aggregation. Personalized pruning enables each client to customize its local model based on its own data distribution, and the adaptive aggregation enables heterogeneous structure model aggregation.

5 DYNAMIC GRADIENT TAILOR

Based on the above insights. It is vital to design a gradient calibration plugin for the FL framework, which should meet the following requirements:

(1) The plugin should enhancing the similarity of the gradients to be aggregated and reducing detrimental interactions while maintaining the strict privacy protection paradigm of conventional FL framework, only model parameter and related gradients transferred between clients and server .
(2) The plugin should adapt to the evolving gradient similarity, various tasks and models. The key parameter should be automatically decided according to the snapshot of the FL environment without expert knowledge and handcraft setting.
(3) Additional client-side computation, communication overhead and time-cost introduced should be strictly limited to facilitate the participation of devices with limited resources, as well as limited server-side computational cost to enable the large-scale deployment.

To fulfill these requirements, we first establish links between the non-iid FL and multi-task learning (MTL) two research fields. In the case of non-iid FL, the optimization task of client private models also can be seen as different tasks, and the federated global model needs to integrate knowledge of different clients to achieve optimal performance at a global distribution which is similar to shared parameters in multi-task learning aims to learn knowledge from different tasks. Inspired by gradient optimization techniques used in MTL, we propose the Dynamic Gradient Tailor (DGT). DGT is a general FL plugin to minimize detrimental interactions and encourage the gradient similarity by directly rotating the direction of uploaded gradients at the federated server. The calibration paradigm of DGT can be abstracted as $DGT(g_k) = \alpha_1 \cdot g_k + \alpha_2 \cdot g_j$, that is DGT replaces the original gradient g_k with a vector consisting of g_k and g_j, where g_j is the calibration target vector.

Within this rotation paradigm, the design of the calibration target vector is crucial as it significantly influences the effectiveness of calibration. Pairwise calibration is proposed in MTL where the gradients of each task are calibrated towards the gradients of all other tasks in a random order. However, we argue that this approach has two drawbacks.

Firstly, Pairwise approach is not suitable for FL due to the potentially massive number of participating clients in FL system, compared to the limited number of tasks in MTL tasks. Applying pairwise calibration in FL framework requires calibrate each gradient to be aggregated with all others individually, resulting in a computational complexity of $O(n)^2$ which is unacceptable. The unacceptable computation overhead hinders the scalability of the FL framework with the DGT plugin, making it inappropriate for deployment in FL systems such as IoT federations, where a large number of clients are participating. Secondly, the pairwise approach projects the original gradients to the other uploaded gradients in random order, which may result in conflicting calibration directions. Thereby introduces uncertainty of the calibration performance. To overcome the above two concern. We propose pre-aggregated optimization plan (POP) as the projection target of DGT. Specifically, for each client k, DGT first pre-aggregate all other uploaded gradients g_j except for g_k in federated server to obtain the calibration target for client k. Subsequently, we employ the following formula to project the gradients

of client k onto the pre-aggregated calibration plane as

$$DGT(g_k) = \alpha_1 g_k + \alpha_2 \cdot POP_k \tag{4}$$

$$POP_k = \sum_{j=1}^{K} g_j, j \neq k \tag{5}$$

By introducing POP, each client needs to be calibrated only once, thus the computation overhead is significantly reduced from $O(n)^2$ to $O(n)$, where n is the number of FL participants. Therefore, in terms of computational overhead, DGT plugin does not requires additional client-side computational overhead, thus can be well deployed on FL systems with a large number of weakly-resourced devices, and the computational overhead on the server-side is also acceptable, in particular federated servers tend to be server devices with high computing power. Meanwhile, the certainty of the calibration performance is improved by projecting the original gradient toward the POP representing the direction of the average gradient of the other clients. In terms of communication overhead, the process of calibrated projection of DGT relies only on the gradient to be aggregated on the server side, thus does not introduce additional communication overhead. It is also worth noting that all clients undergo gradient calibration with DGT module in parallel. Specifically, the calibration process of each client is independent with the others, thus the time cost introduced by the DGT will not increase linearly with the number of FL clients. Above all, the acceptable server-side computational and time cost and zero client-side cost allows DGT plugin to be deployed in large-scale client FL applications.

Utilizing the POP-based calibration paradigm, we fixed $\alpha_1 = 1$ and by applying Law of Sines in the plane of g_k and POP_k, and we further solve for the value of α_2 and derive the calibrated gradient of client k as

$$DGT(g_k) = g_k + \alpha_2 \times POP_k \tag{6}$$

$$\alpha_2 = ||g_k|| \frac{POP_k \sqrt{1-(\Phi_k)^2} - \Phi_k \sqrt{1-(POP_k)^2}}{||POP_k|| \sqrt{1-(POP_k)^2}} \tag{7}$$

To adapt to different tasks, FL models and the evolving gradient similarities during training, we utilize the information about changes in gradient similarity matrix to determine if calibration is necessary. We maintain a historical gradient similarity baseline Φ_k^T for each client, which represents the similarity between the client and the POP before, forming an $N \times N$ gradient similarity matrix throughout the FL process. The similarity baseline is updated in each FL step dynamically by employing a sliding average of their historical gradient similarities as

$$\Phi_k^T = \alpha \cdot \Phi_k^T + (1-\alpha) \cdot \Phi_k^t \tag{8}$$

where Φ_k^t is measured by cosine similarity as $\Phi_k^t = \frac{g_k \cdot POP_k}{\|g_k\| \|POP_k\|}$ and initialized with 0 at the beginning of the FL process. With the help of similarity baselines, gradient of client k will be calibrated if the gradient similarity is decreading in the current round which can be formalized as $\Phi_k^t < \Phi_k^{T-1}$. Decreasing similarity indicates that the difference between the optimization direction of client k and the others is gradually increasing. Conversely, gradient k will be aggregate directly without calibration because the gradient direction is gradually approaching the POP.

Algorithm 2 Procedure of DGT

Input: Number of parties N, uploaded gradients g_k $K = \{1, 2, ..., K\}$

Output: Calibrated Gradient to be aggregated $DGT(g_k)$

```
POP_k ← pre-aggregated gradients of other clients by eq.5
Φ_k^t ← CosineSimilarity(g_k, POP_k)
if Φ_k^t < Φ_k^T then
    α = ||g_k|| (POP_k √(1−(Φ_k)^2) − Φ_k √(1−(POP_k)^2)) / (||POP_k|| √(1−(POP_k)^2))
    DGT(g_k) = g_k + α · POP_k
end if
Φ_k^T ← update similarity baseline of client k by eq.8
return DGT(g_k)
```

The process of the DGT module is illustrated in Algorithm 2. To begin with a given client k and the uploaded gradient denote g_k. DGT first compute the pre-aggregated calibration target plane for the client POP_k with Equation 5. Next, DGT measures the gradient similarity Φ_k^t of g_k and POP_k, and compared with the similarity baseline Φ_k^T to decide if g_k needs to be calibrate. If so, DGT calibrate g_k by applying Equation 7 and the calibrated gradient $DGT(g_k)$ will be returned for subsequent federated aggregation. Otherwise, the raw gradient g_k will be returned directly without alteration. In the end of DGT, the similarity baseline of client k is rebuild with the fresh Φ_k^t with equation 8 no matter if calibration performed.

6 EXPERIMENTS

We first verify the performance and applicability of the DGT in the this session. We first validate the DGT with a classic FL framework Fedavg[13] on two simulated non-i.i.d datasets to verify the performance improvement of DGT on different tasks. Further, we combine DGT with other baselines including Fedprox and Scaffold to verify the DGT plugin can be easily combined to the different mainstream FL frameworks to improves the performance on non-i.i.d data.

6.1 System, Non-iid Simulation, and Model

All experiments were implemented in TensorFlow 2.0 running on an AWS server equipped with an Intel Xeon E5-2630@2.6GHz and Tesla-T4 GPU. The FL system comprised 20 clients, and the training process lasted for 300 rounds. To increase the credibility of the conclusions, each experiment was repeated ten times. The global model was randomly initialized, with a batch size of 256 and 1 epoch for the local model training. To simulate the non-i.i.d setting, client's private data are belonging to two randomly assigned categories. The client and server models in the federated system used the same three-layer CNN network structure for MNIST and CIFAR-10 dataset.

6.2 Performance of the DGT plugin

As illustrated in Figure5 (a) and (b), introducing DGT on the top of the baseline framework significantly enhances both convergence speed and performance. On MNSIT, CIFAR-10, and Fashion-MNIST datasets, the introduction of the DGT plugin improves the convergence speed of the federated global model by 2X, 3X, and 5X, while

also improving the inference accuracy of global performance by 5%, 10%, and 15%.

We believe that the above improvement stems from the DGT plugin, which effectively mitigates harmful interactions and enhances the similarity of gradients to be aggregated, thereby leading to a convergence process that aligns with the sub-objective optimization planes of multiple clients. To further explain the effectiveness of the DGT plugin, we analyzed the changes in similarity of aggregated client gradients on all four datasets. And highlights the impact of DGT plugin to gradients to be aggregated. As shown in Figure 5 (c) and (d), DGT improves the similarity of client gradients to be aggregated throughout the FL process on all three datasets, while noticeably alleviate the detrimental interaction gradient pair with an inner product less than zero.

Secondly, we compared FedAvg-DGT with two widely-used federated learning frameworks, FedProx and Scaffold. The results clearly indicate that introducing DGT into the classical FedAvg framework can achieve performance surpassing mainstream federated frameworks. This results further illustrates that DGT plug-in can bring vital convergence acceleration and performance improvement, which is important for the application of federated learning in noniid scenarios.

Hence, we raise the question: Can DGT be integrated with mainstream federated learning methods to further enhance federated performance on non-iid data?

6.3 Generalization of the DGT plugin

To answer this question, we further combined DGT with two widely-utilized FL frameworks including FedProx and Scaffold.

Figure**??** reflects the performance gains from the introduction of DGT plug-ins on top of mainstream federated learning frameworks. It can be seen that DGT improves inference accuracy of 5% and 10% based on Scaffold, while achieves 2X convergence acceleration. Compared to vanilla Scaffold, introduce DGT plugin brought 5% improvement of inference accuracy and 2X convergence speed up. This fully demonstrates the versatility of DGT plug-in for performance improvement and convergence acceleration of noniid data federation. Importantly, the introduction of the DGT plugin does not require any specific design or modification to the original federated learning framework; rather, it entails gradient calibration through the DGT module prior to performing federated aggregation.

7 CONCLUSION

In this paper, we proposed dynamic gradient tailor (DGT) a novel FL plugin to improve the FL efficiency and performance on Non-i.i.d data. We first analyze the relationship between data distribution consistency, gradient similarity, and federation performance. Next we propose the DGT to reduce harmful interactions among uploaded gradients and encouraging improve gradients similarity by directly rotate gradients to target vector which consists by the other gradients. By applying the pre-aggregated optimization plan and Adopt the asynchronous calibration mechanism, DGT incurs limited server-side overhead. Experimental results confirms that DGT improves both convergence speed by 2-6X and performance by 3.5%-15%, and support that DGT is a general plugin which can be integrated with mainstream FL frameworks to further improve the efficiency and performance on non-i.i.d data.

REFERENCES

[1] Prateek Chaudhari, Sayan Zhang, Josephine Chen, Anisha Suresh, Udaya Kannan, Pieter Abbeel, and Michael I Jordan. 2021. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:2103.03281* (2021).

[2] Kai Chen, H. Brendan Huang, Jinhui Liu, and Qiang Zhang. 2020. Federated meta-learning with fast convergence and efficient communication. In *International Conference on Learning Representations*.

[3] Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang. 2021. Personalized cross-silo federated learning on non-iid data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 7865–7873.

[4] Li Jianyu and Fedprox Pml. 2019. Fedprox: Federated optimization with proximal gradient descent. In *International Conference on Learning Representations*.

[5] Y Lecun, Y Bengio, and G Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436.

[6] Ang Li, Jingwei Sun, Pengcheng Li, Yu Pu, Hai Helen Helen Li, and Yiran Chen. 2021. Hermes: an efficient federated learning framework for heterogeneous mobile clients. (2021).

[7] Jianqing Li, Yujie Chen, Boyan Zhou, Yuxi Qiu, Haoyi Wu, Miao Fang, and Junzhou Huang. 2021. MOCHA: Multi-objective communication-efficient Federated Learning with holistic quality optimization. In *International Conference on Learning Representations*.

[8] Jianyu Li, Xiaoyu Liang, and Lilian Weng. 2020. ArFL: Federated learning with auxiliary task reweighting. In *International conference on machine learning*. PMLR, 6196–6205.

[9] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine* 37, 3 (2020), 50–60.

[10] Xiang Lian, Liang Fu, Shiwei Zhang, Jiashi Feng, and Zhi Xiong. 2020. HeteroFL: A federated learning framework for heterogeneous devices in distributed edge environment. In *International Conference on Learning Representations*.

[11] Anxin Liu, Yang Li, and Jian Tian. 2020. Client selection for federated learning with non-IID data. *IEEE Transactions on Communications* 69, 8 (2020), 5237–5249.

[12] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[13] H. B. Mcmahan, E. Moore, D. Ramage, S. Hampson, and Bay Arcas. 2016. Communication-Efficient Learning of Deep Networks from Decentralized Data.

[14] H. Brendan McMahan, Eider Moore, Daniel Ramage, Sachin Hampson, and Blaise Aguera y Arcas. 2016. Communication-efficient learning of deep networks from decentralized data. In *International Conference on Artificial Intelligence and Statistics*.

[15] Jihong Park, Sumudu Samarakoon, Mehdi Bennis, and Mérouane Debbah. 2019. Wireless network intelligence at the edge. *Proc. IEEE* 107, 11 (2019), 2204–2239.

[16] Felix Sattler, Simon Wiedemann, Klaus-Robert Muller, and Wojciech Samek. 2021. Clustered Federated Learning: Model-Parallelism v.s. Data-Parallelism. In *International Conference on Artificial Intelligence and Statistics*. PMLR.

[17] Xiaoyang Wang, Xiangru Chen, Cong Shi, Qingyang Li, and Meikang Qiu. 2021. Hermes: An Efficient Federated Learning Framework for Heterogeneous Mobile Clients. *IEEE Transactions on Parallel and Distributed Systems* 32, 3 (2021), 703–717.

[18] Yang Wang, Tingting He, Kai Zhao, Kin K Leung, and Khaled B Liu. 2021. Fedmatch: Communication-efficient federated learning with adaptive matchmaking. In *Thirty-second Conference on Neural Information Processing Systems*.

[19] Yong Wang, Xiaoqing Li, Xiaolin Wu, and Jiajia Chen. 2020. FedOpt: A new optimization approach for federated learning. *Sensors* 20, 10 (2020), 2812.

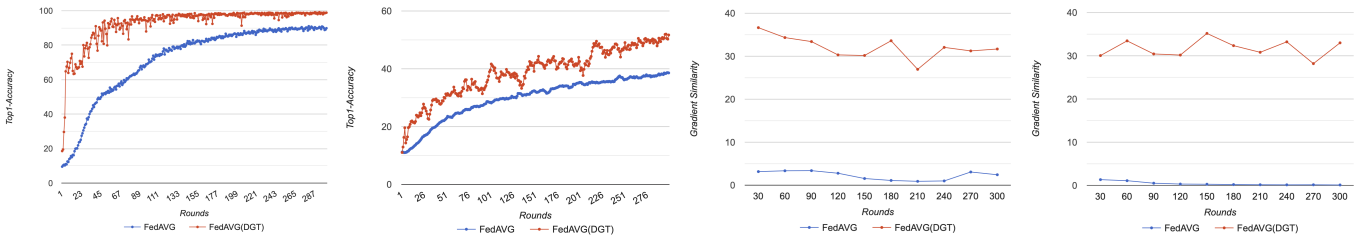
[20] Yang Wang, Toby Tuor, Theodoros Salonidis, Kin K Leung, Christian Makaya, Tingting He, Kai Zhao, and Khaled B Liu. 2020. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127* (2020).

[21] Qiang Yang, Yang Liu, Tianjian Chen, and Yu Tong. 2019. Federated learning via local model aggregation. In *International Conference on Learning Representations*.

[22] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, and Chelsea Finn. 2020. Gradient Surgery for Multi-Task Learning. (2020).

[23] Weiwei Zhao, Zhecheng Du, Tongxin Wang, Ji Liu, Chaoqun Wu, Wenjian Wang, Jian Ma, and Yadong Liu. 2021. Federated Learning with Minimum Contamination Subset Selection. In *International Conference on Learning Representations*.

[24] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. 2018. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582* (2018).



(a) (b) (c) (d)

Figure 5: (a) and (b) shows the performance of DGT on MNIST, and CIFAR-10. On all above dataset, DGT improves the FL efficiency under limited communication resource by achieves better performance with fewer communication rounds. (c) and (d) shows the gradients similarity and conflicts throughout the FL process on MNIST and CIFAR-10 dataset. Indicating that DGT makes the direction of gradients more consistent across clients during global model updating.