On Privacy and Personalization in Machine Learning

Ken Ziyu Liu

CMU-RI-TR-23-17

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania
United States

May 2023

Thesis Committee

Prof. Virginia Smith	MLD
Prof. Artur Dubrawski	RI
Prof. Zhiwei Steven Wu	S3D/MLD
Prof. Elaine Shi	CSD
Shengyuan Hu	MLD

*Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Robotics*

Abstract

While the application of differential privacy (DP) has been well-studied in cross-device federated learning (FL), there is a relative lack of work considering DP and its implications for cross-silo FL, a setting characterized by a limited number of clients each containing many data subjects. In cross-silo FL, the usual notions of *client-level* DP are less suitable as real-world privacy regulations typically concern the in-silo data subjects (e.g. persons) rather than the data silos themselves.

This thesis explores the notion of *silo-specific example-level DP* as a more appropriate privacy model, where data silos set their own privacy targets for their local examples. We establish *mean-regularized multi-task learning* (MR-MTL) as a strong baseline for private cross-silo learning and provide both empirical and theoretical analyses to highlight the interplay between privacy, utility, and statistical heterogeneity across data silos.

In addition, we showcase the practical application of our research through the US/UK PETs Prize Challenge,[1] where our CMU team[2] ("puffle") developed a 1st place solution for the Pandemic Forecasting and Response track[3] based on the research presented in this thesis.

Our work also serves to identify key directions for future research in this area, such as developing auto-tuning algorithms for private model personalization, exploring alternative privacy notions for imbalanced data and graph analyses, and investigating novel training frameworks under the honest privacy cost of hyperparameter search.

[1] https://petsprizechallenges.com/

[2] https://drivendata.co/blog/federated-learning-pets-prize-winners-phases-2-3#puffle

[3] https://www.whitehouse.gov/ostp/news-updates/2023/03/31/us-uk-annouce-winners-innovation-pets-democratic-values/

Contents

List of Figures

List of Tables

Relevant Publications

Ziyu Liu, Shengyuan Hu, Zhiwei Steven Wu, Virginia Smith. "On Privacy and Personalization in Cross-Silo Federated Learning."

- In *Advances in Neural Information Processing Systems* (NeurIPS), 2022.
- Presented at *Theory and Practice of Differential Privacy* (TPDP) workshop at ICML 2022.

Ziyu Liu, Shengyuan Hu, Tian Li, Zhiwei Steven Wu, Virginia Smith. "Pushing the Privacy-Utility Frontier with Heterogeneity-Aware FL." Technical report for the UK/US PETs Prize Challenge (`https://petsprizechallenges.com`), 2023. Content presented through:

- An ML@CMU blog post at `https://blog.ml.cmu.edu`.
- Full solution documentation at `https://kenziyuliu.github.io/blog/pets-challenge`.
- Chapter 2 of this thesis.

Acknowledgements

First and foremost, I would like to thank my advisors at CMU, without whom this thesis (or any research output during my time here) would not have been possible: Virginia Smith, Artur Dubrawski, and (unofficially) Steven Wu. I have been extremely lucky to have Virginia as my research advisor and mentor—not only did she provide me with research directions and crucial guidance, she is also extremely kind and encouraging, even when my projects were not going well. She also funded my studies at CMU while giving me substantial freedom to explore my interests, and I'm extremely grateful for her support. Steven has also been a joy to work with—he always gives insightful feedback on my research problems, encourages me to explore new ideas on top of our current projects, and is just all-around a fun advisor to have (particularly when Snuggle the Corgi is present). I am also very fortunate to have Artur's support—without him, I would not have been able to set up the wonderful advising relationships; to access the GPUs for all the experiments in this thesis; and to think more deeply about the practical aspects of research. I'd also like to thank Elaine Shi for serving on my committee and for providing valuable feedback.

I'm also grateful for my research buddies Shengyuan Hu, Tian Li, and Sebastian Caldas. Shengyuan and Tian have been wonderful labmates and great friends—together, we went through numerous whiteboard sessions, paper discussions, or long chats about life in grad school and beyond. We collaborated on multiple projects that led to this thesis. Sebastian has been a great source of emotional support that I didn't know I needed—he is always there to champion my research and to give me feedback on how to navigate grad school. They have been extremely generous with their time and I have learned so much from them.

I'd also like to thank my mentors from Google Research before CMU: Peter Kairouz and Jakub Konečný. They jumpstarted my research career in private machine learning by introducing me to the field, giving me the opportunities to work on exciting projects, teaching me how to approach research problems and engineer practical systems, and connecting me to the broader research community, including my advisors here at CMU. I wouldn't be here or anywhere beyond without them.

All of the people above are a major reason that I decided to continue into a PhD. I have learned first-hand that good mentorship and lab culture can have a tremendous positive impact on a junior student, and it is, afterall, possible to have fun while doing research in academia. Fortunately, I will have opportunities to continue working with them in the future!

I'm also very grateful for the friends and the fun we had together outside of research at CMU: Amrith Setlur, Bowei Chen, Carl Qi, Chuer Pan, Fan Yang, Gaoyue Zhou, Hanzhe Hu, Haoyu Xiong, Harry Zhang, Heng Yu, Henry Xu, Jack Chen, Jianren Wang, Jinqi Luo, Lun Wang, Meng Zhou, Muyang Li, Pengwei Sui, Pratiksha Thakar, Quanting Xie, Sally Chen, Shivam Duggal, Tianyi Zhang, Tianyuan Zhang, Xuxin Cheng, Yafei Hu, Yi Zhou, Yinong Wang, Yucheng Li, Yuyao Shi, Zhize Li, Zhizhuo Zhou, Zipeng Fu, Zixuan Huang, Ziyun Chi, and many others!

On the personal side, I would like to thank my parents and my siblings for their unconditional support. I have been blessed that they have cultivated an environment where I can freely pursue my goals without burden and worry. The acknowledgement section is too short to express my deepest gratitude.

Finally, I want to thank Amy for her unconditional love and support, and for being there with me through the ups and downs for the past many years.

Chapter 1

Privacy, Personalization, and Cross-Silo Federated Learning

In recent years, differential privacy (DP) has gained significant attention in the field of cross-device federated learning (FL). However, the application and understanding of DP in cross-silo FL, where a limited number of clients each possess data from numerous subjects, remains underexplored. In cross-silo FL, usual notions of client-level DP are less suitable as real-world privacy regulations typically concern the in-silo data subjects rather than the silos themselves. This chapter delves into the notion of *silo-specific example-level DP*, where individual silos set their privacy targets for their own local training examples.

Under this setting, we reconsider the roles of personalization in federated learning. In particular, we show that mean-regularized multi-task learning (MR-MTL), a simple personalization framework, is a strong baseline for cross-silo FL: under stronger privacy requirements, silos are incentivized to federate more with each other to mitigate DP noise, resulting in consistent improvements relative to standard baseline methods. We provide an empirical study of competing methods as well as a theoretical characterization of MR-MTL for mean estimation, highlighting the interplay between privacy and cross-silo data heterogeneity. Our work serves to establish baselines for private cross-silo FL as well as identify key directions of future work in this area.

1.1 Introduction

Recent advances in machine learning often rely on large, centralized datasets [123, 36, 100], but curating such data may not always be viable, particularly when the data contains private information and must remain siloed across clients (e.g. mobile devices or hospitals). Recently, federated learning (FL) [108, 81] has emerged as a paradigm for learning from such distributed data, but it has been shown that its data minimization principle alone may not provide adequate privacy protection for participants [149, 154]. To obtain formal privacy guarantees, there has thus been extensive work applying *differential privacy* (DP) [43, 44] to various parts of the FL pipeline (e.g. [55, 110, 67, 79, 5, 97, 71, 122, 57]).

Existing approaches for differentially private FL are typically designed for *client-level* DP in that they protect the federated clients, such as mobile devices ("user-level"), tasks in multi-task learning ("task-level"), or data silos like institutions ("silo-level"), and DP is achieved by clipping and noising the client model updates. While client-level DP is considered a strong privacy notion as all data of a single client is protected, it may not be suitable for *cross-silo* FL, where there are fewer clients but each hold many data subjects that require protection. For example, when hospitals/banks/schools wish to federate patient/customer/student records, it is the people owning those records rather than the participating silos that should be protected. In fact, laws and regulations may mandate such participation in FL be disclosed publicly [142], compromising the privacy of the federating clients.

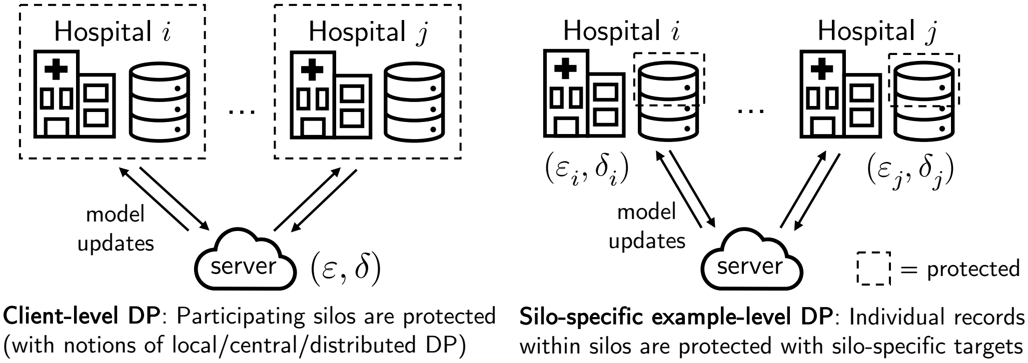
Hospital i
Hospital j
model updates
server
(ε, δ)
Client-level DP: Participating silos are protected (with notions of local/central/distributed DP)
Hospital i
Hospital j
$(\varepsilon_i, \delta_i)$
$(\varepsilon_j, \delta_j)$
model updates
server
= protected
Silo-specific example-level DP: Individual records within silos are protected with silo-specific targets

Figure 1.1: **Client-level** DP vs **silo-specific example-level** DP.

We instead consider a more natural model of *silo-specific example-level privacy* (Fig. 1.1, with variants appearing in [67, 101, 164, 82]): the k-th silo may set its own $(\varepsilon_k, \delta_k)$ example-level DP target for any learning algorithm with respect to its local dataset. With this formulation in mind, we then reconsider the impact of privacy, heterogeneity, and personalization in cross-silo FL. In particular, we explore existing baselines for FL (mostly developed in cross-device settings) across private cross-silo benchmarks, and we find that the simple baseline of mean-regularized MTL (MR-MTL) has many advantages for this setting relative to other more common (and possibly more complex) methods. We then further analyze the performance of MR-MTL under varying levels of heterogeneity and privacy, both in theory and practice. In addition to establishing baselines for cross-silo FL, we also identify interesting future directions in this area (Section 1.7 and Appendix F).

This chapter is summarized as the following:

- We consider the notion of *silo-specific example-level differential privacy* (DP) as a more realistic privacy model for *cross-silo federated learning* (FL). We analyze its implications on existing FL algorithms and, in particular, how it interfaces with data heterogeneity across silos.

- We empirically show that mean-regularized multi-task learning (MR-MTL), a simple form of model personalization, is a remarkably strong baseline under silo-specific example-level DP. Core to its effectiveness is its ability to (roughly) interpolate on the model personalization spectrum between local training and FedAvg with minimal privacy overhead.
- We provide a theoretical analysis of MR-MTL under mean estimation and characterize how MR-MTL navigates the tension between privacy and cross-silo data heterogeneity.
- Finally, we examine the complications of deploying an optimal MR-MTL instance that stem from the privacy cost of *hyperparameter tuning*. Our reasoning also applies to other personalization methods whose advantage over local training and/or FedAvg hinges on selecting the best hyperparameter(s). This raises important questions around the practicality of leveraging personalization to balance the emerging tradeoffs under silo-specific example-level DP.

The code corresponding to this chapter and the related publication [98] can be found at https://github.com/kenziyuliu/private-cross-silo-fl.

1.2 Background and Related Work

Federated Learning (FL) [108, 89, 81] is a distributed learning paradigm with an emphasis on data minimization. Typically:

- a *central server* sends a model to participating *clients*;
- the clients train that model using their own local data and send back updated models; and
- the server aggregates the updates (e.g., via averaging)

and the cycle repeats. Companies like Apple and Google have deployed FL to train models for applications such as predictive keyboards[1], text election[2], and speaker verification[3] in networks of user devices.

A basic instantiation of FL is FedAvg [108], where clients are stateless and the server performs a simple (weighted) average. *Cross-device* FL refers to settings with many clients each with limited data, bandwidth, availability, etc. (e.g. mobile devices). In contrast, *cross-silo* FL typically involves less clients (e.g. banks, schools, hospitals) but each with more resources. Two distinguishing characteristics of cross-silo FL relevant to our work are that (1) silos may have sufficient data to fit a reasonable local model *without* FL, and (2) each data point in a silo tends to map to a data subject (a person) requiring privacy protection.

Although the high-level federated learning workflow described above can help to mitigate systemic privacy risks, past work [81, 154, 149] suggests that FL's data minimization principle alone isn't sufficient for data privacy, as the client models and updates can still reveal sensitive information.

Differential Privacy (DP). Differential privacy is often used in conjunction with FL to ensure that an algorithm (provably) does not leak the privacy of its inputs. DP provides both a formal guarantee and an effective empirical mitigation to attacks like membership inference [28] and data

[1] https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

[2] https://ai.googleblog.com/2023/03/distributed-differential-privacy-for.html

[3] https://machinelearning.apple.com/research/improving-on-device-speaker

poisoning [140]. In a nutshell, DP is a statistical notion of privacy where we add randomness to a query on a "dataset" to create quantifiable uncertainty about whether any one "data point" has contributed to the query output. DP is typically measured by two scalars (ε, δ)—the smaller, the more private.

Definition 1.2.1 (Differential Privacy [43, 44]). *A randomized algorithm $M : \mathcal{X}^n \to \mathcal{Y}$, where $\mathcal{X}^n$ is the set of datasets with n samples and $\mathcal{Y}$ is the set of outputs, is (ε, δ)-DP if for any subset $S \subseteq \mathcal{Y}$ and any neighboring x, x' differing in only one sample (by replacement), we have*

$$\Pr[M(x) \in S] \leq \exp(\varepsilon) \cdot \Pr[M(x') \in S] + \delta. \tag{1.1}$$

To apply DP to a dataset query, one commonly used method is the Gaussian mechanism [44], which involves bounding the contribution (ℓ^2-norm) of each sample in the dataset followed by adding Gaussian noise proportional to that bound onto the aggregate. To apply DP in FL, one needs to define the “dataset” to protect; typically, as in *client-level* DP, this is the set of FL participants and thus the model updates from each participant in every round should be bounded and noised.

In learning settings, we need to repeatedly query a dataset and the privacy guarantee composes. We use DP-SGD [133, 13, 1] for ensuring example-level DP for model training, and we use Rényi DP [114] and zCDP [22] for tight privacy composition, as discussed below. In certain FL algorithms, clients also perform additional work such as cluster selection [106, 56] that incurs privacy overhead with respect to its local dataset that must be accounted for independently from DP-SGD.

Rényi Differential Privacy (RDP). We make use of a relaxation of different privacy known as Rényi Differential Privacy [114] for tight privacy accounting.

Definition 1.2.2 (Rényi Differential Privacy (RDP) [114]). *A randomized algorithm $M : \mathcal{X}^n \to \mathcal{Y}$ is (α, ε)-RDP with order $\alpha > 1$ if for any adjacent datasets $x, x' \in \mathcal{X}^n$,*

$$D_\alpha(M(x) \| M(x')) \leq \varepsilon, \tag{1.2}$$

where $D_\alpha(P\|Q)$ is the Rényi divergence[4] *between distributions P and Q:*

$$D_\alpha(P\|Q) \triangleq \frac{1}{\alpha - 1} \log \mathop{\mathbb{E}}_{x \sim P}\left[\left(\frac{P(x)}{Q(x)}\right)^{\alpha-1}\right]. \tag{1.3}$$

Under Rényi DP, the privacy composition is simple: if every step of an algorithm satisfies (α, ε)-RDP, then over T steps the algorithm satisfies $(\alpha, T\varepsilon)$-RDP. The following lemma from [22, 25] provides a conversion from RDP to standard (ε, δ)-DP guarantees.

Lemma 1.2.3 (Conversion from Rényi DP to approximate DP [22, 25]). *If a mechanism M satisfies $(\alpha, \varepsilon(\alpha))$-RDP, then for any $\delta > 0$, it also satisfies $(\varepsilon(\delta), \delta)$-DP where*

$$\varepsilon(\delta) = \inf_{\alpha > 1} \varepsilon(\alpha) + \frac{1}{\alpha - 1} \log\left(\frac{1}{\alpha\delta}\right) + \log\left(1 - \frac{1}{\alpha}\right). \tag{1.4}$$

Zero-Concentrated Differential Privacy (zCDP). A closely related privacy notion is zero-concentrated DP (zCDP [22]), where ρ-zCDP is equivalent to satisfying $(\alpha, \rho\alpha)$-Rényi DP simultaneously for all orders α. Thus, algorithms that satisfy zCDP guarantees are compatible with standard

[4] The Rényi divergence at $\alpha = 1$ is defined as $D_1(P\|Q) \triangleq \mathop{\mathbb{E}}_{x \sim P}\left[\log\left(\frac{P(x)}{Q(x)}\right)\right] = \lim_{\alpha \to 1} D_\alpha(P\|Q)$, which is also the KL divergence.

RDP accounting routines implemented in open-source libraries (e.g. TensorFlow Privacy [109]). In our work, we make use of zCDP and a related result for the Exponential Mechanism [127] for tight privacy composition when implementing private cluster selection for IFCA [56, 106] (discussed in more details in Section 1.5 and Appendix B.3).

Exponential Mechanism for Private Selection. The Exponential Mechanism (EM) is a standard algorithm for making private selection from a set of candidates based on their scores [111]. Specifically, there is a dataset $x \in \mathcal{X}^n$ requiring DP protection, and a scoring function $s : \mathcal{X}^n \times [G] \to \mathbb{R}$ that evaluates a set of candidates $g \in [G]$. We want to pick the candidate with the highest score (i.e. $\text{argmax}_{g \in [G]} s(x, g)$) subject to $(\varepsilon, 0)$-DP for neighboring datasets x, x'. The mechanism M is defined by setting the probability of choosing any $g \in [G]$ as

$$\Pr[M(x) = g] = \frac{\exp\left(\frac{\varepsilon}{2\Delta} \cdot s(x, g)\right)}{\sum_{g' \in [G]} \exp\left(\frac{\varepsilon}{2\Delta} \cdot s(x, g')\right)}, \tag{1.5}$$

where Δ is the sensitivity of the scoring function. EM also satisfies $\frac{1}{8}\varepsilon^2$-zCDP [127] and thus $(\alpha, \frac{\alpha}{8}\varepsilon^2)$-RDP for all α. A variant of EM is the Permute-and-Flip mechanism [107].

The Exponential Mechanism can be implemented as “Report Noisy Max” with Gumbel noise: we can add independent noises drawn from the Gumbel distribution with scale $\frac{2\Delta}{\varepsilon}$ to the candidate scores $s(x, g)$ for all $g \in [G]$ and simply report the max noisy score. If the score function is a loss metric (where we want the minimum instead of the maximum), we can similarly implement “Report Noisy Min” by subtracting the Gumbel noises from the scores and report the minimum.

Privacy Budgeting for Machine Learning. A typical accounting workflow, as used in our experiments, thus involves (1) composing the RDP guarantees of all private operations in the algorithm and (2) trying a list of α values that give the lowest ε for a target δ when converting back to (ε, δ)-DP that captures the overall privacy cost. For SGD training, we also use existing results on privacy amplification via subsampling [115, 1]: if a gradient step is (ε, δ) w.r.t. the dataset *without* amplification and the gradient is computed with a minibatch (assumed to be a random sample) of batch size $b = q/n$ where q is the sampling ratio and n is the size of the dataset, then the privacy of the gradient step is amplified to $(O(q\varepsilon), \delta)$-DP. In a silo-specific example-level DP setup, the size of the dataset n_k at each silo thus determines the extent of the amplification, and thus even if silos target for the same (ε, δ) example-level DP, they may end up adding different amounts of noise when running DP-SGD. For this reason our experiments (in the coming Section 1.4) primarily focus on having the same privacy target for all silos.

Heterogeneous Differential Privacy. A related privacy notion is *heterogeneous DP* [6, 78], where each item within a dataset to be protected by DP may opt for a different (ε, δ) target. Our setting primarily focuses on different (ε, δ) values for disjoint datasets, and all items within a specific dataset share the same DP target.

Personalized Federated Learning. Model personalization is a key technique for improving utility under data heterogeneity across silos.[5] Past work has examined the roles of local adaptation [147, 157, 32], multi-task learning [132, 129], clustering [56, 35, 106, 129], public data [163, 106], meta learning [76, 88, 47], or other forms of model mixtures [93, 91, 106, 63, 38, 3]. Notably, many methods leverage extra computation to some extent (e.g. extra iterations [93, 91, 32] or cluster selection [56, 106]), which will result in privacy overhead under silo-specific example-level DP as discussed in the following section. Of particular interest is the family of mean-regularized multi-task

[5] Note that “personalization” refers to customizing models for each *client* in FL rather than a specific person.

learning (MR-MTL) methods [46, 136, 63, 64] (see Algorithm 1 for a typical instantiation). We find that MR-MTL, while extremely simple, is a strong baseline for private cross-silo FL.

1.3 Privacy Granularity for Cross-Silo Federated Learning

To date, the prevalent privacy model for federated learning has been to protect the participating clients, i.e. client-level DP. For cross-silo FL, however, several factors render client-level DP less appropriate. First, cross-silo FL often involves a small number of clients and it can be utility-wise more costly to attain the same privacy targets. For example, privacy amplification via sampling [1, 115] may not apply on the client level since all silos typically participate in every round. Second, many existing methods focus on enforcing client-level DP in a non-local model and thus defines a shared privacy target for all participants, but in real-world cross-silo settings, participants under different jurisdictions (e.g. states) may have varying privacy requirements and thus opt for different privacy-utility tradeoffs. Third, while silo-level protection implies example-level protection, it may be too stringent in practice as silos often have large local datasets. These unique properties for private cross-silo learning motivate us to consider *silo-specific example-level* DP as an alternative privacy model (Fig. 1.1):

Definition 1.3.1 (Silo-specific example-level DP)**.** *A cross-silo FL algorithm with K clients (silos) satisfy $\{(\varepsilon_k, \delta_k)\}_{k\in[K]}$-"silo-specific example-level DP" if the local (personalized) model M_k of every silo $k \in [K]$ satisfies $(\varepsilon_k, \delta_k)$-DP w.r.t. the silo's local dataset of training examples.*

Characteristics of silo-specific example-level privacy. Importantly, silo-specific example-level DP is defined over the *disjoint* datasets of the individual silos, rather than the combined dataset of all silos.[6] To instantiate this setup in FL, each silo can simply run DP-SGD [133, 13, 1] with a noise scale calibrated to gradually spend its privacy budget over T training rounds, and return the noisy model update at each round. This privacy notion has several important implications on the dynamics of FL:

1. **Silos incur privacy costs with queries to their data, but *not* with participation in FL.** This follows from DP's robustness to post-processing: the silos' model updates in each round already satisfy their own example-level DP targets, and participation by itself does not involve extra dataset queries (e.g. DP-SGD steps). In contrast, local training without communication can be kept noise-free under client-level DP, but participation in FL requires privatization. Two immediate consequences of the above are that
   (a) local training and FedAvg now have *identical* privacy costs, and
   (b) *local finetuning* for model personalization may no longer work as expected (Fig. 1.2).
2. **Less reliance on a trusted server.** As a corollary of the above, all model updates of silo k satisfy (at least) $(\varepsilon_k, \delta_k)$-DP against external adversaries, including all other silos and the orchestrating server [44, 164]. In contrast, client-level DP under a non-local model necessitates some trust on the server, even for distributed DP methods (e.g. [45, 34, 79, 5, 33, 31]).
3. **Tradeoff emerges between costs from privacy and heterogeneity.** As privacy-perserving noises are added independently on each silo, they are reflected in silos' model updates and can thus be smoothed out when the model updates are aggregated (e.g. via FedAvg), leading to a

[6] A record in such a combined dataset is at most $(\max_i \varepsilon_i,\ \max_i \delta_i)$-DP [112, 156, 97]. Moreover, if multiple records (either within a silo or across silos) map to the same person, then it is more intricate to protect the person rather than their records. Here we focus on the case where each entity has at most one record across the combined dataset (e.g. students attending exactly one school). See Appendix A for discussions.

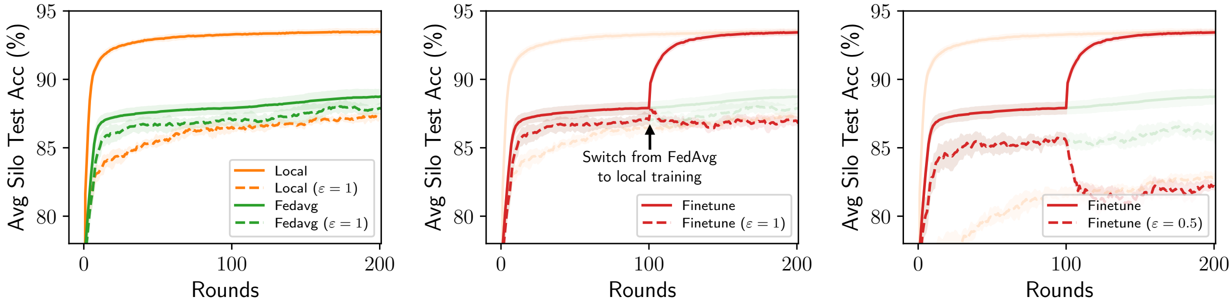

Figure 1.2: **Two notable phenomena under *silo-specific example-level DP***: (1) FedAvg can serve to cancel out per-silo DP noise and thus outperform local training even when the latter works better without privacy (left); (2) Local finetuning [157, 32] (FedAvg followed by local training) may not improve utility as expected, as the effect of noise reduction is removed when finetuning begins (mid & right). Results report mean test acc ± std on the Vehicle dataset over 5 seeds. For simplicity, all silos budgets for the same labeled ε with $\delta = 10^{-7}$. Transparent curves refer to local/FedAvg runs with the same ε labeled for finetuning (compare left & mid).

smaller utility drop due to DP for the shared model. On the other hand, federation also means that the shared model may suffer from *client heterogeneity* (non-iid data across silos). This intuition is observed in Fig. 1.2: while local training may outperform FedAvg without privacy, the opposite can be true when privacy is added.

The first and last in the above are of particular interest because they suggest that *model personalization* can play a key and distinct role in our privacy setting. Specifically, local training (no FL participation) and FedAvg (full FL participation) can be viewed as two ends of a *personalization spectrum* with identical privacy costs; if local training minimizes the effect of data heterogeneity but enjoys no DP noise reduction, and contrarily for FedAvg, it is then natural to ask whether there exist personalization methods that lie in between and achieve better utility, and, if so, what methods would work best.

Related privacy settings. Past work on differentially private FL has concentrated on client-level DP and cross-device FL (e.g. [55, 110, 71, 57, 80, 79, 7]), and the application of DP in cross-silo FL, particularly where each silo defines its own DP targets for records of its own dataset, is relatively underexplored. Privacy notions closest to ours first appeared in [141, 88, 67, 164, 101, 82, 97]. In [141], each client adds its own one-shot noise onto its outgoing update, but in learning scenarios this provides client-level protection. The works of [88, 101, 164, 67, 97] study analogous privacy notions, though they respectively focus on boosting utility [88], analyzing statistical rates [101], adapting FL to f-DP [164, 40], applying security primitives [67], and learning a better global model; the aspects of heterogeneity, DP noise reduction, the personalization spectrum, and their interplay (e.g. Figs. 1.2 and 1.5) were unexplored. The work of [82] also considers a similar privacy notion, but the authors study a disparate trust assumption where DP noise is *not* added to local training/finetuning such that the final personalized models lack privacy guarantees. We note that the trust model most suitable for private cross-silo FL may be application-specific; here, we focus on the setting where the outputs of the FL procedure (the personalized models) must remain differentially private, as, for example, they may be served to non-curator users within the silo through an API.

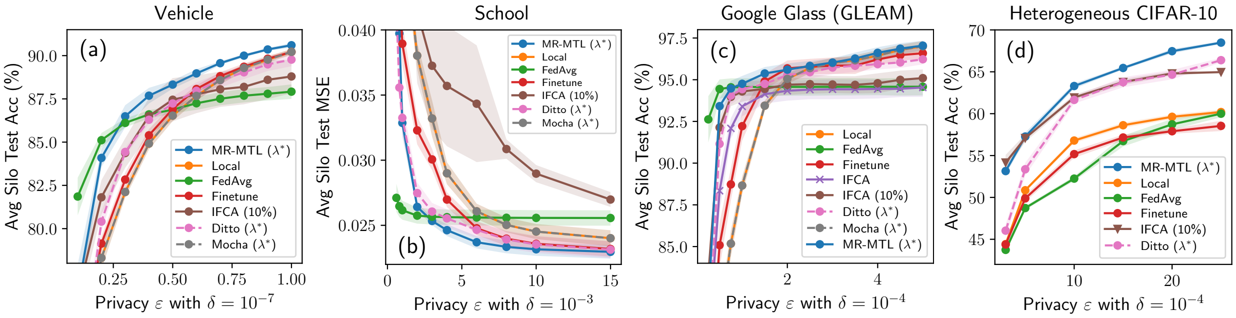

Figure 1.3: **Privacy-utility tradeoffs** (privacy budgets ε vs. test metrics, mean ± std w/ 5 seeds) for various personalization methods on **Vehicle**, **School**, **GLEAM**, and **Heterogeneous CIFAR-10** datasets respectively. For simplicity, every silo targets for the same (ε, δ) under silo-specific example-level DP. λ^* denotes a tuned regularization strength where applicable. "Local" denotes local training (clients train and keep their own models). "IFCA (10%)" denotes forming clusters for only first 10% of training rounds due to privacy overhead (Fig. 1.4).

1.4 Baselines for Private Cross-Silo Federated Learning

With the characteristics from Section 1.3 in mind, we now explore various methods on cross-silo benchmarks. We defer additional details as well as results on more settings and datasets to the appendix.

Datasets. We consider four cross-silo datasets that span regression/classification and convex/non-convex tasks: Vehicle [42], School [58], Google Glass (GLEAM) [121], and CIFAR-10 [85]. The first three datasets have real-world cross-silo characteristics: Vehicle contains measurements of road segments for classifying the type of passing vehicles, School contains student attributes for predicting exam scores, and GLEAM contains motion tracking data to classify wearers' activities. CIFAR-10 has heterogeneous client splits following [136, 130]. See Appendix B.1 for more details and datasets.

Benchmark methods. We consider several representative methods in the personalized FL literature beyond local training and FedAvg [108]: local finetuning [147, 157, 32] (a simple but strong baseline for model personalization), Ditto [91] (state-of-the-art personalization method), Mocha [132] (personalization with task relationship learning), IFCA/HypCluster [56, 106] (state-of-the-art hard clustering method for client models), and the mean-regularized multi-task learning (MR-MTL) methods [46, 136, 63, 64] (which we analyze in Section 1.5). For fair comparison under silo-specific example-level DP, we align all benchmark methods on the total privacy budget by first restricting the total number of iterations over the local datasets and then account for any privacy overheads (in the form of necessary extra steps [91] or cluster selection for IFCA/HypCluster [56, 106]). Importantly, many other personalization methods can either be reduced to one of the above under convex settings (e.g. [76, 93]) or are unsuitable due to large privacy overheads (e.g. large factor of extra steps for [47]).

Training setup. For all methods, we use minibatch DP-SGD in each silo to satisfy silo-specific example-level privacy; while certain methods may have more efficient solvers (e.g. dual form for [132]), we want compatibility with DP-SGD as well as privacy amplification via sampling on the example level for tight accounting. For all experiments, silos train for 1 local epoch in every round (except for [91] which runs ≥ 2 epochs). Hyperparameter tuning is done via grid search for all methods.

Importantly, when comparing the benchmark methods, we do not account for the privacy cost of hyperparameter tuning in order to focus on their inherent privacy-utility tradeoff; we revisit this issue in Section 1.7. For simplicity, we use the same privacy budget for all silos, i.e. $(\varepsilon_i, \delta_i) = (\varepsilon_j, \delta_j)$ for all $i, j \in [K]$; note that this is not a restrictive assumption since the effect of having varying DP noise scales from different budgets can be attained by varying local dataset sizes.

Results. Fig. 1.3 shows the privacy-utility tradeoffs across four datasets. We observe that MR-MTL consistently outperforms a suite of baseline methods, and that it performs at least as good as local training and FedAvg (endpoints of the personalization spectrum), except at high-privacy regimes (possibly different for each dataset). In particular, there exists a range of ε values where MR-MTL can give significantly better utility over local training and FedAvg *under the same privacy budgets* ($\varepsilon \approx 0.5, 6, 1.5$ for Fig. 1.3 (a, b, c) respectively); this is our key regime of interest.

Effects of silo-specific example-level privacy. In Fig. 1.2 we saw that local finetuning may not improve utility as expected, motivating us to reconsider the roles of federation and personalization (Section 1.5 below). In Fig. 1.4, we consider the implication of silo-specific example-level DP from the effects of privacy overhead due to additional dataset queries: (1) If IFCA [56, 106] performs cluster selection at every round (default behavior), then the extra privacy cost can be prohibitive; (2) Despite its similarity to MR-MTL, Ditto [91]'s privacy overhead makes it less competitive (see also Fig. 1.5).

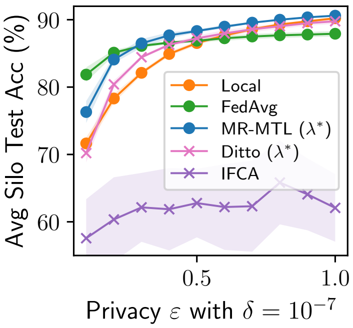

Figure 1.4: The privacy overhead of cluster selection (IFCA) and extra iterations (Ditto) compared to local, FedAvg, and MR-MTL under silo-specific example-level DP.

1.5 On the Effectiveness of Mean-Regularized MTL for Private Cross-Silo FL

Following the observations in Section 1.4, we now examine the desirable properties of a good algorithm under silo-specific example-level privacy and understand why MR-MTL may be an attractive candidate.

Federation as noise reduction. A key message from Section 1.3 and Fig. 1.2 is that the utility cost from DP can be significantly smaller for FedAvg compared to local training even when the latter spends an identical privacy budget. This implies that FedAvg may have inherent benefits for DP noise reduction, despite the noise are added in the *gradient* space rather than the parameter space (as in client-level DP). Consider a simple setting of DP gradient descent: the update rule $w_k^{(t+1)} = w_k^{(t)} - \frac{\eta}{n_k}\left(z^{(t)} + \sum_{i=1}^{n_k} g_{k,i}^{(t)}\right)$ for silo k recursively expands to $w_k^{(T)} = w_k^{(0)} - \frac{\eta}{n_k}\sum_{t=0}^{T-1} z^{(t)} - \frac{\eta}{n_k}\sum_{t=0}^{T-1}\sum_{i=1}^{n_k} g_{k,i}^{(t)}$ over T steps, where $g_{k,i}^{(t)}$ is the clipped gradient of the i-th local example at step t (with norm bound c) out of a total of n_k examples, and $z^{(t)} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$ is the Gaussian noise added to the gradient sum at step t that targets for an overall privacy budget of $(\varepsilon_k, \delta_k)$ over T steps with $\sigma^2 = O\left(c^2 T \ln(1/\delta_k)/\varepsilon_k^2\right)$ [1]. The cumulative noise term

$$Z^{(T)} \triangleq -\frac{\eta}{n_k}\sum_t z^{(t)} \sim \mathcal{N}\left(0,\ O\left(\frac{\eta^2 c^2 T^2 \ln(1/\delta_k)}{n_k^2 \varepsilon_k^2}\right) \cdot \mathbf{I}\right) \tag{1.6}$$

indeed implies that each silo's model update has an independent Gaussian random walk component [128, 8, 148] whose variance can be reduced by averaging with other silos' updates, as in FedAvg.[7] A similar reasoning applies to SGD cases since the additive DP noises are i.i.d. across the minibatches.

Model personalization for privacy-heterogeneity cost tradeoff. A major downside of FedAvg is that it may underperform simple local training due to data heterogeneity (e.g. [157] and Fig. 1.2), particularly given that clients in cross-silo FL often have sufficient data to fit reasonable local models. This suggests an emerging role for model personalization on top of its benefits in terms of utility [147], robustness [157], or fairness [91] under heterogeneity: our privacy model allows local training and FedAvg to be viewed as two endpoints of a *personalization spectrum* that respectively mitigate the utility costs of heterogeneity and privacy noise with identical privacy budgets (recall Section 1.3); this means that personalization methods could be viewed as interpolating between these endpoints and that various personalization methods essentially do so in different ways. However, our empirical observations motivate the following key properties of a good personalization algorithm:

1. **Noise reduction**: The effect of noise reduction is present throughout training so that the utility costs from DP can be consistently mitigated. Local finetuning is a counter-example (Fig. 1.2).
2. **Minimal privacy overhead**: There are little to no additional local dataset queries to prevent extra noise for DP-SGD under a fixed privacy budget. In effect, such privacy overhead can shift the utility tradeoff curve downwards, and Ditto [91] may be viewed as a counter-example (Fig. 1.4).
3. **Smooth interpolation along the personalization spectrum**: The interpolation between local training and FedAvg should be fine-grained (if not continuous) such that an optimal tradeoff should be attainable. Clustering [56, 106] may be viewed as a counter-example when there are no clear heterogeneity strucutre across clients.

These properties are rather restrictive and they render many promising algorithms less attractive. For example, model mixture [93, 16, 38] and local adaptation [157, 32] methods can incur linear overhead in dataset iterations, and so can multi-task learning [91, 132, 129] methods that benefit from additional training. Clustering methods [56, 35, 106, 129] can also incur overhead with cluster selection [56, 106], distillation [35], or training restarts [129], and they discretize the personalization spectrum in a way that depends on external parameters (e.g., the number of clients, clusters, or top-down partitions).

The case for mean-regularization. These considerations point to mean-regularized multi-task learning (MR-MTL) as one of the simplest yet particularly suitable forms of personalization. MR-MTL has manifested in various forms in the literature [46, 160, 136, 63, 32, 71] with the key idea that a personalized model w_k for each silo k should be close to the mean of all personalized models $\bar{w}$ via a regularization penalty $\lambda/2\|w_k - \bar{w}\|_2^2$ (see Algorithm 1 for a typical instantiation). The hyperparameter λ serves as a smooth knob between local training and FedAvg, with $\lambda = 0$ recovering local training and a larger λ forces the personalized models w_k to be closer to each other ("federate more"). However, it is an imperfect knob as $\lambda \to \infty$ may *not* recover FedAvg under a typical

[7] The work of [80] examines the benefits of adding negatively correlated (instead of independent) noises z_t across time steps. While this is a potential orthogonal extension to our use of local DP-SGD *within* each silo, it may not be directly applicable to our main focus of reducing noise variance *across* silos, since for each silo k to satisfy its own $(\varepsilon_k, \delta_k)$ requirement, it must add noise independent to other silos.

optimization setup as the regularization term may dominate the gradient step

$$w_k^{(t+1)} = w_k^{(t)} - \eta \left(g_t + \lambda \left(w_k^{(t)} - \bar{w}^{(t)} \right) \right)$$

where g_t is the noisy clipped gradient, and MR-MTL may thus underperform FedAvg in high-privacy regimes that necessitate a large λ to mitigate DP noise (Fig. 1.3).

MR-MTL has the attractive properties that: (1) noise reduction is achieved throughout training via a soft constraint that personalized models are close to an averaged model; (2) for fixed λ it has *zero* additional privacy cost compared to local training/FedAvg as it does not involve extra dataset queries; and (3) λ provides a smooth interpolation along the personalization spectrum. Moreover, compared to other regularization-based MTL methods, it adds only one hyperparameter λ (cf. [74, 165, 59]); this has important practical implications as will be discussed in Section 1.7. It also has fast convergence [161] and easily extends to deep learning with good empirical performance in the primal [136, 71] (cf. [12, 132, 99]). It is also sufficently extensible to handle structured heterogeneity (discussed below). We argue through the following empirical analyses that these properties make MR-MTL a strong baseline under silo-specific example-level DP.

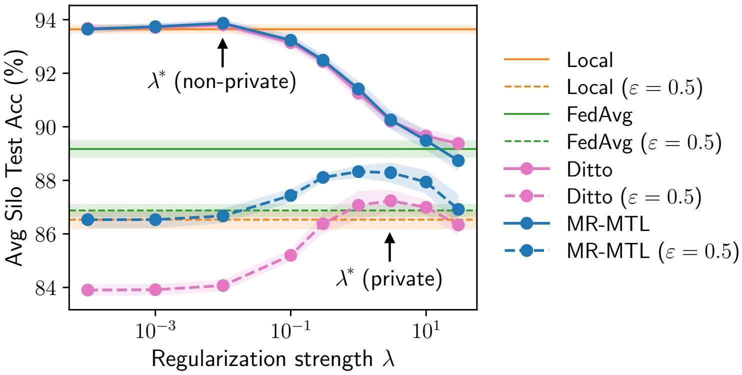

Figure 1.5: **Test acc ± std of MR-MTL with varying** λ (corresponding to $\varepsilon = 0.5$ in Fig. 1.3 (a)). Optimal points (λ^*) exist where it outperforms both ends of the spectrum under the *same* privacy. Ditto [91] gives a similar interpolation but has strictly worse privacy-utility tradeoffs due to its privacy overhead. See Appendix G.2 for extensions of this figure to other privacy settings and datasets.

Navigating the emerging privacy-heterogeneity cost tradeoff. In Fig. 1.5 we study the effect of the regularization strength λ on the model utility directly. There are several notable observations: (1) In both private and non-private settings, λ serves to roughly interpolate between local training and FedAvg. (2) The utility at the best λ^* may outperform both endpoints. This is significant for the private setting since MR-MTL achieves an *identical* privacy guarantee as the endpoints. (3) Moreover, the *advantage* of MR-MTL over the best of the endpoints are also larger under privacy. (4) The value of λ^* also increases under privacy, indicating that the personalized silo models are closer to each other (i.e. silos are encouraged to "federate more") for noise reduction. We will characterize these behaviors in Section 1.6. We also consider Ditto [91] in Fig. 1.5, a state-of-the-art personalization method that resembles MR-MTL and exhibits similar behaviors, to illustrate the effect of privacy overhead from its extra local training iterations.

MR-MTL under structured heterogeneity. We further study (1) the extensibility of MR-MTL as a strong baseline method to handle clustering structures of silo data distributions and (2) its flexibility to handle varying heterogeneity levels, by manually introducing two layers of heterogeneity

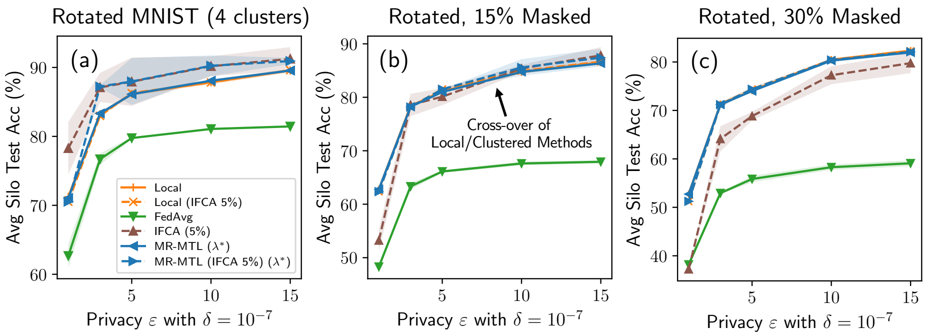

Figure 1.6: **Test acc ± std on Rotated & Masked MNIST**. (IFCA 5%) denotes warm-starting the method by running IFCA [56, 106] for first 5% of rounds followed by running the method within the cluster structures.

to the MNIST dataset [86]. The first layer is 4-way rotations: train/test images are evenly split into 4 groups of 10 silos, with each group applying $\{0°, 90°, 180°, 270°\}$ of rotation to their images. The second layer is *silo-specific* masking: each silo generates and applies its unique random mask of 2×2 white patches to its images, with varying masking probability. Together, the 1st layer creates 4 well-defined silo clusters, and the 2nd layer (gradually) adds *intra-cluster* heterogeneity. Importantly, our goal is not to contrive a utility advantage of MR-MTL (in fact, the added heterogeneity is disadvantageous to mean-regularization), but to examine its extensibility and flexibility as a strong baseline method to match the best methods by construction. Under the 1st layer of heterogeneity, clustered FL methods [56, 106] should be optimal if the correct clusters are formed since there is no intra-cluster heterogeneity; with increasing silo-specific heterogeneity in the 2nd layer, local training should be increasingly more attractive. See Appendix B.1 for more details on the setup and examples of images.

We propose a simple heuristic to precondition or "warm-start" MR-MTL with a small number of training rounds by running private clustering (with IFCA [56, 106]) followed by mean-regularized training *within each formed cluster* (see Appendix C for details). We find that this simple heuristic, with convergence properties carried forward from its components [56, 46, 161], enables MR-MTL to excel at all levels of heterogeneity: in Fig. 1.6 (a), the preconditioning allows MR-MTL to match IFCA (optimal by construction) while local training (full personalization) does not benefit from the same preconditioning; in Fig. 1.6 (b, c, d), MR-MTL remains optimal across different levels of silo-specific heterogeneity (the 2nd layer) while the gains from warm-start gradually drop. We argue that extensibility and flexibility are good properties that make MR-MTL a strong baseline, as heterogeneity in practical settings is likely less adversarial than what we presented.

1.6 Analysis

In this section we provide a theoretical analysis of MR-MTL under mean estimation as a simplified proxy for (single-round) FL using a Bayesian framework extending on [91]. We provide expressions for the Bayes optimal estimator MR-MTL (λ^*) and describe how MR-MTL behaves with varying λ in relation to the personalization spectrum to characterize our observations from Fig. 1.5. Proofs and extensions are deferred to Appendix D.

Setup. We start with a total of K silos where the k-th silo holds n training samples $X_k \triangleq \{x_{k,i} \in$

$\mathbb{R}\}_{i\in[n]}$, each normally distributed around a hidden center w_k with variance σ^2; i.e. $x_{k,i} = w_k + z_{k,i}$ with $z_{k,i} \sim \mathcal{N}(0, \sigma^2)$. To quantify heterogeneity, the silo centers $\{w_k\}_{k\in[K]}$ are also normally distributed around some unknown fixed meta-center θ with variance τ^2; i.e. $w_k = \theta + z$ with $z \sim \mathcal{N}(0, \tau^2)$. A large τ means that the silo centers are distant from each other and thus their local objectives are heterogeneous, and contrarily for a small τ. Our goal is for each silo k to compute a example-level private estimate of w_k that minimizes the *generalization* error (i.e. on unseen points from the same local distribution). Each silo targets (ε, δ) example-level DP and runs the Gaussian mechanism with noise scale $\sigma_{\text{DP}} = c\sqrt{2\ln(1.25/\delta)}/\varepsilon$ and clipping bound c.[8] Under this setting, the MR-MTL objective for the k-th silo is

$$h_k(w) = \tilde{F}_k(w) + \frac{\lambda}{2}\|w - \bar{w}\|_2^2. \tag{1.7}$$

Here, $\tilde{F}_k(w) \triangleq \frac{1}{2}(w - \frac{1}{n}(\xi_k + \sum_{i=1}^n x_{k,i} \cdot \min(1, c/\|x_{k,i}\|_2)))^2$ is the local objective to privately estimate the mean of the local data points with privacy noise $\xi_k \sim \mathcal{N}(0, \sigma_{\text{DP}}^2)$. Since the data are (sub-)Gaussian, we assume one can choose c such that no clipping error is introduced w.h.p., so $\hat{w}_k \triangleq \operatorname{argmin} \tilde{F}_k(w) = \frac{1}{n}\left(\xi_k + \sum_i x_{k,i}\right)$ is the best local estimator. $\bar{w} = \frac{1}{K}\sum_k \hat{w}_k$ is the average estimator across silos, which is the same as the FedAvg estimator under mean estimation. We also consider the *external* average local estimators for silo k, defined as $\hat{w}_{\setminus k} \triangleq \frac{1}{K-1}\sum_{j\neq k}\hat{w}_j$. The following lemma gives the best MR-MTL estimator $\hat{w}_k(\lambda)$ as a function of λ.

Lemma 1.6.1. *Let $\lambda \geq 0$ and $\alpha = \frac{K+\lambda}{(1+\lambda)K} \in (1/K, 1]$. The minimizer of $h_k(w)$ is given by*

$$\hat{w}_k(\lambda) = \alpha \cdot \hat{w}_k + (1 - \alpha) \cdot \hat{w}_{\setminus k}. \tag{1.8}$$

Note that the best λ is always 0 for *training* error (i.e. estimating the empirical mean of the local data $\{x_{k,i}\}$); our hope is that with some $\lambda > 0$, $\hat{w}_k(\lambda)$ yields a better *generalization* error.

We now present the main takeaways. At a high level, the basis of our analysis relies on expressing the true center w_k in terms of $\hat{w}_k$ and $\hat{w}_{\setminus k}$ conditioned on the local datasets $\{X_k\}_{k\in[K]}$. Let

$$\sigma_{\text{loc}}^2 \triangleq \frac{\sigma^2}{n} + \frac{\sigma_{\text{DP}}^2}{n^2} \tag{1.9}$$

denote the "local variance" of $\hat{w}_k$ around w_k due to both data sampling and privacy noise.

Behavior of MR-MTL at optimal λ^*. We first derive the following lemma using Lemma 11 of [105].

Lemma 1.6.2. *Given $\hat{w}_k$, $\hat{w}_{\setminus k}$, and $\{X_k\}_{k\in[K]}$, we can express $w_k = \mu_k + \zeta_k$, where $\zeta_k \sim \mathcal{N}(0, \sigma_w^2)$,*

$$\sigma_w^2 \triangleq \left(\frac{1}{\sigma_{\text{loc}}^2} + \frac{K-1}{K\tau^2 + \sigma_{\text{loc}}^2}\right)^{-1} \quad \textit{and} \quad \mu_k \triangleq \sigma_w^2\left(\frac{1}{\sigma_{\text{loc}}^2} \cdot \hat{w}_k + \frac{K-1}{K\tau^2 + \sigma_{\text{loc}}^2} \cdot \hat{w}_{\setminus k}\right). \tag{1.10}$$

Lemma 1.6.2 expresses the unobserved true silo centers w_k in terms of the (private) empirical estimators $\hat{w}_k$ and $\hat{w}_{\setminus k}$. This expression requires conditioning on the datasets X_k as they form the Markov blankets of $\hat{w}_k$. Combining Lemma 1.6.1 and Lemma 1.6.2 gives the optimal λ.

Theorem 1.6.3 (Optimal MR-MTL estimate)**.** *The best λ^* for the generalization error is given by*

$$\lambda^* = \operatorname*{argmin}_{\lambda} \mathbb{E}\left[(w_k - \hat{w}_k(\lambda))^2 \mid \hat{w}_k, \hat{w}_{\setminus k}, \{X_k\}_{k\in[K]}\right] = \frac{1}{n\tau^2}\left(\sigma^2 + \frac{\sigma_{\text{DP}}^2}{n}\right). \tag{1.11}$$

[8] For simplicity, we start with the same n, σ, σ_{DP} for all silos and extend to silo-specific values in Appendix D.

Theorem 1.6.3 suggests that there indeed exists an optimal point $\hat{w}(\lambda^*)$ on the personalization spectrum. Moreover, λ^* grows smoothly with stronger privacy ($\sigma_{\mathrm{DP}}^2 \to \infty$) to encourage silos to "federate more" with others. This was empirically observed in Fig. 1.5. We now characterize the utility of $\hat{w}(\lambda^*)$.

Corollary 1.6.4 (Optimal error with $\hat{w}(\lambda^*)$)**.** *The MSE of the optimal estimator $\hat{w}(\lambda^*)$ is given by*

$$\mathcal{E}^* \triangleq \mathbb{E}\left[(w_k - \hat{w}_k(\lambda^*))^2 \mid \hat{w}_k, \hat{w}_{\setminus k}, \{X_k\}_{k\in[K]}\right] = \sigma_w^2 = \frac{\sigma_{\mathrm{loc}}^2(\sigma_{\mathrm{loc}}^2 + K\tau^2)}{K(\sigma_{\mathrm{loc}}^2 + \tau^2)}. \tag{1.12}$$

Note also that $\hat{w}(\lambda^*)$ is the MMSE estimator of w_k. Using Corollary 1.6.4, we can compare $\hat{w}_k(\lambda^*)$ against the endpoints of the personalization spectrum (local training and FedAvg) with the following propositions.

Proposition 1.6.5 (Optimal error gap to local training)**.** *Let $\mathcal{E}_{\mathrm{loc}} \triangleq \mathbb{E}\left[(w_k - \hat{w}_k)^2 \mid X_k\right] = \sigma_{\mathrm{loc}}^2$ be the error of the local estimate. Then, compared to the optimal estimator $\hat{w}(\lambda^*)$ (Corollary 1.6.4), the local estimator incurs an additional error of*

$$\Delta_{\mathrm{loc}} \triangleq \mathcal{E}_{\mathrm{loc}} - \mathcal{E}^* = \left(1 - \frac{1}{K}\right) \cdot \frac{\sigma_{\mathrm{loc}}^4}{\sigma_{\mathrm{loc}}^2 + \tau^2}. \tag{1.13}$$

Proposition 1.6.6 (Optimal error gap to FedAvg)**.** *Let $\mathcal{E}_{\mathrm{fed}} \triangleq \mathbb{E}\left[(w_k - \bar{w})^2 \mid \{X_k\}_{k\in[K]}\right]$ be the error under FedAvg. Then, compared to the optimal estimator $\hat{w}(\lambda^*)$ (Corollary 1.6.4), the FedAvg estimator incurs an additional error of*

$$\Delta_{\mathrm{fed}} \triangleq \mathcal{E}_{\mathrm{fed}} - \mathcal{E}^* = \left(1 - \frac{1}{K}\right) \cdot \frac{\tau^4}{\sigma_{\mathrm{loc}}^2 + \tau^2}. \tag{1.14}$$

Together, Propositions 1.6.5 and 1.6.6 suggest that the effects of stronger privacy ($\sigma_{\mathrm{DP}}^2, \sigma_{\mathrm{loc}}^2 \to \infty$) on how MR-MTL compares against the personalization endpoints are mixed, with the benefit of MR-MTL *increasing* against local training and *diminishing* against FedAvg. They also suggest that MR-MTL has an optimal utility advantage over both the endpoints when $\sigma_{\mathrm{loc}}^2 \approx \tau^2$ and local training performs on par with FedAvg, and the utility "bump" under privacy observed in Fig. 1.5 can be viewed as a result of this balance. It is worth noting that since the data variance σ^2 and heterogeneity τ^2 are often fixed in practice, the freedom for silos to vary their privacy targets (ε and σ_{DP}^2) makes the utility advantage of MR-MTL more flexible compared to non-private settings.

Behavior of MR-MTL as a function of λ. The above captures how MR-MTL behaves at its optimum, but in Fig. 1.5 we also observed that MR-MTL has the desirable property that the utility cost from DP *shrinks smoothly* with larger λ (Section 1.5). Lemma 1.6.7 and Theorem 1.6.8 below provides a characterization.

Lemma 1.6.7 (Error of $\hat{w}_k(\lambda)$)**.** *Let $\mathcal{E}(\lambda) \triangleq \mathbb{E}\left[(w_k - \hat{w}_k(\lambda))^2 \mid \hat{w}_k, \hat{w}_{\setminus k}, \{X_k\}_{k\in[K]}\right]$ be the error of MR-MTL as a function of λ. Then,*

$$\mathcal{E}(\lambda) = \left(1 - \frac{1}{K}\right) \frac{\sigma_{\mathrm{loc}}^2 + \lambda^2\tau^2}{(\lambda+1)^2} + \frac{\sigma_{\mathrm{loc}}^2}{K}. \tag{1.15}$$

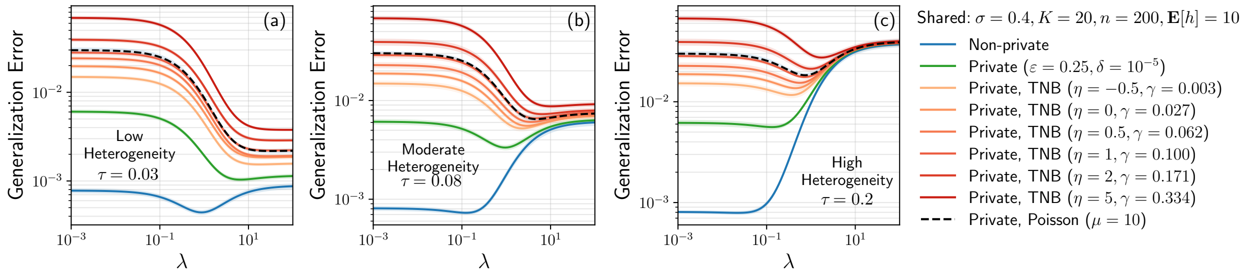

Figure 1.7: **Privacy costs of tuning λ on mean estimation** (setup follows Section 1.6). Labels "**Private**" and "**Non-Private**" denote the errors of varying λ with and without silo-specific example-level DP (the privacy cost of tuning λ is *not* included). "**Private, TNB/Poisson**" [117] denotes the same errors but accounts for the privacy cost of trying on average $\mathbf{E}[h] = 10$ values of λ, with h sampled from the truncated negative binomial distribution with parameters η, γ / the Poisson distribution with parameter μ to arrive at the same $\mathbf{E}[h]$. To interpret, observe that the lowest points of "**Private, TNB/Poisson**" may be still higher than one of the endpoints of "**Private**".

Using Lemma 1.6.7 we can now characterize how λ affects the utility cost from DP (recall from Figs. 1.2 and 1.5 that federation helps with noise reduction). As a side note, Lemma 1.6.7 also suggests that MR-MTL's utility as a function of λ would have a quasi-concave shape, as was empirically observed in Fig. 1.5. This could potentially help make heuristic or automated search over λ easier.

Theorem 1.6.8 (Private utility gap)**.** *Let $\hat{w}_k(\lambda)$ and $\hat{w}_k^{\mathrm{DP}}(\lambda)$ be the non-private and private estimate of w_k with $\sigma_{\mathrm{loc}}^2 \leftarrow \sigma^2/n$ and $\sigma_{\mathrm{loc}}^2 \leftarrow \sigma^2/n + \sigma_{\mathrm{DP}}^2/n^2$, respectively. Let $\mathcal{E}(\lambda)$ and $\mathcal{E}^{\mathrm{DP}}(\lambda)$ be the error of $\hat{w}_k(\lambda)$ and $\hat{w}_k^{\mathrm{DP}}(\lambda)$ respectively as in Lemma 1.6.7. Let $\Delta_{\mathrm{DP}}(\lambda) \triangleq \mathcal{E}^{\mathrm{DP}}(\lambda) - \mathcal{E}(\lambda)$ be the utility cost due to privacy as a function of λ. Then,*

$$\Delta_{\mathrm{DP}}(\lambda) = \left(1 - \frac{1}{K}\right) \frac{1}{(\lambda+1)^2} \frac{\sigma_{\mathrm{DP}}^2}{n^2} + \frac{\sigma_{\mathrm{DP}}^2}{Kn^2}. \tag{1.16}$$

Theorem 1.6.8 suggests that with a larger λ, the utility cost from privacy can be smoothly mitigated by up to a factor of K, matching the empirical observation in Fig. 1.5.

1.7 Discussions

In previous sections, we empirically and theoretically studied the benefits of the best personalization hyperparameter λ^* for MR-MTL, but it remains open as to how such λ^* may be obtained. In this section, we take an honest look at the complications of deploying MR-MTL through the lens of the *privacy cost* of finding λ^*. There are in general several approaches: (1) a non-adaptive search (e.g. grid/random search [15]); (2) an adaptive search (e.g. grad student descent); or (3) an online estimation during training (e.g. [143, 9, 118]). Here, we focus on approach (1) since it is generic to all personalization methods and is a setting for which we have the best privacy accounting tools [95, 117] to our knowledge. We defer technical details and further discussions to Appendix E. Note that while we focus on MR-MTL, our reasoning in principle extends to all personalization methods whose advantage depends on having the best hyperparameter(s).

Recall that for a typical tuning procedure, a baseline algorithm M is executed h times with different hyperparameters and the best result is recorded. The work of [95, 117] shows that, with a constant

h, there exists M that satisfies $(\varepsilon, \delta = 0)$-DP where the output of tuning is not $(\tilde{\varepsilon}, 0)$-DP for any $\tilde{\varepsilon} < h\varepsilon$, with analogous negative results for Rényi DP (thus also for $\delta > 0$). This implies that naive tuning (as done in practice) can incur a prohibitive privacy overhead and obliterates the utility advantage of MR-MTL (λ^*) over local training/FedAvg. Instead, by making h *random*, we can make $\tilde{\varepsilon}$ *constant* w.r.t. h or at most $\tilde{\varepsilon} \leq O(\log \mathbf{E}[h])$ [95, 117]. However, using the simplified setting of mean estimation (Section 1.6), we find that even with this improved randomized protocol, there exist scenarios (Fig. 1.7) where the realistic cost of trying a moderate $\mathbf{E}[h] = 10$ values of λ may significantly diminish, or even *outweigh*, the utility advantage of λ^* over local training and FedAvg—that is, we might be better off by *not privately tuning* λ *at all.*

The above has several important implications. On the negative side, it suggests that the true efficacy of MR-MTL can be smaller in practice. Moreover, it raises the broader open question of whether the emerging privacy-heterogeneity cost tradeoff is best balanced by model personalization, as many existing methods including MR-MTL inherently require at least one hyperparameter to specify "how much to personalize" for general utility improvements over local training and FedAvg. Alternatively, the hyperparameter(s) may be estimated *during* training (approach (3) in the first paragraph), though such procedures may not be general and/or scalable and may need to be tailored to the specific personalization method. On the positive side, it is unclear whether the choice of λ can meaningfully leak privacy in practice. MR-MTL may also be viewed favorably as a strong baseline since it only needs one hyperparameter to attain its benefits, while other existing methods that require more tuning will incur even larger privacy costs from hyperparameter tuning.

1.8 Concluding Remarks

In this chapter, we studied the application of differential privacy in cross-silo FL. We examine *silo-specific example-level DP* as a more appropriate privacy notion for cross-silo FL, and we point out several meaningful ways in which it differs from client-level DP commonly studied under the cross-device setting, particularly when analyzing tensions between privacy, utility, and heterogeneity. We explore and establish baselines under this privacy setting and identify desirable properties for a personalization method for balancing an emerging tradeoff between utility costs from privacy and heterogeneity. We then analyze a simple, promising method (MR-MTL) and discuss key open questions for the area at large. Some future directions include (1) extending the privacy model to cases where data subjects have multiple records across silos, (2) extending our theoretical characterization to deep learning cases or performing a large-scale empirical study, and (3) developing auto-tuning algorithms for personalization hyperparameters with minimal privacy overhead.

Chapter 2

The US/UK Privacy-Enhancing Technologies (PETs) Prize Challenge

In this chapther, we describe our recent entry to the US/UK PETs Prize Challenge,[1] a competition held by the US/UK governments to facilitate the development of privacy-preserving federated learning solutions that provide end-to-end privacy and security protections while harnessing the potential of AI for overcoming significant global challenges. The challenge is split into two tracks: Financial Crime Prevention and Pandemic Forecasting and Response.

Our team ("puffle") at Carnegie Mellon University[2] developed a solution for the Pandemic Forecasting and Response track[3], expanding on the ideas presented in Chapter 1. Our approach incorporated additional components such as graph processing, privacy accounting, and addressing data imbalance. Our solution was awarded 1st place on the US side of the competition, receiving a prize of $100,000 USD, along with an additional $20,000 USD for open-sourcing our solution.[4]

See also coverage by the Summit for Democracy 2023,[5] White House,[6] UK Government,[7]DrivenData,[8] NSF,[9] NIST,[10] and CMU News.[11]

[1] Challenge main page: https://petsprizechallenges.com/

[2] CMU team profile: https://drivendata.co/blog/federated-learning-pets-prize-winners-phases-2-3#puffle

[3] Technical brief of the pandemic forecasting problem: https://iuk.ktn-uk.org/wp-content/uploads/2022/08/PETs-Prize-Challenges_-Public-Health-Technical-Brief-1.pdf

[4] Open-sourcerelease:https://github.com/kenziyuliu/pets-challenge

[5] https://www.youtube.com/watch?v=8nRs3VArnco&t=12544s

[6] https://www.whitehouse.gov/ostp/news-updates/2023/03/31/us-uk-annouce-winners-innovation-pets-democratic-values/

[7] https://www.gov.uk/government/news/at-summit-for-democracy-the-united-kingdom-and-the-united-states-announce-winners-of-challenge-to-drive-innovation-in-privacy-enhancing-technologies

[8] https://drivendata.co/blog/federated-learning-pets-prize-winners-phases-2-3

[9] https://beta.nsf.gov/news/us-uk-winners-prize-challenge-in-privacy-enhancing-tech

[10] https://www.nist.gov/itl/applied-cybersecurity/privacy-engineering/collaboration-space/challenges

[11] https://www.cs.cmu.edu/news/2023/pets-prize

2.1 Introduction and Problem Setup

Federated learning (FL) considers learning from siloed data in a decentralized fashion, thereby mitigating privacy risks. FL shows promise for many real-world scenarios, including the *pandemic forecasting* application explored in the US/UK PETs Prize Challenge. The federated pandemic forecasting problem considers the following:

> *Given a person's attributes, locations, activities, infection history, and the contact network, what is their likelihood of infection in the next t days? Given a person's demographic attributes (e.g. age, household size), locations, activities, infection history, and the contact network, what is the likelihood of infection in the next $t_{\text{pred}} = 7$ days? Can we make predictions while protecting the privacy of the individuals? Moreover, what if the data are siloed across administrative regions?*

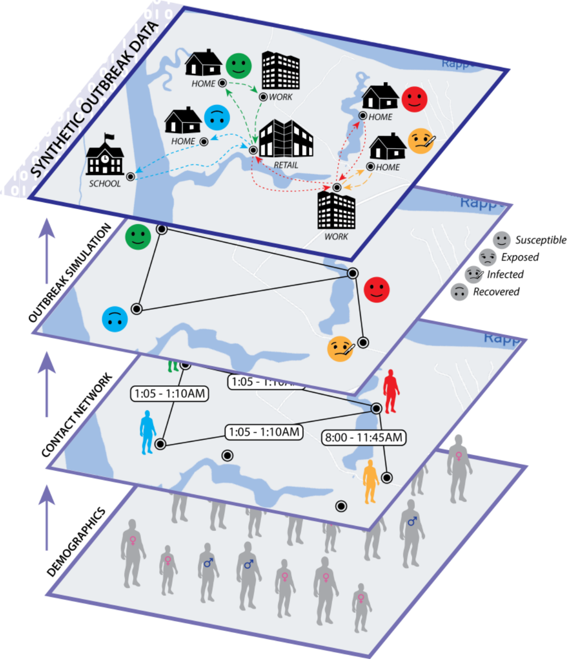

Figure 2.1: Illustration of the pandemic forecasting problem at the US/UK PETs challenge. Image sourced from https://prepare-vo.org/synthetic-pandemic-outbreaks.

While simple to state, the problem above contains many complex and interrelated elements that merit careful examination. First, the pandemic outbreak problem follows a discrete-time SIR model[12] (**S**usceptible → **I**nfectious → **R**ecovered) and we begin with a subset of the population infected. Subsequently,

- each person goes about their usual daily activities, such as working or shopping, and gets into contact with others either directly or indirectly (e.g. at a mall)—this forms a **contact graph** where individuals are nodes and direct contacts are edges;
- each person may get infected with different risk levels depending on a myriad of factors—their age, the nature and duration of their contact(s), their node centrality, their activities, etc.; and
- such infection can also be **asymptomatic**—the individual can appear in the S state while being secretly infectious.

The challenge dataset[13] models a synthetic pandemic outbreak in Virginia and contains roughly **7.7 million** nodes (persons) and **186 million** edges (contacts) with health states over 63 days, with the social contacts repeating every day. Thus, the actual contact graph is fairly large but also quite sparse.

There are a few extra factors that make this problem challenging:

- **Extreme data imbalance**: there are less than 5% of people who are ever in the I or R state and roughly 0.3% of people became infected in the final week.

[12] <https://people.wku.edu/lily.popova.zhuhadar/

[13] https://prepare-vo.org/synthetic-pandemic-outbreaks

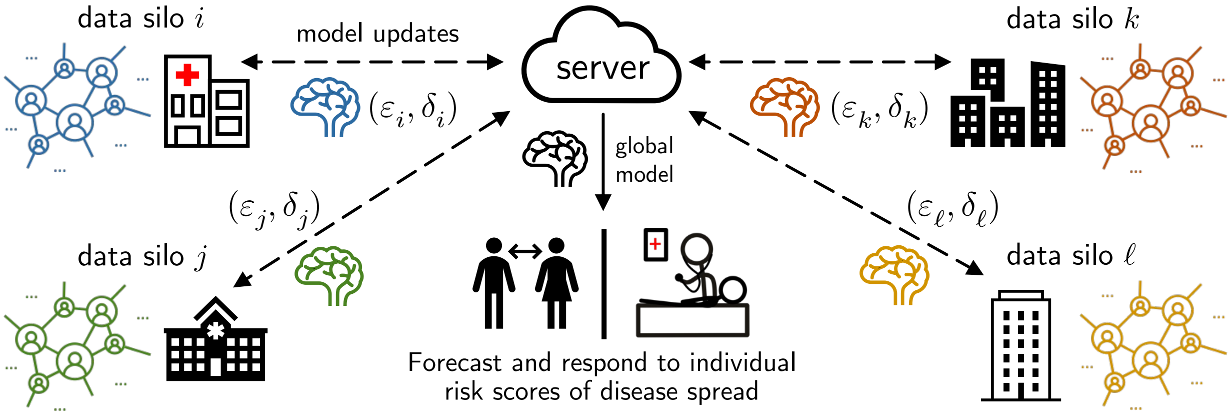
data silo i
model updates
server
$(\varepsilon_i, \delta_i)$
data silo k
$(\varepsilon_k, \delta_k)$
global model
$(\varepsilon_j, \delta_j)$
$(\varepsilon_\ell, \delta_\ell)$
data silo j
data silo ℓ
Forecast and respond to individual risk scores of disease spread

Figure 2.2: Illustration of the federated pandemic forecasting problem.

- **The data are siloed**: the actual contact graph will be cut along administrative boundaries, e.g., by grouped FIPS codes/counties.[14] Each silo only sees a local *subgraph* but people may still travel and make contacts across multiple regions. There are up to 10 silos (clients) and the population size can vary by more than an order of magnitude in the official evaluation (up to 2.6M nodes for a single silo in the official evaluation).

- **The temporal nature of the problem is subtle**: we are given the first $t_{\text{train}} = 56$ days of each person's health states (S/I/R) and asked to predict the risk of infection for any individual any time in the subsequent $t_{\text{pred}} = 7$ days. *What is a training example in this case? How should we perform temporal partitioning? How does this relate to privacy accounting?*

- **Graphs generally complicate DP**: when applying DP to machine learning, we are used to the settings where we can clearly define the *privacy granularity* and how it relates to an actual individual (e.g. tabular data or medical images of patients). This is tricky when it comes to graphs: people can make different number of contacts, each of different nature, and their influence may propagate to their neighbors, neighbors' neighbors, and so on. On a high-level (and as specified by the scope of sensitive data of the competition[15]), what we care about is known as **node-level DP**—the model output is "roughly" the same if we add/remove/replace a node, *along with its edges*. See Section 2.4 for more details.

Having good pandemic forecasting model(s) is useful both for the individual to take precautionary measures and for public health agencies to make informed decisions when allocating resources. This problem is challenging not only because the spread of a disease can be a highly complex process, but also because the relevant data often include sensitive information that cannot be aggregated for centralized analytics. Even fitting a local model to each data silo can pose privacy risks as the model may be susceptible to membership inference [131], attribute inference [52], data poisoning [140], memorization [27], or may otherwise leak data in unexpected ways [117]. Indeed, given only black-box access to a non-private local model for a small administrative region (e.g., a rural town), a de-identified test person's social contacts (and thus their identity and activities) may be easily inferred if the model predicts a high risk of infection for that person. In general, releasing models that are not properly trained with defense mechanisms can pose a serious threat to the

[14] https://en.wikipedia.org/wiki/FIPS_county_code

[15] https://www.drivendata.org/competitions/98/nist-federated-learning-1/page/525/#scope-of-sensitive-data

privacy of individuals.

Despite FL's aim to mitigate privacy risks, data localization alone may not provide adequate privacy protection for participants (e.g. [149, 154, 19]), let alone the aforementioned issues inherent to vanilla model training. There has thus been extensive efforts to augment FL's privacy protection capabitilies with technologies such as homomorphic encrpytion [54, 158], secure multi-party computation (particularly secure aggregation [20, 14]), and differential privacy [43, 1, 110].

Of particular interest to our solution is *differential privacy* (DP), which is widely viewed as the gold standard of privacy protection because it has *provable*, *quantifiable*, and *worst-case* guarantees. DP presents a particularly strong promise in the case of pandemic forecasting because it is also "*future-proof*": with a proper privacy and trust model, it can help ML algorithms capture the underlying trend of the disease spread while making them robust to a range of statistical attacks—both known and unknown—that may leverage emerging information from the *evolving* contact network.

In our solution, we consider the application of *silo-specific sample-level DP*, which we studied in detail in Chapter 1 (recall Fig. 1.1 and Section 1.3). This privacy formulation has several important advantages and implications on the interplay between privacy, utility, and data heterogeneity, as we will detail in the following sections. Our insights lead us to propose a practical framework for training *personalized*, silo-specific models centered around this DP model that can maximize accuracy while offering strong privacy guarantees.

2.2 Threat Model

We begin by discussing the objective and capabilities of an adversary and enumerate the sources of vulnerabilities throughout the FL lifecycle, their potential exploits, and their mitigation under our proposed privacy solution. We also examine important caveats and any novel privacy risks that may emerge as part of our solution.

2.2.1 Adversary's Capabilities and Objective

Capabilities. During deployment, we assume that all trained models will be released publicly, including both the federated model on the server and the personalized models trained on each client (admin region). The adversary has *white-box* access to the model parameters, and they can pass arbitrary inputs to the models and observe their outputs. The adversary may also have unrestricted computational power. In Section 2.2.2, we additionally consider the adversary's capabilities specific to each source of vulnerability *during training*, such as the ability to contaminate training data, control clients, control the central server, or eavesdrop on the communication channels.

Objective. The adversary's objective is to extract information of an *arbitrary* person from the publicly released models. Specifically, this may include but is not limited to: (1) identifying whether a particular person (Alice)'s data was used for training (i.e. membership inference), (2) inferring Alice's information (e.g. residence, social contacts) from the released models with partial knowledge of Alice's data (i.e. attribute inference), (3) reconstructing Alice's data from the released models (i.e. model inversion). We consider the strength of an attack to be correlated with its success probability, ease of execution, and the accuracy and completeness of the extracted information.

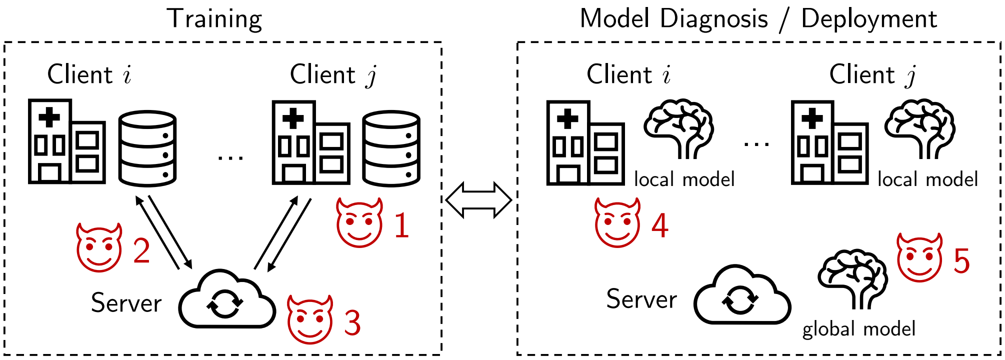

Figure 2.3: **Various sources of vulnerabilities.** During training, the adversary may have full or partial control over (1) the participating client, (2) the communication channel, and (3) the central orchestrating server. During model development/diagnosis or at deployment, the adversary may also have white-box access to (4) the local personalized model and (5) the global model. They may also see all model iterates and intermediate metrics.

2.2.2 Sources of Vulnerabilities, Corresponding Attacks, and Mitigations

Fig. 2.3 summarizes possible sources of vulnerabilities, where (1-3) depicts training-time vulnerabilities and (4-5) depicts other non-interactive and inference-time vulnerabilities. Recall from Section 2.1 that our core privacy tool is **silo-specific example-level** DP, and that this informally refers to each silo k enforcing a *example-level* $(\varepsilon_k, \delta_k)$-DP target locally during training (e.g. via DP-SGD [1]), with model iterates "staying more or less the same" with or without any particular local example. We now motivate this DP model and other design choices by discussing each of the vulnerabilities in a horizontal FL setup suited to pandemic forecasting, how our solution mitigates them, and assumptions on the adversary's capabilities. We focus on privacy risks in the following and defer discussions on other important properties (e.g. robustness) to Section 2.3.

(Source 1) Clients. *Within a client*, we make the minimal assumption that the *data curator* will faithfully execute a private training protocol (e.g. DP-SGD) without nefariously leaking the data directly to an adversary, but the curator may incorporate custom training objectives such as data augmentation or adversarial training as long as it adheres to its $(\varepsilon_k, \delta_k)$-DP budget. *Across clients*, malicious actors may inspect the sequence of model iterates from the server and attempt to extract sensitive information about other honest clients. However, all model iterates from honest clients satisfy example-level DP w.r.t. their own data and are thus robust to *arbitrary post-processing* by malicious clients. By the same token, honest clients' DP guarantees hold against active attacks such as **data poisoning** [139] (e.g. malicious clients train with (intentionally) erroneous S/I/R labels or contact edges in the infection data), **model poisoning** [48] (e.g. malicious clients scale up their updates to dominate the server aggregate), and **backdoor attacks** (e.g. [11, 135, 145, 152]). Note that such active attacks may be hard to detect (and thus bad for *utility*), as the anomaly may not manifest in model metrics (e.g. norm of the update); we discuss counter-measures in Section 2.3.

(Source 2) Communication channels. We assume an adversary may gain full (e.g. man-in-the-middle) or partial (e.g. eavesdropping) control of communication channels. In either case, the clients' example-level DP guarantees continue to hold, as their model updates are differentially private query releases of local examples (and thus resistant to post-processing). Importantly, disclosing model iterates does *not* affect clients' pre-allocated privacy budgets. Our DP guarantees also do *not* rely on security protocols (as opposed to, e.g., distributed (client-level) DP via SecAgg [79, 5]) and thus do not require key distribution (e.g. required by SecAgg [20]) over insecure channels.

(Source 3) Central server. An adversary may also gain control over the central server, making it semi-honest (attempting to extract sensitive information without deviating from the training protocol) or fully malicious (actively sabotaging the training process). Similarly, since the server only has access to client updates that already satisfy DP, even a fully malicious server (e.g. one that tampers with model updates [19] or simulates fake clients [41, 81]) would not be able to compromise the honest clients. In fact, if the malicious server holds no additional information apart from the federated model, it is equivalent to an adversary with full control over all client-server channels.

(Sources 1, 2, 3) Collusion. As a corollary of the above arguments, the example-level DP guarantee for an honest client k is not compromised even if all of the above parties are colluding.

(Sources 4, 5) Other non-interactive / inference-time attacks. An adversary may also perform "non-interactive" or inference-time attacks outside of training, such as during deployment with white-box access to the model(s), or during model development with access to model metrics. We argue that this form of attack—such as membership inference (e.g., extracting memorized data [27]), attribute inference (e.g. [53]), model inversion (e.g. [151]), and linkage/reconstruction (e.g. [39])—is again a form of post-processing on differentially private query releases of the training examples (e.g. model parameters or metrics), implying that the per-silo DP guarantees will still hold. In practice, there has been empirical evidence that (strong) DP effectively defends against such attacks (e.g. [75, 50, 159]). It is worth noting that while an adversary may leverage, e.g., model inversion to recover examples that "look like" actual training examples [159], they are in essence *statistical inference* that does *not* violate the privacy of an actual training example [23].

Overall, our solution emphasizes data privacy and adopts differential privacy as the primary privacy technology for worst-case protection. We expect that proper implementation of DP guarantees (see next section) can significantly reduce the above-identified privacy risks. In addition to DP, we also explore several heuristics to further strengthen privacy in practice, improve robustness, and mitigate the loss of utility; we defer additional discussions to Section 2.3.

2.2.3 Caveats, Emerging Privacy Risks, and Mitigations

The key argument for threat mitigation presented above is that (1) DP is *robust to arbitrary post-processing*, and that (2) such DP guarantees are budgeted and enforced *locally on each client.* These properties imply that silo-specific example-level DP enjoys similar benefits to the local (client-level) DP model, such as requiring minimal trust assumptions and robustness against insecure communication channels. However, there are also some risks to consider with our approach:

- **The definition of an "example" is intricate.** As the scope of sensitive data spans both the people and their contact edges, it can be intricate to define what an "example" means over a network, and whether such "example-level" privacy always translates to tangible protection for a person's data. In Section 2.3, we explore possible definitions of an "example" and formalize our DP model's protection w.r.t. a person, a contact edge, and a local neighborhood.
- **Silo-specific example-level DP provides guarantees over the *disjoint* client datasets, rather their concatenation.** That is, the global FL model (e.g. hosted by a public health agency running the server) satisfies at most $(\max_k \varepsilon_k, \max_k \delta_k)$-DP for any example in the combined

dataset across clients [112, 156, 97], where client k budgets for $(\varepsilon_k, \delta_k)$-DP for its local examples. Compliance issues may arise as the "global" privacy guarantee is dictated by the clients.

- **There may be multiple examples about the same person both *within* a silo and *across* silos.** By construction, silo-specific example-level DP aligns naturally with the setting where each "example" corresponds to a person requiring privacy protection (e.g. voting records across voting stations for a particular election). This may not always hold in practice, as multiple examples within a silo or across silos may contain information about the same person, which may thus enjoy less privacy than the reported DP guarantees. For pandemic forecasting, we may disregard the latter case since contacts are cut across admin regions (silos) and an individual only appears in one silo. In Section 2.4, we carefully apply graph-based privacy arguments to formalize privacy w.r.t. a person (and their contacts) based on our "example" definition in Section 2.3.
- **DP implementations can be error-prone.** Past work has identified a range of vulnerabilities when implementing DP mechanisms including errors from floating point representations [113, 72, 77, 61], precision attacks [62], timing attacks [77], and underestimation of the true sensitivity of a query [29]. These attacks can be practical and vulnerabilities can easily surface with a naive implementation. For the purposes of this challenge, we will stick to best practices for implementing the Gaussian mechanism [44] (e.g. using only vetted public libraries), but we note that despite ongoing efforts to mitigate them (e.g. with secure noise generation [69, 61], discrete noise mechanisms [4, 25, 79, 5], or multiple-precision floating-point representations [51, 138]), these vulnerabilities remain a tangible threat.

2.3 Methods

We now describe our technical approach to the PETs Prize Challenge. We first describe the framework of our solution (Section 2.3.1), and we then describe how we may formulate the machine learning task for the pandemic forecasting problem (Section 2.3.2).

2.3.1 Applying MR-MTL with Silo-Specific Example-Level Privacy

One simple and clean approach to the federated pandemic forecasting problem is to just operate on the **individual level** and view it as **(federated) binary classification**: if we could build a feature vector that summarizes what we need to know about an individual, then risk scores are simply the sigmoid probabilities of near term infection. Of course, the problem lies in what that feature vector (and the corresponding label) is—the following section provides further discussions. Regardless, observe that MR-MTL with silo-specific example-level privacy (from Chapter 1) is an attractive framework for this type of cross-silo learning problems:

- **Model personalization** is likely necessary as the silos are large and somewhat heterogeneous by construction (geographic regions are unlikely all to be similar to each other).
- **Implementation/Feasibility.** MR-MTL is remarkably easy to implement: (1) every client initializes a stateful, personalized model and the server initializes with the mean client model, and (2) in every round, the server broadcasts the mean model; clients simply perform (private) local training while ℓ_2-regularizing towards the mean; and the server updates the mean model using the model updates. The simplicity is often crucial for real-world applications.

- **Expected Privacy-Utility Tradeoff.** In Section 1.4, we performed empirical analyses on several public datasets, and we observed that MR-MTL consistently outperforms many state-of-the-art methods in terms of privacy-utility tradeoffs.

- **Robustness benefits.** In Section 2.2, we reviewed various potential attacks that may harm the *utility* of an honest client, such as model poisoning [48]. Similar to [91], an intrinsic robustness benefit to MR-MTL is that each client can "reduce its participation" in the federation *without affecting its DP guarantees* by setting a smaller λ than usual (or even reducing to local training with $\lambda = 0$) if there is evidence of malicious external actors. In turn, clients can avoid *catastrophic* utility failures from strong attacks, since its utility is lower bounded by local training ($\lambda = 0$), which may produce reasonably good results as silos often have large local datasets (unlike cross-device settings).

- **Efficiency and Scalability.** MR-MTL's simple construction implies minimal resource overhead and high efficiency and scalability. Compared to FedAvg-like algorithms (e.g. [90, 146]), it only additionally asks each silo to maintain its own model, thus linear overhead in memory (which is insignificant due to the small number of clients). Privacy protection is also scalable as it is enforced locally (thus no server workload), and each client only needs to pay a small overhead to implement DP with no cost at inference time. The resource requirement for privacy is also independent of the number of samples on each silo (it is instead a function of the learning algorithm).

- **Adaptability, Usability, and Explainability.** Our framework is highly adaptable to any underlying gradient-based learning algorithms (e.g. to run DP-SGD [1]) and types of data that support the soft proximal constraint, provided that a local "example" can be clearly defined. Moreover, it can be easily implemented and deployed due to its simplicity, the low resource overhead compared to FedAvg, and the ease of implementing DP using off-the-shelf vetted software libraries such as [60, 137]. It also preserves the explainability of the underlying ML algorithm and the applicability of many interpretability methods [116], as our privacy mechanism does not obfuscate individual model weights, updates, and predictions.

- **Extensibility.** One may also deploy heuristics on top of MR-MTL to (qualitatively) improve robustness and privacy-utility tradeoffs. For robustness, the (honest) server may inspect client updates and apply clipping, zero out coordinates with extreme values, or leverage existing robust aggregators (e.g. [17, 119]) to fend off malicious clients. For privacy, we may introduce extra heuristic that can help with the DP-utility tradeoff, such as manually removing extreme outliers (e.g. people in isolated locations) or merging categorical features (e.g. activity types).

2.3.2 Data and Task Formulation

Having established the overall framework for approaching the pandemic forecasting problem, we now describe how the individual features and the contact network can be converted into a tabular dataset for every silo k with access to a local subgraph with n_k individuals.

Recall that our task is to predict if a person is likely to get infected within $t_{\text{pred}} = 7$ days. We formulate this via a set of examples $\{X_k \in \mathbb{R}^{n_k \times d}, Y_k \in \{0,1\}^{n_k}\}$, where the features of each example $X_k^{(i)} \in \mathbb{R}^d$ describe the *local neighborhood* around a person i (see Fig. 2.4), with binary label $Y_k^{(i)}$ denoting if the person is infected in the next t_{pred} days.

Each example's features $X_k^{(i)}$ have two broad categories (depicted by Fig. 2.6):

1. Individual features that correspond to an individual only. For any person i, this includes demographic features like age, gender, and household size; activity features like whether this person has ever been to work, school, church, shopping, etc.; and the individual's *infection history* as concatenated one-hot vectors (which depends on how we create labels; see below).

2. Neighbor/Contact features that encode the social contacts, which include individual features of the people (nodes) in contact, and features of the contacts themselves (edges) such as location, duration, and activity type. For every person i, we encode contacts within *an ℓ-hop neighborhood* (i.e. individuals whose shortest paths to i has $\leq \ell$ edges), as illustrated in Fig. 2.4. Specifically, we *sample* a set of ℓ-hop neighbors and concatenate the edges' and neighbors' features with person i's features (Fig. 2.6), due to the large and exponentially growing neighborhood sizes and the need to equalize feature dimensions across nodes. In general, we may want $\ell > 1$ as higher-degree information can help (e.g. an individual's immediate contacts can be asymptotically infected), but it could also be more expensive to compute.

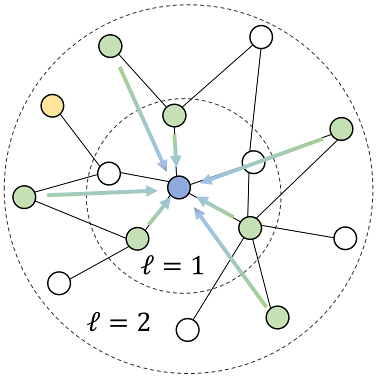

Figure 2.4: Illustration of iterative, ℓ-hop neighborhood feature aggregation. In this case, green nodes are sampled and the yellow node can't be sampled.

Moreover, we use **iterative sampling**, meaning that we first select a set of S_1 1-hop neighbors, and then sample S_2 2-hop neighbors from the neighbors of those 1-hop neighbors, and so on; this is such that the 1-hop edge features can be shared for the sampled 2-hop neighbors and we can keep the feature dimension d low. We experiment with different values of ℓ and $S_{j \leq \ell}$ as it poses a computation-utility tradeoff.

Importantly, we also do **deterministic neighborhood sampling**—the same person (node) always takes the same subset of neighbors—which makes the neighborhood sampling essentially a **graph pruning** step. This drastically reduces computation as the graph/neighborhoods can now be preprocessed and cached. There are also important implications for privacy, as we will discuss in Section 2.4. Fig. 2.5 provides an illustration.

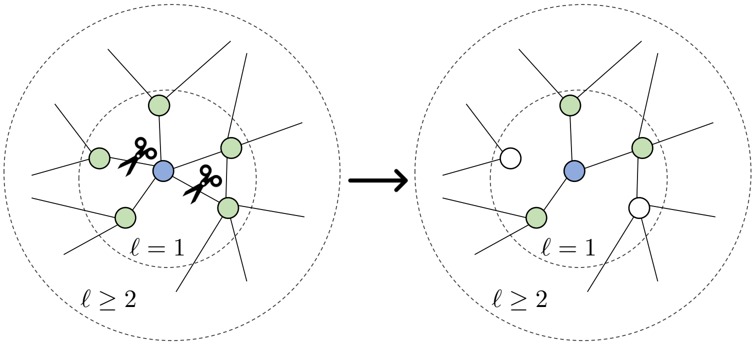

Figure 2.5: Illustration of deterministic neighborhood sampling, possibly implemented as graph pruning.

Putting everything together, Fig. 2.6 illustrates the final neighborhood feature vector that describes a person and their contacts that can then be used as inputs to a binary classifier.

Labels. To generate the labels $Y_k^{(i)}$, we deploy a *random infection window* strategy:

1. Pick a window size t_{window} (e.g., 21 days).

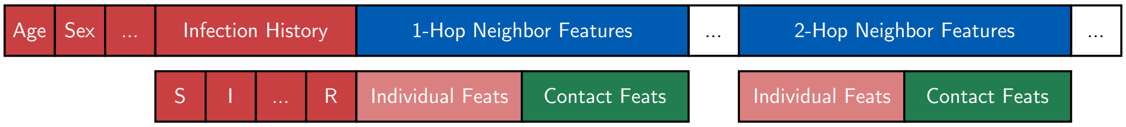

Figure 2.6: Illustration of the feature structure of one training sample, which concatenates the **individual features** and **neighbor features** (neighbors sampled from both 1-hop and 2-hop neighborhood). The feature vector of each neighbor includes the **neighbor's individual features** and the **contact features**.

2. Select a random t' with $t_{\text{window}} \le t' \le 56 - t_{\text{pred}}$.
3. Encode the S/I/R states as one-hot vectors in the past t_{window} days from t' into a training example; this includes both the sampled node (person) i, along with all of its sampled neighbors. Importantly, since the past infection histories of person i's neighbors can help inform the infection status of person i, we need to encode those histories *entirely* (as concatenated one-hot vectors, as opposed to using a binary indicator for person i).
4. The label for the training example is then whether person i transitions into infection in any of the next t_{pred} days from t'.

Fig. 2.7 provides an illustration. Since the contact graph repeats every day (recall Section 2.1), this implies that for any person i there are up to $56 - t_{\text{pred}} - t_{\text{window}} + 1 = 50 - t_{\text{window}}$ examples with the *same* personal features but *different* disease states and labels, thereby possibly different rows in the silo's tabular dataset from multiple random infection windows. In practice, we can pick one random window for every person, every epoch (so n_k nodes imply n_k examples), rather than actually generating all the possible windows at once.

Also, since we are interested in modeling *transitions into an infected status*, we only need to summarize the past infection window as a *binary* indicator of whether the person is in S state for all days; this is because a transition is only possible if the person remains susceptible.

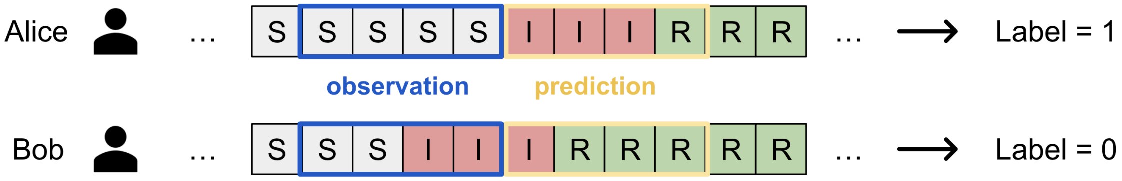

Figure 2.7: During training, we take a *random window* of infection states to use as features (the "observation" window) and labels (1 iff the person transitions into infection during the "prediction" window) every time we sample a person, and his/her neighboring nodes will use the same window for building the neighborhood feature vector. During testing, we deterministically take the latest days of the infection history.

This method of training set construction also implicitly assumes that a person's infection risk is *individual*: whether Bob gets infected depends only on his own activities and contacts in the past (say) 21-day window. This is definitely not perfect as it ignores *population-level* modeling (e.g. dense areas increase likelihood of infection), but it makes the ML problem very simple.

Inference. At test time, we build the same neighborhood features, except that the infection window is now deterministically the window right before the prediction period.

Connections to Graph Neural Networks (GNNs). Intriguingly, this way of training set construction makes the per-silo personalized models a simplified variant of a graph neural network:

there is a single step of *non-parameterized* neighborhood aggregation followed by feature transform and prediction (cf. SGC models [150]).

Remark on Generalizability. Importantly, observe that our framework fundamentally applies to all datasets or models where the concept of an "example", as well as its relationship to a data subject (e.g. person) requiring privacy protection, can be clearly defined. Such construction inherently provides high generalizability to various types of data, including tabular data, images, text, graphs, etc., and the various models that train on them. However, a caveat is that our solution only applies directly to horizontal FL setups, where the feature space is shared across clients.

2.4 Proof of Privacy

Having defined an "example" in Section 2.3.2, we now provide formal statements on our privacy guarantees. We first define DP and *silo-specific example-level* DP in the context of our method.

Definition 2.4.1 (Differential Privacy [43, 44])**.** *A model $M : \mathcal{X}^n \to \mathcal{Y}$, where $\mathcal{X}^n$ is the set of datasets with n data points and $\mathcal{Y}$ is the set of outputs, is (ε, δ)-DP if for any subset $Y \subseteq \mathcal{Y}$ and any neighboring x, x' differing in only one sample (by addition, removal, or replacement), we have*

$$\Pr[M(x) \in Y] \leq \exp(\varepsilon) \cdot \Pr[M(x') \in Y] + \delta.$$

Definition 2.4.2 (Silo-Specific Example-Level DP [98, 164, 82, 101, 96])**.** *A personalized FL system with K clients (silos) satisfy $\{(\varepsilon_k, \delta_k)\}_{k \in [K]}$-"silo-specific example-level DP" if the personalized model for every silo $k \in [K]$ satisfies $(\varepsilon_k, \delta_k)$-DP w.r.t. the silo's local training examples.*

In Section 2.3.2, we defined each example in silo (admin region) to be a feature vector for the (subsampled) *local contact neighborhood* of a person, thus example-level DP in this context is "***neighborhood-level DP***". However, the scope of sensitive data[16] and the adversary's objective (Section 2.2) implies that the personalized models should satisfy ***node-level*** (i.e. ***person-level***) DP on the siloed graphs:

Definition 2.4.3 (Node-Level Differential Privacy [18, 84, 124])**.** *A model $M : \mathcal{G} \to \mathcal{Y}$, where $\mathcal{G}$ is the space of undirected contact graphs and $\mathcal{Y}$ is the set of outputs, is (ε, δ)-node-level DP if for any subset $Y \subseteq \mathcal{Y}$ and any neighboring graphs G, G' differing in one node (by addition, removal, or replacement) and* ***all of its incident edges****, we have*

$$\Pr[M(G) \in Y] \leq \exp(\varepsilon) \cdot \Pr[M(G') \in Y] + \delta.$$

If a personalized model satisfies node-level DP, it intuitively means that an attacker cannot confidently identify, for any individual, their personal information (e.g. activities and health states) *and* their social contacts, which is precisely our privacy desideratum.

However, a key challenge is that when learning from neighborhood vectors (Fig. 2.6), it is difficult to obtain gradients w.r.t. an *individual node*, and thus privacy arguments of subsampled DP-SGD do not apply directly. To this end, the work of [37] proposed the following theorem for neighborhood aggregation methods (e.g. GNNs):

[16] https://www.drivendata.org/competitions/98/nist-federated-learning-1/page/525/#scope-of-sensitive-data

Theorem 2.4.4 (Node-level Rényi DP [37])**.** *For a (sub)graph with n nodes (hence n neighborhoods and n examples) and maximum neighborhood size S, training for T steps with batch size b with noise multiplier σ yields $(\alpha, \gamma T)$-node-level RDP, where*

$$\gamma = \frac{1}{\alpha - 1} \ln \mathbb{E}_\rho \left[\exp \left(\alpha(\alpha - 1) \frac{2\rho^2}{\sigma^2} \right) \right], \rho \sim \text{Hypergeometric}(n, S, b).$$

The idea is that the number of times ρ a certain node u appears in a batch of examples (neighborhoods) can be seen as following a **Hypergeometric distribution**— if all possible n neighborhoods are coins, and those S of them that contain u are heads, then ρ is the number of heads we'll get if we take b coins without replacement — and we can account for this when performing privacy analysis (computing Rényi divergence). Note that since the accounting depends on the graph size, it's easier to think about *replacement* DP instead of addition/removal.

Theorem 2.4.4 would be an analogy of *subsampling amplification* [115] on graphs, but it still implies that the DP guarantee depends (linearly) on the neighborhood size S (when batch size $b \gg S$) and this may be loose. Fig. 2.8 can provide some grounding; note that ε degrades drastically with larger maximum degree (neighborhood size if we take $\ell = 1$ hop only).

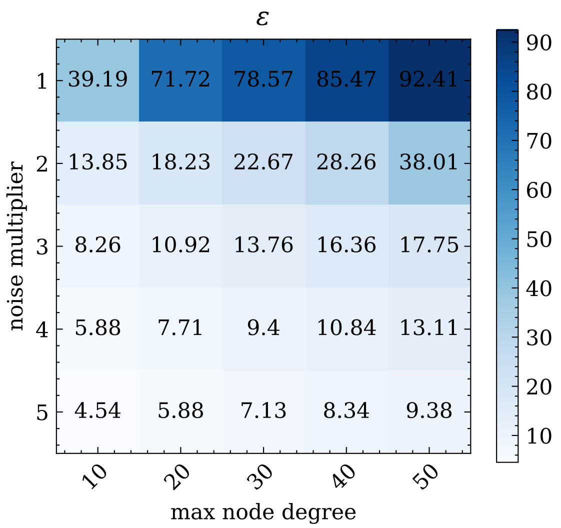

Figure 2.8: Privacy costs for training for 1 epoch with batch size $b = 512$ for a graph with number of nodes $n = 2,609,976$ and $\ell = 1$ (only aggregate 1-hop neighbors). $\delta = 1/n$.

Remark 2.4.5. *Theorem 2.4.4 implies that our solution—training personalized models with MR-MTL and silo-specific example-level privacy on tabular data that encode local neighborhoods—**provably satisfies silo-specific node-level DP** w.r.t. any individual and all of their social contacts.*

Deterministic vs randomized neighborhood selection. Recall from Section 2.3.2 that as part of constructing training examples, we may want to deterministically prune the graph instead of randomly picking neighbors for a node every time we visit it during training. By inspecting Theorem 2.4.4, we see that the former is important in ensuring that each node has (or ever sees) up to $S - 1$ neighbors—or more precisely, is included as some other nodes' neighborhoods up to $S - 1$ times—during training for the privacy accounting to hold.

Privacy of temporal partitioning. How does our temporal partitioning / label creation strategy affect the accounting? Because the random infection windows and the corresponding labels are essentially different feature subsets of the same neighborhood features, the DP guarantees should not degrade and may even be boosted by the temporal randomness. The latter could be an interesting future direction.

Over-/under-sampling. One simple technique to improve utility is to perform data over-sampling or under-sampling since infection status is highly imbalanced in the provided dataset (recall Section 2.1). However, the former implies that we are introducing multiple views of the same data, thereby violating the desired randomness of the data sampling (or even the independence assumption of DP) and making node-level sensitivity analysis difficult; and the latter implies that the node sampling rate may become much higher and worsen privacy-utility tradeoffs. See Section 2.6 for more discussions.

2.5 Experiments and Results

We can now see the solution coming together: each silo builds a tabular dataset using neighborhood vectors for features and infection windows for labels, and each silo trains a personalized binary classifier under MR-MTL with silo-specific example-level privacy (convertible to node-level DP).

In this section, we describe the experiments and results that inform and complete our submission to the challenge. Unless otherwise stated, we experiment on two custom data-subsets extracted from the provided dataset:

1. "**Smoke**" dataset, corresponding to `client01` of the example partition[17] provided by the organizers with $n = |\mathcal{V}| = 2,609,976$ nodes and $|\mathcal{E}| = 66,073,257$ edges; and
2. "**Small**" dataset, a small partition with all households with longitude < -80.5 and latitude ≥ 38.5, corresponding to the southwest region of Virginia state, with 421,893 nodes and 8,808,360 edges.

Note that these smaller data splits are subgraph cuts from the originally provided 7.6 million-node graph. They allow us to iterate experiments faster and we apply learnings to our final submissions. As our solution leverages *model personalization* (Section 1.5) and *silo-specific example-level DP accounting* (Definition 2.4.2), the "central" performance on these subgraphs would be highly indicative of the federated performance.

Unless otherwise stated, we'll focus on **average precision** (AP or AUPRC) as the metric; it summarizes a binary classifier's performance across all operating thresholds, and random guess corresponds to the fraction of positive examples (≈ 0.002 in our dataset), instead of 0.5.

2.5.1 Overview of the Learning Algorithm

Recall from Section 2.3.2 that our learning algorithm involves, for each node:

1. sampling its neighborhood and a particular infection time window,
2. building a neighborhood feature vector and a corresponding label of infection status, and
3. training a model to predict that status (e.g. as a binary classifier).

[17] https://www.drivendata.org/competitions/103/nist-federated-learning-2-pandemic-forecasting-federated/data/

Design choice	**Values**
Model architecture	Logistic regression (linear classifier applied to neighborhood features)
Loss function	Focal Loss ($\gamma = 2, \alpha = 0.75$) [94]
Node features	Age, gender, household size, activity participation, total activity duration
Edge features	Location & activities info at contact, contact duration and start time
Graph node features	Normalized node degree and undirected PageRank
MR-MTL & FL parameters	Mean-regularization $\lambda \in \{0.0003, 0.001\}$, number of rounds $T = 2$
DP parameters	ℓ_2 clip bound $C_k \in \{5, 10, 15\}$, noise multipliers $\in \{0.01, 0.1, 0.5, 1.0, 2.75\}$
Neighborhood	$\ell = 1$ (# hops), $S_1 = 20$ (# sampled neighbors)
Other Hyperparams	Infection history $t_{\text{window}} = 21$, batch size $m = 512$, local iters $E_k = N_k/m$

Table 2.1: Design choices for our instantiated learning algorithm

Loss Function (values $\times 10^{-2}$)	AP	Recall
Binary Cross Entropy	3.096	0.05
Focal Loss ($\alpha = 0.25$)	3.036	0.0
Focal Loss ($\alpha = 0.50$)	3.165	0.259
Focal Loss ($\alpha = 0.75$)	**3.222**	**1.940**

Table 2.2: Loss functions on **Smoke**.

Features (values $\times 10^{-2}$)	AP
Node features only	0.282
Node + Contact Activity	0.639
Node + Contact Activity + Location	0.999
Node + Contact Activity (start/duration only) + Location	**1.239**

Table 2.3: Feature selection experiments on **Small**.

In our final solution, the instantiation of our algorithm involves picking relevant node and edge features, suitable model architecture, loss functions, and other hyperparameters (e.g. t_{window}). Table 2.1 summarizes such choices; see our open-source repo[18] for more implementation details. We motivate and analyze these design choices in the next section.

Our solution can be viewed as a simplified variant of a **graph neural network** (GNN) with a single step of (non-parameterized) neighborhood aggregation, followed by a single step of feature transformation using a linear layer. These simplifications, theoretically justified by [150], have made our solution a *logistic regression* over the neighborhood vectors, yielding natural benefits of efficiency, scalability, generalizability, usability, and low variance, as we discuss below. Utility-wise, our solution is within the top-2 on the leaderboards at the time of submission.

2.5.2 Component Design, Experiments, and Discussions

Model ($\times 10^{-2}$)	**Small**	**Smoke**
Random Guess	0.160	0.266
3-layer MLP	0.198	0.329
Logistic Regression	**0.282**	**2.049**

Table 2.4: AP for model architectures (using node features only).

Model architecture. Although the model design space is large for generic tabular datasets, we are specifically interested in models amenable to gradient-based private optimization and parameter averaging (recall Section 2.3). While there is recent work proposing to integrate *gradient-boosted trees* (which are exceptionally suited for the imbalanced tabular data in our case) into the federated setting with example-level DP [104], we argue that such solutions have yet to mature as the implementation can be complex with many underlying assumptions. We thus compare a simple linear model (logistic regression) and a 3-layer fully-connected neural net (MLP). In Table 2.4 we observe that logistic regression *substantially* outperforms MLPs, likely due to the high variance of the data (either due to the inherent data quality or our training data construction pipeline), and thus modeling benefits from simpler models.

[18] https://github.com/kenziyuliu/pets-challenge

Loss functions and data imbalance. Because the data are highly imbalanced—only $< 0.3\%$ of the population is positive in the final week and only up to 5% is ever in either I or R state—training naively will suffer from low recall and AP. While there are established over-/under-sampling techniques to deal with such imbalance, they are at odds with our privacy amplification (Section 2.4) and their benefits may be offset by the worsened privacy-utility tradeoffs.[19]

We instead leverage *focal loss* [94] from the computer vision literature specifically designed to emphasize hard examples (positive cases) and to down-weigh easier examples (negative cases). It is directly compatible with subsampling-amplified DP-SGD and Table 2.2 shows that it improves both the AP and recall considerably over the standard BCE loss.

However, loss weights can be slightly problematic when it comes to DP-SGD implementation because they can exacerbate *the mismatch of gradient magnitudes* between the positive and negative examples. As we will see in the following subsection, Fig. 2.9 illustrates this problem by examining the norms of the gradients in a typical batch. Data imbalance could be a fundamental issue for DP, which inherently asks for uniform guarantees across examples.

Feature selection. For node features, we first included personal information such as age (standardized), gender (binary), household size (Box-Cox transformed), binary indicators for activity participation, and the fraction of day spent at work/school/shopping. We also added graph-based features including normalized node degrees and PageRank as high centrality correlates with a higher likelihood of infection. For edge features, we included the start time and duration of contact, as well as the one-hot encodings of the activities and the binary indicators of the location (e.g. whether it supports work). Table 2.3 presents selected results. Overall, more features tend to give better utility. Curiously, *dropping* the *activity* one-hot encodings but keeping the location encodings at contact helped improve the validation AP.

Neighborhood Sampling. We are interested in the interplay between *utility* and *efficiency* when sampling neighbors in the contact graph. If we select more neighbors S_ℓ or take neighbors from higher hops ℓ, we expect better utility but more computation (e.g. for feature generation and memory usage). In particular, the 1-hop neighborhood size S_1 is easier to scale while $\ell \geq 2$ grows computation exponentially. For efficiency, we limit $\ell = 1$ for our submission. Note that such sampling is done *deterministically* for each node, equivalent to graph sparsification and capping maximum node degree to S_1. This improves both the efficiency (sampled neighbors can be cached) and privacy (S_1 remains the same over multiple epochs) at a small utility cost. Table 2.5 shows that while $S_1 = 50$ gives the best AP, it will fail with the full dataset. We use $S_1 = 20$ to balance utility and efficiency.

# **neighbors**	AP $\times 10^{-2}$	Peak RAM
$S_1 = 15$	3.013	$\approx$ 18GiB
$S_1 = 20$	3.158	$\approx$ 20GiB
$S_1 = 30$	3.244	$\approx$ 25GiB
$S_1 = 50$	**3.279**	OOM

Table 2.5: Effects of # sampled 1-hop neighbors on **Smoke**.

2.5.3 Privacy and Accuracy

We now study how the noise multiplier σ for privatizing gradients may affect the validation AP and yield different node-level DP guarantees. Table 2.6 summarizes the privacy scenarios and their tradeoffs with utility, with precise ε and AP results computed on the **Smoke** split. Importantly, since we enforce silo-specific node-level DP (Section 2.4), the privacy scenarios with the corresponding σ

[19] Alternatively, we may target for *class-specific* DP guarantees such that over-/under-sampling can be implemented and accounted for within each class. We leave this for future work. See also Section 2.5.3 for discussions.

would apply to silos with N ranging from 200k to 7.6M nodes (full centralized dataset). For our final submission, we aim to include both a "Weaker" ($\sigma = 0.01$) run and a "Moderate" ($\sigma = 0.5$) run for a privacy-utility tradeoff.

While some readers may find the large ε values mildly disturbing at a first glance and may take this out of context, it is essential to note that providing a framework to achieve DP is orthogonal to the design decision for this specific challenge. Several crucial factors linking theory and practice should also be taken into account:

Node-level DP	Noise mult σ	AP $\times 10^{-2}$
Random ($\varepsilon = 0$)	-	0.266
Strong ($\varepsilon < 10$)		
$\varepsilon \approx 9.9$	4.50	0.578
Moderate ($\varepsilon < 500$)		
$\varepsilon \approx 84$	1.00	1.351
$\varepsilon \approx 245$	0.50	2.253
Weak ($\varepsilon \geq 500$)	0.10	2.570
Weaker	0.01	3.113
Non-Private	0.00	3.244

Table 2.6: Privacy scenarios with ε and AP reported on **Smoke**. $\delta = 1/N$.

Weak DP mitigates a range of attacks in practice as DP is inherently pessimistic. Attacks such as data poisoning [140], membership inference (MI) [73, 28] and backdoors [135, 73, 145] can be meaningfully mitigated with weak DP. In fact, strong gradient clipping *alone* can cripple state-of-the-art MI attacks [28] and backdoors [73], and yield *empirically audited* $\tilde{\varepsilon} < 1$.

Linear models have a limited capacity for memorization. Unlike large models that rely on some degree of memorization [26, 27], logistic regression simply puts a weight on different attributes and it is generally impractical to infer attributes of inliers by examining such weights, particularly when datasets are large. While the decision boundary may still be skewed by outliers, adding DP would provide sufficient deniability that any single outlier is responsible for the skew.

Our node-level DP parameters (clip bound C and noise level σ) and guarantees are calibrated w.r.t. the positive examples, which are the outliers. However, the inliers or examples with lower losses may enjoy stronger guarantees than the worst-case bound [155]. Fig. 2.9 illustrates the ℓ_2-norms of a typical batch of gradient vectors. Observe that most negative examples have a very small norm (< 1), while positive examples have a large norm (partially due to the use of focal loss). Now consider the "Moderate" privacy scenario as an example: with $C = 5$ and $\sigma = 1$ ($\varepsilon \approx 84$ on **Smoke**), the noise standard deviation $C\sigma = 5$ would be *empirically* equivalent to using a noise multiplier of $\sigma_{\text{neg}} = 5$ if negative gradients have norm ≤ 1. This corresponds to an *empirical* $\tilde{\varepsilon}_{\text{neg}} \approx 7.87$, substantially improving on the worst case ε. More importantly, if every node (person) is to receive the same level of protection regardless of their infection status (matching the privacy desiderata for this task), then this *de facto* protection *covers* $> 99.5\%$ *of the population* following the class imbalance.

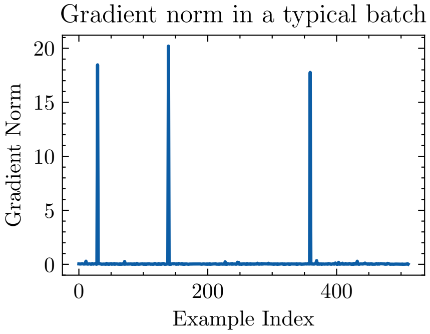

Figure 2.9: ℓ_2-norm of gradient vectors in a typical batch of 512 examples. The spikes correspond to 3 positive examples.

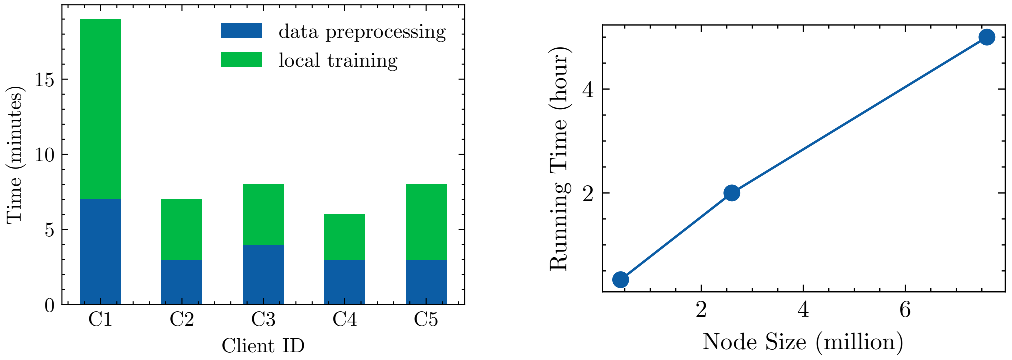

Figure 2.10: Run time metrics. Left: The fraction of processing times on clients of different size; the one-off data preprocessing (including neigihborhood sampling) can be expensive, but further training can used the preprocessed data. Right: Our method's end-to-end running time scales linearly as the node size.

Node-level DP accounting is loose and an active research area. Our accounting method builds on [37], which assumes the worst case when calculating node-level sensitivity (e.g. that a node appears in *all* of its neighbors' subgraphs). In the average case, one expects a lower degree of subgraph overlap and thus the empirical protection would be stronger. We leave improving the theory of node-level accounting to future work.

Privacy-utility tradeoffs can be easily adjusted based on need. A key benefit of offering silo-specific node-level DP vs. cryptographic guarantees is that privacy is not binary—if we expect stronger adversaries, σ can be tuned for stronger theoretical guarantees. A finite ε also means that the model enters a regime where we can quantify further necessary privacy improvements.

2.5.4 Efficiency and Scalability

Efficiency. Logistic regression is inherently efficient—it is up to 3.7× faster to run than the 3-layer MLP (e.g., 20mins versus 74mins in total for an end-to-end run on **Smoke**) and runs fast on CPUs. We also provide an efficient implementation, including vectorizing feature construction, caching features for each client across rounds and stages, and using JAX [21] to substantially speed up per-example operations of private optimization. We visualize the data preprocessing time and local training time on each client in Fig. 2.10 (top). Note that while data preprocessing takes a non-trivial amount of time, it only needs to be done once. Communication costs are negligible due to small model sizes.

Scalability. We empirically demonstrate the scalability of our submission in the centralized training setting (without federation). We consider three samples of the entire dataset with different numbers of nodes—**Small** (420k nodes), **Smoke** (2.6M), and **full dataset** (7.6M). End-to-end training times are plotted in Figure 2.10 (bottom). We see that running time grows roughly linearly with the number of nodes in the contact graph, which indicates the scalability of our algorithm and implementation. Logistic regression inherently scales to higher dimensions, and if more RAM is available, our feature generation and data loading can be also made embarrassingly parallel and drastically speed up training.

2.6 Discussions

The above concludes the main building blocks of our solution to the PETs challenge. While our solution has received positive reception from the challenge, there are indeed many areas for future work. For example:

- **Limited hyperparameter tuning**: Due to the time limit, we only tried a few configurations (or simply set a reasonable default) for hyperparameters such as infection window size t_{window}, number of hops for neighborhood aggregation ℓ, regularization strength λ, and number of local training iterations. We expect proper tuning could see utility improvements.
- **Sub-optimal implementation of example-level DP for logistic regression**: The convexity of the ML task actually offers alternatives for implementing (amplified) DP-SGD (e.g. weight-space perturbations [30], privacy amplification by iteration [49], DP-FTRL [80], novel accounting [153]) that may allow better privacy-utility tradeoffs or even data over-/under-sampling to deal with data imblance.
- **High variance of the training pipeline**: Our experiments on model architectures indeed suggest that the variance inherent in the training pipeline (partly due to our dataset construction) can lead to models underperforming; this may also cause problems with reliability/explainability.
- **Lack of justification for random (or sliding) infection windows**: Our key simplifying heuristic is that a person's likelihood of infection is inherently individual-level and depends only on her own most recent activities (specifically up to the recent t_{window} days), but this may not be the case in reality. We leave more sophisticated modeling techniques on the individual level to future work.

Chapter 3

Conclusion

In conclusion, this thesis has contributed to the growing body of research on the application of differential privacy in federated learning, offering novel insights in handling privacy, utility, and data heterogeneity in cross-silo settings.

In Chapter 1, we introduced the notion of *silo-specific example-level DP* as a more suitable privacy model for cross-silo federated learning, and analyzed several meaningful ways in which it differs from client-level DP commonly studied under the cross-device setting. We established mean-regularized multi-task learning (MR-MTL) as a simple but strong baseline under the interplay between privacy and cross-silo data heterogeneity, and our findings provide essential guidance for future work in this area, such as extending the privacy model to handle cases where data subjects have records spanning multiple data silos, extending theoretical characterizations, and developing auto-tuning algorithms for model personalization with minimal privacy overhead.

In Chapter 2, we showcased the practical application of our research through the US/UK PETs Prize Challenge, where our team's solution incorporating MR-MTL and other components, such as graph processing and privacy accounting, emerged as the winning entry in the Pandemic Forecasting and Response track. The challenge helps demonstrate the real-world impact of our work, but at the same time it also highlights the potential for further advancements in privacy-preserving federated learning solutions.

Despite the many subtleties to fully build out a working system, the main ideas presented in this thesis were exceptionally simple: data silos simply train personalized models with DP and proximity constraints. On the other hand, our work identified several important future work in this context, which we summarize below.

DP under data imbalance. DP is inherently a *uniform* guarantee, but data imbalance implies that examples are *not* created equal—minority examples (e.g., disease infection, credit card fraud) are more informative and tend to give off (much) larger gradients during model training. Should we instead apply *class-specific* (group-wise) DP or refine heterogeneous DP [78, 6] or outlier DP [103] notions to better cater for the discrepancy between data points?

Graphs and privacy. Another fundamental basis of DP is that we could delineate what is an isn't an *individual*. But as we have seen in Chapter 2, the information boundaries are often nebulous when an individual is a node in graph (think social networks and gossip propagation), particularly when the node is arbitrarily well connected. Instead of having rigid constraints (e.g., imposing a max node degree and accounting for it), are there alternative privacy definitions that offer varying degrees of protection for varying node connectedness?

Scalable private and federated trees for tabular data. Decision trees/forests tend to work extremely well for tabular data such as ours, *even with data imbalance.* However, despite recent progress [104], we argue that they are not yet mature under private and federated settings due to some underlying assumptions.

Novel training frameworks. While MR-MTL is a simple and strong baseline under our privacy granularity, it has clear limitations in terms of modeling capacity. Are there other methods that can also provide similar properties to balance the emerging privacy-heterogeneity cost tradeoff?

Honest privacy cost of hyperparameter search. When searching for better frameworks, the dependence on *hyperparameters* is particularly interesting: Section 1.7 made a surprising but somewhat depressing observation that the honest privacy cost of just tuning (on average) 10 hyperparameter configurations (values of λ in this case) may *already outweigh* the utility advantage of the best tune MR-MTL(λ^*). What does this mean if MR-MTL is already a strong baseline with just a single hyperparameter?

We hope that our work provides a useful foundation and faciltates future work in addressing the above issues and developing better privacy-preserving federated learning solutons.

Appendix A

Additional Discussions

A.1 Limitations

We discuss below some limitations of our work in addition to Section Section 1.7.

When multiple records map to the same entity. In this paper we studied the application of silo-specific example-level differential privacy in cross-silo federated learning. While this is an important initial step towards a more suitable privacy model for cross-silo FL (in contrast to the commonly studied client-level DP model), we assume that each entity that requires privacy protection has at most one record (training example) across silos (e.g. a single patient has one medical record at a hospital).

There are two characteristic cases where this assumption does not hold for all items in a silo:

- **Multiple records within a silo map to the same entity.** One example would be a student re-enrolling at the same school for multiple degree programs, thus creating multiple student records at the same silo. In such cases, the silo curator may need to carefully apply group privacy or other methods for ensuring entity-level privacy [87] to protect the entity rather than its records.
- **Multiple records across silos map to the same entity.** One example would be a person having multiple credit cards at different banks. This case is more intricate as it is harder to precisely account for the DP guarantee for this entity without knowing (1) the silos in which this entity has appeared and (2) the specific privacy targets for each of those silos. In this case, the silos may cooperate to run *private set intersection* (e.g. as considered in [102]) to privately identify this scenario, but this would by itself come at a privacy cost.

These cases are interesting avenues for future research on private cross-silo learning.[1]

Extending the analysis to deep learning cases. In Section Section 1.6 we use federated mean estimation as a simplified setting for analyzing the behavior of MR-MTL under silo-specific example-level privacy. While the analysis provides adequate insights into the empirical phenomena in Fig. 1.5, it is a simple model that does not consider the dynamic aspects of the learning settings, including (1) the Gaussian random walk component of the model updates due to DP noise applied over many training rounds, (2) the effect of communication frequency on the effect of noise reduction, (3) the

[1] The case of having individual records corresponding to multiple entities at once (e.g. one record for all family members) is slightly less interesting since example-level privacy would protect all of the entities.

concept of "client drifts" (as considered in [83]) as a result of heterogeneity and how it interfaces with the DP noises, and (4) how overparameterization may affect all of the above.

Caveats of cross-silo learning with very large local datasets. In contrast to cross-device federated learning, cross-silo federated learning is typically characterized by having a limited number of clients, each with a large local dataset. The term "large" is relative because it describes the sufficiency of the datasets for fitting good local models of a *specific class*; for example, 500 examples are likely sufficient to fit linear regression of 10 parameters, but very likely insufficient to learn a transformer [144]. In this sense, many FL problems in practice – such as large commercial banks running regression on tabular data – will in fact have local training to be the *optimal* strategy, as long as there are sufficient local data and the data from other silos are not of the same local distribution. In these cases, one should expect MR-MTL to opt for $\lambda^* \approx 0$ as federated learning is not needed at all, and thus its advantages under privacy will also be minimal.

A.2 Potential Negative Societal Impact

Our work studies the empirical behaviors that arise when applying an alternative model of differential privacy to cross-silo federated learning, and we provide a strong baseline method (MR-MTL) that fares well in this setting. In this sense, our work sheds light on and facilitates the development of a previously underexplored area of differentially private federated learning. However, because MR-MTL requires selecting a good regularization stength λ, one potential negative impact is that users may excessively tune λ on a private dataset and inadvertently leak privacy via the choice of λ (perhaps qualitatively rather than quantitatively); for example, if a silo chose a large λ for better performance, then in principle its data would look somewhat more similar to the "average" of the silo datasets. Moreover, our privacy model requires that silos add their own independent noises for their own DP targets, and this requirement may not be followed correctly (either deliberately or inadvertently) to provide vacuous DP guarantees for people's data.

Appendix B

Additional Experimental Details

B.1 Datasets and Models

Table B.1 summarizes the datasets, tasks, and models considered in our experiments. In the following, we provide details on each.

Dataset	**Task**	**# Clients (Silos)**	**Input Dim**	**Min n_k**	**Max n_k**	**Learner**
Vehicle	Classification	23	100	872	1933	SVM
School	Regression	139	28	15	175	Linear
Google Glass (GLEAM)	Classification	38	180	699	776	SVM
Heterogenous CIFAR-10	Classification	30	$32 \times 32 \times 3$	1515	1839	ConvNet
Rotated & Masked MNIST	Classification	40	$28 \times 28 \times 1$	1500	1500	ConvNet
Subsampled ADNI	Regression	9	$32 \times 32 \times 1$	45	2685	ConvNet

Table B.1: Summary of datasets, tasks, and models for our empirical studies. n_k denotes the number of training examples on client k.

Vehicle [42]. The Vehicle Sensor dataset is a binary classification dataset containing $K = 23$ data silos. Each silo (sensor) has acoustic and seismic measurements for a road segment, with each data point being a 100-dimensional feature vector describing the measurements when a vehicle passes through the road segment. The goal is to predict between two predetermined types of vehicles. We use a train/test split of 75%/25% following previous work [132], yielding $872 \leq n_k^{\text{train}} \leq 1933$ training examples on each client. We use simple linear SVMs for classification following [132, 91]. It is a suitable dataset for cross-silo FL because the number of silos K is small while each silo has sufficient data to fit a good local model, as opposed to cross-device datasets such as FEMNIST [24] where K is large but each silo has little data to learn a useful model. Moreover, we can use tight privacy budgets due to reasonably large local datasets (in terms of sufficiency for fitting a good local model) and SVMs (which are relatively noise-tolerant since decision boundaries only depend on support vectors). The dataset is accessible from the original authors.[1]

School [58, 10, 166]. The School dataset originated from the now-defunct Inner London Education

[1] https://web.archive.org/web/20110515133717/http://www.ece.wisc.edu:80/~sensit/

Authority.[2] It is a regression dataset for predicting the exam scores of 15,362 students distributed across 139 secondary schools. Each school has records for between 22 and 251 students, and each student is described by a 28-dimensional feature vector capturing attributions such as the school ranking, student birth year, and whether the school provided free meals. We perform 80%/20% train/test split in Fig. 1.3 (with $15 \leq n_k^{\text{train}} \leq 175$ training examples in each silo), and additionally consider 50%/50% and 20%/80% train/test split in Fig. G.7. We use simple linear regression models following previous work (e.g. [10, 166, 59]) to predict student scores. Like the Vehicle dataset, the School dataset is a natural cross-silo FL dataset with a limited number of clients K, each with roughly sufficient data to fit a reasonable local model. The dataset is available from [166].

Google Glass Eating and Motion (GLEAM) [121]. We also benchmark on the GLEAM dataset, a real-world head motion tracking dataset for binary classification. The motion data is collected with Google Glass, and the task is to classify the activity of the wearer (eating or not). There are in total $K = 38$ silos (wearers) and 27800 data points, with each silo containing $699 \leq n_k \leq 776$ data points. Each data point is a 180-dimensional feature vector capturing head movement of the wearers. Linear models yield reasonable utility on GLEAM and thus we use linear SVMs following previous work [132]. Like Vehicle and School, this is a suitable dataset for cross-silo FL given a small K and relatively large local datasets.

Heterogeneous CIFAR-10 Dataset. We additionally evaluate on CIFAR-10 [85], with heterogeneous client data split following previous work [136, 130] (based on the code provided by [130]). The dataset has $K = 30$ clients (silos) in total, and data heterogeneity is generated with each client having a random number of samples from 5 randomly chosen classes out of the 10 classes. Each client has $1515 \leq n_k \leq 1839$ training examples. We use a simple convolutional network with the following layers: [Conv 3×3 with 32 channels, ReLU, MaxPool 2×2 with stride 2, Conv 3×3 with 64 channels, ReLU, MaxPool 2×2 stride 2, Linear]. No padding is used for convolutional layers.

Rotated & Masked MNIST. We adapt the original MNIST dataset [86] to study the effect of structured heterogeneity on MR-MTL. For the 60000/10000 train/test images, we first perform a shuffle and then evenly separate them into $K = 40$ clients (silos), each with 1500/250 train/test images with roughly uniform distribution on the labels. We then randomly separate the silos into 4 groups of 10, and apply rotations of $\{0°, 90°, 180°, 270°\}$ to each group respectively; silos within the same group have the same rotations applied to the images, thus forming 4 natural silo clusters. To add *intra-cluster* heterogeneity, we then apply *silo-specific* random masks of 2×2 white patches; that is, all images in the same silo has the same mask, and no two silos have the same mask with very high probability. The random masks are akin to those considered in [66]. The white patches of the random mask do not overlap, and the mask ratio is the probability of a patch being applied (so the specified percentage of masked area is an expectation). Examples of generated images are shown in Fig. B.1. These image transformations introduce two types of heterogeneity identified by [81]: "covariate shift" (skew of feature distributions) and "concept drift" (same label, different features). The model architecture is the same as the one used for heterogeneous CIFAR-10.

Subsampled Alzheimer's Disease Neuroimaging Initiative (ADNI) Dataset [2]. We additionally benchmark on the ADNI dataset, which is a real-world dataset containing brain PET scans of Alzheimer's disease patients, patients with mild cognitive impairment, and healthy people taken from multiple institutions [120]. It is a regression dataset for predicting the SUVR value (a scalar ranged roughly between 0.8 and 2) from the PET scan images of a brain. We simplified the

[2] https://en.wikipedia.org/wiki/Inner_London_Education_Authority

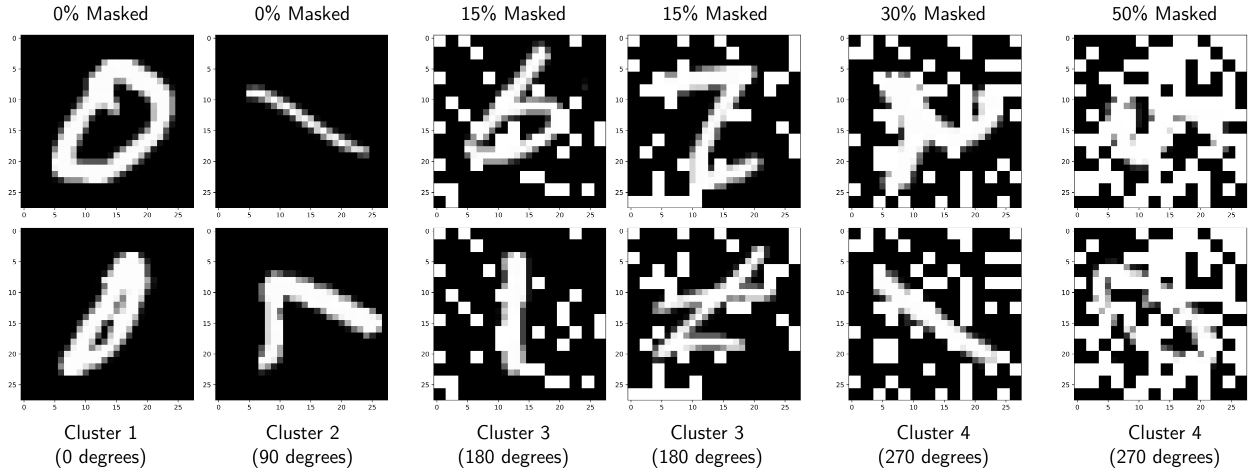

Figure B.1: **Example images of the Rotated & Masked MNIST dataset**. Each column corresponds to two images from the same (random) client from the specified cluster (thus they have the same client-specific random mask). Labels for each column from left to right: (0, 0), (1, 7), (9, 1), (2, 7), (2, 1), (5, 0).

full dataset for faster training by subsampling the axial slices generated for each brain PET scan (96 slices for scan), turning them into a gray scale image, downsampling them to size 32×32, and randomly splitting them into a 75%/25% train/test sets. There are in total $K = 9$ silos containing a total of 11040 images; each silo corresponds to a different equipment that took the PET scans and contains $45 \leq n_k^{\text{train}} \leq 2685$ training examples. See Fig. 4 of [126] and Fig. 4 of [120] for sample images. The model architecture is a simple convolutional network with the following layers: [Conv 5×5 with 32 channels, ReLU, MaxPool 2×2 with stride 2, Conv 5×5 with 64 channels, ReLU, MaxPool 2×2 stride 2, Linear]. No padding is used for convolutional layers.

B.2 License/Usage Information for Datasets

Vehicle. The Vehicle dataset was made publicly available by the original authors as a research dataset [42] and license information was unavailable. It has been subsequently used in many work (e.g. [132]).

School. The original entity that collected the School dataset [58] is defunct and license information was unavailable. The dataset has been made publicly available [166] and used extensively in previous work (e.g. [166, 162]).

Google Glass (GLEAM). The GLEAM dataset was made publicly available by the original authors and can be used for any non-commercial purposes. See this URL[3] for license and usage information.

Heterogeneous CIFAR-10. The original CIFAR-10 dataset is available under the MIT license.

Rotated & Masked MNIST. The original MNIST dataset is available under the CC BY-SA 3.0 license.

Subsampled ADNI. As per the Data Use Agreement of the ADNI dataset:[4]

[3] http://www.healthailab.org/data.html

[4] https://adni.loni.usc.edu/wp-content/uploads/how_to_apply/ADNI_Data_Use_Agreement.pdf

> Data used in preparation of this manuscript were obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (adni.loni.usc.edu). As such, the investigators within the ADNI contributed to the design and implementation of ADNI and/or provided data but did not participate in analysis or writing of this manuscript. A complete listing of ADNI investigators can be found at this URL.[5]

The data sharing and publication policy of the ADNI dataset can be found at this URL.[6] Access to the dataset must be approved by ADNI.

B.3 Benchmark Methods and Implementation

We provide more details on our benchmark personalization methods below.

Local finetuning [147, 157, 32] is one of the simplest but most effective personalization methods: once clients obtain a shared model via federated training (e.g. FedAvg), they can personalize it with additional training steps over their local dataset. This simple strategy has been shown to work very well empirically [147, 157, 32], with the work of [32] providing theoretical support that it can asymptotically achieve comparable performance to other more sophisticated methods. Moreover, in contrast to other local adaptation methods like distillation [157], local finetuning's privacy footprint under our privacy model can be easily controlled (by limiting the number of finetuning and total training steps) without qualitatives change in its behavior. In our experiments, local finetuning is implemented as FedAvg followed by local training, each taking 50% of the total number of training rounds to ensure an identical privacy budget as other baseline methods.

Ditto [91] is the current state-of-the-art method for personalization with provable benefits of robustness and fairness. It is closely related to MR-MTL because it similarly trains personalized models while ℓ^2-regularizing them towards a global model, but it differs from MR-MTL in that its global model can be obtained by a standalone solver. In particular, when no privacy is added, Ditto's modularity allows it to perfectly interpolate between local and FedAvg training with its regularization strength λ. In our experiments, we implement Ditto with the FedAvg solver and use a minimal number of iterations over the local datasets to avoid excessive privacy overhead.

Mocha [132] is a multi-task learning framework tailored for federated settings. During training, it simultaneously learns the personalized models as well as a client-relationship matrix which can model both positive and negative client relationships (in contrast, clustering methods only focus on positive relationships). One disadvantage of Mocha is that it applies to convex problems only. In particular, the original paper uses a dual formulation for efficient training, but for fair comparison and compatibility with privacy primitives (especially DP-SGD [133, 13, 1]), we implemented Mocha in its primal form and trained it with SGD in our experiments. Similarly, we align Mocha to other baselines in terms of the number of training steps to prevent privacy overhead.

IFCA [56] (and the conceptually similar **HypCluster** [106]) is a simple clustering framework proposed as an extension to FedAvg. In every round of IFCA training, the server sends k models (cluster centroids) to all clients; each client locally evaluates them over its local training data and selects the one with the lowest loss. Each client then locally trains on the selected model and returns updates only for this model, along with its index. Clients that selected the same model indices can

[5] https://adni.loni.usc.edu/wp-content/uploads/how_to_apply/ADNI_Acknowledgement_List.pdf

[6] https://adni.loni.usc.edu/wp-content/uploads/how_to_apply/ADNI_DSP_Policy.pdf

be viewed as belonging to the same cluster. We use IFCA as the representative clustering method due to its performance, simplicity, and practicality.

The privacy overhead of IFCA comes in the form of *private cluster selection*: when each client evaluates the incoming models (cluster centroids) on the local datasets, this procedure must be privatized as the selection itself may leak information about the dataset. Private selection can be implemented via the exponential mechanism [111] with sharp accounting via a bounded range analysis [127] (discussed in Section 1.2), but one must decide (e.g. as a privacy budget $\varepsilon_{\text{select}}$ for the same δ) how to share the selection cost with DP-SGD under a fixed total privacy budget $\varepsilon_{\text{total}}$.

Mitigating strategies for private cluster selection. In our experiments, we observed that if private selection is implemented naively, it can incur a prohibitive privacy overhead and destroys the final utility (e.g. Fig. 1.4). There are two important reasons: (1) unlike DP-SGD, no privacy amplification applies to private selection, and (2) the sensitivity of the training loss (which is used to select cluster centroids) is unbounded in general and must be clipped to a reasonable value (e.g. ≤ 1). We propose two mitigation strategies:

1. **Use accuracy instead of loss.** For the cluster selection metric (i.e. the score function $s(x, g)$ where x is the local dataset and g is a particular cluster centroid), we use the error rate (1 − accuracy) instead of the loss (which is used by the original authors [56, 106]). The rationale is that accuracy is a low-sensitivity function, particularly in cross-silo settings: one can show that, by enumerating the cases where the differing example between the neighboring datasets x and x' are correctly/incorrectly classified under addition/removal/replacement notions of DP, the sensitivity Δ_{acc} of s is bounded as

$$\Delta_{\text{acc}} = \max_{g} \max_{x,x'} \left| s(x, g) - s(x, g') \right| \leq \frac{1}{n-1} \tag{B.1}$$

   where n is the size of the local dataset. Since n can be large in cross-silo settings, the sensitivity can be orders of magnitude smaller than that of the loss function. With small sensitivity, we heuristically set the per-round selection privacy budget to a very small $\varepsilon_{\text{select}} = 0.03 \cdot \varepsilon_{\text{total}}$. Despite this, however, the cost of private selection can still grow quickly over the entire training process and considerably eat into DP-SGD's privacy budget.

2. **Truncate the number of cluster selection rounds.** Our second strategy is to simply run less rounds of cluster selection (e.g. to 10% of total number of training rounds, as in Fig. 1.3). This follows from the empirical analysis of [56] as well as our own experimental observation that clusters tend to converge quickly, though in some cases, clusters may not fully converge within 10% of training rounds.

Despite these strategies, however, the private selection cost can still lead to a steep utility hit. Note also that the new hyperparameter $\varepsilon_{\text{select}}$ can be tuned; for a fixed total budget $\varepsilon_{\text{total}}$, a small $\varepsilon_{\text{select}}$ means the budget for DP-SGD $\varepsilon_{\text{train}}$ is not affected by much, but the selected clusters would be very noisy and inaccurate; a large $\varepsilon_{\text{select}}$ leads to less noisy clusters (and thus smaller intra-cluster heterogeneity), but DP-SGD will correspondingly use larger noise and hurt optimization.

B.4 Training Settings

Optimizers. For simplicity of hyperparameter tuning and experimental controls, we use minibatch DP-SGD for client local training without local or server momentum for all experiments (in fact, FedAvgM [70] and FedAdam [125] were not found to be helpful on Vehicle and School). While there are more efficient solvers for the Vehicle and School datasets since we are dealing with convex problems, we want compatibility with DP-SGD [133, 13, 1] as well as tight privacy accounting with privacy amplification via subsampling (i.e. via minibatch training).

Hyperparameters. For all datasets and all methods, we set silos to train for 1 local epoch in every round (except Ditto [91] which takes 2 local epochs). For Vehicle, GLEAM, School, Heterogeneous CIFAR-10, Rotated & Masked MNIST, and subsampled ADNI respectively, the local batch size across all silos are fixed with $B = 64, 64, 32, 100, 100, 64$, and the clipping norm for per-example gradients are heuristically set to $c = 6, 6, 1, 8, 1, 0.5$. Vehicle uses $T = 400$ rounds for most experiments (except Fig. 1.2 which trains for $T = 200$ rounds); School, Google Glass, Heterogeneous CIFAR-10, and Rotated & Masked MNIST use $T = 200$; and ADNI uses $T = 500$.

For multi-task learning methods (MR-MTL, Ditto [91], Mocha [132]), we sweep the regularization strength across a grid of $\lambda \in [0.0001, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 3, 10]$ to find the best λ^* wherever applicable (e.g. Figs. 1.3, 1.5 and G.7). To compensate for the change in the gradient magnitude, we also sweep different client learning rates across a grid of $\eta \in [0.001, 0.003, 0.01, 0.03, 0.1, 0.3]$; for fair comparison, the same grid of η is swept for methods that do not involve λ (e.g. IFCA, local finetuning).[7] For Fig. 1.2 and Fig. G.1, the learning rate is fixed to $\eta = 0.01$. For all datasets, the chosen privacy parameter δ satisfy $\delta < n_k^{-1.1}$ where n_k is the local training dataset size.

Evaluation Protocol. For all datasets, we evaluate methods by the average test metric (accuracy or MSE) across the silos, weighted by their respective test sample counts. Weighted averaging allows the final test metric to reflect a method's performance over the individual test samples of the combined dataset across silos, thus fairer and more aligned (compared to uniform averaging of silo test metrics) with our privacy model where each test sample represents an entity requiring protection.

B.5 Hardware

Experiments for Vehicle, School, and Google Glass (GLEAM) from Chapter 1 are run on commodity CPUs and experiments for Heterogeneous CIFAR-10, Rotated & Masked MNIST and ADNI are run on four NVIDIA RTX A6000 GPUs. Experiments for the PETs Prize Challenge from Chapter 2 are run on commodity CPUs.

B.6 Code

Our experiments are implemented in Python with NumPy [65], JAX [21] and Haiku [68]. For private training, JAX is used to vectorize DP-SGD over per-example gradients [134]. Code for Chapter 1 is available at `https://github.com/kenziyuliu/private-cross-silo-fl`. Code for Chapter 2 is available at `https://github.com/kenziyuliu/pets-challenge`.

[7] As discussed in Section Section 1.7, releasing the results from repeated experiments (possibly with different hyperparameters) may technically compromise the privacy of the datasets [95, 117]. In our case, we are primarily interested in understanding the behaviors and tradeoffs that emerge under silo-specific example-level DP, and thus for experimental control and ease of comparison we do not account for the privacy costs from hyperparameter tuning and repetitions. We also use only public datasets in our experiments.

Appendix C

Additional Algorithmic Details

C.1 Mean-Regularized Multi-Task Learning (MR-MTL)

Algorithm 1 describes the canonical instantiation of MR-MTL [136]. Its key ingredient is the mean-regularization (Line 6 of Algorithm 1) that forces the local personalized models w_k to be close to their mean $\bar{w}$. Silo-specific example-level privacy is added by privatizing the local gradients as in DP-SGD [133, 13, 1].

Privacy of MR-MTL. Since the iterates of $\bar{w}^{(t)}$ are already differentially private (as they are the average of the private iterates $w_k^{(t)}$), the additional regularization term $\frac{\lambda}{2}\|w_k^{(t)} - \bar{w}_k^{(t-1)}\|_2^2$ (and hence MR-MTL) does not incur privacy overhead compared to local and FedAvg training.

Weighted vs Unweighted Model Updates. It is customary to weigh the model updates from each client (silo) by its training example counts, as in the original FedAvg implementation [108]. However, note that under silo-specific example-level privacy with *addition/removal* notions of DP, the example counts on each silo may itself leak sensitive information (e.g. when a silo only has one record). This is less of an issue with the *replacement* notion of DP, since neighboring datasets would have the same example counts. In our experiments we use weighted model updates and thus implicitly assume the replacement notion of DP. Nevertheless, the resulting privacy guarantees are constant factors apart and empirically we did not observe significant changes in performance when using unweighted aggregation.

C.2 IFCA Preconditioning / Warm-Starting

In Section Section 1.5, we considered an extension to MR-MTL where training is "warm-started" by a small number of rounds of clustering (via IFCA [56, 106]), followed by MR-MTL training *within* each formed cluster. Pseudocode for this procedure is shown in Algorithm 2.

Privacy of IFCA Preconditioning. Observe that as with MR-MTL, the gradient steps satisfy silo-specific example-level DP regardless or whether the steps are made on the cluster models or the personalized models. IFCA preconditioning introduces privacy overhead in the form of private cluster selection (Line 6 of Algorithm 2; discussed in Section 1.2 and Appendix B.3), which splits

Algorithm 1 Mean-Regularized Multi-Task Learning

```
Input: Initial client models $\{w_k^{(0)}\}_{k \in [K]}$, and mean model $\bar{w}^{(0)}$.
for training round $t = 1, ..., T$ do
    Server sends $\bar{w}^{(t-1)}$ to every client.
    for client $k = 1, ..., K$ in parallel do
        Set model iterate: $w_k^{(t)} \leftarrow w_k^{(t-1)}$.
        For every batch $(x, y)$, client updates $w_k^{(t)}$ using SGD or DP-SGD with batch loss
        $\ell(w_k^{(t)}, x, y)$ and gradient
            $\nabla_{w_k^{(t)}} \left[ \ell \left( w_k^{(t)}, x, y \right) + \frac{\lambda}{2} \left\| w_k^{(t)} - \bar{w}^{(t-1)} \right\|_2^2 \right].$
        Return model update $\Delta_k^{(t)} = w_k^{(t)} - w_k^{(t-1)}$.
    Server updates $\bar{w}^{(t)} = \bar{w}^{(t-1)} + \frac{1}{K} \sum_{k=1}^{K} \Delta_k^{(t)}$ (may weigh $\Delta_k^{(t)}$ by client example counts).
Output: Personalized models $w_k^{(T)}$ for all $i \in [K]$.
```

the total privacy budget with DP-SGD. As a result the noise scale for DP-SGD would be increased and can be numerically determined.

Algorithm 2 IFCA-Preconditioned MR-MTL

```
Input: Initial client models $\{w_k^{(0)}\}_{k \in [K]}$, number of clusters $G$, initial cluster models $\{\bar{w}_g^{(0)}\}_{g \in [G]}$,
    total number of rounds $T$, and the number of initial clustering rounds $T_{\text{cluster}}$.
# IFCA preconditioning rounds
for IFCA training round $t = 1, ..., T_{\text{cluster}}$ do
    Server sends cluster models $\{\bar{w}_g^{(t-1)}\}_{g \in [G]}$ to every client.
    for client $k = 1, ..., K$ in parallel do
        Use Exponential Mechanism (Section 1.2) to select best cluster $\bar{w}_{g^*(k)}^{(t-1)}$ from
        $\{\bar{w}_g^{(t-1)}\}$ with loss/error rate function $s$ and local dataset $(X_k, Y_k)$.
        Set model iterate: $w_k^{(t)} \leftarrow \bar{w}_{g^*(k)}^{(t-1)}$.
        For every batch $(x, y)$, client updates $w_k^{(t)}$ using SGD or DP-SGD with batch loss
        $\ell(w_k^{(t)}, x, y)$ and gradient $\nabla_{w_k^{(t)}} \ell\left(w_k^{(t)}, x, y\right)$.
        Return model update $\Delta_k^{(t)} = w_k^{(t)} - \bar{w}_{g^*(k)}^{(t-1)}$ and selected cluster index $g^*(k)$.
    for each cluster $g \in [G]$ do
        Server applies (weighted) model updates to cluster $g$ with the associated client indices:

        $$\bar{w}_g^{(t)} = \bar{w}_g^{(t-1)} + \frac{1}{|\mathcal{K}_g|} \sum_{k \in \mathcal{K}_g} \Delta_k^{(t)}, \quad \text{where } \mathcal{K}_g \equiv \{k \in [K] \mid g^*(k) = g\}. \qquad \text{(C.1)}$$

# MR-MTL rounds with regularization towards frozen cluster centroids
for MR-MTL training round $t = T_{\text{cluster}} + 1, ..., T$ do
    Server sends cluster models $\{\bar{w}_g^{(t-1)}\}_{g \in [G]}$ to every client.
    for client $k = 1, ..., K$ in parallel do
        Client retrieves the last selected cluster centroid $\bar{w}_{g^*(k)}^{(t-1)}$.
        Set model iterate: $w_k^{(t)} \leftarrow w_k^{(t-1)}$ (using the personalized model from last round).
        For every batch $(x, y)$, client updates $w_k^{(t)}$ using SGD or DP-SGD [133, 13, 1] with batch loss
        $\ell(w_k^{(t)}, x, y)$ and gradient

        $$\nabla_{w_k^{(t)}} \left[ \ell\left(w_k^{(t)}, x, y\right) + \frac{\lambda}{2} \left\| w_k^{(t)} - \bar{w}_{g^*(k)}^{(t-1)} \right\|_2^2 \right].$$

        Return model update $\Delta_k^{(t)} = w_k^{(t)} - \bar{w}_{g^*(k)}^{(t-1)}$ and the cluster index $g^*$.
    for each cluster $g \in [G]$ do
        Server applies (weighted) model updates to cluster $g$ with the associated client indices:

        $$\bar{w}_g^{(t)} = \bar{w}_g^{(t-1)} + \frac{1}{|\mathcal{K}_g|} \sum_{k \in \mathcal{K}_g} \Delta_k^{(t)}, \quad \text{where } \mathcal{K}_g \equiv \{k \in [K] \mid g^*(k) = g\}. \qquad \text{(C.2)}$$

Output: Personalized models $w_k^{(T)}$ for all $i \in [K]$.
```

Appendix D

Additional Analysis Details

In this section we provide additional details for the analysis presented in Section Section 1.6. We also extend the analysis to the case with varying n, σ, σ_{DP} for each silo in Appendix D.5.

D.1 Notations

Unless otherwise specified, we used the following notations throughout the analysis:

- K denotes the total number of silos (clients) with indices $k \in [K]$;
- n denotes the number of examples on each silo;
- w_k denotes the true center of silo k's data distribution;
- $X_k \equiv \{x_{k,i}\}_{i \in [n]}$ denotes the local dataset with n data points;
- $\hat{w}_k$ denotes the best local estimator of w_k;
- $\bar{w}$ denotes the average of local estimators;
- $\hat{w}_{\setminus k} \triangleq \frac{1}{K-1} \sum_{j \neq k, j \in [K]} \hat{w}_j$ denotes the external average estimators from the perspective of k;
- σ^2 denotes the sampling variance of the local data X_k;
- τ^2 denotes the sampling variance of the local data centers w_k (hence a measure of data heterogeneity across silos); and
- σ_{DP}^2 denotes the DP noise variance on each silo to satisfy silo-specific example-level privacy.

D.2 MR-MTL Formulation

The general formulation of the mean-regularized MTL objective may be expressed as

$$\min_{w_k, k \in [K]} \frac{1}{K} \sum_{k=1}^{K} h_k(w_k) \quad \text{with} \quad h_k(w) \triangleq F_k(w) + \frac{\lambda}{2} \|w - \bar{w}\|_2^2, \tag{D.1}$$

where $F_k(\cdot)$ is the local objective for client k, $\bar{w} \triangleq \frac{1}{K}\sum_{k=1}^{K} w_k$ is the average model, and $\lambda \geq 0$ is the regularization strength. A larger λ enforces the models to be closer to each other, and $\lambda = 0$ reduces the problem to local training. In particular, unlike Ditto [91], MR-MTL may *not* recover FedAvg [108] under SGD as $\lambda \to \infty$, since λ essentially changes the ratio between the local objective gradients and the regularization gradients. For the purposes of our analysis, we assume F_k is strongly convex for all $k \in [K]$.

D.3 Assumptions

Our characterization of MR-MTL on makes the following simplifying assumptions.

Assumption D.3.1. *All clients (silos) have the same number of data points n, data sampling variance σ^2, and DP noise variance σ^2_{DP}.*

Assumption D.3.2. *A sufficiently large clipping c can be selected such that $\|x_{k,i}\|_2 \leq c$ with high probability.*

Note that Assumption D.3.1 primarily serves to make results cleaner and easily interpretable (see Appendix D.5 for extensions). Assumption D.3.2 is mild since Gaussians have strong tail decay.

D.4 Omitted Details and Proofs

Our analysis (particularly Lemma 1.6.2) makes use of the following lemma to determine the posterior of an unknown parameter given several independent Gaussian observations.

Lemma D.4.1 (Lemma 11 of [105])**.** *Let $\theta \in \mathbb{R}$ be a constant (non-informative prior). Let $\{\phi_k \triangleq \theta + z_k\}_{k\in[K]}$ with $z_k \sim \mathcal{N}(0, \sigma_k^2)$ be m independent noisy observations of θ with variances $\{\sigma_k^2\}_{k\in[K]}$. Then, conditioned on $\{\phi_k\}_{k\in[K]}$, with $\sigma_\theta^2 \triangleq (\sum_{k\in[K]} 1/\sigma_k^2)^{-1}$, we have*

$$\theta = \sigma_\theta^2 \sum_{k\in[K]} \frac{\phi_k}{\sigma_k^2} + z, \quad \textit{where } z \sim \mathcal{N}(0, \sigma_\theta^2). \tag{D.2}$$

This lemma allows us to express the local true centers w_k conditioned on the local empirical estimates $\hat{w}_k$, the external empirical estimates $\hat{w}_{\setminus k}$, and the local datasets $\{X_k\}_{k\in[K]}$.

Below we present omitted proofs for the main results presented in Section Section 1.6. We restate the lemmas and theorems for convenience.

Lemma 1.6.1. The minimizer of the silo-specific objective $h_k(w)$ (see Eq. (1.7), Eq. (D.1)) is

$$\hat{w}_k(\lambda) = \alpha \cdot \hat{w}_k + (1-\alpha)\cdot \hat{w}_{\setminus k}, \quad \text{where} \quad \alpha \triangleq \frac{K+\lambda}{(1+\lambda)K} \in (1/K, 1]. \tag{D.3}$$

Proof of Lemma 1.6.1. First note that $\bar{w}$ is independent of λ. Since we assume F_k is convex, h_k is also convex, and the proof follows from taking the derivative of h_k and setting it to 0:

$$\frac{\partial h_k(w)}{\partial w} = \frac{1}{n}\sum_{i=1}^{n}(w - x_{k,i}) + \lambda(w - \bar{w}) = (w - \hat{w}_k) + \lambda(w - \bar{w}), \tag{D.4}$$

$$\hat{w}_k(\lambda) = \frac{1}{1+\lambda}\hat{w}_k + \frac{\lambda}{1+\lambda}\bar{w} = \frac{K+\lambda}{(1+\lambda)K} \cdot \hat{w}_k + \frac{\lambda(K-1)}{(1+\lambda)K}\hat{w}_{\setminus k}. \tag{D.5}$$

Note here that in the non-private case, the local estimator is given by the empirical average: $\hat{w}_k = \frac{1}{n}\sum_{i=1}^n x_{k,i}$. In the private case, under Assumption D.3.2, the local estimator is $\hat{w}_k = \frac{1}{n}\left(\xi_k + \sum_{i=1}^n x_{k,i}\right)$ where $\xi_k \sim \mathcal{N}(0, \sigma_{\mathrm{DP}}^2)$ is the one-shot privacy noise, and we can similarly arrive at the same $\hat{w}_k(\lambda)$ expression. □

Lemma 1.6.2. Let $\sigma_{\mathrm{loc}}^2 \triangleq \sigma^2/n + \sigma_{\mathrm{DP}}^2/n^2$ denote the local variance on each silo as a result of data sampling and DP noise. Then, given $\hat{w}_k$ and $\hat{w}_{\setminus k}$, w_k can be expressed as

$$w_k = \mu_k + \zeta_k \tag{D.6}$$

$$\text{where} \quad \zeta_k \sim \mathcal{N}(0, \sigma_w^2), \tag{D.7}$$

$$\text{with} \quad \sigma_w^2 \triangleq \left(\frac{1}{\sigma_{\mathrm{loc}}^2} + \frac{K-1}{K\tau^2 + \sigma_{\mathrm{loc}}^2}\right)^{-1} \tag{D.8}$$

$$\text{and} \quad \mu_k \triangleq \sigma_w^2 \left(\frac{1}{\sigma_{\mathrm{loc}}^2} \cdot \hat{w}_k + \frac{K-1}{K\tau^2 + \sigma_{\mathrm{loc}}^2} \cdot \hat{w}_{\setminus k}\right). \tag{D.9}$$

Proof of Lemma 1.6.2. When given θ, we can express $w_k = \theta + z_k$ where $z_k \sim \mathcal{N}(0, \tau^2)$, and by symmetry when given w_k,

$$\theta = w_k + z_k \quad \text{where} \quad z_k \sim \mathcal{N}(0, \tau^2). \tag{D.10}$$

Note also that when given the local dataset X_k, $\hat{w}_k$ is a noisy observation of w_k with Gaussian noise from data sampling and added privacy:

$$\hat{w}_k = w_k + \hat{z}_k \quad \text{where} \quad \hat{z}_k \sim \mathcal{N}(0, \sigma_{\mathrm{loc}}^2). \tag{D.11}$$

Moreover, when given the local datasets $\{X_j\}_{j\in[K], j\neq k}$, $\hat{w}_{\setminus k}$ can be viewed as a noisy observation of θ with Gaussian noise from the silo heterogeneity and the empirical mean of the local estimators:

$$\hat{w}_{\setminus k} = \theta + \hat{z}_{\setminus k} \quad \text{where} \quad \hat{z}_{\setminus k} \sim \mathcal{N}\left(0, \frac{\tau^2 + \sigma_{\mathrm{loc}}^2}{K-1}\right). \tag{D.12}$$

Combining (D.12) with (D.10) we have, when given w_k and the local datasets $\{X_j\}_{j\in[K], j\neq k}$,

$$\hat{w}_{\setminus k} = w_k + z_{\setminus k} \quad \text{where} \quad z_{\setminus k} \sim \mathcal{N}\left(0, \tau^2 + \frac{\tau^2 + \sigma_{\mathrm{loc}}^2}{K-1}\right) \equiv \mathcal{N}\left(0, \frac{K\tau^2 + \sigma_{\mathrm{loc}}^2}{K-1}\right). \tag{D.13}$$

Invoking Lemma D.4.1 on Eq. (D.11) and Eq. (D.13) we have the desired μ_k and σ_w^2.

A key observation for this derivation is that the local datasets $\{X_k\}_{k\in[K]}$ form the Markov blankets of $\hat{w}_k$ and $\hat{w}_{\setminus k}$ (as they are sampled *after* w_k are sampled, and the estimators $\hat{w}_k$ and $\hat{w}_{\setminus k}$ are computed based on the datasets). Thus, given $\{X_k\}_{k\in[K]}$, $\hat{w}_k$ and $\hat{w}_{\setminus k}$ can be viewed as independent observations of w_k and Lemma D.4.1 applies. □

Theorem 1.6.3 (Optimal MR-MTL estimate). Let λ^* be the optimal λ that minimizes the generalization error. Then,

$$\lambda^* = \operatorname*{argmin}_{\lambda} \mathbb{E}\left[(w_k - \hat{w}_k(\lambda))^2 \mid \hat{w}_k, \hat{w}_{\setminus k}, \{X_k\}_{k\in[K]}\right] = \frac{\sigma_{\mathrm{loc}}^2}{\tau^2} = \frac{1}{n\tau^2}\left(\sigma^2 + \frac{\sigma_{\mathrm{DP}}^2}{n}\right). \tag{D.14}$$

Proof of Theorem 1.6.3. The objective of finding λ^* equates to finding an expression for λ such that when w_k is viewed as a noisy observation of $\hat{w}_k(\lambda)$ (which is an interpolation of $\hat{w}_k$ and $\hat{w}_{\setminus k}$ from Lemma 1.6.1), the coefficients for $\hat{w}_k$ and $\hat{w}_{\setminus k}$ in $\hat{w}_k(\lambda)$ matches those of w_k (Lemma 1.6.2). That is, we want λ^* such that

$$\frac{K+\lambda^*}{(1+\lambda^*)K} = \frac{\sigma_w^2}{\sigma_{\text{loc}}^2} = \frac{1}{\sigma_{\text{loc}}^2} \cdot \left(\frac{1}{\sigma_{\text{loc}}^2} + \frac{K-1}{K\tau^2 + \sigma_{\text{loc}}^2} \right)^{-1}. \tag{D.15}$$

Simpifying gives $\lambda^* = \sigma_{\text{loc}}^2 / \tau^2$. Note that $\hat{w}_k(\lambda^*)$ is the (conditional) MMSE estimator of w_k. □

Note that Eq. (D.14) measures the *generalization* error because w_k is the (unobserved) true center of the local data distribution around which testing data points would be drawn.

Proposition 1.6.5 (Optimal error gap to local training). Denote the error of the local estimate as

$$\mathcal{E}_{\text{loc}} \triangleq \mathbb{E}\left[(w_k - \hat{w}_k)^2 \mid X_k \right], \tag{D.16}$$

and let $\Delta_{\text{loc}} \triangleq \mathcal{E}_{\text{loc}} - \mathcal{E}^*$ be its error gap to the optimal estimate. Then,

$$\Delta_{\text{loc}} = \left(1 - \frac{1}{K}\right) \cdot \frac{\sigma_{\text{loc}}^4}{\sigma_{\text{loc}}^2 + \tau^2}. \tag{D.17}$$

Proposition 1.6.6 (Optimal error gap to FedAvg). Denote the error of the global (FedAvg) estimate as

$$\mathcal{E}_{\text{fed}} \triangleq \mathbb{E}\left[(w_k - \bar{w})^2 \mid \{X_k\}_{k \in [K]} \right], \tag{D.18}$$

and let $\Delta_{\text{fed}} \triangleq \mathcal{E}_{\text{fed}} - \mathcal{E}^*$ be its error gap to the optimal estimate. Then,

$$\Delta_{\text{fed}} = \left(1 - \frac{1}{K}\right) \cdot \frac{\tau^4}{\sigma_{\text{loc}}^2 + \tau^2}. \tag{D.19}$$

Proofs of Propositions 1.6.5 and 1.6.6. From Corollary 1.6.4 we know that the error of the optimal estimator $\hat{w}_k(\lambda^*)$ of w_k is $\mathcal{E}^* = \sigma_w^2 = \frac{\sigma_{\text{loc}}^2(\sigma_{\text{loc}}^2 + K\tau^2)}{K(\sigma_{\text{loc}}^2 + \tau^2)}$. For $\mathcal{E}_{\text{loc}}$, we know from earlier that the local estimator $\hat{w}_k = \hat{w}_k(0)$ of w_k has an MSE/variance of $\mathcal{E}_{\text{loc}} = \sigma_{\text{loc}}^2$. For $\mathcal{E}_{\text{fed}}$, we can view $\bar{w}$ as a noisy observation of w_k with MSE/variance $\mathcal{E}_{\text{fed}} = \frac{\sigma_{\text{loc}}^2 + (K-1)\tau^2}{K}$. The results of Propositions 1.6.5 and 1.6.6 thus follow from re-arranging terms of $\mathcal{E}_{\text{loc}} - \mathcal{E}^*$ and $\mathcal{E}_{\text{fed}} - \mathcal{E}^*$ respectively. □

Lemma 1.6.7 (Error of $\hat{w}_k(\lambda)$). Let $\mathcal{E}_\lambda \triangleq \mathbb{E}\left[(w_k - \hat{w}_k(\lambda))^2 \mid \hat{w}_k, \hat{w}_{\setminus k}, \{X_k\}_{k \in [K]} \right]$ be the error of MR-MTL as a function of λ. Then,

$$\mathcal{E}_\lambda = \left(1 - \frac{1}{K}\right) \frac{\sigma_{\text{loc}}^2 + \lambda^2 \tau^2}{(\lambda+1)^2} + \frac{\sigma_{\text{loc}}^2}{K}. \tag{D.20}$$

Proof of Lemma 1.6.7. The result follows from the variance of $\hat{w}_k(\lambda)$ when viewed as a noisy observation of w_k. Specifically, from Lemma 1.6.1 we know that $\hat{w}_k(\lambda)$ is an interpolation (parameterized by $\alpha \triangleq \frac{K+\lambda}{(1+\lambda)K}$) between $\hat{w}_k$ and $\hat{w}_{\setminus k}$. We thus have (with some rearrangement),

$$\mathcal{E}_\lambda = \alpha^2 \cdot \sigma_{\text{loc}}^2 + (1-\alpha)^2 \cdot \frac{K\tau^2 + \sigma_{\text{loc}}^2}{K-1} = \left(1 - \frac{1}{K}\right) \frac{\sigma_{\text{loc}}^2 + \lambda^2 \tau^2}{(\lambda+1)^2} + \frac{\sigma_{\text{loc}}^2}{K}. \tag{D.21}$$

□

The proof of Theorem 1.6.8 follows directly from Lemma 1.6.7 by subtracting the common terms between private and non-private MR-MTL estimates.

D.5 The Case of Heterogeneous Privacy Requirements

In the main analysis, we assumed for simplicity that each silo has the same values of local dataset size n, local data sampling variance σ^2, and DP noise variance σ^2_{DP}. In this section, we consider the case where each silo k has custom values for the above, denoted as n_k, σ^2_k, and $\sigma^2_{k,\mathrm{DP}}$, to arrive at a silo-specific "local variance" $\tilde{\sigma}^2_k$:

$$\tilde{\sigma}^2_k \triangleq \frac{\sigma^2_k}{n} + \frac{\sigma^2_{k,\mathrm{DP}}}{n^2}. \tag{D.22}$$

With a slight abuse of notation, let us use $C+\mathcal{N}(\mu,\sigma^2)$ to denote drawing an iid sample of Gaussian noise with mean μ and variance σ^2 and add it to some value C. Then, following Lemma 1.6.2 and the same set of conditions, we can express the local model $\hat{w}_k$ and the non-local model $\hat{w}_{\setminus k}$ as:

$$\begin{aligned}
\hat{w}_k &= w_k + \mathcal{N}(0, \tilde{\sigma}^2_k), && \text{(D.23)}\\
\hat{w}_{\setminus k} &= \theta + \mathcal{N}\left(0, \frac{\tau^2}{K-1} + \frac{\sum_{j\neq k}\tilde{\sigma}^2_j}{(K-1)^2}\right) && \text{(D.24)}\\
&= w_k + \mathcal{N}\left(0, \tau^2 + \frac{\tau^2}{K-1} + \frac{\sum_{j\neq k}\tilde{\sigma}^2_j}{(K-1)^2}\right) && \text{(D.25)}\\
&= w_k + \mathcal{N}\left(0, \frac{K\tau^2}{K-1} + \frac{\sum_{j\neq k}\tilde{\sigma}^2_j}{(K-1)^2}\right), && \text{(D.26)}
\end{aligned}$$

and the true local center w_k can be expressed in terms of $\hat{w}_k$ and $\hat{w}_{\setminus k}$ with silo-specific local variances:

Lemma D.5.1. *Given $\hat{w}_k$ and $\hat{w}_{\setminus k}$, w_k can be expressed as*

$$w_k = \bar{\mu}_k + \mathcal{N}(0, \bar{\sigma}^2_k) \tag{D.27}$$

with

$$\begin{aligned}
\bar{\sigma}^2_k &\triangleq \left(\frac{1}{\tilde{\sigma}^2_k} + \frac{(K-1)^2}{(K-1)K\tau^2 + \sum_{j\neq k}\tilde{\sigma}^2_j}\right)^{-1}, && \text{(D.28)}\\
\bar{\mu}_k &\triangleq \frac{\bar{\sigma}^2_k}{\tilde{\sigma}^2_k}\cdot\hat{w}_k + \frac{(K-1)^2\bar{\sigma}^2_k}{(K-1)K\tau^2 + \sum_{j\neq k}\tilde{\sigma}^2_j}\cdot\hat{w}_{\setminus k}. && \text{(D.29)}
\end{aligned}$$

Proof. The proof follows similarly from Lemma 1.6.2: we apply Lemma D.4.1 to the expressions of $\hat{w}_k$ and $\hat{w}_{\setminus k}$ as noisy observations of w_k. □

From here we can derive the best λ^*_k *specific to each silo.*

Theorem D.5.2 (Optimal λ for silo k)**.** *The optimal choice of the* MR-MTL *regularization strength for silo k is given by*

$$\lambda_k^* = \frac{\tilde{\sigma}_k^2}{\tau^2 + \dfrac{1}{K}\left(\dfrac{\sum_{j\neq k}\tilde{\sigma}_j^2}{K-1} - \tilde{\sigma}_k^2\right)} \tag{D.30}$$

Proof. The result follows from re-arranging and solving for λ_k^* in the following equation, as in Theorem 1.6.3:

$$\frac{K+\lambda_k^*}{(1+\lambda_k^*)K} = \frac{\bar{\sigma}_k^2}{\tilde{\sigma}_k^2} \tag{D.31}$$

where μ_k is defined in Lemma D.5.1. □

Theorem D.5.2 suggests that if silos have varying local variance (as a result of varying local dataset sizes, inherent data variances, and silo-specific example-level DP requirements), then it is optimal to have each silo choose its own λ_k^*. In particular, the term $\dfrac{\sum_{j\neq k}\tilde{\sigma}_j^2}{K-1} - \tilde{\sigma}_k^2$ suggests that the optimal λ_k^* depends on how much of an "outlier" is silo k compared to the rest of silos: if it has a smaller local variance, then it benefits from a smaller λ_k^* since other silos are noisy; if it has a larger local variance, then λ_k^* would be larger to encourage silo k to "conform". Note that under this simple analysis, one could have $\lambda_k^* < 0$ (when $\tilde{\sigma}_k^2 > K\tau^2 + \frac{1}{K-1}\sum_{j\neq k}\tilde{\sigma}_j^2$); this case stems from having a gradually larger $\tilde{\sigma}_k^2$ and it suffices to choose $\lambda^* \to \infty$ (i.e. fall back to FedAvg training).

Appendix E

Additional Details for Private Hyperparameter Tuning (Section 1.7)

In Section Section 1.7 we took an alternative view on the benefits of personalization by examining the *privacy cost* of tuning the hyperparameter responsible for navigating the tradeoff between costs from heterogeneity and privacy. In this section, we expand on the omitted details and provide more discussions.

E.1 Additional Background on Privacy Costs of Hyperparameter Tuning

We consider the typical hyperparameter tuning procedure (denoted as HPO) where a base algorithm M, such as one run of a machine learning algorithm to produce a model, is executed h times with different hyperparameters and the best result is recorded. The hyperparameters may be high-dimensional; e.g. vector of learning rate and batch size. Typically, h is a constant (e.g. we sweep a fixed grid of learning rates and batch sizes).

On a high level, the works of Liu and Talwar [95] and Papernot and Steinke [117] show that, with a constant h, one can construct M where the privacy guarantee of the HPO output is (roughly) a factor of h weaker than the original privacy guarantee. Specifically, [117] gives the following proposition that applies to both pure DP and Rényi DP (and thus related to approximate DP).

Proposition E.1.1 (Proposition 17 of [117])**.** *For any $\varepsilon > 0$ and any $\alpha > 1$, there exist some algorithm M that satisfy $(\varepsilon, 0)$-DP and the corresponding HPO algorithm A that runs M for h times and returns only the best result, such that*

1. *A is not $(\tilde{\varepsilon}, 0)$-DP for any $\tilde{\varepsilon} < h\varepsilon$, and*
2. *A is not $(\alpha, \tilde{\varepsilon}(\alpha))$-Rényi DP for any $\tilde{\varepsilon}(\alpha) < \hat{\varepsilon}(\alpha)$ where*

$$\hat{\varepsilon}(\alpha) = h\varepsilon - \frac{h \log(1 + \exp(-\varepsilon))}{\alpha - 1}. \tag{E.1}$$

Both the pure DP and RDP results suggest that running the typical HPO procedure with a fixed h can be as bad as running M and h times and releasing *all* results.

The authors of [95] and [117] provided a mitigating strategy where one simply makes h (the number of tuning runs) a random variable. The authors of [117] provide an improved analysis and considered two distributions for h in particular: the truncated negative binomial (TNB) distribution and the Poisson distribution.

Here, we focus on TNB as it provides a spectrum of results for privacy-utility tradeoff, allowing us to choose several points on this spectrum that offer small (but realistic) privacy overhead. We combine it our analysis in Section Section 1.6 to motivate our discussions on the practical complications of deploying personalization methods.

Specifically, the probability mass function of the TNB distribution is given by

$$f(h) = \begin{cases} \frac{(1-\gamma)^h}{\gamma^{-\eta}-1} \prod_{\ell=0}^{h-1} \left(\frac{\ell+\eta}{\ell+1}\right) & \text{for } \eta \neq 0, \\ \frac{(1-\gamma)^h}{h \log(1/\gamma)} & \text{for } \eta = 0, \end{cases} \quad \text{with expected value } \mathbf{E}[h] = \begin{cases} \frac{\eta(1-\gamma)}{\gamma(1-\gamma^\eta)} & \text{for } \eta \neq 0, \\ \frac{1/\gamma - 1}{\log(1/\gamma)} & \text{for } \eta = 0, \end{cases}$$

where η and γ are parameters of the TNB distribution. The privacy of the HPO output using h sampled from the TNB distribution is given by the following theorem from [117].

Theorem E.1.2 (Theorem 2 of [117])**.** *For any $\eta > 1$, any $\gamma \in (0, 1)$, and any algorithm M that satisfy both $(\alpha_1, \varepsilon(\alpha_1))$-RDP and $(\alpha_2, \varepsilon(\alpha_2))$-RDP with $\alpha_1 > 1$ and $\alpha_2 \geq 1$, the HPO algorithm A that runs M for h times and returns only the best result with $h \sim \mathrm{TNB}(\eta, \gamma)$ satisfies $(\alpha_1, \tilde{\varepsilon})$-RDP with*

$$\tilde{\varepsilon} = \varepsilon(\alpha_1) + (1+\eta)\left(1 - \frac{1}{\alpha_2}\right)\varepsilon(\alpha_2) + \frac{(1+\eta)\log(1/\gamma)}{\alpha_2} + \frac{\log \mathbf{E}[h]}{\alpha_1 - 1}. \tag{E.2}$$

Theorem E.1.2 tells us that by drawing h from the TNB distribution, the privacy cost of the HPO procedure is a *constant* factor of the original cost of M for pure DP (when $\alpha_1, \alpha_2 \to \infty$), or up to a factor of $\log \mathbf{E}[h]$ for approximate DP. This substantially improves the default HPO procedure.

E.2 Interpretation of Fig. 1.7

In Fig. 1.7, we considered 6 pairs of (η, γ) to arrive at the same $\mathbf{E}[h] = 10$ (i.e. on average we want to try 10 values of λ when tuning MR-MTL). Intuitively, for a fixed $\mathbf{E}[h]$, we can expect a smaller η (and consequently a smaller γ) to give a tighter privacy guarantee from Theorem E.1.2 (i.e. the privacy cost of hyperparameter tuning is smaller). At the same time, however, the TNB distribution would be more concentrated around $h = 1$ with a smaller η, which means we may end up trying only 1 hyperparameter even when $\mathbf{E}[h]$ is large. (η, γ) thus provides an implicit privacy-utility tradeoff.

In Fig. 1.7 we chose a list of η values that favors strong privacy while being realistic, since picking $\eta \to -1$ (or just $\eta < -0.5$) would give a even smaller privacy overhead, but it will perform very poorly utility-wise and was not considered by [117]. We then applied Theorem E.1.2 with these (η, γ) values to the the expected error of MR-MTL as a function of λ under mean estimation, by computing and applying the extra noise needed to account for the privacy cost of tuning. Here, the closed form of the expected error of MR-MTL is derived in Lemma 1.6.7, but in Fig. 1.7 we ran numerical simulations with 500 repetitions.

Fig. 1.7 (a) and (c) suggests that there are settings in which, after applying Theorem E.1.2, the best λ^* one could find would already give an error that is larger than simply running FedAvg and local training *without any tuning*, respectively (the **green** curves). Fig. 1.7 (b) suggests that in less extreme cases, the private hyperparameter tuning cost would significantly diminish the utility improvement of MR-MTL.

E.3 Implementing Hyperparameter Tuning under Silo-Specific example-level DP

Under silo-specific example-level privacy, it can be intricate to implement the improved, randomized hyperparameter tuning protocol described above in Appendix E.1. For silo k to satisfy its own $(\varepsilon_k, \delta_k)$ example-level DP target, it must draw h *independently* to all other silos, and it must only return the best result of hyperparameter tuning at the *end of training* (instead of returning the best model iterate at every round).

Using MR-MTL as an example, the above has several implications:

1. Each silo should now maintain a list of personalized models, each corresponding to a choice of λ.
2. Depending on the specific distribution of h, silos may or may not end up trying the same set of λ values.
3. During training rounds, silos must return model updates for – and consequently regularize their personalized models towards – a "pivot" model using some public $\bar{\lambda}$.

The analysis in Appendix D.5 suggests that having silo-specific choices of λ (#2 above) should not have adverse effects, but the above adaptations in general may influence final utility or convergence properties in iterative learning settings.

Other more sophisticated implementation (potentially tailored to the specific personalization method) may also be possible. It would be an interesting future direction to study the best approaches for implementing private hyperparameter tuning under silo-specific example-level privacy.

Appendix F

Future Directions for Auto-tuning / Online Estimation of λ

Apart from tuning λ for MR-MTL (or other hyperparameters for other personalization methods) non-adaptively with grid search or random search, another approach is to leverage some form of online estimation as in [143, 9, 118] (mentioned in Section Section 1.7). That is, λ is adjusted at every training round or iteration such that it gradually converges to a good λ^* (in expectation).

This line of approaches would be promising if their privacy overhead (if any) can be contained within a small factor of the original privacy guarantee—smaller than that offered by the randomized tuning procedure discussed above in Appendix E.1. However, since the hyperparameters of interest are those that navigate the tradeoff between costs from heterogeneity and privacy (such as λ for MR-MTL), there are several factors that make auto-tuning particularly challenging in our setting:

- **A universal and practical measure of heterogeneity is generally unavailable**. For MR-MTL, our analysis in Section Section 1.6 suggests that the optimal λ^* depends on a measure of data heterogeneity across silos (τ^2). However, in practice, the level of heterogeneity is abstract and hard to quantify even with non-private oracle access to the silo datasets. Past work has devised custom measures of heterogeneity such as the variance of client (silo) model or gradient updates [83, 90], or the difference between the true global loss and the aggregate of the true local losses, but these measures may not be consistent throughout training [83, 90] such that they cannot provide a useful estimate of the heterogeneity which should be invariant during training. These metrics may also simply not be obtainable in practice [92]. The work of [132] considers the use of a client-relationship matrix, but such matrix may be too large (K^2 elements) and too noisy under privacy as we may need to directly or implicitly noise every coordinate of the matrix. Directly estimating the dataset statistics may also be challenging, since silos may apply custom dataset transformations (e.g. standardization or augmentation) that can drastically alter such statistics without affecting the learning dynamics.

- **The estimated heterogeneity may have high variance.** In addition to the above, heterogeneity would generally be measured on the client distribution (as some form of "variance" across clients, as in [83]), but there is generally a limited number of clients in cross-silo settings, meaning that this estimated "variance" as a measure of heterogeneity may itself have high variance. Moreoever, such measures must also be estimated privately, which means that it would have even higher variance from the additional noise. This could be problematic since the best hyperparameter for personalization may depend on such heterogeneity measure (as is the case for MR-MTL in simplified settings), and the precision of a potential auto-tuning precedure may suffer as a result.

- **Auto-tuning procedures may need to be developed specifically for each method.** For example, local finetuning may use the number of federated training rounds as the hyperparameter for determining "how much" to personalize, but such hyperparameter would likely be auto-tuned in a drastically different way compared to auto-tuning for λ for MR-MTL. It is in general an open question as to whether method-specific auto-tuning procedures will always outperform the simple randomized procedure described in Appendix E.1, in terms of both precision and privacy overhead.

Appendix G

Additional Experimental Results

G.1 Local Finetuning (Extension of Fig. 1.2)

In Fig. 1.2, we made the observation that local finetuning [147, 157, 32] as a personalization strategy (warm-starting from a global model and continue training using local data) may not always improve utility as expected. Extending on Fig. 1.2, we examine the behavior of local finetuning starting at different stages of training (under a fixed total number of rounds) as well as under varying privacy budgets in Fig. G.1. We observe that the phenomenon illustrated in Fig. 1.2 is indeed reflected across different settings: as soon as local finetuning begins, the DP utility gap can quickly widen, and the utility of finetuning under privacy can roughly reduce to the utility of local training.

Note, however, that these observations serve to give insights into an interesting behavior of local finetuning under silo-specific example-level DP and *do not* preclude the possibility that local finetuning can still outperform both local training and FedAvg. Indeed, local finetuning was observed to outperform local training in, e.g., Fig. 1.3 (d) and Fig. G.7.

G.2 Utility of MR-MTL as a Function of λ (Extension of Fig. 1.5)

We also extend on Fig. 1.5 to consider varying privacy budgets, and the results are shown in Fig. G.2 (Vehicle), Fig. G.3 (School), and Fig. G.4 (Heterogeneous CIFAR-10). Note that on School, the test metric is MSE thus lower is better. There are several notable observations:

- **λ^* decreases as ε grows (weaker privacy)**: With weaker privacy (larger ε), we have a smaller optimal λ^* (i.e. the location of the utility "bump" under DP gradually shifts to the left). This behavior is characterized by Theorem 1.6.3, where under stronger privacy, silos benefit from more federation as a means to reduce the effect of privacy noise.
- **MR-MTL does not always outperform FedAvg** (at high privacy regimes): One minor caveat is that Proposition 1.6.6 says that the utility gap from MR-MTL to FedAvg is always nonnegative under federated mean estimation; however, as discussed in Sections Section 1.5 and Section 1.6, MR-MTL does not approach FedAvg with larger λ in general learning settings since the MR-MTL objective may become too hard to solve via (DP-)SGD. Thus, despite its utility advantage

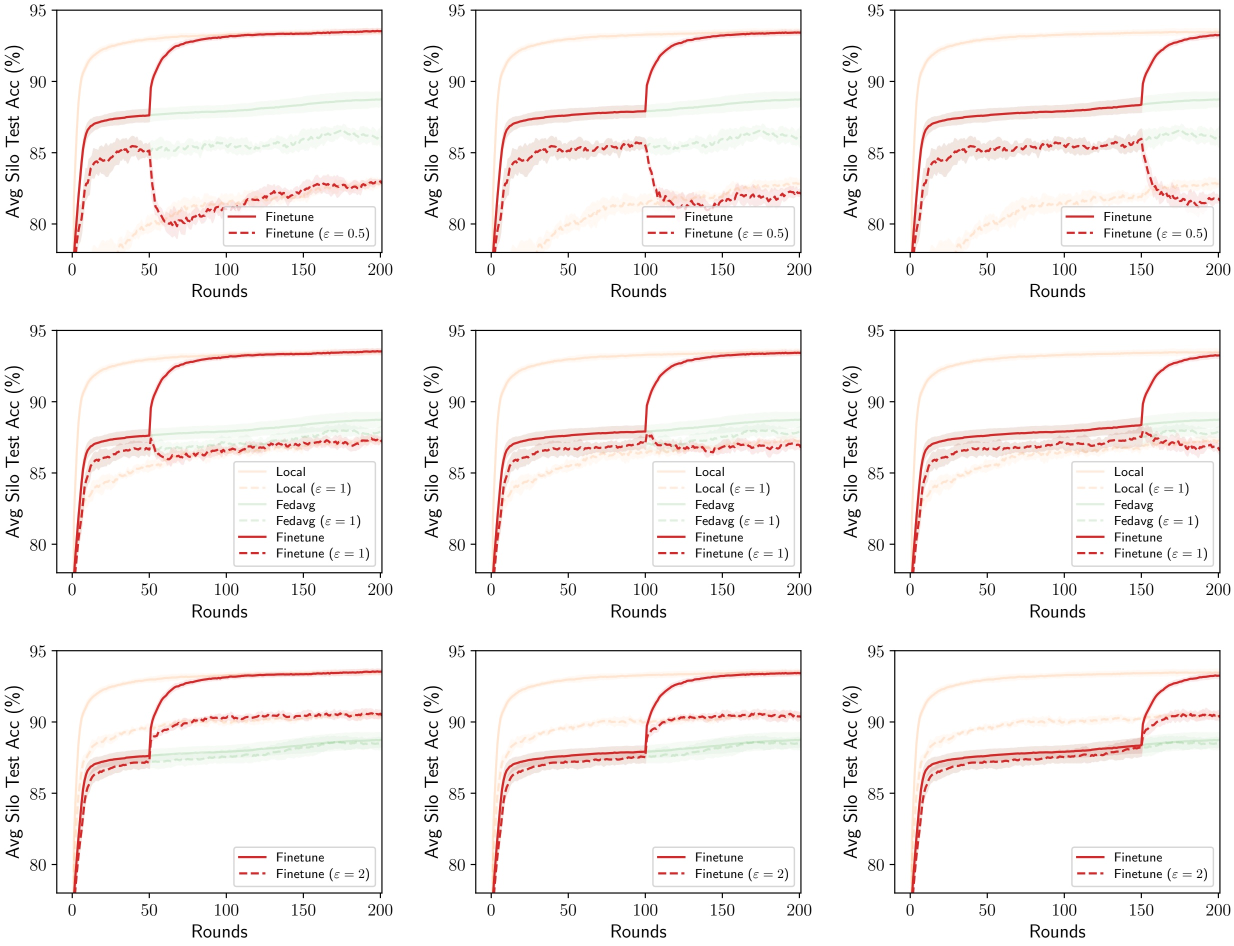

Figure G.1: (Extension of Fig. 1.2) **Behavior of local finetuning** starting at different stages of training (25%, 50%, 75%) and under varying privacy budgets ($\varepsilon \in [0.5, 1, 2], \delta = 10^{-7}$) for $T = 200$ rounds on the **Vehicle** dataset.

over local training at λ^*, it may never reach the performance of FedAvg. See, e.g., subplots of $\varepsilon \in [0.1, 0.2]$ in Fig. G.2.

- **Utility advantage of MR-MTL (λ^*) over local/FedAvg changes with ε:** Observe that with larger ε, the utility advantage of MR-MTL (λ^*) is larger compared to FedAvg and is smaller compared to local training. This behavior is characterized by Propositions 1.6.5 and 1.6.6.

Note also that for Heterogeneous CIFAR-10 (Fig. G.4), Ditto [91] exhibits a slightly different interpolation behavior compared to MR-MTL (e.g. it has larger optimal λ^* under privacy), though at λ^* its utility underperforms that of MR-MTL under privacy.

G.3 Subsampled ADNI Dataset

We further evaluate the suite of personalization methods on the subsampled ADNI dataset, and the results are shown in Fig. G.5. There are several notable observations:

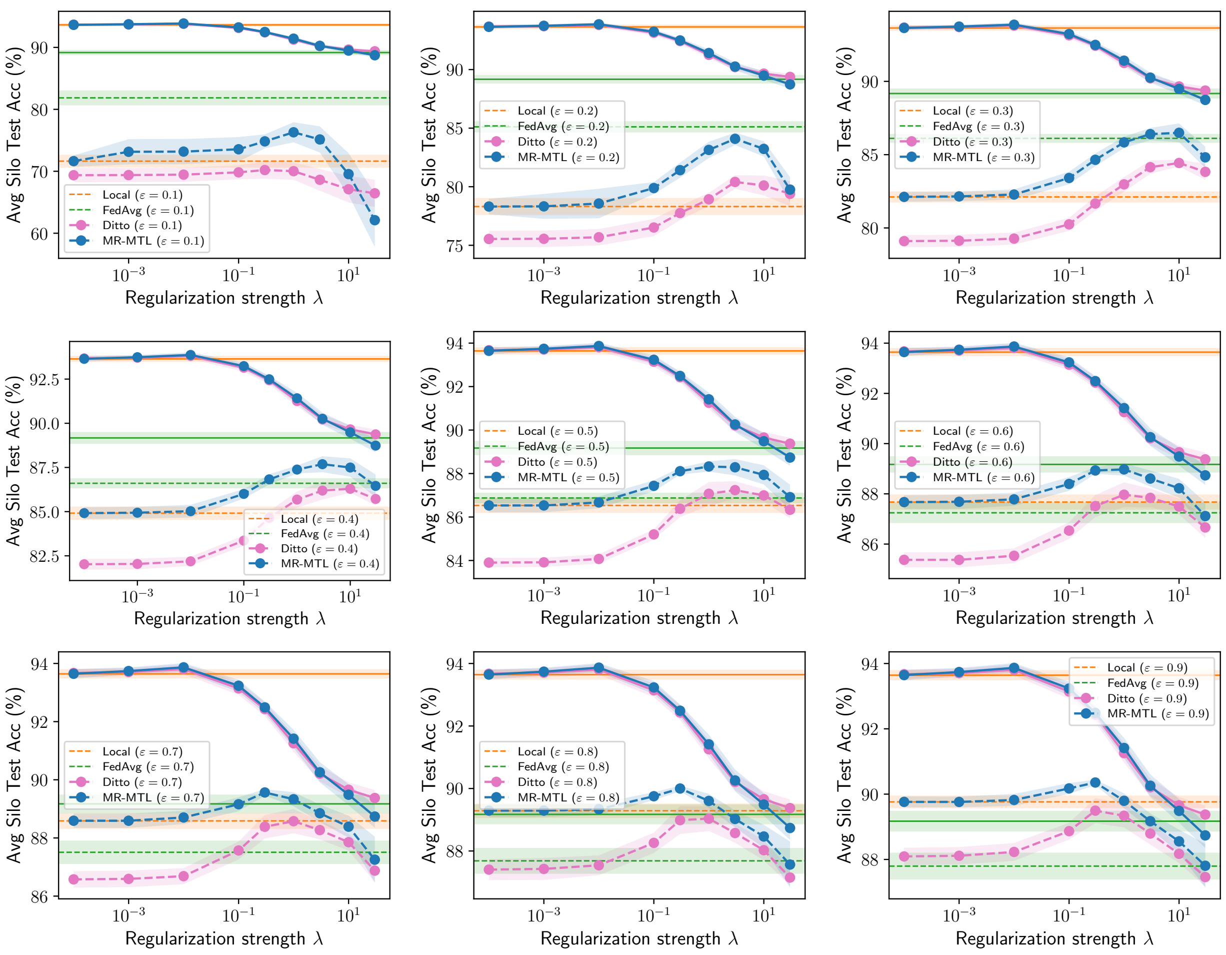

Figure G.2: (Extension of Fig. 1.5) **Behavior of MR-MTL as a function of λ across varying privacy budgets** ($\varepsilon \in [0.1, 0.2, ..., 0.9], \delta = 10^{-7}$) for $T = 400$ rounds on the **Vehicle** dataset. Solid lines refer to the non-private runs (same across all plots).

1. MR-MTL is essentially recovering local training due to the high degree of heterogeneity present in this dataset, with $\lambda^* \approx 0$ for most ε values except around $\varepsilon \approx 3.5$ (annotated with arrows in Fig. G.5 (a, b)).

2. Unlike results on other datasets, the privacy regime of interest where MR-MTL outperforms both local/FedAvg (around $\varepsilon \approx 4$) is extremely narrow, compared to $\varepsilon \approx 0.5$ for Vehicle (Fig. 1.3 (a)), $\varepsilon \approx 6$ for School (Fig. 1.3 (b)), and $\varepsilon \approx 1.5$ for GLEAM (Fig. 1.3 (c)). The utility advantage of MR-MTL is also insignificant.

3. The underlying heterogeneity structure of the dataset is rather pathological in that it is not ameanable to *both* mean-regularization and clustering. First, observe from Fig. G.5 (a) that MR-MTL still opts for a small λ^* in high-privacy regimes even when FedAvg performs better; one would expect MR-MTL to opt for a larger λ for lower DP noise (at a cost of higher heterogeneity). Second, from Fig. G.5 (b, c), we observe that both clustering and cluster-preconditoning did not lead to significant utility improvements, although the latter reduces the degree of heterogeneity and allows MR-MTL to opt for a larger λ^* in the privacy regime of interest.

In general, determining how to better model such heterogeneity (especially in high privacy regimes)

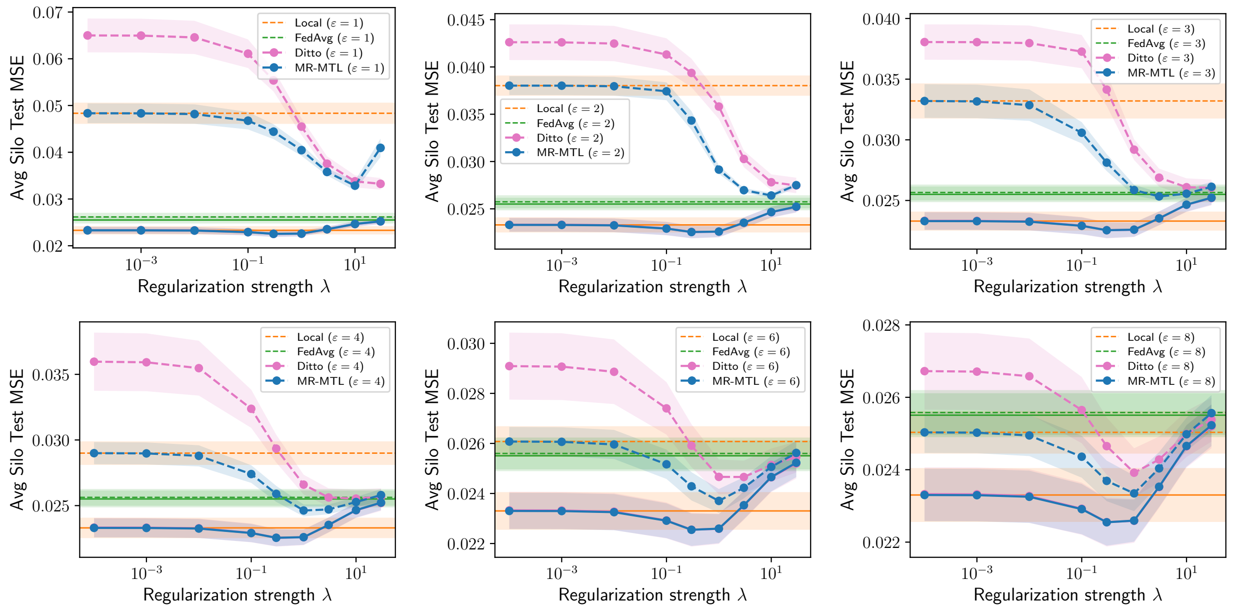

Figure G.3: (Extension of Fig. 1.5) **Behavior of MR-MTL as a function of λ across varying privacy budgets** ($\varepsilon \in [1, 2, 3, 4, 6, 8], \delta = 10^{-3}$) for $T = 200$ rounds on the **School** dataset. Cf. Fig. 1.3 (b) and Fig. G.7, where the privacy regime of interest is around $\varepsilon \approx 6$. Solid lines refer to the non-private runs (same across all plots).

is an interesting direction of future work.

G.4 Additional Discussions

Behavior on larger privacy budgets (Fig. G.6). We extend the results on Vehicle (Fig. 1.3 (a, b)) by considering larger privacy budgets, and the results are shown in Fig. G.6. We note that as privacy requirement loosens, personalized learning (where each silo maintains its own model) tend to perform better than the case where models are shared across clients (FedAvg and IFCA). This is expected as clients in cross-silo datasets tend to have sufficient data for learning a good local model. Moreover, this also means MR-MTL remains competitive as it can use a smaller λ.

Effect of dataset subsampling (Fig. G.7). Subsampling local datasets would in principle turn the cross-silo learning setting closer to a cross-device learning setting, in which local training becomes less attractive as due to insufficient local data and may opt for federated training despite data heterogeneity. In Fig. G.7, we examine this behavior on the School dataset by using different train/test split ratios of 80%/20%, 50%/50%, and 20%/80%. We observe that with less local training data: (1) FedAvg performs better since silos benefit from federation despite data heterogeneity; (2) the cross-over point between local training and FedAvg shifts to larger ε (or they may not be a cross-over point); and (3) MR-MTL may no longer provide an optimal point on the personalization spectrum, since small local datasets with silo-specific example-level DP necessitate a larger λ (cf. Theorem 1.6.3) but MR-MTL may not recover FedAvg under (DP-)SGD. In these cases, client-level DP protection (as is commonly used for cross-device FL) may be more appropriate.

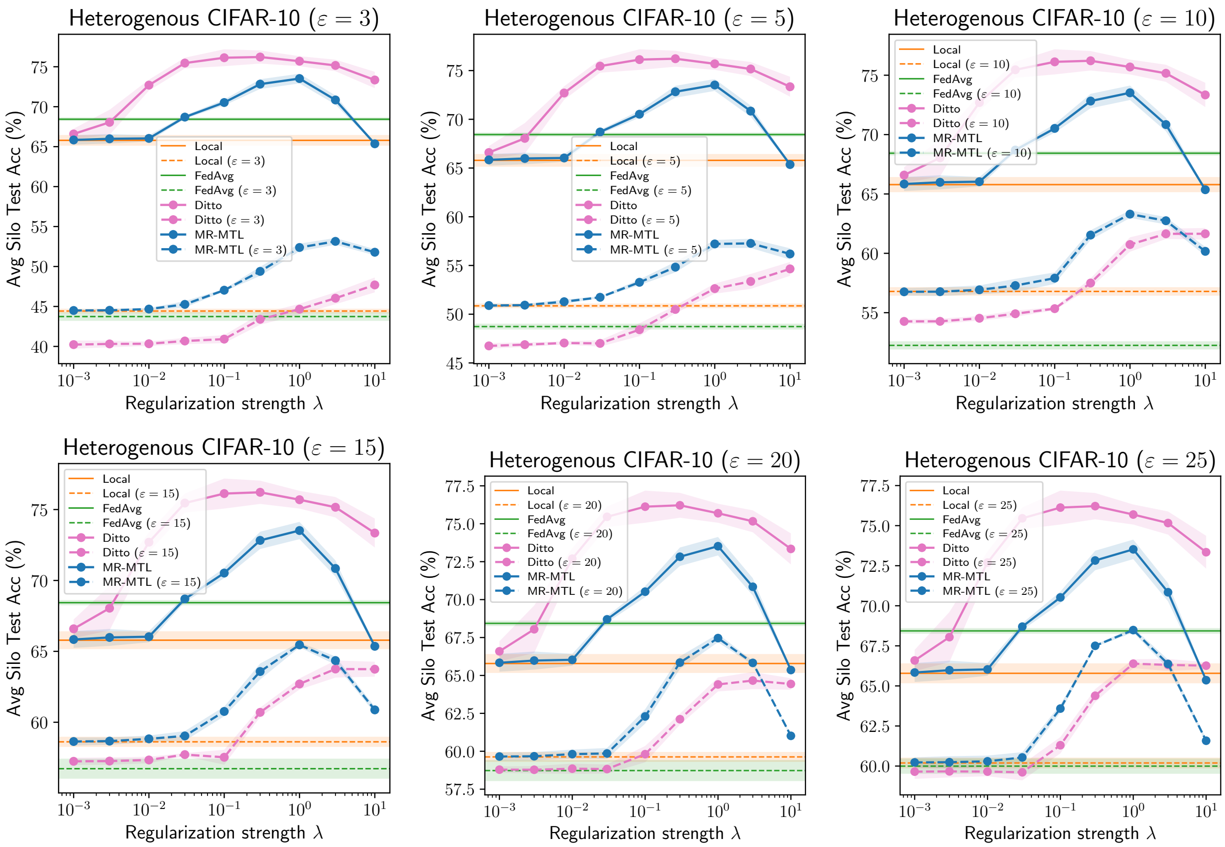

Figure G.4: (Extension of Fig. 1.5) **Behavior of MR-MTL as a function of λ across varying privacy budgets** ($\delta = 10^{-4}$) for $T = 200$ rounds on **Heterogeneous CIFAR-10**. Solid lines refer to non-private runs (same across all plots) and dashed/dotted lines refer to private runs (with silo-specific example-level DP).

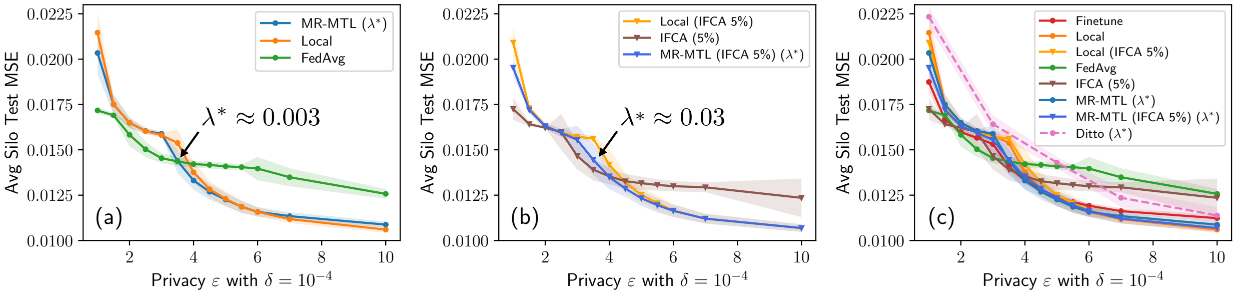

Figure G.5: **Test accuracy** (mean ± std with 3 seeds) vs **privacy budgets** ε for various personalization methods on the **subsampled ADNI** dataset with $T = 500$ rounds. The heterogeneity present in this dataset is unique in that it is not ameanable to both mean-regularization (a) and clustering (b, c), though clustering can mitigate the level of heterogeneity by allowing a larger λ^* for MR-MTL. In such cases of high heterogeneity, local performance tends to be superior for most privacy budgets, and we find that MR-MTL recovers this behavior. Determining how to better model such heterogeneity (especially in high privacy regimes) is an interesting direction of future work.

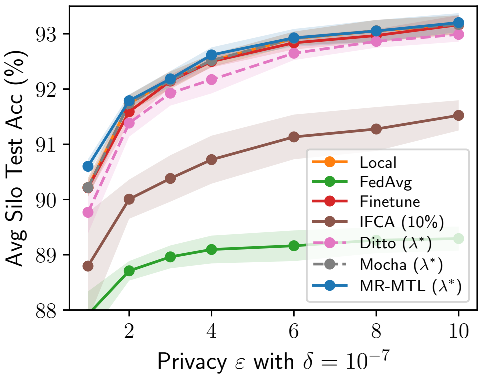

Figure G.6: (Extension of Fig. 1.3 (b)) Results on the **Vehicle** dataset for low privacy regimes ($1 \leq \varepsilon \leq 10$). As privacy becomes weaker, the need for federation diminishes and personalization methods (including local training) perform better. MR-MTL can recover local training by using a small λ.

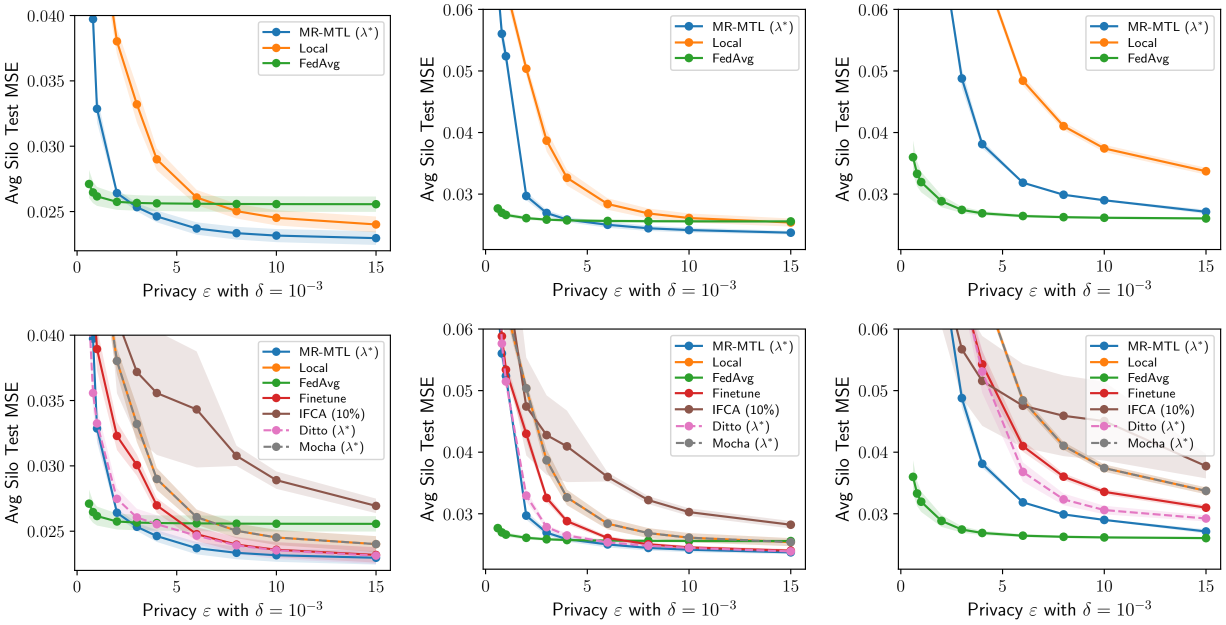

Figure G.7: **Test MSE vs total privacy budgets on the School dataset** (80%/20%, 50%/50%, and 20%/80% train/test split of the local dataset by columns from left to right). The top row compares MR-MTL to local training/FedAvg, which form the endpoints of the personalization spectrum with constant privacy costs, and the bottom row compares against other personalization methods. Under data subsampling (as little as 20% training data in the 3rd column), we obtain a setting closer to cross-device FL where FedAvg outperforms personalization since clients benefit from others' training data despite their heterogeneity.

Bibliography

[1] 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*, pages 308–318, 2016.

[2] The Alzheimer's Disease Neuroimaging Initiative (ADNI). The alzheimer's disease neuroimaging initiative (adni). adni.loni.usc.edu, 05 2022. https://adni.loni.usc.edu/.

[3] Alekh Agarwal, John Langford, and Chen-Yu Wei. Federated residual learning. *arXiv preprint arXiv:2003.12880*, 2020.

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

[5] Naman Agarwal, Peter Kairouz, and Ziyu Liu. The skellam mechanism for differentially private federated learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[6] Mohammad Alaggan, Sébastien Gambs, and Anne-Marie Kermarrec. Heterogeneous differential privacy. *Journal of Privacy and Confidentiality*, 7(2), 2016.

[7] Nasser Aldaghri, Hessam Mahdavifar, and Ahmad Beirami. Feo2: Federated learning with opt-out differential privacy. In *NeurIPS 2021 Workshop on New Frontiers in Federated Learning: Privacy, Fairness, Robustness, Personalization and Data Ownership*, 2021.

[8] Guozhong An. The effects of adding noise during backpropagation training on a generalization performance. *Neural computation*, 8(3):643–674, 1996.

[9] Galen Andrew, Om Thakkar, Brendan McMahan, and Swaroop Ramaswamy. Differentially private learning with adaptive clipping. *Advances in Neural Information Processing Systems*, 34, 2021.

[10] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine learning*, 73(3):243–272, 2008.

[11] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, 2020.

[12] Aviad Barzilai and Koby Crammer. Convex multi-task learning by clustering. In *Artificial Intelligence and Statistics*, pages 65–73. PMLR, 2015.

[13] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *IEEE Symposium on Foundations of Computer Science*, 2014.

[14] James Henry Bell, Kallista A Bonawitz, Adrià Gascón, Tancrède Lepoint, and Mariana Raykova. Secure single-server aggregation with (poly) logarithmic overhead. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 1253–1269, 2020.

[15] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.

[16] Alberto Bietti, Chen-Yu Wei, Miroslav Dudik, John Langford, and Zhiwei Steven Wu. Personalization improves privacy-accuracy tradeoffs in federated optimization. In *International Conference on Machine Learning*. PMLR, 2022.

[17] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems*, 2017.

[18] Jeremiah Blocki, Avrim Blum, Anupam Datta, and Or Sheffet. Differentially private data analysis of social networks via restricted sensitivity. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 87–96, 2013.

[19] Franziska Boenisch, Adam Dziedzic, Roei Schuster, Ali Shahin Shamsabadi, Ilia Shumailov, and Nicolas Papernot. When the curious abandon honesty: Federated learning is not private. *arXiv preprint arXiv:2112.02918*, 2021.

[20] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

[21] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

[22] Mark Bun and Thomas Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*, pages 635–658. Springer, 2016.

[23] Mark Bun, Damien Desfontaines, Cynthia Dwork, Moni Naor, Kobbi Nissim, Aaron Roth, Adam Smith, Thomas Steinke, Jonathan Ullman, and Salil Vadhan. Statistical inference is not a privacy violation. DifferentialPrivacy.org, 06 2021. https://differentialprivacy.org/inference-is-not-a-privacy-violation/.

[24] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. In *NeurIPS 2019 Workshop on Federated Learning for Data Privacy and Confidentiality*, 2019.

[25] Clément L Canonne, Gautam Kamath, and Thomas Steinke. The discrete gaussian for differential privacy. *Advances in Neural Information Processing Systems*, 33:15676–15688, 2020.

[26] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *USENIX Security Symposium*, volume 267, 2019.

[27] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[28] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914. IEEE, 2022.

[29] Sílvia Casacuberta, Michael Shoemate, Salil Vadhan, and Connor Wagaman. Widespread underestimation of sensitivity in differentially private libraries and how to fix it. *arXiv preprint arXiv:2207.10635*, 2022.

[30] Chen Chen, Jaewoo Lee, and Dan Kifer. Renyi differentially private erm for smooth objectives. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2037–2046. PMLR, 2019.

[31] Wei-Ning Chen, Ayfer Ozgur, and Peter Kairouz. The poisson binomial mechanism for unbiased federated learning with secure aggregation. In *International Conference on Machine Learning*, pages 3490–3506. PMLR, 2022.

[32] Gary Cheng, Karan Chadha, and John Duchi. Fine-tuning is fine in federated learning. *arXiv preprint arXiv:2108.07313*, 2021.

[33] Albert Cheu. Differential privacy in the shuffle model: A survey of separations. *arXiv preprint arXiv:2107.11839*, 2021.

[34] Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. Distributed differential privacy via shuffling. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 375–403. Springer, 2019.

[35] Yae Jee Cho, Jianyu Wang, Tarun Chiruvolu, and Gauri Joshi. Personalized federated learning for heterogeneous clients with clustered knowledge transfer. *arXiv preprint arXiv:2109.08119*, 2021.

[36] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[37] Ameya Daigavane, Gagan Madan, Aditya Sinha, Abhradeep Guha Thakurta, Gaurav Aggarwal, and Prateek Jain. Node-level differentially private graph neural networks. *arXiv preprint arXiv:2111.15521*, 2021.

[38] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020.

[39] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 202–210, 2003.

[40] Jinshuo Dong, Aaron Roth, and Weijie J. Su. Gaussian differential privacy. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 84(1):3–37, 2022. doi: https://doi.org/10.1111/rssb.12454. URL https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12454.

[41] John R Douceur. The sybil attack. In *International workshop on peer-to-peer systems*, pages 251–260. Springer, 2002.

[42] Marco F Duarte and Yu Hen Hu. Vehicle classification in distributed sensor networks. *Journal of Parallel and Distributed Computing*, 64(7):826–838, 2004.

[43] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[44] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[45] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2468–2479. SIAM, 2019.

[46] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 109–117, 2004.

[47] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in Neural Information Processing Systems*, 33:3557–3568, 2020.

[48] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to {Byzantine-Robust} federated learning. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1605–1622, 2020.

[49] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. Privacy amplification by iteration. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 521–532. IEEE, 2018.

[50] Tiantian Feng, Raghuveer Peri, and Shrikanth Narayanan. User-level differential privacy against attribute inference attack of speech emotion recognition in federated learning. *arXiv preprint arXiv:2204.02500*, 2022.

[51] Laurent Fousse, Guillaume Hanrot, Vincent Lefèvre, Patrick Pélissier, and Paul Zimmermann. Mpfr: A multiple-precision binary floating-point library with correct rounding. *ACM Transactions on Mathematical Software (TOMS)*, 33(2):13–es, 2007.

[52] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333, 2015.

[53] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An {End-to-End} case study of personalized warfarin dosing. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 17–32, 2014.

[54] Craig Gentry. *A fully homomorphic encryption scheme*. Stanford university, 2009.

[55] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. In *NIPS 2017 Workshop: Machine Learning on the Phone and other Consumer Devices*, 2017.

[56] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems*, 33: 19586–19597, 2020.

[57] Antonious Girgis, Deepesh Data, Suhas Diggavi, Peter Kairouz, and Ananda Theertha Suresh. Shuffled model of differential privacy in federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2521–2529. PMLR, 2021.

[58] Harvey Goldstein. Multilevel modelling of survey data. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 40(2):235–244, 1991.

[59] Pinghua Gong, Jieping Ye, and Changshui Zhang. Robust multi-task feature learning. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 895–903, 2012.

[60] Google. TensorFlow Privacy. https://github.com/tensorflow/privacy, 2022.

[61] Google Differential Privacy Team. Secure Noise Generation. https://github.com/google/differential-privacy/blob/main/common_docs/Secure_Noise_Generation.pdf, 2020.

[62] Samuel Haney, Damien Desfontaines, Luke Hartman, Ruchit Shrestha, and Michael Hay. Precision-based attacks and interval refining: how to break, then fix, differential privacy on finite computers. *arXiv preprint arXiv:2207.13793*, 2022.

[63] Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516*, 2020.

[64] Filip Hanzely, Slavomír Hanzely, Samuel Horváth, and Peter Richtárik. Lower bounds and optimal algorithms for personalized federated learning. *Advances in Neural Information Processing Systems*, 33:2304–2315, 2020.

[65] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

[66] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2022.

[67] Mikko A Heikkilä, Antti Koskela, Kana Shimizu, Samuel Kaski, and Antti Honkela. Differentially private cross-silo federated learning. In *Privacy Preserving Machine Learning (PPML) and Privacy in Machine learning (PriML) Joint Workshop at NeurIPS 2020*, 2020.

[68] Tom Hennigan, Trevor Cai, Tamara Norman, and Igor Babuschkin. Haiku: Sonnet for JAX, 2020. URL `http://github.com/deepmind/dm-haiku`.

[69] Naoise Holohan and Stefano Braghin. Secure random sampling in differential privacy. In *European Symposium on Research in Computer Security*, pages 523–542. Springer, 2021.

[70] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. In *Workshop on Federated Learning for Data Privacy and Confidentiality, NeurIPS 2019*, 2019.

[71] Shengyuan Hu, Zhiwei Steven Wu, and Virginia Smith. Private multi-task learning: Formulation and applications to federated learning. *arXiv preprint arXiv:2108.12978*, 2021.

[72] Christina Ilvento. Implementing the exponential mechanism with base-2 differential privacy. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 717–742, 2020.

[73] Matthew Jagielski, Jonathan Ullman, and Alina Oprea. Auditing differentially private machine learning: How private is private sgd? *Advances in Neural Information Processing Systems*, 33:22205–22216, 2020.

[74] Ali Jalali, Sujay Sanghavi, Chao Ruan, and Pradeep Ravikumar. A dirty model for multi-task learning. *Advances in neural information processing systems*, 23, 2010.

[75] Bargav Jayaraman and David Evans. Evaluating differentially private machine learning in practice. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 1895–1912, 2019.

[76] Yihan Jiang, Jakub Konečný, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

[77] Jiankai Jin, Eleanor McMurtry, Benjamin IP Rubinstein, and Olga Ohrimenko. Are we there yet? timing and floating-point attacks on differential privacy systems. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 473–488. IEEE, 2022.

[78] Zach Jorgensen, Ting Yu, and Graham Cormode. Conservative or liberal? personalized differential privacy. In *2015 IEEE 31St international conference on data engineering*, pages 1023–1034. IEEE, 2015.

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

[80] 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*, pages 5213–5225. PMLR, 2021.

[81] 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, 2021.

[82] Pallika Kanani, Virendra J Marathe, Daniel Peterson, Rave Harpaz, and Steve Bright. Private cross-silo federated learning for extracting vaccine adverse event mentions. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 490–505. Springer, 2021.

[83] 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*, pages 5132–5143. PMLR, 2020.

[84] Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Analyzing graphs with node differential privacy. In *Theory of Cryptography Conference*, pages 457–476. Springer, 2013.

[85] A Krizhevsky. Learning multiple layers of features from tiny images. *Master's thesis, University of Toronto*, 2009.

[86] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[87] Daniel Levy, Ziteng Sun, Kareem Amin, Satyen Kale, Alex Kulesza, Mehryar Mohri, and Ananda Theertha Suresh. Learning with user-level privacy. *Advances in Neural Information Processing Systems*, 34, 2021.

[88] Jeffrey Li, Mikhail Khodak, Sebastian Caldas, and Ameet Talwalkar. Differentially private meta-learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rJgqMRVYvr.

[89] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[90] 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.

[91] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, pages 6357–6368. PMLR, 2021.

[92] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJxNAnVtDS.

[93] Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B Allen, Randy P Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. In *NeurIPS 2019 Workshop on Federated Learning*, 2020.

[94] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *International Conference on Computer Vision*, 2017.

[95] Jingcheng Liu and Kunal Talwar. Private selection from private candidates. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 298–309, 2019.

[96] Junxu Liu, Jian Lou, Li Xiong, Jinfei Liu, and Xiaofeng Meng. Projected federated averaging with heterogeneous differential privacy. *Proceedings of the VLDB Endowment*, 15(4):828–840, 2021.

[97] Junxu Liu, Jian Lou, Li Xiong, Jinfei Liu, and Xiaofeng Meng. Projected federated averaging with heterogeneous differential privacy. In *International Conference on Very Large Databases.* VLDB Endowment, 2022.

[98] Ken Liu, Shengyuan Hu, Steven Z Wu, and Virginia Smith. On privacy and personalization in cross-silo federated learning. *Advances in Neural Information Processing Systems*, 35:5925–5940, 2022.

[99] Sulin Liu, Sinno Jialin Pan, and Qirong Ho. Distributed multi-task relationship learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 937–946, 2017.

[100] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022.

[101] Andrew Lowy and Meisam Razaviyayn. Private federated learning without a trusted server: Optimal algorithms for convex losses. *arXiv preprint arXiv:2106.09779*, 2021.

[102] Linpeng Lu and Ning Ding. Multi-party private set intersection in vertical federated learning. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 707–714. IEEE, 2020.

[103] Edward Lui and Rafael Pass. Outlier privacy. In *Theory of Cryptography Conference*, pages 277–305. Springer, 2015.

[104] Samuel Maddock, Graham Cormode, Tianhao Wang, Carsten Maple, and Somesh Jha. Federated boosted decision trees with differential privacy. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2249–2263, 2022.

[105] Hessam Mahdavifar, Ahmad Beirami, Behrouz Touri, and Jeff S Shamma. Global games with noisy information sharing. *IEEE Transactions on Signal and Information Processing over Networks*, 4(3):497–509, 2017.

[106] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

[107] Ryan McKenna and Daniel R Sheldon. Permute-and-flip: A new mechanism for differentially private selection. *Advances in Neural Information Processing Systems*, 33:193–203, 2020.

[108] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[109] H Brendan McMahan, Galen Andrew, Ulfar Erlingsson, Steve Chien, Ilya Mironov, Nicolas Papernot, and Peter Kairouz. A general approach to adding differential privacy to iterative training procedures. In *NeurIPS 2018 Privacy Preserving Machine Learning Workshop*, 2018.

[110] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BJ0hF1Z0b.

[111] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 94–103. IEEE, 2007.

[112] Frank D McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 19–30, 2009.

[113] Ilya Mironov. On significance of the least significant bits for differential privacy. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 650–661, 2012.

[114] Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th computer security foundations symposium (CSF)*, pages 263–275. IEEE, 2017.

[115] Ilya Mironov, Kunal Talwar, and Li Zhang. Rényi differential privacy of the sampled gaussian mechanism. *arXiv preprint arXiv:1908.10530*, 2019.

[116] Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

[117] Nicolas Papernot and Thomas Steinke. Hyperparameter tuning with renyi differential privacy. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=-70L8lpp9DF.

[118] Venkatadheeraj Pichapati, Ananda Theertha Suresh, Felix X Yu, Sashank J Reddi, and Sanjiv Kumar. Adaclip: Adaptive clipping for private sgd. *arXiv preprint arXiv:1908.07643*, 2019.

[119] Krishna Pillutla, Sham M Kakade, and Zaid Harchaoui. Robust aggregation for federated learning. *arXiv preprint arXiv:1912.13445*, 2019.

[120] Liangqiong Qu, Niranjan Balachandar, and Daniel L Rubin. An experimental study of data heterogeneity in federated learning methods for medical imaging. *arXiv preprint arXiv:2107.08371*, 2021.

[121] Shah Atiqur Rahman, Christopher Merck, Yuxiao Huang, and Samantha Kleinberg. Unintrusive eating recognition using google glass. In *2015 9th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth)*, pages 108–111. IEEE, 2015.

[122] Swaroop Ramaswamy, Om Thakkar, Rajiv Mathews, Galen Andrew, H Brendan McMahan, and Françoise Beaufays. Training production language models without memorizing user data. *arXiv preprint arXiv:2009.10031*, 2020.

[123] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. Technical report, OpenAI, 2022.

[124] Sofya Raskhodnikova and Adam Smith. Lipschitz extensions for node-private graph statistics and the generalized exponential mechanism. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 495–504. IEEE, 2016.

[125] Sashank J Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2021.

[126] F Reith, ME Koran, G Davidzon, and G Zaharchuk. Application of deep learning to predict standardized uptake value ratio and amyloid status on 18f-florbetapir pet using adni data. *American Journal of Neuroradiology*, 41(6):980–986, 2020.

[127] Ryan Rogers and Thomas Steinke. A better privacy analysis of the exponential mechanism. DifferentialPrivacy.org, 07 2021. https://differentialprivacy.org/exponential-mechanism-bounded-range/.

[128] Thorsteinn Rögnvaldsson. On langevin updating in multilayer perceptrons. *Neural computation*, 6(5):916–926, 1994.

[129] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE transactions on neural networks and learning systems*, 32(8):3710–3722, 2020.

[130] Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. Personalized federated learning using hypernetworks. In *International Conference on Machine Learning*, pages 9489–9502. PMLR, 2021.

[131] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[132] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. *Advances in neural information processing systems*, 30, 2017.

[133] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. Stochastic gradient descent with differentially private updates. In *IEEE Global Conference on Signal and Information Processing*, 2013.

[134] Pranav Subramani, Nicholas Vadivelu, and Gautam Kamath. Enabling fast differentially private sgd via just-in-time compilation and vectorization. *Advances in Neural Information Processing Systems*, 34, 2021.

[135] Ziteng Sun, Peter Kairouz, A. T. Suresh, and H. Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.

[136] Canh T Dinh, Nguyen Tran, and Josh Nguyen. Personalized federated learning with moreau envelopes. *Advances in Neural Information Processing Systems*, 33:21394–21405, 2020.

[137] The OpenDP Team. The OpenDP White Paper. https://projects.iq.harvard.edu/files/opendp/files/opendp_white_paper_11may2020.pdf, 2020.

[138] The OpenDP Team. SmartNoise Core Differential Privacy Library. https://github.com/opendp/smartnoise-core, 2022.

[139] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data poisoning attacks against federated learning systems. In *European Symposium on Research in Computer Security*, pages 480–501. Springer, 2020.

[140] Florian Tramèr, Reza Shokri, Ayrton San Joaquin, Hoang Le, Matthew Jagielski, Sanghyun Hong, and Nicholas Carlini. Truth serum: Poisoning machine learning models to reveal their secrets. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2779–2792, 2022.

[141] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM workshop on artificial intelligence and security*, pages 1–11, 2019.

[142] Akhil Vaid, Suraj K Jaladanki, Jie Xu, Shelly Teng, Arvind Kumar, Samuel Lee, Sulaiman Somani, Ishan Paranjpe, Jessica K De Freitas, Tingyi Wanyan, Kipp W Johnson, Mesude Bicak, Eyal Klang, Young Joon Kwon, Anthony Costa, Shan Zhao, Riccardo Miotto, Alexander W Charney, Erwin Böttinger, Zahi A Fayad, Girish N Nadkarni, Fei Wang, and Benjamin S Glicksberg. Federated learning of electronic health records improves mortality prediction in patients hospitalized with covid-19. *medRxiv*, 2020. doi: 10.1101/2020.08.11.20172809. URL https://www.medrxiv.org/content/early/2020/08/14/2020.08.11.20172809.

[143] Koen Lennart van der Veen, Ruben Seggers, Peter Bloem, and Giorgio Patrini. Three tools for practical differential privacy. In *Privacy Preserving Machine Learning (PPML) Workshop at NeurIPS 2018*, 2018.

[144] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[145] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning. *Advances in Neural Information Processing Systems*, 33:16070–16084, 2020.

[146] 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.

[147] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. Federated evaluation of on-device personalization. *arXiv preprint arXiv:1910.10252*, 2019.

[148] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011.

[149] Yuxin Wen, Jonas Geiping, Liam Fowl, Micah Goldblum, and Tom Goldstein. Fishing for user data in large-batch federated learning via gradient magnification. *arXiv preprint arXiv:2202.00580*, 2022.

[150] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR, 2019.

[151] Xi Wu, Matthew Fredrikson, Somesh Jha, and Jeffrey F Naughton. A methodology for formalizing model-inversion attacks. In *2016 IEEE 29th Computer Security Foundations Symposium (CSF)*, pages 355–370. IEEE, 2016.

[152] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. Dba: Distributed backdoor attacks against federated learning. In *International Conference on Learning Representations*, 2019.

[153] Jiayuan Ye and Reza Shokri. Differentially private learning needs hidden state (or much faster convergence). *arXiv preprint arXiv:2203.05363*, 2022.

[154] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. See through gradients: Image batch recovery via gradinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16337–16346, 2021.

[155] Da Yu, Gautam Kamath, Janardhan Kulkarni, Jian Yin, Tie-Yan Liu, and Huishuai Zhang. Per-instance privacy accounting for differentially private stochastic gradient descent. *arXiv preprint arXiv:2206.02617*, 2022.

[156] Lei Yu, Ling Liu, Calton Pu, Mehmet Emre Gursoy, and Stacey Truex. Differentially private model publishing for deep learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 332–349. IEEE, 2019.

[157] Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. Salvaging federated learning by local adaptation. *arXiv preprint arXiv:2002.04758*, 2020.

[158] Chengliang Zhang, Suyi Li, Junzhe Xia, Wei Wang, Feng Yan, and Yang Liu. {BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning. In *2020 USENIX annual technical conference (USENIX ATC 20)*, pages 493–506, 2020.

[159] Qiuchen Zhang, Jing Ma, Yonghui Xiao, Jian Lou, and Li Xiong. Broadening differential privacy for deep learning against model inversion attacks. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 1061–1070. IEEE, 2020.

[160] Sixin Zhang, Anna E Choromanska, and Yann LeCun. Deep learning with elastic averaging sgd. *Advances in neural information processing systems*, 28, 2015.

[161] Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[162] Han Zhao, Otilia Stretcu, Alexander J Smola, and Geoffrey J Gordon. Efficient multitask feature and relationship learning. In *Uncertainty in Artificial Intelligence*, pages 777–787. PMLR, 2020.

[163] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[164] Qinqing Zheng, Shuxiao Chen, Qi Long, and Weijie Su. Federated f-differential privacy. In *International Conference on Artificial Intelligence and Statistics*, pages 2251–2259. PMLR, 2021.

[165] Jiayu Zhou, Jianhui Chen, and Jieping Ye. Clustered multi-task learning via alternating structure optimization. *Advances in neural information processing systems*, 24, 2011.

[166] Jiayu Zhou, Jianhui Chen, and Jieping Ye. Malsar: Multi-task learning via structural regularization. *Arizona State University*, 21:1–50, 2011.