6

Analysis of Privacy Preservation Enhancements in Federated Learning Frameworks

Z. Anastasakis[1], S. Bourou[1], T. H. Velivasaki[1], A. Voulkidis[1], and D. Skias[2]

[1]Synelixis Solutions S.A., Greece
[2]Netcompany-Intrasoft S.A., Greece
E-mail: anastasakis@synelixis.com; bourou@synelixis.com; terpsi@synelixis.com; voulkidis@synelixis.com; Dimitrios.Skias@netcompany-intrasoft.com

Abstract

Machine learning (ML) plays a growing role in the Internet of Things (IoT) applications and has efficiently contributed to many aspects, both for businesses and consumers, including proactive intervention, tailored experiences, and intelligent automation. Traditional cloud computing machine learning applications need the data, generated by IoT devices, to be uploaded and processed on a central server giving data access to third parties raising privacy and data ownership concerns. Federated learning (FL) is able to overcome these privacy concerns by enabling an on-device collaborative training of a machine learning model without sharing any data over the network. However, model sharing can also potentially reveal sensitive information. Therefore, federated learning needs additional privacy-preserving techniques to enable fully private machine learning model sharing and training. In this chapter, privacy-preserving techniques for federated learning are studied. In addition, a comparative analysis of state-of-the-art federated learning frameworks against privacy-preserving techniques is presented. The analysis comprises the identification of main advantages and disadvantages for eight

FL frameworks as well as the investigation of the frameworks under criteria related to their FL features and privacy preservation options.

Keywords: Federated learning, privacy preserving, Internet of Things, artificial intelligence, machine learning.

6.1 Introduction

Artificial intelligence (AI) produces insights by automatically identifying patterns and detecting anomalies on data collected or generated using IoT sensors and other devices. Machine learning (ML) is almost everywhere nowadays, from small wearable devices and smartphones to powerful supercomputers ensuring fast and accurate data analysis. Moreover, IoT devices generate a great amount of data every day and, thus, raise significant concerns about privacy and ownership of the collected or generated data.

Traditional machine learning applications require their training and testing data to be located in a central cloud server. This raises privacy and data ownership concerns. Furthermore, IoT devices are already capable of processing a vast amount of data due to their powerful hardware specifications, making it possible for local data processing and analysis. Thus, edge computing is witnessing great interest especially after the emergence of 5G.

Nevertheless, data privacy is the most fundamental objective regarding data access and processing. This has led to the elaboration of strict data privacy legislations such as the Consumer Privacy Bill of Rights in the U.S. and the European Commission's General Data Protection Regulation (GDPR). For example, Articles 5 and 6 of the GDPR state that data collection and storage should be restricted to only what is user-consented and decidedly indispensable for processing.

To address privacy issues, Google [1] introduced federated learning (FL), a specific approach in edge computing. Federated learning is able to overcome the privacy concerns that emerge in a central cloud-based architecture by enabling an on-device collaborative training of a machine learning model without sharing any data over the network. This is achieved by initializing the training of a global machine learning model on a central server for a few iterations to obtain some initial weights. These model weights are then sent to the participants (data owners), which use their own resources to locally train the machine learning model. After training, each client sends its own updated weights to the server, which is responsible to aggregate the weights from all the different clients and produce a new global model. This process

is repeated for several iterations until the global model reaches a certain desired accuracy level or reaches the limit set for the number of iterations. Federated learning aims to train an ML model privately by sharing model parameters (weights of the model) than sharing the data itself. This feature enables machine learning models to run on local and private data. However, model sharing can also potentially reveal sensitive information. Therefore, FL needs additional privacy-preserving techniques to enable fully private machine learning model sharing and training. Differential privacy (DP) and secure multiparty computation and homomorphic encryption (HE) constitute the most popular privacy-preserving techniques for FL systems.

6.2 Privacy-preserving Federated Learning

6.2.1 Federated learning frameworks

Several open-source federated learning frameworks have been developed to apply distributed learning on decentralized data but also to enhance privacy and security. Google proposed TensorFlow Federated [2], an open-source framework for federated learning and other computations on decentralized data. Another open-source federated learning framework is PySyft, which was introduced by OpenMined [3]. PySyft is suitable for research in FL and allows the users to perform private and secure deep learning. PySyft is also integrated into PyGrid [4], a peer-to-peer platform for federated learning and data privacy, which can be used for private statistical analysis on the private dataset as well as for performing FL across multiple organization's datasets. WeBank's AI department introduced FATE (federated AI technology enabler) [5], an open-source framework that supports FL architectures and secure computation of various machine learning algorithms. FATE is an industrial-grade framework mostly oriented toward enterprise solutions. The authors in [6] presented Flower, a friendly open-source federated learning framework that is ML framework agnostic and provides higher-level abstractions to enable researchers to experiment and implement on top of a reliable stack. Another promising open-source federated learning framework is Sherpa.ai, which is presented in [7] and incorporates federated learning with differential privacy. Sherpa.ai results as a combination of machine learning applications in a federated manner with differential privacy guidelines. FedML [8] is an open-source federated learning framework and benchmarking tool for federated machine learning. FedML supports three computing paradigms: on-device training for edge devices, distributed computing, and single-machine

simulation. FedML promotes diverse algorithmic research due to the generic API design and the comprehensive reference baseline implementations. Another well-known open-source federated learning framework is the PaddleFL [9]. In PaddleFL, researchers can easily replicate and compare different federated learning algorithms while they can easily be deployed in large-scale scenarios. Leaf [10] is a modular benchmarking framework for federated learning with applications including federated learning, multi-task learning, meta-learning, and on-device learning. OpenFL [11] is another open-source federated learning framework for training ML algorithms using the data-private collaborative learning paradigm of FL. OpenFL works with machine learning pipelines built on top of TensorFlow and PyTorch and is easily customizable to support other machine learning and deep learning frameworks. NVIDIA FLARE [12] is a domain-agnostic, open-source, and extensible SDK for federated learning, which allows porting existing ML/DL workflow to federated settings and supports common privacy preservation techniques. In the following sub-sections, a more extended analysis is given for each framework. In Section 6.3.2, a thorough comparative analysis on these federated learning frameworks is presented toward the scope of IoT-NGIN.

6.2.2 Privacy preservation in federated learning

While FL is resilient and resolves, up to a point, data governance and ownership issues, it does not guarantee security and privacy by design. A lack of encryption can allow adversaries to abduct personally identifiable data directly from the processing nodes or interfere with the communication process, expose network vulnerabilities, and perform attacks. In addition, the decentralized nature of the data complicates data handling and curation. Moreover, in the case where algorithms running on the nodes are not encrypted, or the updates are not securely aggregated, the possibility of data leakage grows. Additionally, the algorithms can be tampered with, reconstructed, or get stolen (parameter inference), which can be strictly forbidden for most applications. Federated learning can be vulnerable to various backdoor threats (bug injection, inference, and model attacks) on different processing steps. Therefore, additional measures are essential to protect data from adversarial attack strategies such as data poisoning and model poisoning attacks. In Table 6.1, three major attacks against the dataset with their description and a basic example for each case are listed, while in Table 6.2, algorithmic-based attacks are presented.

Table 6.1 Various attacks against the data in a federated learning system.

Attacks against the dataset	*Description*	*Example*
Re-identification attack	Recover an individual's identity by exploiting similarities to other datasets and exposing the data characteristics.	Exploiting similarities between data distributions and actual values from other datasets in which the same individual is contained.
Dataset reconstruction attack	Determine an individual's characteristics from the training process without accessing the data itself.	Using multiple statistical information (probabilities, distributions, etc.) to get data points that correspond to a single individual.

Table 6.2 Major attacks against algorithms that run in a federated learning system.

Attacks against algorithm	*Description*	*Example*
Adversarial attack	Manipulation of the input to an algorithm with the goal of altering it, most often in a way that makes the manipulation of the input data impossible to detect by humans.	Compromising the computation result by introducing malicious training examples (model poisoning).
Model-inversion/reconstruction attack	Derivation of information about the dataset stored within the algorithm's weights by observing the algorithm's behavior.	Using generative algorithms to recreate parts of the training data based on algorithm parameters.

In general, the goal of an adversary during data poisoning is to alter the data according to their preferences. This can be done by ingesting a mixture of clean and false data into the training flow. For example, in [13], the result of an image classification learning task can be vulnerable to a data poisoning attempt by a mislabeling or a false-labeling operation. Wang refers to different defense mechanisms from simple data management to more sophisticated and robust approaches. Data sanitization is a rather basic defense, while pruning (removing neurons in a network) seems more reliable. Nonetheless, the pruning technique raises concerns regarding privacy-preserving in federated learning. In [14], [15], and [16], some legitimate defenses for these attacks are proposed, although backdoor attacks become stronger and more adjective.

Model poisoning attack refers to partial or full model replacement during training. The authors in [17] and [18] describe possible attacks and argue about various defenses (SMC, DP, etc.). Generative adversarial networks (GANs) [19] can be one of the most vicious threats in federated learning. The authors in [20] exploit defenses against GAN-based attacks and present the anti-GAN framework to prevent adversaries from learning the real distribution of the training data. On the other hand, GANs in [21] are utilized as a defense mechanism against adversarial attacks in federated learning systems. As a conclusion, FL is vulnerable to various attacks and great attention must be given to the defense mechanisms and tools; otherwise, it will not be possible for an FL system to fulfill its privacy-preserving objectives.

6.2.3 State-of-the-art approaches in privacy-preserving federated learning

Although FL enables on-device machine learning, it does not guarantee security and privacy. The fact that the private data are not shared with the central server is for sure an advantage; yet, there are ways to extract private information from the data. After the shared model is trained on the user's device based on its own private data, the trained parameters (model weights) are sent to the central server, and through an aggregation mechanism, the global model is composed. During the model transfer, it is possible for an adversary to extract information about the private data from those trained parameters. For example, in [22], the authors indicate that it is possible to extract sensitive text patterns, e.g., the credit card number, from a recurrent neural network that is trained on users' data. Therefore, additional mechanisms are required to protect data disclosure from attack strategies, which are subject to privacy-preserving methods in FL. The major approaches that can be employed in FL for data protection are differential privacy, homomorphic encryption, and secure multiparty computation.

Differential privacy (DP) is a method that randomizes part of the mechanism's behavior to provide privacy [23], [24]. The motivation behind adding randomness (either Laplacian or Gaussian) into a learning algorithm is to make it impossible to reveal data patterns or insights that correspond either to the model and the learned parameters or to the training data. Therefore, the DP provides privacy against a wide range of attacks (e.g., differencing attacks, linkage attacks, etc.) [25]. The method of introducing noise to the data can result in great privacy but may compromise accuracy. Therefore, there is a tradeoff between applying differential privacy and achieving a high

level of model accuracy. However, the authors in [25] present a method, which applies privacy-preserving without sacrificing accuracy.

Another privacy-preserving technique is the secure multiparty computation (SMC), a well-defined cryptographic-based technique that allows a number of mutually suspicious parties to jointly compute a function before training a model while preserving the privacy of the input data [26], [27]. In the case of ML applications, the function can be the model's loss function at training, or it could be the model itself during inference. The challenge of applying SMC on a large-scale distributed system is the communication overhead, which increases significantly with the number of participating parties.

Homomorphic encryption [28] secures the learning process by applying computations (e.g., addition) on encrypted data. Specifically, an encryption scheme is characterized as homomorphic, when standard operations can be applied directly to the cypher data, in such a way that the decrypted result is equivalent to performing analogous operations to the original encrypted data [29], [30]. For machine learning methods, homomorphic encryption can be applied when training or inference is performed directly on encrypted data (cyphertexts). In scenarios, where large mathematical functions are implemented to cyphertext space, a major bottleneck of homomorphic encryption emerges. The properties of homomorphic encryption schemes confront several limitations, related to encryption performance.

Alternative hybrid approaches that combine SMC with DP and account dishonest participants exist. In [31], authors confront the inference risk of SMC and the low accuracy that DP presents due to the noise injection by combining them. Furthermore, they propose a tunable trust parameter attribute by additively HE, which considers many trust scenarios. HybridAlpha method [32] establishes a multi-input functional encryption (public-key cryptosystem) scheme to prevent inference attacks on SMC. HybridAlpha introduces a trusted third party to derive public keys to parties who intend to encrypt their data before training. Wang [33] presented HDP: a differential private framework for vertical federated learning (cross-silo). HDP-VFL does not rely on HE or on third-party collaborators to assure data privacy; therefore, it is easy to implement and is rather fast. Chain-PPFL [34] can achieve privacy-preserving without compromising the model accuracy using SMC and DP in a "trust-but-curious" way. The proposed communication mechanism constructs a serial chain frame that transfers masked information between participants. In addition, chain-PPFL does not require encryption or obfuscation before transmitting information because

it uses the P2P encrypted secure transmitted channel, thus requiring less resources. The authors in [35] present a fully decentralized federated learning process (BlockFlow) as a more resilient approach against adversarial and inference attacks. BlockFlow adopts blockchains as computational platforms and, contrarily to other methods, does not require a central trusted part. Unlike other methods, there is no need for a centralized test dataset and different parties share DP models with each other.

6.2.4 Comparison of federated learning frameworks considering privacy preservation

Considering the extensive analysis presented above, for the FL methods/tools and the privacy-preserving approaches, comparative analysis for federated learning frameworks is conducted and presented in this section. The comparison refers to the FL frameworks analyzed in Section 6.2.1 and for which the main benefits and drawbacks are briefly presented in Table 6.3.

The comparison among the FL frameworks listed in Table 6.3 is based on the following criteria:

- **Criterion 1:** This criterion is based on basic federated learning features. The operating system support, the federated learning categorization, e.g., if it supports cross-silo or cross-device setups, which machine learning and deep learning libraries (TensorFlow, PyTorch, etc.) do the framework supports and if there is a Federated attack simulator.
- **Criterion 2:** This includes three computing paradigms; the standalone simulation that gives the possibility for a user to apply FL scenarios in simulation; the distributed computing capability that shows if an FL framework is capable of performing in a distributed environment where participants are different devices; the capability of on-device training for IoT and other mobile devices that normally have limited hardware resources.
- **Criterion 3:** If FL frameworks include common FL algorithms and configurations like federated average [36], decentralized FL, vertical FL, and split learning [37].
- **Criterion 4:** An essential characteristic for an FL framework is the existence of privacy-preserving mechanisms and also what types of privacy-preserving methods are supported by the frameworks. In cases where privacy-preserving techniques are not presented, the FL framework must give the capability to integrate such mechanisms.

Table 6.3 Main pros and cons of the federated learning frameworks.

FL framework	*Main pros*	*Main cons*
NVIDIA FLARE	1. It supports training in real-life scenarios 2. It supports a high number of clients 3. It is customizable, supporting the integration of ML models implemented via state-of-the-art ML frameworks, such as TensorFlow and PyTorch 4. It supports privacy reserving methods, such as percentile privacy, homomorphic encryption, and MPC, which can also be combined 5. It comes with good documentation and large community	1. It does not support on-device training 2. Its performance drops as the number of parties increases 3. It does not support heterogeneous clients
FATE	1. Production ready 2. High-level interface 3. Provides many FL algorithms 4. Containerized − Kubernetes support	1. It does not establish any differential privacy algorithms 2. Its high-level interface relies too much on a poorly documented domain-specific language 3.It does not have a core API; so developers must modify the source code of FATE to implement custom FL algorithms 4. It does not use GPUs for training
Flower	1. Provides a template API that allows users to easily transform ML pipelines to FL 2. Very easy to develop and ML framework-agnostic 3. Supports a great number of clients 4. It is really customizable	1. It does not have any differential privacy algorithms 2. It is relatively new and the support community is not that big 3. It does not provide secure aggregation
PySyft & PyGrid	1. Rather easy to use 2. It has the largest community of contributors among the FL frameworks	1. PySyft is only for one server and one client (duet) and can run only in simulation mode 2. PyGrid is needed in order to develop real FL scenarios
TFF	1. It integrates seamlessly with existing TensorFlow ML models	1. As of the time of writing, it can be used only in the simulation mode because it does not support

Table 6.3 (Continued.)

FL framework	*Main pros*	*Main cons*
	2. It is easy to use due to its familiarity	the federated operation mode 2. The data used for training cannot be loaded from the remote worker itself but must be partitioned and transferred through the central server
Sherpa.ai	1. Relatively easy to use because of the Jupiter notebooks, etc. 2. Implements FL algorithms and it is easy to customize them	1. Poor documentation 2. Small community with only seven contributors 3. The project's repository is not active (4+ months after the latest update) 4. Can run only in the simulation mode 5. Limited applicable scenarios
FedML	1. On-device training for edge devices including smartphones and Internet of Things (IoT) 2. Distributed computing 3. Growing community 4. Multi-GPU training support	1. No privacy-preserving techniques are applied. Only a secure aggregation technique is implemented 2. The multiple available modules for different situations might lead to drawbacks and create overheads
PaddleFL	1. It provides a high-level interface for some basic and well-known FL aggregators and implements a differentially private algorithm 2. It provides enough privacy-preserving methods such as DP, MPC, and secure aggregation	1. It is fairly difficult to use it because it uses a little-known DL platform 2. It has poor documentation and has a small community – only 12 contributors 3. It is not compatible with other frameworks and that is a major drawback
Leaf	1. It provides some basic federated learning mechanisms such as the federated averaging aggregator 2. It is modular and adaptive 3. It enables reproducible science	1. It does not provide any benchmark for preserving privacy in an FL setting 2. It does not offer as much official documentation or tutorials 3. Limited federated learning capabilities; it is mainly for production purposes

- **Criterion 5:** In order for an FL framework to be flexible and adaptive, documentation, tutorials, and community support are significant.

Table 6.4 Federated learning framework comparison ($\frac{1}{2}$).

FL framework	*NVIDIA FLARE*	*FATE*	*Flower*	*PySyft + PyGrid*	*TFF*
Standalone simulation	Yes	Yes	Yes	Yes	Yes
Distributed computing	Yes	Yes	Yes	Yes	Yes
On-device training (mobile, IoT)	No	No	Yes – depends on the network	Yes	No
FedAvg	Yes	Yes	Yes	Yes	Yes
Decentralized FL	Yes	No	Yes	No	No
FedNAS	No	No	Yes	No	No
Vertical federated learning		Yes	Yes	No	No
Split learning	Yes	No	Yes	Yes	No
Privacy-preserving methods	Yes (HE, percentile privacy, exclude Vars, DP)	No	(Yes) (PATE – implemented in IoT-NGIN, known as FedPATE [39])	Yes (SMC, HE)	Yes (DP)
DP noise type	No	No	Yes	No	No
Adaptive differential privacy	No	No	No	No	No
Subsampling methods to increase privacy	No	No	No	No	No
Documentation and community support	Large	Partial – mostly in Chinese	Yes Growing rapidly	Yes	Yes
Secure aggregation	Yes	Yes	Future implementation		No

- **Criterion 6:** Secure aggregation [38] algorithm implementation to further enhance privacy.

Table 6.5 Federated learning framework comparison (2/2).

FL framework	*Sherpa.ai*	*FedML*	*PaddleFL*	*OpenFL*
Standalone simulation	Yes	Yes	Yes	Yes
Distributed computing	No	Yes	Yes	Yes
On-device training (mobile, IoT)	No	Yes	No	No
FedAvg	Yes	Yes	Yes	Yes
Decentralized FL	No	Yes	Yes	Yes
FedNAS	No	Yes	No	No
Vertical federated learning	No	Yes	Yes	Yes
Split learning	No	Yes	Yes	Yes
Privacy-preserving methods	Yes (DP)	No	Yes (SMC, DP)	Yes (SMC, DP)
DP noise type	Yes	No	Yes	Yes
Adaptive differential privacy	Yes	No	Yes	Yes
Subsampling methods to increase privacy	Yes	No	Yes	Yes
Documentation and community support	Yes	Stable	Partial	Partial but growing
Secure aggregation	No	Future implementation	Yes	Yes

- **Criterion 7:** Nowadays, training on GPUs especially for deep learning tasks is essential. Especially for limited hardware resources on devices, GPUs have shown remarkable computation capabilities compared to CPUs.
- **Criterion 8:** All the FL frameworks in comparison are open-sourced but with different licenses and therefore of different usage limitations.
- **Criterion 9:** More general properties and characteristics of the FL frameworks. To be more specific, an FL framework must be easy to use, adaptive, preserve interoperability, flexibility, and privacy.

The characteristics of each FL framework against the nine identified criteria are tabulated in Tables 6.4 and 6.5.

Based on the tables above, privacy-preserving methods are available for NVIDIA FLARE, Flower, PySyft & PyGrid, TensorFlow Federated, Sherpa.ai, PaddleFL, and OpenFL; however, the exact privacy-preserving methods supported differ across the FL frameworks. On the other hand, Flower, PySyft & PyGrid, and FedML support on-device training.

6.3 Conclusion

This chapter has provided a critical review of federated learning theory, tools, and algorithms in relation to providing string privacy protection guarantees for individual nodes' data and models. We have explained why federated learning is necessary for privacy-preserving machine learning with many clients on decentralized data. We have proceeded with providing an extensive comparative analysis over open-source FL tools, mainly under the prism of privacy preservation, providing guidance for experimentation, according to underlying application requirements. Considering the outcomes of this analysis, three FL frameworks (NVIDIA FLARE, and Flower with PATE and TFF) have been selected for applying privacy-preserving federated learning in pilot use cases. Specifically, the project considers NVIDIA FLARE in "Traffic Flow & Parking Prediction" and "Crowd Management" use cases in the Smart City Living Lab, as well as in "Crop diseases prediction & irrigation precision" in the Smart Agriculture Living Lab. Moreover, Flower (integrated with PATE) has been considered in training ML models for classification tasks, relevant to the scope of the "Crop diseases prediction & irrigation precision" use case, as well. In addition, research on training ML models in large-scale settings for tabular data classification in the scope of network attack detection has been considered for the Smart Energy Living Lab. Future work aims at enhancing privacy in state-of-the-art open source FL frameworks, suitable for researching FL under real settings in the context of ensuring data privacy across the integrated IoT, edge, and cloud continuum.

Acknowledgement

The work presented in this document was funded through H2020 IoT-NGIN project. This project has received funding from the European Union's Horizon 2020 research and innovation program under Grant Agreement No. 957246.

References

[1] Google Research, "Federated Learning: Collaborative Machine Learning without Centralized Training Data," Google, 2017. [Online]. Available: https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.W428W7321

[2] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage and J. Roselander, "Towards Federated Learning at Scale: System Design," in Proceedings of Machine Learning and Systems, Palo Alto, CA, USA, 2019. W428W7321

[3] T. Ryffel, A. Trask, M. Dahl, B. Wagner, J. Mancuso, D. Rueckert and J. Passerat-Palmbach, "A Generic Framework for Privacy Preserving Deep Learning," in NeurIPS 2018 Workshop on Privacy-Preserving Machine Learning, Montréal, 2018. W428W7321

[4] J. Konecný, H. B. McMahan, D. Ramage and P. Richtárik, "Federated Optimization: Distributed Machine Learning for On-Device Intelligence.," ArXiv, 2016. W428W7321

[5] Y. Liu, T. Fan, T. Chen, Q. Xu and Q. Yang, "FATE: An Industrial Grade Platform for Collaborative Learning With Data Protection," Journal of Machine Learning Research, 2021. W428W7321

[6] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. Hei Li, T. Parcollet, P. Porto Buarque de Gusmão and N. D. Lane, "Flower: A Friendly Federated Learning Research Framework," arXiv, 2020. W428W7321

[7] N. Rodríguez-Barroso, G. Stipcich, D. Jiménez-López, J. A. Ruiz-Millán, E. Martínez-Cámara, G. González-Seco, M. V. Luzón, M. A. Veganzones and F. Herrera, "Federated Learning and Differential Privacy: Software tools analysis, the Sherpa.ai FL framework and methodological guidelines for preserving data privacy," Information Fusion, vol. 64, pp. 270-292, 2020. W428W7321

[8] C. He, S. Li, J. So, X. Zeng, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu, X. Zhu, J. Wang, L. Shen, P. Zhao, Y. Kang, Y. Liu, R. Raskar, Q. Yang and Annavaram, "FedML: A Research Library and Benchmark for Federated Machine Learning," CoRR, 2020. W428W7321

[9] D. Dong, W. Zhang and Q. Jing, "PaddleFL," GitHub, [Online]. Available: https://github.com/PaddlePaddle.[Accessed2022].W428W7321

[10] S. Caldas, S. M. Karthik Duddu, P. Wu, T. Li, J. Konečný, H. B. McMahan, V. Smith and A. Talwalkar, "LEAF: A Benchmark for Federated Settings," Workshop on Federated Learning for Data Privacy and Confidentiality, 2019. W428W7321

[11] P. Foley, M. J. Sheller, B. Edwards, S. Pati, W. Riviera, M. Sharma, P. N. Moorthy, S.-h. Wang , J. Martin, P. Mirhaji, P. Shah and S. Bakas, "OpenFL: the open federated learning library," Physics in Medicine & Biology, 2022. W428W7321

[12] H. R. Roth, Y. Cheng, Y. Wen, I. Yang, Z. Xu, Y.-T. Hsieh, K. Kersten, A. Harouni, C. Zhao, K. Lu, Z. Zhang, W. Li, A. Myronenko, D. Yang, S. Yang, N. Rieke, A. Quraini and C. Chen, "NVIDIA FLARE: Federated Learning from Simulation to Real-World," in nternational Workshop on Federated Learning (NeurIPS 2022), New Orleans, USA, 2022. W428W7321

[13] H. Wang, K. Sreenivasan , S. Rajput, H. Vishwakarma, S. Agarwal, J.-y. Sohn, K. Leew and D. Papailiopoulos, "Attack of the Tails: Yes, You Really Can Backdoor Federated Learning," in 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada, 2020. W428W7321

[14] K. Liu, B. Dolan-Gavitt and S. Garg, "Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks," in International Symposium on Research in Attacks, Intrusions, and Defenses, 2018. W428W7321

[15] J. Steinhardt, P. Wei Koh and P. Liang, "Certified Defenses for Data Poisoning Attacks," in 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 2018. W428W7321

[16] S. Wang, X. Wang, S. Ye, P. Zhao and X. Lin, "Defending DNN Adversarial Attacks with Pruning and Logits Augmentation," in 2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP), 2018. W428W7321

[17] Z. Sun, P. Kairouz, A. Theertha Suresh and . H. B. McMahan, "Can You Really Backdoor Federated Learning?," in 2nd International Workshop on Federated Learning for Data Privacy and Confidentiality at NeurIPS 2019, 2019. W428W7321

[18] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin and V. Shmatikov, "How To Backdoor Federated Learning," in Proceedings of the 23rdInternational Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy, 2020. W428W7321

[19] I. Goodfellow, J. Pouget-Abadie , M. Mirza , B. Xu , D. Warde-Farley , S. Ozair , A. Courville and Y. Bengio, "Generative Adversarial Networks," Commun. ACM, vol. 63, no. 11, p. 139–144, 2020. W428W7321

[20] X. Zhang and X. Luo, "Exploiting Defenses against GAN-Based Feature Inference Attacks in Federated Learning," arXiv, 2020. W428W7321

[21] S. Taheri, A. Khormali, M. Salem and J.-S. Yuan, "Developing a Robust Defensive System against Adversarial Examples Using Generative Adversarial Networks.," Big Data Cogn. Comput., vol. 4, no. 11, 2020. W428W7321

[22] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos and D. Song, "The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks," in USENIX Security 2019, 2019. W428W7321

[23] M. Abadi , A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar and L. Zhang, "Deep Learning with Differential Privacy," in Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16), Vienna, Austria, 2016. W428W7321

[24] F. McSherry and K. Talwar, "Mechanism Design via Differential Privacy," in 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07), 2007. W428W7321

[25] C. Dwork and A. Roth, The Algorithmic Foundations of Differential Privacy, vol. 9, Now Publishers Inc., 2014. W428W7321

[26] Q. Yang, Y. Liu, T. Chen and Y. Tong, "Federated Machine Learning: Concept and Applications," ACM Trans. Intell. Syst. Technol., vol. 10, no. 2, pp. 2157-6904, 2019. W428W7321

[27] C. Zhao, S. Zhao, M. Zhao, Z. Chen, C.-Z. Gao, H. Li and Y.-a. Tan, "Secure Multi-Party Computation: Theory, practice and applications," Information Sciences, vol. 479, pp. 357-372, 2019. W428W7321

[28] C. Lefebvre, "On data," Journal of Pidgin and Creole Languages, vol. 115, no. 2, 2000. W428W7321

[29] L. T. Phong, Y. Aono, T. Hayashi, L. Wang and S. Moriai, "Privacy-Preserving Deep Learning via Additively Homomorphic Encryption," IEEE Transactions on Information Forensics and Security, vol. 13, no. 5, pp. 1333-1345, 2018. W428W7321

[30] C. Gentry, "A Fully Homomorphic Encryption Scheme," Dissertation, 2009.W428W7321

[31] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang and Y. Zhou, "A Hybrid Approach to Privacy-Preserving Federated

Learning," in Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security (AISec'19), London, United Kingdom, 2019. W428W7321

[32] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar and H. Ludwig, "HybridAlpha: An Efficient Approach for Privacy-Preserving Federated Learning," in Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security (AISec'19), London, United Kingdom, 2019. W428W7321

[33] C. Wang, J. Liang, M. Huang, B. Bai, K. Bai and H. Li, "Hybrid Differentially Private Federated Learning on Vertically Partitioned Data," ArXiv, 2020. W428W7321

[34] Y. Li, Y. Zhou, A. Jolfaei, D. Yu, G. Xu and X. Zheng, "Privacy-Preserving Federated Learning Framework Based on Chained Secure Multiparty Computing," IEEE Internet of Things Journal, vol. 8, no. 8, pp. 6178-6186, 2021. W428W7321

[35] V. Mugunthan, R. Rahman and L. Kagal, "BlockFLow: An Accountable and Privacy-Preserving Solution for Federated Learning," ArXiv, 2020. W428W7321

[36] I. Mironov, "Rényi Differential Privacy," in 2017 IEEE 30th Computer Security Foundations Symposium (CSF), 2017. W428W7321

[37] N. Papernot and I. Goodfellow, "Privacy and machine learning: two unexpected allies?," cleverhans-blog , 2018. [Online]. Available: http://www.cleverhans.io/privacy/2018/04/29/privacy-and-machine-learning.html.W428W7321

[38] C. He, M. Annavaram and S. Avestimehr, "FedNAS: Federated Deep Learning via Neural Architecture," in CVPR 2020 Workshop on Neural Architecture Search and Beyond for Representation Learning, 2020. W428W7321

[39] IoT-NGIN, "D3.3 - Enhanced IoT federated deep learning/ reinforcement ML," H2020 - 957246 - IoT-NGIN Deliverable Report, 2022.W428W7321