LanFL: Differentially Private Federated Learning with Large Language Models using Synthetic Samples

1st Huiyu Wu
Industrial Engineering and Management Sciences
Northwestern University
Evanston, USA
huiyuwu2025@u.northwestern.edu

2nd Diego Klabjan
Industrial Engineering and Management Sciences
Northwestern University
Evanston, USA
d-klabjan@northwestern.edu

***Abstract*—Federated Learning (FL) is a collaborative, privacy-preserving machine learning framework that enables multiple participants to train a single global model. However, the recent advent of powerful Large Language Models (LLMs) with tens to hundreds of billions of parameters makes the naive application of traditional FL methods to LLMs impractical due to high computational and communication costs. Furthermore, end users of LLMs often lack access to full architectures and weights of the models, making it impossible for participants to fine-tune these models directly. This paper introduces a novel FL scheme for LLMs, named LanFL, which is purely prompt-based and treats the underlying LLMs as black boxes. We have developed a differentially private synthetic sample generation mechanism to facilitate knowledge sharing among participants, along with a prompt optimization scheme that enables learning from synthetic samples. Our extensive experiments demonstrate that LanFL successfully facilitates learning among participants while preserving the privacy of local datasets across various tasks.**

I. Introduction

Federated Learning (FL) is a privacy preserving machine learning scheme that allows many participants to train a single global model without sharing their local training samples [1]. FL relies on participants sharing their local gradient updates or models, and a central server or some decentralized framework to aggregate the updates [2], [3]. Since only gradient updates or models are shared, each client's local data set privacy is preserved. This is especially crucial nowadays as there are increasing regulations on data sharing such as EU's General Data Protection Regulation [4] and the California Consumer Privacy Act [5]. It has been applied in many fields including finance, edge computing, and healthcare. However, despite its popularity, FL still faces challenges in privacy threats, heterogeneity, and communication overhead, and new paradigms for FL are needed [6], [7].

Large Language Models (LLMs) are sophisticated deep neural networks based on transformer architectures, designed to comprehend, generate, and manipulate natural language [8], demonstrating performances that approach, and sometimes exceed, human capabilities, LLMs excel in tasks such as logical reasoning, translation, and complex examinations [9]. Their exceptional abilities render them applicable across various domains similar to those in Federated Learning (FL), including customer service, finance, law, and education [10]. Despite their widespread adoption and impressive functionalities, LLMs are associated with significant safety concerns, including risks of fraud, impersonation, and misinformation dissemination [11]. Consequently, developers restrict access to the models' weights, architecture, and parameter details for prominent LLMs like Chat-GPT 4, Claude 3, and Gemini [9], [12], [13]. It is also worth noting that with these models containing tens or hundreds of billions of parameters, transmitting complete models several times or conducting extensive fine-tuning is often impractical. Instead, prompt engineering emerges as a viable alternative to leverage these models effectively [14]. Prompts serve as inputs to LLMs; it has been demonstrated that the outputs from these models are highly sensitive to the nature of the prompts provided. Well-constructed prompts enhance the relevance, logical coherence, and accuracy of the outputs.

At a first glance, the application of FL with LLMs appears impractical. Federated Learning primarily relies on local gradient updates or local models; however, in the context of LLMs, users often lack access to the model weights. Furthermore, even if FL clients could access the weights of LLMs, computing gradients over numerous iterations would not be computationally efficient. Additionally, the significant communication overhead associated with transmitting potentially hundreds of billions of parameters renders the process inefficient. Nevertheless, there are some methods available to enable FL for LLMs, which typically involve introducing trainable parameters into the LLM architecture while maintaining the pre-trained components unchanged [15], [16]. It is important to recognize that although these existing LLM FL methods address the issues of large communication and training costs, they still necessitate knowledge of the specific architecture and parameters of the underlying LLM. This requirement restricts the practical applications of such approaches. For instance, a consortium of medium-sized hospitals is likely not to prefer to invest in open source LLMs and going down a rabbit hole to buy costly GPUs and hire experts. Instead, they may opt to utilize the best pre-trained LLMs available from large technology companies, where the architectural details and parameters remain confidential.

There are also several notable distinctions between FL with LLMs and traditional FL approaches. Traditionally, FL typically involves training a model from scratch, whereas FL with LLMs generally starts with pre-trained LLMs, focusing on fine-tuning these models for specific tasks rather than training new ones from the ground up. Another key difference lies in model ownership and accessibility: in conventional FL, clients own the models and have full access to all model aspects. However, this is not always the case with FL involving LLMs. For instance, banks participating in an FL scheme using LLMs might opt to utilize powerful pre-trained models from third-party providers without the intention or need to train their own models, thereby lacking access to the model's architecture and weights. Additionally, unlike traditional FL where often clients use the same models, in the LLM context, different participants may choose to use various third-party models based on factors such as cost or other specific requirements.

Inspired by the capability of LLMs to learn from contextual examples in prompts and their ability to generate synthetic samples [17], [18], we propose a new LLM FL scheme, LanFL, which aims to bridge the gap between FL and LLMs. LanFL is designed to enable participants to engage in FL without requiring access to the underlying architecture and weights of the LLM. On a high level, in one LanFL step, participants first create synthetic samples using their local LLMs. Then the participants generate prompts using the synthetic samples and their own knowledge. The next step is participants sharing the prompts to other participants. Since LLMs can learn from prompts, combining local data and the prompts with synthetic data received, participants can improve model performance without accessing other participants' local data sets. We also show that LanFL is differentially private. In the experiments LanFL performs well when clients have heterogeneous data sets.

Our contributions are as follows. First, to the best of our knowledge, we are the first work enabling FL for LLMs using only prompts, without accessing and modifying underlying weights and architectures. This advancement is crucial as it circumvents the need for participants to access the weights directly, thereby also allowing for the use of different underlying LLMs by various participants. Second, we introduce an innovative method for generating and selecting synthetic samples that ensures their effectiveness while also confirming that they are sufficiently distinct from the original training samples. Additionally, our mechanism for synthetic sample generation is designed to be differentially private. Third, our experimental results indicate that LanFL performs robustly across a variety of datasets, and different levels of data heterogeneity.

This paper is organized as follows. In Section 2 we examine related works, while in Section 3 we explain the details of LanFL operations and study the differential privacy properties of LanFL. In Section 4 we explore our comprehensive experimental results, and finally in Section 5 we present the conclusions.

II. Related Works

There are a few works focusing on FL of LLMs, and all requires full knowledge of the architecture of the underlying model. Che et al propose FedPepTAO [15] which utilizes techniques in prefix-tuning and adaptive optimization. On a high level, participants perform prefix-tuning by adding trainable parameters to each layer of the LLM [19]. In each round a set of participants only update prompt parameters of specific layers. FedPepTAO is parameter efficient since each round only a few layers of prefixes are updated, and it does not modify the weights of the underlying LLM. However, it still requires full knowledge of the underlying LLM architecture and does not scale well as future LLMs increase in the number of layers and parameters. Work by Hou et al, [20], employs differentially private synthetic samples to fine-tune LLMs. Their approach generates synthetic samples through a multi-round process using only textual data, and the fine-tuning step requires knowledge of the underlying model architecture. In contrast, our approach generates synthetic samples in a single round and accommodates numerical features. Additionally, we leverage in-context learning techniques, which eliminate the need for knowledge of the underlying model architecture.

Similarly, Kim et al propose using adapter mechanisms to LLMs for FL [16]. Specifically, they add adapter layers such as LoRA [21] into the LLM architectures, and only fine-tune the parameters of added layers while keeping the pre-trained LLM parameters frozen. Fan et al implement this framework named FATE-LLM that utilizes trainable adapters [22]. Again, this approach requires knowing the exact architecture of the underlying LLM and all experiments are run with not the larges LLMs having less than 10 billion parameters.

Another idea for FL LLM is proposed by Su et al [23]. The TITANIC framework they propose first splits the LLM into multiple parts by layers, and each participant receive a part of the LLM. FL is facilitated by having each participant perform a forward pass, sending intermediate outputs to the next participant who possesses the subsequent segment, followed by a backward loss propagation step. While TITANIC demonstrates superior performance compared to previous frameworks, it is not without challenges; specifically, it encounters issues related to privacy and significant communication overhead. In contrast, our LanFL approach is based on the exchange of prompts not requiring weights, which circumvents these issues, offering a more efficient and secure method of collaborative learning.

Prompting techniques are also closely related to our work. They involve using specific inputs to elicit high-quality and accurate responses from LLMs. The effectiveness of these outputs is heavily dependent on the construction of well-designed prompts [14]. Few-shot prompting [17] supplies LLMs with example inputs that guide the model to produce outputs that are not only formatted like the examples but also contextually relevant to the topics addressed by these examples. Chain-of-thought (CoT) prompting, another related technique, encourages LLMs to articulate a sequence of logical reasoning steps before arriving at a final answer. This method has been shown to enhance the logical coherence of the outputs and is particularly effective in logic reasoning tasks [24]. Building on the CoT framework, approaches such as self-consistency [25] and tree-of-thought [26] recognize that different reasoning pathways can lead to correct answers. These methods allow LLMs to generate multiple reasoned responses before selecting the most appropriate final output, thus enhancing the robustness and reliability of the model's decision-making process.

The generation of synthetic datasets using LLMs represents a key advancement upon which our LanFL relies. Frameworks such as ZeroGen, ProGen, and SuperGen facilitate production of synthetic datasets by conditioning on labels and employing task-specific models to iteratively select and filter synthetic samples [27], [28], [29]. These processes operate independently of human-annotated samples and treat pre-trained LLMs as black boxes. Nonetheless, these frameworks primarily focus on textual data, whereas LanFL also accommodates numerical data. Furthermore, in FL operations, participants typically possess well-annotated samples, a feature that LanFL leverages. ClinGen represents another LLM-based framework for generating synthetic samples, specifically targeting medical tasks; however, it does not generalize to numerical data [30]. Borisov et al introduce GReaT, a framework designed for generating tabular data [18]. We build upon this framework by incorporating advanced prompting techniques to enhance its efficacy with numerical tabular data.

III. LanFL

A. LanFL Operations

As discussed in the introduction, FL with LLMs are challenging because of the large model sizes and black-box nature of the models. Our proposed LanFL method addresses these challenges and is the first to employ a prompt-based framework for FL with LLMs. It enables clients to engage in FL while treating the underlying LLMs as black boxes. As shown in Figure 1, at a high level, LanFL consists of three steps: generate synthetic samples, share knowledge, and learn using prompts. LanFL steps start with each participant creating synthetic samples and generate logic solutions to the synthetic samples based on its local training samples. The next step is to share the synthetic samples with labels and reasoning among all other participants. Finally, participants add all synthetic samples to the local training set, and learn from the samples by optimizing for the best prompt and use the optimized prompt for downstream tasks.

In detail, the initial step involves participants generating synthetic samples using LLMs with few-shot prompting. Specifically, participants randomly select a few local samples that include chain-of-thought reasoning and use these as examples to prompt the LLM to generate new samples. This process is repeated to create a synthetic dataset. The chain-of-thought reasoning for the local samples can be either manually obtained or generated by prompting the LLM as

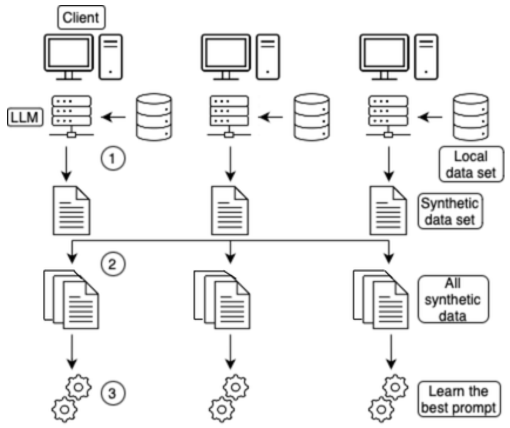


Fig. 1: LanFL Operations. Step 1: Clients generate synthetic data sets by prompting LLMs using local data sets. Step 2: Clients share the synthetic data sets among themselves. Step 3: Clients learn the best prompt utilizing the synthetic data sets received.

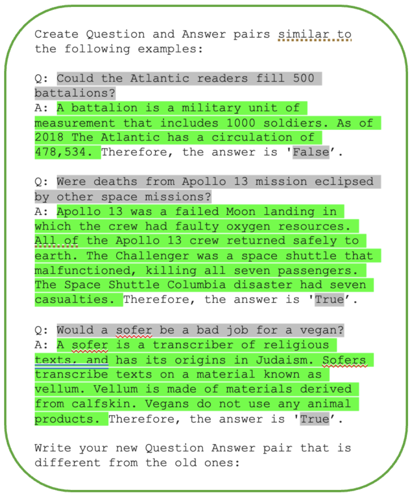
a separate pre-processing step. Consequently, the resulting synthetic samples contain not only features and labels but also the chain-of-thought reasoning, making them immediately usable as examples in prompts. The second step involves the sharing of knowledge, wherein participants exchange the synthetic data sets they have generated. Since these synthetic data sets encompass chain-of-thought reasoning derived from the participants' local samples, the act of sharing these data sets inherently facilitates the dissemination of knowledge. Finally, upon receiving synthetic datasets from other participants, the clients aim to learn from these samples by designing few-shot prompts that optimize in-context learning. The learning is relied on LLMs' in-context learning capabilities [31], and by utilizing the synthetic samples containing knowledge from other participants, privacy is achieved.

B. Synthetic Samples

A key element in the LanFL framework is the generation of synthetic samples. Inspired by chain-of-thought and few-shot prompting [17], [24], we propose the following two-step mechanism to generate synthetic samples. For a participant with local data set $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, and an LLM L with temperature set to 0, the first step is to generate the chain of thought reasoning r_i for each of the sample (x_i, y_i). Specifically

$$r_i = L(M_r(x_i, y_i)).$$

Here M_r can be any function that converts (x_i, y_i) into prompt so that the LLM outputs a chain-of-thought reasoning r_i for specific sample (x_i, y_i). Note that x_i, y_i, r_i and the output of function M_r are all sequences of tokens. The first step can be skipped if the participant has readily available reasoning steps in the local data set or it decides to manually provide chain-of-thought reasoning for all the local samples. During the second step, the participant randomly selects k samples, $D_k =$



```
Create Question and Answer pairs similar to
the following examples:

Q: Could the Atlantic readers fill 500
battalions?
A: A battalion is a military unit of
measurement that includes 1000 soldiers. As of
2018 The Atlantic has a circulation of
478,534. Therefore, the answer is 'False'.

Q: Were deaths from Apollo 13 mission eclipsed
by other space missions?
A: Apollo 13 was a failed Moon landing in
which the crew had faulty oxygen resources.
All of the Apollo 13 crew returned safely to
earth. The Challenger was a space shuttle that
malfunctioned, killing all seven passengers.
The Space Shuttle Columbia disaster had seven
casualties. Therefore, the answer is 'True'.

Q: Would a sofer be a bad job for a vegan?
A: A sofer is a transcriber of religious
texts, and has its origins in Judaism. Sofers
transcribe texts on a material known as
vellum. Vellum is made of materials derived
from calfskin. Vegans do not use any animal
products. Therefore, the answer is 'True'.

Write your new Question Answer pair that is
different from the old ones:
```

Fig. 2: Example prompt used to generate synthetic sample (Output of M_{syn})

$\{(x_{s_1}, r_{s_1}, y_{s_1}), (x_{s_2}, r_{s_2}, y_{s_2}), \ldots, (x_{s_k}, r_{s_k}, y_{s_k})\}$, and uses these samples as examples in the prompt to generate a synthetic sample

$$syn_{D_k} = L(M_{syn}(D_k)).$$

We can then extract $(x_{D_k}, r_{D_k}, y_{D_k})$ from syn_{D_k}. Here M_{syn} is a prompt template that converts the samples into a few-shot prompt that instructs the LLM to generate a synthetic sample. To create the synthetic data set, the participant simply repeats the previous step to reach the desired number of synthetic samples.

To be more concrete, M_r can simply list the tokens x_i, and asks the LLM why y_i is the answer. For example, the output of M_r can be: '*Weng earns \$12 an hour for babysitting. Yesterday, she just did 50 minutes of babysitting. How much did she earn? Think step by step and answer why she earned \$10.*' For questions that require domain knowledge to generate logic reasoning, the participant can first manually provide logical reasoning for a few selected samples, add background knowledge, and use them as examples in prompt and instruct the LLM to provide reasoning for other samples. For synthetic sample generation, Figure 2 shows an example output of M_{syn} using the CommonsenseQA data set [32]. The sentences marked gray are x_i and y_i, sentences marked green are chain-of-thought reasoning, and the remaining sentences are meta prompt and formatting added by M_{syn}. It is important to note that by incorporating logical reasoning and facts into the examples within the prompt, the resulting synthetic samples encompass not only the questions and their answers but also the associated logical reasoning steps. This approach is more efficient, as it allows for the simultaneous generation of synthetic samples and their logical reasoning, rather than producing the synthetic samples first and subsequently generating the reasoning as a separate step.

For numerical features, we need one pre-processing step to convert the features into tokens that LLMs can understand better. Instead of the simple textual encoder in GReaT where we add '*is*' between feature name and the feature value [18], we require the participant to connect the features into a cohesive paragraph. It may require domain knowledge for specific tasks. As an example, for features concerning a credit card holder's information, instead of using '*Gender is male, education is high school, age is 30, payment is 400, usage is 600,*' we use '*A high-school educated 30 year-old male has used \$600 of his credit and has made a \$400 payment.*' as x_i.

Another consideration for a participant is that of all the synthetic samples created, the participant is recommended to discard samples that are similar to those in its local data set. For example, for each of the synthetic sample x_{syn}, the participant should compute n BLEU scores $b_1, b_2, \ldots b_n$ for each of the local sample $x_1, x_2, \ldots x_n$ [33]. Then the participant can set a threshold t depending on the application and discard any synthetic samples that have $b_{max} > t$ where $b_{max} = max(b_1, b_2, \ldots b_n)$. For numerical features, the participant can utilize the $L2$ distance between the feature vectors to determine whether or not to discard any of the synthetic samples.

C. Differential Privacy

Differential Privacy is a formal definition of privacy and it characterizes if an output mechanism leaks information if the underlying data set differs by one sample [34]–[36]. To establish our synthetic sample generation method is differential privacy, we first formalize the synthetic sample generation mechanism.

Mechanism 1. *We define the random synthetic sample generation mechanism f as follows. Let us have two numbers $N, k \in \mathbb{N}$, and $N \geq k + 1$. The mechanism f generates synthetic sample $f(D) = x_{syn}$ given a data set $D \in dom(f)$ by following the following two steps. First, we select $A \subseteq D$ such that $|A| = k$ by uniform randomly selecting elements one by one from D without replacement. Then, we apply function L to A to generate $L(A) = x_{syn}$, where $L(\cdot)$ is any deterministic function that outputs a sequence of tokens. We also restrict the $dom(f)$ to satisfy $D \in dom(f)$, $|D| \geq N$.*

A technicality is that here the function L applies to the set A, thus we are implicitly assuming the order of the samples in A does not matter. The results are similar if ordering of the elements in A matters (epsilon and delta take different values, but the proof is similar). Note that L is deterministic and the only randomness of f comes from the sampling of the subset A. The restriction on the domain of f means we can only apply the mechanism if we have a data set that has more than N samples. In the FL with LLM, we can pick N to be the smallest number of samples of the data sets among the FL participants.

Note that the mechanism f is the same as the procedure to generate one synthetic sample in LanFL. Specifically, in LanFL the set A is the data set containing the k random local examples, and becuase we set the temperature of the LLM to 0, M_{syn} and the LLM is deterministic function L. Now we introduce the differential privacy result.

Theorem 1. *Mechanism 1 is (δ, ϵ) differentially private for $\delta = \frac{k}{N+1}$ and $\epsilon = 0$. Specifically, for any two data sets D_1, D_2 in the domain of f that differ by only one element by addition or deletion, and any subset S in the range of f we have*

$$P(f(D_1) \in S) \leq \frac{k}{N+1} + e^0 P(f(D_2) \in S).$$

Theorem 1 shows that our synthetic sample generation mechanism for a single synthetic sample is (δ, ϵ) differentially private by definition. Generally we have $N \gg k$ meaning our sample size is much greater than the number of samples in the prompt used to generate synthetic samples, therefore δ is small indicating a strong privacy guarantee. The proof of Theorem 1 is in the Appendix. In practice, participants generate several synthetic samples, and we have the following Corollary showing generating w synthetic samples is also differentially private.

Corollary 1. *We call f_w the process of performing Mechanism 1 for a total of w times generating w synthetic samples. Then Mechanism f_w is (δ, ϵ) differentially private for $\delta = \frac{kw}{N+1}$ and $\epsilon = 0$. Specifically, for any two data sets D_1, D_2 in the domain of f_w that differ by only one element by addition or deletion, and any subset S in the range of f_w we have*

$$P(f(D_1) \in S) \leq \frac{kw}{N+1} + e^0 P(f(D_2) \in S).$$

This Corollary follows from the Sequential Theorem in [35]. It is important to note that, in practical applications, we generate only w synthetic samples, which usually represents a small fraction of the overall size of the dataset N. Therefore, $\frac{kw}{N+1}$ is small, indicating a strong differential privacy guarantee.

D. Prompt Optimization

The last step in the LanFL operations is participants optimizing for a best prompt utilizing all the synthetic samples received from other participants in addition to their local data sets. It is known that LLMs are sensitive to input prompts, for example, the number and ordering of the examples in the prompt matters [31], [37]–[39]. Therefore, by optimizing the number of samples and portions of synthetic samples used as examples in a prompt, participants should learn from other participants' samples.

It is impractical to iterate over all the combinations of samples and synthetic samples in the prompt. Therefore, we propose Algorithm 1, a greedy optimization method. First, the participant sets aside a portion of the local data set as a test set, and let the remaining samples be a training set. Then the participant selects a different number of samples from the training set as examples in the prompt and evaluates the results on the test set, and finds the optimal number of samples to include in the prompt (lines 2-10). After collecting all the synthetic samples and using the best number of samples n_1 in the prompt, the participant finds the best number of synthetic

Algorithm 1 LanFL Operation

```
1:  Initialize training set $X_{train}$ and test set $X_{test}$
2:  for $n = 1, 2, 3, \ldots$ do
3:      Set cost $c_n = 0$
4:      for Each test sample $x_t \in X_{test}$ do
5:          Randomly select $x_{i_1}, x_{i_2} \ldots x_{i_n} \in X_{train}$
6:          Compute cost $c$ using some metric between
            $Y_{train}$ and $Y_{predicted}$
7:          Update cost $c_n$ += $c$
8:      end for
9:  end for
10: Record optimal cost $c_{train} = min\{c_i | 1 \leq i \leq n\}$
11: Optimal number of samples in prompt $n_1 =$
    $argmin_i\{c_i | 1 \leq i \leq n\}$
12: Collect synthetic samples $X_{syn}$
13: for $n = 1, 2, 3, \ldots n_1$ do
14:     Set cost $c_n = 0$
15:     for Each test sample $x_t \in X_{test}$ do
16:         Randomly select $x_{i_1}, x_{i_2} \ldots x_{i_n} \in X_{syn}$
17:         Randomly select $x_{i_n}, x_{i_{n+1}} \ldots x_{i_{n_1}} \in X_{train}$
18:         Compute cost $c$ using some metric between
            $Y_{train}$ and $Y_{predicted}$
19:         Update cost $c_n$ += $c$
20:     end for
21: end for
22: Record optimal cost $c_{syn} = min\{c_i | 1 \leq i \leq n\}$
23: Optimal synthetic samples in prompt $n_2 =$
    $argmin_i\{c_i | 1 \leq i \leq n\}$
```

samples n_2 to use in prompts by testing on the test set again (lines 12-21). Algorithm 1 summarizes the operations to obtain the number of total samples and the number of synthetic samples in the prompt. The cost can be interpreted as the number of incorrect answers or the mean squared error for continuously valued tasks.

Another benefit of LanFL operation is by comparing the cost of using the optimal number of training samples c_{train} and the cost of using the optimal number of synthetic samples with training samples c_{syn}, we can examine the effectiveness of LanFL operations. If $c_{syn} - c_{train} > 0$, then we know it is beneficial to receive the synthetic samples from other participants and LanFL achieves the goal of learning. Finally, we want to point out that loops in lines 4 and 15 should be repeated a few times to reduce the randomness in results.

IV. Experiments

A. LanFL Experimental Setup

The goal of the experiments is to show LanFL steps enable learning, therefore in the following experiments we aim to show after receiving synthetic samples from other clients, that a participant can obtain better results by adjusting prompts compared to using only its local data set. For each of the experiments, we first set aside 10% of the samples as the validation set and 10% of samples as the test set, then partition the remaining data set into 10 subsets representing the participants. Each of the participants uses the local samples to generate 100 synthetic samples with $k = 3$. We aggregate the synthetic samples together and perform Algorithm 1, LanFL operation, on the validation set to select the best n_1 and n_2, and finally, we test the performance on the test set using the optimal mixture of synthetic samples. We repeat 3 times for each of the experiments due to the stochastic nature of Algorithm 1. The reported numbers are expected values with standard deviations in select charts and tables. Note that we experimented with different number of participants and prompt samples k, and the findings are similar. We utilize Gemini-1.5-Flash [13], Llama3.2-3B [40], and Mixtral-8×7B [41] in our experiments. These models exhibit strong performance and efficiency, encompass both transformer-based and mixture-of-experts architectures, and are readily accessible for reproducibility and further research. The Gemini experiments are run on Google Colab, and other models are run with Ollama locally with Nvidia GeForece RTX 3080Ti GPU and Intel i9-12900k CPU.

We test LanFL on the StrategyQA, UCI Credit Card Default dataset [42], and the eICU critical care database [43]. StrategyQA, a challenging textual reasoning dataset, assesses LanFL's ability to handle complex language and strategic reasoning. The UCI Credit Card Default dataset, with its numerical features, evaluates LanFL's performance in working with numerical features. The eICU database, a real-world dataset, tests LanFL's ability to handle complex, real-world data, particularly in the FL amenable healthcare setting. By combining these diverse datasets, we aim to demonstrate LanFL's versatility and effectiveness across a wide range of tasks and data types. Additionally, the eICU experiment highlights the potential of LanFL to contribute to real-world applications.

B. LanFL Results

The experimental results are presented in Table I. We report the F1 scores and the standard deviations in parentheses of the models on the test set using LanFL, as well as the F1 scores achieved through in-context learning using only the original samples. The F1 score is selected as our evaluation metric due to its ability to address class imbalance and its alignment with our primary objective: accurately identifying credit card defaults and patients in critical conditions. We use the F1 score for StrategyQA as well for consistency. We also include a comparison with a random guess benchmark to validate that the LLMs effectively leverage the information in the samples and their inherent capabilities. The results indicate that all tested LLMs exhibit improved performance across all tasks when synthetic samples are incorporated into the prompts through LanFL optimization. This demonstrates that synthetic samples contain valuable information from other clients that can aid in answering test samples. Furthermore, LanFL enables information sharing among participants while preserving privacy. Notably, the LanFL-enhanced results significantly outperform the random guess benchmark.

TABLE I: Experimental Results (F1 score %)

	Gemini	Mixtral	Llama	Random Guess
StrategyQA-LanFL	55.27	**63.67**	59.78	46.72
	(6.76)	(1.05)	(1.78)	–
StrategyQA	53.29	**63.03**	57.02	46.72
	(10.23)	(2.26)	(1.74)	–
UCI-LanFL	32.07	**38.78**	37.21	22.12
	(2.29)	(1.33)	(0.71)	–
UCI	13.69	14.52	**30.31**	22.12
	(3.78)	(2.08)	(1.87)	–
eICU-LanFL	16.74	17.24	**24.85**	8.33
	(0.52)	(3.88)	(1.84)	–
eICU	14.54	15.54	**17.72**	8.33
	(1.05)	(2.79)	(1.56)	–

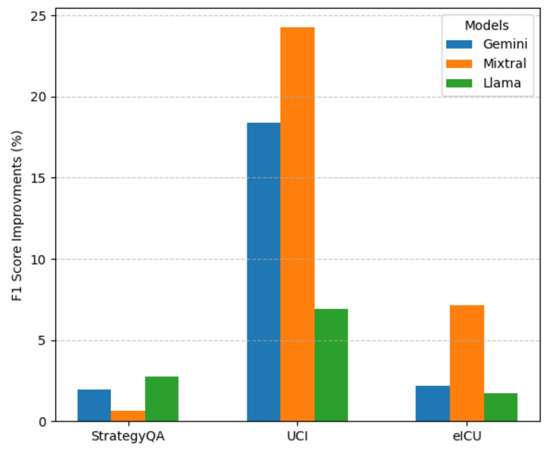


Fig. 3: LanFL improvements

Another key finding is that no specific model consistently performs the best across all tasks, although LanFL appears to be particularly beneficial for certain tasks. Figure 3 illustrates the F1 score improvements achieved using LanFL compared to in-context learning with original samples. When combined with the full results, we observe that no single model outperforms the others across all tasks. However, an intriguing observation from the same plot is the variation in LanFL's performance across tasks. Specifically, LanFL performs best on the UCI credit card default dataset and worst on StrategyQA. We attribute this to the nature of StrategyQA, which relies more on knowledge of specific facts rather than the shared information or logic contained in the samples from other participants. As a result, synthetic samples provide limited benefits in this context. Conversely, credit card defaults can occur under diverse circumstances, and the additional information and reasoning derived from other samples significantly benefit the LLMs. Lastly, we believe there is potential for greater LanFL benefits on the eICU dataset by utilizing LLMs with more extensive medical knowledge.

C. Effect of Different LLMs at Client Level

In real-world settings, it is unlikely that all clients use the same LLM for their tasks. Different clients often select LLMs of varying sizes and capabilities, tailored to their specific

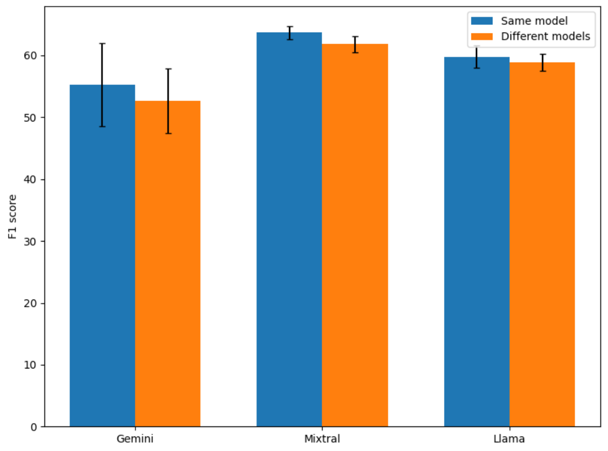


Fig. 4: LanFL with Different Models

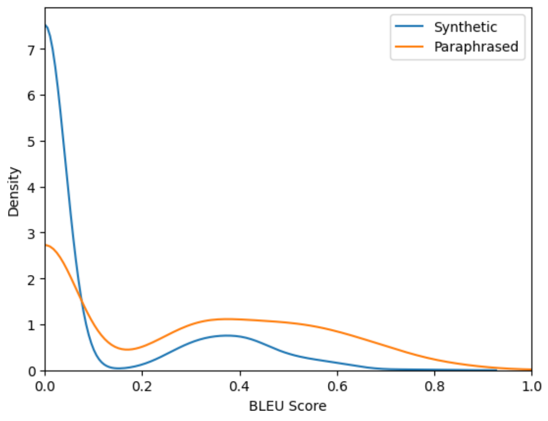


Fig. 5: BLEU scores distributions for synthetic samples and paraphrased training samples

business needs. In this section, we explore the impact of this more realistic scenario—where different clients use different LLMs—on our LanFL operation. Using a setup similar to Section IV-A, we partition the training dataset into 10 subsets representing the participants. However, in this case, each participant is randomly assigned an LLM among Gemini, Mixtral, or Llama. Each client generates 100 synthetic samples, which are aggregated, and all clients subsequently perform LanFL optimization. Clients with the same LLM are grouped together, and we report the average final F1 score on the test set per group using the StrategyQA dataset.

Figure 4 summarizes the F1 scores from experiments where clients either use the same LLM or different LLMs. The first key finding is that the relative performance ranking of the models remains consistent regardless of whether clients use the same or different LLMs. This suggests that the majority of learning in LanFL occurs during the optimization process, which is largely influenced by the LLMs used by the clients. Consequently, the quality of the synthetic samples plays a less significant role. This has a practical benefit for employing the LanFL algorithm, as those with more capable LLMs can maintain their edge over clients with less capable LLMs. Additionally, clients need not be overly concerned about providing higher-quality synthetic samples to clients with less powerful LLMs.

Another observation is that performance tends to be slightly lower when clients use mixed LLMs compared to when all clients use the same LLM. We attribute this to the fact that LLMs are better at learning from synthetic samples generated by themselves than from those generated by other LLMs, which may differ in style and characteristics.

D. Synthetic Samples Evaluation

We use another experiment to show the feasibility of using synthetic samples for our purpose. We evaluate our synthetic samples using two criteria. First, how different are they from the training samples, and second, how good are they in downstream tasks. In this section, since the synthetic samples are purely text-based, we use PaLM 2 as the underlying LLM, given that it is a text-only model and serves as the foundation for Gemini. We use StrategyQA as our dataset [44], [45] and generate synthetic samples with $k = 3$.

Although we have already showed analytically differential privacy of our synthetic sample generation mechanism, the property only states that probabilistically one cannot infer whether one specific sample is in the training data set based on a synthetic sample. However, there is no guarantee that synthetic samples are different from the training samples. To show our mechanism produces sufficiently different samples, we compute b_{max} for each synthetic sample in Section III-B. Additionally, we also paraphrase the training samples used to generate synthetic samples using the same LLM and compute b_{max} for each of the paraphrased sample as a benchmark. Figure 5 shows the distribution of b_{max} of synthetic and paraphrased samples. We find not only the synthetic samples are different from the original samples by having small BLEU scores, but also significantly different from paraphrased training samples as well.

To quantify how good the synthetic samples are, we propose the following experiment. We test the performance of LLM on the same task using the following 4 different prompts, zero-shot no prompt, three-shot using training set samples with CoT reasoning, three-shot using synthetic samples with CoT reasoning, and three-shot with cohesive but irrelevant paragraphs. The final accuracies are 69.75%, 74.67%, 74.76%, and 70.20% respectively. We find that using synthetics samples as examples in the prompt show a similar performance to using training samples, and they are both better than not using examples or using irrelevant examples. Since the goal of LanFL is to use the synthetic samples to represent the knowledge in the training samples, a similar performance indicate that synthetic samples are indeed good for LanFL operations.

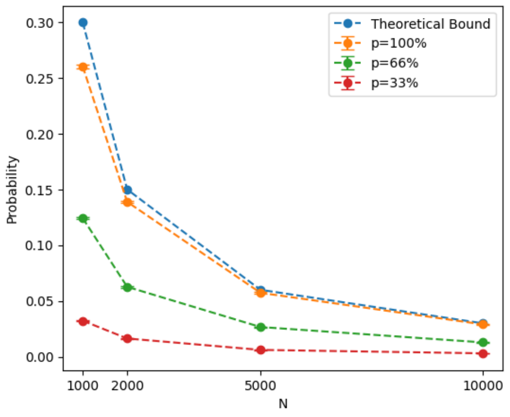


Fig. 6: Probability of success

E. Evaluation of Differential Privacy

To investigate differential privacy in LanFL operations, particularly in synthetic sample generation, we designed an experiment to simulate the following scenario. Consider a participant with a dataset D of size N and a privacy auditor who possesses a percentage p copy of this dataset. The privacy auditor then introduces a new sample from a different participant, adding it to the participant's dataset D and increasing its size to $N+1$, resulting in a new dataset D'. The participant executes the LanFL synthetic sample generation procedure M times with respect to D', producing w synthetic samples in each run, with each sample generated using k original samples. The privacy auditor's goal is to determine, across the M runs, the proportion of instances in which they can identify with certainty that the w synthetic samples were generated using D' instead of D. Specifically, the privacy auditor generates all possible synthetic samples from D. For each of the M runs, if the w synthetic samples are fully contained within the possible synthetic samples of D, the auditor assumes the participant is using D. Conversely, if the w synthetic samples include instances not possible from D, the auditor concludes that the dataset in use is D'. This process serves as a verification method for the differential privacy mechanism introduced in Section III-C, aiming to detect whether the LanFL mechanism leaks information when the underlying dataset differs by a single sample. Based on our analysis, the probability of such leakage is theoretically bounded by $\delta = \frac{kw}{N+1}$.

We conducted experiments by varying N and the percentage p, setting $w = 100$ and $k = 3$ consistent with LanFL experimental setup in the previous sections. Figure 6 illustrates the theoretical upper bound δ for differential privacy across different values of N, alongside the observed probability of successful detection by the privacy auditor. Our findings show that the actual probabilities are consistently below the theoretical values. Furthermore, as N increases or p decreases, the probability of the privacy auditor's success drops significantly, reaching levels below 5%. It is important to highlight that our experimental setup is intentionally biased against the participant, as the privacy auditor is assumed to have access to portions of an exact replica of the dataset and full knowledge of the synthetic sample generation procedure. Despite these unfavorable conditions, our results demonstrate that LanFL operations maintain differential privacy in practical scenarios.

V. Conclusion

In this study, we introduced LanFL, a prompt-based FL scheme specifically designed for LLMs. LanFL offers the advantage of treating models as black boxes, enhancing its applicability in a wide range of real-world scenarios. This scheme integrates breakthroughs in synthetic sample generation with LLMs and sophisticated prompt engineering techniques. As an FL scheme, LanFL safeguards participant privacy through our differentially private synthetic sample generation mechanism and facilitates learning via optimized prompting strategy. Our experiments demonstrate that the proposed LanFL scheme effectively facilitates learning from all participants. Notably, utilizing synthetic samples from other participants enhances test set performance. The results are also robust across various popular LLMs on different tasks.

References

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *International Conference on Artificial Intelligence and Statistics*, 2016.

[2] 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 Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, p. 3347–3366, 2023.

[3] Z. Wang and Q. Hu, "Blockchain-based federated learning: A comprehensive survey," *arXiv preprint arXiv:2110.02182*, 2021.

[4] J. P. Albrecht, "How the GDPR will change the world," *European Data Protection Law Review*, vol. 2, no. 3, 2016.

[5] P. Bukaty, *The California Consumer Privacy Act (CCPA): An implementation guide*. IT Governance Publishing, 2019.

[6] J. Wen, Z. Zhang, Y. Lan, Z. Cui, J. Cai, and W. Zhang, "A survey on federated learning: Challenges and applications," *International Journal of Machine Learning and Cybernetics*, vol. 14, pp. 513–535, 2023.

[7] H. Wu and D. Klabjan, "Robust softmax aggregation on blockchain based federated learning with convergence guarantee," *IEEE International Conference on Omni-layer Intelligent Systems*, 2024.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, p. 6000–6010.

[9] OpenAI, "GPT-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[10] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, and A. Mian, "A comprehensive overview of large language models," *arXiv preprint arXiv:2307.06435*, 2024.

[11] M. Mozes, X. He, B. Kleinberg, and L. D. Griffin, "Use of LLMs for illicit purposes: Threats, prevention measures, and vulnerabilities," *arXiv preprint arXiv:2308.12833*, 2023.

[12] Anthropic, "The Claude 3 model family: Opus, Sonnet, Haiku," *https://www.anthropic.com/claude*, 2024.

[13] G. T. Google, "Gemini: A family of highly capable multimodal models," *arXiv preprint arXiv:2312.11805*, 2024.

[14] B. Chen, Z. Zhang, N. Langrené, and S. Zhu, "Unleashing the potential of prompt engineering in large language models: a comprehensive review," *arXiv preprint arXiv:2310.14735*, 2023.

[15] T. Che, J. Liu, Y. Zhou, J. Ren, J. Zhou, V. Sheng, H. Dai, and D. Dou, "Federated learning of large language models with parameter-efficient prompt tuning and adaptive optimization," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 7871–7888.

[16] G. Kim, J. Yoo, and S. Kang, “Efficient federated learning with pre-trained large language model using several adapter mechanisms,” *Mathematics*, vol. 11, 2023.

[17] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, “Language models are few-shot learners,” in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901.

[18] V. Borisov, K. Seßler, T. Leemann, M. Pawelczyk, and G. Kasneci, “Language models are realistic tabular data generators,” in *The Eleventh International Conference on Learning Representations, ICLR*, 2023.

[19] X. L. Li and P. Liang, “Prefix-tuning: Optimizing continuous prompts for generation,” in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, 2021, pp. 4582–4597.

[20] C. Hou, A. Shrivastava, H. Zhan, R. Conway, T. Le, A. Sagar, G. Fanti, and D. Lazar, “PrE-Text: training language models on private federated data in the age of LLMs,” in *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[21] E. J. Hu, yelong shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “LoRA: Low-rank adaptation of large language models,” in *International Conference on Learning Representations*, 2022.

[22] T. Fan, Y. Kang, G. Ma, W. Chen, W. Wei, L. Fan, and Q. Yang, “FATE-LLM: A industrial grade federated learning framework for large language models,” *arXiv preprint arXiv:2310.10049*, 2023.

[23] N. Su, C. Hu, B. Li, and B. Li, “Titanic: Towards production federated learning with large language models,” *IEEE INFOCOM*, 2024.

[24] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou, “Chain-of-thought prompting elicits reasoning in large language models,” in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2022.

[25] X. Wang, J. Wei, D. Schuurmans, Q. V. Le, E. H. Chi, S. Narang, A. Chowdhery, and D. Zhou, “Self-consistency improves chain of thought reasoning in language models,” in *The Eleventh International Conference on Learning Representations*, 2023.

[26] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. R. Narasimhan, “Tree of thoughts: Deliberate problem solving with large language models,” in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[27] J. Ye, J. Gao, Q. Li, H. Xu, J. Feng, Z. Wu, T. Yu, and L. Kong, “Zerogen: Efficient zero-shot learning via dataset generation,” in *Conference on Empirical Methods in Natural Language Processing*, 2022.

[28] J. Ye, J. Gao, Z. Wu, J. Feng, T. Yu, and L. Kong, “ProGen: Progressive zero-shot dataset generation via in-context feedback,” in *Findings of the Association for Computational Linguistics: EMNLP 2022*, 2022, pp. 3671–3683.

[29] Y. Meng, J. Huang, Y. Zhang, and J. Han, “Generating training data with language models: Towards zero-shot language understanding,” in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 462–477.

[30] R. Xu, H. Cui, Y. Yu, X. Kan, W. Shi, Y. Zhuang, M. D. Wang, W. Jin, J. Ho, and C. Yang, “Knowledge-infused prompting: Assessing and advancing clinical text data generation with large language models,” in *Findings of the Association for Computational Linguistics: ACL 2024*, 2024, pp. 15 496–15 523.

[31] Q. Dong, L. Li, D. Dai, C. Zheng, J. Ma, R. Li, H. Xia, J. Xu, Z. Wu, B. Chang, X. Sun, L. Li, and Z. Sui, “A survey on in-context learning,” in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024, pp. 1107–1128.

[32] A. Talmor, J. Herzig, N. Lourie, and J. Berant, “CommonsenseQA: A question answering challenge targeting commonsense knowledge,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Jun. 2019, pp. 4149–4158.

[33] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “BLEU: a method for automatic evaluation of machine translation,” in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 2002, p. 311–318.

[34] C. Dwork, “Differential privacy,” in *Automata, Languages and Programming*, 2006, pp. 1–12.

[35] Z. Ji, Z. C. Lipton, and C. Elkan, “Differential privacy and machine learning: a survey and review,” *arXiv preprint arXiv:1412.7584*, 2014.

[36] C. Dwork, “Differential privacy: A survey of results,” in *Theory and Applications of Models of Computation*, 2008, pp. 1–19.

[37] M. Sclar, Y. Choi, Y. Tsvetkov, and A. Suhr, “Quantifying language models’ sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting,” in *The Twelfth International Conference on Learning Representations*, 2024.

[38] S. M. Xie, A. Raghunathan, P. Liang, and T. Ma, “An explanation of in-context learning as implicit bayesian inference,” *International Conference on Learning Representations*, 2022.

[39] T. Zhao, E. Wallace, S. Feng, D. Klein, and S. Singh, “Calibrate before use: Improving few-shot performance of language models,” in *International Conference on Machine Learning*, 2021.

[40] MetaAI, “The Llama 3 herd of models,” *arXiv preprint arXiv:2407.21783*, 2024.

[41] MistralAI, “Mixtral of experts,” *arXiv preprint arXiv:2401.04088*, 2024.

[42] I.-C. Yeh, “Default of Credit Card Clients,” UCI Machine Learning Repository, 2016.

[43] T. Pollard, A. Johnson, J. Raffa, L. Celi, R. Mark, and O. Badawi, “The eICU collaborative research database, a freely available multi-center database for critical care research,” *Scientific Data*, 2018.

[44] Google, “PaLM 2 technical report,” *arXiv preprint arXiv:2305.10403*, 2023.

[45] M. Geva, D. Khashabi, E. Segal, T. Khot, D. Roth, and J. Berant, “Did Aristotle use a laptop? a question answering benchmark with implicit reasoning strategies,” *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 346–361, 2021.

Appendix

A. Proof of Theorem 1

Proof. Consider two data sets D_1, D_2 that differ by only one element by addition/deletion in the domain of mechanism f. Without loss of generality, assume $D_1 = D_2 \cup \{x_a\}$ for some element x_a and $|D_1| = c$. We have $|D_2| = c - 1$.

For any subset S in the range of mechanism f, let n_1 be the number of subset $E \subseteq D_2$ such that $|E| = k$ and $L(E) \in S$, and n_2 be the number of subsets $E \subseteq D_1$ such that $|E| = k, x_a \in E$ and $L(E) \in S$. Here n_1 and n_2 may or may not be independent, but individually, their ranges are

$$0 \le n_1 \le \binom{c-1}{k}, \qquad 0 \le n_2 \le \binom{c-1}{k-1}.$$

We then compute

$$P(f(D_1) \in S) = \frac{n_1 + n_2}{\binom{c}{k}}, \qquad P(f(D_2) \in S) = \frac{n_1}{\binom{c-1}{k}}.$$

Taking the difference we have

$$\begin{aligned} P(f(D_1) \in S) - P(f(D_2) \in S) &= \frac{\frac{\binom{c-1}{k}}{\binom{c}{k}} n_1 + \frac{\binom{c-1}{k}}{\binom{c}{k}} n_2 - n_1}{\binom{c-1}{k}} \\ &= \frac{\frac{c-k}{c} n_2 - \frac{k}{c} n_1}{\binom{c-1}{k}} \\ &\le \frac{\frac{c-k}{c} \binom{c-1}{k-1} - \frac{k}{c} * 0}{\binom{c-1}{k}} \\ &= \frac{k}{c}. \end{aligned} \tag{1}$$

Additionally, we also have

$$\begin{aligned}
& P(f(D_2) \in S) - P(f(D_1) \in S) \\
= & \frac{n_1}{\binom{c-1}{k}} - \frac{n_1 + n_2}{\binom{c}{k}} \\
\leq & \frac{n_1}{\binom{c-1}{k}} - \frac{n_1}{\binom{c}{k}} \\
= & \frac{n_1 - \frac{c-k}{c} n_1}{\binom{c-1}{k}} \\
= & \frac{\frac{k}{c} n_1}{\binom{c-1}{k}} \\
\leq & \frac{k}{c}.
\end{aligned} \tag{2}$$

Note that inequality (1) implies that $P(f(D_1) \in S) \leq \frac{k}{c} + e^0 P(f(D_2) \in S)$, and inequality (2) implies that $P(f(D_2) \in S) \leq \frac{k}{c} + e^0 P(f(D_1) \in S)$. Therefore, for any two data sets D_1, D_2 that differ by only one element by addition/deletion in the domain of mechanism f, and for any subset S in the range of mechanism f, we have

$$P(f(D_1) \in S) \leq \frac{k}{m} + e^0 P(f(D_2) \in S),$$

where $m = min(|D_1|, |D_2|) + 1$. By the restriction placed on $dom(f)$, $min(|D_1|, |D_2|) \geq N$, we have $m \geq N + 1$. Note that $\frac{k}{m}$ decreases in m for $0 < k < N + 1$, therefore,

$$P(f(D_1) \in S) \leq \frac{k}{N+1} + e^0 P(f(D_2) \in S).$$

■