AFM3D: An Asynchronous Federated Meta-Learning Framework for Driver Distraction Detection

Sheng Liu, *Graduate Student Member, IEEE*, Linlin You, *Senior Member, IEEE*, Rui Zhu, Bing Liu, Rui Liu, Han Yu, *Senior Member, IEEE*, and Chau Yuen, *Fellow, IEEE*

***Abstract*— Driver Distraction Detection (3D) is of great significance in helping intelligent vehicles decide whether to remind drivers or take over the driving task and avoid traffic accidents. However, the current centralized learning paradigm of 3D has become unpractical because of rising limitations on data sharing and increasing concerns about user privacy. In this context, 3D is further facing three emerging challenges, namely data islands, data heterogeneity, and the straggler issue. To jointly address these three issues and make the 3D model training and deployment more practical and efficient, this paper proposes an Asynchronous Federated Meta-learning framework called AFM3D. Specifically, AFM3D bridges data islands through Federated Learning (FL), a novel distributed learning paradigm that enables multiple clients (i.e., private vehicles with individual data of drivers) to learn a global model collaboratively without data exchange. Moreover, AFM3D further utilizes meta-learning to tackle data heterogeneity by training a meta-model that can adapt to new driver data quickly with satisfactory performance. Finally, AFM3D is designed to operate in an asynchronous mode to reduce delays caused by stragglers and achieve efficient learning. A temporally weighted aggregation strategy is also designed to handle stale models commonly encountered in the asynchronous mode and in turn, optimize the aggregation direction. Extensive experiment results show that AFM3D can boost performance in terms of model accuracy, recall, F1 score, test loss, and learning speed by 7.61%, 7.44%, 7.95%, 9.95%, and 50.91%, respectively, against five state-of-the-art methods.**

***Index Terms*— Driver distraction detection, federated learning, asynchronous federated learning, federated meta-learning.**

Manuscript received 4 December 2022; revised 2 June 2023, 17 October 2023, and 17 January 2024; accepted 18 January 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62002398; in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2023A1515012895; in part by the National Research Foundation Singapore and DSO National Laboratories under the AI Singapore Program (AISG) under Award AISG2-RP-2020-019; and in part by the RIE 2020 Advanced Manufacturing and Engineering (AME) Programmatic Fund, Singapore, under Grant A20G8b0102. The Associate Editor for this article was S. A. Birrell. *(Corresponding author: Linlin You.)*

Sheng Liu and Linlin You are with the School of Intelligent Systems Engineering, Sun Yat-sen University, Shenzhen 518107, China, and also with the Guangdong Provincial Key Laboratory of Intelligent Transportation Systems, Guangzhou 510275, China (e-mail: youllin@mail.sysu.edu.cn).

Rui Zhu is with the Institute of High Performance Computing, A*STAR, Singapore 138632.

Bing Liu is with BYD Auto Company Ltd., Shenzhen 518118, China, and also with the School of Computer Science and Technology, University of Science and Technology of China, Hefei 230026, China.

Rui Liu and Han Yu are with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798.

Chau Yuen is with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore 639798.

Digital Object Identifier 10.1109/TITS.2024.3357138

I. Introduction

In RECENT years, to meet the increasing travel demands, the number of vehicles has risen significantly. Moreover, thanks to the rapid development and adoption of advanced technologies such as Mobile Edge Computing (MEC), Artificial Intelligence (AI), and Information and Communication Technology (ICT), vehicles nowadays are equipped with diverse sensors (such as cameras) as well as sufficient computing capacities to support various AI tasks, e.g., to train a deep learning model based on image data to detect driver distraction [1], [2].

Since fully autonomous driving vehicles still have a long way to go due to technical and ethical issues, humans are still the main operators of vehicles in the field. However, unlike machines that can always focus their attention on the driving task, humans may be distracted and thus, endanger the safety of driving. According to several survey reports [3], [4], most traffic accidents are directly or indirectly caused by driver distraction. Thus, intelligent vehicles are increasingly being equipped with Driver Distraction Detection (3D) capabilities to perform timely interventions, e.g., reminding drivers of their states or taking over the driving task, in order to improve road safety and avoid potential traffic accidents.

However, the increasing restrictions on data security and user privacy make it unfeasible to solve the 3D problem with a centralized learning paradigm. The sensitive data of drivers collected by vehicles may leak user privacy when transmitted to a data center for model training [5]. Although AI-Generated Context (AIGC) [6] model such as GPT-4 may generate artificial data, it still relies on massive data for training in the cloud (which may also violate the regulations). Moreover, in view of data diversity and model robustness, artificial data can't completely reflect real-world scenarios, hence the real and private data of drivers are still necessary for 3D. As an emerging learning paradigm, Federated Learning (FL) [7] can bridge data islands caused by regulations and leverage computing and communication resources owned by multiple entities in a distributed and collaborative manner. In a general FL system, multiple clients train local models based on their own data and then upload the model parameters to a server for global aggregation. Once the global model is updated on the server side, it will be transmitted to the client side for another round of local training. Although FL has been widely applied to various data-sensitive domains such as healthcare [8],

the feasibility study of FL in the context of 3D is still limited.

With the support of FL, although the data island issue of 3D can be tackled, the data heterogeneity and straggler issues remain open. Since data sharing is not permitted in FL, and local data are related to driver behavior and preference, there exists intrinsic data heterogeneity [9], [10], such as imbalanced data sizes, missing certain classes, and non-identical data distributions, which impedes the application of FL in 3D. There is still a lack of discussion about how to learn a shareable model with better initial parameters collaboratively based on such heterogeneous data and then localize it quickly at each client for a performance boost. Moreover, as intelligent devices to manage and process driver data, vehicles, in general, have different communication capacities due to their prices, manufacturers, and running environments. Thus, it is not well suited for synchronous FL frameworks (such as FedAvg) requiring all clients to work at the same pace. This is because the stragglers [11], whose computing powers are lower than others, may affect the overall learning performance. How to unify vehicles with dynamic and unstable communication conditions and make the collaboration mode more practical, is still an open problem.

To jointly tackle the above-mentioned three challenges faced by 3D in the new context, this paper proposes a novel asynchronous federated meta-learning framework, named AFM3D. First, to bridge data islands, a general FL framework is designed to utilize the data of drivers and also the dispersed resources of vehicles in a collaborative and private-preserving manner. Second, to address the data heterogeneity issue and train an initialized model that can adapt to new data and tasks quickly, AFM3D incorporates the FL framework with meta-learning to train a global meta-model with high generability. Finally, to tackle the straggler issue in the synchronous mode of FL in the context of 3D, as shown in Fig. 1 (A), AFM3D introduces an asynchronous mode, i.e., once the predefined condition is satisfied (i.e., the maximum waiting time is reached), the server will aggregate all received models from clients immediately to update the global model as illustrated in Fig. 1 (B). Since the asynchronous FL (AFL) doesn't need to wait for stragglers as compared to the synchronous FL (SFL), a temporally weighted aggregation strategy is also implemented in the framework to further address the latent model issue caused by AFL.

In general, the main contributions of this paper are summarized as follows:

- Compared to conventional solutions, the proposed framework AFM3D can not only accommodate devices with capabilities and availabilities changing spatiotemporally but also train a sharable and personalizable model with user privacy protected;
- To address the spatiotemporal heterogeneity among devices, AFM3D implements an asynchronous learning procedure that can not only accelerate the learning speed through unblocking collaboration among devices but also improve the model performance by the temporally weighted global model aggregation;
- To support the needs of rapid adaptation and accurate personalization, the proposed framework separates the model learning and deployment phases to first train an initial model with structural knowledge and then fine-tune it locally for personalized models to better support the 3D task of different users.

TABLE I
LIST OF ABBREVIATIONS USED IN THIS PAPER

Abbr.	Meaning
3D	Driver Distraction Detection
FL	Federated Learning
AFL	Asynchronous Federated Learning
SFL	Synchronous Federated Learning
KD	Knowledge Distillation
MAML	Model-Agnostic-Meta-Learning
FOMAML	First-Order Model-Agnostic-Meta-Learning
FM	Federated Meta-learning
AFM	Asynchronous Federated Meta-learning
DNNs	Deep Neural Networks
TW	Temporal Weight
SFD	State-Farm-Distracted-driver-detection dataset
AUC	American University in Cairo distracted driver detection dataset

To the best of our knowledge, this paper is the first of its kind to support practical 3D tasks by investigating Federated Meta-learning (FM) and implementing it in asynchronous mode. According to the holistic evaluation based on two standard 3D datasets and six popular AI models, the proposed framework AFM3D can improve test accuracy, recall, and F1 score by 7.61%, 7.44%, and 7.95%, respectively, reduce test loss by 9.95%, and shorten learning time by 50.91% compared with the best among five state-of-the-art methods.

The remainder of this paper is organized as follows. Section II summarizes related work about 3D. AFM3D is presented and evaluated in Section III and Section IV, respectively. Section V concludes the work and sketches the future research directions. Note that for legibility, the abbreviations used in this paper are summarized in Table I.

II. RELATED CHALLENGES AND SOLUTIONS

In this section, first, three challenges encountered by 3D in the new context are identified, and then related solutions are summarized to disclose the current research gap.

A. Challenges of Driver Distracted Detection (3D)

In general, three urgent challenges are faced by 3D, namely:

- **C1: Data Island.** With the promulgation of regulations related to data security and user privacy worldwide, the traditional centralized paradigm, i.e., sharing the private data of drivers sensed by vehicles, is impracticable, resulting in the data island issue, i.e., a large amount of isolated data are dispersed in each vehicle and can not be

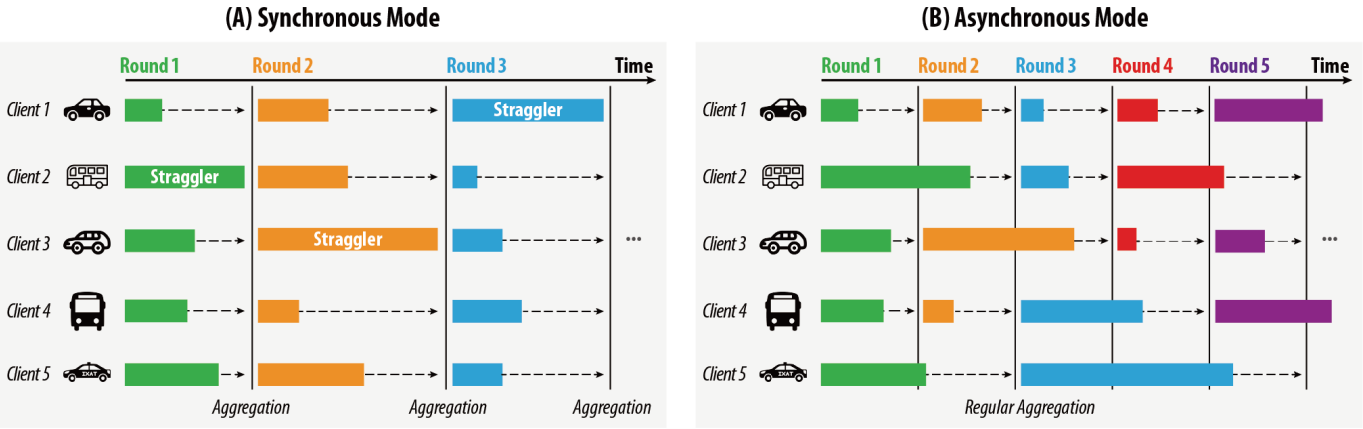


Fig. 1. The comparison between FL in synchronous mode and asynchronous mode.

fully utilized to train a high-performance AI model and support the 3D task efficiently and effectively [5], [12].

- **C2: Data Heterogeneity.** Since data sharing among vehicles is forbidden, and the data collection is related to specific driver behavior, the local data is heterogeneous from each other. E.g., some vehicles may have massive driver data with low resolution while other vehicles may have very small amounts of driver data with high quality. Without tackling such a heterogeneity issue properly, it will cause model deterioration [13], e.g., the model performance dropping dramatically on unseen data [14].
- **C3: Straggler Issue.** Since moving vehicles are equipped with different computing and communication devices, and their connection status may vary according to the actual environments (e.g., suburbs with poor communication infrastructures), when collaboratively training a shareable model, some impotent vehicles, i.e., the stragglers, may impede the overall learning process [15] and also the overall training efficiency.

Accordingly, several solutions have been proposed to address the three challenges separately, which will be discussed in the following subsections.

B. Solutions to Data Island

As a novel distributed learning paradigm, FL is drawing increasing attention from both academic and industry communities to bridge data islands efficiently and effectively. A typical FL framework contains a server and several clients to collaboratively learn a global model [7]. Although the sensitive data of clients will not be shared, the parameters of the local model trained based on local data will be transmitted to the server to help form a global model. Moreover, FL has been widely applied to the transportation domain to fulfill specific tasks, e.g., an FL and graph representation-based algorithm called FedSTN is proposed to predict urban traffic flow accurately [16]; and an FL and Internet of Electric Vehicles-based mechanism named FL-PDMIM is designed to place mobile charging stations effectively [17]. Specifically, an SFL-based framework, called CNN-Bi-LSTM, is present to detect driver distraction [5]. However, it is still missing to conduct in-depth research on FL for 3D to harness isolated data in a spatiotemporally changeable environment.

C. Solutions to Data Heterogeneity

In general, meta-learning and knowledge distillation (KD) are two widely discussed solutions about data heterogeneity. As a learning to learn approach, meta-learning can encode meta-knowledge into an initialized model with high generalizability [18]. In this context, the data heterogeneity can be efficiently tackled and model localization or personalization can be easily conducted to support 3D tasks in new contexts with optimized performance [19]. As for the meta-model training algorithms, such as MAML (Model-Agnostic-Meta-Learning) [20], [21] and its various variants, e.g., Reptile [22], have been proposed and widely used for gradient-based models in classification, regression, and reinforcement learning [23]. Moreover, FM [24], [25] has been utilized in several real-world transportation tasks to harness sensitive and heterogeneous data, e.g., adopting FM to help car parks with poor-data predict parking occupancy [26], [27], and using layer-wise FM to consider data growth phenomenon and tackle communication consumption issue for image classification tasks [28]. However, how to adopt FM to support 3D is still undiscussed.

As for KD, it aims to learn a student model using knowledge distilled from teacher models. Canonical KD for FL research requires a proxy dataset on the server side to measure and minimize the discrepancy between the student model and teacher model [29], which is impractical in many real-world scenarios due to data limitations. Such that, more recently, the data-free KD has been studied to distillate knowledge without proxy dataset via generative learning [30], [31]. However, compared with meta-learning, KD methods mainly focus on the general performance instead of the personalized performance of the local model required by different users [32], [33].

D. Solutions to Stragglers

Generally speaking, the straggler issues can be addressed by managing clients asynchronously or hierarchically. As for asynchronization, it is well-known as AFL, which relieves

TABLE II
THE OVERALL EVALUATION OF REVIEWED SOLUTIONS (● SUPPORTED ○ NOT SUPPORTED)

Solutions	Data Island	Data Heterogeneity	Straggler Issue	Highlights (+ pros and - cons)
Rishu et al. [5]	●	○	○	A privacy-enabled SFL algorithm for driver behavior analysis: + Training CNN-Bi-LSTM model without sharing raw data - Data heterogeneity and straggler issues are undiscussed
Finn et al. [21]	○	●	○	A general online meta-learning algorithm: + Extending MAML to online learning - How to apply FL in 3D is not discussed
Guo et al. [28]	●	●	○	A synchronous FM-based approach for 3D: + Using a layer-wise mechanism of DNNs to tackle incremental data - Asynchronous mode to train DNNs in FL is undiscussed
Shang et al. [33]	●	●	○	An adaptive KD loss-based SFL algorithm for 3D: + Using bidirectional KD and attention loss to tackle heterogeneous data - Straggler issue is not addressed
Doshi et al. [32]	●	●	○	A communication-efficient SFL algorithm for driver activity recognition: + Using KD to transfer information efficiently within resource-limited devices - The asynchronous learning and personalized deployment are not studied
Ma et al. [34]	●	○	●	A semi-asynchronous FL approach with convergence guarantee: + Optimizing hyper-parameters in FL to improve performance - Model adaptation ability required by 3D is not discussed
Damaskinos et al. [35]	●	○	●	A staleness-awareness online AFL mechanism: + Using similarity of tasks to update gradients with delay adaptively - How to apply FM in 3D is undiscussed
Wang et al. [36]	●	○	●	A multiple clusters-based hierarchical AFL algorithm with convergence analysis: + Optimizing the number of clusters with resource constraints - Fast adaptation ability to support 3D in different users is not studied
Chai et al. [37]	●	○	●	A tier-based mechanism combing AFL with SFL: + Training intro-tier models synchronously and cross-tier models asynchronously - Data heterogeneity issue in 3D is not tackled
This paper (AFM3D)	●	●	●	A general, integrated, and optimized AFM framework for 3D: + Devising deploy-friendly FL mechanism to tackle data island issue easily + Utilizing meta-learning to tackle data heterogeneity issue effectively + Supporting AFL to tackle straggler issue efficiently

the straggler problem by performing local training and global aggregation at an arbitrary time [38], [39]. As the first AFL algorithm, FedAsync [40] can update the global model immediately with a mixture aggregation parameter once a local model is received by the server. As the successor, ASO-Fed [41] inherits such an asynchronous manner and introduces online learning into AFL, i.e., the local data of AFL clients is accumulated gradually during the training process. However, asynchronous updates introduce the challenge of staleness, as the outdated model may add noise to the training procedure, deteriorate model performance or prevent convergence [42]. To resolve that, Ma et al. [34] propose a semi-asynchronous FL mechanism called FedSA to resist the staleness effect by aggregating a certain number of local models based on the communication budget. FLEET [35] relieves the staleness effect by dampening the impact of outdated results. The dampening factor is determined by time staleness and the local data novelty. Liu et al. [11] aggregate deep and shallow layers of Deep Neural Networks (DNNs) with different frequencies and assign aggregation weights adaptively according to the freshness of model parameters. However, since AFL is still in its infancy, its integration with meta-learning for 3D is still missing.

Moreover, the hierarchical structure can be implemented by clustering, which is to manage heterogeneous clients into different clusters according to their computing or communication capabilities. Such that, within a cluster, the time cost among clients is similar and the unnecessary long waiting can be avoided. E.g., a mechanism called FedCH is proposed to optimize the number of clusters considering resource budgets and then form an efficient cluster topology [36]; and a system named FedAT is present to organize clients into tiers based on their response latencies. Accordingly, it can execute intra-tier training synchronously and cross-tier training asynchronously [37]. However, these cluster-based methods can easily introduce biases and high complexity into training.

E. Summary

As summarized in Table II, current research can address one or two of the three emerging challenges faced by 3D. However, an integrated mechanism that can jointly solve three issues in 3D is still missing. To fill this gap, we propose a framework named AFM3D that can simultaneously bridge data islands with FL, tackle data heterogeneity with meta-learning, and relieve straggler issues with asynchronous mode.

III. AFM3D: AN AFM FRAMEWORK FOR 3D

As illustrated in Fig. 2, the proposed framework AFM3D mainly contains four consecutive steps, i.e., 1) 3D task release: target vehicles release the 3D task according to their needs; 2) local meta-training: source vehicles (i.e., clients) respond to the tasks voluntarily and execute local meta-training based on the training configuration and their own data, and then upload the updated local model parameters to the server; 3) global aggregation: the server receives local models from clients constantly and executes the global aggregation regularly; and 4) model adaptation: after the global model is learned, it is deployed to target vehicles for their own 3D tasks through fast adaptation of the global meta-model. In the following

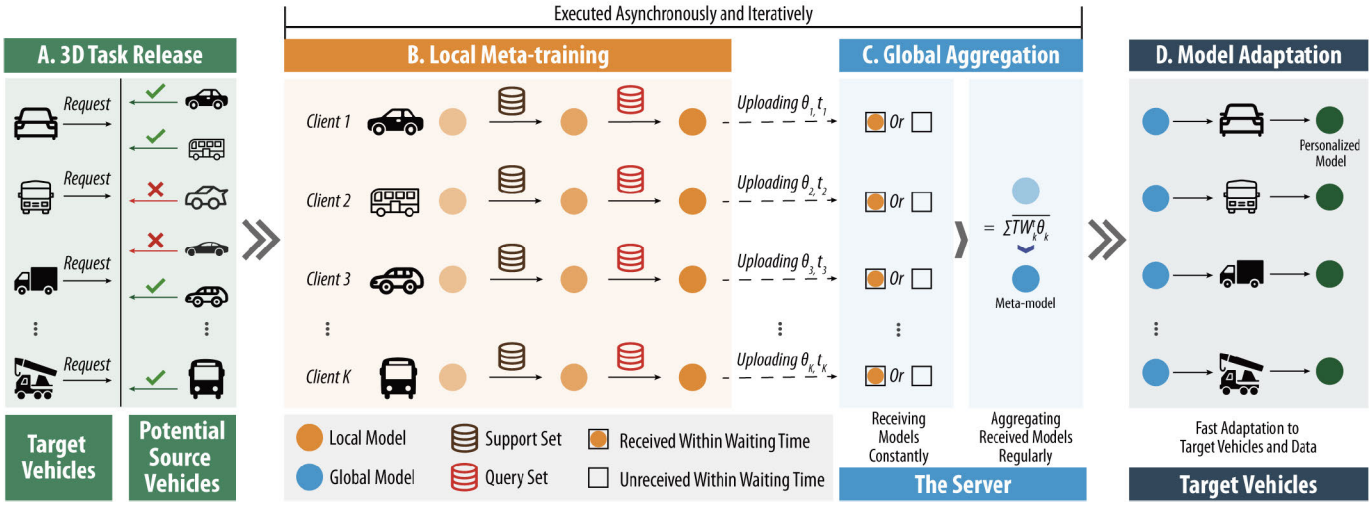


Fig. 2. The overview of AFM3D. The local meta-training at the client side and the global aggregation at the server side are executed asynchronously and iteratively to train a global model collaboratively once a 3D task is released by several target vehicles. After the above AFM3D training, the learned meta-model can adapt to target vehicles quickly by their own driver data with high performance.

TABLE III
LIST OF KEY NOTATIONS USED IN AFM3D

Notation	Meaning
K	The total number of clients
S_k^{spt}	The support set of client k
S_k^{qry}	The query set of client k
$\mathcal{L}_k$	The loss function of client k
θ_k	The local model of client k
$\hat{\theta}_k$	The intermediate model of client k
α	The inner learning rate
β	The outer learning rate
θ^{t+1}	The global model in the $t+1^{th}$ round
η_k	The aggregation weight of client k
K_t	The number of local models aggregated in the t^{th} round
t_k	The round when the local model of client k is created
TW_k^t	The normalized aggregation weight of client k in the t^{th} round
ϕ	The trained meta-model
ϕ_j	The personalized model of target vehicle j
T_j	The local data of target vehicle j for personalization

subsections, first, the research problem is defined, and then, the four steps are introduced. Note that for readability, the notations used in AFM3D are summarized in Table III.

A. Problem Definition

In view of regulations about data security and user privacy, assume that the sharing and exchanging of sensitive driver data are strictly forbidden, and as shown in Fig. 2, an asynchronous federated meta-learning framework is considered with K vehicles (which are clients in the training of model) and one server to learn a global model satisfying the demands of target vehicles. Specifically, each client has a private dataset S_k with labels, i.e., $S_k = (X_k, Y_k)$. Moreover, S_k is divided into support set S_k^{spt} and query set S_k^{qry} for meta-learning. Assume that each client has the abilities of model training and model uploading, however, the running states may be various due to heterogeneous communication and computation resources.

Based on the above settings, the goals of this framework are 1) to train an initialized model by managing heterogeneous clients asynchronously with low training cost (i.e., less learning time), and then, 2) to adopt the meta-model in individual 3D task rapidly with high model performance (i.e., high accuracy or low error). Accordingly, the optimization problems are defined in Formula 1 and Formula 2, respectively.

$$\min_{C} \sum_{t=1}^{\widehat{T}} c_t \tag{1}$$

$$\min_{\theta} \sum_{k=1}^{K} \mathcal{L}_k(\theta, S_k^{qry}) \tag{2}$$

where C is the accumulated training cost; $\widehat{T}$ is the first round when the desired performance target is reached; c_t is the cost in the t^{th} round; θ stands for the trained model; and $\mathcal{L}_k$ is the loss function of client k, e.g., cross-entropy.

In summary, this paper intends to resolve the two optimization problems jointly by proposing a collaborative and asynchronous FM framework for 3D, which includes 3D task release, local meta-training, global aggregation, and model adaptation as shown in Fig. 2.

B. 3D Task Release

Various types of vehicles have urgent demands of 3D, e.g., buses or long-distance trucks, whose safeties are extremely important and driver distractions are greatly dangerous. However, they may not have sufficient and high-quality data to train a high-performance model by themselves due to various reasons, e.g., vehicles newly put into use possess limited data. Thus, they need the data and knowledge from other vehicles/drivers to help them train a fine model.

These target vehicles can release a 3D task via the proposed framework to meet their needs. Then, the server will respond

to the request and orchestrate multiple clients as participants to train a model under the framework. Note that the server will transmit necessary information to clients before the training according to the needs of target vehicles, e.g., model architecture and hyper-parameters. Specifically, as a general framework, AFM3D supports various deep learning models, such as DenseNet, ResNet, MobileNet, and MnasNet series.

C. Local Meta-Training

First, once a source vehicle joins AFM3D as a client k, it will receive the latest global model θ^{t_k} from the server as its current local model θ_k. Second, an intermediate model $\hat{\theta}_k$ is computed based on S_k^{spt} according to Formula 3,

$$\hat{\theta}_k = \theta_k - \alpha \nabla_{\theta_k} \mathcal{L}_k(\theta_k, S_k^{spt}) \tag{3}$$

where α is the inner learning rate.

As the goal is to find the model that has fast adaptability to new data, the updated intermediate model is further evaluated on the query set with the loss $\mathcal{L}(\hat{\theta}_k, S_k^{qry})$ and update the model in the direction that can fast adapt to query set with Formula 4,

$$\theta_k = \theta_k - \beta \nabla_{\theta_k} \mathcal{L}(\hat{\theta}_k, S_k^{qry}) \tag{4}$$

where β is the outer learning rate. In this way, the trained model θ_k is sensitive to local data changes and can be adopted to new tasks quickly. Note that there is a derivative operation in the expansion of $\hat{\theta}_k$, thus Formula 4 requires to compute the second-order derivation. Specifically, the expanded form can be expressed as Formula 5,

$$\begin{aligned} \nabla_{\theta_k} \mathcal{L}(\hat{\theta}_k, S_k^{qry}) = \nabla_{\theta_{k,\tau}} \mathcal{L}(\theta_{k,\tau}, S_{k,\tau}^{qry}) \\ \cdot \prod_{i=0}^{\tau} (I - \alpha \nabla_{\theta_{k,i-1}} (\nabla_{\theta_k} \mathcal{L}(\theta_{k,i-1}))) \end{aligned} \tag{5}$$

where τ denotes the number of local gradient update steps. Since second-order derivation is complex, the First-Order Model-Agnostic Meta-Learning (FOMAML) [20] is adopted to get an approximate form as shown in Formula 6,

$$\nabla_{\theta_k} \mathcal{L}(\hat{\theta}_k, S_k^{qry}) \approx \nabla_{\theta_{k,\tau}} \mathcal{L}(\theta_{k,\tau}, S_{k,\tau}^{qry}) \tag{6}$$

Such that, Formula 4 can be rewritten into Formula 7.

$$\theta_k = \theta_k - \beta \nabla_{\hat{\theta}_k} \mathcal{L}(\hat{\theta}_k, S_k^{qry}) \tag{7}$$

Finally, after the local training is completed, client k uploads local model θ_k to the server immediately and then, waits for the new global model from the server for the next round of local meta-training.

D. Global Aggregation

In the designed asynchronous mode, the server receives updated local models from clients constantly and runs the aggregation process regularly, which is triggered by a predefined timer (e.g., waiting for 10s). Since the connectivity of vehicles may vary over time, the number of local models received by the server can be different in each round. Hence, the global aggregation function can be written as Formula 8,

$$\theta^{t+1} = \sum_{k=1}^{K_t} (\eta_k \times \theta_k) \tag{8}$$

where K_t is the number of local models received in the t^{th} round, η_k is the aggregation weight of client k, and θ^{t+1} is the updated global model. Note that in canonical federated meta-learning methods, $\eta_k = \frac{1}{K_t}$ indicates an average operation, i.e., each local model is considered to contribute equally to the global model.

Moreover, as illustrated in Fig. 1 (B), in asynchronous mode, some uploaded local models may be stale in the aggregation procedure, e.g., the local model of client 5 created in round 3 is to be aggregated in round 5. Intuitively, the latest model, e.g., the model of client 3 created and aggregated in round 5 deserves a higher aggregation weight than the stale local model mentioned above, as it provides the newest information. However, the average aggregation operation may become inefficient to address such a temporal heterogeneity in asynchronous mode, as it considers local models equally.

Hence, in this study, a temporal weight (TW) strategy is proposed as defined in Formula 9,

$$TW_k^t = \begin{cases} e^{-(t-t_k)}, & \exp \\ \dfrac{1}{t - t_k + 1}, & \text{inv} \\ \dfrac{1}{log(t - t_k + 1) + 1}, & \log \end{cases} \tag{9}$$

where TW_k^t is the TW of client k in the t^{th} round, e is the natural logarithm, and t_k stands for the round when the local model of client k is created. Note that the choice of the three functions, i.e., exponential, inverse, and logarithmic functions, is based on related research about AFL [40], which shows that these functions can add values to the global model by alleviating the impact of stale models. As for the performance comparison of the three TW functions, related results can be found in Section IV-E.

Then, TW is applied in the global aggregation, and accordingly, it transfers Formula 8 into Formula 10,

$$\theta^{t+1} = \sum_{k=1}^{K_t} (\overline{TW_k^t} \times \theta_k) \tag{10}$$

where $\overline{TW_k^t} = (TW_k^t / \sum_{k=1}^{K_t} TW_k^t)$ is the normalized weight. Such that, the more recent the local model is, the higher temporal weight it has in the aggregation. Intuitively, by computing aggregation weights TW_k^t adaptively, θ^{t+1} can alleviate the effect of stale models to improve the performance of the global model.

After the global model is updated, the server will distribute it to the clients to start the next round of local meta-training if the stop condition is not matched (e.g., the maximum learning time, or the target accuracy); otherwise, the local adaption of the global model is ignited by target vehicles.

E. Model Adaptation

Once the global meta-model ϕ is trained based on the above three steps, the target vehicle j downloads ϕ from the server as the initialized model, and adapts it for the context defined by the local data T_j according to Formula 11,

$$\phi_j = \phi_j - \alpha \nabla_{\phi_j} \mathcal{L}_j(\phi_j, T_j) \tag{11}$$

where ϕ_j is the personalized model of target vehicle j. In general, the meta-model is sensitive to new data T_j. Even if the data size of T_j is small, a few updating steps (even one step) of Formula 11 can improve the performance of ϕ_j significantly. Such that, the target vehicle can better support its 3D task with the adopted local model.

F. Algorithm of AFM3D

There are two types of important models in AFM3D, namely 1) the meta-model, which is trained by collaborating with source vehicles; and 2) the personalized model, which is adopted by the target vehicle. Hence, the algorithms to train the two models are introduced, respectively.

Algorithm 1 AFM3D for Meta-Model

PART 1: Executed in each source client

for $k \leq K$ in parallel **do**
  Receiving the global model θ_{t_k} as θ_k
  Training θ_k according to Formulas 3 and 7
  Uploading θ_k and t_k to the server
end for

PART 2: Executed in the AFM3D server

Thread 1: Receiving θ_k and t_k from clients consistently
Thread 2: Aggregating local models regularly
while the waiting time is reached **do**
  $TW^t = 0$
  for $k \leq K_t$ **do**
    Calculating TW_k^t according to Formula 9
    $TW^t += TW_k^t$
  end for
  $\overline{TW_k^t} \leftarrow TW_k^t / TW^t$
  Aggregating according to Formula 10
  Transmitting θ_{t+1} to clients
end while

Algorithm 2 AFM3D for Personalized Model

PART 1: Executed in the AFM3D server

Waiting for the download request from target vehicles
Distributing the meta-model ϕ to target vehicles

PART 2: Executed in each target vehicle

Sending the request for the meta-model to the server
Downloading the meta-model ϕ from the server
Initializing the personalized model $\phi_j = \phi$
Updating the model ϕ_j according to Formula 11
Applying the model ϕ_j for the 3D task

1) Algorithm of AFM3D for Meta-Model: As shown in Algorithm 1, the algorithm for meta-model consists of two parts, namely:

- *Part 1: In Each Client.* First, AFM3D client k receives the global model θ_{t_k} from the server in the t_k round as its local model θ_k. Second, θ_k is updated according to Formulas 3 and 7. Finally, after the local meta-training, θ_k and t_k are uploaded to the server for global aggregation.
- *Part 2: In the Server.* First, the AFM3D server receives local models from clients continuously. Second, once the predefined waiting time is reached, the server will start the global aggregation, in which, normalized aggregation weights TW_k^t will be first calculated and applied in the aggregation function to update the global model. Finally, the updated global model θ_{t+1} is transmitted to clients and the next global round begins.

2) Algorithm of AFM3D for Personalized Model: As shown in Algorithm 2, once the meta-model is trained, the server will inform each target vehicle and wait for their requests. Then, the meta-model is transmitted to target vehicles by the server. After the target vehicle j requests and receives the meta-model ϕ, the personalized model ϕ_j is adapted with its own data T_j according to Formula 11.

G. Convergence Analysis of AFM3D

Compared to traditional methods, the meta-learning and asynchronous mode in AFM3D may affect its convergence. The recent research [34], [43] has proved the convergence of AFL even in heterogeneous environments, and has shown that AFL can run faster than SFL. Such that, the convergence of AFM (asynchronous federated meta-learning) is further analyzed by giving three assumptions on loss function $\mathcal{L}_k$.

Assumption 1: $\mathcal{L}_k$ is bounded below and its gradient is also bounded by B_k, i.e., $min_{\theta_k}\mathcal{L}_k(\theta) > -\infty$, $||\nabla\mathcal{L}_k(\theta)|| \leq B_k$.

Assumption 2: $\mathcal{L}_k$ is L_k-*smooth* and μ_k-*Lipschitz continuous Hessian*, i.e., for $\forall\theta_1, \theta_2$, $||\nabla\mathcal{L}_k(\theta_2) - \nabla\mathcal{L}_k(\theta_1)|| \leq L_k||\theta_2 - \theta_1||$, $||\nabla^2\mathcal{L}_k(\theta_2) - \nabla^2\mathcal{L}_k(\theta_1)|| \leq \mu_k||\theta_2 - \theta_1||$.

Assumption 3: The stochastic gradient $\nabla\mathcal{L}_k(x, y; \theta)$ and Hessian $\nabla^2\mathcal{L}_k(x, y; \theta)$ calculated with a data sample (x, y) have bounded variance, i.e., $||\nabla\mathcal{L}_k(x, y; \theta) - \nabla\mathcal{L}_k(\theta)|| \leq \sigma_g^2$, $||\nabla^2\mathcal{L}_k(x, y; \theta) - \nabla^2\mathcal{L}_k(\theta)|| \leq \sigma_h^2$.

To ease the expression, $B_{max} = max_k\{B_k\}$, $L_{max} = max_k\{L_k\}$, $\mu_{max} = max_k\{\mu_k\}$,and $L' = 4L_{max} + \alpha\mu_{max}B_{max}$ are used. Furthermore, according to the convergence analysis in [24], Theorem 1 can be given as:

Theorem 1: After Algorithm 1 runs for R rounds with local epoch τ, if $\alpha \in (0, \frac{1}{L_{max}}]$ and $\beta \leq \frac{1}{10\tau L'}$ then we have

$$\frac{1}{\tau R}\sum_{t=0}^{R-1}\sum_{\epsilon=0}^{\tau-1} E\left[\left\|\nabla\mathcal{L}\left(\bar{\theta}_\epsilon^{t+1}\right)\right\|^2\right] \leq \frac{4\left(\mathcal{L}\left(\theta_0\right) - \mathcal{L}^*\right)}{\beta\tau R} + \mathcal{O}(1)(\beta L'\left(1 + \beta L'\tau(\tau-1)\right)\sigma_F^2 + \beta L'\gamma_F^2(\frac{1-p}{p(K-1)} + \beta L'\tau(\tau-1)) + \frac{\alpha^2 L_{max}^2\sigma_g^2}{D}) \quad (12)$$

where $\bar{\theta}_\epsilon^{t+1} = \sum_{k=1}^{K_t}(\overline{TW_k^t} \times \theta_{k,\epsilon}^{t+1})$ denotes the weighted model; $\mathcal{L}^*$ represents the optimal function; $p = \frac{\sum_{t=1}^{R} K_t}{K \times R}$ is the average participant fraction; σ_F and γ_F are parameters defined in Lemma of [24], which are related to σ_h, B_{max} or L_{max}; and D represents the batch size.

Fig. 3. The image examples of the SFD dataset, where c0, c1, c2, c3, c4, c5, c6, c7, c8, and c9 represent safe driving, texting-right, talking on the phone-right, texting-left, talking on the phone-left, operating the radio, drinking, reaching behind, hair and makeup, and talking to passenger, respectively.

Therefore, theoretically, AFM3D can make the model converge, which is also demonstrated by the evaluation results presented in the next section.

In summary, AFM3D implements unblocked local meta-training and global aggregation as well as model adaptation processes to train a personalized model that can support the 3D task effectively and efficiently. Moreover, a temporally weighted aggregation method is proposed to tackle the temporal heterogeneity of local models caused by the asynchronous mode. Through AFM3D, the overall learning performance can be improved significantly as revealed in Section IV.

IV. Evaluation

In this section, the performance of AFM3D is evaluated and discussed. First, experiment setups are introduced. Second, the performances of AFL, meta-learning, and AFM3D to train different models are analyzed and compared to demonstrate the supremacy and universality of AFM3D in supporting the local 3D task. Third, a comprehensive test on AFM3D performance influencers is conducted. Finally, the discussion is presented to provide some insights from the results.

A. Setups

1) Dataset: AFM3D is evaluated based on two widely used datasets, i.e., the State-Farm-Distracted-driver-detection dataset (SFD[1]) and American University in Cairo distracted driver detection dataset (AUC [44]). SFD contains 22,424 images (640 × 480 pixels) of 26 drivers with 10 classes as shown in Fig. 3. Similar to SFD, AUC also consists of 10 categories with 12,977 training samples and 4,331 testing samples (1080 × 1920 for each sample). In the preprocessing step, each image is randomly cropped with the size of 224 × 224 according to [2]. SFD is split by driver ID, accordingly 18 drivers are randomly selected as training clients, and the rest 8 drivers are used as testing clients. Since the driver ID information is not provided in AUC, 15 training clients and 6 testing clients are simulated with heterogeneous local data based on Dirichlet distribution (the Dirichlet parameter is 0.5 set according to [45]). For each training client, local data are divided into support and query sets equally; and for each testing client, half local data are randomly chosen for personalization and the other half are utilized for testing. Note that the data of testing clients are untouched during the training and only used for model adaptation and evaluation.

As for the communication capability, the transmission time of each client ranges from 3s to 20s randomly, and the predefined waiting time for the AFL server to start the global aggregation is 5s as default (except the initial round takes 10s to have more local models aggregated), while the participation rate of source clients in synchronous mode is 0.2.

2) Models: The following six representative models are trained during the evaluation, including:

- **DenseNet121:** A dense convolutional network that adopts shorter connections between layers [46].
- **ResNet18:** A residual learning framework that can support deeper networks efficiently [47].
- **DeiT:** A data-efficient image transformer model that achieves high-performing based on attention [48].
- **ShuffleNetV2:** A model that can achieve the tradeoff between speed and accuracy [49].
- **MobileNetV2:** A architecture that is suitable for mobile and resource-constrained environments [50].
- **MnasNet:** A mobile model that achieves a good trade-off between accuracy and latency [51].

Note that the model training settings are summarized in Table IV. As for the inference time and memory requirement of each model, they are recorded in Table V. The framework is developed based on Pytorch libraries by using Python language, and we execute the simulation on a server equipped with 4 NVIDIA GeForce RTX 3090 GPUs and installed with Windows 10 Pro and CUDA v11.4. Note that the source code of the framework can be downloaded from the link.[2]

[1]https://www.kaggle.com/competitions/state-farm-distracted-driver-detection/data

[2]https://github.com/IntelligentSystemsLab/AFM3D

TABLE IV
THE SUMMARY OF TRAINING SETTINGS

Dataset	Model	Inner lr*	Outer lr	Batch
SFD	DenseNet121	0.0001	0.0001	40
	ResNet18	0.0001	0.0001	40
	DeiT	0.003	0.003	40
	ShuffleNetV2	0.003	0.003	40
	MobileNetV2	0.001	0.001	40
	MnasNet	0.003	0.003	40
AUC	DenseNet121	0.001	0.001	40
	ResNet18	0.001	0.001	40
	DeiT	0.01	0.01	40
	ShuffleNetV2	0.01	0.01	40
	MobileNetV2	0.003	0.003	40
	MnasNet	0.003	0.003	40

* lr represents the learning rate.

TABLE V
THE SUMMARY OF MODEL INFERENCE TIME AND MEMORY REQUIREMENT

Model	IT*↓(s)	MM*↓(MB)	TM*↓(MB)
DenseNet121 [46]	0.0160	603.91	476.13
ResNet18 [47]	**0.0020**†	**216.68**	207.56
DeiT [48]	0.0080	472.09	470.93
ShuffleNetV2 [49]	0.0070	221.40‡	**93.92**
MobileNetV2 [50]	0.0080	389.51	275.63
MnasNet [51]	0.0060‡	313.06	185.61‡

* IT, MM, and TM represent Inference Time, Maximum Memory, and Total Memory, respectively.
† Bold numbers are the best value.
‡ Numbers with underlines are the second best values.

3) Comparative Methods: The following seven methods are compared, namely:

- **FedAvg:** The most recognized synchronous federated learning method [7].
- **TWF:** The asynchronous federated learning method with temporal differences among local models addressed [9].
- **MOON:** The effective federated learning method with optimized local training based on the similarity between model representations [45].
- **FedDecorr:** The federated learning method with dimensional collapse issue addressed [52] (Since FedDecorr cannot be used alone, it is adopted with TWF).
- **SFMeta:** The synchronous federated meta-learning method with average aggregation [53].
- **AFM3D-Aver:** The proposed AFM3D framework with average aggregation.
- **AFM3D-TW:** The proposed AFM3D framework with temporally weighted aggregation strategy.

By comparing FL methods (i.e., FedAvg, TWF, MOON, and FedDecorr) with FM methods (i.e., SFMeta, AFM3D-Aver, and AFM3D-TW), the effects of integrating meta-learning with FL can be revealed. Moreover, by comparing two AFM3D methods with SFMeta, the performance of FM by using synchronous and asynchronous modes can be analyzed.

4) Metrics: Five metrics are utilized, namely:

- **Test accuracy:** The average test accuracy of testing clients after one step of localization is used.
- **Test loss:** Cross entropy loss is used, which is the most recognized loss for classification tasks.
- **Recall:** It measures the proportion of correctly identified instances.
- **F1 Score (F1):** It combines precision and recall to evaluate the overall performance.
- **Training speed:** The time when the target accuracy is reached. Note that the target accuracies in SFD for DenseNet121, ResNet18, DeiT, ShuffleNetV2, MobileNetV2, and MnasNet are 75%, 75%, 65%, 65%, 65%, and 65%, respectively. As for AUC, the target accuracies are 65%, 65%, 55%, 60%, 55%, and 60%, respectively.

Specifically, the model accuracy, loss, recall, and F1 are calculated according to Formula 13,

$$\begin{aligned} accuracy &= \frac{TP+TN}{TP+FP+FN+TN} \\ loss_{CE} &= -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{M} y_{ic} log(p_{ic}) \\ recall &= \frac{TP}{TP+FN} \\ F1 &= 2 \times \frac{P \times R}{P+R} \end{aligned} \tag{13}$$

where TP, TN, FP, and FN represent true positives, true negatives, false positives, and false negatives, respectively; N and M are the number of samples and categories, respectively; $y_{ic} = 1$ if sample i belongs to category c; otherwise, $y_{ic} = 0$; and p_{ic} is the predicted probability; P and R denote Precision and Recall, respectively.

B. Impact of AFL

By comparing the FL methods in synchronous mode (FedAvg and MOON) and asynchronous mode (TWF and FedDecorr), the impact of AFL on the 3D task can be analyzed. First, as summarized in Table VI and Table VII, TWF and FedDecorr can improve model performance in terms of test accuracy, loss, recall, and F1 against FedAvg and MOON in most cases. Specifically, for the 6 models trained in 2 datasets, an average boost of 3.98%/5.41%/6.41% in accuracy/recall/F1 and an average reduction of 3.77% in loss are observed when comparing the best asynchronous method with the best synchronous mode. Moreover, AFL methods are faster than SFL methods according to Fig. 4 and Fig. 5, since the accuracy curves and loss curves of TWF and FedDecorr are always superior to the ones of FedAvg and MOON.

The results indicate that asynchronous mode can indeed boost model performance as well as training speed compared with synchronous mode, hence is more suitable for the 3D task in the context of FL.

C. Impact of Meta-Learning

By comparing the two asynchronous methods, i.e., AFL (TWF and FedDecorr) and AFM (AFM3D-TW), the impact

TABLE VI
THE SUMMARY OF EVALUATION RESULTS IN SFD DATASET

Method	DenseNet121 [46]					ShuffleNetV2 [49]				
	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓
FedAvg [7]	67.72%	68.08%	64.99%	1.309	-	60.48%	60.24%	57.58%	1.170	-
TWF [9]	76.49%	75.63%	73.34%	0.9093	490	66.19%	66.08%	63.74%	1.014	345
MOON [45]	71.67%	71.13%	67.59%	1.1561	-	62.80%	60.46%	55.94%	1.024	-
FedDecorr [52]	75.58%	74.82%	74.18%	0.9016	480	67.72%	66.19%	64.89%	0.9983	325
SFMeta [53]	76.12%	75.69%	74.38%	0.7974	482	70.87%	71.01%	66.83%	0.8702	393
AFM3D-Aver	79.02%†	78.83%	76.82%	0.7304	265	69.26%	70.92%	66.34%	0.9505	200
AFM3D-TW	**79.27%***	**79.02%**	**76.97%**	**0.7014**	**215**	**76.10%**	**77.15%**	**73.38%**	**0.7711**	**150**

Method	ResNet18 [47]					MobileNetV2 [50]				
	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓
FedAvg	66.33%	64.79%	62.84%	1.053	-	59.83%	60.27%	58.13%	1.185	-
TWF	72.55%	72.28%	69.07%	0.8935	-	60.94%	60.43%	57.28%	1.237	-
MOON	68.97%	67.71%	66.42%	0.8610	-	60.22%	57.66%	53.92%	1.054	-
FedDecorr	75.22%	73.01%	70.52%	0.8953	495	65.20%	60.94%	59.36%	1.100	360
SFMeta	75.22%	74.31%	71.33%	0.7863	504	68.11%	68.87%	65.37%	0.9450	430
AFM3D-Aver	75.25%	75.84%	72.81%	0.8111	490	72.76%	73.21%	69.71%	0.9220	200
AFM3D-TW	**77.26%**	**77.14%**	**74.82%**	**0.7388**	**250**	**73.74%**	**73.78%**	**70.52%**	**0.7997**	**175**

Method	DeiT [48]					MnasNet [51]				
	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓
FedAvg	64.43%	64.34%	61.83%	1.091	-	62.11%	61.05%	57.80%	1.104	-
TWF	67.20%	67.17%	65.28%	1.005	325	68.10%	68.51%	64.21%	0.975	255
MOON	66.79%	64.35%	63.87%	1.036	438	63.68%	54.64%	50.64%	1.002	-
FedDecorr	67.89%	67.18%	66.86%	1.021	295	68.69%	68.10%	66.23%	0.9661	230
SFMeta	69.43%	71.79%	70.55%	0.9741	333	69.06%	69.20%	66.09%	0.8927	375
AFM3D-Aver	69.69%	69.70%	67.57%	0.9611	165	70.42%	70.63%	66.27%	0.9152	215
AFM3D-TW	**74.18%**	**74.22%**	**72.27%**	**0.8475**	**140**	**73.78%**	**74.26%**	**70.41%**	**0.8163**	**145**

* Bold numbers are the best performance.
† Numbers with underlines are the second best values.

of meta-learning on the 3D task can be analyzed. First, as illustrated in Table VI and Table VII, AFM3D-TW can significantly promote test accuracy, recall, and F1 compared with the best one of TWF and FedDecorr, specifically, average improvements of 6.85%, 8.35%, and 7.38% on DenseNet121; 5.81%, 8.32%, and 8.68% on ResNet18; 9.11%, 8.68%, and 5.77% (DeiT); 11.76%, 10.96%, and 9.82% on ShuffleNetV2; 11.61%, 14.19%, and 15.57% on MobileNetV2); 7.49%, 7.56%, and 7.89% on MnasNet are observed, respectively.

Second, according to Table VI and Table VII, the maximum, minimum, and average reductions of test loss are 24.93%, 11.84%, and 16.65%, respectively, when comparing AFM3D-TW with the best of TWF and FedDecorr.

Finally, as shown in Table VI and Fig. 5, TWF can not reach the target accuracies for ResNet18 and MobileNetV2 in SFD, while AFM3D-TW can always reach the target accuracies successfully with an average transmission time of 179s for the 6 models, which is 50.91% faster than FedDecorr. According to the results of AUC listed in Table VII, the average boost in training speed is 53.78% when comparing AFM3D-TW with the best of TWF and FedDecorr. It shows that meta-learning integrated with AFL can indeed achieve fast adaptation and good performance for the 3D task.

D. Performance of AFM3D

Two variants of AFM3D, namely AFM3D-Aver and AFM3D-TW (with average and TW aggregation strategies applied separately), are evaluated. First, according to the results in Table VI and Table VII, AFM3D-TW can consistently maintain the highest accuracies, recalls, and F1s with average improvements of 7.49%, 7.29%, and 7.35% against the best performance among the compared methods. In the meanwhile, AFM3D-TW spends the shortest time to achieve the target accuracies with an average acceleration of 56.94% against the best baseline. Such improvements are also observed in Fig. 4, in which, the accuracy curves of AFM3D-TW can surpass other methods and reach the target accuracies more quickly.

In addition, as reflected in Fig. 5, the loss curves of AFM3D-TW are better than the ones of FedAvg, TWF, MOON, FedDecorr, SFMeta, and AFM3D-Aver throughout

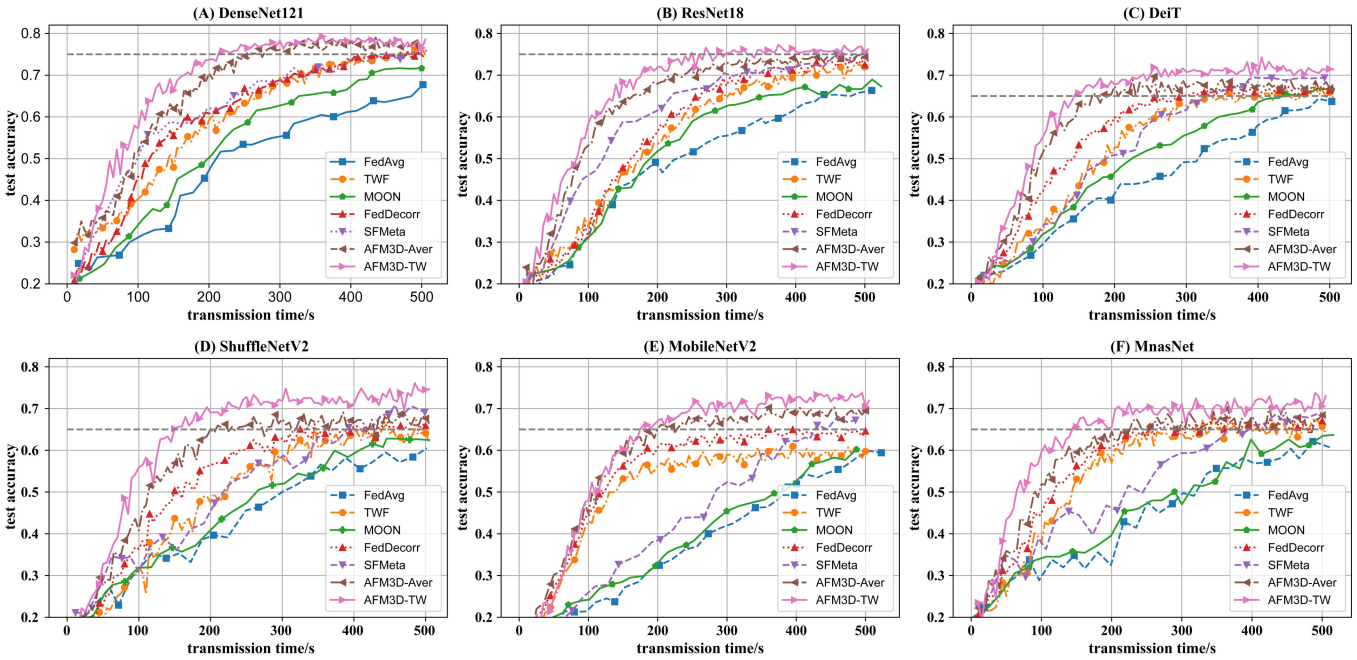


Fig. 4. The accuracy curve of the seven methods in SFD dataset for (A) DenseNet121, (B) ResNet18, (C) DeiT, (D) ShuffleNetV2, (E) MobileNetV2, and (F) MnsaNet.

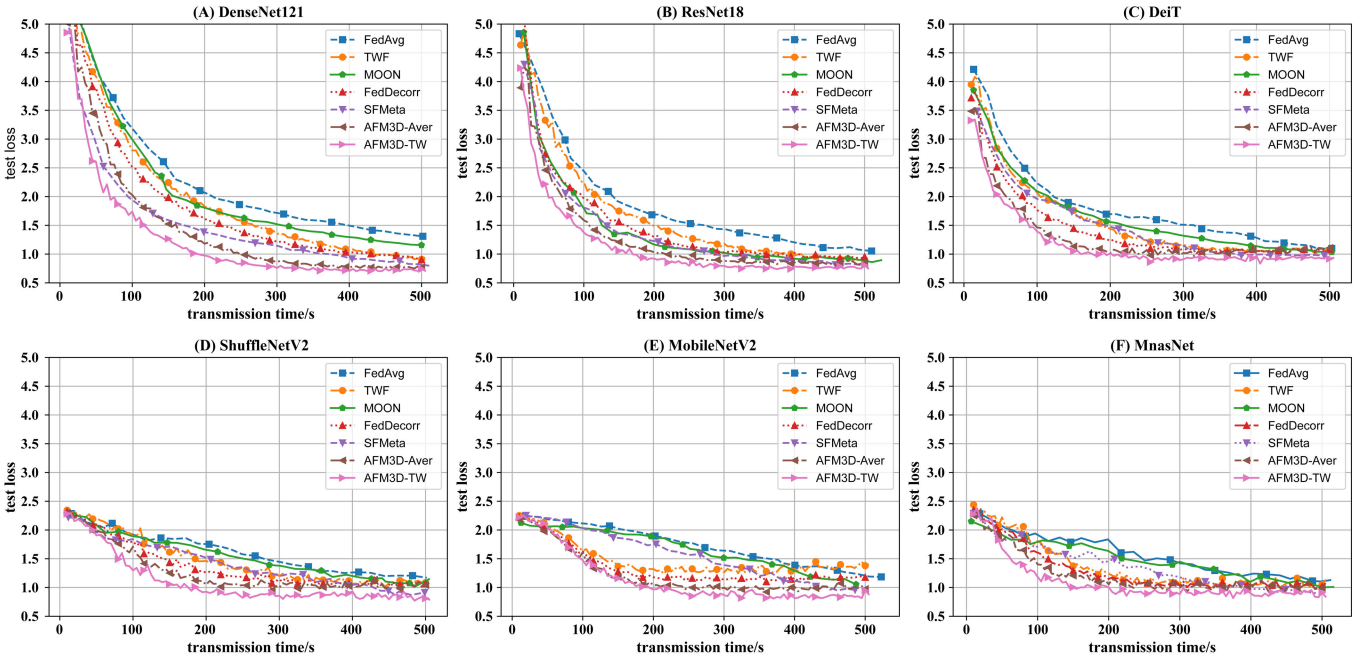


Fig. 5. The loss curve of the seven methods in SFD dataset for (A) DenseNet121, (B) ResNet18, (C) DeiT, (D) ShuffleNetV2, (E) MobileNetV2, and (F) MnsaNet.

the learning process. As a result, AFM3D-TW can achieve the lowest test loss with an average reduction of 14.80% against the best baseline according to Table VI and Table VII.

Finally, as for AFM3D-Aver, its performance is unstable and even can not surpass SFMeta in terms of test accuracy and loss when training ShuffleNetV2. It, in turn, illustrates the necessity of using TW in asynchronous mode.

E. Test of Performance Influencers in AFM3D

To further validate the effectiveness and efficiency of AFM3D, the following five experiments are executed to test the performance influencers of AFM3D (including meta-learning methods, temporal weight functions, asynchronous aggregation strategies, way-shot configurations, and number of local adaptation steps), by taking the case using SFD to train ResNet18 and DenseNet121 as an example.

First, two meta-learning methods, namely FOMAML and Reptile, are compared. As shown in Table VIII, they can both achieve good performance. However, under the same configuration, FOMAML can surpass Reptile, showing its strong robustness in handling 3D data. Specifically, for ResNet18 and DenseNet121, the average accuracy improvements are 5.41%

TABLE VII
THE SUMMARY OF EVALUATION RESULTS IN AUC DATASET

Method	DenseNet121 [46]					ShuffleNetV2 [49]				
	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓
FedAvg [7]	65.62%	55.04%	52.00%	1.2947	3249	55.84%	51.98%	52.91%	1.5545	-
TWF [9]	66.09%	55.50%	54.58%	1.2260	2175	61.48%	59.86%	59.92%	1.5031	1440
MOON [45]	66.10%	56.53%	52.11%	1.1896	2872	61.67%	60.04%	60.30%	1.5561	1739
FedDecorr [52]	67.12%	60.28%	56.87%	1.1432	1920	61.59%	59.93%	60.02%	1.4639	1375
SFMeta [53]	66.77%	55.64%	54.70%	1.1185	3030	62.62%	61.76%	60.67%	1.4499	1332
AFM3D-Aver	70.61%†	63.80%	60.26%	1.1740	1900	62.06%	60.92%	60.76%	1.4080	665
AFM3D-TW	**73.88%***	**67.65%**	**63.13%**	**1.0281**	**510**	**68.46%**	**63.14%**	**63.95%**	**1.3811**	**485**
Method	**ResNet18** [47]					**MobileNetV2** [50]				
	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓
FedAvg	65.63%	53.11%	50.47%	1.7132	1682	57.47%	54.84%	50.86%	1.4443	3214
TWF	65.89%	59.80%	57.73%	1.5321	1240	59.17%	55.17%	52.46%	1.4484	1470
MOON	67.77%	59.15%	57.28%	1.1526	1348	58.15%	53.51%	53.93%	1.4330	2267
FedDecorr	66.34%	60.07%	57.93%	1.3246	1150	59.22%	56.56%	53.85%	1.4172	1140
SFMeta	66.19%	61.00%	60.70%	1.2201	1278	58.45%	54.71%	53.83%	1.5324	1897
AFM3D-Aver	71.40%	65.79%	63.64%	1.3094	930	64.85%	59.45%	59.76%	1.3431	725
AFM3D-TW	**72.25%**	**66.67%**	**64.45%**	**1.0545**	**680**	**65.21%**	**60.69%**	**60.50%**	**1.2383**	**570**
Method	**DeiT** [48]					**MnasNet** [51]				
	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓	Accuracy↑	Recall↑	F1↑	Loss↓	Time↓
FedAvg	57.57%	52.22%	54.28%	1.5384	3342	56.84%	54.12%	52.81%	1.5423	-
TWF	58.01%	56.67%	55.10%	1.5349	1685	60.55%	55.32%	52.73%	1.5384	1685
MOON	59.04%	55.76%	56.23%	1.4752	2157	59.15%	52.51%	53.14%	1.5693	-
FedDecorr	60.10%	59.33%	59.47%	1.4563	1480	60.71%	57.35%	56.01%	1.5072	1230
SFMeta	59.41%	58.92%	58.12%	1.5467	1574	61.55%	56.82%	55.91%	1.4501	2405
AFM3D-Aver	65.16%	61.11%	61.31%	1.4177	940	64.87%	60.19%	60.67%	1.3943	780
AFM3D-TW	**65.48%**	**63.41%**	**61.52%**	**1.3398**	**870**	**65.31%**	**61.21%**	**61.31%**	**1.3499**	**585**

* Bold numbers are the best performance.
† Numbers with underlines are the second best values.

and 5.23%, respectively. Such that, FOMAML is used as the default meta-learning method in AFM3D.

Second, four different weight functions, namely none/aver (i.e., AFM3D-Aver as described in Formula 8 with $\eta_k = \frac{1}{K_t}$), exp, inv, and log (i.e., three AFM3D-TW variants as described in Formula 9) are evaluated. According to Table VIII, significant accuracy differences between the averagely weighted aggregation function and the three temporally weighted aggregation functions can be observed. Specifically, under the same conditions, the best among exp, inv, and log functions can surpass the aver function by 2.42%. Moreover, the diversities among the three AFM3D-TW variants are not prominent since any one of them can be the best in the various configurations. Considering the exp function holds the most number of best values, it is chosen as the default function in AFM3D-TW.

Third, three asynchronous aggregation strategies, namely aggregated every 5s, 10s, and 15s, are created to verify the robustness of the proposed method in different asynchronous settings. As summarized in Table VIII, all of them can achieve satisfactory performance. Specifically, for FOMAML-based ResNet18, the worst and the best values are 71.08% (10s) and 78.68% (5s), respectively. As for FOMAML-based DenseNet121, the accuracies vary from 77.04% (10s) to 79.55% (5s). Since the strategy of 5s can achieve a higher performance than the rest stratgeis, the default waiting time is set as 5s in the asynchronous mode.

Fourth, AFM3D is tested under four different way and shot configurations, i.e., 5-way 1-shot, 5-way 2-shot, 10-way 1-shot, and 10-way 2-shot [25]. As summarized in Table IX, AFM3D can maintain a steady performance with low variance, which validates the robustness of the proposed framework in various few-shot scenarios. Specifically, with the increase of the number of ways and shots, the accuracies of AFM3D can be promoted slightly, which may attribute to the limited total label number and high model complexity.

Finally, the fast adaptation capacity of AFM3D is evaluated. As shown in Table X, for AFM3D-TW, the accuracy of the initialized model (0-step) can achieve 54.96% (improved by 10.83% compared with the best baseline FedDecorr). After one step of local adaptation, the accuracy can be lifted to 77.26%. When the personalization continues and after three steps, the accuracy of AFM3D can reach 84.46%

TABLE VIII
THE ACCURACY↑ RESULTS OF ABLATION STUDIES

Meta-learning Method	Time Weight Function	ResNet18			DenseNet121		
		5s*	10s*	15s*	5s	10s	15s
FOMAML	none (aver)	75.25%	70.49%	75.54%	79.02%	77.04%	78.68%
	exp	77.26%	71.37%	75.82%	79.27%	**78.57%**	**79.22%**
	inv	77.85%	**72.08%**	**76.13%**	79.07%	78.42%	78.98%
	log	**78.68%**	71.08%	75.77%	**79.55%**	78.09%	79.11%
Reptile	none (aver)	70.46%	67.44%	70.66%	73.51%	69.75%	73.82%
	exp	**72.31%**	**70.29%**	71.72%	**75.74%**	72.29%	**75.49%**
	inv	72.15%	70.03%	**72.24%**	75.02%	72.18%	75.24%
	log	**72.31%**	69.62%	71.80%	74.87%	**72.33%**	74.79%

* 5, 10, and 15s denote the asynchronous aggregation strategies, i.e., the waiting time for a round.

TABLE IX
THE ACCURACY↑ OF DIFFERENT WAY AND SHOT CONFIGURATIONS TO TRAIN RESNET18 ON SFD

AFM3D variant	5-way 1-shot	5-way 2-shot
Aver	59.52±1.66%	65.08±1.48%
TW	67.50±1.42%	72.51±1.33%
AFM3D variant	**10-way 1-shot**	**10-way 2-shot**
Aver	62.23±1.37%	64.78±1.26%
TW	71.75±1.25%	73.76±1.08%

TABLE X
THE ACCURACY↑ IN MODEL ADAPTATION TO TRAIN RESNET18 ON SFD

Method	0-step	1-step	2-step	3-step
FedAvg [7]	46.37%	66.33%	74.66%	78.50%
TWF [9]	49.33%	72.55%	78.91%	82.06%
MOON [45]	48.19%	68.97%	79.89%	82.05%
FedDecorr [52]	49.59%	75.22%	78.80%	81.44%
SFMeta [53]	48.47%	75.22%	80.04%	81.96%
AFM3D-Aver	50.04%†	75.25%	80.62%	82.22%
AFM3D-TW	**54.96%***	**77.26%**	**82.17%**	**84.46%**

* Bold numbers are the best performance.
† Numbers with underlines are the second best values.

with an improvement of 2.92% against the best baseline TWF. The results indicate that AFM3D can indeed train a meta-model with high generability and also can rapidly adapt it to new data or tasks for better-personalized performance.

F. Discussion

According to the above-mentioned holistic evaluation results, the following insights can be observed:

- **The asynchronous mode of FL is more suitable than its synchronous mode for 3D tasks.** Compared with the synchronous FL method (FedAvg and MOON), the asynchronous FL method (TWF and FedDecorr) can generally get a higher accuracy/recall/F1 and a lower loss within a shorter learning time. A similar result can also be observed between SFMeta and AFM3D-TW. It indicates that the asynchronous mode (implemented in AFM3D) is more efficient and effective than the synchronous mode in the context of 3D to address the straggler issue;
- **Meta-learning integrated with FL can improve the performance of 3D remarkably.** The distinct improvements between AFM3D-TW and TWF/FedDecorr show that meta-learning (utilized by AFM3D) can help target vehicles localize the global model quickly and boost the model performance significantly to address the data heterogeneity issue in the context of FL and 3D;
- **TW is necessary and beneficial for AFM3D.** Although AFM3D-TW behaves well, the model performance of AFM3D-Aver is variable and is inferior to SFMeta in some cases. Such observations show that in the asynchronous mode, the temporal heterogeneity needs to be addressed as the model deterioration may happen, and the proposed TW strategy in AFM3D is simple but efficient to tackle such an issue;
- **AFM3D-TW can boost the model training performance.** AFM3D-TW can outperform other methods in training 3D models with high performance in terms of learning accuracy, recall, F1, loss, and speed. Moreover, various models, i.e., DenseNet, ResNet, Transformer, ShuffleNet, MobileNet, and MnasNet series, experience performance elevation by using AFM3D-TW, which indicates the supremacy of the proposed framework.
- **The performance stability of AFM3D can be improved.** Although the performance of AFM3D is superior to other methods, a slight performance fluctuation

during the learning process under different task configurations can still be observed, which indicates that the robustness of AFM3D can be further optimized in the future.

Due to the limitation in collecting large amount of self-own data, the standard dataset is used to design distributed training scenarios that simulate the actual applications of compared methods for a fair comparison. In the future, we will develop an open source framework that can deploy and manage federated learning tasks on smart devices. It is worth noting that a prototype has been developed and can be downloaded via the link.[3] To conduct the experiment on real vehicles by using such a framework, a smart device (e.g., android smartphone or pad) can be mounted at each vehicle and connected to a central server. By recruiting volunteers, a collaborative and active user group can be created, based on which, we can run the experiment privately by using the local data and computation power of each smart device to train the meta-model and also deploy it for local usage. Accordingly, the performance of the proposed method and other compared methods in real applications can be better evaluated.

V. Conclusion

This paper proposes a novel asynchronous federated meta-learning framework called AFM3D to tackle the data island, data heterogeneity, and straggler challenges of 3D jointly in a collaborative and privacy-preserving way. Specifically, AFM3D 1) adopts FL to bridge private data islands (i.e., vehicles containing private data of drivers); 2) applies meta-learning based on FOMAML to train a meta-model that can adapt to target clients quickly; 3) employs the asynchronous mode to alleviate the impact of stragglers; and 4) designs a temporally weighted strategy to tackle the staleness issue in asynchronous mode.

As shown by the evaluation results based on the SFD and AUC datasets, AFM3D can improve test accuracy, recall, F1 score by about 7.61%, 7.44%, and 7.95%, respectively, reduce test loss by about 9.95%, and boost learning speed by about 50.91% against the best baseline (i.e., FedAvg, TWF, MOON, FedDecorr or SFMeta). Moreover, as a general framework, AFM3D can dramatically elevate the training performance of various AI models (i.e., DenseNet, ResNet, Transformer, ShuffleNet, MobileNet, and MnasNet series) for 3D.

In the future, first, to further extend the capability of AFM3D, the usage of AIGC models (such as GPT-4) will be investigated and integrated with the framework to reduce the iterative training burden of clients, especially in the context that the edge computing powers are restricted, and to further secure the learning process by using generated contents while gradient attracts exist to sniff the private data of drivers. Second, since the client-server connection of AFM3D is vulnerable to the dynamic states of cruising vehicles that may change over time and place, a layer-based paradigm will be investigated to make the runtime communication of learning participants more robust and stable. Third, the aggregation weights in asynchronous mode can be further optimized by utilizing heuristic methods, and an aggregation algorithm that can achieve the trade-off between model performance and computation complexity will be studied. Finally, AFM3D will be enhanced to support other similar human distraction detection problems, e.g., workers at work, etc.

[3]https://github.com/IntelligentSystemsLab/generic_and_open_learning_federator

References

[1] Y. F. Payalan and M. A. Guvensan, "Towards next-generation vehicles featuring the vehicle intelligence," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 1, pp. 30–47, Jan. 2020.

[2] C. Duan, Y. Gong, J. Liao, M. Zhang, and L. Cao, "FRNet: DCNN for real-time distracted driving detection toward embedded deployment," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 9, pp. 9835–9848, Oct. 2023.

[3] D. Lu, F. Guo, and F. Li, "Evaluating the causal effects of cellphone distraction on crash risk using propensity score methods," *Accident Anal. Prevention*, vol. 143, Aug. 2020, Art. no. 105579.

[4] J. Wang et al., "A survey on driver behavior analysis from in-vehicle cameras," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 10186–10209, Aug. 2022.

[5] R. Chhabra, S. Singh, and V. Khullar, "Privacy enabled driver behavior analysis in heterogeneous IoV using federated learning," *Eng. Appl. Artif. Intell.*, vol. 120, Apr. 2023, Art. no. 105881.

[6] M. Xu et al., "Unleashing the power of edge-cloud generative AI in mobile networks: A survey of AIGC services," 2023, *arXiv:2303.16129*.

[7] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[8] S. Warnat-Herresthal et al., "Swarm learning for decentralized and confidential clinical machine learning," *Nature*, vol. 594, no. 7862, pp. 265–270, Jun. 2021.

[9] L. You, S. Liu, Y. Chang, and C. Yuen, "A triple-step asynchronous federated learning mechanism for client activation, interaction optimization, and aggregation enhancement," *IEEE Internet Things J.*, vol. 9, no. 23, pp. 24199–24211, Dec. 2022.

[10] L. You, Z. Guo, B. Zuo, Y. Chang, and C. Yuen, "SLMFed: A stage-based and layer-wise mechanism for incremental federated learning to assist dynamic and ubiquitous IoT," *IEEE Internet Things J.*, early access, pp. 1–19, 2024, doi: 10.1109/JIOT.2024.3353793.

[11] S. Liu, Q. Chen, and L. You, "Fed2A: Federated learning mechanism in asynchronous and adaptive modes," *Electronics*, vol. 11, no. 9, p. 1393, Apr. 2022.

[12] Y. Wang, X. Cui, Z. Gao, and B. Gan, "Fed-SCNN: A federated shallow-CNN recognition framework for distracted driving," *Secur. Commun. Netw.*, vol. 2020, pp. 1–10, Nov. 2020.

[13] X. Zhou, W. Liang, J. She, Z. Yan, and K. I. Wang, "Two-layer federated learning with heterogeneous model aggregation for 6G supported Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 70, no. 6, pp. 5308–5317, Jun. 2021.

[14] H. Yang, H. Liu, Z. Hu, A.-T. Nguyen, T.-M. Guerra, and C. Lv, "Quantitative identification of driver distraction: A weakly supervised contrastive learning approach," *IEEE Trans. Intell. Transp. Syst.*, early access, Sep. 27, 2023, doi: 10.1109/TITS.2023.3316203.

[15] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4298–4311, Apr. 2020.

[16] X. Yuan et al., "FedSTN: Graph representation driven federated learning for edge computing enabled urban traffic flow prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 8, pp. 8738–8748, May 2022, doi: 10.1109/TITS.2022.3157056.

[17] L. Liu, Z. Xi, K. Zhu, R. Wang, and E. Hossain, "Mobile charging station placements in Internet of Electric vehicles: A federated learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 24561–24577, Dec. 2022.

[18] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5149–5169, Sep. 2022.

[19] S. Liu, H. Qu, Q. Chen, W. Jian, R. Liu, and L. You, "AFMeta: Asynchronous federated meta-learning with temporally weighted aggregation," in *Proc. IEEE Smartworld, Ubiquitous Intell. Comput., Scalable Comput. Commun., Digit. Twin, Privacy Comput., Metaverse, Auto. Trusted Vehicles (SmartWorld/UIC/ScalCom/DigitalTwin/PriComp/Meta)*, Dec. 2022, pp. 641–648.

[20] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.

[21] C. Finn, A. Rajeswaran, S. Kakade, and S. Levine, "Online meta-learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1920–1930.

[22] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," 2018, *arXiv:1803.02999*.

[23] Q. Chen, S. Liu, H. Qu, R. Zhu, and L. You, "TWAFR-GRU: An integrated model for real-time charging station occupancy prediction," in *Proc. IEEE Smartworld, Ubiquitous Intell. Comput., Scalable Comput. Commun., Digit. Twin, Privacy Comput., Metaverse, Auto. Trusted Vehicles (SmartWorld/UIC/ScalCom/DigitalTwin/PriComp/Meta)*, Dec. 2022, pp. 1611–1618.

[24] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 3557–3568.

[25] X. Li, Y. Li, J. Wang, C. Chen, L. Yang, and Z. Zheng, "Decentralized federated meta-learning framework for few-shot multitask learning," *Int. J. Intell. Syst.*, vol. 37, no. 11, pp. 8490–8522, Nov. 2022.

[26] H. Qu, S. Liu, Z. Guo, L. You, and J. Li, "Improving parking occupancy prediction in poor data conditions through customization and learning to learn," in *Proc. Int. Conf. Knowl. Sci., Eng. Manage.* Springer, 2022, pp. 159–172.

[27] H. Qu, S. Liu, J. Li, Y. Zhou, and R. Liu, "Adaptation and learning to learn (ALL): An integrated approach for small-sample parking occupancy prediction," *Mathematics*, vol. 10, no. 12, p. 2039, Jun. 2022.

[28] Z. Guo, L. You, S. Liu, J. He, and B. Zuo, "ICMFed: An incremental and cost-efficient mechanism of federated meta-learning for driver distraction detection," *Mathematics*, vol. 11, no. 8, p. 1867, Apr. 2023.

[29] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 2351–2363.

[30] Z. Zhu, J. Hong, and J. Zhou, "Data-free knowledge distillation for heterogeneous federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 12878–12889.

[31] L. Zhang, L. Shen, L. Ding, D. Tao, and L.-Y. Duan, "Fine-tuning global model via data-free knowledge distillation for non-IID federated learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10164–10173.

[32] K. Doshi and Y. Yilmaz, "Federated learning-based driver activity recognition for edge devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2022, pp. 3337–3345.

[33] E. Shang, H. Liu, Z. Yang, J. Du, and Y. Ge, "FedBiKD: Federated bidirectional knowledge distillation for distracted driving detection," *IEEE Internet Things J.*, vol. 10, no. 13, pp. 11643–11654, Oct. 2023.

[34] Q. Ma, Y. Xu, H. Xu, Z. Jiang, L. Huang, and H. Huang, "FedSA: A semi-asynchronous federated learning mechanism in heterogeneous edge computing," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3654–3672, Dec. 2021.

[35] G. Damaskinos, R. Guerraoui, A.-M. Kermarrec, V. Nitu, R. Patra, and F. Taiani, "FLeet: Online federated learning via staleness awareness and performance prediction," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 5, pp. 1–30, Oct. 2022.

[36] Z. Wang, H. Xu, J. Liu, Y. Xu, H. Huang, and Y. Zhao, "Accelerating federated learning with cluster construction and hierarchical aggregation," *IEEE Trans. Mobile Comput.*, vol. 22, no. 7, pp. 3805–3822, Jul. 2022.

[37] Z. Chai, Y. Chen, A. Anwar, L. Zhao, Y. Cheng, and H. Rangwala, "FedAT: A high-performance and communication-efficient federated learning system with asynchronous tiers," in *Proc. Int. Conf. for High Perform. Comput., Netw., Storage Anal.*, Nov. 2021, pp. 1–17.

[38] L. You, S. Liu, T. Wang, B. Zuo, Y. Chang, and C. Yuen, "AiFed: An adaptive and integrated mechanism for asynchronous federated data mining," *IEEE Trans. Knowl. Data Eng.*, early access, pp. 1–17, 2023, doi: 10.1109/TKDE.2023.3332770.

[39] L. You, S. Liu, B. Zuo, C. Yuen, D. Niyato, and H. V. Poor, "Federated and asynchronized learning for autonomous and intelligent things," *IEEE Netw.*, early access, Oct. 9, 2023, doi: 10.1109/MNET.2023.3321519.

[40] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," 2019, *arXiv:1903.03934*.

[41] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-IID data," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2020, pp. 15–24.

[42] C.-H. Hu, Z. Chen, and E. G. Larsson, "Device scheduling and update aggregation policies for asynchronous federated learning," in *Proc. IEEE 22nd Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Sep. 2021, pp. 281–285.

[43] A. Koloskova, S. U. Stich, and M. Jaggi, "Sharper convergence guarantees for asynchronous SGD for distributed and federated learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 17202–17215.

[44] Y. Abouelnaga, H. M. Eraqi, and M. N. Moustafa, "Real-time distracted driver posture classification," 2017, *arXiv:1706.09498*.

[45] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 10708–10717.

[46] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.

[47] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[48] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 10347–10357.

[49] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in *Computer Vision—ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham, Switzerland: Springer, 2018, pp. 122–138.

[50] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

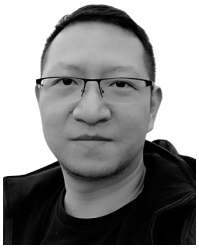
[51] M. Tan et al., "MnasNet: Platform-aware neural architecture search for mobile," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2815–2823.

[52] Y. Shi, J. Liang, W. Zhang, V. Y. F. Tan, and S. Bai, "Towards understanding and mitigating dimensional collapse in heterogeneous federated learning," 2022, *arXiv:2210.00226*.

[53] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He, "Federated meta-learning with fast convergence and efficient communication," 2018, *arXiv:1802.07876*.

Sheng Liu (Graduate Student Member, IEEE) received the B.Eng. degree from the School of Intelligent Systems Engineering, Sun Yat-sen University, China, in 2021, where he is currently pursuing the master's degree. His research interests include artificial intelligence (AI), machine learning, federated learning, data mining, and their applications in smart cities, the Internet-of-Things (IoT), and intelligent transportation systems (ITS).

Linlin You (Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Pavia in 2015. He was a Senior Post-Doctoral Researcher with the Singapore-MIT Alliance for Research and Technology and a Research Fellow with the Architecture and Sustainable Design Pillar, Singapore University of Technology and Design. He is currently an Associate Professor with the School of Intelligent Systems Engineering, Sun Yat-sen University, and also a Researcher affiliate with the Intelligent Transportation System Laboratory, Massachusetts Institute of Technology. He published more than 50 journal and conference papers in the research fields of smart cities, service orchestration, multi-source data fusion, machine learning, and federated learning.



Rui Zhu received the B.Sc. degree from Nanjing Normal University, China, the M.Sc. degree from the KTH Royal Institute of Technology, Sweden, and the Ph.D. degree from The Hong Kong Polytechnic University (PolyU), Hong Kong SAR, with an exchange study with Université Laval, Canada. He was a Research Assistant Professor with PolyU and a Post-Doctoral Associate with the MIT Senseable City Laboratory. He is currently a Scientist with the Institute of High Performance Computing, A*STAR, Singapore. He has published more than 50 journals and conference papers in research fields, such as urban mobility, urban environment, and solar cities. He has served as the PI/the Co-I for many academic projects and has been engaging with several academic journals as the guest editor, a reviewer, and an international organization.

Bing Liu received the Ph.D. degree in engineering from the Technical University of Munich in 2022. She is currently an HMI Engineer with BYD Auto Company Ltd., and also a Post-Doctoral Researcher with the University of Science and Technology of China. Her research interests include mixed reality, visualization and spatial learning, user experience, and navigation.

Rui Liu received the B.Eng. degree from the Harbin Institute of Technology (HIT), Harbin, China, in 2014, and the Ph.D. degree from the Singapore University of Technology and Design (SUTD), Singapore, in 2019. Currently, she is a Research Fellow with the School of Computer Science and Engineering (SCSE), Nanyang Technological University (NTU), Singapore. Her research interests include graph neural networks, federated learning, and brain–computer interfaces.

Han Yu (Senior Member, IEEE) received the Ph.D. degree from the School of Computer Science and Engineering (SCSE), Nanyang Technological University (NTU), Singapore. He held the prestigious Lee Kuan Yew Post-Doctoral Fellowship (LKY PDF) from 2015 to 2018. He is currently a Nanyang Assistant Professor (NAP) with SCSE, NTU. He has published more than 200 research papers and book chapters in leading international conferences and journals. He is the coauthor of the book *Federated Learning*—the first monograph on the topic of federated learning. His research interests include federated learning and algorithmic fairness. He is a Senior Member of *CCF*. His research works have won multiple awards from conferences and journals.

Chau Yuen (Fellow, IEEE) received the B.Eng. and Ph.D. degrees from Nanyang Technological University, Singapore, in 2000 and 2004, respectively.

He was a Post-Doctoral Fellow with Lucent Technologies Bell Labs, Murray Hill, in 2005. From 2006 to 2010, he was with the Institute for Infocomm Research, Singapore. He was a Visiting Assistant Professor with The Hong Kong Polytechnic University in 2008. From 2010 to 2023, he was an Associate Professor with the Engineering Product Development Pillar, Singapore University of Technology and Design. Since 2023, he has been with the School of Electrical and Electronics Engineering, Nanyang Technological University. He has three U.S. patents and published more than 500 research papers in international journals or conferences.

Dr. Yuen was a recipient of the Lee Kuan Yew Gold Medal, the Institution of Electrical Engineers Book Prize, the Institute of Engineering of Singapore Gold Medal, the Merck Sharp and Dohme Gold Medal, and twice a recipient of the Hewlett Packard Prize. He received the IEEE Asia–Pacific Outstanding Young Researcher Award in 2012 and the IEEE VTS Singapore Chapter Outstanding Service Award in 2019. He serves as an Editor for IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE SYSTEMS JOURNAL, and IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, where he was awarded as the IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING Excellent Editor Award and the Top Associate Editor for IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY from 2009 to 2015. He also served as the Guest Editor for several special issues, including IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, *IEEE Wireless Communications Magazine*, *IEEE Communications Magazine*, *IEEE Vehicular Technology Magazine*, IEEE TRANSACTIONS ON COGNITIVE COMMUNICATIONS AND NETWORKING, and *Applied Energy* (Elsevier). He is a Distinguished Lecturer of the IEEE Vehicular Technology Society and also a Highly Cited Researcher by Clarivate Web of Science.