Asynchronous Federated Learning: Methods and applications

YiFan Liu

Mathematics and Applied Mathematics, Shanghai Lixin University of Accounting and Finance, Shanghai, China

YFLiu@stu.sqmc.edu.cn

Abstract. Federated Learning (FL) is a distributed alternative to traditional machine learning frameworks that computes a global model on a centralized aggregation server according to the parameters of local models, which can address the privacy leakage problem caused by collecting sensitive data from local devices. However, the classic FL methods with synchronous aggregation strategies, in many cases, shall suffer from limitations in resource utilization due to the need to wait for slower devices (stragglers) to aggregate during each training epoch. In addition, the accuracy of the global model can be affected by the uneven distribution of data among unreliable devices in real-world scenarios. Therefore, many Asynchronous Federated Learning (AFL) methods have been developed on many occasions to improve communication efficiency, model performance, privacy, and security. This article elaborates on the existing research on AFL and its applications in many areas. The paper first introduces the concept and development of FL, and then discusses in detail the related work and main research directions of AFL, including dealing with stragglers, staleness, communication efficiency between devices, and privacy and scalability issues. Then, this paper also explores the application of AFL in different fields, especially in the fields of mobile device edge computing, Internet of Things devices, and medical data analysis. Finally, the article gives some outlook on future research directions and believes that it is necessary to design efficient asynchronous optimization algorithms, reduce communication overhead and computing resource usage, and explore new data privacy protection methods.

Keywords: Asynchronous Federated Learning, Secure Aggregation, Differential Privacy, Distributed Machine Learning.

1. Introduction

Federated learning (FL) [1, 2] is a distributed alternative to traditional training methods, in which models are obtained by distributing training data on mobile devices and aggregating updates and sharing of local computations, so it has better performance in representing and generalizing large amounts of training data from different users. In terms of specific application practice, the relevant research of early researchers in the field of Parameter Servers[3, 4, 5] solved the management and coordination of model parameters of each participating device in FL and provided support for the development of FL. While parallel training on different devices to process large sample datasets, FL should also be able to meet the legal requirements for privacy protection in different regions: some countries/regions, such as the European Economic Area (EEA), have clear data protection and privacy rules under their General Data Protection Regulation (GDPR). Similarly, the USA, China, India, Australia, and Russia, as well as other countries, have their own specific data protection and privacy laws. There are some methods with which the protection of privacy can be guaranteed in federated learning, including Differential Privacy [6] and Secure Aggregation [7].

Several disadvantages become apparent when using classical FL on resource-constrained devices [8]: (1) Unreliability. The presence of heterogeneous devices poses a challenge for the aggregation server being delayed for waiting for local gradients updated from each device. However, these devices can go offline unexpectedly due to their inherent unreliability. (2) The aggregating efficiency is reduced. In each training epoch, the faster device is forced to wait for an outdated local model from the slower device (the straggler). This delay is caused by the dual factors of device heterogeneity (differences in resources between devices) and data heterogeneity (uneven distribution of training data among devices). (3) Communication efficiency. Current node selection algorithms are inefficient, often resulting in the selection of multiple capable devices that are rarely selected to participate. (4) Security and privacy breaches. Classical FL methods are vulnerable to various security threats, such as poisoning and backdoor attacks. Additionally, there are concerns about privacy due to potential data leakage during training.

To address the challenges of device unreliability, reduced aggregation efficiency, and low resource utilization, asynchronous federated learning (AFL) emerges as a promising solution. In AFL, the central server starts global model aggregation immediately after receiving the local model. Since AFL can ignore abnormally offline devices, concerns about device unreliability are alleviated. AFL improves aggregation efficiency by eliminating the need to wait for slow devices for local model uploads before aggregation. AFL also improves the utilization of computing resources across heterogeneous systems by allowing devices with different operating efficiencies to train their respective local models at their own pace.

Therefore the choice of whether to use synchronous or asynchronous algorithms is the most basic decision, which will directly affect the design framework of the overall model [9]. Although there have been many developments and applications of asynchronous training in earlier deep learning research, synchronous large-batch training has been the focus of many researchers in federated learning, in contrast to asynchronous federated learning that has been overlooked in many potentially more suitable applications and research situations. Especially when plenty of devices participate in computing, and plenty of stragglers may occur at the moment, making global synchronization difficult [10]. (This is because availability and completion time vary from device to device, more specifically by the device's limited computing power and different battery times.)

There is already a precedent for asynchronous training in traditional distributed Stochastic Gradient Descent (SGD) tasks [11, 12], in which researchers use regularized local optimization to complete SGD. In AFL, Xie et al.[10] Based on the combination of the above asynchronous training method and FL, proposed FedAysnc, an asynchronous federated optimization scheme, to finish aggregation tasks by regularizing local optimization and adopting the non-blocking update of the global model. FedAysnc had a profound impact on subsequent research on AFL.

In this article, we will carefully sort out and analyze some important works under the existing AFL framework and its practical application in the fields of the Internet of Things (such as industry 4.0, smart home, healthcare, autonomous driving, etc.), and put forward exploratory suggestions for possible cross-application directions in the future.

2. Related Work

FL is born out of distributed machine learning, that is, efficient machine learning processing on multiple computing nodes, which faces the challenges of large data volume and complex computing tasks, so it has received extensive attention and research in recent years.

Early distributed learning research mainly focused on how to extend traditional machine learning algorithms to distributed environments to solve problems such as large data volumes and limited computing resources. One of the most representative is Zinkevich, et al[13], who discusses how to achieve distributed linear regression by stochastic gradient descent (SGD). On this basis, further workers began to explore the optimization strategy of distributed machine learning at a deeper level. For example, Dean, et al[14] mentioned the training method of deep networks under distributed systems. However, these methods also have some problems, such as uneven data distribution, large communication costs, and difficulty in ensuring data security. In response to these questions, FL is proposed and provides a new way of thinking.

In 2016, Google first introduced the concept of FL, the core idea of which is to let devices train with local data under the premise of guaranteed privacy of data, and then govern the sharing model update. This reduces the need for data transmission while protecting user privacy. Based on H.B. McMahan, et al.[2] first proposed the FedAvg algorithm and elaborated on the working principle and advantages of FL, Kairouz, et al.[15] further discussed the confronting challenges and future developments of FL.

In detail, there are four main problems faced by FL [2, 15]:

i. **Communication efficiency issues**: Each round of training requires the transfer of model parameters between the server and the local computer, which may lead to a large communication overhead. A representative solution to the problem in this direction was proposed by Sattler [16] to reduce the communication load through a thinned gradient update method.

ii. **Non-IID Data problem**: Since data is uploaded and obtained locally on every device, FL often needs to deal with non-independent homogeneous data. Zhao, et al. [17] proposed an FL algorithm that can share a small amount of data to optimize non-IID data.

iii. **Security and privacy issues**: Although the original intention of FL is to protect the data privacy of users[2], there may still be potential security threats in practice, for example, attackers may infer users' private data by analyzing the information received by the central server; Or break updates to the global model by interfering with local model training. Bonawitz, et al. [7] discussed these potential threats in their study and proposed several countermeasures.

iv. **System design and implementation issues**: How to effectively design and implement an efficient and scalable federated learning system is also a major challenge. To solve this problem, Google developed TensorFlow Federated (TFF) and shared its design philosophy and usage in the official blog [18].

3. Asynchronous Federated Learning

Asynchronous federated learning (AFL) is an essential extension of FL, which is mainly used to solve asynchronous problems caused by device participation inconsistency, network latency, and limited computing power. In the past few years, remarkable research progress has been made in this field. Early asynchronous machine learning research focused on problems in single-machine, multi-threaded, or clustered environments, such as Recht, et al. [19] who proposed the HOSWILD! algorithm, which achieves efficient parallel SGD through lock-free programming. After the introduction of federated learning, due to the uncertainty of device availability and network conditions, asynchronous algorithms are more critical in this scenario.

3.1. Stragglers

In traditional federated learning, every device is required to wait for other slower devices (the stragglers) to complete local training before uploading gradients to a central server, which limits the speed of training. To address this problem, Xie, et al. [10] introduced an AFL protocol with an averaging algorithm called FedAsync, which allows non-blocking communication between the client and server by using an asynchronous communication mechanism. FedAysnc decouples training from communication so that a client can independently train locally and upload gradients, thereby improving parallel efficiency. Specifically, in the gradient update, the client will calculate the gradient according to its local data and the current global model, upload it to the server, and gain the latest global parameters from the server. Such global parameters are fused by the server periodically handing the gradient of the client and using weighted average or more complex methods. Sprague, et al. [20] leverage FedAsync-like protocols in geospatial applications to train global models asynchronously and additionally allow new devices to join midway. For the same application scenario, Zhou, et al. [21] also proposed an AFL algorithm based on practical applications, namely image-based geolocation using state-of-the-art convolutional neural networks (DNNs). These results lay the foundation for using FL on large-scale ocasstions, constantly updating machine learning models with unknown codings.

3.2. Staleness

However, one major problem with FedAsync, and some of the protocols it derives, is that in AFL systems, due to different devices updating the model and training at different times, the gradients received by the server may be based on an outdated global model. This means these gradients may not be optimal and could even lead to a decline in model performance [6].

To reduce the impact of gradient staleness, many researchers have done a lot of work. In [22], the researchers restrain the number of devices trained simultaneously in the AFL system. A throttled size cache with a weighted averaging algorithm has been deployed on the server to reduce the negative influence of model staleness. The experimental results support the increase in convergence speed and model accuracy. In other respects, by introducing a parameter that explains staleness, the staleness can be reduced and the weight of the latest local model increased during aggregation. Several papers have taken this approach. In [10], a time-weighted aggregation method is introduced that shifts the weight of the lately updated local model when aggregating at both simple and deep levels. Experiments of neural networks (CNN and LSTM) illustrate an increase in the convergence speed and accuracy of the global model as well as another time-based algorithm is introduced in [23].

Also, in [22], a weighted aggregation algorithm based on stale caching is proposed. In [24], attenuation coefficients with a similar effect are proposed, balancing previous and current models. Devices that have more data or less communication ability are compensated by dynamic learning step size. Experimental results on three real datasets show that their scheme converges quickly, and a double-weighted gradient update strategy is proposed, in which the global model is separated into multiple branches, and the aggregation process can be transformed into a branch-weighted merging process. And dynamically adjust aggregate weights based on the training accuracy of all devices in order to protect the global model from overfitting which is caused by uploading gradients too frequently.

3.3. Communication-Efficiency

In some cases derived from AFL, the server may be overwhelmed by receiving too many local updates gained from a large number of devices [6]. At the same time, Chen, et al. [25] found that gradient staleness can have a considerable negative impact on the accuracy of asynchronous optimization, while synchronous optimization has no effect on gradient staleness but is dragged down by stragglers, so they proposed the concept of introducing alternate workers to improve synchronous optimization algorithms. Further, Nishio & Yonetani [26] propose a new FL method (FedCS) that can filter clients by their wireless channel state, computing power (e.g., whether CPUs or GPUs can be freed up to update the model), and data resource sizes related to related data resources, thereby alleviating inefficiencies and efficiently performing optimization tasks. Then a semi-asynchronous federated learning protocol, combining synchronous and asynchronous training, is proposed by Wu et al. [6]. This protocol incorporates a client selection method that decouples the central server and the selected client to reduce the average turn time. Additionally, it utilizes a lag tolerance mechanism model distribution to address the trade-off between faster convergence and lower communication overhead. As a result, the Semi-Asynchronous protocol is ideal for unreliable and constrained training environments.

Another idea to improve communication efficiency is to cluster devices into tiers based on their response latency [27]. Faster layers are responsible for faster convergence tasks, while slower layers are required to help improve model accuracy. In addition, the scheme adopts a compression algorithm based on polyline coding to improve communication efficiency. And experiments on multiple datasets and models confirm that their model has lower communication costs and higher accuracy.

3.4. Scalability and Privacy

Besides, the study of scalability is also an important direction in AFL to reduce staleness and improve communication efficiency. [28] claims a model called Fedbuff, in which a dedicated buffer containing a certain number of model updates is designed on the aggregation server, and the convergence of the model is mathematically guaranteed. FedBuff addresses scalability and privacy issues in AFL by buffering asynchronous aggregations, specifically by aggregating updates from K clients in a safe buffer before updating the global model. This framework greatly improves the training efficiency, and thanks to the high concurrent training, the framework is less sensitive to the diminishing return problem caused by increasing the number of clients than traditional FL methods. In addition, FedBuff' is the first asynchronous federated optimization framework compatible with secure aggregation and global user differential privacy, outperforming synchronous federated learning at low privacy settings and still being quite competitive at high privacy settings. In addition, [29] introduces a model that leverages a secure buffer on the server. This model includes a secure aggregation protocol, specifically designed to prevent the server from gaining any information about local updates.

4. Applications of Asynchronous Federated Learning

AFL has found adoption in diverse application scenarios due to its efficient and adaptable training procedure across heterogeneous devices while ensuring the privacy of local training data. This makes AFL a highly researched and promising area with significant potential across multiple industries and applications.

4.1. Mobile Device Edge Computing

The portable and anytime-and-anywhere nature of mobile devices often faces challenges such as unstable network connections and battery consumption. In this case, AFL is very useful, and an early application example is Google's Gboard, which uses FL to improve the next-word prediction feature of its keyboard application [1].

In addition to Google, other companies and organizations are applying or researching AFL for various use cases:

- ❿ **Apple** uses federated learning techniques in its iOS ecosystem, specifically for improving Siri’s voice recognition.
- ❿ **OpenMined** is a collaborative open-source community dedicated to the research, development, and promotion of tools that enable secure, privacy-preserving, and value-aligned artificial intelligence. This community also actively explores the use of AFL as part of its initiatives.
- ❿ **Nvidia** also incorporates federated learning into its software stack with a solution known as Nvidia Clara. It is mainly used for collaborative AI model training across multiple hospitals while preserving patient privacy.
- ❿ **Owkin**, a start-up, uses federated learning in healthcare for better prediction and understanding of diseases. Their product Owkin Connect applies these techniques for researchers to collaborate without sharing raw data.

These examples indicate that AFL is increasingly being recognized and adopted for its advantages, especially in scenarios where privacy and resource optimization are critical.

4.2. Internet of Things(IoT) Device

IoT devices tend to be scattered and work at different times. For example, in a smart grid, a collection device may upload electricity usage data at different times. In this environment, AFL can provide a more efficient solution. In [30], the application of AFL in the field of Internet of Things is mainly divided into 6 categories:

- **Industry 4.0**: Optical Character Recognition (OCR) for the labels, smart and automatic Incoming Quality Control (IQC), and smart Process Quality Control (PQC).
- **Healthcare**: Smart wearable devices are used to monitor the health status of patients, such as heartbeat, blood pressure, and glucose level
- **Smart Home**: Wake-Up-Word speech recognition and Automatic Speech Recognition (ASR)
- **Smart City**: Various IoT devices that enable city managers to control physical objects in real-time and provide intelligent information to citizens from all aspects of life will be a task that requires a fairly high level of privacy assurance
- **Autonomous Driving**: With the deepening of the development of vehicle automatic control technology, the sensor on the vehicle will collect more data in the process of the vehicle's time and space changes, and the potential risk of privacy leakage will increase. The large data transmission and limited network bandwidth can lead to communication overhead, which in turn may result in inaccurate real-time response of the driving system to spatial changes. However, AFL is capable of effectively addressing this issue.
- **Metaverse and VR**: As the metaverse is formed to connect local devices to people, there will be a lot of interaction and measurement data, which will inevitably come with the risk of privacy exposure. Using AFL balances accuracy and speed of interaction with privacy.

4.3. Medical Data Analysis

Different from the IoT framework built by medical devices. Local data centers in hospitals and healthcare facilities often hold large amounts of patient data, but this data is often not shared directly due to privacy protection and data security needs. However, AFL can enable healthcare organizations to share the results of learning algorithms while maintaining data privacy. According to the latest results [31, 32, 33, 34], AFL methods can be initially applied as an auxiliary tool in the field of disease identification and diagnosis.

5. Conclusions

AFL is a new method of training machine learning models in parallel, which is mainly used in distributed environments and allows participating devices to complete their model updates at different times. Although this approach has shown advantages in dealing with communication delay, device failure, and data privacy protection, the problems of balancing gradient staleness and model stability, as well as server data overload, have not been fully solved.

Considering the advantages and limitations of AFL, we believe that the future research direction mainly includes the following parts: 1) Design efficient asynchronous optimization algorithms to solve the problem of global model instability. You may want to consider how to balance the weights of each device and how to handle differences in data distribution between devices. 2) Research how to reduce communication overhead and the use of computing resources while ensuring model performance. This may require the introduction of compression and bandwidth-saving techniques such as model compression, gradient compression, etc. 3) Explore new ways to protect data privacy to ensure the security of user data. This may include further research and improvements to cryptographic techniques such as differential privacy, secure multi-party computation, and more.

At the same time, AFL has a wide range of application prospects. For example, in mobile device edge computing scenarios, AFL can be used for real-time intelligent analysis and prediction. In IoT devices, it is possible to handle the large amount of data uploaded by devices that are distributed everywhere; In medical data analysis, through AFL, hospitals and medical institutions can share the results of learning algorithms while keeping data private, thereby enabling cross-institutional knowledge sharing. However, how to effectively apply AFL to practical problems also needs to solve a series of key problems, such as how to build suitable machine learning models, how to deal with non-independent and homogeneous data between devices, and how to evaluate and ensure the quality of learning results.

In general, AFL, as an emerging machine learning method, although still in the development stage, has shown great potential. With the development and improvement of related technologies, AFL will play an increasingly important role in the future.

References

[1] Konečný, Jakub, et al. "Federated learning: Strategies for improving communication efficiency." *arXiv preprint arXiv:1610.05492* (2016).
[2] McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." *Artificial intelligence and statistics*. PMLR, 2017.
[3] Ho, Qirong, et al. "More effective distributed ml via a stale synchronous parallel parameter server." *Advances in neural information processing systems* (2013).
[4] Li, Mu, et al. "Scaling distributed machine learning with the parameter server." *11th USENIX Symposium on operating systems design and implementation (OSDI 14)*. 2014.
[5] Li, Mu, et al. "Communication efficient distributed machine learning with the parameter server." *Advances in Neural Information Processing Systems*27 (2014).
[6] McMahan, H. Brendan, et al. "Learning differentially private recurrent language models." *arXiv preprint arXiv:1710.06963* (2017).
[7] Bonawitz, Keith, et al. "Practical secure aggregation for privacy-preserving machine learning." Proceedings *of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 2017.
[8] Wu, Wentai, et al. "SAFA: A semi-asynchronous protocol for fast federated learning with low overhead." *IEEE Transactions on Computers* 70.5 (2020): 655-668.
[9] Bonawitz, Keith, et al. "Towards federated learning at scale: System design." *Proceedings of machine learning and systems* 1 (2019): 374-388.
[10] Xie, Cong, Sanmi Koyejo, and Indranil Gupta. "Asynchronous federated optimization." *arXiv preprint arXiv:1903.03934* (2019).
[11] Lian, Xiangru, et al. "Asynchronous decentralized parallel stochastic gradient descent." *International Conference on Machine Learning*. PMLR, 2018.
[12] Zheng, Shuxin, et al. "Asynchronous stochastic gradient descent with delay compensation." *International Conference on Machine Learning*. PMLR, 2017.
[13] Zinkevich, Martin, et al. "Parallelized stochastic gradient descent." *Advances in neural information processing systems* 23 (2010).
[14] Dean, Jeffrey, et al. "Large scale distributed deep networks." *Advances in neural information processing systems* 25 (2012).
[15] Kairouz, Peter, et al. "Advances and open problems in federated learning." *Foundations and Trends® in Machine Learning* 14.1–2 (2021): 1-210.
[16] Sattler, Felix, et al. "Robust and communication-efficient federated learning from non-iid data." *IEEE transactions on neural networks and learning systems* 31.9 (2019): 3400-3413.
[17] Zhao, Yue, et al. "Federated learning with non-iid data." *arXiv preprint arXiv:1806.00582* (2018).
[18] https://www.tensorflow.org/federated
[19] Recht, Benjamin, et al. "Hogwild!: A lock-free approach to parallelizing stochastic gradient descent." *Advances in neural information processing systems* 24 (2011).
[20] Sprague, Michael R., et al. "Asynchronous federated learning for geospatial applications." *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Cham: Springer International Publishing, 2018.
[21] Sprague, Michael R., et al. "Asynchronous federated learning for geospatial applications." *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Cham: Springer International Publishing, 2018.

[22] Zhou, Chendi, et al. "TEA-fed: time-efficient asynchronous federated learning for edge computing." Proceedings of the 18th ACM International Conference on Computing Frontiers. 2021.

[23] Shi, Guomei, et al. "HySync: Hybrid federated learning with effective synchronization." *2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. IEEE, 2020.

[24] Chen, Yujing, et al. "Asynchronous online federated learning for edge devices with non-iid data." *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020.

[25] Chen, Jianmin, et al. "Revisiting distributed synchronous SGD." *arXiv preprint arXiv:1604.00981*(2016).

[26] Nishio, Takayuki, and Ryo Yonetani. "Client selection for federated learning with heterogeneous resources in mobile edge." *ICC 2019-2019 IEEE international conference on communications (ICC)*. IEEE, 2019.

[27] Chai, Zheng, et al. "Fedat: A communication-efficient federated learning method with asynchronous tiers under non-iid data." *ArXivorg* (2020).

[28] Nguyen, John, et al. "Federated learning with buffered asynchronous aggregation." *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022.

[29] So, Jinhyun, et al. "Secure aggregation for buffered asynchronous federated learning." *arXiv preprint arXiv:2110.02177* (2021).

[30] Zhang, Tuo, et al. "Federated learning for the internet of things: Applications, challenges, and opportunities." *IEEE Internet of Things Magazine* 5.1 (2022): 24-29.

[31] Sakib, Sadman, et al. "On COVID-19 prediction using asynchronous federated learning-based agile radiograph screening booths." *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021.

[32] Yaqoob, Muhammad Mateen, et al. "Modified Artificial Bee Colony Based Feature Optimized Federated Learning for Heart Disease Diagnosis in Healthcare." *Applied Sciences* 12.23 (2022): 12080.

[33] Sakib, Sadman, et al. "Asynchronous federated learning-based ECG analysis for arrhythmia detection." *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE, 2021.

[34] Khan, Muhammad Amir, et al. "Asynchronous Federated Learning for Improved Cardiovascular Disease Prediction Using Artificial Intelligence." *Diagnostics* 13.14 (2023): 2340.