A Survey for Federated Learning Evaluations: Goals and Measures

Di Chai†* Leye Wang‡* Liu Yang† Junxue Zhang† Kai Chen† Qiang Yang†
† Department of CSE, HKUST, Hong Kong, China
{dchai,lyangau}@cse.ust.hk,jzhangcs@connect.ust.hk,{kaichen,qyang}@cse.ust.hk
‡ Peking University, Beijing, China leyewang@pku.edu.cn
* Equal Contribution

***Abstract*—Evaluation is a systematic approach to assessing how well a system achieves its intended purpose. Federated learning (FL) is a novel paradigm for privacy-preserving machine learning that allows multiple parties to collaboratively train models without sharing sensitive data. However, evaluating FL is challenging due to its interdisciplinary nature and diverse goals, such as utility, efficiency, and security. In this survey, we first review the major evaluation goals adopted in the existing studies and then explore the evaluation metrics used for each goal. We also introduce *FedEval*, an open-source platform that provides a standardized and comprehensive evaluation framework for FL algorithms in terms of their utility, efficiency, and security. Finally, we discuss several challenges and future research directions for FL evaluation.**

***Index Terms*—Introduction and Survey, Evaluation, Security and Privacy Protection, Efficiency, Performance measures**

I. Introduction

Federated learning (FL) is an emerging technology that aims to address data privacy concerns in real-world applications. Data privacy has become an increasingly severe issue today as more and more real-life applications are driven by cross-domain private data. Companies that fail to protect users' privacy may face a hefty fine. For instance, the Federal Trade Commission (FTC) fined Facebook $5 billion to force new privacy measures [1], and Luxembourg's National Commission for Data Protection (CNPD) imposed a record-breaking fine of $887 million on Amazon for misusing customer data for targeted advertising purposes [2]. In this situation, federated learning (FL) has received many research and industry interests as a new paradigm of privacy-preserving machine learning [3]. Rather than collecting massive user data for model training, FL sets up a joint training scenario in which the clients' devices participate in model training under a joint agreement with a central authority. The client devices only upload specific model parameters to the cloud server for aggregation. Recently, FL has appeared on the Gartner 'Hype Cycle for Data Science and Machine Learning' at the innovation trigger stage, indicating the importance and widespread acceptance of the FL technique [4].

Evaluation plays a critical role in designing various FL algorithms and systems, owing to the need for rigorous performance assessment, providing comparative analysis between different algorithms, ensuring robustness across diverse environments, and identifying limitations for further improvement. Conceptually, evaluation is a systematic method to investigate how well a model, framework, or system meets its intended purposes. Essentially, two fundamental questions must be answered during the evaluation process: (1) *what are the **goals** that need to be achieved?*, and (2) *how can the ability to achieve these goals be **measured**?* For example, in the case of image classification, achieving *high accuracy* is a primary goal; to measure accuracy, many research works have evaluated their models on the well-known public dataset, ImageNet, leading to the creation of the ImageNet leaderboard.[1] In this article, we aim to provide clarity on the two evaluation issues for FL systems, namely goals and measurements. By doing so, we hope to assist researchers in conducting FL system evaluations in a more comprehensive and accessible manner and contribute to the healthy development of the entire FL community.

The evaluation of FL is challenging as it is a multi-objective and cross-domain research topic that leverages techniques from machine learning, distributed systems, cryptography, *etc*. The typical FL process usually contains three steps [3]: 1) all parties perform local updates using private data; 2) all parties send the locally updated parameters to a third-party server, which will perform an aggregation on the received updates to produce the global updated parameter; 3) all parties download the global parameter to replace the local one and continue the next round of training. Generally, studies from the machine learning domain aim to improve the model utility, studies from distributed systems aim to improve efficiency, and privacy-preserving researchers mainly focus on privacy protections. Existing studies [5, 6] have shown that these targets are not independent objectives and exhibit substantial interrelation. Enhancing one target usually has negative impacts on the other targets [5, 6]. For example, increasing the number of local updates before global synchronization (*i.e.*, reducing global synchronization frequency) can improve communication efficiency but harm model accuracy. With more local updates, a model trained on heterogeneous, non-identical-and-independent distributed (non-IID) data across clients will deviate further from the global optimum, which is known as the non-IID issue [7]. This illustrates the trade-off between communication efficiency and model utility, and we will discuss more trade-offs between different targets

[1] https://paperswithcode.com/sota/image-classification-on-imagenet

in Section II-D. Appropriate and comprehensive evaluations can guide our future research directions by fully revealing the tradeoffs between different objectives, as well as the theoretical upper and lower bounds on the performance of different methods under varying conditions (*e.g.*, different data distributions).

Appropriate evaluation is crucial to not only promote the healthy development of FL, but the evaluations themselves can enable further applications.

- **Evaluation as Quality Control.** Real-world applications prefer FL models with excellent performance. FL models with significant issues, such as private data leakage, are unsuitable for practical applications. Therefore, FL system evaluation serves as a quality control measure for FL models before they can be used in real-world scenarios.
- **Evaluation for Incentive Design.** FL system evaluation can also work with incentive mechanisms during federated training. Specifically, the contribution of each data provider needs to be quantitatively evaluated, and then the payoff of the federation can be allocated fairly according to these evaluations [8, 9].
- **Evaluation as Online Verification.** Existing FL studies often make assumptions, particularly for security-related assumptions such as semi-honest behavior. However, these assumptions may not always hold in practice. FL system evaluation can serve as an online verification tool to ensure that FL participants adhere strictly to the predefined protocol.

In contrast, the inappropriate evaluation will produce biased assessments, and the undiscovered limitations in FL algorithms or systems will damage real-world applications. For example, undiscovered privacy vulnerabilities will not only leak data providers' privacy but also decrease people's trust and willingness to further contribute data in the federated systems; FL algorithms untested under different data distributions may achieve poor model quality in applications as the data distribution in real-world applications can be highly heterogeneous [7, 10–16]; FL systems not evaluated on real-world network conditions may fail to achieve expected efficiency in applications due to the limited bandwidth and high latency in real-world applications.

In this survey, we first summarize the evaluation goals for FL. We then introduce various well-studied metrics and procedures for measuring these evaluation goals. Furthermore, we will present an open-source platform for FL evaluation called *FedEval*.[2] This platform can aid researchers in implementing a standardized and comprehensive FL evaluation procedure with ease. Finally, we will discuss the challenges and future directions for FL system evaluations.

Necessity of our evaluation survey The fast development of FL has motivated many survey studies to summarize the advances and challenges of FL. Specifically, existing FL survey studies [3, 17–20] introduced the concepts and applications of FL, [21] emphasized the non-IID studies, [22–24] focused on the security and privacy in FL, [25] focused on the incentive design, [26–29] emphasized the internet of things (IoT) scenario, [30–32] summarized the medical and health case applications of FL, [33] and [34] introduced the application of smart city and graph learning, respectively. Existing FL surveys focus on elaborating the new techniques and applications of FL, and the survey study on the evaluation of FL has been lacking. However, the evaluation of FL is a complicated problem since FL is a cross-domain topic that consists of machine learning, distributed systems, and privacy-preserving techniques, making the evaluation of FL contains many targets, *e.g.*, utility, robustness, privacy preservation, *etc*. An unreasonable evaluation process will cause an unjustified assessment of FL methods and may bring severe issues in real-world applications, *e.g.*, one not well-evaluated FL algorithm in the health care application can cause medical accidents. Thus, the survey study on the evaluation of FL to comprehensively analyze the evaluation targets and uncover the challenges in FL evaluation is urgently required to promote the healthy development of FL.

[2]https://github.com/Di-Chai/FedEval

II. Federated Learning Evaluation Goals

In this section, we summarize all the goals that need to be considered in the evaluation of FL (Figure 1). In general, there are two main types of FL processes: horizontal federated learning (HFL) and vertical federated learning (VFL). HFL assumes that parties have the same feature space but different sample spaces; generally, HFL is applied in edge computing scenarios, *e.g.*, different edge users collaboratively train the next-word-prediction model [35]. VFL assumes that parties have the same sample space but different feature spaces; VFL is typically a to-business paradigm of FL, which happens between organizations, *e.g.*, banks need data from online shopping companies to decide whether to approve one user's credit card application. The evaluation goals and measures presented in this survey do not restrict the type of FL and work with both HFL and VFL.

A. Goal 1: Utility

FL generally learns a model based on data from multiple parties without directly collecting data together to meet data protection requirements in many laws and regulations. Hence, the primary goal is to obtain a federated model with almost the same predictive power as the model directly trained from all parties' data to ensure the high *utility* of the FL model. We discuss utility from two aspects: *effectiveness* and *robustness*.

Goal 1.1: Effectiveness. FL aims to train a global model collaboratively using data distributed across participants. Ideally, the FL training should be able to achieve the same prediction accuracy as centralized training (*i.e.*, collecting all the data in one place). For the FL system that can approximate a centralized model's predictive power, we then call this FL system with *high effectiveness*.

Goal 1.2: Robustness. In practice, FL systems cannot always run in an ideal experimental environment, and various incidents may occasionally happen. Hence, a comprehensive evaluation of the FL system should pre-define such scenarios as

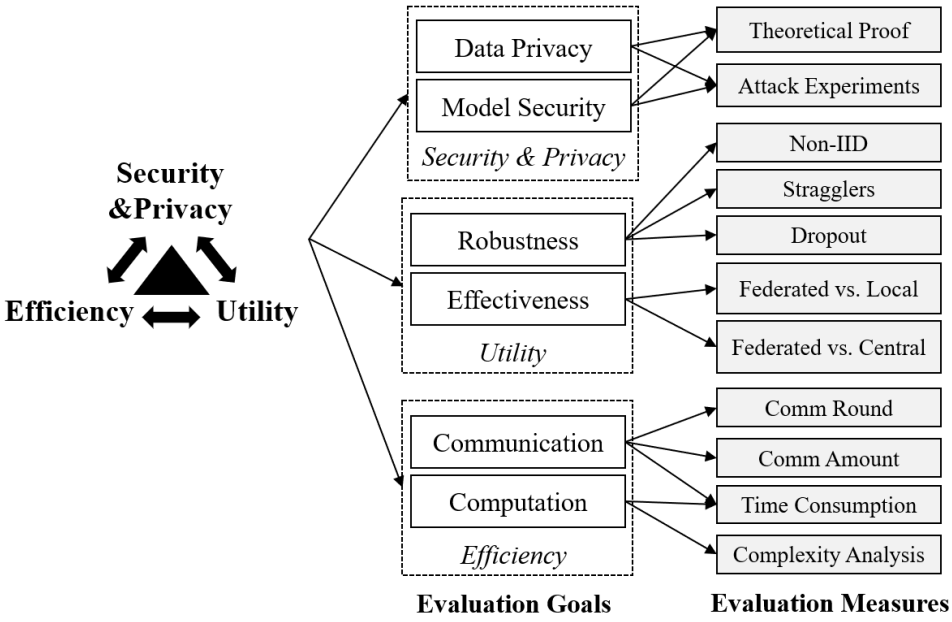


Fig. 1: An overview of FL evaluation goals and measures. Briefly, we categorize the evaluation goals of FL into three types: security & privacy, utility, and efficiency (Section II). Then, we summarize how to measure these goals in detail (Section III).

much as possible to reflect the system's *robustness* in practice. In particular, many participants indicate a significant disparity in devices. Data distributions, communication networks (3G, 4G, WiFi), computing resources (CPU, GPU), *etc*, may vary among parties. These diversities and uncertainties could cause issues that significantly affect the FL system [36].

B. Goal 2: Efficiency

Unlike conventional distributed machine learning, which is carried out on different machines in one data center [37], FL is performed on cross-data-center machines or edge devices, which have lower networking or computing resources [36]. Consequently, a deep neural network that could be trained in minutes using centralized machines may take hours to finish the training in FL [38]. Thus, efficiency is essential in FL and needs to be carefully evaluated. Based on existing works, we categorize the efficiency evaluation into two aspects: **communication** efficiency and **computation** efficiency.

Goal 2.1: Communication Efficiency. In HFL, the edge devices have limited networking resources, *e.g.*, low bandwidth and high latency, making the communication between the server and devices expensive [36]. In VFL, the federation usually consists of machines from different data centers (*i.e.*, from different companies). The cross-data-center communication is slow and has high latency [39]. Moreover, each party, in both HFL and VFL, may join more than one federation, and the FL training tasks from different federations will compete for resources [9, 40], making the communication efficiency issue more severe.

Goal 2.2: Computation Efficiency. In HFL, although the edge devices tend to have more powerful hardware, they still cannot match the ability of centralized computing servers, especially when dealing with large models [41]. Thus, the low computation efficiency problem cannot be dismissed in the federated learning scenario. Moreover, different parties often hold distinct computation resources, which may incur significant differences in computation speed between parties [42]. This can further impact the whole FL method's efficiency in a complicated manner.

C. Goal 3: Security & Privacy

Security and privacy are the foundation of FL systems. HFL algorithms, *e.g.*, FedAvg, perform aggregation on model parameters, and the risk of private data leakage can be reduced since the users' data never leaves their devices. However, recent works have shown that gradients can reveal input data and labels [43, 44]. Apart from the private data leakage threats to data holders, there are also model security threats to model users. Malicious edge parties could use data poisoning or model poisoning attacks to damage or backdoor the model. Specifically, FL is often expected to achieve the following two security and privacy goals:

Goal 3.1: Data Privacy. FL enables different parties to jointly train machine learning models without exchanging raw data, and only intermediate results are exchanged. However, recent works have shown that the intermediate results (*e.g.*, gradients) could be used to recover FL parties' private data [43, 44] when no privacy-preserving techniques are adopted (*e.g.*, homomorphic encryption), resulting in the data privacy issue.

Goal 3.2: Model Security. Federated learning happens over a bunch of distributed parties (*e.g.*, mobile devices), and there is no root of trust in existing methods, *i.e.*, every party could be malicious from the model users' perspective. Thus, the

participants could easily attack the model using poisoning methods, resulting in the model security issue [45, 46].

D. Trade-off between Utility, Efficiency, and Security & Privacy

It is worth noting that an FL system may not simultaneously improve all the goals, including *utility*, *efficiency*, and *security & privacy*. While a new algorithm improves one goal, it remains essential to comprehensively evaluate performance on other goals as well since trade-offs exist between different goals. The comprehensive analysis helps determine whether an algorithm represents unambiguous progress over state-of-the-art solutions by improving one aspect without detriment to others or gains on one goal induce losses on others, reflecting an ambiguous contribution. To this end, comprehensively evaluating an FL system from all three aspects becomes extremely important to deeply understand the advantages and disadvantages of FL systems (algorithms, models). Next, we would like to demonstrate more details about the trade-offs between the goals.

Utility vs. Efficiency. Federated SGD (FedSGD) and Federated Average (FedAvg) are two mostly well-known FL methods proposed by Google [35]. FedSGD inherits the settings of large-batch synchronous SGD (the state-of-the-art machine learning method used in data centers). In FedSGD, all clients synchronize the gradients before updating the local model weights. In contrast, only part of the clients participate in each round of training in FedAvg and the clients perform multiple rounds of local training before the synchronization.

FedSGD and FedAvg reveal the trade-off of utility and efficiency in FL. On the one hand, FedAvg improves communication efficiency (*i.e.*, fewer communication rounds) by increasing clients' local training rounds before the global synchronization. On the other hand, the increased clients' local training rounds unexpectedly drift the global model away from the global optimum under heterogeneous data distributions, making FedAvg reach worse model utility than FedSGD.

Apart from FedSGD and FedAvg, there are also other FL studies that encounter the trade-off between utility and efficiency. For example, some studies utilize gradient compression techniques to improve communication efficiency [12]; however, the model utility may decrease since only partial model parameters are synchronized.

Efficiency vs. Security & Privacy. While many privacy-preserving techniques are adopted in FL to enhance privacy and security protection, there is no free lunch. Privacy protection generally downgrades the efficiency of the system.

- Homomorphic Encryption (HE): HE is a special encryption algorithm that enables us to perform computations directly on encrypted numbers without decryption. HE is widely applied in FL to protect the intermediate results, *e.g.*, the gradients [47, 48]. The encrypted numbers (*i.e.*, ciphertext) bring the efficiency overhead in two aspects. First, the ciphertext consumes larger storage space than plaintext, which brings communication overhead. Second, the computation on ciphertext is more complicated than plaintext, which brings computation overhead.
- Secret Sharing (SS) [49]: SS is a secure multi-party computation framework, in which different participants secretly share their data among all participants. Each participant only holds one data partition, which leaks no private information about the raw data. Basic operations, like addition and multiplication, are defined under the partitioned data, and then computations like polynomial functions could be carried out. SS mainly brings communication overhead, especially when doing multiplication [50]. More specially, SS is very sensitive to the networking latency.
- Secure Aggregation (SA): SA is utilized in horizontal FL to combine the parameter updates from clients in a manner that protects the privacy of the individual client's local updates from a semi-honest server [51]. SA operates in a way similar to the addition operation in SS but with the added benefit of enhancing the resilience of the aggregation process when some clients may disconnect. Similar to SS, SA also introduces communication overhead.

It is worth noting that, the above protection techniques can often be incorporated into various FL algorithms [35, 52] to further enhance the protection level. Meanwhile, it would incur communication and/or computation overhead. Hence, in practice, the FL system designer should decide whether these extra protection methods are necessary according to the application scenario to balance efficiency and privacy protection.

Utility vs. Security & Privacy. In addition to efficiency, some privacy-preserving techniques may also degrade the utility of FL systems.

- Differential Privacy (DP): a well-known privacy-preserving technique adopted in FL is differential privacy (DP) [53]. Clients locally add DP noise to the data or model to protect the private data. DP-based FL solutions reveal the trade-off between model utility and privacy. Adding more noise will have better privacy preservation, however, will significantly downgrade the model's utility.
- Partial Homomorphic Encryption (PHE): another case of the trade-off between model utility and security & privacy in FL is adopting partial homomorphic encryption (PHE) in vertical federated logistic regression (LR) [54], in which PHE is utilized to protect the intermediate results. Since PHE cannot support non-linear functions (*e.g.*, Sigmoid activation function), Taylor polynomials are used to approximate the non-linear functions, which bring nonnegligible loss of model utility.

E. Necessity of comprehensively analyzing all the goals.

Based on our survey, we highly recommend new FL algorithm or systems to perform a comprehensive analysis on all the goals, including security and privacy, utility, and efficiency, for two reasons: 1) comprehensive analysis is the foundation of a fair comparison, and 2) comprehensive analysis is the key to find all the limitations before applied in real-world applications. Specifically, the comparison between different FL studies on partial goals is unfair because different goals form trade-offs and superiority in partial goals does not mean

superiority in all goals. For instance, many works do not analyze privacy protection, which will bring unfair efficiency comparisons. Because FL algorithms' efficiency varies greatly under different privacy-protection methods. For example, differential privacy (DP) and homomorphic encryption (HE) employ different privacy mechanisms and have very different efficiencies. However, claiming the DP-based method is much more efficient than the HE-based method as a major innovation is problematic without understanding their relative privacy guarantees. The major disadvantage of DP is that it harms model utility while HE does not. Comprehensive analysis is also essential to thoroughly assess one algorithm or system and discover all the limitations, such that the issue (*e.g.*, privacy or efficiency problems) could be fixed before being applied in real-world applications.

The major challenge of performing comprehensive analysis is the workload required for evaluations. To address this, we propose two solutions: 1) We develop a standardized evaluation platform, FedEval, to produce comparable and comprehensive results while reducing evaluation workload, and the detailed is introduced Section IV; 2) For incremental methods that only improve one or two goals based on an existing solution, another option is to analyze that the remaining goals have identical performance to prior studies that already reported comprehensive evaluation results. However, if the remaining goals were also not previously evaluated, assessments across all goals remain necessary.

III. Federated Learning Evaluation Measures

In this section, we review existing evaluation measures for different goals, including utility, efficiency, and security & privacy. For each goal, we introduce the commonly adopted evaluation measurements and factors considered in the literature.

A. Utility Evaluation Measures

For utility evaluation, we care about the predictive power of the obtained machine learning model. Adequate data is usually an indispensable condition for achieving satisfactory prediction accuracy, especially when deep learning is applied. However, such a condition usually cannot be satisfied in the real world due to privacy-preserving restrictions. Each data owner can only access their local data, also known as the isolated data islands problem [3]. FL systems should be able to break such isolation and achieve performance, *FL Effectiveness*, better than *Local Effectiveness* (*i.e.*, training model locally without joining any federations). In FL, we typically learn the global model by solving the following problem [35]:

$$\min_{w} f(w) = \sum_{k=1}^{N} p_k \cdot F_k(w) = \mathbb{E}_k[F_k(w)] \tag{1}$$

where N is the number of clients, $p_k \geq 0$ and $\sum_k p_k = 1$. $F_k(w)$ is defined as the empirical loss over the local data samples, *i.e.*, $F_k(w) = \frac{1}{n_k}\sum_{i=1}^{n_k} l_i(w)$ [55], where n_k is the number of samples at the k-th party, and we set $p_k = n_k/n$ where $n = \sum_k n_k$ is the total number of samples.

Definition 1 (FE - FL Effectiveness): We define the FL effectiveness as $\sum_{k=1}^{N} p_k \cdot Acc(h(w, x_k), y_k)$, where w is the model parameter learned from Equation 1, $h(w, x_k)$ outputs a probability distribution over the classes or categories that can be assigned to $x_k \sim D_k$, Acc function computes accuracy of $h(w, x_k)$ regarding the label y_k, and we set $p_k = n_k/n$.

Definition 2 (LE - Local Effectiveness): Using the same notation in Definition 1, we define the local effectiveness as $\sum_{k=1}^{N} p_k \cdot Acc(h(w_k, x_k), y_k)$, where w_k is the local model parameter learned by minimizing the local objective: $w_k = \arg\min_w F_k(w)$, and we set $p_k = n_k/n$.

Definition 3 (CE - Central Effectiveness): We define the central effectiveness as $Acc(h(w, x), y)$, where w is the model parameter trained by $\min_w F(w) := \mathbb{E}_{x\sim D}[f(w, x)]$, x represents data that collected from all the clients, and D is the global data distribution.[3]

Effectiveness. We can compare FE and CE/LE to measure the improvement brought by FL. The definition of central effectiveness (*i.e.*, Definition 3) follows accuracy definition from conventional machine learning, *i.e.*, the ratio of correctly predicted samples in the whole evaluation dataset [56]. While the definitions of local effectiveness (LE) and FL effectiveness (FE) are more complicated since the data is distributed across the clients. Empirically, we can compute the effectiveness of each client and then aggregate all clients' results [55, 57–72]. One problem is how to set the aggregation weights, which intuitively have two approaches: uniform weights or weighted by the number of samples. Very few studies explain which approach they use in the evaluation, but we see both types of implementations when investigating the open-sourced code on GitHub (*e.g.*, [57][4] used weights by sample and [58][5] used uniform weights). Theoretically, these two types of weights are identical if all clients hold the same number of samples. However, the number of data samples held by each client could be very heterogeneous in real-world applications, making these two weights produce incompatible results. In this survey, we recommend using weights by the number of samples, the reasons are 1) weights by the number of samples matches the loss of FL [35], which is also weighted averaged by the number of training samples; 2) uniform weights could produce biased evaluation since the clients with very small amount of data may dominate the final accuracy; If the system is specifically optimized for clients with small amount of data, we recommend to report effectiveness for these clients separately, instead of mixing with other rich-data clients. In this survey, to produce standardized and compatible measures, we use weights by the number of samples to define the effectiveness of FL and local training, which is formulated in Definition 1 and Definition 2.

- *FE vs. CE*. FL systems aim to obtain approximately the same accuracy as centralized machine learning systems, meaning that $FE \leq CE$ in general cases. If $FE \approx CE$,

[3]Centralized data collection and training is only an ideal experimental situation that represents a theoretical accuracy upper bound. In reality, we usually cannot put all the data in one place due to the restriction of privacy regulations.

[4]https://github.com/desternylin/perfed

[5]https://github.com/yaodongyu/TCT

then the FL system demonstrates no significant decline in accuracy compared to centralized learning, which is often the optimal case for an FL algorithm.
- *FE vs. LE*. For a practically useful FL system, FE should be larger than LE, meaning that FL gets performance improvements compared to learning only on local data. If $FE \leq LE$, the FL system fails to leverage the distributed knowledge to improve the model performance and should not be used in the application.

Robustness. In practice, various factors may vary to impact the performance of FL systems. Hence, these factors need to be clearly configured to evaluate an FL system's utility.

- *Non-IID Data & Model Personalization*. FL aims at fitting a model to data generated by different participants. Each participant collects the data in a non-IID manner across the network. The amount of data held by each participant may also significantly differ. The non-IID issue poses challenges to the training of FL. The model will be more difficult to reach convergence under non-IID data distribution, which could be further categorized into two main types [73].
  - *Non-IID feature setting*: The $P(y|x)$ of different parties are the same while the $P(x)$ are different. For example, in the FEMNIST dataset, different clients hold the same label space containing the same set of symbols, but they have different handwriting styles on the same symbols.
  - *Non-IID label setting*: The $P(x|y)$ of different parties are the same while the $P(y)$ are different. For instance, in the MNIST dataset, the non-IID data is usually simulated by allocating different labels to different parties [35] such that $P(y)$ are different while the feature distributions under the same label are the same.

  These two non-IID settings may impact model performance differently, so it is desirable to consider both of them for a robustness experiment on an FL system. Besides, non-IID data distribution may also lead to the necessity of *model personalization*, *i.e.*, each party attempts to learn a personalized model suitable to its local data distribution for better utility. We can measure the effectiveness of personalization by comparing a personalized model with a non-personalized (global) one.
- *Stragglers*. FL stragglers are defined as participants that fall behind the others regarding submitting the computation results [36]. FL stragglers could be caused by low computing power or small network bandwidth, which widely exist in practical FL system deployments. Suppose an FL system does not consider stragglers in its algorithm design (*e.g.*, relying on a purely synchronous updating strategy). In that case, stragglers may bring significant utility loss to the FL system [74]. If the FL follows a synchronous updating strategy, the stragglers will bring large efficiency overhead to the system. In the evaluation, stragglers could be simulated using the random delay to a certain part of the participants. Then, evaluate how the system efficiency is affected by the stragglers.
- *Dropout*. FL dropouts are defined as participants that fail to submit the computation results in training (*e.g.*, out of battery) [36]. Dropouts could be caused by networking drop-off or system out of service. Dropouts unexpectedly change the data distribution during the FL training, which may cause a convergence issue. A typical way of evaluating dropouts is by simulating dropout clients in the system and observing the influence on model performance.

Existing Works on Utility Evaluation. Table I outlines representative FL studies and their evaluation measures for utility. Our analysis reveals that most studies have at least one experiment focused on utility, such as comparing FL prediction accuracy with centralized, local, or other baseline FL methods' prediction accuracy. This is particularly true for papers published in database and AI conferences, where utility is usually the primary evaluation goal. Meanwhile, regarding the robustness evaluation, most of the AI studies focused on evaluating the performance under the non-IID data and overlooked the evaluation of heterogeneous systems, *i.e.*, when systems contain stragglers and dropouts. Specifically, only two papers [62, 72] evaluated the heterogeneous system in the surveyed representative studies. Experiments on straggler and dropout impact primarily appear in system papers [38, 41], while non-IID issues are mainly addressed by AI papers. One major reason is that the impact of non-IID data is usually modeled as a learning problem [60, 71, 74, 103, 104], and various solutions are proposed by AI studies. However, the system heterogeneity is an essential challenge in FL since real-world FL applications usually deal with millions of clients, making it challenging to coordinate [38, 41], and the heterogeneous system could decrease both efficiency and utility [38]. Thus the evaluation of heterogeneous systems is overlooked by existing studies and should be strengthened in future studies.

B. Efficiency Evaluation Measures

Since efficiency entails both communication and computation aspects, we provide an overview of their respective measures one by one.

Communication. Communication efficiency evaluation usually involves the following two metrics:

- *Communication Round (CR)*: CR measures how many rounds of communication are needed to jointly train a machine learning model from scratch to converge. Many research works draw CR-to-Accuracy curves to compare communication efficiency [35, 74, 109–111]. In some cases if the model requires a long time to converge, we can also fix a certain number of communication rounds and compare the accuracy [35, 61]. For instance, we may fix the CR to 500, method A has better communication rounds efficiency than method B if A shows higher accuracy than B after 500 rounds of training.
- *Communication Amount (CA)*: CA measures the amount of data transmitted during the FL training. Less CA could reduce the burden brought by the limited network bandwidth. A frequently used evaluation method is plotting the CA-to-Accuracy curve, which shows how much data

Venues	Papers	Primitive Design Goals and Keywords	Effecti--veness	Robustness *Non-IID*	Robustness *Straggler*	Robustness *Dropout*
Top System	Oort [38]	Efficiency, Participant Selection	●	●	●	●
	SFSL [41]	Privacy, Large-Scale Edge Computing, Recommender System	●	●	●	●
Top Security	FLTrust [75]	Security, Byzantine-robust FL	○	●	○	○
	SecAgg [51]	Privacy, Secure Aggregation	○	○	●	●
	Poseidon [48]	Privacy, Apply Fully HE in FL	○	○	○	○
	PrivaCT [76]	Privacy, Local Differential Privacy, Clustering	●	○	○	○
	Cerberus [77]	Utility, Privacy&Security, Apply FL in Security Events Prediction	●	●	○	○
	EIFFeL [78]	Privacy&Security, SecAgg on Verified Updates	○	○	○	●
	Pasquini et al. [79]	Privacy, Attack to SecAgg	○	○	○	○
	DP-GDBT [80]	Privacy, Differentially Private GBDT	●	○	○	○
	Shejwalkar et al. [81]	Security, Benchmark of Poisoning Attacks	●	●	○	○
	Snarkblock [82]	Privacy, Federated Anonymous Blocking	○	○	○	○
	Fang et al. [45]	Security, Local Data Poisoning Attacks	●	●	○	○
	Fu et al. [83]	Privacy, Label Inference Attack, Vertical FL	○	○	○	○
	FLDP [84]	Privacy, Efficiency, Differentially Private SecAgg	○	○	○	●
	FLAME [85]	Security, Defending Backdoor Attacks	●	●	○	○
Top Database	Refiner [86]	Security, Incentive-Driven FL	○	○	○	○
	Frog [87]	Privacy, Utility, Efficiency, Federated Debugging	○	○	○	○
	FedGraph [88]	Efficiency, Federated Subgraph Matching	●	●	○	○
	PFA [89]	Utility, Efficiency, Heterogeneous Differential Privacy	●	●	○	○
	FML [90]	Privacy, Federated Matrix Factorization, Recommender System	○	○	○	●
	CELU-VFL [91]	Efficiency, Vertical FL	○	○	○	○
	SMM [92]	Privacy, Utility, Mixing DP with MPC	●	○	○	○
	OpBoost [93]	Utility, Privacy, Optimizing DP for VFL	○	○	○	○
	VF^{2}Boost [39]	Efficiency, Efficient Vertical Federated GBDT	●	○	○	○
	BlindFL [94]	Privacy, Utility, Support Various kinds of Features in VFL	●	○	○	○
	Xiang et al. [95]	Privacy, Security, Differentially-private and Byzantine-robust FL	●	●	○	○
	FEAST [96]	Utility, Efficiency, Federated Feature Selection	●	○	○	○
	Li et al. [97]	Privacy, Differential Private Vertical Federated Clustering	●	○	○	○
	FedDSR [98]	Privacy, Utility, Federated Deep Reinforcement Learning	●	○	○	○
	MGFNAS [99]	Privacy, Federated Neural Architecture Search	●	●	○	○
	Zhang et al. [100]	Privacy, Security, Incentive, Game-Theoretical FL	●	○	○	○
	DSANLS [101]	Privacy, Efficiency, Federated Nonnegative Matrix Factorization	●	○	○	○
	VERTICOX [102]	Utility, Federated Survival Analysis	●	○	○	○
Top AI	q-FFL [55]	Utility, Fair Resource Allocation in FL	●	●	○	○
	Per-FedAvg [60]	Utility, Personalized FL	●	●	○	○
	pFedMe [103]	Utility, Personalized FL	●	●	○	○
	HeteroFL [104]	Efficiency, FL for Heterogeneous Clients	●	●	○	○
	FedMix [105]	Utility, Mixup for FL, Data Augmentation	●	●	○	○
	PartialFed [106]	Utility, Cross-domain Personalized FL	●	●	○	○
	FRL [107]	Efficiency, Utility, Constructing Initial Model for FL via Meta Learning	●	●	○	○
	Pillutla et al. [61]	Utility, Convergence Analysis	●	●	○	○
	Orchestra [59]	Utility, Efficiency, Unsupervised FL	●	●	○	○
	FedPU [62]	Utility, FL with Positive and Unlabeled Data	●	●	●	○
	FactorizedFL [63]	Utility, Personalized FL, Parameter Factorization	●	●	○	○
	SoteriaFL [64]	Privacy, Efficiency, Differentially Private FL, Communication Compression	●	○	○	○
	FedRolex [65]	Utility, Model-Heterogeneous FL	●	●	○	○
	FedNTD [108]	Utility, Forgetting Issues in FL, Continual Learning	●	●	○	○
	MR-MTL [67]	Privacy, Utility, Differentially Private Cross-silo FL	●	●	○	○
	Fed-EF [68]	Efficiency, Utility, Compressed FL with Error Feedback	●	●	○	○
	VerFedGNN [69]	Utility, Vertical Federated Graph Neural Network	●	○	○	○
	FED-PUB [70]	Utility, Personalized Sub-graph FL	●	●	○	○
	FedGMM [71]	Utility, Improving Effectiveness of FL on Unseen Data	●	●	○	○
	GuardHFL [66]	Privacy, Efficiency, Heterogeneous Client Capabilities, Customized Model	●	●	○	○
	PFL [72]	Efficiency, Asynchronized and Parallel FL	●	●	●	●

TABLE I: Utility evaluations in recent representative FL papers. To better identify the characteristics of each work, we present the papers' system names, primitive design goals, and keywords, which are summarized based on the papers' abstract and introduction. We use the authors' names as substitutes if the paper does provide a system name (*e.g.*, Pasquini et al. [79]). In the table, the black and white dots indicate whether the research work considers the corresponding measurements in the evaluation or not, which is investigated from the evaluation sections of the paper.

is transmitted when reaching a certain model accuracy [12, 109].

Computation. Computation efficiency evaluation typically employs the following two measures:

- *Theoretical Complexity Analysis:* FL carries out a privacy-preserving distributed model training, which unavoidably brings computation overhead. For example, FedAvg brings computation overhead regarding server aggregation. Apart from the computation overhead brought by the distributed training, the widely adopted privacy-preserving techniques in FL, *e.g.*, homomorphic encryption, also bring large computation overhead and need careful analysis [48, 112]. One fundamental method to evaluate computational efficiency is doing computation complexity analysis. Method A is better than B if A has a lower order of computation complexity.
- *Time Consumption:* Apart from the complexity analysis, experimental time consumption results are also frequently used to evaluate the efficiency of FL methods. Generally, we can draw a time-to-accuracy curve to compare the time consumption of different methods when reaching the same model performance [38, 106]. It is worth noting that computation time is influenced by the software and hardware environments. Some studies also report the time consumption by considering both communication and computation, *i.e.*, the total time consumption of an FL system [113]. Thus, when reviewing an FL paper's time consumption results, it is crucial to comprehend how time consumption is calculated.

FL applications can involve numerous participants, such as Google's federated mobile keyboard prediction with millions of participants [35]. Hence, To evaluate the practical efficiency of an FL system, conducting large-scale participant experiments may be necessary. An ideal solution would be to conduct experiments directly on a large number of devices, where each device represents a participant. However, only a few research institutions have the capacity to maintain and conduct evaluations on a large number of devices. A practical alternative is simulating all participants using a few computing servers. Specifically, virtual machine techniques, such as *Docker* containers [114], are commonly used to simulate multiple FL participants on a single server. It is also important to note that some efficiency measurements (*e.g.*, time consumption) can be affected by the hardware and software used in developing and deploying the system. Therefore, when conducting a comprehensive efficiency evaluation of FL systems, it is important to configure experiment parameters (*e.g.*, network bandwidth) during simulation.

Existing Works on Efficiency Evaluation. Table II lists the efficiency evaluation considerations in representative studies. Most of the studies report efficiency evaluation regarding communication or computation since efficiency is an essential metric that highly affects the practicality of FL methods. It is worth noting that about 75% of the surveyed representative FL studies do not evaluate efficiency regarding both communication and computation, which could lead to biased conclusions regarding the efficiency of FL systems. For example, communication rounds are commonly used as an efficiency metric in literature, but they may not always reflect the overall efficiency of the FL method. In particular, increasing local training rounds for every update in

Venues	Papers	Scale (# Party)	Comm		Comp	
			Round	*Amount*	$O(*)$	*Time*
Top System	Oort [38]	Millions	●	○	○	●
	SFSL [41]	Billions	●	●	●	○
Top Security	FLTrust [75]	Hundreds	●	○	○	○
	SecAgg [51]	Hundreds	○	●	●	●
	Poseidon [48]	<Hundred	○	○	●	●
	PrivaCT [76]	Thousands	○	○	○	○
	Cerberus [77]	<Hundred	○	○	○	○
	EIFFeL [78]	Thousands	●	○	○	○
	Pasquini et al. [79]	\	●	○	○	○
	DP-GDBT [80]	\	○	○	○	○
	Shejwalkar et al. [81]	Thousands	●	○	○	○
	Snarkblock [82]	\	○	○	○	●
	Fang et al. [45]	Hundreds	○	○	○	○
	Fu et al. [83]	\	○	○	○	○
	FLDP [84]	Thousands	○	○	○	●
	FLAME [85]	Hundred	●	○	○	●
Top DB	Refiner [86]	\	○	○	○	○
	Frog [87]	<Hundred	○	○	○	○
	FedGraph [88]	\	○	○	○	●
	PFA [89]	<Hundred	●	●	○	○
	FML [90]	<Hundred	○	○	○	○
	CELU-VFL [91]	<Hundred	●	○	○	●
	SMM [92]	\	○	○	○	○
	OpBoost [93]	<Hundred	○	●	○	●
	VF^{2}Boost [39]	<Hundred	○	○	○	●
	BlindFL [94]	<Hundred	●	○	○	○
	Xiang et al. [95]	<Hundred	○	○	○	○
	FEAST [96]	<Hundred	○	●	○	●
	Li et al. [97]	<Hundred	○	●	○	●
	FedDSR [98]	Hundreds	○	○	○	○
	MGFNAS [99]	<Hundred	●	○	○	○
	[100]	Hundreds	○	○	○	○
	DSANLS [101]	Hundreds	●	○	○	●
	VERTICOX [102]	<Hundred	●	○	○	●
Top AI	q-FFL [55]	Thousands	●	○	○	○
	Per-FedAvg [60]	<Hundred	○	○	○	○
	pFedMe [103]	Hundreds	●	○	○	○
	HeteroFL [104]	Thousands	●	●	○	○
	FedMix [105]	Hundreds	●	○	○	●
	PartialFed [106]	<Hundred	○	○	○	●
	FRL [107]	<Hundred	●	○	○	○
	Pillutla et al. [61]	Thousands	○	○	○	○
	Orchestra [59]	Hundred	●	○	○	●
	FedPU [62]	<Hundred	○	○	○	○
	FactorizedFL [63]	<Hundred	●	●	○	○
	SoteriaFL [64]	<Hundred	●	●	○	○
	FedRolex [65]	>Thousands	○	○	○	○
	FedNTD [108]	Hundreds	●	○	○	○
	MR-MTL [67]	Hundreds	○	○	○	○
	Fed-EF [68]	Hundreds	●	●	○	○
	VerFedGNN [69]	Thousands	○	●	○	○
	FED-PUB [70]	<Hundred	●	○	○	○
	FedGMM [71]	Hundreds	○	○	○	○
	GuardHFL [66]	<Hundred	●	●	○	●
	PFL [72]	\	●	○	○	●

TABLE II: Efficiency evaluations in existing works. $O(*)$ is the computation complexity analysis. Black dots indicate that a given study incorporated the corresponding measure in its evaluation, while white dots denote that the paper did not include that measure. Meanwhile, we also summarize the scale of efficiency evaluation in different studies, represented by the number of clients.

FedAvg [35] can reduce communication rounds but may not decrease overall time consumption, as it requires more local computation time for each party [115]. Another example that demonstrates the necessity of considering communication and computation simultaneously in the efficiency evaluation is when comparing the efficiency of two different privacy protection techniques: SS [66, 78, 94, 116] and HE [48, 94]. Intuitively, HE has higher computation complexity than SS but is more communication efficient than SS [117]. Biased efficiency comparison may happen if we compare HE and SS towards only one aspect of computation and communication. Regarding the number of clients used in the evaluation, we found that ∼20% of studies used thousands of clients, ∼20% used hundreds of clients, and ∼60% used fewer than one hundred clients.

C. Security & Privacy Evaluation Measures

The evaluation of FL methods regarding security and privacy could be generally conducted from both theoretical and empirical aspects:

- Theoretical: Are there privacy proofs analyzing the security and privacy of proposed methods?
- Empirical: Are there experiment results showing that the proposed methods can protect participants against existing attack methods?

While theoretical analysis is a mathematically rigorous way of validating security and privacy protection, it is still rare in existing FL papers.[6] In addition, security and privacy measures are typically evaluated in an adversarial manner, assuming certain types of attacks. Common threats considered in existing literature include:

[*Data Privacy*] Data Reconstruction Attacks. In FL, exchanging intermediate results is necessary for jointly training a machine learning model while keeping private data locally. Some pioneering FL studies leave these intermediate results unprotected, such as uploading local updates without protection in FedAvg [35]. Follow-up studies have shown that raw private data could be recovered from these exchanged intermediate results, including gradients and model parameters [43, 44, 52, 118, 119]. Moreover, malicious participants may be able to reconstruct training data using model inversion attacks with only the final FL model [120, 121].

[*Data Privacy*] Inference Attack.: In some cases, the intermediate training results and the final FL models are not enough to recover raw data precisely, but some sensitive attributes can still be inferred. For instance, adversaries can utilize intermediate information to train an attack model that infers whether a party/sample participates in FL model training, which is known as membership inference attack [122, 123].

[*Data Privacy*] ID Leakage. In VFL, directly sending sample IDs and computing the intersection could leak sensitive information about a party's customers. Hence, most VFL methods use private set intersection (PSI) for ID alignment [3]. However, PSI still leaks the sample IDs inside the intersection, revealing which users have registered accounts with other participants.

[*Model Security*] Byzantine Attacks. Malicious parties can launch data or model poisoning attacks during the federated training process so as to downgrade the FL model's performance, which is known as *Byzantine attacks* [45]. Data poisoning attacks involve injecting malicious data samples before the learning process starts, while model poisoning attacks assume that adversaries can directly manipulate the model parameters sent from FL parties to the server.

[*Model Security*] Backdoor Attacks. *Backdoor attacks* aim to control an FL model's prediction for an attacker-chosen subtask [46]. Specifically, such attacks can cause a backdoored FL model to misclassify a data sample to an attacker-chosen label. In facial recognition applications, this could allow an attacker to generate a fake ID, posing significant security risks. Different from Byzantine attacks, backdoor attacks aim to modify the model's behavior on a small portion of data without affecting the overall prediction accuracy significantly. Hence, backdoor attacks can be particularly challenging to detect since they often do not show up during normal FL evaluation and testing procedures.

Threat Model. It is worth noting that a research paper on FL usually defends against only partial attacks from the above list. It is essential to first define what are the threats (*i.e.*, the threat model) before analyzing the security & privacy. Typically, the following assumptions would be made for potential adversaries:

- Security Definition: The security definition defines the degree of honesty of participants. Generally, two types of security definitions are used in FL studies:
  - *Honest But Curious (Semi-Honest)*: The honest but curious setting, also known as semi-honest, assumes that the participants strictly adhere to the pre-defined protocol but attempt to learn as much information as possible from the received messages. This setting is commonly considered in security and privacy analyses presented in FL papers.
  - *Malicious*: The malicious participants will not strictly follow the pre-defined protocol, and take any action to achieve their goal. To model malicious behavior during joint model training in FL, it is necessary to consider the specific threats that need to be protected against. However, defending against such parties is challenging, and only a few FL studies have considered them.
- Collusion Party Number: The ability of an FL system's defense against attacks from a single party does not guarantee protection against collusion between multiple parties. Therefore, it is essential to consider the number of parties that could collude to conduct attacks when evaluating an FL system's privacy and security levels.

Existing Works on Security & Privacy Evaluation. Table III summarizes the security and privacy evaluation measures in representative FL papers. It is notable that papers published in security conferences prioritize security and privacy evaluations. In addition, database papers also give significant

[6] We investigated 60+ FL papers published on NeurIPS, ICML, ICLR, KDD, CCS, NDSS, OSDI, *etc*. in the last five years, and found that less than 10% provided rigorous proofs.

Venues	Papers	Security Definitions		Theoretical Proof		Empirical Experiments	
		Semi Honest	*Malicious*	*Model Security*	*Data Privacy*	*Model Security*	*Data Privacy*
Top Sys	Oort [38]	○	○	○	○	○	○
	SFSL [41]	●	○	○	●	○	○
Top Security	FLTrust [75]	○	●	●	○	●	○
	SecAgg [51]	●	○	○	●	○	○
	Poseidon [48]	●	○	○	●	○	○
	PrivaCT [76]	○	○	○	●	○	○
	Cerberus [77]	○	●	○	○	●	○
	EIFFeL [78]	○	●	●	○	●	○
	Pasquini et al. [79]	○	●	○	○	●	○
	DP-GDBT [80]	●	○	○	○	○	○
	Shejwalkar et al. [81]	○	●	○	○	●	○
	Snarkblock [82]	○	○	○	○	○	○
	Fang et al. [45]	○	●	○	○	●	○
	Fu et al. [83]	○	●	○	○	●	○
	FLDP [84]	●	●	○	●	○	○
	FLAME [85]	●	○	●	○	●	○
Top DB	Refiner [86]	○	●	○	○	○	○
	Frog [87]	●	○	○	●	○	○
	FedGraph [88]	○	○	○	●	○	○
	PFA [89]	○	○	○	●	○	○
	FML [90]	○	○	○	●	○	○
	CELU-VFL [91]	○	○	○	○	○	○
	SMM [92]	○	○	○	●	○	○
	OpBoost [93]	○	○	○	○	○	○
	VF^{2}Boost [39]	○	○	○	○	○	○
	BlindFL [94]	●	○	○	●	○	○
	Xiang et al. [95]	○	●	○	●	●	●
	FEAST [96]	○	○	○	●	○	○
	Li et al. [97]	●	○	○	●	○	○
	FedDSR [98]	○	○	○	○	○	○
	MGFNAS [99]	●	○	○	●	○	○
	Zhang et al. [100]	○	●	○	●	●	○
	DSANLS [101]	●	○	○	●	○	○
	VERTICOX [102]	○	○	○	○	○	○
Top AI	q-FFL [55]	○	○	○	○	○	○
	Per-FedAvg [60]	○	○	○	○	○	○
	pFedMe [103]	○	○	○	○	○	○
	HeteroFL [104]	○	○	○	○	○	○
	FedMix [105]	○	○	○	○	○	○
	PartialFed [106]	○	○	○	○	○	○
	FRL [107]	○	○	○	○	○	○
	Pillutla et al. [61]	○	○	○	○	○	○
	Orchestra [59]	○	○	○	○	○	○
	FedPU [62]	○	○	○	○	○	○
	FactorizedFL [63]	○	○	○	○	○	○
	SoteriaFL [64]	○	○	○	●	○	○
	FedRolex [65]	○	○	○	○	○	○
	FedNTD [108]	○	○	○	○	○	○
	MR-MTL [67]	○	○	○	●	○	○
	Fed-EF [68]	○	○	○	○	○	○
	VerFedGNN [69]	●	○	○	●	○	●
	FED-PUB [70]	○	○	○	○	○	○
	FedGMM [71]	○	○	○	○	○	○
	GuardHFL [66]	●	○	○	●	○	○
	PFL [72]	○	○	○	○	○	○

TABLE III: Privacy evaluations in existing works. Similarly, the black and white dots represent whether the studies considered the corresponding measures in the evaluation or not, respectively. Regarding the security definition, we also summarize the threat models used in representative works (*i.e.*, semi-honest, malicious, or not defined in the paper).

attention to security and privacy concerns in their method design. Existing work mainly has two approaches to evaluate data privacy: 1) Provide theoretically proofs to show that the solutions are differentially private (*e.g.*, [80, 84, 89, 100]) or all the intermediate results are protected by HE (*e.g.*, [48]) and secret sharing (*e.g.*, [66, 78, 94]); 2) Perform empirical attack experiments to show that the solutions are secure against the state-of-the-art (SOTA) attacks (*e.g.*, [69, 95]). Regarding the model security, existing studies also explored the evaluation in two ways: 1) Provide security analysis to show solutions' ability to defend the attacks (*e.g.*, the utility loss is bounded under the poisoning attacks [75, 78, 85]); 2) Perform empirically poisoning attacks to show the solutions' utility loss under the attacks (*e.g.*, [45, 83, 95]). We also observe that most FL papers presented at AI conferences do not explicitly discuss security and privacy issues. Considering that security and privacy are primary motivations for developing FL systems, we suggest that AI papers should also give more attention to these concerns.

IV. FedEval: A Platform for FL System Evaluation

After reviewing existing FL studies, it is clear that a standard and easy-to-reproduce procedure for comprehensive evaluation of utility, efficiency, and security & privacy is still lacking. We have developed an open-source platform called *FedEval* to standardize and simplify the evaluation of FL algorithms. An overview of our evaluation platform is presented in Figure 2. To use FedEval, users only need to provide a single script that contains the necessary FL functions or callback functions, such as how the server aggregates the parameters from different clients, to evaluate a new FL algorithm or test new attack/defense methods. The platform consists of three key modules.

- Data Config and the *FedData* module: FedEval currently provides seven standard FL datasets, including MNIST, CIFAR10, CIFAR100, FEMNIST, CelebA, Sentiment140, and Shakespeare. Different data settings (*e.g.*, non-IID data) can be implemented by changing the data configs. Self-defined data is also supported. We only need to inherit the *FedData* class and define the *load_data* function to add a new dataset, which will share the same processing functions with the built-in datasets.
- Model Config and the *Keras.Model* module: Currently, three machine learning models are built inside our system, including *MLP*, *LeNet*, and *StackedLSTM*. We use TensorFlow [124] as the backend, and all the models are made via subclassing the Keras model. Thus, adding new machine learning models is very simple in FedEval.
- Runtime Config and the *strategy* module: One of the essential components in FedEval is the *strategy* module, which defines the protocol of the federated training. Briefly, the FL strategy module supports the following customization:
  - *Customized uploading message*, *i.e.*, which parameters are uploaded to the server from the clients.
  - *Customized server aggregation method*, *e.g.*, weighted average.

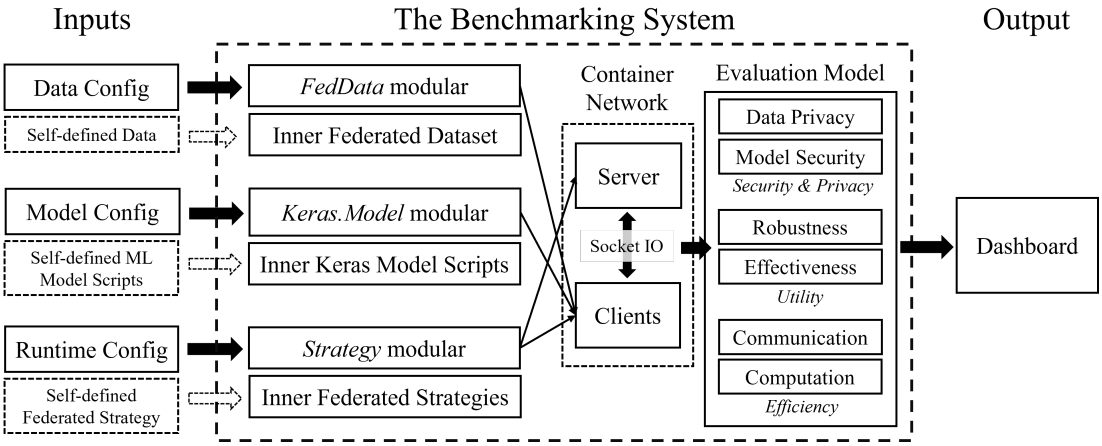


Fig. 2: An overview of the FedEval evaluation platform. Users can evaluate existing algorithms using preset datasets in FedEval under different scenarios by providing the data, model, and runtime configs. Users can also evaluate new algorithms on new datasets by customizing the data, model, and strategy modules. Using the built-in evaluation goals and measures, FedEval significantly reduces the workload of the FL evaluation and produces standardized evaluation results.

  - *Customized training method for clients*, *e.g.*, the clients' model can be trained using regular gradient descent method or other solutions like knowledge distillation.
  - *Customized method for incorporating the global and local model*, *e.g.*, one popularly used method is replacing the local model with the global one before training.

Compared with conventional machine learning, the major challenge of obtaining standard FL evaluation metrics is how to appropriately simulate heterogeneous clients and capture metrics (*e.g.*, communication costs) that reflect real-world conditions. We introduce the FedEval platform's approach to addressing this challenge.

- Participants and Network Simulation. A widely-used method for simulating multiple participants is using multiprocessing, but we think it has the following problems: 1) it is hard to control the hardware resources (*e.g.*, CPU and memory) used by each process; 2) it is hard to evaluate the performance under different network settings (*i.e.*, bandwidth and latency). Our solution is putting all the participants into different docker containers, in which the hardware resources used by each participant could be fully controlled, including the CPU, GPU, memory, disk storage, *etc*. The server and clients from different containers communicate through WebSocket. Container networks bridge the communication between containers. Under such an architecture design, it is easy to change the network settings (*i.e.*, bandwidth and latency) by directly configuring the virtual network interface card (NIC).
- Communication Evaluation. Communication size is an essential evaluation metric for FL algorithms since the participants in FL tend to have limited network bandwidth, and a large communication size may bring significant efficiency overhead. A naive solution for evaluating the communication size, which is used in many existing FL studies, is directly measuring the size of the transmitted objects in the memory, and many utility packages (*e.g.*, the "getsizeof()" function in Python) could be used. However, such evaluation implementation may have two issues: 1) Different packages usually have different results; 2) Not all the objects could be accurately assessed using this method. To solve these problems, we measure the communication size by directly collecting data from the virtual NIC, which automatically records the amount of data sent out and received. Compared with measuring the transmitted data size in memory, our solution is more accurate and significantly reduces the implementation complexity.
- Time Evaluation. The implementation of time evaluation in FL is challenging because it may have many variations based on different purposes. For example, apart from the overall time consumption in each training round, we would also like to provide other time consumption statistics to help the users improve the FL algorithms, *e.g.*, the computation and communication time of the clients, the aggregation time at the server, *etc*. The naive implementation of these time evaluation metrics is complicated and requires significant modifications to the platform's source code. Our solution is providing a flexible time evaluation by collecting a group of timestamps, through which multiple time evaluation metrics could be calculated. Specifically, as illustrated in Figure 3, we put four timestamps in the platform, which are the time of server sends parameters (t_1), clients receive parameters (t_2), clients send parameters (t_3), and server receives parameters (t_4). Assuming we have k clients in the training, then $\{(t_1^i, t_2^i, t_3^i, t_4^i)|1 \leq i \leq k\}_n$ represents all the timestamps collected in the i-th round. Different combinations of these timestamps have different meanings:
  - Client computation time (average): $\frac{1}{k}\sum_{i=1}^{k}(t_3^i - t_2^i)$.
  - Server aggregation time in the n-th round: $sa = min(\{t_1^i|1 \leq i \leq k\}_n) - max(\{t_4^i|1 \leq i \leq k\}_{n+1})$

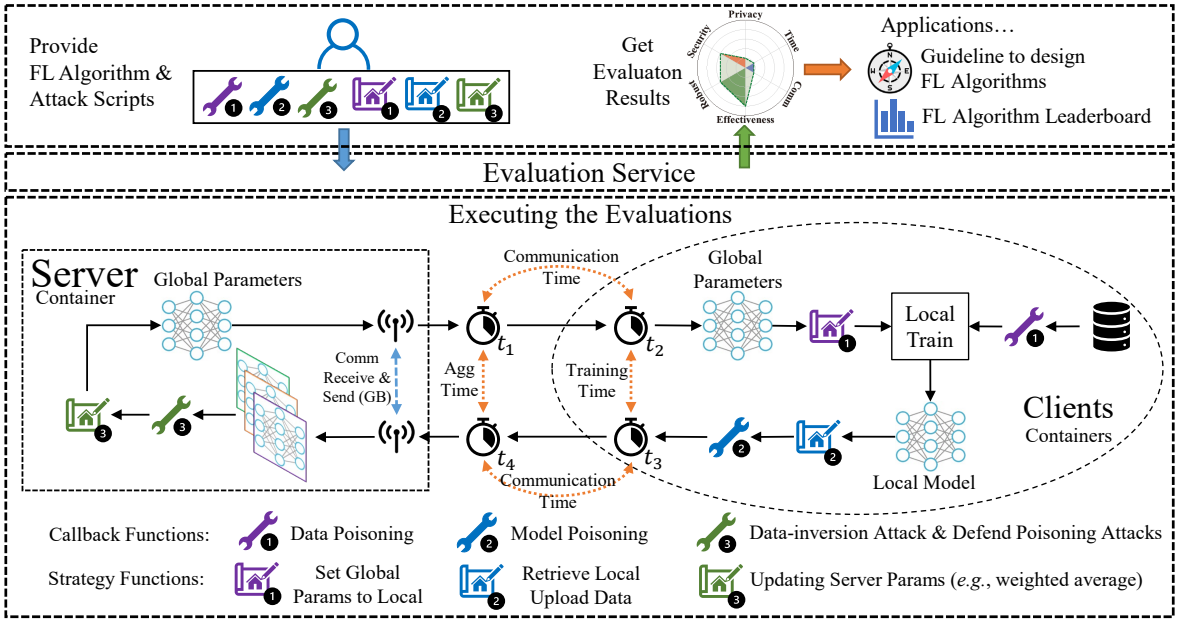


Fig. 3: The FedEval's detailed workflow when evaluating customized algorithms. Users can provide scripts encompassing different strategy functions, enabling the assessment of various customized algorithms. For instance, these functions can customize the aggregation of parameters and the process of updating the global parameters to the local models. Additionally, users can test diverse attacking and defending techniques through different callback functions. As illustrated, clients can perform customizable data poisoning prior to local training and model poisoning before uploading updates. Conversely, the server can execute customizable data-revealing attacks and defend against poisoning attacks originating from the client side. We put the full description of the function interface of FedEval in Appendix A.

- Real-world time consumption in the n-th round: $min(\{t_1^i|1 \leq i \leq k\}_n) - min(\{t_1^i|1 \leq i \leq k\}_{n+1})$
- Federated time consumption in the n-th round: $sa + max(\{t_4^i - t_1^i|1 \leq i \leq k\}_n)$

Our platform records all the timestamps and outputs the real-world and federated time consumption. The users can compute more metrics based on these timestamps.

With appropriate client simulation, resource control, and efficiency measurements, the other metrics could be easily obtained. For example, the straggler evaluation in the utility could also be done by allocating clients with heterogeneous computing or networking resources. The entire system is open-sourced, and the essential components, such as datasets, ML models, and FL strategies, can be easily used or self-defined. Researchers can easily implement their new FL method ideas and evaluate them with FedEval (*e.g.*, *FedSVD* [113]).

To demonstrate the usability of FedEval, we present its detailed workflow when evaluating customized algorithms in Figure 3. As illustrated in the figure, the researchers can provide strategy functions to customize the behaviors of the FL algorithm, *e.g.*, how the parameters are aggregated at the server and how to set the global updates to the local model. Meanwhile, the researchers can use customized callback functions to perform experiments of attacking and defending against the attacks. On the client side, we can use callback functions to poison the data before local training or poison the model before uploading local updates. On the server side, we can use callback functions to perform data-revealing attacks when receiving individual client updates and detect the poisoning updates before the aggregation. Due to the space limitation, we put the full description of the function interface of FedEval in Appendix A.

An important characteristic of FedEval is its capability to evaluate an FL algorithm's performance from a holistic perspective including utility, efficiency, and security & privacy. We have tested representative FL algorithms, including FedSGD [35], FedAvg [35], FedProx [74], FedOpt [110], *etc*. Table IV shows the utility evaluation of these four algorithms, *i.e.*, comparing the effectiveness to local and central training and the effectiveness under non-IID data. The utility evaluation shows that all the tested FL algorithms have significantly better performance than local training and show a small decrease in accuracy compared to centralized training on most datasets. Regarding the robustness under non-IID data setting, FedProx has the best performance and yields the best average effectiveness under non-IID data, which matches the results reported from the original paper. Figure 4 shows the efficiency comparison of these four algorithms regarding the communication rounds, communication amounts, and time consumption. The efficiency evaluation shows that FedSGD tends to have worse efficiency compared to the other three algorithms, and FedOpt shows superior efficiency on a relatively large dataset (*i.e.*, Shakespeare), which also matches the results report from the original paper. Figure 6 shows the data reconstruction attack [43] between FedSGD and FedAvg. Theoretically, FedProx

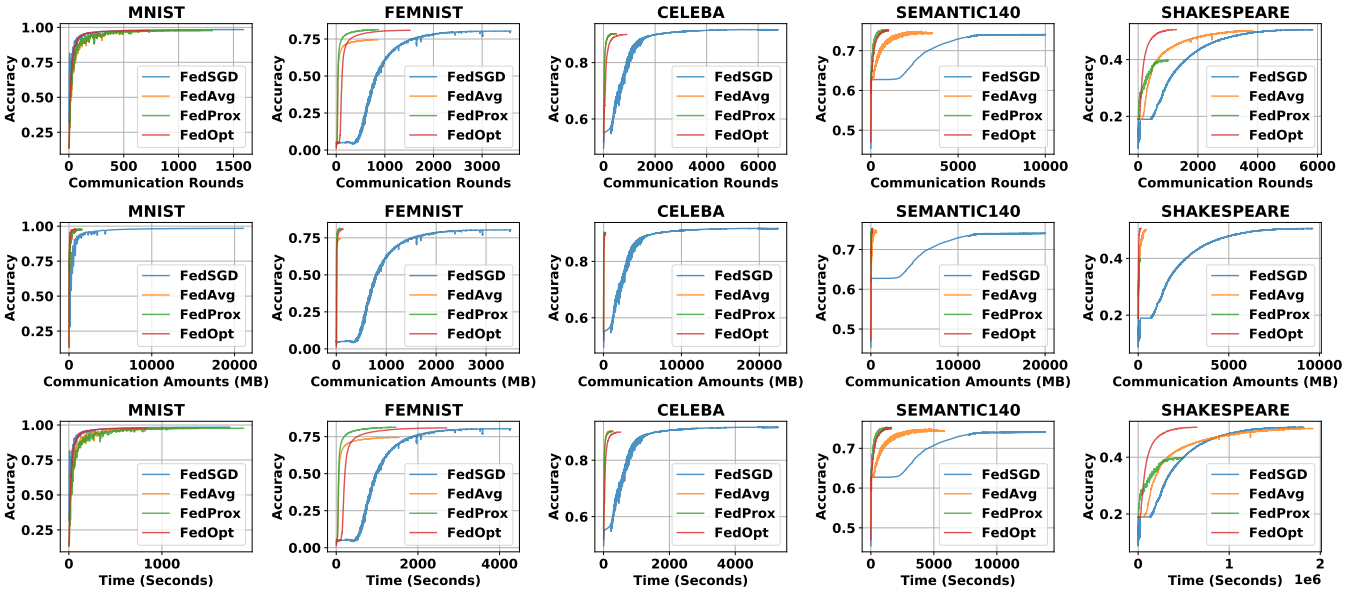


Fig. 4: Efficiency evaluation of four popular FL methods through FedEval on four datasets. The results show that FedSGD has the worst efficiency regarding both communications and computations, and FedOpt has superior efficiency on the larger dataset (*i.e.*, Shakespeare), which match the results reported by original papers.

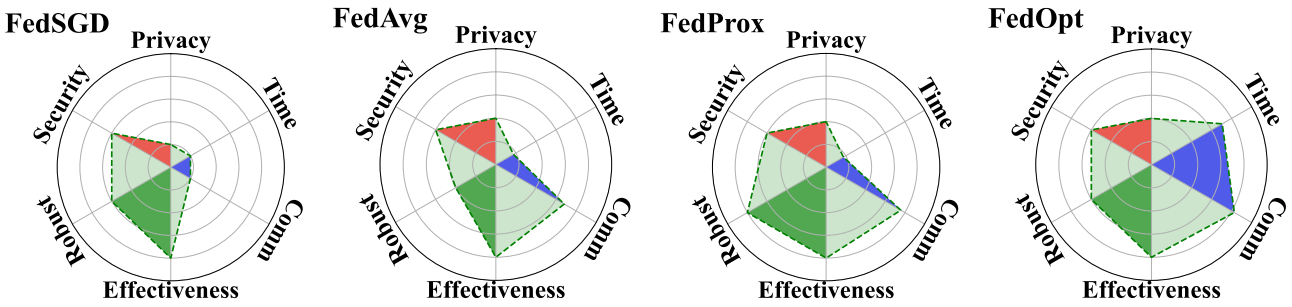


Fig. 5: Visualizing the FedEval evaluation results through radar charts which compare four most popular FL algorithms from security and privacy, utility (*i.e.*, robustness and effectiveness), and efficiency (*i.e.*, communication and time consumption).

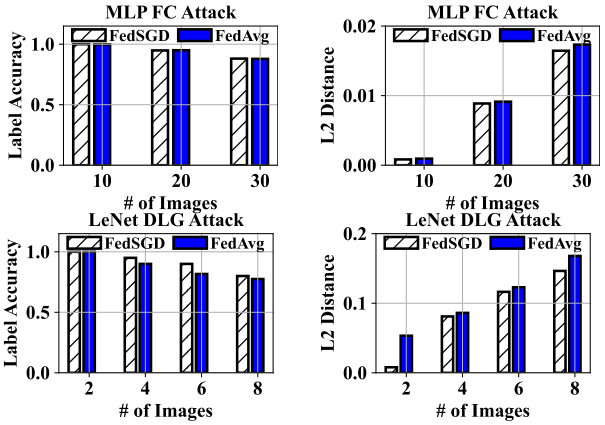


Fig. 6: FedSGD vs. FedAvg under the data-reconstruction attack [43]. FedAvg has better performance than FedSGD by having lower attack label accuracy and higher L2 distance between the recovered and real samples.

and FedOpt have the same attack results as FedAvg since clients in these protocols upload the same information (*i.e.*, parameters after multiple rounds of local updates) to the server. Figure 6 shows that FedAvg has better performance than FedSGD. The possible reason is that the parameters uploaded in FedAvg contain multiple rounds of local training while FedSGD only has one round of training, and the accumulated local updates in the parameters make it harder to recover the raw data.

While the above table and figures independently present the evaluation results regarding utility, efficiency, and privacy, we also attempt to merge the evaluation results into one radar chat to provide an overview as well as highlight the strengths and weaknesses of each algorithm. The final results are presented in Figure 5. We put the detailed methods for obtaining the radar charts on an online document[7] due to the space limitation and ease of future updates, *i.e.*, we will also continue evaluating more algorithms and the radar charts may also be updated accordingly. For more detail of FedEval, *e.g.*, the interface design, please refer to our technical report [115]

[7]https://fedeval.readthedocs.io/en/latest/benchmark/benchmark.html

TABLE IV: Utility evaluation of four popular FL methods through FedEval on four datasets. All the experiments are repeated ten times, and the average values and standard error (*i.e.*, values in parentheses) are reported. The MNIST dataset adopts the non-IID label setting, while the other datasets adopt the non-IID feature settings.

Dataset	IID	Local	Central	FedSGD	FedAvg	FedProx	FedOpt
mnist	N	0.11319 (0.013)	0.98614 (0.001)	0.98390 (0.001)	0.97843 (0.006)	0.97874 (0.003)	0.97679 (0.003)
mnist	Y			0.98341 (0.002)	0.98651 (0.001)	0.98683 (0.001)	0.98351 (0.001)
femnist	N	0.48231 (0.056)	0.84961 (0.002)	0.80461 (0.015)	0.81234 (0.004)	0.81288 (0.005)	0.80783 (0.003)
femnist	Y			0.81351 (0.012)	0.83476 (0.004)	0.83385 (0.002)	0.83187 (0.004)
celebA	N	0.70307 (0.007)	0.92400 (0.005)	0.91707 (0.005)	0.90170 (0.005)	0.90120 (0.007)	0.89913 (0.008)
celebA	Y			0.91867 (0.006)	0.90267 (0.012)	0.90210 (0.011)	0.89957 (0.011)
sent140	N	0.74447 (0.006)	0.79263 (0.002)	0.74131 (0.006)	0.75578 (0.003)	0.75626 (0.003)	0.75263 (0.004)
sent140	Y			0.74024 (0.005)	0.76504 (0.004)	0.75839 (0.005)	0.74955 (0.007)
Average	N	0.51076	0.88809	0.86172	0.86206	0.86227	0.85909
Average	Y			0.86395	0.87224	0.87029	0.86612

as well as the online document[8].

In summary, FedEval provides a flexible framework for researchers to produce standardized evaluation results that closely mimic real-world settings using the measurements summarized in this survey. FedEval also reduces the workload required for comprehensive analysis since researchers only need to define the FL workflow (*i.e.*, through scripts), and evaluations can be automatically completed using the built-in metrics on the platform. While being a platform that makes a significant contribution to the evaluation of FL, FedEval also has two limitations. Firstly, while the platform provides good support for utility and efficiency evaluations, the attacks for privacy and security evaluation still need to be enriched. Secondly, the automated evaluation of vertical FL algorithms is currently not supported by FedEval. We will keep updating the platform in the future to solve these two limitations, *i.e.*, adding more attacks regarding the privacy and security evaluation and adding support for the evaluation of vertical FL.

V. Future Directions

In this section, we summarize several challenges and future research directions in FL evaluation.

A. A Comprehensive Evaluation Procedure

While existing works focus on one or two issues in FL, their evaluation results are also restricted to the corresponding areas. For example, FedAvg [35] tries to reduce the communication rounds by adding the number of clients' local updates. However, the resulting increased local running time is not evaluated; non-IID issues are not thoroughly tested. FLTrust [75] proposed a Byzantine attack-robust FL framework by carefully verifying clients' uploaded updates; however, individual updates for verification may bring the risk of private data leakage. As trade-offs widely exist in FL system design (Sec. II-D), only a comprehensive evaluation process can help practitioners make the optimal decision on the design of practical FL systems and applications.

[8]https://fedeval.readthedocs.io/

B. Standard Evaluation Metrics

Although the comprehensive evaluation gives us a thorough assessment of FL frameworks, comparing different FL studies is still very difficult because the existing evaluation metrics are incompatible. Different studies usually have different focuses in the evaluation. For example, model A improves the FL communication efficiency by 10%, and model B improves the FL computation efficiency by 15%. We cannot conclude that model B is better than model A and vice versa since none of these two metrics (*i.e.*, communication and computation) are always more important than the other one in different applications.

Thus, we need a set of FL evaluation metrics that are commonly agreed to be compatible with different scenarios, *i.e.*, a set of *standard* evaluation metrics. In other words, FL studies could be compared using these standard metrics under different scenarios with no ambiguity.

One good example of a compatible metric is the energy and carbon footprint [125] since environmental wellness is one of the most important tasks of our society. FL models with fewer carbon emissions are better when achieving the same effectiveness.

C. Real-time and Continuous Evaluations

The evaluation of FL systems should be a real-time and continuous process. Specifically, the evaluation system should have the following functionalities:

- *Utility & Efficiency Evaluation*: Requiring an easy-to-use evaluation interface and a group of benchmarking results (*e.g.*, FL leaderboard). The system should enable researchers to evaluate new modes quickly, *e.g.*, by uploading a simple script, and the system will automatically evaluate the new model. The evaluation results could be presented using a leaderboard, from which the researchers could quickly specify the state-of-the-art FL model and make performance comparisons.
- *Security & Privacy Evaluation*: Requiring a real-time and continuous verification to detect the attacks. Most of the FL studies use semi-honest security definitions, however, the security under the semi-honest assumption is not good enough for real-world applications because the parties that participated in the distributed training cannot fully trust each other, *i.e.*, they will not believe that the others are semi-honest. Thus, real-time verification is essential to monitor each party's behavior and detect malicious participants deviating from the protocol. Furthermore, as we mentioned in section Section III-C, private data leakage or model tampering may happen before, during,

and after the FL training. Thus, security and privacy verification should be a real-time and continuous process.

D. Contribution Evaluation for Incentive Design

While not discussed in detail in this article, the incentive is also significant for FL, as parties work together only when incentives are designed satisfactorily. A suitable incentive mechanism in FL should satisfy the participants' rationality, meaning that each party's reward should be greater than the cost of joining the federation. Meanwhile, the parties with more contributions should gain more rewards to achieve fairness. There are also many other targets of designing an incentive mechanism for FL, such as reducing the delay in distributing rewards [9]. The evaluation plays a vital role in the incentive mechanism, especially when evaluating the participants' contributions. Intuitively, one participant's contribution could be evaluated by comparing the model performance when trained with and without its datasets, *e.g.*, Shapley values [126] is often adopted. The evaluation system could incorporate real-time contribution evaluation and reward distribution to serve as an incentive mechanism.

E. Evaluation on FL platforms

FL platforms are those frameworks that support simulating FL algorithms locally for research purposes or running FL in a distributed manner for industry applications. With the development of FL, many platforms have appeared: *e.g.*, FATE [127], FedML [128], FedScale [129], *etc*. However, in real-world applications or research studies, it is usually hard for users to determine which platform is the best choice under a certain scenario. Thus, evaluating these platforms is essential to benchmark and compare their efficiency and effectiveness under different scenarios. Meanwhile, we can also perform attack experiments on those platforms to assess privacy protection and uncover potential privacy issues before utilizing them in real-world applications. Notably, we can extend the evaluation goals and measures in this survey from evaluating algorithms into platforms, containing utility, security & privacy, and efficiency. We discuss the extensibility of FedEval to evaluate different FL platforms in Appendix B.

VI. Conclusion

In this survey, we provide a comprehensive overview of the evaluation goals and measures for FL studies. We categorized the key evaluation goals into utility, efficiency, and security & privacy. For each goal, we reviewed commonly used metrics and evaluation methods from existing literature. We also discussed the necessity of conducting comprehensive evaluations across all goals due to the trade-offs between them. To facilitate such comprehensive analysis, we introduced FedEval, an open-source platform that simplifies implementing standardized FL evaluations.

We also summarized several open challenges and future directions for FL evaluations. First, establishing standardized evaluation metrics that are compatible with different scenarios would enable fairer comparisons between different FL solutions. Second, developing capabilities for real-time verification of efficiency, utility, and especially security would be highly valuable for practical deployments. Third, evaluating the contributions of participants could support the design of incentive mechanisms.

Overall, as FL continues maturing from the research domain towards real-world applications, strong evaluation methodologies will play an indispensable role in ensuring system quality and user trust. We hope this survey provides a useful reference for future efforts in advancing FL evaluation.

Acknowledgments

The work is supported by the Key-Area Research and Development Program of Guangdong Province (2021B0101400001), the NSFC Grant no. 61972008, the Hong Kong RGC TRS T41-603/20R, the National Key Research and Development Program of China under Grant No.2018AAA0101100.

References

[1] FTC, "ftc imposes $5 billion penalty and sweeping new privacy restrictions on facebook," https://www.ftc.gov/news-events/news/press-releases/2019/07/ftc-imposes-5-billion-penalty-sweeping-new-privacy-restrictions-facebook, accessed: 2019-07-24.

[2] CNN, "amazon hit by record $887 million eu privacy fine," https://edition.cnn.com/2021/07/30/tech/amazon-eu-privacy-fine/index.html, accessed: 2021-07-30.

[3] 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. 12:1–12:19, 2019.

[4] Gartner. (2020) hype cycle for data science and machine learning. [Online]. Available: https://www.gartner.com/en/documents/3988118

[5] X. Zhang, H. Gu, L. Fan, K. Chen, and Q. Yang, "No free lunch theorem for security and utility in federated learning," *ACM Trans. Intell. Syst. Technol.*, vol. 14, no. 1, pp. 1:1–1:35, 2023.

[6] X. Zhang, Y. Kang, K. Chen, L. Fan, and Q. Yang, "Trading off privacy, utility and efficiency in federated learning," *CoRR*, vol. abs/2209.00230, 2022.

[7] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *CoRR*, vol. abs/1806.00582, 2018.

[8] M. Cong, H. Yu, X. Weng, and S. Yiu, "A game-theoretic framework for incentive mechanism design in federated learning," in *Federated Learning*, ser. Lecture Notes in Computer Science. Springer, 2020, vol. 12500, pp. 205–222.

[9] H. Yu, Z. Liu, Y. Liu, T. Chen, M. Cong, X. Weng, D. Niyato, and Q. Yang, "A sustainable incentive scheme for federated learning," *IEEE Intell. Syst.*, vol. 35, no. 4, pp. 58–69, 2020.

[10] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "Fedbn: Federated learning on non-iid features via local batch normalization," in *ICLR*. OpenReview.net, 2021.

[11] H. Yang, M. Fang, and J. Liu, "Achieving linear speedup with partial worker participation in non-iid federated learning," in *ICLR*. OpenReview.net, 2021.

[12] F. Sattler, S. Wiedemann, K. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-iid data," *CoRR*, vol. abs/1903.02891, 2019.

[13] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S. Kim, "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data," *CoRR*, vol. abs/1811.11479, 2018.

[14] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," *CoRR*, vol. abs/1907.02189, 2019.

[15] H. Yang, M. Fang, and J. Liu, "Achieving linear speedup with partial worker participation in non-iid federated learning," in *ICLR*. OpenReview.net, 2021.

[16] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "Fedbn: Federated learning on non-iid features via local batch normalization," in *ICLR*. OpenReview.net, 2021.

[17] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowl. Based Syst.*, vol. 216, p. 106775, 2021.

[18] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, "Federated learning: A survey on enabling technologies, protocols, and applications," *IEEE Access*, vol. 8, pp. 140 699–140 725, 2020.

[19] S. A. Rahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani, "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5476–5497, 2021.

[20] K. M. J. Rahman, F. Ahmed, N. Akhter, M. Hasan, R. Amin, K. E. Aziz, A. K. M. M. Islam, M. S. H. Mukta, and A. K. M. N. Islam, "Challenges, applications and design aspects of federated learning: A survey," *IEEE Access*, vol. 9, pp. 124 682–124 700, 2021.

[21] H. Zhu, J. Xu, S. Liu, and Y. Jin, "Federated learning on non-iid data: A survey," *Neurocomputing*, vol. 465, pp. 371–390, 2021.

[22] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Gener. Comput. Syst.*, vol. 115, pp. 619–640, 2021.

[23] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 3347–3366, 2023.

[24] X. Yin, Y. Zhu, and J. Hu, "A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions," *ACM Comput. Surv.*, vol. 54, no. 6, pp. 131:1–131:36, 2022.

[25] Y. Zhan, J. Zhang, Z. Hong, L. Wu, P. Li, and S. Guo, "A survey of incentive mechanism design for federated learning," *IEEE Trans. Emerg. Top. Comput.*, vol. 10, no. 2, pp. 1035–1044, 2022.

[26] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for internet of things: A comprehensive survey," *IEEE Commun. Surv. Tutorials*, vol. 23, no. 3, pp. 1622–1658, 2021.

[27] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surv. Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[28] L. U. Khan, W. Saad, Z. Han, E. Hossain, and C. S. Hong, "Federated learning for internet of things: Recent advances, taxonomy, and open challenges," *IEEE Commun. Surv. Tutorials*, vol. 23, no. 3, pp. 1759–1799, 2021.

[29] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "A survey on federated learning for resource-constrained iot devices," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 1–24, 2022.

[30] D. C. Nguyen, Q. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, O. A. Dobre, and W. Hwang, "Federated learning for smart healthcare: A survey," *ACM Comput. Surv.*, vol. 55, no. 3, pp. 60:1–60:37, 2023.

[31] R. S. Antunes, C. A. da Costa, A. Küderle, I. A. Yari, and B. M. Eskofier, "Federated learning for healthcare: Systematic review and architecture proposal," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 4, pp. 54:1–54:23, 2022.

[32] B. Pfitzner, N. Steckhan, and B. Arnrich, "Federated learning in a medical context: A systematic literature review," *ACM Trans. Internet Techn.*, vol. 21, no. 2, pp. 50:1–50:31, 2021.

[33] J. Jiang, B. Kantarci, S. F. Oktug, and T. Soyata, "Federated learning in smart city sensing: Challenges and opportunities," *Sensors*, vol. 20, no. 21, p. 6230, 2020.

[34] X. Fu, B. Zhang, Y. Dong, C. Chen, and J. Li, "Federated graph machine learning: A survey of concepts, techniques, and applications," *SIGKDD Explor.*, vol. 24, no. 2, pp. 32–47, 2022.

[35] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS*, ser. Proceedings of Machine Learning Research, vol. 54. PMLR, 2017, pp. 1273–1282.

[36] V. Smith, C. Chiang, M. Sanjabi, and A. Talwalkar, "Federated multi-task learning," in *NIPS*, 2017, pp. 4424–4434.

[37] J. Chen, R. Monga, S. Bengio, and R. Józefowicz, "Revisiting distributed synchronous SGD," *CoRR*, vol. abs/1604.00981, 2016.

[38] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient federated learning via guided participant selection," in *OSDI*. USENIX Association, 2021, pp. 19–35.

[39] F. Fu, Y. Shao, L. Yu, J. Jiang, H. Xue, Y. Tao, and B. Cui, "Vf^{2}boost: Very fast vertical federated gradient boosting for cross-enterprise learning," in *SIGMOD*

Conference. ACM, 2021, pp. 563–576.

[40] S. Onn and M. Tennenholtz, "Determination of social laws for multi-agent mobilization," *Artif. Intell.*, vol. 95, no. 1, pp. 155–167, 1997.

[41] C. Niu, F. Wu, S. Tang, L. Hua, R. Jia, C. Lv, Z. Wu, and G. Chen, "Billion-scale federated learning on mobile clients: a submodel design with tunable privacy," in *MobiCom*. ACM, 2020, pp. 31:1–31:14.

[42] L. Li, H. Xiong, Z. Guo, J. Wang, and C. Xu, "Smartpc: Hierarchical pace control in real-time federated learning system," in *RTSS*. IEEE, 2019, pp. 406–418.

[43] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *NeurIPS*, 2019, pp. 14 747–14 756.

[44] J. Zhu and M. B. Blaschko, "R-GAP: recursive gradient attack on privacy," in *ICLR*. OpenReview.net, 2021.

[45] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *USENIX Security Symposium*. USENIX Association, 2020, pp. 1605–1622.

[46] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *AISTATS*, ser. Proceedings of Machine Learning Research, vol. 108. PMLR, 2020, pp. 2938–2948.

[47] S. Kim, J. Kim, D. Koo, Y. Kim, H. Yoon, and J. Shin, "Efficient privacy-preserving matrix factorization via fully homomorphic encryption: Extended abstract," in *AsiaCCS*. ACM, 2016, pp. 617–628.

[48] S. Sav, A. Pyrgelis, J. R. Troncoso-Pastoriza, D. Froelicher, J. Bossuat, J. S. Sousa, and J. Hubaux, "POSEIDON: privacy-preserving federated neural network learning," in *NDSS*. The Internet Society, 2021.

[49] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2017, pp. 19–38.

[50] D. Demmler, T. Schneider, and M. Zohner, "ABY - A framework for efficient mixed-protocol secure two-party computation," in *NDSS*. The Internet Society, 2015.

[51] K. A. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *CCS*. ACM, 2017, pp. 1175–1191.

[52] D. Chai, L. Wang, K. Chen, and Q. Yang, "Secure federated matrix factorization," *IEEE Intell. Syst.*, vol. 36, no. 5, pp. 11–20, 2021.

[53] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 3454–3469, 2020.

[54] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne, "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," *CoRR*, vol. abs/1711.10677, 2017.

[55] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," in *ICLR*. OpenReview.net, 2020.

[56] G. James, D. Witten, T. Hastie, R. Tibshirani *et al.*, *An introduction to statistical learning*. Springer, 2013, vol. 112.

[57] S. Lin, Y. Han, X. Li, and Z. Zhang, "Personalized federated learning towards communication efficiency, robustness and fairness," in *NeurIPS*, 2022.

[58] Y. Yu, A. Wei, S. P. Karimireddy, Y. Ma, and M. I. Jordan, "TCT: convexifying federated learning using bootstrapped neural tangent kernels," in *NeurIPS*, 2022.

[59] E. S. Lubana, C. I. Tang, F. Kawsar, R. P. Dick, and A. Mathur, "Orchestra: Unsupervised federated learning via globally consistent clustering," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 162. PMLR, 2022, pp. 14 461–14 484.

[60] A. Fallah, A. Mokhtari, and A. E. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," in *NeurIPS*, 2020.

[61] K. Pillutla, K. Malik, A. Mohamed, M. G. Rabbat, M. Sanjabi, and L. Xiao, "Federated learning with partial model personalization," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 162. PMLR, 2022, pp. 17 716–17 758.

[62] X. Lin, H. Chen, Y. Xu, C. Xu, X. Gui, Y. Deng, and Y. Wang, "Federated learning with positive and unlabeled data," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 162. PMLR, 2022, pp. 13 344–13 355.

[63] W. Jeong and S. J. Hwang, "Factorized-fl: Personalized federated learning with parameter factorization & similarity matching," in *NeurIPS*, 2022.

[64] Z. Li, H. Zhao, B. Li, and Y. Chi, "Soteriafl: A unified framework for private federated learning with communication compression," *CoRR*, vol. abs/2206.09888, 2022.

[65] S. Alam, L. Liu, M. Yan, and M. Zhang, "Fedrolex: Model-heterogeneous federated learning with rolling sub-model extraction," *CoRR*, vol. abs/2212.01548, 2022.

[66] H. Chen, M. Hao, H. Li, K. Chen, G. Xu, T. Zhang, and X. Zhang, "Guardhfl: Privacy guardian for heterogeneous federated learning," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 202. PMLR, 2023, pp. 4566–4584.

[67] K. Z. Liu, S. Hu, Z. S. Wu, and V. Smith, "On privacy and personalization in cross-silo federated learning," *CoRR*, vol. abs/2206.07902, 2022.

[68] X. Li and P. Li, "Analysis of error feedback in federated non-convex optimization with biased compression: Fast convergence and partial participation," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 202. PMLR, 2023, pp. 19 638–19 688.

[69] P. Mai and Y. Pang, "Vertical federated graph neural network for recommender system," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 202. PMLR, 2023, pp. 23 516–23 535.

[70] J. Baek, W. Jeong, J. Jin, J. Yoon, and S. J. Hwang,

"Personalized subgraph federated learning," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 202. PMLR, 2023, pp. 1396–1415.

[71] Y. Wu, S. Zhang, W. Yu, Y. Liu, Q. Gu, D. Zhou, H. Chen, and W. Cheng, "Personalized federated learning under mixture of distributions," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 202. PMLR, 2023, pp. 37 860–37 879.

[72] F. Zhang, X. Liu, S. Lin, G. Wu, X. Zhou, J. Jiang, and X. Ji, "No one idles: Efficient heterogeneous federated learning with parallel edge and server computation," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 202. PMLR, 2023, pp. 41 399–41 413.

[73] X. Ma, J. Zhu, Z. Lin, S. Chen, and Y. Qin, "A state-of-the-art survey on solving non-iid data in federated learning," *Future Gener. Comput. Syst.*, vol. 135, pp. 244–258, 2022.

[74] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *MLSys*. mlsys.org, 2020.

[75] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," in *NDSS*. The Internet Society, 2021.

[76] A. Kolluri, T. Baluta, and P. Saxena, "Private hierarchical clustering in federated networks," in *CCS*. ACM, 2021, pp. 2342–2360.

[77] M. Naseri, Y. Han, E. Mariconti, Y. Shen, G. Stringhini, and E. D. Cristofaro, "CERBERUS: exploring federated prediction of security events," in *CCS*. ACM, 2022, pp. 2337–2351.

[78] A. R. Chowdhury, C. Guo, S. Jha, and L. van der Maaten, "Eiffel: Ensuring integrity for federated learning," in *CCS*. ACM, 2022, pp. 2535–2549.

[79] D. Pasquini, D. Francati, and G. Ateniese, "Eluding secure aggregation in federated learning via model inconsistency," in *CCS*. ACM, 2022, pp. 2429–2443.

[80] S. Maddock, G. Cormode, T. Wang, C. Maple, and S. Jha, "Federated boosted decision trees with differential privacy," in *CCS*. ACM, 2022, pp. 2249–2263.

[81] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. Ramage, "Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning," in *SP*. IEEE, 2022, pp. 1354–1371.

[82] M. Rosenberg, M. Maller, and I. Miers, "Snarkblock: Federated anonymous blocklisting from hidden common input aggregate proofs," in *SP*. IEEE, 2022, pp. 948–965.

[83] C. Fu, X. Zhang, S. Ji, J. Chen, J. Wu, S. Guo, J. Zhou, A. X. Liu, and T. Wang, "Label inference attacks against vertical federated learning," in *USENIX Security Symposium*. USENIX Association, 2022, pp. 1397–1414.

[84] T. Stevens, C. Skalka, C. Vincent, J. Ring, S. Clark, and J. P. Near, "Efficient differentially private secure aggregation for federated learning via hardness of learning with errors," in *USENIX Security Symposium*. USENIX Association, 2022, pp. 1379–1395.

[85] T. D. Nguyen, P. Rieger, H. Chen, H. Yalame, H. Möllering, H. Fereidooni, S. Marchal, M. Miettinen, A. Mirhoseini, S. Zeitouni, F. Koushanfar, A. Sadeghi, and T. Schneider, "FLAME: taming backdoors in federated learning," in *USENIX Security Symposium*. USENIX Association, 2022, pp. 1415–1432.

[86] Z. Zhang, D. Dong, Y. Ma, Y. Ying, D. Jiang, K. Chen, L. Shou, and G. Chen, "Refiner: A reliable incentive-driven federated learning system powered by blockchain," *Proc. VLDB Endow.*, vol. 14, no. 12, pp. 2659–2662, 2021.

[87] Y. Liu, W. Wu, L. Flokas, J. Wang, and E. Wu, "Enabling sql-based training data debugging for federated learning," *Proc. VLDB Endow.*, vol. 15, no. 3, pp. 388–400, 2021.

[88] Y. Yuan, D. Ma, Z. Wen, Z. Zhang, and G. Wang, "Subgraph matching over graph federation," *Proc. VLDB Endow.*, vol. 15, no. 3, pp. 437–450, 2021.

[89] J. Liu, J. Lou, L. Xiong, J. Liu, and X. Meng, "Projected federated averaging with heterogeneous differential privacy," *Proc. VLDB Endow.*, vol. 15, no. 4, pp. 828–840, 2021.

[90] Z. Li, B. Ding, C. Zhang, N. Li, and J. Zhou, "Federated matrix factorization with privacy guarantee," *Proc. VLDB Endow.*, vol. 15, no. 4, pp. 900–913, 2021.

[91] F. Fu, X. Miao, J. Jiang, H. Xue, and B. Cui, "Towards communication-efficient vertical federated learning training via cache-enabled local update," *Proc. VLDB Endow.*, vol. 15, no. 10, pp. 2111–2120, 2022.

[92] E. Bao, Y. Zhu, X. Xiao, Y. Yang, B. C. Ooi, B. H. M. Tan, and K. M. M. Aung, "Skellam mixture mechanism: a novel approach to federated learning with differential privacy," *Proc. VLDB Endow.*, vol. 15, no. 11, pp. 2348–2360, 2022.

[93] X. Li, Y. Hu, W. Liu, H. Feng, L. Peng, Y. Hong, K. Ren, and Z. Qin, "Opboost: A vertical federated tree boosting framework based on order-preserving desensitization," *Proc. VLDB Endow.*, vol. 16, no. 2, pp. 202–215, 2022.

[94] F. Fu, H. Xue, Y. Cheng, Y. Tao, and B. Cui, "Blindfl: Vertical federated machine learning without peeking into your data," in *SIGMOD Conference*. ACM, 2022, pp. 1316–1330.

[95] Z. Xiang, T. Wang, W. Lin, and D. Wang, "Practical differentially private and byzantine-resilient federated learning," *Proc. ACM Manag. Data*, vol. 1, no. 2, pp. 119:1–119:26, 2023.

[96] R. Fu, Y. Wu, Q. Xu, and M. Zhang, "FEAST: A communication-efficient federated feature selection framework for relational data," *Proc. ACM Manag. Data*, vol. 1, no. 1, pp. 107:1–107:28, 2023.

[97] Z. Li, T. Wang, and N. Li, "Differentially private vertical federated clustering," *Proc. VLDB Endow.*, vol. 16, no. 6, pp. 1277–1290, 2023.

[98] W. Huang, J. Liu, T. Li, T. Huang, S. Ji, and J. Wan, "Feddsr: Daily schedule recommendation in a federated deep reinforcement learning framework," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 3912–3924, 2023.

[99] Z. Pan, L. Hu, W. Tang, J. Li, Y. He, and Z. Liu,

"Privacy-preserving multi-granular federated neural architecture search - A general framework," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 3, pp. 2975–2986, 2023.

[100] L. Zhang, T. Zhu, P. Xiong, W. Zhou, and P. S. Yu, "A robust game-theoretical federated learning framework with joint differential privacy," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 3333–3346, 2023.

[101] Y. Qian, C. Tan, D. Ding, H. Li, and N. Mamoulis, "Fast and secure distributed nonnegative matrix factorization," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 2, pp. 653–666, 2022.

[102] W. Dai, X. Jiang, L. Bonomi, Y. Li, H. Xiong, and L. Ohno-Machado, "VERTICOX: vertically distributed cox proportional hazards model using the alternating direction method of multipliers," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 2, pp. 996–1010, 2022.

[103] C. T. Dinh, N. H. Tran, and T. D. Nguyen, "Personalized federated learning with moreau envelopes," in *NeurIPS*, 2020.

[104] E. Diao, J. Ding, and V. Tarokh, "Heterofl: Computation and communication efficient federated learning for heterogeneous clients," in *ICLR*. OpenReview.net, 2021.

[105] T. Yoon, S. Shin, S. J. Hwang, and E. Yang, "Fedmix: Approximation of mixup under mean augmented federated learning," in *ICLR*. OpenReview.net, 2021.

[106] B. Sun, H. Huo, Y. Yang, and B. Bai, "Partialfed: Cross-domain personalized federated learning via partial initialization," in *NeurIPS*, 2021, pp. 23 309–23 320.

[107] Y. Park, D. Han, D. Kim, J. Seo, and J. Moon, "Few-round learning for federated learning," in *NeurIPS*, 2021, pp. 28 612–28 622.

[108] G. Lee, M. Jeong, Y. Shin, S. Bae, and S. Yun, "Preservation of the global knowledge by not-true distillation in federated learning," in *NeurIPS*, 2022.

[109] H. Wang, M. Yurochkin, Y. Sun, D. S. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," in *ICLR*. OpenReview.net, 2020.

[110] S. J. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," in *ICLR*. OpenReview.net, 2021.

[111] D. Rothchild, A. Panda, E. Ullah, N. Ivkin, I. Stoica, V. Braverman, J. Gonzalez, and R. Arora, "Fetchsgd: Communication-efficient federated learning with sketching," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 119. PMLR, 2020, pp. 8253–8265.

[112] J. Zhang, X. Cheng, W. Wang, L. Yang, J. Hu, and K. Chen, "FLASH: towards a high-performance hardware acceleration architecture for cross-silo federated learning," in *NSDI*. USENIX Association, 2023, pp. 1057–1079.

[113] D. Chai, L. Wang, J. Zhang, L. Yang, S. Cai, K. Chen, and Q. Yang, "Practical lossless federated singular vector decomposition over billion-scale data," in *KDD*. ACM, 2022, pp. 46–55.

[114] C. Boettiger, "An introduction to docker for reproducible research," *ACM SIGOPS Oper. Syst. Rev.*, vol. 49, no. 1, pp. 71–79, 2015.

[115] D. Chai, L. Wang, L. Yang, J. Zhang, K. Chen, and Q. Yang, "Fedeval: A holistic evaluation framework for federated learning," *arXiv preprint arXiv:2011.09655*, 2020.

[116] K. A. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *CCS*. ACM, 2017, pp. 1175–1191.

[117] C. Chen, J. Zhou, L. Wang, X. Wu, W. Fang, J. Tan, L. Wang, A. X. Liu, H. Wang, and C. Hong, "When homomorphic encryption marries secret sharing: Secure large-scale sparse logistic regression and applications in risk control," in *KDD*. ACM, 2021, pp. 2652–2662.

[118] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients - how easy is it to break privacy in federated learning?" in *NeurIPS*, 2020.

[119] H. Weng, J. Zhang, F. Xue, T. Wei, S. Ji, and Z. Zong, "Privacy leakage of real-world vertical federated learning," *CoRR*, vol. abs/2011.09290, 2020.

[120] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *CCS*. ACM, 2015, pp. 1322–1333.

[121] S. Hidano, T. Murakami, S. Katsumata, S. Kiyomoto, and G. Hanaoka, "Model inversion attacks for online prediction systems: Without knowledge of non-sensitive attributes," *IEICE Trans. Inf. Syst.*, vol. 101-D, no. 11, pp. 2665–2676, 2018.

[122] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2017, pp. 3–18.

[123] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *IEEE Symposium on Security and Privacy*. IEEE, 2019, pp. 739–753.

[124] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. A. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *OSDI*. USENIX Association, 2016, pp. 265–283.

[125] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, and N. D. Lane, "Flower: A friendly federated learning research framework," *CoRR*, vol. abs/2007.14390, 2020.

[126] S. Wei, Y. Tong, Z. Zhou, and T. Song, "Efficient and fair data valuation for horizontal federated learning," in *Federated Learning*, ser. Lecture Notes in Computer Science. Springer, 2020, vol. 12500, pp. 139–152.

[127] Y. Liu, T. Fan, T. Chen, Q. Xu, and Q. Yang, "FATE: an industrial grade platform for collaborative learning with data protection," *J. Mach. Learn. Res.*, vol. 22, pp. 226:1–226:6, 2021.

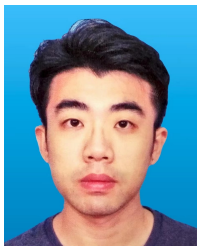
[128] C. He, S. Li, J. So, M. Zhang, H. Wang, X. Wang,

P. Vepakomma, A. Singh, H. Qiu, L. Shen, P. Zhao, Y. Kang, Y. Liu, R. Raskar, Q. Yang, M. Annavaram, and S. Avestimehr, "Fedml: A research library and benchmark for federated machine learning," *CoRR*, vol. abs/2007.13518, 2020.

[129] F. Lai, Y. Dai, S. S. V. Singapuram, J. Liu, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Fedscale: Benchmarking model and system performance of federated learning at scale," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 162. PMLR, 2022, pp. 11 814–11 827.

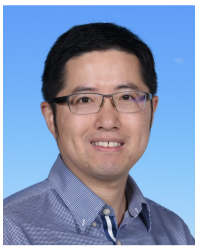
Di Chai is a Ph.D. student in computer science and engineering at Hong Kong University of Science and Technology (HKUST). He got his master degree of science from HKUST in 2018. His research interests include federated learning and privacy-preserving machine learning.

Leye Wang is an assistant professor at Key Lab of High Confidence Software Technolo- gies (Peking University), MOE, and School of Computer Science, Peking University, China. He received a Ph.D. in computer science from TELECOM SudParis and University Paris 6, France, in 2016. He was a postdoc researcher with Hong Kong University of Science and Technology. His research interests include ubiquitous computing, mobile crowdsensing, and urban computing.



Liu Yang is a PhD student of computer science at iSINGLab, Hong Kong University of Science and Technology (HKUST). He is under supervision of Prof. Qiang Yang and Prof. Kai Chen. His research interests include federated learning and recommendation system. Before pursuing PhD, he received his BEng and MSc from Sun Yat-sen University and HKUST, respectively.

Junxue Zhang obtained his Ph.D. from computer science & engineering at iSINGLab, Hong Kong University of Science and Technology, supervised by Prof. Kai CHEN. Before joining HKUST, he received his BS and MS from Southeast University. His research interests are data center networking, machine learning systems and privacy-preserving computation. His research work has been published in many top conferences and journals such as SIGCOMM, NSDI, CoNEXT, TON, *etc*.



Kai Chen is the Professor with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong. He received his Ph.D. degree in Computer Science from Northwestern University, Evanston, IL, USA in 2012. His research interests include data center networking, machine learning systems and privacy-preserving computing.

Qiang Yang is a Fellow of Canadian Academy of Engineering (CAE) and Royal Society of Canada (RSC), Chief Artificial Intelligence Officer of WeBank, a Chair Professor of Computer Science and Engineering Department at Hong Kong University of Science and Technology (HKUST). He is the Conference Chair of AAAI-21, the Honorary Vice President of Chinese Association for Artificial Intelligence(CAAI) , the President of Hong Kong Society of Artificial Intelligence and Robotics(HKSAIR) and the President of Investment Technology League (ITL). He is a fellow of AAAI, ACM, CAAI, IEEE, IAPR, AAAS. He was the Founding Editor in Chief of the ACM Transactions on Intelligent Systems and Technology (ACM TIST) and the Founding Editor in Chief of IEEE Transactions on Big Data (IEEE TBD). He received the ACM SIGKDD Distinguished Service Award in 2017. He had been the Founding Director of the Huawei's Noah's Ark Research Lab between 2012 and 2015, the Founding Director of HKUST's Big Data Institute, the Founder of 4Paradigm and the President of IJCAI (2017-2019). His research interests are artificial intelligence, machine learning, data mining and planning.