Federated Learning Approach Decouples Clients from Training a Local Model and with the Communication with the Server

Konstantinos D. Stergiou
Department of Applied Informatics
University of Macedonia
Thessaloniki, Greece
kster@uom.edu.gr

Konstantinos E. Psannis
Department of Applied Informatics
University of Macedonia
Thessaloniki, Greece
kpsannis@uom.edu.gr

***Abstract*— Traffic sign recognition and autonomous vehicles computing are a few of the innovative applications which are emerging in the domain of mobile edge computing. Distributed machine learning in the form of Federated Learning (FL) has been applied to mobile edge computing through a range of methodologies and techniques for intelligent feature classification approaches. The challenges that research on such FL methods is facing is twofold: identify an optimal distributed architecture and algorithm components to each side to meet the demand of heavy data processing, and enhance the algorithm components with heuristics that fit to the problem domain and optimize the key parameters of the algorithms. In this prospect, we present a Federated Learning implementation based on a neural network architecture with emphasis to traffic sign image recognition. Our benchmark was tested with two FL strategies seeking an optimal performance model and in reference to a corresponding data set. We present the results of this work while we define the scope of future improvements to our model.**

Keywords*— Federated Learning, Mobile Edge Computing, Traffic Sign Image Recognition, Convolutional Neural Networks*

I. INTRODUCTION

Federated Learning emerged as client devices hardware characteristics evolved and there were able to perform much more complex computations. Client devices become more powerful and last more most of the times does not execute only one task, e.g. act as a receiver, but they execute many parallel actions, e.g. mobile phones are able simultaneously to handle calls, receive messages and upload data. For this reason, client devices most of the times act as a multi-purpose device that are able to perform many actions. Furthermore, the evolution of networks enable all kind of devices to transmit much faster and more reliable huge amount of data, e.g. 5G.

The models trained on the data existing on each client and only the updated parameters are sent to the central server. The central server aggregates the updated parameters sent by the clients and produces the final machine learning model. A complete ecosystem has been developed around containers enabling software engineers to develop more complex and advanced applications. Containers exist in the most services / solutions that are currently online either in on premises infrastructure or in the cloud.

II. COLLABORATIVE APPROACH TO MACHINE LEARNING

Federated learning is a relatively new, collaborative approach to machine learning that avoids the aggregation of all trained data at a certain point. This is particularly important since the data remains throughout the training on the devices that produce it [1]. Limiting unnecessary communication of data helps to protect the training data from various types of attacks (e.g., poisoned data by malicious clients, backdoor attacks, advanced persistent threat attacks, etc.) and reduces the risk of unwanted access to them [17].

A. Communication of data to protect the training data

Through federal learning, the training of a machine learning model, such as a neural network, is taking advantage of many local datasets. Client devices are becoming more powerful and longer-lived, and in most cases not just one task, such as acting as a receiver, but performing many parallel operations, such as the phone can make calls, receive messages at the same time and upload data. Because of this, client devices mostly act as multipurpose devices that can perform many operations. This data remains permanently in local nodes and are not accessible outside of them. Instead of moving the data itself [2], certain local models are trained based on the corresponding local data sets and then transfer their parameters, which are then transferred only their weights and polarities. Through this process, a single and global model is finally concluded, which is trained without ever having access to the data itself. In summary, the copies of a central model are distributed to a number of distributed nodes, which train them based on their own local datasets and send the results to the central node in order to synthesise the general model.

One of the most known algorithms used to train the model in Federated Learning is Federated Averaging algorithm. The model can be trained if we execute the following steps (Figure 1):

1. Server selects n clients from the pool
2. Server sends to these clients the initial parameters Θ_t
3. Each client receives initial parameters Θ_t from the server
4. Each client in order to produce an updated parameter Θ' runs some iterations of SGD (Stochastic Gradient Descent)
5. Each client returns $\Theta' - \Theta_t$ to the server
6. Server calculates weighted average of the client updates as $\Theta_{t+1} = \Theta_t - data$

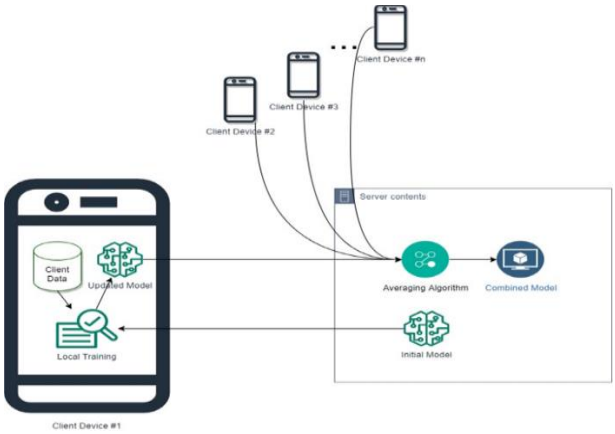

Fig. 1. Federated Learning Proposal

B. Data privacy and security challenge

In order to achieve the protection of personal data, it is not enough to comply to an appropriate legislative framework. The design of the systems should address the requirement of processing personal data on the principles of privacy [3]. As far as machine learning systems are concerned, various methods have been proposed and developed to protect privacy. For example, [16] presents four privacy-preserving categories: encryption-based, perturbation-based, anonymization-based, and hybrid approaches.

The encryption-based techniques include homomorphic encryption, secret sharing, and secure multi-party computation. The perturbation -based techniques (e.g., differential privacy, additive perturbation, multiplicative perturbation) add statistical noise to the original data, so that the reverse mapping of perturbed data to the original data to become almost impossible.

Anonymization techniques (e.g., k-anonymity, l-diversity, and t-closeness) are establishing group-based anonymization without sacrificing the utility of the produced data.

C. Lenet-5 CNN model

The main advantage of Convolutional Neural Networks (CNN) compared to other Neural Network models of federated learning is that it automatically detects important features without any human supervision [5]. They use special concatenation and grouping functions and perform parameter sharing making it computationally efficient and allowing it to work on any device. Indeed, experiments with a combined Lenet-5 for local training and FedAvg for the training of the global model [6] approach has been executed by the authors with a large database (GTSRB) of traffic sign images. The experiments validated the hypothesis that the Lenet CNN architecture provides higher quality on the local training model.

Our objective is to deepen our research into the domain of edge computing. Thereby, experiments should be carried out using our proposed algorithm as part of a multi-layer architecture for client, regional and cloud aggregation. This is for handling edge computing constraints such as lack of resilience to mobile client communications [7], capacity restrictions and data privacy. Furthermore, the algorithm may be evolved to incorporate a client selection method so that a proportion of the connected clients may be contributing to local training at each communication round without reducing accuracy [8]. An approach that is currently our research by the authors is the aggregation of client data to an edge layer before transmitting them to the global model. Gradient Descent, and Adam Optimizer are some of alternative methods that could be used instead of the Lenet-5 CNN model. In a similar traffic sign classification problem using the GTSRB database, reports that Gradient Descent maintains a consistent training accuracy of 100% after 20 iterations[9], whereas the performance of Lenet-5 and Adam Optimizer is fluctuating. Also, although our Lenet-5 reaches 100% of training accuracy in less than 20 iterations, the accuracy of the model is fluctuating as well. Interestingly, Adam Optimizer performs better than the other algorithms with regards to testing accuracy. Therefore, a useful experiment would be the combination of the Lenet-5 CNN model with Adam optimizer [10] to reach the accuracy of Gradient Descent but in less iterations. This is based on the advantage of our model and Adam Optimizer of a non-static learning rate in comparison to Gradient Descent.

The systems will autonomously perform predefined scaling actions and meet the pre-agreed performance requirements with the minimum resource requirement. The mechanisms and workflows used by the system to fulfil flexibility goals, as well as [11] any evaluation criteria and support any decision-making process vary from one system to another or from one application to another, even within the same system.

D. Complexity of the proposed CNN model

The computational complexity of the proposed Lenet-5 algorithm takes into account the resources required and the convolutional steps taking place at each layer. As shown on table [12] the convolutional layers consist of a number of cores to reduce the dimensionality of the input images into a featue map. Then a mean sampling level method is executed to reduce the size of each feature map and an activation function to provide the output dimensions as input to the next layer. The algorithm goes through leveling plane methods of 3 more layers (FC1, FC2, FC3) which convert the feature maps into vectors of elements. The number of neurons used per each layer is equal to the number of different categories to be identified. Finally, a Softmax activation function is used at the final layer to normalize the output of each neuron in the scale of (0,1).

Layer	**Input dimensions**	**No of weights**	**Output dimensions**
Convolutional 1 (C1)	1x32x32	60	6x30x30
Average sampling 1 (AP1)	6x30x30	0	6x15x15
Convolutional 1 (C2)	6x15x15	880	16x13x13
Average sampling 2 (AP2)	16x13x13	0	16x6x6
Leveling (FL)	16x6x6	0	1x1x576
Fully Interconnected 1 (FC1)	1x1x576	69240	1x1x120
Fully Interconnected 2 (FC2)	1x1x120	10164	1x1x84
Fully Interconnected 3 (FC3)	1x1x84	850	1x1x10

III. FL ARCHITECTURE FOR AUTONOMOUS VEHICLE SERVICE.

Traffic sign recognition is considered one of the fundamental techniques for autonomous vehicles. Colors are

very important clues to identify traffic signs however research work has indicated that high accuracy or consideration of light variation requires advanced methods to process them, for example Deep Learning [13]. Convolutional neural networks are such a deep learning method which parses raw images, extracts and learns hierarchically high-level features. Inputs of CNNs are processed either as gray images or as three independent color channels and other transformations need to be applied to explore and encode all features of traffic signs.

We propose a Federated Learning model based on a 5 layers Lenet CNN algorithm to extract hierarchically high-level features of traffic sign images. We take as case study the GTSRB data set. Experimental results illustrate that the proposed method can yield correct recognition results and achieve an efficient computing architecture which scales up by allocating optimally resources.

The client model makes use of specialized classes to parse the raw input and extract its high level features which are then forwarded to the Lenet CNN model for training and validation. Specific parameters (e.g. number of clients, epochs, learning rate, classifier) are available for further tuning of the performance and scalability metrics of the client. The server implements the Federated Averaging algorithm for updating the global model and return to the clients the updated weights.

Our initial experiments have been done using the pytorch framework. The FL client includes three parts: a) The Lenet CNN class which implements the CNN model, b) the GTSRB Client class which implements the parsing of the road signs and feeding into the CNN model, and c) the local client which implements the local training, validation functions and communication with the server.

The GTSRB Client class uses as input the following training settings:

- Data (default='data'): the folder where data is located
- batch-size (default=64): the input batch size for training
- Epochs (default=100): the number of epochs to train
- Lr (default=0.0001): the learning rate
- Seed (default=1): defines a random seed
- log-interval (int, default=10): defines how many batches to wait before logging training status

A function *load_data* in this class takes as input the aforementioned parameter settings to prepare the reading of the image data set. The function applies data transformations to augment the input images with a specific set of features (‘filters’) such as brightness, hue, saturation, rotation, hvflip, alignment. This is to filter any jitter occurring to the images due to horizontal lines of image frames which are randomly displaced due to the corruption of synchronization signals or electromagnetic interference during transmission. The function is invoked returning an object which holds in memory the set of data to be used for training.

The server side implements the global training model on the received data enforcing a Federated Averaging strategy for updating the global weights on the trained data.

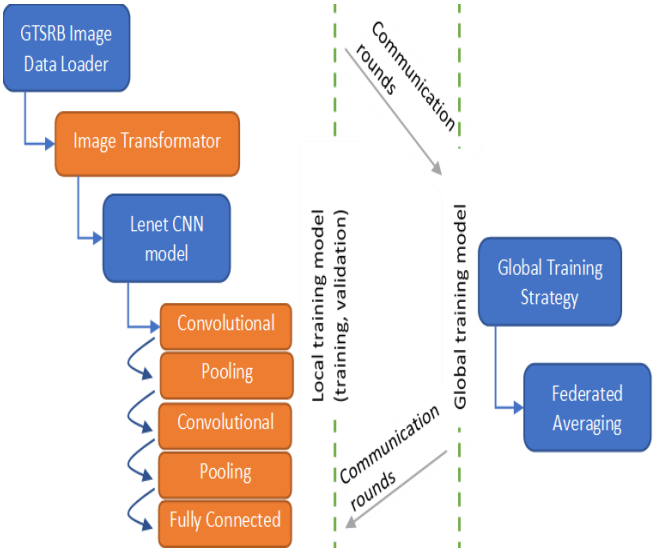

Fig 2, Lenet CNN model (FL client)

The Lenet CNN class includes two main functions: *init and forward*. As Fig 2 elaborates, the *init* function actually invokes the required objects to define the layers of the convolutional neural network. The *Conv2d* builds a convolutional layer of two dimensions and supports Tensor Float of 32 cores which are designed to achieve better performance on matmul and convolutions by rounding input data to have 10 bits of mantissa. The *Linear* applies a linear transformation to the incoming data and supports Tensor Float of 32 cores as well. The *CrossEntropyLoss* function is used as the error function. A prerequisite for its use is that the data set categories are mutually exclusive. In the GTSRB this condition is met, since each image represents only one handwritten digit.

```
class MyLeNet(input parameters):
 def init():
  conv1 = Conv2d (3, 6, 5)
  conv2 = Conv2d (6, 16, 5)
  fc1 = Linear (16 * 5 * 5, 120)
  fc2 = Linear (120, 84)
  ceriation = CrossEntropyLoss()

 def forward(Tensor) -> Tensor:
     Tensor = conv1 (Tensor)
     Tensor = F.max_pool2d (Tensor, 2, 2)
     Tensor = relu (Tensor)
     Tensor = self.conv2 (Tensor)
     Tensor = max_pool2d (Tensor, 2, 2)
     Tensor = relu (Tensor)
     Tensor = view (-1, 16 * 5 * 5)
     Tensor = fc1 (Tensor)
     Tensor = fc2 (Tensor)
     return Tensor
```

The forward function takes as input the defined CNN model (a Tensor object) forwarding the output of one layer as input to another layer. The Max_pool2d applies a two dimensional max pooling over an input signal composed of several input planes. The ReLU function applies the rectified

linear unit function element-wise. The View function returns a new tensor with the same data as the input tensor but of a different shape. The fc1 and fc2 functions return a new tensor with the same data as the input tensor but of a different size.

A. *Federated Algorithm strategies*

Different heuristics apply depending on the server model and client of each proposed algorithm. We are implementing a Federated Averaging strategy for the global model. The strategy considers a sample of 10% of available clients for the next communication round, while we are setting a minimum number of clients to be sampled for the next round (default is 2) and a minimum number of clients that need to be connected to the server before a training round can start (default 2). As Fig 3 illustrates, we define the fitness configuration with a default batch size and learning rate.

```
# Define configuration parameters of the strategy
def fit_config(): configuration object
  set parameter learning_rate: 0.001 in configuration
  set parameter batch_size: 0.001 in configuration
  return configuration

# Define the Federated Learning Strategy
strategy =FedAvg (fraction_fit=0.1, min_fit_clients=2,
min_available_clients=2, on_fit_config_fn=
Configuration)

#Start the server with the configured strategy
start server (IP address: port, no of communication
rounds:20, strategy=strategy)
```

As alternative option, we have considered the implementation of the Lenet CNN model at client side in combination with a Federated Averaging with Momentum (FedAvgM) strategy at server side [18]. We inherit the assumption of the FedAvgM strategy that it may handle more efficiently convergence issues if the data distribution is non independent identically distributed (non-IID) [18]. Our assumption is that this approach may handle more efficiently convergence issues if the data distribution is non independent identically distributed (non-IID). We are evaluating the hypothesis of whether the momentum factor on top of the Lenet CNN model of local client updates is improving performance in terms of accelerating the training and dampening the oscillations [18].
In addition to the parameters used in FedAvg strategy, FedAvgM, as an enhancement of FedAvg, includes the following parameters and functions:

- fraction_eval : Fraction of clients used during validation. Defaults to 0.1.
- min_eval_clients : Minimum number of clients used during validation. Defaults to 2.
- eval_fn : Optional function used for validation.
- on_fit_config_fn : Function used to configure training.
- on_evaluate_config_fn : Function used to configure validation.
- accept_failures : Whether or not accept rounds containing failures.
- server_learning_rate: Server-side learning rate used in server-side optimization.
- server_momentum: Server-side momentum factor used for FedAvgM.

IV. PERFORMANCE EVALUATION

Given the preparation of the GTSRB data set, we have proceeded to benchmark the performance of the Lenet CNN-FedAvg model in comparison to the Lenet CNN- FedAvgM strategy. For the purpose of our investigation on the domain of traffic sign recognition, our benchmark was setup to a server of the following resources:

- Processor: Intel Xeon CPU E5504 @ 2.00GHz, 1995 Mhz, 4 Core(s), 4 Logical Processor(s)
- RAM: 32GB

The table below explains the performance of the two strategies in terms of time and accuracy.

	Comm Rounds	Duration (secs)	Accuracy
FedAvg	20	31868	88,00%
FedAvg	40	47569	81,96%
FedAvg	60	71867	59,22%
FedAvg	80	145642	88,00%
FedAvg	100	189257	87,98%

FedAvg and FedAvgM have run under client batch size B = 32, local epoch counts E ∈ {5}, and 2 clients participating in every single round, respectively for a total of {20, 40, 60, 80, 100} communication rounds. The following figures explain the performance of the two strategies in terms of time and accuracy.

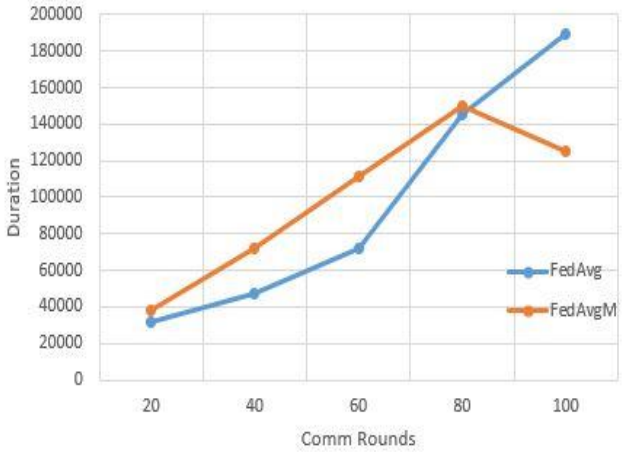

Fig 3. Experiment results: algorithm performance (time)

In Fig. 3, it is validated that the time is strongly dependent to the number of communication rounds (C) and number of epochs. As the number of epochs remains the same, the more the communication rounds the more it takes to produce the final outcome. For the majority of the communication rounds, FedAvg requires less time (in seconds) than FedAvgM whereas it maintains a constant peak of increase in duration. On the other hand, we notice that the effect of learning with

non-identical data and with server momentum is apparent for C=100. FedAvgM requires much less time than FedAvg to reach an optimal learning model.

The table below explains the performance of the two strategies in terms of time and accuracy.

	Comm Rounds	Duration (secs)	Accuracy
FedAvgM	20	38078	86,00%
FedAvgM	40	71881	83,71%
FedAvgM	60	111491	74,39%
FedAvgM	80	149712	50,13%
FedAvgM	100	125643	86,00%

Although the test accuracy of both FedAvg and FedAvgM maintains performance which is mostly above 80%, both do not stay constant along the line of the communication rounds. The performance of both algorithms fall rapidly below 80% when C={60, 80} and start to improve and converge when C=100. Overall, FedAvg stays relatively higher than FedAvgM, thereby the effect of learning with non-identical data and with server momentum is not apparent in this benchmark (Fig. 4). Further improvements to both strategies will be required.

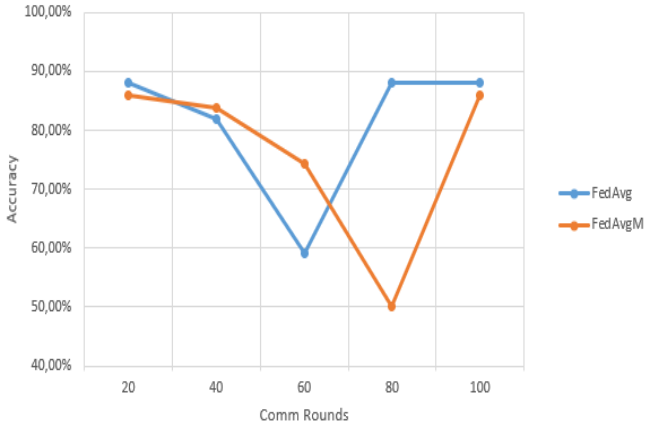

Fig. 4. Experiment results: algorithm performance (accuracy)

V. CONCLUSIONS

Our proposed federated learning approach decouples clients from training a local model and with the communication with the server. The only responsibility of the client is to create a container, send both data and initial parameters to it and destroy container upon creation of final model from the server. Even if the data are huge, if we are working on image processing, due to the evolutions of networks (e.g. 5G) this won't be a problem. So, we can use as client general purposes devices, like mobile phones, that will have various sensors and move all the processing to the cloud.

Our proposal will be evaluated against a predefined set of scenarios. This evaluation will help us to identify the value of our proposal, to find any weaknesses might exist and further develop it. These scenarios will include a predefined number of client devices and a set of expected results. The evaluation part is very crucial in our approach and it will use predefined metrics to evaluate the proposal we make.

REFERENCES

[1] Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., ... & Zhao, S. (2021). Advances and open problems in federated learning. Foundations and Trends® in Machine Learning, 14(1–2), 1-210.

[2] Gruendner, J., Schwachhofer, T., Sippl, P., Wolf, N., Erpenbeck, M., Gulden, C., ... & Toddenroth, D. (2019). KETOS: Clinical decision support and machine learning as a service–A training and deployment platform based on Docker, OMOP-CDM, and FHIR Web Services. PloS one, 14(10), e0223010.

[3] Sheng, Y. B., & Zhou, L. (2017). Distributed secure quantum machine learning. Science Bulletin, 62(14), 1025-1029.

[4] Singh, S., Sulthana, R., Shewale, T., Chamola, V., Benslimane, A., & Sikdar, B. (2021). Machine-Learning-Assisted Security and Privacy Provisioning for Edge Computing: A Survey. IEEE Internet of Things Journal, 9(1), 236-260.

[5] Aissaoui, Khalid & Ait, Hafsa & Belhadaoui, Hicham & Rifi, Mounir. (2017). Survey on data remanence in Cloud Computing environment. 1-4. 10.1109/WITS.2017.7934624.

[6] Choi, J., Nazareth, D. L., & Ngo-Ye, T. L. (2018). The effect of innovation characteristics on cloud computing diffusion. Journal of Computer Information Systems, 58(4), 325-333..

[7] Klinaku, F., Hakamian, A., & Becker, S. (2021, September). Architecture-based Evaluation of Scaling Policies for Cloud Applications. In 2021 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS) (pp. 151-157). IEEE

[8] Tabrizchi, H., & Kuchaki Rafsanjani, M. (2020). A survey on security challenges in cloud computing: issues, threats, and solutions. The journal of supercomputing, 76(12), 9493-9532.

[9] Gong, C., Li, L., Li, Z., Ji, H., Stern, A., Xia, Y., ... & Zhang, X. (2017). Discovery of intrinsic ferromagnetism in two-dimensional van der Waals crystals. Nature, 546(7657), 265-269.

[10] YIN, Xuefei; ZHU, Yanming; HU, Jiankun. A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. ACM Computing Surveys (CSUR), 2021, 54.6: 1-36.

[11] SHARMA, Neha; JAIN, Vibhor; MISHRA, Anju. An analysis of convolutional neural networks for image classification. Procedia computer science, 2018, 132: 377-384.

[12] B. Piedade, J. Pedro Dias, and F. F. Correia (2020). An empirical study on visual programming docker compose configurations. Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings. Association for Computing Machinery, New York, NY, USA, Article 60, 1–10. DOI:https://doi.org/10.1145/3417990.3420194

[13] WU, Yue, et al. Deep convolutional neural network with independent softmax for large scale face recognition. In: Proceedings of the 24th ACM international conference on Multimedia. 2016. p. 1063-1067.

[14] HSU, Tzu-Ming Harry; QI, Hang; BROWN, Matthew. Measuring the effects of non-identical data distribution for federated visual classification. arXiv preprint arXiv:1909.06335, 2019.

[15] Huang, Y., cai, K., Zong, R., & Mao, Y. (2019, March). Design and implementation of an edge computing platform architecture using docker and kubernetes for machine learning. In Proceedings of the 3rd International Conference on High Performance Compilation, Computing and Communications (pp. 29-32).

[16] Xuefei Yin, Yanming Zhu, and Jiankun Hu. 2021. A Comprehensive Survey of Privacy-preserving Federated Learning: A Taxonomy, Review, and Future Directions. ACM Comput. Surv. 54, 6, Article 131 (July 2021), 36 pages, DOI: https://doi.org/10.1145/3460427.

[17] Liu, P., Xu, X. & Wang, W. Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives. Cybersecurity 5, 4 (2022). https://doi.org/10.1186/s42400-021-00105-6

[18] Hsu, T., Qi, H., Brown, M.: Measuring the Effects of Non-Identical Data Distribution for Federated Visual Classification. CoRR abs/1909.06335 (2019)

Konstantinos D. Stergiou received the BSc degree in Electronic Engineering and the MSc degree in Applied Informatics with specialization in Computer Systems and Network Technologies from School of InformationSciences of University of Macedonia. He is currently a PhD student in the Department of Applied Informatics, School of Information Sciences, University of Macedonia, Greece. His doctoral thesis focuses on "A combinatorial framework of machine learning algorithms and high-speed communication". His additional research interests include computer networks, wireless and mobile communications, multimedia transmission over 5G networks, Big Data Analytics (BDA), Wireless Sensor Networks (WSNs), Internet of Things (IoT), Artificial Intelligence (AI), Network security.

Prof. Konstantinos E. Psannis is currently Associate Professor in Communication Systems and Networking at the Department of Applied Informatics, School of Information Sciences, University of Macedonia, Greece, Director of Mobility2net Research & Development & Consulting JP-EU Lab, member of the EU-JAPAN Centre for Industrial Cooperation and Visiting Consultant Professor, Graduate School of Engineering, Nagoya Institute of Technology, Nagoya 466-8555, Japan. Konstantinos received a degree in Physics, Faculty of Sciences, from Aristotle University of Thessaloniki, Greece, and the Ph.D. degree from the School of Engineering and Design, Department of Electronic and Computer Engineering of Brunel University, London, UK. From 2001 to 2002 he was awarded the British Chevening scholarship. The Chevening Scholarships are the UK government's global scholarship programme, funded by the Foreign and Commonwealth Office (FCO) and partner organisations. The programme makes awards to outstanding scholars with leadership potential from around the world to study at universities in the UK. Prof. Psannis' research spans a wide range of Digital Media Communications, media coding/synchronization and transport over a variety of networks, both from the theoretical as well as the practical points of view. His recent work has been directed toward the demanding digital signals and systems problems arising from the various areas of ubiquitous Big Data/AI-IoT/Clouds and communications. This work is supported by research grants and contracts from various government organisations. Dr. Psannis has participated in joint research works funded by Grant-in-Aid for Scientific Research, Japan Society for the Promotion of Science (JSPS), KAKENHI Grant, The Telecommunications Advancement Foundation, International Information Science Foundation, as a Principal Investigator and Visiting Consultant Professor in Nagoya Institute of Technology, Japan. Konstantinos E. Psannis was invited to speak on the EU-Japan Co-ordinated Call Preparatory meeting, Green & Content Centric Networking (CCN), organized by European Commission (EC) and National Institute of Information and Communications Technology (NICT)/Ministry of Internal Affairs and Communications (MIC), Japan (in the context of the upcoming ICT Work Programme 2013) and International Telecommunication Union. (ITU-founded in 1865), SG13 meeting on DAN/CCN, Berlin, July 2012, amongst other invited speakers. Konstantinos received a joint-research Award from the Institute of Electronics, Information and Communication Engineers, Japan, Technical Committee on Communication Quality, July 2009 and joint-research Encouraging Prize from the IEICE Technical Committee on Communication Systems (CS), July 2011. Dr. Psannis has more than 75 publications in international scientific journals and more than 107 publications in international conferences 22 Book Chapters and 11 Technical Reports and received more than 4700 citations (h-index 30, i10-index 65). Professor Konstantinos has several highly cited papers powered by Web of Science - Clarivate. Dr. Psannis supervises three post-doc students and eight PhD students and more than 150 M.Sc. Thesis. Prof. Konstantinos E. Psannis is serving as an Associate Editor for IEEE Access and IEEE Communications Letters. He is Lead Associate Editor for the Special Issue on Roadmap to 5G: rising to the challenge, IEEE Access, 2019. He is a Guest Editor for the Special Issue on Compressive Sensing-Based IoT Applications, Sensors, 2020. He is a Guest Editor for the Special Issue on Advances in Baseband Signal Processing, Circuit Designs, and Communications, Information, 2020. He is a Lead Guest Editor for the Special Issue on Artificial Intelligence for Cloud Based Big Data Analytics, Big Data Research, 2020. He is TPC Co-Chair at the International Conference on Computer Communications and the Internet (ICCCI 2020), Nagoya Institute of Technology Japan, ICCCI 2020, June 26-29 at Nagoya, Japan, and will be held in 2021 June 25-27, at Nagoya, [http://iccci.org/] and Conference Chair at the World Symposium on Communications Engineering held at University of Macedonia, Thessaloniki, Greece, October 9-11, 2020 and to be held at University of Macedonia, November 25-28, 2021, Thessaloniki, Greece (WSCE 2021 - http://wsce.org/). Professor Psannis is TPC Co-Chair and Session Chair, entitled, Next Generation (NG) 6G-Enabled Artificial Intelligence of Things - Digital twins- and Cobot Intelligence, in 5th World Symposium on Communication Engineering (WSCE 2022) to be held in Nagoya University during September 16-18, 2022. He is Invited Speaker, entitled, Next G-IoT, in the 10th International Conference on Computer and Communications Management, July 29-31, 2022, Okayama University, Japan, invited speaker, entitled, 6G-Enabled Massive Internet of Things, and TPC Co-Chair in the IEEE International Conference on Computer Communication and the Internet (ICCCI 2022), Nagoya Institute of Technology, will be held in June 24-226, 2022, at Nagoya, Japan and invited Speaker, entitled, Massive Internet of Things, Information Technology & Applications Symposium (ITAS), to be held in Chiba, Japan, during July 1-3, 2022. Professor Konstantinos E. Psannis has been included in the list of Top 2% influential researchers globally (prepared by Scientists from Stanford University USA), October 2020 (https://lnkd.in/dhSwdgB) and October 2021 (https://lnkd.in/gCk8FAxu).