PrE-Text: Training Language Models on Private Federated Data in the Age of LLMs

Charlie Hou[2*] **Akshat Shrivastava**[1] **Hongyuan Zhan**[1] **Rylan Conway**[1]
Trang Le[1] **Adithya Sagar**[1] **Giulia Fanti**[2] **Daniel Lazar**[3*]
Meta[1], CMU[2], Coldrays[3]

Abstract

When training machine learning (ML) models over private, federated user data, a common approach is to train models on-device using distributed, differentially-private optimization techniques (e.g., federated learning with DP-SGD). This has several drawbacks: (1) user devices are too resource-limited to train large models on-device, (2) training on-device is communication- and computation-intensive for users, and (3) on-device training can be complicated to deploy. To address these problems, we propose Private Evolution-Text (PrE-Text), a method for generating differentially private (DP) synthetic textual data from federated client data, which can be used to train or finetune language models centrally. First, we show that across multiple datasets, training small models (i.e., models that fit on user devices) with PrE-Text synthetic data outperforms those same model architectures trained on-device in the high privacy regime ($\epsilon = 1.29$), in the sense that they achieve lower test cross-entropy loss. We achieve these results while using 7x less total client computation and 40x less communication than on-device training. Altogether, these results suggest in the high-privacy regime, training on DP synthetic data could be a better option than training models on-device on private distributed data.

1 Introduction

Federated Learning (FL) (McMahan et al., 2017a) is a prominent technique for learning machine learning models from private client data; it trains models on user devices (on-device training) and aggregates the resulting models (model updates) at a central server. FL combined with differential privacy (DP) (Dwork, 2006)—a combination we refer to as DP-FL—protects privacy while also improving model performance in user applications (McMahan et al., 2017b; Kairouz et al., 2021b;a; Nguyen et al., 2022; Xu et al., 2023a).

On-device training or federated learning has several drawbacks. (1) Due to limited on-device storage and computation, client devices cannot be used to train large language models (LLMs) (Radford et al., 2019; Touvron et al., 2023). As LLMs become more critical in many use-cases, this becomes more limiting Charles et al. (2023). (2) On-device training can have high communication and computation costs for clients (Cai et al., 2022). Indeed, there is a large body of literature studying how to improve the efficiency of on-device training (Wang et al., 2020; Li et al., 2020; Karimireddy et al., 2020; Hou et al., 2021; Wang et al., 2021; Mishchenko et al., 2022; Sadiev et al., 2022; Grudzień et al., 2023). (3) On-device training is difficult to deploy and debug (Augenstein et al., 2019), requiring extensive infrastructure investment (Authors, 2018; Zhao et al., 2023; Authors, 2024).

An alternative paradigm: Train or finetune on differentially private (DP) synthetic data. We propose to have the central server first generate DP synthetic data from private client data, then centrally finetune a pretrained language model on that private synthetic data. As in FL, clients send DP information to the server; this is used by the server to generate high-quality synthetic data. Unlike FL, clients do not need to help train the downstream model. Finetuning on DP synthetic data located on-server (1) eliminates the model size constraints of on-device training, (2) is easier to debug as we can observe the training process without compromising DP, and (3) does not require new training infrastructure unlike on-device training.

[*]Work done while at Meta

Table 1: PrE-Text provides the privacy guarantees of on-device training while being nearly as easy to use as centralized non-private training when data is federated among users.

Method	**Train LLMs?** [(a)]	**Is comm. cheap?** [(b)]	**Is comp. cheap?** [(c)]	**Easy to deploy?** [(d)]	**Privacy preserving?**
Private on-device training DP-FedAvg (McMahan et al., 2017b)	✗	✗	✗	✗	✓
Centralized non-private training	✓	✓	✓	✓	✗
PrE-Text (proposed)	✓	✓	✓	✓	✓

[(a)] **Training LLMs.** On-device finetuning of large models (like LLMs) is not feasible because LLMs are too big. PrE-Text allows us to finetune LLMs because the resulting synthetic data is located on-server.
[(b)] **Communication cost.** PrE-Text required 40x less communication per round in our experiments.
[(c)] **Client computation cost.** PrE-Text required 7x less client computation per round on-device training in our experiments. This comes at the cost of server-side computation resources, which is less constrained.
[(d)] **Practicality.** PrE-Text produces synthetic data on-server, which allows practitioners to see the training process end-to-end; this improves debugability (Augenstein et al., 2019). On-device training only returns final model weights.

Unfortunately, existing techniques for generating DP synthetic language data from federated clients are too low-quality to train or fine-tune a lanugage model Augenstein et al. (2019). We fill this gap by leveraging Private Evolution (PE) (Lin et al., 2023), a recent algorithmic breakthrough in DP synthetic data. PE is a framework for generating realistic DP synthetic image data (Lin et al., 2023), which achieves high scores in realism metrics like FID (Heusel et al., 2017). However, Lin et al. (2023) do not apply PE to text, nor do they show that training on DP synthetic data is a competitive alternative to direct DP training (DP-FL or DP-SGD (Abadi et al., 2016)) on private data. Our work utilizes PE in the natural language setting (which is a nontrivial adaptation) as part of our overall algorithm, then demonstrates empirically that the synthetic data can produce competitive models with DP-FL at a fraction of the cost. We list our contributions below:

(1) PrE-Text (Private Evolution-Text) algorithm. We propose PrE-Text, a new algorithm for DP synthetic text generation. We build on the following insights: (1) In Lin et al. (2021), PE must generate variations of samples (e.g., similar images). We adapt this requirement to the language domain by carefully utilizing mask-filling models (Devlin et al., 2018; Lewis et al., 2019) instead of the diffusion models used for images. (2) PE alone does not generate enough high-quality synthetic data to effectively finetune an LLM. Hence, we add a post-processing phase, in which we use the outputs of PE to seed high-quality LLMs trained on public data to generate more similar text.[1]

(2) Experimental results. We produce high-quality DP synthetic language data using PrE-Text, and demonstrate its usefulness for finetuning models that are small enough to fit on user devices (i.e, the kinds of models that might otherwise be trained using FL). We demonstrate that in this setting, models trained on synthetic data produced by PrE-Text achieve similar or better test loss than models trained on-device when $\epsilon = 1.29$ (a privacy used in prior DP-FL literature (McMahan et al., 2017b)), with ~40x less communication per round, ~7x lower client computation per round, and a similar number of rounds. However, when $\epsilon = 7.58$ (a typical setting in the DP-FL literature (Kairouz et al., 2021b)), on-device training outperforms PrE-Text in eval loss. This suggests that, at least in the high-privacy regime ($\epsilon \approx 1$), PrE-Text is a compelling alternative to on-device training.

We present a detailed related work section in Appendix A.2.

[1]In concurrent work released after the submission of our paper, Lin et al. (2021) extend PE to the text domain using similar techniques to ours. Two differences are: (1) their approach relies on a closed-source model (ChatGPT), whereas ours relies on open-source models (LLaMA). This matters because the prompts for synthetic data generation that work on ChatGPT did not work well on LLaMA, so we use different prompts and a post-processing phase to address this issue. (2) Their work does not explore the downstream effects of using synthetic data as a replacement for on-device training, which is the main focus of our work.

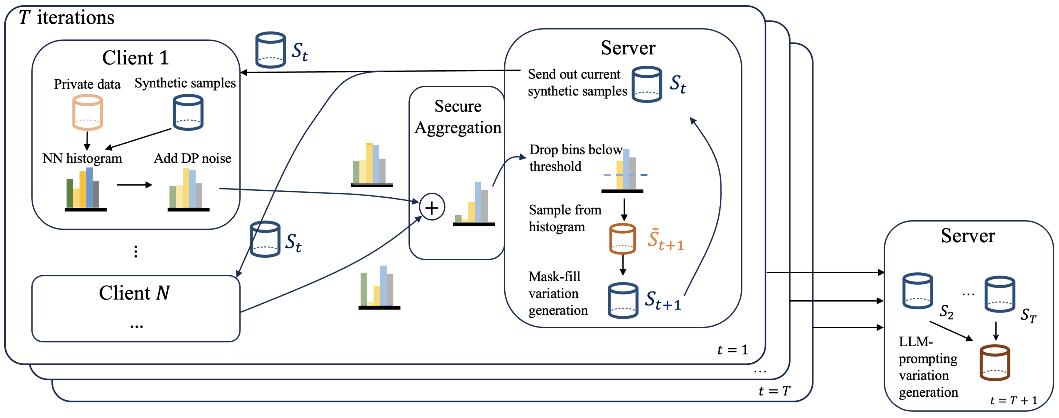


Figure 1: A high-level description of PrE-Text (Section 3). PrE-Text consists of two main phases: (1) (Iterative) DP synthetic seed collection, (2) (Single-shot) synthetic seed expansion.

2 Preliminaries

Definition 2.1 (Neighboring datasets)**.** Two federated datasets X, X' are said to be neighboring (denoted $X \sim X'$) if they differ at most with respect to only one user's data. Note that we consider a *user-level* (McMahan et al., 2017b) notion of neighboring datasets.

Definition 2.2 (Differential Privacy)**.** A randomized algorithm $\mathcal{A}$ is (ϵ, δ)-differentially private (DP) if for any pair of neighboring datasets X, X' and for all subsets E of outputs, we have

$$\Pr[\mathcal{A}(X) \in E] \leq e^{\epsilon} \Pr[\mathcal{A}(X') \in E] + \delta. \tag{1}$$

In this work, we use the Gaussian Mechanism (Dwork, 2006) for DP, which adds Gaussian noise of a specific scale to released statistics. The required scale of noise depends on the *sensitivity* of the statistical query we wish to release.

Definition 2.3 (ℓ_2 sensitivity (Dwork et al., 2014))**.** Let $g : X \to \mathbb{R}^p$ be a vector-valued function operating on datasets. Let X, X' be neighboring datasets. The ℓ_2-sensitivity of g is defined as $\Delta g := \max_{X \sim X'} \|g(X) - g(X')\|_2$.

2.1 Problem formulation

Private clients setup. We consider a setting where a central server wishes to learn a model $\mathcal{M}$ from N user devices (or, "clients"). Client i has the language dataset $\mathcal{D}_i$ which consists of $|S_i|$ language samples. M of these clients, $\mathcal{C}_{\text{test}} \subset [N]$ are considered test clients, and we cannot access their data during synthetic data generation and the training process. The remaining $N - M$ clients, $\mathcal{C}_{\text{train}} \subset [N]$, are considered training clients, and their data can be accessed during model training and/or synthetic data generation. We assume that the client language datasets are drawn from a distribution of possible client datasets $\hat{\mathcal{D}}$, so each $\mathcal{D}_i$ is drawn independently from $\hat{\mathcal{D}}$, $\mathcal{D}_i \sim \hat{\mathcal{D}}$. We study models which can fit on client devices.

Server setup. The server has access to pretrained LLMs (for example, the LLaMA models in our setting (Touvron et al., 2023)) which were trained only on public data.

Task. We focus on the language modeling task, where a language model predicts token s_k from the previous tokens $s_0, \cdots, s_{k-1}$ for each text sample. The server's final goal is to learn a language model that performs well on next-token-prediction on the private test dataset $\mathcal{C}_{\text{test}}$.

Privacy and threat model. We consider an honest-but-curious threat model (Nguyen et al., 2022). Using secure aggregation (Bonawitz et al., 2016), the server does not see individual client uploads, but rather aggregated uploads from all clients. By adding DP noise, clients prevent the server from inferring any single client's data from the aggregated upload. The server aims to learn an (ϵ, δ)-DP language model (central DP); during the training process, the clients' data shared with the server should satisfy a local DP guarantee. We consider user-level DP (McMahan et al., 2017b).

3 PrE-Text

The main intuition of PrE-Text (and PE) is that to generate DP synthetic data similar to private data, we steer the output of a public foundation model (privately) towards the user data in a multi-round process. Briefly, we make several important changes to the PE algorithm: (1) we adapt it to the text setting; (2) we exploit synthetic data from the *entire* PE process, rather than only the last round; and (3) we add a post-processing phase that uses the output of PE as seeds for another LLM. We provide pseudocode in Algorithm 1, and highlight the new contributed steps in light blue.

(1) Population of synthetic samples. We start with an initial population (set) of text samples S_1 (Line 5). These samples can come from many different sources as long as they do not contain private information—for example, public samples collected from the Internet or samples randomly generated by a public generative foundation model.

Algorithm 1 `PrE-Text`

```
    Input: Private data: Clients $\{C_i\}_{i\in\mathcal{C}_{\text{train}}}$
 1:     Number of iterations: $T$
 2:     Number of generated samples: $N_{\text{syn}}$
 3:     Noise multiplier for histogram: $\sigma$
 4:     Threshold for histogram: $H$
 5:     Initial population: $S_1$
 6: Output: Synthetic data $S_{\text{syn}}$
 7: for $t \leftarrow 1 \ldots T$ do
 8:     // SEE ALGORITHM 2
 9:     $\text{hist}_t \leftarrow \texttt{DPNN}(\{C_i\}_{i\in\mathcal{C}_{\text{train}}}, S_t, \sigma, H)$
10:     $P_t \leftarrow \text{hist}_t/\text{sum}(\text{hist}_t)$
11:     $S'_t \leftarrow$ draw $K_{\text{syn}}$ samples from $P_t$
12:     $S_t \leftarrow \texttt{Variation}(S'_t)$
13: end for
14: $S_{\text{syn}} \leftarrow \texttt{Expand}(\bigcup_{t=1}^{T} \texttt{Set}(S'_t))$
15: return $S_{\text{syn}}$
```

Algorithm 2 `DPNN`

```
    Input: Private data: Clients $\{C_i\}_{i\in\mathcal{C}_{\text{train}}}$
 1:     Generated samples $S = \{z_i\}_{i=1}^{n}$
 2:     Number of generated samples: $N_{\text{syn}}$
 3:     Noise for DP-NN histogram: $\sigma$
 4:     Threshold for DP-NN Histogram: $H$
 5:     Distance function: $d(\cdot,\cdot)$
 6: Output: DP-NN histogram on $S$
 7: for $i \in \mathcal{C}_{\text{train}}$ do
 8:     $\text{histogram}_i \leftarrow [0,\ldots,0]$
 9:     for $x_{\text{priv}} \in \mathcal{D}_i$ do
10:         $l = \text{argmin}_{j\in[n]} d(x_{\text{priv}}, z_j)$
11:         $\text{hist}_i[l] \leftarrow \text{hist}[l] + 1$
12:     end for
13:     $\text{hist}_i \leftarrow \text{hist}_i + \mathcal{N}(0, (\sigma/|\mathcal{C}_{\text{train}}|)I)$
14: end for
15: $\text{hist} \leftarrow \sum_{i\in\mathcal{C}_{\text{train}}} \text{hist}_i$
16: $\text{hist} \leftarrow \max(\text{hist} - H, 0)$ (elementwise)
17: return hist
```

(2) Clients vote for the best synthetic samples. For round $t \geq 1$, the clients determine which of the samples in S_t best represent the clients' private samples. In round 1, the samples in S_t may not be synthetic, but for rounds $t > 1$, the samples in S_t will be generated by a generative model. The server first sends all the samples in S_t to each client. Each client counts for each sample $s \in S_t$ how many of its private samples in $\mathcal{D}_i$ had s as their nearest neighbor in S_t. The higher this count, the "better" a generated sample is. Thus, each client produces a nearest neighbors histogram with $|S_t|$ entries (Line 8). We determine nearest neighbors according to a distance function $d(y,z) = \|\Phi(y) - \Phi(z)\|_2$, where Φ is an embedding model.

Lookahead. To more accurately assess the closeness of a synthetic sample to a private sample, we amend the distance function from $d(y,z) = \|\Phi(y)-\Phi(z)\|_2$ to $d(y,z) = \|\Phi(y)-\frac{1}{K}\sum_{i=1}^{K}\Phi(z^i)\|_2$, where $z^0,\ldots,z^K$ are K variations of z produced by using `Variation`. Lin et al. (2023) also use this modification. Instead of sending the actual generated samples directly to all the clients, we send $\frac{1}{K}\sum_{i=1}^{K}\Phi(z^i)$ to all the clients for every i for their nearest neighbors calculation (Line 10).

(a) DP Noise. In Line 13 each client adds noise to their nearest neighbors histogram to ensure differential privacy. We compute client-level sensitivity assuming a known upper bound on the number of samples per client (e.g., via thresholding).

(b) Federated Secure Aggregation In Line 15 we securely aggregate the histograms across the users. Because the generated samples given to all the clients are the same, we sum them using secure aggregation (Bonawitz et al., 2016) to get a histogram representing the "best" generated samples.

(c) Thresholding. When we generate many samples, the majority of the probability mass of the histogram will be noise. We improve the signal-to-noise ratio by thresholding the histogram at H in Line 16 (Lin et al., 2023).

(3) Use votes to choose the surviving samples. We sample from the nearest neighbors histogram to produce surviving generated samples in Line 11, S'_{t+1}. This new list of generated samples (there may be duplicates) tends to give more representation to generated samples that had more private samples close to them.

(4) Produce variations of surviving samples. We use `Variation` to generate a variation of the surviving samples S'_{t+1} as the new population of samples (Line 12). In PE, this was accomplished using diffusion models, which cannot be used for text. We instead implement `Variation` as follows. For each sample in S'_t, we produce a variation of it by masking MASK% of the tokens randomly, then filling in those masked tokens in with a masked language model, like BERT (Devlin et al., 2018). The resulting sample is then masked and filled-in again. This mask-fill process happens W_{steps} times, and then we return the final variation. We use RoBERTa-large (Liu et al., 2019) as the masked language model.

(5) Making efficient use of iterates We make several major modifications to the core Private Evolution algorithm to improve our usage of the iterates. First, Lin et al. (2023) use the final S_T (Line 12) as the synthetic dataset. However, S'_T contains more information about the private dataset than S_T because `Variation` destroys information. Therefore, we use S'_T instead of S_T. Second, we find that S'_t for $t = 1, \dots, T$ all have valuable synthetic samples and show significant diversity between iterations. Therefore, we choose to utilize $\bigcup_{t=1}^{T} \texttt{Set}(S'_t)$ (Line 14) instead of just S'_T. This more effectively utilizes our privacy budget.

(6) Post-processing: Use LLM to expand the DP seed set. PE originally used the final S_T as the target synthetic data. However, we find that these samples are not sufficiently high-quality to finetune an LLM. We instead use `Expand` to generate more samples similar to the synthetic samples $\bigcup_{t=1}^{T} \texttt{Set}(S'_t)$, which we use as our (DP) seed set to prompt the LLM. `Expand` utilizes the synthetic data generation capabilities of large language models (LLMs) to generate a larger and more useful synthetic dataset. Note that by the post-processing property of differential privacy (Dwork, 2006), `Expand` will not leak any additional privacy. Next we describe how `Expand` works.

Inspired by highly successful synthetic text generation (Taori et al., 2023; Wang et al., 2022; Honovich et al., 2022; Rozière et al., 2023), we generate synthetic text by using large foundational language models. Like Honovich et al. (2022), we randomly choose three text samples to emulate from $\bigcup_{t=1}^{T} \texttt{Set}(S'_t)$ (our DP seed set) and then ask a large language model to produce more text samples. We use open-source LLaMA-2-7B (Touvron et al., 2023) as our large language model. We provide the full prompt (Fig. 2).

Privacy analysis. As noted in Lin et al. (2023), the only function that utilizes private information is Algorithm 2: the histogram that counts the quality of each synthetic sample with respect to the private datasets (Line 15) contains private information. Like Lin et al. (2023), we use the Gaussian mechanism with constant noise multiplier σ each time we receive a histogram. As this is a basic Gaussian mechanism, we can use the moments accountant from the Opacus library (Yousefpour et al., 2021). In particular, we use OPACUS.ACCOUNTANTS.ANALYSIS.RDP to compute the privacy guarantee for both the DP-FL baseline (it is built into FLSim (Zhao et al., 2023)) and PrE-Text (we input $T = 11$ steps, $q = 1.0$, and set the NOISE_MULTIPLIER to be the ratio of σ to the sensitivity (the max number of samples per client for PrE-Text), setting σ to the value that gets us the desired ϵ value).

4 Experiments

Our goal in this section is to compare PrE-Text against the most common technique for privacy-preserving on-device learning, DP-FL.

Models. To compare with DP-FL, we require a downstream language model that can fit on device; we use DistilGPT2 as a representative such model, which has 82M parameters. We use RoBERTa-large (Liu et al., 2019) for mask-filling. To compute text embeddings, we use all-MiniLM-L6-v2, all-mpnet-v2 (Reimers & Gurevych, 2019), BART-base, and BART-large (Lewis et al., 2019).

Table 2: We compare the eval cross-entropy losses achieved by FL and PrE-Text, under $(1.29, 3 \times 10^{-6})$-DP and $(7.58, 3 \times 10^{-6})$-DP. We see that PrE-Text either outperforms or performs similarly to DP-FL on all datasets when $\epsilon = 1.29$, but the opposite is true when $\epsilon = 7.58$. These test performances come at a dramatic reduction in communication and per-round runtime (Section 4).

Method	Privacy Budget	JOBS (↓)	FORUMS(↓)	MICROBLOG (↓)
DP-FL	$\epsilon = 1.29$	4.185 ± 0.014	4.258 ± 0.021	4.317 ± 0.080
PrE-Text		**3.814 ± 0.001**	4.284 ± 0.005	**4.070 ± 0.002**
DP-FL	$\epsilon = 7.58$	**3.073 ± 0.018**	**3.157 ± 0.007**	**3.243 ± 0.024**
PrE-Text		3.702 ± 0.004	4.268 ± 0.001	4.053 ± 0.005

MiniLM-L6-v2 and all-mpnet-v2 are designed to capture the meanings of sentences and are meant to be more invariant to stylistic language choices. BART-base and BART-large are mask-filling models, so their embeddings are meant to be sensitive to style choices. Finally, we use LLaMA-2-7B (Touvron et al., 2023) for synthetic seed expansion.

Datasets. We use the c4 English (c4-en) dataset (Raffel et al., 2019) to construct 3 datasets: JOBS, FORUMS, and MICROBLOG which are federated datasets of 1250 clients each. JOBS, FORUMS, and MICROBLOG are constructed from text found on linkedin.com, reddit.com, and twitter.com respectively. We give details on their construction in Appendix B.2.

Baseline. We use the FedAvg federated optimization algorithm (McMahan et al., 2017a) to fully finetune DistilGPT2 (we do not use linear probing, as it has been shown to perform poorly in DP language models (Li et al., 2021)). We perform full participation where every client participate in every round, both for DP-FL and for PrE-Text. We ensure that the training has a privacy guarantee of (ϵ, δ)-DP where $\delta = 3 \cdot 10^{-6}$ and $\epsilon = 1.29$ or $\epsilon = 7.58$, to evaluate two privacy regimes. We include hyperparameter tuning details on learning rate, batch size, sequence length, and communication rounds in Appendix B.3. We report the best validation loss observed.

PrE-Text. We first perform PrE-Text such that the privacy guarantee is $(1.29, 3 \cdot 10^{-6})$ or $(7.58, 3 \cdot 10^{-6})$ DP depending on the privacy regime to produce 2M synthetic samples. We keep roughly the same number of communication rounds with DP-FL (10 rounds for DP-FL vs 11 for PrE-Text). Then we fully finetune DistilGPT on the synthetic samples. More details on the parameters for PrE-Text T, N_{syn}, σ, Φ, H, W_{steps}, MASK% as well as the finetuning parameters are reported in Appendix B.3.

Task. We focus on the language modeling task, and report evaluation cross entropy losses as our evaluation metric.[2]

Results. Table 2 gives the evaluation cross-entropy loss achieved by DP-FL and PrE-Text at two different privacy regimes. We see that PrE-Text either outperforms or performs similarly to the FL baseline on every dataset when $\epsilon = 1.29$ (comparable to McMahan et al. (2017b), which has $\epsilon = 1.152$) while the DP-FL baseline outperforms PrE-Text on all datasets when $\epsilon = 7.58$ (comparable to Kairouz et al. (2021b), which has $\epsilon = 7.51$). The results show that in the high privacy regime, PrE-Text is able to outperform DP-FL. As synthetic data generation methods improve (for example, better prompts and models for `expand`), it is plausible that synthetic data may outperform DP-FL in other privacy regimes as well.

Efficiency differences. We compare the efficiency of DP-FL vs PrE-Text for clients with 8 samples each in Section 4; a detailed explanation of these numbers can be found in Appendix B.1. These results demonstrate that PrE-Text is more efficient than DP-FL in per-round runtime (7× improvement) and communication (40× improvement).

5 CONCLUSION

We propose PrE-Text, a method for generating DP synthetic text from user devices. Our results suggest that one can use PrE-Text's DP synthetic text data generation process to replace DP-FL

[2]Code is provided at https://anonymous.4open.science/r/PrE-Text-8745/README.md

Method	**Per-Round Communication** (Number of floats transmitted)		**Per-Round Runtime** (s)
	Upload	*Download*	
DP-FL	82M	82M	24
PrE-Text	1024	1.8M	3.1
Reduction	80,000×	46×	7.7×

Table 3: Communication and runtime cost of PrE-Text vs DP-FL. We find that PrE-Text achieves at least a 40× reduction in communication and a 7× reduction in per-round runtime in our experiments. These results are dataset-agnostic and run for private clients with 8 samples each.

in the high privacy regime; however, we have not observed similar gains in the low-privacy regime. Understanding exactly when PrE-Text outperforms DP-FL is an important direction for future work.

References

United states median country speeds october 2023. `https://www.speedtest.net/global-index/united-states`, 2023. Accessed: 2023-11-27.

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

Naman Agarwal, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, and Brendan McMahan. cpsgd: Communication-efficient and differentially-private distributed sgd. *Advances in Neural Information Processing Systems*, 31, 2018.

Sean Augenstein, H Brendan McMahan, Daniel Ramage, Swaroop Ramaswamy, Peter Kairouz, Mingqing Chen, Rajiv Mathews, et al. Generative models for effective ml on private, decentralized datasets. *arXiv preprint arXiv:1911.06679*, 2019.

The PyTorch Mobile Authors. PyTorch Mobile, 2024. URL `https://pytorch.org/mobile/home/`.

The TensorFlow Federated Authors. TensorFlow Federated, December 2018. URL `https://github.com/tensorflow/federated`.

K. A. Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for federated learning on user-held data. In *NIPS Workshop on Private Multi-Party Machine Learning*, 2016. URL `https://arxiv.org/abs/1611.04482`.

Dongqi Cai, Yaozong Wu, Shangguang Wang, Felix Xiaozhu Lin, and Mengwei Xu. Fedadapter: Efficient federated learning for modern nlp. *arXiv preprint arXiv:2205.10162*, 2022.

Tianshi Cao, Alex Bie, Arash Vahdat, Sanja Fidler, and Karsten Kreis. Don't generate me: Training differentially private generative models with sinkhorn divergence. *Advances in Neural Information Processing Systems*, 34:12480–12492, 2021.

George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4750–4759, 2022.

Zachary Charles, Kallista Bonawitz, Stanislav Chiknavaryan, Brendan McMahan, et al. Federated select: A primitive for communication-and memory-efficient federated learning. *arXiv preprint arXiv:2208.09432*, 2022.

Zachary Charles, Nicole Mitchell, Krishna Pillutla, Michael Reneer, and Zachary Garrett. Towards federated foundation models: Scalable dataset pipelines for group-structured learning. *arXiv preprint arXiv:2307.09619*, 2023.

Jinyu Chen, Wenchao Xu, Song Guo, Junxiao Wang, Jie Zhang, and Haozhao Wang. Fedtune: A deep dive into efficient federated fine-tuning with pre-trained transformers. *arXiv preprint arXiv:2211.08025*, 2022.

Liam Collins, Shanshan Wu, Sewoong Oh, and Khe Chai Sim. Profit: Benchmarking personalization and robustness trade-off in federated prompt tuning. *arXiv preprint arXiv:2310.04627*, 2023.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Tim Dockhorn, Tianshi Cao, Arash Vahdat, and Karsten Kreis. Differentially private diffusion models. *arXiv preprint arXiv:2210.09929*, 2022.

Cynthia Dwork. Differential privacy. In *International colloquium on automata, languages, and programming*, pp. 1–12. Springer, 2006.

Cynthia Dwork, Kunal Talwar, Abhradeep Thakurta, and Li Zhang. Analyze gauss: optimal bounds for privacy-preserving principal component analysis. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pp. 11–20, 2014.

Antonious M Girgis, Deepesh Data, Suhas Diggavi, Peter Kairouz, and Ananda Theertha Suresh. Shuffled model of federated learning: Privacy, accuracy and communication trade-offs. *IEEE journal on selected areas in information theory*, 2(1):464–478, 2021.

Michał Grudzień, Grigory Malinovsky, and Peter Richtárik. Can 5th generation local training methods support client sampling? yes! In *International Conference on Artificial Intelligence and Statistics*, pp. 1055–1092. PMLR, 2023.

Xin Gu, Gautam Kamath, and Steven Wu. Choosing public datasets for private machine learning via gradient subspace distance. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023.

Tao Guo, Song Guo, Junxiao Wang, Xueyang Tang, and Wenchao Xu. Promptfl: Let federated participants cooperatively learn prompts instead of models-federated learning in age of foundation model. *IEEE Transactions on Mobile Computing*, 2023.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. Unnatural instructions: Tuning language models with (almost) no human labor. *arXiv preprint arXiv:2212.09689*, 2022.

Charlie Hou, Kiran K Thekumparampil, Giulia Fanti, and Sewoong Oh. Fedchain: Chained algorithms for near-optimal communication cost in federated learning. *arXiv preprint arXiv:2108.06869*, 2021.

Charlie Hou, Hongyuan Zhan, Akshat Shrivastava, Sid Wang, Sasha Livshits, Giulia Fanti, and Daniel Lazar. Privately customizing prefinetuning to better match user data in federated learning. *arXiv preprint arXiv:2302.09042*, 2023.

Peter Kairouz, Ziyu Liu, and Thomas Steinke. The distributed discrete gaussian mechanism for federated learning with secure aggregation. In *International Conference on Machine Learning*, pp. 5201–5212. PMLR, 2021a.

Peter Kairouz, Brendan McMahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. Practical and private (deep) learning without sampling or shuffling. In *International Conference on Machine Learning*, pp. 5213–5225. PMLR, 2021b.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021c.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pp. 5132–5143. PMLR, 2020.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019. URL `http://arxiv.org/abs/1910.13461`.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. *arXiv preprint arXiv:2110.05679*, 2021.

Zinan Lin, Vyas Sekar, and Giulia Fanti. On the privacy properties of gan-generated samples. In *International Conference on Artificial Intelligence and Statistics*, pp. 1522–1530. PMLR, 2021.

Zinan Lin, Sivakanth Gopi, Janardhan Kulkarni, Harsha Nori, and Sergey Yekhanin. Differentially private synthetic data via foundation model apis 1: Images. *arXiv preprint arXiv:2305.15560*, 2023.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL `http://arxiv.org/abs/1907.11692`.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 20–22 Apr 2017a. URL `https://proceedings.mlr.press/v54/mcmahan17a.html`.

H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963*, 2017b.

Konstantin Mishchenko, Grigory Malinovsky, Sebastian Stich, and Peter Richtárik. Proxskip: Yes! local gradient steps provably lead to communication acceleration! finally! In *International Conference on Machine Learning*, pp. 15750–15769. PMLR, 2022.

John Nguyen, Kshitiz Malik, Hongyuan Zhan, Ashkan Yousefpour, Mike Rabbat, Mani Malek, and Dzmitry Huba. Federated learning with buffered asynchronous aggregation. In *International Conference on Artificial Intelligence and Statistics*, pp. 3581–3607. PMLR, 2022.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*, 2019.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL https://arxiv.org/abs/1908.10084.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

Abdurakhmon Sadiev, Dmitry Kovalev, and Peter Richtárik. Communication acceleration of local gradient methods via an accelerated primal-dual algorithm with an inexact prox. *Advances in Neural Information Processing Systems*, 35:21777–21791, 2022.

Rui Song, Dai Liu, Dave Zhenyu Chen, Andreas Festag, Carsten Trinitis, Martin Schulz, and Alois Knoll. Federated learning via decentralized dataset distillation in resource-constrained edge environments. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10. IEEE, 2023.

Xinyu Tang, Richard Shin, Huseyin A Inan, Andre Manoel, Fatemehsadat Mireshghallah, Zinan Lin, Sivakanth Gopi, Janardhan Kulkarni, and Robert Sim. Privacy-preserving in-context learning with differentially private few-shot generation. *arXiv preprint arXiv:2309.11765*, 2023.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623, 2020.

Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*, 2022.

Zheng Xu, Maxwell Collins, Yuxiao Wang, Liviu Panait, Sewoong Oh, Sean Augenstein, Ting Liu, Florian Schroff, and H Brendan McMahan. Learning to generate image embeddings with user-level differential privacy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7969–7980, 2023a.

Zheng Xu, Yanxiang Zhang, Galen Andrew, Christopher A Choquette-Choo, Peter Kairouz, H Brendan McMahan, Jesse Rosenstock, and Yuanbo Zhang. Federated learning of gboard language models with differential privacy. *arXiv preprint arXiv:2305.18465*, 2023b.

Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. Opacus: User-friendly differential privacy library in PyTorch. *arXiv preprint arXiv:2109.12298*, 2021.

Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, et al. Differentially private fine-tuning of language models. *arXiv preprint arXiv:2110.06500*, 2021.

Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Guoyin Wang, and Yiran Chen. Towards building the federated gpt: Federated instruction tuning. *arXiv preprint arXiv:2305.05644*, 2023a.

Tuo Zhang, Tiantian Feng, Samiul Alam, Mi Zhang, Shrikanth S Narayanan, and Salman Avestimehr. Gpt-fl: Generative pre-trained model-assisted federated learning. *arXiv preprint arXiv:2306.02210*, 2023b.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*, 2020.

Haodong Zhao, Wei Du, Fangqi Li, Peixuan Li, and Gongshen Liu. Reduce communication costs and preserve privacy: Prompt tuning method in federated learning. *arXiv preprint arXiv:2208.12268*, 2022.

Jessica Zhao, John Nguyen, and Sandesh Shetty. Flsim. `https://github.com/facebookresearch/FLSim`, 2023.

A Appendix

A.1 Ablation studies

We chose the hyperparameters in our algorithm based on results from empirical evaluations. In the following section we study the effect of changes in the following parameters: N_{syn} (the number of synthetic samples generated per round by the server before the `Expand` part of PrE-Text, the choice of embedding vector, and M_STEPS (the amount to vary each sample when executing `Variation`).

Experimental setup. In this section, we start with a baseline setting for PrE-Text and change one factor at a time. The baseline setting is as follows: we set $N_{\text{syn}} = 1024$, the embedding vector is computed from concatenating all-mpnet-v2 and BART-large, $W_{\text{steps}} = 2$, and MASK% = 30%. During `Expand` we generate 100,000 samples, and set the context length during generation and during the finetuning evaluation to be 64. The model we finetune is DistilGPT2, and we use a batch size of 8192. We finetune for 10 epochs and use the final eval loss. We use only the JOBS dataset. Otherwise, we follow the experimental setup of **??**. All results are means of eval cross entropy over three trials.

1. Choosing N_{syn}. We use the baseline setting and vary N_{syn}. Unlike in Lin et al. (2023), we see larger N_{syn} is not always better. Moreover, we found that N_{syn} of 1024 or 4096 work well. As a result of this ablation, we chose to use $N_{\text{syn}} = 1024$ in our experiments.

N_{syn}	1024	2048	4096
JOBS (↓)	**4.374**	4.380	4.372

2. Choosing the embedding model. In this section, we compare three embedding models: BART-large (Lewis et al., 2019) (bart), all-mpnet-v2 (Reimers & Gurevych, 2019), and the concatenation of the two (mpnet+bart). Note BART-large is a model that fixates more on sentence structure and style, while all-mpnet-v2 (mpnet) focuses on meaning.

Embedding	mpnet	bart	mpnet+bart
JOBS (↓)	4.414	4.410	**4.374**

We find that mpnet+bart performs the best. For our main experiments, we chose to use mpnet+bart for Φ.

3. Choosing the amount of variation. In this section, we study the impact of the amount and type of variation. MASK% controls the amount of masking per step and W_{steps} is the number of times we mask and fill per communication round. We find that $W_{\text{steps}} = 2$, and MASK% = 30%.

MASK% W_{steps}	0.3 2	0.15 5	0.3 4	0.15 1
JOBS (↓)	**4.374**	4.391	4.379	4.394

A.2 RELATED WORK

DP Federated Learning. Differentially Private Federated Learning (FL) is a widely-used approach for learning ML models from distributed private data (McMahan et al., 2017b; Kairouz et al., 2021c). In DP-FL, model weights are sent to users who train the model locally on their private data; private local model updates are then collected at a central server. Researchers have many techniques for improving privacy-utility tradeoffs in DP-FL. Some include shuffle-based privacy amplification Bonawitz et al. (2016); Girgis et al. (2021); Agarwal et al. (2018), pretraining on public datasets Xu et al. (2023b), private selection of the best pretraining datasets Hou et al. (2023); Gu et al. (2022), and DP-FL methods that do not rely on uniform sampling/shuffling (Kairouz et al., 2021b).

Today, growing efforts study how to train larger models on client data. Charles et al. (2022) propose to have users optimize slices of large models, though they have not demonstrated the approach on models larger than shallow logistic regression and convolutional neural network models. Collins et al. (2023); Cai et al. (2022); Zhang et al. (2023a); Zhao et al. (2022); Guo et al. (2023) only tune sub-components of the models in the federated setting to reduce client computational burden and communication, but this still requires having clients store and perform inference with large models on their device. Zhang et al. (2023b) use foundation models to produce synthetic data to pretrain smaller models. None of these methods consider privacy. Prior to our work, there was no clear path on how to train large models over federated client data, even without privacy.

Synthetic Data. Synthetic data is increasingly being used to train language models (Taori et al., 2023; Wang et al., 2022; Honovich et al., 2022). Common approaches involve carefully designing prompts to ChatGPT (Radford et al., 2019) to generate synthetic training data for another open source model, e.g. LLaMA (Touvron et al., 2023), to replicate ChatGPT behavior. In these works, the synthetic data is used solely to enhance final model utility, and does not satisfy any formal privacy guarantees. In the image setting, useful synthetic data is often produced using dataset distillation (Wang et al., 2018; Zhao et al., 2020; Cazenavette et al., 2022). This approach has been adapted to the federated setting (Song et al., 2023).

DP Synthetic Data. Much of the work on producing *private* (i.e., DP) synthetic data from deep generative models is in the image domain (Lin et al., 2021; Cao et al., 2021; Dockhorn et al., 2022; Chen et al., 2022). Common approaches involve training a generative adversarial network (GAN) or diffusion model in a differentially private way, e.g., via DP-SGD, a DP version of stochastic gradient descent (SGD) (Abadi et al., 2016). Tang et al. (2023) generate DP synthetic text in the federated setting, but their method requires users to send private data to ChatGPT (which is located on-server). Such actions are not allowed under our threat model, where the central server is not trusted to hold private data. They do not train any models on their synthetic data.

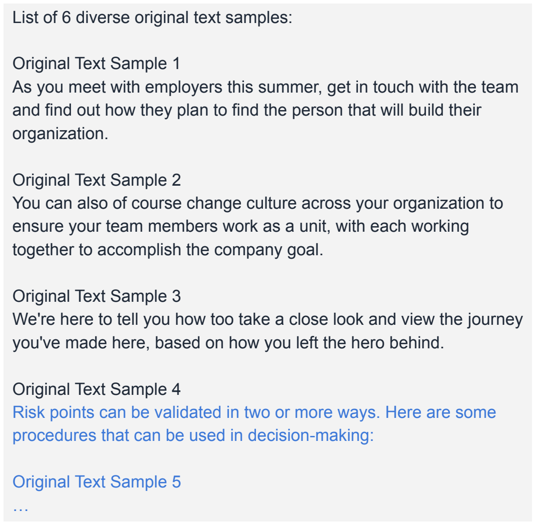
DP Language Models. In the text domain, (Li et al., 2021; Yu et al., 2021) demonstrate that it is possible to finetune pretrained large language models in a central-DP manner (i.e., the private data is available to the model developer). This is a different privacy setting from ours.

B PROMPT

We provide the prompt used for `expand` here.

B.1 DETAILED EFFICIENCY ANALYSIS

(1) Communication cost. DP-FL requires each client to download and upload the model (DistilGPT2) to/from the server. DistilGPT2 has a size of 82M parameters, which means clients are downloading and uploading 82M floats each round. On the other hand, PrE-Text requires clients to download 1024 embedding vectors of size 1792 representing the synthetic samples each round. This means each client downloads $1024 \times 1792 = 1.8$M floats. On upload, clients upload 1024


List of 6 diverse original text samples:

Original Text Sample 1
As you meet with employers this summer, get in touch with the team and find out how they plan to find the person that will build their organization.

Original Text Sample 2
You can also of course change culture across your organization to ensure your team members work as a unit, with each working together to accomplish the company goal.

Original Text Sample 3
We're here to tell you how too take a close look and view the journey you've made here, based on how you left the hero behind.

Original Text Sample 4
Risk points can be validated in two or more ways. Here are some procedures that can be used in decision-making:

Original Text Sample 5
…

Figure 2: The synthetic data generation prompt for `Expand`. The blue text after "Original Text Sample 4" is generated. We parse the generated text for the text between Original Text Sample 4 and Original Text Sample 5 and use that as a synthetic sample.

floats (the size of the histogram). So client download cost is 40$\times$ cheaper with PrE-Text per round, and upload cost is 80,000$\times$ cheaper with PrE-Text per round. Conservatively, this is at least a 40$\times$ improvement in communication cost per round when using PrE-Text (likely much more, given that upload speeds are usually 15x slower than download (spe, 2023)). We do not account for downloading the embedding models such as BART-large or all-mpnet-v2, as this can be done offline in one communication round (not during the course of running the algorithm).

(2) Client computation cost. PrE-Text is also much more computation-efficient for the clients. We assume that user devices only have access to CPU computation, as most smartphones do not have GPUs (or have fairly limited GPUs). When tested on a VM with five Intel(R) Xeon(R) Gold 6248 CPU @ 2.50GHz, training on 8 samples (each client has 8 samples in our experiment) using DistilGPT2 takes 24 seconds while the client computation component of PrE-Text takes 1.18 seconds for the private embedding calculation and 2 seconds for the nearest neighbors calculation. This computation gain comes from the fact that clients perform inference rather than training: inference requires fewer operations than training and can be sped up with various tricks (Kwon et al., 2023; Reimers & Gurevych, 2019). This gives at least a 6x improvement in computation cost per round for clients with PrE-Text. Meanwhile, the total number of rounds used is similar (10 or 100 for DP-FL and 11 for PrE-Text). This comes at the cost of more server-side compute in our experiments. However, using server compute is often more acceptable.

Hence, PrE-Text's DP synthetic language data offers many practical advantages over training on-device: (1) comparable or better model performance, (2) efficiency gains (3) the ability to inspect (synthetic) training data (4) no need to implement FL training infrastructure like TensorFlow-Federated, Pytorch-iOS, etc.

B.2 Dataset construction details

We use the train split of the c4 English (c4-en) dataset (Raffel et al., 2019). We start by producing three federated private datasets from c4-en: Jobs, Forums, and Microblog. We illustrate the process for Jobs; the process is similar for the other two. First, we take the first 10,000 samples in the c4-en dataset that come from linkedin.com. The private evaluation set consists of the next 1,000 samples in the c4-en dataset from linkedin.com. The federated dataset is then constructed by splitting the 10,000 training samples into 1250 clients of 8 samples each. The same is done for Forums (from reddit.com) and Microblog (from twitter.com). For the initial population used in Algorithm 1, we take random samples from c4-en that are not in the private training sets. Note that many LLMs do not document what datasets were used in the pretraining, which means the only sure way to prevent dataset contamination is to use data that was created after the release of the model. (Touvron et al., 2023). We used the most recent large-scale dataset we could find (though it was released before LLaMA-2) that is (a) compliant with terms of service (for example, many sources of recent text data have closed their APIs for ML training) and (b) readily accessible–figuring out ways to systematically detect consequential dataset contamination is an important open problem in large language model research (Gunasekar et al., 2023)

B.3 EXPERIMENTAL DETAILS

B.3.1 FEDAVG EXPERIMENTAL DETAILS

We use the FedAvg federated optimization algorithm (McMahan et al., 2017a) to fully finetune DistilGPT2 (we do not use linear probing, as it has been shown to perform poorly in DP language models (Li et al., 2021)). We use a batch size of 2, a sequence length of 64, and full participation when training (i.e. every client participates in every round). For DP, we use secure aggregation (Bonawitz et al., 2016) and add Gaussian noise (McMahan et al., 2017b). We report evaluation cross entropy scores across different choices for the number of training epochs over the clients. We tune the learning rate in $\{3.0, 1.0\}$ (Kairouz et al., 2021b), the number of communication rounds in $\{10, 100\}$ and choose the model that performs the best on the evaluation set to report. The noise is scaled such that the resulting privacy guarantee will be (ϵ, δ)-DP where $\delta = 3 \cdot 10^{-6}$ and $\epsilon = 1.29$ or $\epsilon = 7.58$, to evaluate two different privacy regimes.

B.3.2 PRE-TEXT EXPERIMENTAL DETAILS

We use $T = 11$, $N_{\text{syn}} = 1024$. We scale σ such that the privacy guarantee is $\delta = 3 \cdot 10^{-6}$ and $\epsilon = 1.29$ (with $H = 94.4$) or $\epsilon = 7.58$ (with $H = 16.0$). The Φ we use is a concatenation of the mean pooled representation from BART-large and the representation from all-mpnet-v2 (i.e., we pass a text sample through BART-large to get a vector of dimension 1024, then pass the same text sample through all-mpnet-v2 to get a vector of dimension 768, and concatenate these vectors together to get a vector of dimension 1792 that altogether represents the text sample). In `Variation` we use RoBERTa-large as the mask filling model with top_p parameter set to 1.0 and temperature set to 1.0, $W_{\text{steps}} = 2$, and $\text{MASK}\% = 30\%$. In `Expand` we use LLaMA-2-7B with top_p set to 1.0 and temperature set to 1.0. When implementing `Expand` we use the library vLLM to speed up inference (Kwon et al., 2023). DistilGPT2 is then fully finetuned on 2 million DP synthetic samples that were generated using `Expand`. Finetuning is done for 10 epochs with a batch size of 65536. We report the best validation loss, as done in the baselines.