An Interactive Framework for Implementing Privacy-Preserving Federated Learning: Experiments on Large Language Models

Kasra Ahmadi
University of South Florida
ahmadi1@usf.edu

Rouzbeh Behnia
University of South Florida
behnia@usf.edu

Reza Ebrahimi
University of South Florida
ebrahimim@usf.edu

Mehran Mozaffari Kermani
University of South Florida
mehran2@usf.edu

Jeremiah Birrell
Texas State University
jbirrell@txstate.edu

Jason Pacheco
University of Arizona
pachecoj@cs.arizona.edu

Attila A. Yavuz
University of South Florida
attilaayavuz@usf.edu

***Abstract*—Federated learning (FL) enhances privacy by keeping user data on local devices. However, emerging attacks have demonstrated that the updates shared by users during training can reveal significant information about their data. Differential Privacy (DP) is considered the gold standard for safeguarding user data. However, DP guarantees are highly conservative, providing worst-case privacy guarantees. This can result in overestimating privacy needs, which may compromise the model's accuracy. Additionally, interpretations of these privacy guarantees have proven to be challenging in different contexts. This is further exacerbated when other factors, such as the number of training iterations, data distribution, and specific application requirements, can add further complexity to this problem. In this work, we proposed a framework that integrates a human entity as a privacy practitioner to determine an optimal trade-off between the model's privacy and utility. Our framework is the first to address the variable memory requirement of existing DP methods in FL settings, where resource-limited devices (e.g., cell phones) can participate. To support such settings, we adopt a recent DP method with fixed memory usage to ensure scalable private FL.**

We evaluated our proposed framework by fine-tuning a BERT-based LLM model using the GLUE dataset (a common approach in literature), leveraging the new accountant, and employing diverse data partitioning strategies to mimic real-world conditions. As a result, we achieved stable memory usage, with an average accuracy reduction of 1.33% for $\epsilon = 10$ and 1.9% for $\epsilon = 6$, when compared to the state-of-the-art DP accountant which does not support fixed memory usage.

***Index Terms*—Differential Privacy, Federated Learning, Privacy Cost, Fine-Tuning, LLM**

1. Introduction

Foundation models have achieved superior performance across various domains, including finance, healthcare, and cybersecurity. These models are trained on public data for general tasks and later fine-tuned by different industries for application-specific downstream tasks. The performance of these models relies heavily on the quality and diversity of their training data. This is especially important for sensitive applications where the performance and robustness of the model are crucial. However, for these applications, access to distributed and diverse data is often restricted through internal privacy policies or regulations such as HIPAA. This limitation is widely acknowledged as a primary barrier to training and fine-tuning models in the medical field [1]. Federated learning (FL) mitigates this issue by maintaining the locality of the data and only requiring the participating clients to share their local model (trained on their data) with a central server.

While maintaining data locality can contribute to data privacy, recent attacks (e.g., [2], [3]), demonstrated that updates shared with the server can leak significant information about the user dataset. These attacks can be categorized as follows, sorted by their privacy implications. 1) Membership Inference [4], [5]: The attacker's goal is to determine whether a specific data point was part of the dataset used to train the model. 2) Model inversion [6]: A more severe privacy attack, where the attacker adversary attempts to recover information about the training data (i.e., approximate reconstruction). 3) Training Data Extraction [7]: This is the most potent privacy attack where the attacker attempts to recover the original training samples precisely.

Differential privacy (DP) has long been the gold standard for mitigating these attacks [8]. This is achieved by injecting a measured noise into model gradients to minimize the influence of any single data point on the model parameters. The magnitude of the noise is determined based on the desired privacy guarantee, measured by the privacy cost (ϵ, δ), which serves as a key parameter in DP algorithms.

In the FL setting, DP can be applied in different stages and by different parties to protect against data privacy attacks. Depending on the privacy goals, often dictated by privacy policies or regulation requirements (e.g., HIPAA), DP can be applied by the clients adding noise to their gradient before sending it to the central server; this is often referred to as local DP (LDP). Alternatively, in central DP

(CDP), the central server injects noise into the aggregated global model at the end of each training iteration. In CDP, the server has access to gradients sent by the users and is, therefore, assumed to be trusted. However, in applications that involve sensitive user data, relying on a trusted server can pose significant privacy risks and potentially violate privacy policies and regulations [9]. LDP can provide privacy at different stages of the model's life cycle (from model training to deployment) without assuming a trusted server.

DP methods provide a worst-case privacy guarantee, which could result in utility loss due to the excessive added noise. In practice, the worst-case DP guarantees may not always be necessary for certain applications, potentially sacrificing performance by overprotecting the model. For instance, in some applications, user participation might be public (e.g., social media), and only the user data should be protected. In such cases, the privacy goal is to defend against data reconstruction attacks to protect user data, which often require significantly less noise compared to the noise needed to defend against membership inference attacks [2].

Another critical aspect of privacy-preserving FL is the selection of the DP method and the training parameters. In highly distributed FL applications (e.g., [10]), the choice of DP method and parameters can significantly impact user participation. Certain DP methods tend to overestimate the necessary noise or impose a high computational overhead, making them impractical for low-end devices, particularly those from underrepresented groups. For example, as recently highlighted in [11], most of the widely adopted DP methods, such as Renyi Differential Privacy (`RDP`) [12], rely on Poisson subsampling which results in producing variable size minibatches. This directly affects the machine's memory usage (Figure 1). While this might be manageable in traditional centralized settings, where the model training happens on powerful servers, in distributed FL applications with low-end devices, this can lead to an out-of-memory (OOM) error, potentially preventing certain devices from participating in the training process. This can directly affect the representation of different user groups in the training process, impacting the data diversity and potentially the model fairness [13], [14]. Lastly, the selection of FL parameters and the AI models to be trained in the federated setting can significantly impact the performance and utility of the trained model and user participation rates. For example, while a larger batch size may reduce the noise required for differential privacy, it can also introduce additional performance overhead for users. These challenges highlight a key research gap.

To address this, in this work, we present a framework for Federated Learning Implementation with Privacy (`FLIP`) that integrates a privacy practitioner into the training process to guide privacy-aware decision-making. The practitioner's role is to help guide the selection of these parameters based on application requirements, AI model, privacy objectives, system specifications, resource constraints, and user participation dynamics. This approach ensures a more informed, context-sensitive decision-making process, optimizing both privacy protection and overall system performance and leading to the training of robust AI models. Figure 2 provides a high-level overview of `FLIP`.

1.1. Our Contributions

The contributions of our work as as follows. To our knowledge, our work is the first to empirically highlight the importance and effect of this selection process in the private FL training of AI models. Our contributions are as follows.

• **Adoption of a Privacy Practitioner:** Our work is the first to empirically highlight the significance and impact of selecting DP and FL parameters in privacy-preserving federated learning (FL) for AI models. We introduce a novel framework where a privacy practitioner assists in tuning these parameters based on factors such as privacy requirements, performance trade-offs, computational constraints, and system specifications.

•**Adopting a Fixed Mini-Batch DP Method:** `FLIP` is the first privacy-preserving FL framework to: 1) highlight the side effects of variable mini-batch sizes in existing DP methods (e.g., `RDP`) on FL model training, 2) adopt a fixed mini-batch approach (i.e., `FSRDP`) to ensure stable memory usage throughout training, and 3) empirically compare the impact of this choice on model accuracy.

• **Comprehensive Study on DP, FL Parameters, and Data Distribution:** To our knowledge, `FLIP` is the first framework to study the impacts of various DP and FL parameters and the data distributions among the users in FL settings on model performance. We conduct our experiments by fine-tuning Large Language Models (LLMs) on four well-known natural language processing tasks from the GLUE dataset [15]. Fine-tuning LLMs in the context of studying the impact of DP on model performance is a well-established approach [16], [17]. For example, simulating our framework in fine-tuneing BERT (with 109M parameters) [18] with two different target privacy costs, $\epsilon = 6$ and $\epsilon = 10$, while adopting `FSRDP` as our DP method, incurs up to a 5% accuracy loss compared to the non-private model. However, this gap can be reduced to as low as 2% when optimized parameters and data distribution are enforced by the practitioner. Our framework is open-source for public verification and testing using the following link:

https://github.com/KasraAhmadi/FL-Privacy-LLM

2. Preliminaries

In this section, we examine three key areas of the literature: (1) FL, a distributed machine learning approach that trains models across multiple devices; (2) Differentially private deep learning, a comprehensive framework that ensures rigorous privacy during the learning process; and (3) The benefits of using differentially private stochastic gradient descent with fixed-size minibatches compared to Poisson-subsampled `RDP`.

2.1. Federated Learning

Federated Learning (FL) enables the distributed training of a central model, with contributions from a set of clients who each train a local copy of the model using their own data [19]. FL considers a central server who aggregates the updates shared by the clients. A generic FL system consists of a central server and k clients. Each client C_i holds a local dataset D_i, where $i \in \{1, 2, \ldots, k\}$. The server's objective is to train a model using data distributed across the k clients. When a client actively participates in local training, it aims to optimize a vector $\mathbf{w}$ for an AI model by minimizing a specified loss function. The server then aggregates the model weights received from the k clients as follows:

$$\mathrm{w} = \sum_{i=1}^{k} p_i \mathrm{w}_i$$

Here, w_i represents the parameter vector trained by the i-th client, and w is the aggregated parameter vector at the server. k denotes the total number of clients, while $p_i = \frac{|D_i|}{|D|} \geq 0$ satisfies $\sum_{i=1}^{k} p_i = 1$, with $|D| = \sum_{i=1}^{k} |D_i|$ being the total number of data samples across all clients. This optimization problem can be expressed as:

$$\mathrm{w}^* = \mathrm{argmin}_{\mathrm{w}} \sum_{i=1}^{k} p_i F_i(\mathrm{w})$$

where $F_i(w)$ represents the local loss function for the i-th client.

In the FL process, the k clients work together to train a machine-learning model with the assistance of a server. After several rounds of local training and updates exchanged between the server and the clients, the solution to the optimization problem is expected to converge to the globally optimal learning model.

2.2. Differential Privacy

Differential privacy [8], [20] is a rigorous privacy framework that effectively mitigates the privacy risks associated with deep learning [21]. The primary distinction between DP-based deep learning and standard deep learning lies in whether the gradient is released with privacy guarantees.

Definition 1: A randomized algorithm M is (ϵ, δ)-differentially private if, for any two neighboring datasets S and S' (i.e., S' can be obtained by adding or removing a single data point from S), and for any event E, the following condition holds :

$$P[M(S) \in E] \leq e^{\epsilon} P[M(S') \in E] + \delta.$$

We consider the (ϵ, δ)-DP definition, where smaller values of ϵ and δ indicate a stronger privacy guarantee.

Differentially Private Stochastic Gradient Descent (DP-SGD) [21] ensures differential privacy by introducing noise during the training process of machine learning models. DP-SGD modifies the standard mini-batch SGD algorithm by adding two additional steps:

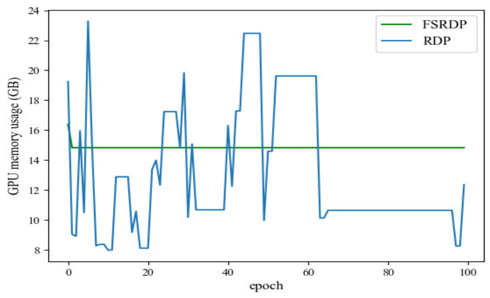

Figure 1. Comparison of `FSRDP` and `RDP` accountant memory usage with a batch size of 120 and a dataset size of 50,000 per training epoch [11].

- Gradient Clipping: For each per-example gradient $g(x_i)$, where x_i is a data point in the selected mini-batch, clip the l_2-norm to a predefined threshold C:

$$g(x_i) \leftarrow \frac{g(x_i)}{\max(1, \|g(x_i)\|_2 / C)}.$$

- Noise Addition: Add Gaussian noise to the aggregated gradient of the mini-batch, where L is the mini-batch size and σ is the noise scale.:

$$g \leftarrow \frac{1}{L} \left(\sum_i g(x_i) + \mathcal{N}(0, \sigma^2 C^2) \right),$$

2.3. `RDP` and `FSRDP`

DP-SGD enables the use of a technique known as the moments accountant to sequentially monitor privacy leakage. This approach is encompassed by Rényi Differential Privacy (`RDP`) [12], a relaxed version of standard Differential Privacy [22]. `RDP` is widely applied in private deep learning and is incorporated into modern DP libraries like Opacus [23]. The underlying computation in `RDP` relies on subsampling techniques that use a privacy amplification lemma to enhance the privacy guarantees provided by the added noise.

While there have been previous attempts to compute privacy costs with a fixed mini-batch size, such as the works of Balle et al. [24] for (ϵ, δ)-DP and Wang et al. [25] for `RDP`, these approaches had significant shortcomings. The earlier (ϵ, δ)-DP methods did not compose easily over multiple training steps, often leading to privacy leakage, making them impractical for iterative processes like SGD. Similarly, Wang's `RDP` accountant was not as tight, resulting in suboptimal privacy bounds. In contrast, `FSRDP` [11] is the first privacy accountant capable of computing privacy costs with fixed-size mini-batches while achieving much tighter bounds. In fact, `FSRDP` is very close to the theoretical lower bound in many practical cases, offering significantly improved privacy guarantees over previous RDP-based methods.

This offers a significant advantage of consistent memory usage compared to the variable-sized mini-batches in Poisson subsampling. While the results in [11] were purely theoretical, in this paper, by highlighting the importance of fixed memory usage in FL settings, we adopt their accountant, and after conducting extensive experiments, we show that for certain applications and data distributions, the accuracy loss, compared to `RDP` is insignificant. Figure 1 depicts the memory consumption of `FSRDP` and `RDP` accountants. In contrast to `RDP`, `FSRDP` maintains a constant memory footprint throughout the training process.

3. Proposed Framework

The goal of our framework, `FLIP`, is to optimize model performance given the required privacy requirements. In our proposed framework, the privacy practitioner (human expertise) is crucial in improving security and privacy through collaboration with clients and the privacy engine. The suggested data flow architecture, as depicted in Figure 2, includes four entities: Requirement, Privacy Practitioner, Clients, and Privacy Engine.

3.1. Clients

A client is a device, user, or entity involved in the decentralized training process, where it locally stores and processes its own private data. Clients contribute to training a shared machine learning model using their private data and periodically transmit model updates to the privacy engine. Since in `FLIP` we aim to achieve full-stack privacy (during training and after deployment), DP is implemented on the client side. Therefore, after receiving the FL and DP parameters from the practitioner, aside from model training, the client has to compute and inject the noise to its update.

3.2. Requirements

In Federated Learning (FL), clients collaboratively train a centralized model, which may either be deployed for their own use or commercialized by the entity that organizes the federation. This entity facilitates client participation, often in exchange for a service or financial compensation. Therefore, the selection of requirements is determined either by a coalition of clients or by the organizing entity overseeing the federation. These requirements can be categorized into privacy and learning process requirements.

3.2.1. Privacy Requirements.

- **Target privacy requirement:** This key parameter is central to calculating the privacy cost (ϵ) and helps the practitioner determine the optimal trade-off between individual privacy and model accuracy. The selection of ϵ varies by application, requiring a trade-off, as lower values of ϵ enhance privacy but often come at the cost of reduced model accuracy.
  The interpretation of ϵ depends on many factors, including the number of training iterations (both local and global), data type, AI model, and other system specific parameters. However, this interpretation has proven to be very challenging [26], as ϵ does not directly translate to an intuitive measure of privacy risk across different settings . Instead, they can specify a privacy goal, such as mitigating the membership inference attack (MIA) or reconstruction attack, and a privacy practitioner can determine the suitable ϵ value accordingly. Alternatively, if the DP is solely being implemented to meet certain requirements, enforced by regulations, the privacy cost could be computed by the privacy practitioner based on these requirements.
  MIA and reconstruction privacy goals are applied differently across various real-world scenarios. For example, in financial applications, MIA is generally not a primary concern. This is because certain information, such as the knowledge of which bank a person has a credit card with, is often considered public and not sensitive. However, more private details like one's Social Security Number (SSN) and account balance are crucial to protect. In such cases, the goal is not necessarily to prevent MIA but to guard against reconstruction attacks, where an adversary might attempt to reconstruct sensitive information about an individual from available data.

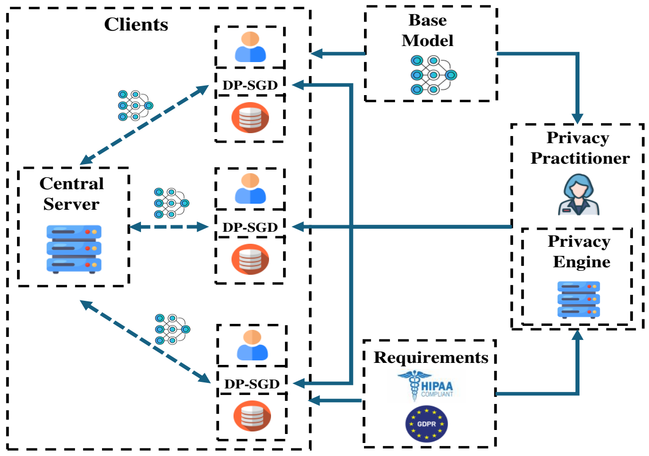

Figure 2. The data flow architecture for the proposed framework

3.2.2. Learning Process Requirements.

- **Number of Clients:** The number of clients in the FL requires careful consideration. A larger client base increases diversity and robustness but necessitates efficient communication and aggregation strategies to address scalability issues. On the other hand, fewer clients make management simpler but may reduce the model's representativeness and resilience.
- **Data Distribution:** In some applications, no information about user data can be shared with the privacy practitioner. However, when feasible, insights into data distribution can significantly aid the practitioner in setting an effective privacy target and

ultimately computing an appropriate ϵ. As demonstrated in our experiments, when data is uniformly distributed across all users, the model generally achieves higher accuracy and can tolerate a lower ϵ while maintaining utility. Conversely, maintaining high accuracy under strict privacy constraints becomes more challenging in heterogeneous settings where data distributions vary significantly across clients. Certain data partitions may hold greater significance in the training process in such cases, requiring the privacy engine to incorporate adaptive strategies that account for these constraints, balancing privacy preservation with model performance.
In addition to aiding in the selection of the appropriate ϵ, knowledge of data distribution can also enable a tailored client selection during each training round. By carefully choosing clients with better uniform data distributions, the practitioner can improve overall data diversity across participants, model performance, and privacy. This tailored client selection approach can also help address other challenges posed by heterogeneous data distributions, ensuring a more balanced and effective training process in federated learning while maintaining privacy targets.
- **Client's Computation Constraints:** Computation constraints on the client side are another key concern in federated learning, as clients often consist of resource-limited devices such as smartphones, IoT devices, or edge devices. These limitations significantly impact the efficiency and feasibility of the learning process. In this context, we identify the benefit of the `FSRDP` accountant [11]. Although not explicitly designed for resource-constrained devices, it is particularly valuable in our framework because it uses constant memory when calculating the noise for a given ϵ, making it well-suited for these applications. Furthermore, the batch size, a parameter that directly influences both the training duration and the DP-SGD algorithm, is closely tied to the available memory on each client. It is the responsibility of the privacy practitioner to determine the batch size based on the memory capacity of individual clients (that can be provided in the requirements).

3.3. Privacy Practitioner

- **Security Parameters Calculation:** By utilizing its local privacy engine, the practitioner must determine the appropriate ϵ value based on the target security constraints, which are influenced by the need to mitigate potential risks such as membership inference attacks (MIA) or reconstruction attacks, as well as to meet the privacy requirements set for the application. After selecting the suitable ϵ value, the practitioner also chooses the appropriate differential privacy accountant. For memory-constrained applications, in our framework, `FSRDP` is selected, as it ensures constant memory usage but may result in slightly lower model accuracy. For cases where higher model accuracy is desired, `RDP` is chosen, which may involve non-constant memory usage. Once the ϵ and δ (where $\delta = \frac{1}{|D_i|}$, representing the inverse of the dataset size) are determined, these parameters are provided clients. The client then utilizes their privacy engine, which calculates the necessary noise to be added to the client's gradient.
- **Set Batch size:** The batch size affects the efficiency and effectiveness of DP-SGD by determining how many samples are used in each model update. The ideal batch size strikes a balance between model security, training speed, memory usage, and overall performance. The privacy practitioner sets this parameter according to the available memory on the client side and the desired model performance.

3.4. Privacy Engine

- **Noise Calculation:** The privacy engine calculates the necessary noise using the specified accountant, ϵ, batch size, and δ values. This noise is then injected into the client's gradient via the DP-SGD algorithm.
- **Requirement Adherence Tracking:** In certain scenarios, the framework may fail to achieve the required accuracy due to factors such as the client's data partition, target security level, and target minimum accuracy. This limitation is primarily caused by the small size of partitioned data, where adding high noise to a gradient computed on a limited dataset can result in reduced accuracy. In such cases, the privacy engine generates a warning to indicate that the desired accuracy cannot be met. To address this, clients can either expand their data partitions or increase memory resources to adopt a different accountant, such as `RDP` instead of `FSRDP`, to reduce the noise added during the DP-SGD algorithm.

4. Experiments

4.1. Experiment Setup

Our work was conducted using a single NVIDIA RTX 6000 Ada Generation GPU, equipped with 18,176 CUDA cores and 48GB of dedicated memory. We selected the pre-trained BERT base model (cased), which contains 109 million parameters, to ensure that LLM loading and fine-tuning could be accommodated within the internal memory. The model is available at Hugging Face hub[1].
We utilized the Flower framework[2] to simulate a federated learning environment for fine-tuning LLMs, while the Transformers library was used for tasks such as training, tokenization, and evaluation. The GLUE dataset[3], accessed via Hugging Face, was used for data loading. In particular, we

1. https://huggingface.co/google-bert/bert-base-cased
2. https://flowerai.net/
3. https://huggingface.co/datasets/nyu-mll/glue

utilized four specific datasets from the GLUE benchmark, which are described below.

- QNLI: The Question-Answering Natural Language Inference (QNLI) dataset, sourced from Wikipedia, comprises 110,400 question-paragraph pairs. Each paragraph contains only one sentence that answers the associated question. The task for the language model is to identify whether a given sentence contains the correct answer to the question.
- QQP [27]: The Quora Question Pairs (QQP) dataset contains more than 400,000 pairs of questions, each labeled to show whether the questions are semantically equivalent, meaning they are paraphrases of each other. The task for the language model is to determine if one question is a paraphrase of the other.
- SST2 [28]: The Stanford Sentiment Treebank (SST2) dataset consists of 68,800 sentences from movie reviews, each annotated with its sentiment. The task for the language model is to classify the sentiment of a given sentence as either positive or negative.

We defined the training and testing splits for each dataset as follows: QNLI with 105,000 samples for training and 5,460 for testing; QQP with 364,000 samples for training and 391,000 for testing; and SST-2 with 67,000 samples for training and 1,820 for testing.

We simulate a federated learning environment with 4 distinct clients, each assigned its own training and testing dataset. The setup includes 5 training rounds, a learning rate of $2e-5$, and a batch size of 550. We adopt the FedAvg algorithm of McMahan et al. [19]

To replicate real-world data generation across decentralized devices, we distribute the training and testing data among clients using 4 distinct partitioning strategies: Iid, Linear, Square, and Exponential.
In the Iid policy, the partitioner creates partitions by randomly and uniformly sampling data from the dataset.
In the Linear policy, partitions are created such that the size of each partition is linearly proportional to its ID. The amount of data assigned to each client increases linearly with the partition ID. For example, if the IDs range from 1 to k, the client with ID 1 receives 1 unit of data, client 2 receives 2 units, and so on, until client k, which receives k units.
In the Squared policy, the data assigned to each client is proportional to the square of the partition ID. For instance, if the IDs range from 1 to k, the client with ID 1 receives 1 unit of data, client 2 receives 4 units, and so on, up to client k, which receives k^2 units.
In the Exponential policy, the data allocation is based on the exponential value of the partition ID. For example, if the IDs range from 1 to M, the client with ID 1 receives e^1 units of data, client 2 receives e^2 units, and so on, up to client k, which receives e^k units.

We assessed and pre-computed the necessary noise for the specified security parameters and data partition size to

TABLE 1. Max Accuracy Across Datasets and Partition Policies Using `Non-Private`, RDP, and FSRDP Accountant for $\epsilon = 6$ and $\epsilon = 10$

Dataset	Partition Policy	`Non-Private`	ϵ = 10		ϵ = 6	
			RDP	**FSRDP**	**RDP**	**FSRDP**
QQP	Iid	88%	87%	86%	87%	86%
	Linear	88%	88%	86%	86%	85%
	Square	89%	88%	85%	85%	84%
	Exponential	89%	87%	85%	88%	84%
QNLI	Iid	87%	87%	86%	86%	85%
	Linear	88%	88%	87%	87%	83%
	Square	88%	88%	86%	86%	84%
	Exponential	88%	85%	84%	88%	83%
SST2	Iid	91%	90%	89%	90%	89%
	Linear	90%	90%	89%	89%	87%
	Square	92%	90%	89%	89%	88%
	Exponential	91%	91%	90%	90%	89%

TABLE 2. Data Partition size based on Iid, Linear, Square, and Exponential partition Policies

Dataset	Partition Policy	Partition 1	Partition 2	Partition 3	Partition 4
QQP	Iid	90962	90962	90962	90962
	Linear	36384	72769	109153	145540
	Square	12128	48512	109153	194053
	Exponential	11664	31707	86188	234287
QNLI	Iid	26186	26186	26186	26185
	Linear	10474	20948	31422	41899
	Square	3491	13965	31422	55865
	Exponential	3357	9127	24811	67448
SST2	Iid	16838	16837	16837	16837
	Linear	6734	13469	20204	26942
	Square	2244	8979	20204	35922
	Exponential	2159	5869	15953	43368

be added during the learning process, utilizing two widely-used state-of-the-art differential privacy accountants: Renyi Differential Privacy (`RDP`) [12], [21] and `FSRDP` Accountant [11]. We developed a function in the Flower framework that incorporates noise at the client side during training. The noise is added after each round, scaled by the standard deviation divided by the batch size (550 in our experiments). For security parameters, we set $\epsilon = 10, 6$ and $\delta = 1e-6$ for larger datasets (e.g., QNLI, and QQP, each containing several hundred thousand samples) and $\delta = 1e-5$ for the smaller dataset (e.g., SST-2, with tens of thousands of samples). Clipping norm is set to 3 for all experiments. T

4.2. Experiments Results

Figures 3, 4, and 5 depict the accuracy results for three noise addition methods—`Non-Private`, `FSRDP`, and `RDP`—across various datasets and partitioning policies in 5 rounds of the federated learning process. We examined the noise levels needed for each accountant to achieve a target ϵ and evaluated how different partitioning policies influence the maximum accuracy attained.

4.2.1. Various Partition Policies Impact on Max Accuracy. Table I presents the maximum accuracy achieved across various datasets, partition policies, ϵ values, and accountant methods. For small datasets like SST2, different partitioning policies have minimal impact on accuracy for

TABLE 3. Required Noise per Data Partition to Achieve $\epsilon = 6$ and $\epsilon = 10$ for accountants RDP and FSRDP

Dataset	Partition Policy	$\epsilon = 10$								$\epsilon = 6$							
		Partition 1		Partition 2		Partition 3		Partition 4		Partition 1		Partition 2		Partition 3		Partition 4	
		RDP	FSRDP	RDP	FSRDP	RDP	FSRDP	RDP	FSRDP	RDP	FSRDP	RDP	FSRDP	RDP	FSRDP	RDP	FSRDP
QQP	Iid	0.84	1.66	0.84	1.66	0.84	1.66	0.84	1.66	0.91	1.79	0.91	1.79	0.91	1.79	0.91	1.79
	Linear	0.96	1.88	0.86	1.71	0.82	1.63	0.8	1.59	1.12	2.09	0.95	1.85	0.89	1.75	0.86	1.69
	Square	1.28	2.39	0.91	1.8	0.82	1.63	0.77	1.55	1.65	2.84	1.03	1.98	0.89	1.75	0.83	1.63
	Exponential	1.29	2.42	0.98	1.92	0.84	1.67	0.76	1.52	1.68	2.88	1.16	2.16	0.92	1.81	0.81	1.6
QNLI	Iid	1.02	1.99	1.02	1.99	1.02	1.99	1.02	1.99	1.24	2.26	1.24	2.26	1.24	2.26	1.24	2.26
	Linear	1.34	2.49	1.08	2.09	0.98	1.92	0.93	1.84	1.76	2.99	1.34	2.4	1.17	2.17	1.07	2.03
	Square	2.13	3.77	1.22	2.3	0.98	1.92	0.89	1.76	2.93	4.8	1.56	2.71	1.17	2.17	1	1.93
	Exponential	2.17	3.85	1.41	2.6	1.04	2.01	0.87	1.72	2.99	4.9	1.86	3.15	1.26	2.3	0.96	1.87
SST2	Iid	1.15	2.19	1.15	2.19	1.15	2.19	1.15	2.19	1.45	2.56	1.45	2.56	1.45	2.56	1.45	2.56
	Linear	1.58	2.88	1.23	2.32	1.09	2.1	1.02	1.98	2.13	3.54	1.58	2.75	1.36	2.43	1.23	2.25
	Square	2.7	4.79	1.42	2.61	1.09	2.1	0.96	1.88	3.75	6.27	1.88	3.17	1.36	2.43	1.12	2.1
	Exponential	2.77	4.9	1.68	3.03	1.17	2.22	0.93	1.83	3.84	6.45	2.27	3.74	1.48	2.6	1.06	2.02

both RDP and FSRDP. For large datasets without noise, partitioning policies similarly show little effect on accuracy. When dealing with large datasets, high security ($\epsilon = 6$), and the FSRDP accountant, the Iid policy delivers the best performance. In contrast, for the same security level and large datasets using the RDP accountant, the Exponential policy performs best. For large datasets requiring lower accuracy ($\epsilon = 10$) with both FSRDP and RDP accountants, partitioning policies have a negligible impact on performance. Table II describes data partition size based on various partition policies. Finally, Table III outlines the necessary noise standard deviation for $\epsilon = 10$ and $\epsilon = 6$, which must be added during the training phase of each client at the end of each training round.

4.2.2. Accountant Type Impact on Noise for Target ϵ. In reference to Table III, achieving a higher privacy level (lower ϵ) necessitates the addition of more noise. Furthermore, the FSRDP accountant requires greater noise levels compared to the RDP accountant to attain the same epsilon value under the add-remove adjacency relation. We note, however, that under replace-one adjacency, another commonly used adjacency notion, RDP and FSRDP require nearly identical noise levels to achieve the same ϵ value; see the discussion in Section 4 of [11], including the comparison in Figure 4. More specifically, the noise levels for RDP in Table III would need to be approximately doubled in order to provide the same ϵ guarantee under both adjacency relations, while the noises currently listed for FSRDP in Table III guarantee the stated ϵ for both adjacency relations. Thus, when requiring the privacy guarantees to extend to replace-one adjacency, the benefits of fixed-size subsampling are even more apparent. As the proper choice of adjacency relation is debatable, one might reasonably require guarantees that cover both, in which case the benefits of FSRDP are even more apparent.

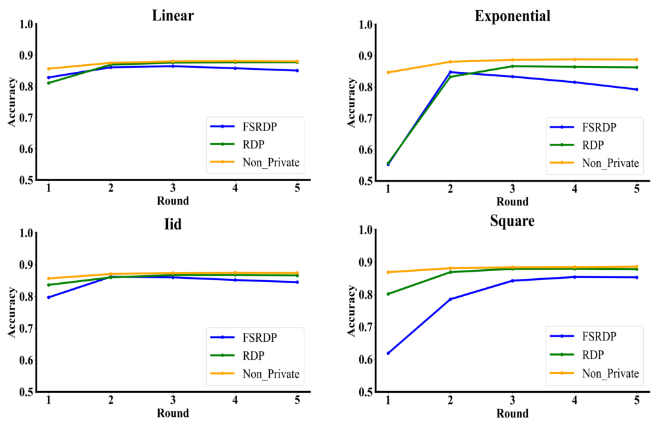

Figure 3. QQP Accuracy Across 5 Rounds with $\epsilon = 10$

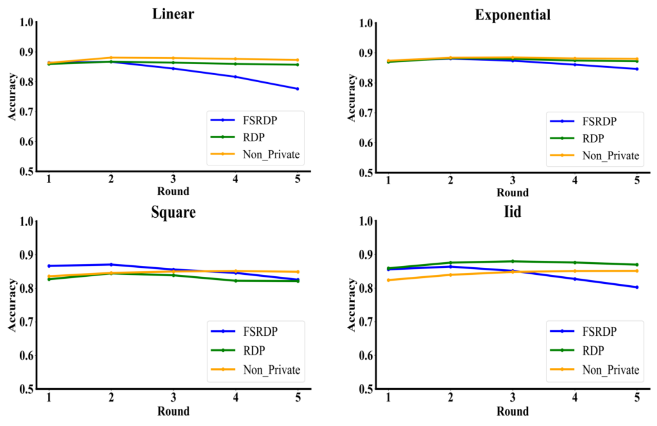

Figure 4. QNLI Accuracy Across 5 Rounds with $\epsilon = 10$

4.3. Discussion

Our research is the first in FL-DP to highlight the critical role of federated learning and differential privacy parameters, as well as their combined effect on model privacy and utility. FLIP integrates a human practitioner who suggests privacy and FL parameters to help strike an optimal balance between privacy and utility in these environments. We focus on the widely adopted task of fine-tuning large language models (LLMs) and illustrate how key parameters such as privacy cost, data distribution, and client selection strategies affect model performance. FLIP is also the first privacy-preserving framework to address the memory constraints of mobile devices in an FL setting by employing a privacy accountant with fixed memory requirements, achieved through a fixed minibatch size. Finally, we offer a detailed comparison between fixed-size minibatch accounting and

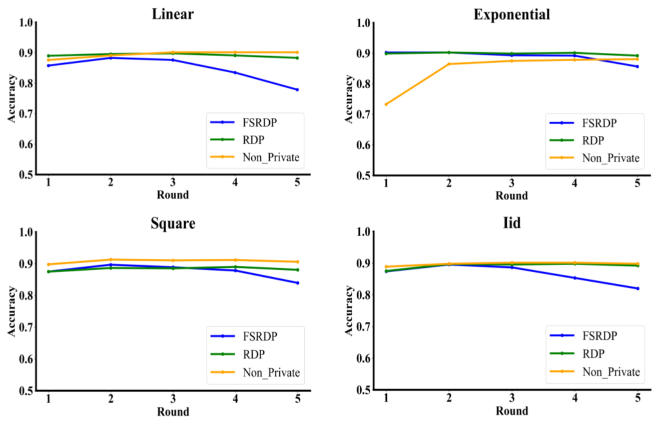

Figure 5. SST2 Accuracy Across 5 Rounds with $\epsilon = 10$

the state-of-the-art RDP approach, highlighting the trade-offs in privacy and utility. The FSRDP accountant provides advantages such as achieving an acceptable maximum accuracy and uniform memory usage. However, our experiments indicate that while the model performs well in the initial rounds using FSRDP accountant, its accuracy may decline in the later stages. This drop is caused by the cumulative effect of noise introduced by the FSRDP accountant and imbalances in client contributions. To overcome these challenges, we propose strategies like dynamically adjusting noise levels during training to better balance privacy and accuracy, as well as ensuring balanced client sampling to improve stability in the later rounds.
In the proposed framework, the privacy practitioner may be either a human expert or an AI agent, each with distinct security implications. A human practitioner offers adaptability but is susceptible to errors, biases, and inconsistencies when setting privacy parameters. Conversely, an AI agent ensures consistency but may be prone to algorithmic bias, adversarial exploitation, or limited flexibility. To mitigate bias and ensure reliability, we propose a hybrid approach where privacy parameters are selected based on standardized security guidelines, with AI-assisted recommendations and human oversight when needed.

5. Conclusion

Our study highlights the critical role of parameter selection and the interpretation of privacy costs in different application settings. By examining the interplay between federated learning and differential privacy, we demonstrate how thoughtful parameter tuning can significantly impact both model utility and privacy guarantees. A promising direction for future work is to expand this study with more comprehensive experiments, considering diverse data types such as images and text to further generalize our findings. Additionally, further evaluation of FSRDP with the replace-one adjacency relation could provide deeper insights into its effect on privacy guarantees and model utility, offering valuable guidance for privacy-preserving federated learning deployments.

References

[1] Q. Liu, X. Wu, X. Zhao, Y. Zhu, D. Xu, F. Tian, and Y. Zheng, "When moe meets llms: Parameter efficient fine-tuning for multi-task medical applications," in *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1104–1114, 2024.

[2] B. Balle, G. Cherubin, and J. Hayes, "Reconstructing training data with informed adversaries," in *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1138–1156, IEEE, 2022.

[3] J. Hayes, B. Balle, and S. Mahloujifar, "Bounding training data reconstruction in dp-sgd," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[4] M. Nasr, R. Shokri, and A. Houmansadr, "Machine learning with membership privacy using adversarial regularization," in *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, pp. 634–646, 2018.

[5] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," in *2018 IEEE 31st computer security foundations symposium (CSF)*, pp. 268–282, IEEE, 2018.

[6] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song, "The secret sharer: Evaluating and testing unintended memorization in neural networks," in *28th USENIX security symposium (USENIX security 19)*, pp. 267–284, 2019.

[7] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, *et al.*, "Extracting training data from large language models," in *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

[8] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pp. 265–284, Springer, 2006.

[9] A. Groce, J. Katz, and A. Yerukhimovich, "Limits of computational differential privacy in the client/server setting," in *Theory of Cryptography: 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings 8*, pp. 417–431, Springer, 2011.

[10] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays, "Applied federated learning: Improving google keyboard query suggestions," *arXiv preprint arXiv:1812.02903*, 2018.

[11] J. Birrell, M. Ebrahimi, R. Behnia, and J. Pacheco, "Differentially private stochastic gradient descent with fixed-size minibatches: Tighter RDP guarantees with or without replacement," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[12] I. Mironov, "Rényi differential privacy," in *2017 IEEE 30th computer security foundations symposium (CSF)*, pp. 263–275, IEEE, 2017.

[13] L. Fu, H. Zhang, G. Gao, M. Zhang, and X. Liu, "Client selection in federated learning: Principles, challenges, and opportunities," *IEEE Internet of Things Journal*, 2023.

[14] H. Xiao, J. Zhao, Q. Pei, J. Feng, L. Liu, and W. Shi, "Vehicle selection and resource optimization for federated learning in vehicular edge computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11073–11087, 2021.

[15] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," in *International Conference on Learning Representations*, 2019.

[16] R. Behnia, M. R. Ebrahimi, J. Pacheco, and B. Padmanabhan, "Ew-tune: A framework for privately fine-tuning large language models with differential privacy," in *2022 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 560–566, IEEE, 2022.

[17] D. Yu, S. Naik, A. Backurs, S. Gopi, H. A. Inan, G. Kamath, J. Kulkarni, Y. T. Lee, A. Manoel, L. Wutschitz, *et al.*, "Differentially private fine-tuning of language models," *arXiv preprint arXiv:2110.06500*, 2021.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

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

[20] C. Dwork, A. Roth, *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[21] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

[22] C. Dwork and G. N. Rothblum, "Concentrated differential privacy," *arXiv preprint arXiv:1603.01887*, 2016.

[23] opacus, 2025. https://opacus.ai/docs/introduction, Last accessed on 2025-1-8.

[24] B. Balle, G. Barthe, and M. Gaboardi, "Privacy amplification by subsampling: Tight analyses via couplings and divergences," *Advances in neural information processing systems*, vol. 31, 2018.

[25] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan, "Subsampled rényi differential privacy and analytical moments accountant," in *The 22nd international conference on artificial intelligence and statistics*, pp. 1226–1235, PMLR, 2019.

[26] A. Triastcyn and B. Faltings, "Bayesian differential privacy for machine learning," in *Proceedings of the 37th International Conference on Machine Learning*, vol. 119 of *Proceedings of Machine Learning Research*, pp. 9583–9592, PMLR, 2020.

[27] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Superglue: A multi-task benchmark and analysis platform for natural language understanding," *Advances in Neural Information Processing Systems*, vol. 32, pp. 3261–3275, 2019.

[28] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.