Proceedings Paper:

Naseh, D., Shinde, S.S. and Tarchi, D. (2024) Multi-Layer Distributed Learning for Intelligent Transportation Systems in 6G Aerial-Ground Integrated Networks. In: European Conference on Networks and Communications (EuCNC). 2024 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit), 03-06 Jun 2024, Antwerp, Belgium. Institute of Electrical and Electronics Engineers, pp. 711-716. ISBN: 979-8-3503-4500-1. ISSN: 2475-6490. EISSN: 2575-4912.

https://doi.org/10.1109/eucnc/6gsummit60053.2024.10597130

Multi-Layer Distributed Learning for Intelligent Transportation Systems in 6G Aerial-Ground Integrated Networks

David Naseh*, Swapnil Sadashiv Shinde*†, and Daniele Tarchi*
*Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi",
University di Bologna, 40136 Bologna, Italy, Email: {david.naseh2,daniele.tarchi}@unibo.it
†CNIT - University of Bologna Research Unit, Bologna, Italy, Email: swapnil.shinde2@unibo.it

***Abstract*—Federated Learning (FL) is a widely used distributed learning (DL) method for intelligent transportation systems (ITS) in the upcoming era of 6G-enabled ITS. In this work, we present the concept of Generalized Federated Split Transfer Learning (GFSTL) as a highly efficient and secure distributed learning framework for resource-limited ITS applications. The proposed GFSTL solution performs better in terms of overall training latency and accuracy and is useful for enabling ITS services in Aerial-Ground Integrated Networks (AGIN). Through comprehensive simulations carried out in vehicular scenarios, our results validate the efficacy of GFSTL on multilayered DL using Road-Side Units (RSUs) and High-Altitude Platforms (HAPs) in AGIN, demonstrating significant improvements in addressing the demands of intelligent vehicular networks. Through the integration of advanced DL techniques and the use of HAPs, our proposed framework holds promise for paving the way for an intelligent and connected vehicular network in the future.**

***Index Terms*—Distributed Learning, Federated Learning, Transfer Learning, Split Learning, Intelligent Transportation Systems, Non-Terrestrial Networks**

I. Introduction

6G-enabled transportation networks are expected to play an important role in shaping the society of the 2030s toward a more advanced, digitized, and connected world. With the help of 6G and other innovative technologies such as the Internet of Things (IoT), Machine Learning (ML), and edge computing, it is expected that transportation networks evolve into Intelligent Transportation Systems (ITS) [1]. ITS aims to improve the safety, efficiency, and sustainability of traditional transportation systems while serving users. In this context, a significant volume of data can be generated by heterogeneous IoT devices installed on different components of ITS. These data can be used to facilitate intelligent solutions [2]. However, the process of collecting and evaluating data from various components of ITS can present a significant obstacle. Additionally, with advanced hardware technologies, IoT nodes can generate high-quality data samples, e.g., high-definition images and videos, and sensor data. In latency-critical ITS, it is no longer feasible to transmit all IoT data to generate a common dataset. Therefore, the utilization of centralized learning methods in ITS is restricted. On the other hand, Distributed Learning (DL) techniques, such as Federated Learning (FL) [3], are emerging to effectively address these challenges and train ML models in a distributed way.

FL is a DL paradigm for collaborative model training without sharing the individual element's data. Devices can train local models and update a central server that then aggregates and applies updates to the shared model. However, the drawback of FL lies in the requirement for each client to train the entire resource-intensive ML model. This is particularly true for Deep Neural Networks (DNNs) deployed on end devices with limited resources, such as those utilized in ITS [4]. Although FL is a privacy-preserving training approach, recently, new concerns have emerged, mainly due to the iterative transmission of local and global model parameters, leading to problems such as poisoning, attacks, and model inversions [5]. Several of these issues can be related to the possible requirement of transmitting a complete model in traditional FL environments.

Split Learning (SL) is another DL method that can enable efficient distributed ML model training on resource-constrained devices. With SL, the ML model is split into two parts, where individual parts can be trained on server and client devices [6]. Unlike FL, with SL, only a portion of the model is trained on client nodes, effectively reducing the processing and communication load on resource-limited devices. The communication process is limited to cut-layer activation, ensuring model privacy. With these advantages, SL can be useful in reducing overall training costs and privacy concerns in federated environments, and can even enable FL frameworks to train more advanced DNN models. Some recent FL frameworks that take advantage of SL can be found in [7].

On the other hand, Transfer Learning (TL) solutions from the meta-learning family can increase the efficiency of ML model training [8] by knowledge transfer from previous source tasks to new target tasks. TL accelerates convergence, reduces data needs, and improves ML robustness in diverse vehicular contexts. Therefore, TL can further complement the FL process enabled by SL to further improve the distributed training process, in different ITS scenarios [9].

From a networking perspective, Non-Terrestrial Networks (NTNs) have acquired a central position in the 6G vision, mainly to provide global coverage and capacity boosts for

This work was partially supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 - program "RESTART")

traditional terrestrial networks. Different NTN platforms can help Vehicular Users (VUs) to enable flexible intelligent solutions. In particular, High-Altitude Platforms (HAPs), with their reduced transmission distances, higher coverage, and easy and flexible deployments, can complement terrestrial vehicular settings to enable efficient DL solutions [10].

With these motivations in mind, in this work, we propose a novel DL methodology named Generalized Federated Split Transfer Learning (GFSTL) to effectively train ML models in Aerial-Ground Integrated Networks (AGIN) for resource-constrained ITS scenarios. Building on [11], where we presented a Federated Split Transfer Learning (FSTL) framework tailored for a group of vehicles with homogeneous application requirements, we now introduce a GFSTL solution capable of catering to vehicular users with heterogeneous intelligent service demands. More specifically, the GFSTL approach enables multiple groups of VUs to participate in the FSTL procedure by employing multiple edge computing layers of an AGIN. The proposed architectural design holds promise in terms of reduced latency and an increased number of users involved in the federated training process, allowing for higher accuracy. Additionally, it can inherently reduce privacy concerns with the integrated model split approach. Next, we evaluate the proposed method in a typical AGIN vehicular scenario with ResNet on the MNIST dataset. The simulation results demonstrate better performance compared to traditional FL methods in terms of convergence rate, training accuracy, and overall latency.

II. Generalized Federated Split Transfer Learning

We consider a vehicular scenario composed of a set of n VUs $\mathcal{V} = \{v_1, \cdots, v_i, \cdots, v_N\}$ having their own labeled dataset $\mathcal{D}_i = \{(\mathbf{x}_k, y_k)\}$, for $1 \leq k \leq K_i$, with K_i data samples, where $\mathbf{x}_k \in \mathcal{R}^n$ is the n dimensional feature vector associated with the kth data sample, while y_k is the corresponding label. VUs aim to solve a learning task p and the objective is to learn a global model W_p that minimizes a given loss function L^p among all participants without sharing the raw data.

Supposing to use FL, in each round t, the participant v_i updates its local model parameters by performing a gradient descent step on its local loss function $L_i^p(\mathcal{D}_i, W_t)$, where W_t represents the global model based on the parameter updates received by the server in the previous round. The updated local parameters are then sent to a central server, which aggregates the model updates between participants using a weighted averaging scheme (e.g., FedAvg) to obtain the global model defined as $W_{t+1} = \frac{1}{N}\sum_{\forall i \in N} W_{i,t}^p$, where $W_{i,t}^p$ is the local model parameters of v_i. The FL process iterates until it achieves predefined stopping criteria, e.g., the number of iterations and the convergence value of the loss function.

Although FL offers benefits in DL, data privacy, and data transmission costs, it often requires many iterations for good performance. Each iteration strains resource-limited ITS nodes, affecting latency and energy. Repeated communication raises privacy concerns, making it challenging to train complex ML models in traditional centralized FL settings.

SL is an ML paradigm that can efficiently train complex models via model splitting. In an ITS scenario, the model W^p can be split into (i) Vehicle split model S, processing local raw data and transmitting intermediate representations $H_i = S(\mathcal{D}_i)$ to (ii) the Server split model M (e.g., Road Side Units - RSU), which performs further calculations and updates the global model W^p. This addresses FL's cost and privacy concerns. However, SL presents issues such as limited split-part power, the risk of performance loss, and high delays due to serial training. To address these, SplitFed (Federated Split Learning) combines FL and SL [6]. SplitFed or FSL blends FL's collaborative and privacy strengths with SL's local processing. In the SplitFed framework, each participant v_i applies the split model to process their local dataset $\mathcal{D}_i$, resulting in H_i. This H_i is then shared with a server. After merging, these updates are used to enhance the model W^p through Federated Learning (FL), promoting collaboration and model improvement.

The performance of the FSL approach can be further improved through TL integration, since with the use of a pre-trained neural network model, the initial knowledge can be transferred to local devices, providing a valuable initial start for the training process [12]. In the proposed solution, every VU is provided with a pre-trained ML model denoted as $W^{p'}$, which is specifically designed for the task p'. This task can involve transfer learning, such as fine-tuning with a pre-trained neural network. This approach can accelerate convergence compared to the untrained model W^p and reduce the amount of time and computational resources needed for training. Furthermore, TL allows devices with limited computational capabilities to participate in the FSL process, thus enhancing the training performance of the FSL model.

In our approach, we consider a pre-trained neural network model $W^{p'}$ with L layers. Next, $W^{p'}$ is split at a specific layer $\bar{k}$, called the cut (or splitting) layer, such that $1 < \bar{k} < L$, with the first part $W_{VU}^{p'} = W^{p'}(l \leq \bar{k})$ serving as the VU split model and the rest $W_S^{p'} = W^{p'}(l > \bar{k})$ as the SL server side model. During the training process, any ith VU uses vehicular data D_i and the VU-side TL model $W_{VU}^{p'}$, to produce the intermediate representation H_i that is then transmitted to the SL server-side model ($W_S^{p'}$). The SL server-side model performs further computations using these intermediate representations and updates the global model parameters.

By incorporating TL into FSL, we can leverage the advantages of pre-trained models in the VU split model ($W_{VU}^{p'}$), obtaining the FSTL. The pre-trained layers up to the splitting layer $\bar{k}$ capture general patterns and characteristics, while the remaining layers in the SL server-side model ($W_S^{p'}$) enable collaborative learning and model improvement in distributed VUs. In Fig. 1 the system architecture is represented where, unlike traditional SL, all clients communicate with the SL and FL servers simultaneously while performing their computations in parallel, so a higher convergence rate is expected. This

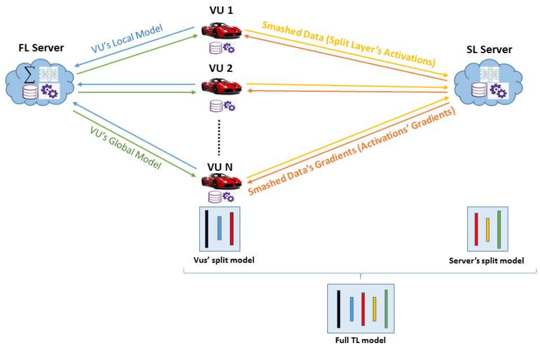

Fig. 1. Overall structure of FSTL

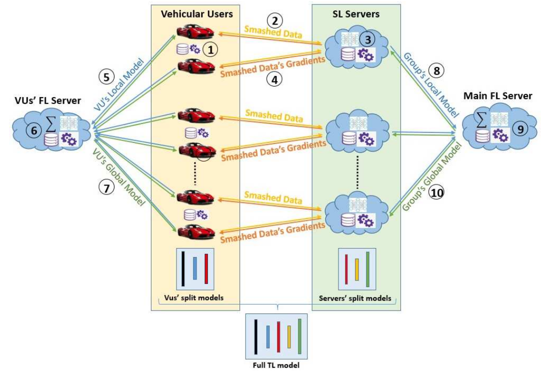

Fig. 2. Generalized FSTL

integration also allows efficient knowledge transfer, improved model performance, and faster convergence in the vehicular scenario while addressing resource constraints and privacy concerns.

In this process, the VU split model $W_{VU}^{p'}$ performs local processing on vehicular data to generate intermediate representations. These representations are then shared and merged on the central server, allowing collaborative learning and model improvement. The merged representations are used to update the global model parameters θ through FL procedures. Each VU then uses the updated global model parameters to update their model parameters θ_i. This iterative process enables collective learning and adaptation of the FSTL model across distributed vehicle units while preserving privacy, addressing resource constraints, and leveraging TL to enhance model performance in the vehicular scenario.

The GFSTL is a generalization of FSTL in which the main server initially contemplates m groups, each group comprising several VUs. In a relatively straightforward setting, each group's VUs are distinct and no group possesses the same VU. Subsequently, the server model undergoes training and updating sequentially within the group, exactly the same as in FSTL, with only one copy of the server model being kept, while each group operates independently and in parallel. This process culminates in the acquisition of m submodels trained by the server at the end of each round. It is pertinent to note that, despite the label-sharing scenario presented in this study, it is feasible to apply various other configurations, such as without label-sharing, extended split, and vertically partitioned data. This procedure is represented in Fig. 2, where the appropriate numbers describe the procedure for the GFSTL as follows. At the beginning of training ①, each VU is furnished with an identical VU model and the server begins with its designated server model. The server selects the number of groups and maintains the same throughout the training process. Once training commences, all VUs undertake forward propagation on their respective VU models on their local data, independently and in parallel. They then transmit their smashed data to the server simultaneously ②. The server arbitrarily assigns all smashed data originating from a single VU to a specific group ③. Forward-backward operations within a group occur sequentially to the VUs' smashed data, while the groups operate in parallel ④. Each group provides a single-server model. The VUs obtain the corresponding gradients of their respective smashed data, acquired towards the end of the backpropagation on the server model, following which they complete their backpropagation, resulting in individual local VU models. In summary, the training sequence ensures that VUs obtain identical models initially with a uniform server model. Random data allocation among groups evens the computational load. Group parallelization optimizes computing resources, speeding training. The use of local data by VUs ensures privacy by transmitting only smashed data. Gradients from the server's smashed data and backpropagation ensure model convergence. VU training updates the models locally, reducing server-VU communication. Sequential forward-backward operations in a group process accurately smashed the data. This process balances computational efficiency, privacy, and model accuracy. The VUs' FL server integrates the VU split models⑤-⑦, while the main FL aggregates and integrates server and VU split models into a unified global model for the next round. Aggregation occurs synchronously at the end of each round ⑧-⑩.

Although we can simplify the procedure by using a single server, practical scenarios involve the main and client servers independently aggregating split models. This structure enhances privacy by isolating client split models from the main server, since privacy protection is crucial in FL due to sensitive data. These measures ensure secure and safe FL benefiting all parties involved.

Algorithm 1 details the GFSTL process for each group of VU-RSU in the vehicle scenario considered. The procedure begins by initializing the VU local model parameters and the server model parameters, namely θ_i and θ_s, with the pre-trained parameters of the used DNN (ResNet), i.e., $\theta_i(0)$ and $\theta_s(0)$, respectively, with i and N_j being the VU index and the total number of VUs in group j (lines 1–4). Then, each VU simultaneously performs forward propagation on its local model using its local data. The output of the forward propagation at v_i is indicated as H_i, which will then be sent to the server for further forward propagation (lines 5–6). The SL server in the RSU performs forward propagation in its server model using these smashed data received from each VU (line 7). Lines 8–10 describe how the SL server calculates the gradients of the parameters of the server model θ_s and sends the gradients, $\nabla\theta_{sj}$, back to the appropriate VU for additional backpropagation. The parameters of the local model are updated using an optimizer, such as stochastic gradient descent (line 11). Here, η is the learning rate. Next, the FL Server aggregates the gradients received from all VUs by performing federated weighted averaging. Let w_i denote the weight associated with v_i and define G_i as the gradients $\nabla\theta_s$ received from v_i (line 12). Subsequently, the parameters of the server model are updated using the aggregated gradients. The SL Server sends back the updated server model parameters θ_s to all participating VUs to update their local model parameters by performing federated averaging (line 13). Here, $\theta_{avg,i}$ represents the federated average of local updates at v_i. After that, the parameters of the local model are updated (line 14), where α is a hyperparameter that controls the weights of the server model. The final aggregation occurs on the HAP as the central FL server, resulting in a comprehensive and intelligent global model (lines 17-19) ready to be deployed across a spectrum of IoV tasks.

Algorithm 1 GFSTL iterative algorithm

Input: $N, \theta_i(0), \theta_s(0), \sigma, \alpha, \eta$
Output: θ_i, θ_s

1: **for** $1 \leq j \leq m$ **do**
2: Initialize FSL server j: $\theta_{sj} = \theta_{sj}(0)$
3: **for** $1 \leq i \leq N_j$ **do**
4: Initialize vehicular users: $\theta_i = \theta_i(0)$
5: Forward propagate on the v_i model using local data
6: Send intermediate smashed data (H_i) to RSU_j
7: Forward propagate on the server model (RSU-side model) using H_i
8: Back propagate $\nabla\theta_{sj}$ on RSU
9: Send back $\nabla\theta_{sj}$ from RSU to UAV_i
10: Back propagate on the v_i model using $\nabla\theta_{sj}$
11: Update v_i local model using an optimizer (e.g., SGD):
$$\theta_i \leftarrow \theta_i - \eta \cdot \nabla\theta_i$$
12: Aggregate (federated average) on RSU:
$$G_{avg} = \frac{1}{n} \cdot \sum(w_i \cdot G_i)$$
13: Update RSU model using an optimizer (e.g. SGD):
$$\theta_s \leftarrow \theta_s - \eta \cdot G_{avg}$$
14: Update VU_i local model parameters:
$$\theta_i \leftarrow \alpha \cdot \theta_s + (1 - \alpha) \cdot \theta_{avg,i}$$
15: **end for**
16: **end for**
17: Send model parameters from RSU-VUs groups to the main FL server (HAP)
18: Aggregate the parameters to form a global model
19: Send back global model to the groups
20: **return** θ_i, θ_s

A. Latency Analysis for GFSTL

The overall latency is represented by the combined computation and communication time. It is important to note that all methods utilize a uniform data distribution. Let us assume that d represents the total size of the data, h denotes the size of the smashed layer, R signifies the data rate based on the communication medium, T indicates the training time required to train the entire DNN model, while T' and T'' represent the training time in FSTL and GFSTL using a pre-trained TL model. Furthermore, T_{FedAvg}, T_{Merge}, T_{VUFL}, and T_{MainFL} represent the time required to perform the aggregation of the full model in FL, the smashed parameters on the VU side, and the main FL servers, respectively. Furthermore, q denotes the total number of parameters in the full model and r means the ratio of the size of the VU submodel to that of the full models. The summarized results are presented in Table I. The presence of factor 2 in several terms, such as $2pr$, $\frac{2dh}{n}$, etc., is due to the download and upload of VU-side model updates before and after training. Numerical analysis reveals that when there is a large number of clients, SL can become inefficient, since the total time is directly proportional to n, mainly due to the serial training operation, whereas other methods perform the training in parallel. Furthermore, we can see that as n increases, the total time cost increases in this order: $FSTL < FSL < FL < SL$. To be more specific, FSTL is faster than FSL because it uses a pre-trained network ($T'' < T' < T$), and FSL has a lower latency than FL because it aggregates fewer parameters ($T_{Merge} < T_{FedAvg}$). Regarding GFSTL, the total latency depends on the number of VUs in the jth group, that is, n_j. We used the operator $\max(\cdot)$ since the total communication time depends on the slowest group.

III. GFSTL in Vehicular AGIN

Aerial networks have emerged as a promising solution to expand connectivity and support advanced applications in challenging environments [10]. Vehicular scenarios, characterized by high mobility, intermittent connectivity, and dynamic data distribution, pose significant hurdles to traditional ML techniques. In [11], the same authors introduced the concept of FSTL in the context of vehicular scenarios. In the current study, we extend its applicability and integrate it with aerial networks to establish a unified AGIN for ITS, where GFSTL can be fully exploited. Integration of the capabilities of HAPs with FSTL allows us to use the GFSTL concept and harness the potential of aerial networks to enable efficient and secure training and inference for VUs in vehicular networks.

In the proposed scenario, RSUs serve as SL and FL Servers for VUs and execute FSTL, while HAPs are the main FL servers. SL enables the VUs to keep the raw data on their local devices while offloading computationally demanding tasks to the HAPs, such as model training. This approach preserves privacy and reduces the communication burden, since only model updates are transmitted between the VUs and the HAPs. The FL paradigm also enables collaborative model training across multiple VUs, promoting knowledge sharing and adaptation to diverse vehicular environments.

The integration of FSTL with HAPs opens up new possibilities for vehicular scenarios. First, the high-altitude placement of HAPs ensures broader coverage and reduced interference, enabling seamless connectivity for VUs even in remote areas or areas with limited terrestrial infrastructure. Second, FL architecture fosters collective intelligence among VUs, leading to improved models that can adapt to the varying conditions and scenarios encountered by different vehicles. Using AGIN and GFSTL through RSUs and HAPs, our proposed approach paves the way for intelligent vehicular networks that offer

TABLE I
LATENCY ANALYSIS OF FIVE DL METHODS PER ROUND

Learning Method	Training+Aggregation Time	Communications per VU / server	Total communications / server	Total communication time	Total latency
FL	$T + T_{\text{FedAvg}}$	$2q$	$2nq$	$\frac{2q}{R}$	$T + T_{\text{FedAvg}} + \frac{2q}{R}$
SL	T	$\frac{2dh}{n} + 2qr$	$2dh + 2nqr$	$\frac{2dh}{R} + \frac{2nqr}{R}$	$T + \frac{2dh}{R} + \frac{2nqr}{R}$
FSL	$T + T_{\text{Merge}}$	$\frac{2dh}{n} + 2qr$	$2dh + 2nqr$	$\frac{2dh}{nR} + \frac{2qr}{R}$	$T + T_{\text{Merge}} + \frac{2dh}{nR} + \frac{2qr}{R}$
FSTL	$T' + T_{\text{Merge}}$	$\frac{2dh}{n} + 2qr$	$2dh + 2nqr$	$\frac{2dh}{nR} + \frac{2qr}{R}$	$T' + T_{\text{Merge}} + \frac{2dh}{nR} + \frac{2qr}{R}$
GFSTL	$T'' + T_{\text{VUFL}} + T_{\text{MainFL}}$	$\frac{2dh}{n_j} + 2qr$	$2dh + 2n_jqr$	$\max\left\{\frac{2dh}{n_jR}\right\} + \frac{2qr}{R}$	$T'' + T_{\text{VUFL}} + T_{\text{MainFL}} + \max\left\{\frac{2dh}{n_jR}\right\} + \frac{2qr}{R}$

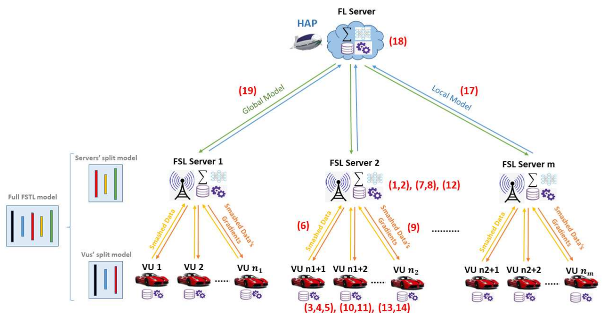

Fig. 3. GFSTL structure for AGIN.

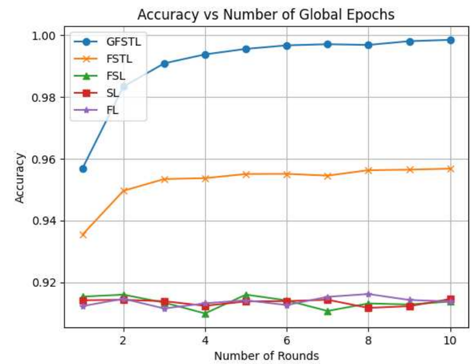

Fig. 4. Accuracy of different DL methods vs the number of rounds.

enhanced connectivity, privacy preservation, and improved decision-making capabilities.

Here, GFSTL is proposed in vehicular scenarios to enable collaborative learning across multiple vehicles while preserving data privacy. The procedure involves a central server (FSL Server, which here is an RSU) for each group of VUs and multiple VUs exchanging model updates and gradients. Subsequently, once complete FSTL models are obtained for all m groups of VU-RSU are obtained, the main aggregation is carried out on a HAP, which serves as the main FL server.

Finally, once the FSTL process is done on all groups of VU-RSUs, the complete FSTL model is sent to the main FL server (HAP) by all the RSUs, to complete the process by conducting aggregation for all the model parameters gathered.

In the next section, to assess the efficacy of the suggested GFSTL-based architecture, we consider a vehicular AGIN that consists of one HAP (as the main FL server), four RSUs, each serving as an FSTL server for 20 VUs, as illustrated in Fig. 3. The numbers in the figures correspond to the steps of the Algorithm 1.

IV. Simulations and Performance Evaluations

The proposed GFSTL method in vehicular AGIN is simulated on a Python-based platform with the help of additional ML libraries, including Pandas, Numpy, Mathplotlib. In addition, the NVIDIA® Tesla® T4-based GPU accelerator is also used for reduced training intervals.

The training has been conducted with ResNet on the MNIST dataset, which contains 60,000 training images and 10,000 testing images. To compare the convergence rate of the DL methods, the accuracy of GFSTL versus the number of training rounds has been first evaluated (Fig. 4). To conduct this simulation, we have split the data between four groups of five VUs; consequently, we have four SL servers and one main FL server. A notable observation is that GFSTL exhibited a much faster increase in accuracy compared to both the original FL and SL and their combination, i.e., FSL as well. This can be attributed to the leverage of knowledge transfer, which is facilitated by using TL. Since TL allows leveraging pre-trained models, which have already learned useful features from large-scale datasets, by initializing the client models with pre-trained weights, GFSTL can lead to improved model performance compared to starting from scratch in SL, FL, or FSL. In this way, we expect GFSTL to be able to handle scenarios with heterogeneous data between client devices. The pre-trained model's knowledge provides a robust starting point for all client devices, even if their local datasets vary in size or quality. This helps address the challenges associated with the heterogeneity of the data.

In scenarios where VUs have limited or heterogeneous local data, our proposed architecture helps overcome the challenges of data scarcity. By leveraging the knowledge from a pre-trained model, the client models can benefit from the available local data and achieve good performance even with smaller datasets. In Fig. 5, we have compared our proposed architectures' final testing accuracy after 10 global epochs with that of FL, SL, FSL and FSTL versus the number of batches of VUs, each batch consists of 10 VUs. For this purpose, we split the dataset between users, so we decreased the amount

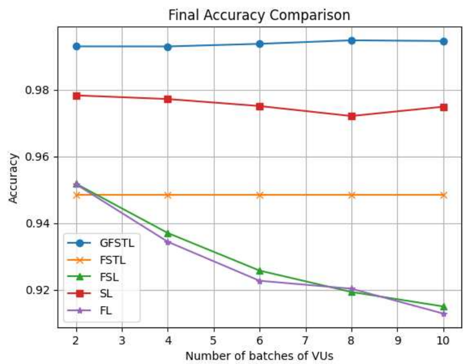

Fig. 5. Accuracy of different DL methods vs the number of batches of VUs.

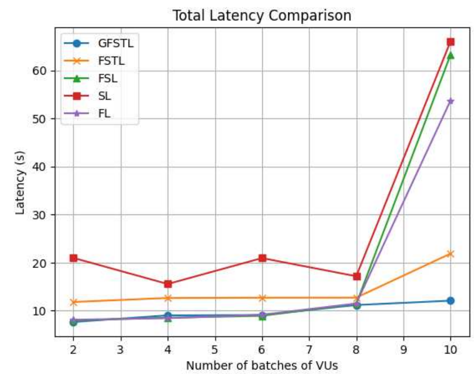

Fig. 6. Latency of different DL methods vs the number of batches of VUs.

of data available for local training. It is evident that in FL the accuracy decreases with an increase in the number of users. On the contrary, GFSTL is less sensitive to the number of devices and the amount of data at hand, as we observe almost the same accuracy even with different numbers of VUs. This is a significant advantage of GFSTL over other distributed methods.

GFSTL also reduces communication overhead compared to traditional FL. Instead of sending raw data or gradients, only intermediate representations (derived from SL) are communicated between client devices and the server. This reduces bandwidth requirements and speeds up the training process. To demonstrate this, we conducted a simulation to demonstrate the superior performance of GFSTL compared to other methods regarding latency, as the number of users increases. From Fig. 6, it is evident that with a higher number of VUs, the latency, which is the sum of computation time and communication delay, is significantly higher in SL. The reason for this is that the training process in SL is serial, unlike parallel aggregation in FL and FL-inspired methods such as GFSTL, which makes it much slower compared to the other methods. Additionally, our proposed architecture only trains a portion of the model on resource-constrained devices and communicates solely the gradients of the final (cut) layer on the user side, which is less than the entire model parameters in FL. Therefore, according to Shannon's formula, the communication delay, which is proportional to the data, is reduced, and the processing time in GFSTL is not only minimal compared to other methods, but also almost constant with a change in the number of VUs, which is a great advantage of our proposed method. In conclusion, our study demonstrates the effectiveness of GFSTL in mitigating latency issues in FSL as the number of users increases.

V. Conclusion

In this work, we have presented a novel multilayer DL approach called GFSTL to enable an efficient FL process by integrating SL and TL while leveraging AGIN as a framework for intelligent vehicular networks in the upcoming era of 6G-enabled ITS. Our proposed GFSTL approach for HAPs overcomes the restrictions of conventional DL methods and provides significant benefits in terms of learning efficiency, accuracy, privacy protection, and overall latency. Furthermore, our analysis of latency and simulations have shown that by altering the number of DL participants (VUs) while retaining the same data quantity, the performance in terms of accuracy and latency can be improved. Although challenges such as compatibility, domain change, and model drift require further attention, our framework sets the stage for intelligent VNs by offering enhanced connectivity, privacy preservation, and improved decision-making capabilities.

References

[1] F. Tang *et al.*, "Future intelligent and secure vehicular network toward 6G: Machine-learning approaches," *Proc. IEEE*, vol. 108, no. 2, pp. 292–307, 2020.

[2] A. A. Brincat *et al.*, "The internet of things for intelligent transportation systems in real smart cities scenarios," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, 2019, pp. 128–132.

[3] S. S. Shinde *et al.*, "On the design of federated learning in latency and energy constrained computation offloading operations in vehicular edge computing systems," *IEEE Trans. Veh. Technol.*, vol. 71, no. 2, pp. 2041–2057, 2022.

[4] L. Ridolfi *et al.*, "Implementation and evaluation of a federated learning framework on raspberry PI platforms for IoT 6G applications," *Future Internet*, vol. 15, no. 11, p. 358, 2023.

[5] H. Chen *et al.*, "Privacy and fairness in federated learning: On the perspective of tradeoff," *ACM Comput. Surv.*, vol. 56, no. 2, pp. 1–37, Sep. 2023, article no. 39.

[6] Y. Gao *et al.*, "Evaluation and optimization of distributed machine learning techniques for internet of things," *IEEE Trans. Comput.*, vol. 71, no. 10, pp. 2538–2552, 2022.

[7] X. Liu *et al.*, "Wireless distributed learning: A new hybrid split and federated learning approach," *IEEE Trans. Wireless Commun.*, vol. 22, no. 4, pp. 2650–2665, 2023.

[8] R. Vilalta and M. M. Meskhi, "Transfer of knowledge across tasks," in *Metalearning: Applications to Automated Machine Learning and Data Mining*. Cham: Springer International Publishing, 2022, pp. 219–236.

[9] W. Qi *et al.*, "A resource-efficient cross-domain sensing method for device-free gesture recognition with federated transfer learning," *Trans. Green Commun. Netw.*, vol. 7, no. 1, pp. 393–400, 2023.

[10] S. S. Shinde and D. Tarchi, "Joint air-ground distributed federated learning for intelligent transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 9, pp. 9996–10 011, Sep. 2023.

[11] D. Naseh *et al.*, "Enabling intelligent vehicular networks through distributed learning in the Non-Terrestrial networks 6G vision," in *European Wireless 2023 (EW 2023)*, Rome, Italy, Oct. 2023, preprint available at https://doi.org/10.48550/arXiv.2310.05899.

[12] N. Girelli Consolaro *et al.*, "Analysis and performance evaluation of transfer learning algorithms for 6G wireless networks," *Electronics*, vol. 12, no. 15, p. 3327, 2023.