Asynchronous Online Federated Learning with Limited Storage

Ioannis Schoinas
Information Technologies Institute (ITI)
Centre for Research & Technology Hellas
Thessaloniki, Greece &
Electrical and Computer Engineering
University of Western Macedonia
Kozani, Greece
i.shoinas@iti.gr

Anna Triantafyllou
Electrical and Computer Engineering
University of Western Macedonia
Kozani, Greece
atriantafyllou@uowm.gr

Anastasios Drosou
Information Technologies Institute (ITI)
Centre for Research & Technology Hellas
Thessaloniki, Greece
drosou@iti.gr

Dimitrios Tzovaras
Information Technologies Institute (ITI)
Centre for Research & Technology Hellas
Thessaloniki, Greece
dimitrios.tzovaras@iti.gr

Panagiotis Sarigiannidis
Electrical and Computer Engineering
University of Western Macedonia
Kozani, Greece
psarigiannidis@uowm.gr

***Abstract*—The Internet of Things (IoT) generates vast amounts of data through sensors, enabling AI to train accurate models and develop Industry 4.0 (I4.0) systems that adapt to real-time changes. Federated Learning (FL), a distributed machine-learning paradigm, allows training across multiple devices while preserving data privacy by keeping the data local. However, classical FL's synchronous aggregation strategy is inefficient for heterogeneous IoT devices. Additionally, IoT edge devices face storage limitations and generate continuous data streams, requiring adaptive online learning. We propose ASynchronous Online Limited Storage Federated Learning (ASOLS-Fed), where edge devices perform online learning on limited local data streams. ASOLS-Fed updates the global model asynchronously, mitigating straggler effects caused by slow or dropped devices. Experiments on benchmark datasets show that ASOLS-Fed converges well, maintaining stronger predictive performance even with limited training samples.**

***Index Terms*—Federated Learning, artificial intelligence, industrial IoT, elastic weight consolidation, asynchronous federated learning, online federated learning**

I. Introduction

Statista [1] forecasts that Internet of Things (IoT) devices worldwide will double from 15.9 billion in 2023 to more than 32.1 billion devices in 2023. The rapid adoption of IoT is fueled among other factors by recent advancements in connectivity, hardware costs and integration with Artificial Intelligence (AI). Traditional centralized AI approaches face issues such as latency, bandwidth, and privacy concerns affecting their effectiveness in IoT environments. Federated Learning (FL) offers a promising solution by leveraging on-device training, reducing communication cost, and safeguarding data privacy. In FL, only the model gradients are transmitted to the server, significantly reducing bandwidth requirements and enhancing security, as no actual data leaves the device.

However, traditional FL approaches assume synchronous aggregation and that devices can adequately store data indefinitely, assumptions that do not hold in the heterogeneous IoT systems domain. In real world applications, edge devices often have limited storage that prevents them from storing streaming data leading to potential loss of knowledge and 'catastrophic forgetting'. Additionally, such systems are also characterized by heterogeneity among the devices and variable network reliability. These challenges highlight the need for an FL framework that supports asynchronous aggregation and effective learning under storage requirements. Even though several works have been focused on addressing the straggler effect with asynchronous communication protocols, the area of limited storage on the device remains underexplored.

This paper addresses these gaps by proposing ASynchronous Online Limited Storage Federated Learning (ASOLS-Fed), a framework designed to handle asynchronous client updates while preserving knowledge due to training on devices with limited storage capabilities. Key contributions of this work include:

- Addressing the limitations of traditional FL in IoT edge devices, which often operate under storage constraints and storing extensive past data is infeasible.
- Incorporating Elastic Weight Consolidation (EWC) [2] and FedProx [3] proximal term with an asynchronous aggregation strategy on the server. EWC preserves parameters critical for the data that are no longer available in the device overcoming catastrophic forgetting, while FedProx addresses data heterogeneity among clients. Asynchronous communication mitigates the straggler effect where slow or dropout devices increase overall training

time and result in under utilization of faster devices.
- Demonstrating through empirical evaluation that the proposed method achieves stable convergence and higher accuracy compared to other asynchronous and online FL methods in conditions of extreme storage limitations.

The remainder of the paper is structured as follows: Section II presents a review of the relevant literature, while Section III presents the proposed algorithm.. Section IV presents the performance evaluation and results. Finally, Section V contains the conclusions of this paper as well future research directions.

II. Related Work

This section presents the literature review of asynchronous and online FL as well as methods that employ EWC [2].

A. Asynchronous Federated Learning

Various studies have been focused on refining and optimizing the asynchronous federated learning approach to make it more effective and practical for real-world applications. In [4] employ an asynchronous aggregation scheme paired with an adaptive weight aggregation method. The Euclidean distance between the global model and the stale model is used to adapt the weight of each client update to reduce the impact of outdated updates. In [5] the authors address the issue of imbalanced learning due to asynchronous updates using a binary weights adjustment method. The method adjusts both sample and parameter weights based on the staleness of the gradients and the amount of data in the device. In [6] an asynchronous FL system is proposed based on two key ideas to regularize the objective function and to utilize a weighting mechanism during aggregation that considers staleness of the update.

B. Asynchronous Online Federated Learning

In ASO-Fed [7] an ASynchronous Online Federated Learning framework is presented which focuses on asynchronous communication with convergence guarantees to maintain optimal performance. The work in ASO-Fed is closely related to IoT and Industrial settings since it considers not only heterogeneous devices and data but also considers the existence of continuous data streams and addresses this by employing an online learning procedure. To ensure an optimal global model, the deviation of the local gradients is constrained with the use of a threshold imposed locally on the devices. Client balance training on new data using a decay coefficient. Aggregation on the server employs feature learning to extract a cross-client feature representation which is used to dynamically adjust the client learning rates. The inspiration for feature learning comes from attention mechanisms, which have been proven to be highly effective in identifying crucial features. The dynamic learning step size which is calculated based on feature representation is used to address stragglers and improve performance. Stragglers are assumed to have a smaller activation rate when using the global model and thus their learning step size is increased.

C. Elastic Weight Consolidation in Federated Learning

In [8] the authors utilize EWC towards estimating the importance of the parameters of the neural network. The authors integrate the Fischer Information Matrix with Bayesian Parameter Importance to estimate the importance of the parameters and guide the parameter-wise elastic weighted averaging on the server improving efficiency in non- Independent and Identically Distributed (IID) data environments. In [9] EWC is integrated in a blockchain-based FL system and aims to retain knowledge learned in a dynamic environment where devices join or leave the training process at different times. The authors in [10], FedAF, proposes EWC to guide the unlearning process of target data without forgetting knowledge regarding non-target data. In [10] the authors employ multi-task learning to fine-tune and adapt the local model on the client side and propose utilizing EWC to prevent forgetting of the global knowledge. The authors in [11] utilize EWC in a similar way to regularize the fine-tuning process of self-supervised learning models. Similar to our approach [12], EWC is used to ensure that the important parameters of the global task are preserved.

Even though EWC has been used mainly for addressing statistical heterogeneity in the clients datasets there is no work utilizing EWC to address storage limitation in FL. Furthermore, the work closer to ours is [12] which also modifies the local objective function of the clients, but in this work the Fischer Information Matrix is used to evaluate the importance of the parameters for the global task instead of the importance of the parameters for the local task of the previous round which is the case in our work.

D. Motivation

The above-mentioned algorithms offer significant improvements, but they do not consider practical client resources. When performing FL in real-world applications, clients may have limited resources and they are most likely unable to store large volumes of data samples. Previous methods assume that all incoming data are available for consecutive rounds however, when data are discarded due to storage limitations, training faces the challenge of catastrophic forgetting, where the model is unable to retain previously learned knowledge. This is particularly problematic in online learning, where the model must be able to integrate knowledge from new data without affecting its performance on data samples that are not available in the current round. For this reason, it is essential to develop methods that consider the limitations of client devices in the asynchronous and online settings.

III. Proposed Method

The following section presents the FL algorithm proposed as well as details about the functionality of the server which is responsible for aggregating the collected client updates asynchronously. The client module trains a local model on streaming data, evaluates the importance of the trained parameters calculating a Fisher Information Matrix (FIM) and sends the update to the server.

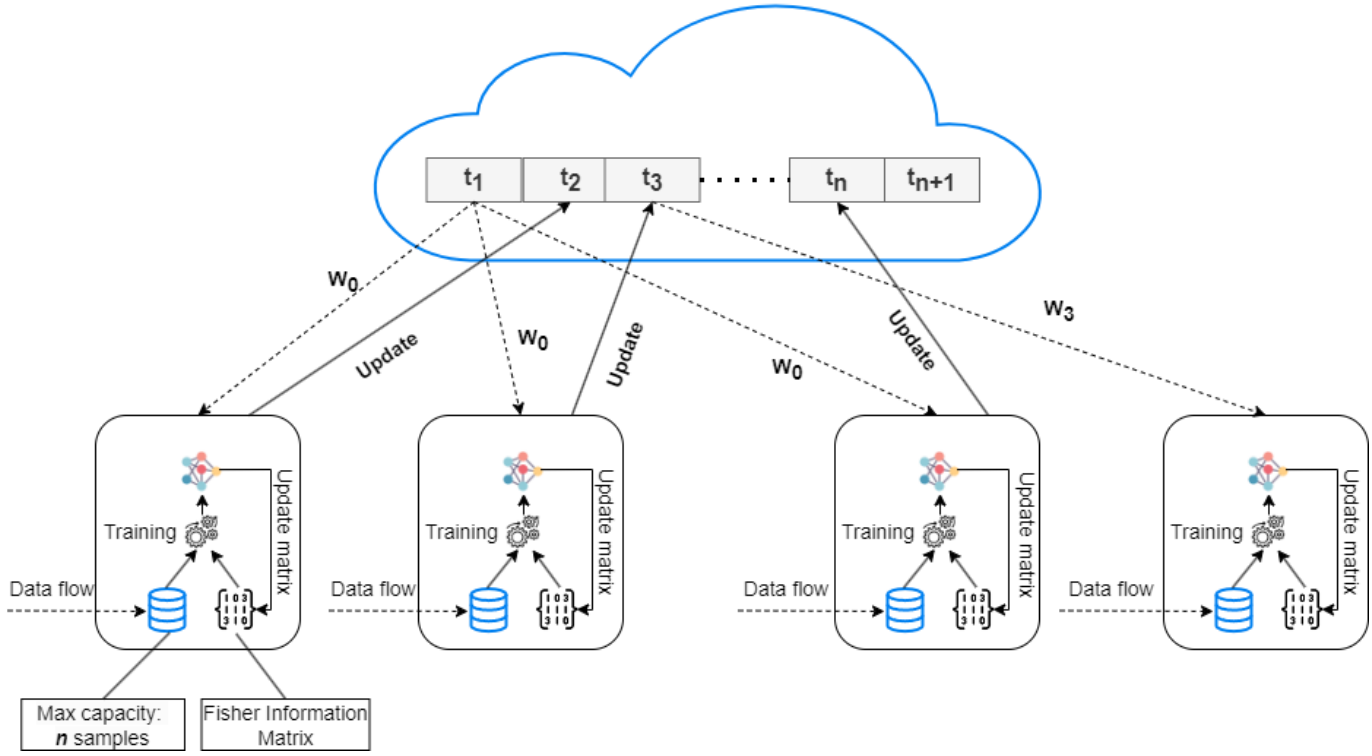

Fig. 1. Asynchronous and Online Federated Learning for Limited Storage.

A. Central aggregation

In asynchronous FL the server aggregates received updates immediately when they arrive without waiting for the rest of the clients to finish their training. This allows the clients to send their updates at different times without stalling the procedure. At each step the server performs a weighted aggregation of the global model with the client's update thus maintaining this way a continuously updated global model. In our setting we utilize an extension of the FedAvg algorithm that supports asynchronous aggregation. The updated global model w^{t+1} is computed as a weighted average of the previous global model w^t and the new local model w_k^{t+1} received from client k:

$$w^{t+1} = w^t - \frac{n_k}{N}\Delta w_k \tag{1}$$

We denote by n the number of training samples used for training of the local model by the client and by N, the total number of samples processed by all the clients so far.

B. Learning on Client

In Online FL clients train their models from streaming data that arrive over time. In such settings, each time a client trains on new data can be viewed as a new task for the model to learn.

EWC is an important method that can be integrated in FL clients to mitigate catastrophic forgetting when the model is trained without access to past data thus leveraging techniques from transfer learning into FL to reduce storage requirements. EWC utilizes the Fisher Information Matrix to identify and preserve parameters that are important for previous samples that are no longer available.

Another significant challenge is data heterogeneity among clients in federated networks. Non-IID data among clients can lead to instability and poor convergence during the training process. FedProx [3] is an extension to the standard Federated Averaging algorithm and is designed to address this issue. The key idea behind FedProx lies in the addition of a proximal term to the objective function of each client. The purpose of the proximal term is to penalize large deviations from the current global model mitigating the impact of heterogeneity.

Combining EWC with FedProx Our proposed FL client module integrates EWC and the proximal term derived from FedProx to address the online learning of heterogeneous FL clients with limited storage. To integrate the EWC with FedProx the local objective function of the client is modified to include both the EWC regularization term and the FedProx proximal term.

The EWC regularization term is defined in Equation 2. Where λ is a hyper-parameter and denotes the regularization strength, which sets how much the importance of parameters of the previous round affect training, F_i denotes the Fisher Information Matrix (FIM) [2], w_i are current parameters and w_i^* are the optimal parameters from the previous round.

$$\frac{\lambda}{2}\Sigma_i F_i (w_i - w_i^*)^2 \tag{2}$$

The FIM is calculated based on the second-order derivatives of the log-likelihood of the client data given the new optimal

parameters. Calculating a full FIM for neural network may result in an extremely large matrix. Therefore, for practical and computational reasons the diagonal approximation of the FIM is used. For the approximation the following formula is used:

$$F_i = \mathbb{E}\left[\left(\nabla_{w_i} L(w)\right)^2\right] \tag{3}$$

In practice, the expectation is approximated by averaging the value of $\left(\nabla_{w_i} L(w)\right)^2$ for every sample currently available. The calculation of the FIM occurs at the end of each round to capture the importance of the parameters for the current data that will not be available in the next rounds.

The FedProx Proximal term is defined in Equation 4. We denote μ the proximal term strength, w the parameters of the model and w_t the parameters of the global model at round t.

$$\frac{\mu}{2}||w - w_t||^2 \tag{4}$$

The combined loss function for client k becomes:

$$S_k(w : w_t) = L_k(w) + \frac{\lambda}{2}\Sigma_i F_i (w_i - w_i^*)^2 + \frac{\mu}{2}||w - w_t||^2 \tag{5}$$

The local objective function $S_k(w : w_t)$ is composed of three main terms.

- The first part, consist of the local loss term $L_k(w)$ which corresponds to the standard local loss on client k data and minimizes the error of the model on these data through optimization.
- The second term regards the EWC regularization term $\frac{\lambda}{2}\Sigma_i F_i (w_i - w_i^*)^2$. This term reduces the change for parameters that are important for the data from the previous rounds. The $(w_i - w_i^*)^2$ calculates the squared difference between the current parameters and the optimal parameters of the previous round indicating how much the parameters changed. The F_i is the importance matrix which represents the importance of each parameter for the data of the previous round and are used to scale the difference accordingly. Finally, the sum of the scaled squared differences are multiplied by $\frac{\lambda}{2}$ producing the final EWC penalty value.
- The third term regards the FedProx regularization term. The term consist of the squared Euclidean distance between the current parameters and the global parameters scaled by $\frac{\mu}{2}$ to produce the final proximal penalty term. μ is a hyper-parameter that controls how much the term affects training.

The proposed objective loss function allows the clients to integrate knowledge learned from samples in the previous rounds and that are no longer available. Also, the proximal term prevents the local model from strongly deviating from the global model, reducing the impact of Non-IID data. After the client calculated the optimal parameters for the specific round and available updates the Fischer Information Matrix and stores it for use in the next round. Fig. 1 illustrates the proposed method, showing data flow, weight updates, and Fisher Information Matrix usage across multiple clients connected to a central server that aggregates client updates asynchronously.

Algorithm 1 Asynchronous and Online FL with Limited Storage Algorithm

1: **Input:** Global model w_0, learning rate η, regularization parameter λ, proximal term μ
2: **Initialize** global model w_0
3: **Procedure at Central Server**
4: **for** global iteration t=1,2,...,T **do**
5: compute w_t:

$$w_t = w_{t-1} - \frac{n_k}{N}\Delta w_k'$$

6: **end for**
7: **Procedure of Local Client k at round t**
8: receive w^t from the server
9: Client k calculates the local update:

$$S_k(w; w_t) = L_k(w) + \frac{\lambda}{2}\Sigma_i F_i (w_i - w_i^*)^2 + \frac{\mu}{2}||w - w_t||^2$$

10:

$$w_k' = w_t - \eta \nabla S_k(w; w_t)$$

11: Client k updates $F_i = \mathbb{E}\left[\left(\nabla_{w_i} L(w_k')\right)^2\right]$ using the current data and the updated model.
12: Client k sends updated model w_k' to the server.

IV. Experimental Setup

To evaluate the proposed algorithm, empirical experiments have been conducted to compare the method against two baseline methods that have been identified in the literature on a benchmark dataset for Image Classification.

Dataset and federated split. Fashion-MNIST regards a collection of 60,000 training images and 10,000 testing images. Each sample consists of a 28x28 grayscale image of a fashion item and a label from 10 classes. The dataset has been divided into 20 devices as in [7]. First the data are sorted based on their label, and the samples of each label are divided into 4 different sizes 2000, 2750, 3250, 4000. Finally each client is assigned 2 different sizes.

Model. The architecture of the neural network for this 10-class classification task is a simple convolutional neural network with two convolutional layers, each followed by ReLU activation. After downsampling with max pooling, it flattens the feature maps and passes them through a fully connected layer to produce the final output.

Configuration. The configurations used in the experiments are shown in Table I. We denote as $\frac{|C_t|}{|C|}$ the client participation rate, η the learning rate, e the number of local epochs and B_c the storage capacity, expressed as fraction of the total number of samples each clients possesses. To simulate the setting of streaming data, we set the device to randomly sample To simulate the setting of streaming data, we set the on-device data arrival to be $ns = \frac{(total\ client\ training\ samples)}{(number\ of\ rounds)}$, which

means that each device $c \in C$ will receive ns data samples in each communication round.

Baselines. In the evaluation experiments, we compare the proposed method ASOLS-Fed, ASynchronous and Online with Limited Storage, algorithm against the standard baseline method for Asynchronous Federated Optimization FedAsync [6], a baseline method for online FL Federated Online Mirror Descent (FedOMD) [13] and the only other asynchronous and online method Asynchronous and Online FL (ASO-Fed) [7].

- Asynchronous Federated Optimization (FedAsync) is a baseline method in federated learning systems that supports asynchronous aggregation employing FedAvg [14] for aggregation on the server.
- FedProx [3] works by adding a proximal term in the objective function of the clients and is a well known method to address heterogeneity in federated learning systems. In our experiments the method is paired with an asynchronous aggregation mechanism on the server similar to the one used in FedAsync. [14] for aggregation on the server.
- Federated Online Mirror Descent (FedOMD) is a baseline method that considers online learning on the clients. The method employs mirror descent algorithm during local training on the device. The method is paired with an asynchronous aggregation mechanism similar to the one used in FedAsync.
- Asynchronous and Online FL (ASO-Fed) method is the only other FL strategy for aggregating data online and asynchronously. This method introduces and utilizes feature learning to calculate a cross-client feature representation based on an attention mechanism and is employed to address potential effects from asynchronous updates.

Simulation. The experiments were run on a single computer with a GPU using Pytorch and the simulation engine from the Flower FL framework [15].

TABLE I
EXPERIMENT CONFIGURATION

Dataset	**Samples**	**Devices**	$\frac{\lvert C_t \rvert}{\lvert C \rvert}$	η	e	$\lvert B_c \rvert$
Fashion-MNIST	70,000	20	10%	$1e^{-3}$	1	0.3

Results.
Empirical results shown in Fig. 2, demonstrate that our proposed asynchronous aggregation mechanism achieves convergence of the model and higher accuracy compared to all the other methods used. Compared to the baselines, ASOLS-Fed greatly outperforms the other algorithms achieving around to 60% accuracy, while the rest do not exceed 46%. Specifically ASO-Fed, the only strategy that considers asynchronous aggregation of continuous streaming local data achieves a maximum accuracy of 28%. All the algorithms exhibit great instability suggesting that they struggle to converge under the extreme storage limitations. Our proposed approach is the most stable and continuously improving method among the three.

Fig. 3 also validates the above observations. Even though the rest algorithms in the initial few rounds demonstrate greater reduction of loss, ASO-Fed demonstrates greater reduction of final loss. In addition the loss curve of the other algorithms stabilize at a very high point indicating that their global models do not continue to learn. In contrast the loss curve of the proposed method exhibits a steady downward trend that suggest that the model continues to learn effectively.

Experimental results utilizing the Fashion-Mnist dataset, indicate that existing algorithms struggle to perform in the case of asynchronous and online training with limited storage. The proposed algorithm achieves convergence of the global model indicating that combining EWC and FedProx is effective in improving both learning stability and performance in scenarios with storage limitations.

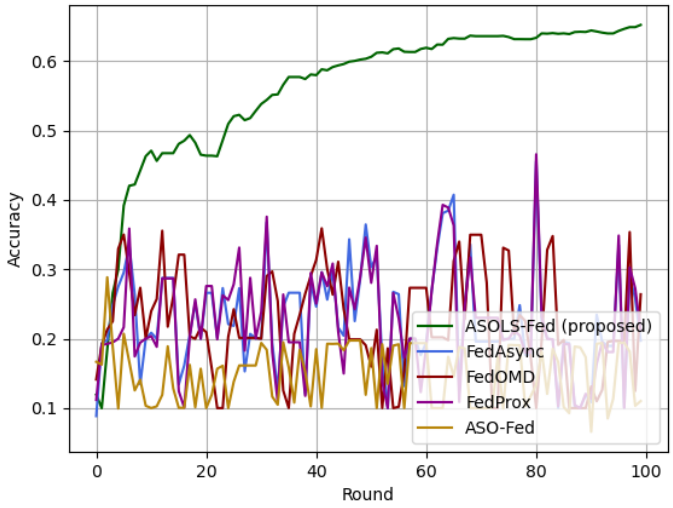

Fig. 2. Accuracy of proposed ASOLS-Fed on Fashion-MNIST dataset.

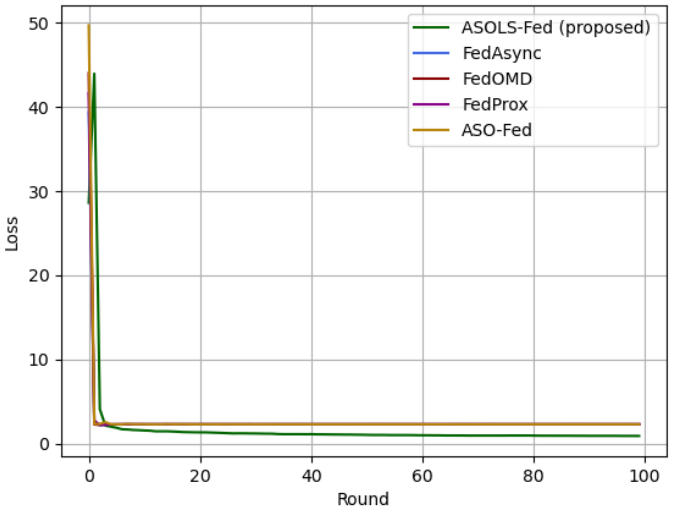

Fig. 3. Loss of proposed ASOLS-Fed on Fashion-MNIST dataset.

V. Conclusion and future steps

In this work, we extend research in the field of limited on-device storage and streaming data, an area that has yet to be fully explored in FL systems. We then presented the implementation of ASOLS-Fed, which is an asynchronous and online FL method for devices with limited storage. The method consists of an updated objective function that retains knowledge from data that is no longer available on the device. Empirical results demonstrate that the method is able to converge in cases where only a small portion of data is available on the device in each round while state-of-the-art methods do not succeed.

As industries increasingly adopt decentralized and distributed systems that align with IEC 61499 standard for distributed industrial automation, the application of FL systems becomes more relevant. FL, as a distributed system across heterogeneous devices, shares the same principals defined by the IEC 61499. The algorithm presented in this work can be integrated into industrial systems that abide by the standard, enabling more intelligent, adaptive, and robust control systems. In future work, a practical implementation of this method in an actual industrial setting to evaluate the synergies between the proposed method and a standardized distributed control architecture will take place.

Regarding future work, the current considerations for improvement of the proposed method are twofold. First, we plan to incorporate an adaptive aggregation mechanism that considers staleness when weighting the collected updates. Staleness of local updates is used extensively in the literature to improve the aggregation mechanism. By giving more weight to recent updates the aggregated global model reflects better the latest data distribution. In addition, several other aspects of the IoT domain can be considered during aggregation including device capabilities, connectivity, and energy constraints. Adaptive aggregation can alleviate these challenges that arise from these limitations and enhance performance of the proposed method. Secondly we plan to incorporate sample level selection techniques, such as ODE [16], to improve storage efficiency further. These approaches are beneficial in cases of large volume of streaming data, such as that produced by IoT sensors, allowing for efficient use of storage resources and potentially enhance training by selecting a more representative dataset while removing corrupt, redundant or even malicious data.

Acknowledgment

This work has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No. 101057083 – Zero-SWARM

References

[1] Statista, "Statista market insights, internet of things - worldwide," 2023. [Online]. Available: https://www.statista.com/outlook/tmo/internet-of-things/worldwide

[2] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

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

[4] Q. Wang, Q. Yang, S. He, Z. Shi, and J. Chen, "Asyncfeded: Asynchronous federated learning with euclidean distance based adaptive weight aggregation," *arXiv preprint arXiv:2205.13797*, 2022.

[5] A. Khalid, Z. Aziz, and M. S. Fathi, "Privacy shield: A system for edge computing using asynchronous federated learning," *Scientific Programming*, vol. 2022, no. 1, p. 7465640, 2022.

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

[7] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-iid data," in *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020, pp. 15–24.

[8] J. Bai, A. Sajjanhar, Y. Xiang, X. Tong, and S. Zeng, "Fedewa: Federated learning with elastic weighted averaging," in *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2022, pp. 1–8.

[9] K. Kopparapu and E. Lin, "Fedfmc: Sequential efficient federated learning on non-iid data," *arXiv preprint arXiv:2006.10937*, 2020.

[10] Y. Li, C. Chen, X. Zheng, and J. Zhang, "Federated unlearning via active forgetting," *arXiv preprint arXiv:2307.03363*, 2023.

[11] A. Ovsianas, J. Ramapuram, D. Busbridge, E. G. Dhekane, and R. Webb, "Elastic weight consolidation improves the robustness of self-supervised learning methods under transfer," *arXiv preprint arXiv:2210.16365*, 2022.

[12] Z. Ma, Y. Lu, W. Li, and S. Cui, "Efl: elastic federated learning on non-iid data," in *Conference on Lifelong Learning Agents*. PMLR, 2022, pp. 92–115.

[13] A. Mitra, H. Hassani, and G. J. Pappas, "Online federated learning," in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 4083–4090.

[14] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[15] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, H. L. Kwing, T. Parcollet, P. P. d. Gusmão, and N. D. Lane, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020.

[16] C. Gong, Z. Zheng, F. Wu, Y. Shao, B. Li, and G. Chen, "To store or not? online data selection for federated learning with limited storage," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 3044–3055.