Formal Privacy Guarantee in Predictive Autoscaling by Differentially Private Federated Training

Alan Chuang
Department of Computer Science
San Jose State University
San Jose, CA, USA
Email: alan.chuang@sjsu.edu

Melody Moh
Department of Computer Science
San Jose State University
San Jose, CA, USA
Email: melody.moh@sjsu.edu

Teng-Sheng Moh
Department of Computer Science
San Jose State University
San Jose, CA, USA
Email: teng.moh@sjsu.edu

***Abstract*—Modern resource management in networks and cloud systems increasingly relies on machine learning. Predictive autoscaling often centralizes fine-grained logs, exposing membership and traffic-profile risks. We study training-time privacy and robustness for autoscaling forecasters without raw-log sharing by combining client-side differentially private stochastic gradient descent (DP-SGD) with federated aggregation. We report composed privacy at the end of training ($\varepsilon = 13.76$, $\delta = 10^{-5}$) and at the best-validation checkpoint ($\varepsilon = 9.69$, $\delta = 10^{-5}$). We audit membership inference using loss-threshold and confidence attacks, analyze heterogeneity with shard-size weighting, shard filtering, and advanced baselines (FedProx, SCAFFOLD), and quantify robustness to adversaries, including $\rho = 0.3$ Byzantine clients (median, trimmed mean, Krum) and multiple malicious-aggregator variants. Runtime privacy is addressed in a companion systems paper [1]. On GCT v3, a DP-LSTM remains operationally useful at moderate privacy budgets, reducing membership-inference advantage to about 2–3% on machines excluded from training.**

***Index Terms*—Federated Learning, Differential Privacy, Membership Inference, Heterogeneity, Autoscaling**

I. Introduction

Most existing machine learning (ML) models used in networked and cloud systems optimize prediction accuracy, but very few have simultaneously addressed training-time privacy risks and robustness, especially when supporting heterogeneous clients. We study training-time privacy and robustness for autoscaling forecasters without raw-log sharing. Our contributions are: (i) a modular client-side DP-SGD plus federated aggregation pipeline, (ii) complete reporting of composed (ε, δ) at best-validation and at the full run, (iii) calibrated membership-inference audits, and (iv) heterogeneity experiments with shard-size weighting and shard filtering, plus comparisons to FedProx and SCAFFOLD under identical DP budgets. Runtime privacy controls are out of scope, but a companion systems paper details runtime privacy mechanisms and serving-time performance [1].

Predictive autoscaling is widely used in ML models for resource management in networked and cloud systems. It uses short-horizon forecasts to scale resources before demand arrives. Forecasting often uses time-series models such as Long Short-Term Memory (LSTM) networks [2] or Autoregressive Integrated Moving Average (ARIMA) [3], alongside practical baselines like Prophet [4] and domain-specific variants [5]–[7].

While accuracy has improved, the *training pipeline* behind many systems still centralizes fine-grained usage logs (CPU, memory, I/O, network) to build models. Centralization simplifies engineering, but it also creates training-time privacy exposure and weakens robustness when tenants contribute uneven amounts of data.

This paper focuses on the *training* side of the problem. We assume no raw-log sharing and an honest-but-curious aggregator: clients keep logs local and exchange only model updates (Section III-A). Under this setting, two intertwined challenges arise:

1) **Utility under Differential Privacy (DP) noise.** Differentially Private Stochastic Gradient Descent (DP-SGD) should reduce membership-inference advantage without collapsing short-horizon forecast accuracy.
2) **Federated heterogeneity.** Aggregation must remain stable when client shards differ in size or participation, avoiding the degradation seen with naive FedAvg.

The remainder of this paper details the training-time threat model, methods, and results, and reports reproducible artifacts (DP hyperparameters and per-round accountant logs) to support auditing of (ε, δ).

II. Related Work

This paper aims to train an autoscaling forecaster with formal privacy guarantees that resists membership inference and remains robust under client heterogeneity, without sharing raw logs. In this section, we review previous works based on which this study may build a privacy-preserving training pipeline for predictive autoscaling. The following topics are included: differentially private training, federated learning under heterogeneity, and membership-inference attacks, highlighting their strengths and integration gaps.

A. Privacy in Training: Differential Privacy and Federated Learning

Differential privacy (DP) has become a standard for protecting sensitive training data, particularly in healthcare and finance. Techniques such as DP-SGD provide formal (ε, δ) guarantees by adding calibrated noise to gradients during training [8]. We use Rényi Differential Privacy (RDP) for composition [9] and its subsampled/analytical accountant [10].

Frameworks like Opacus simplify the integration of DP into deep learning applications [11].

Federated Learning (FL) decentralizes model training by aggregating updates from clients without collecting raw data. Early approaches like FedAvg [12] assumed homogeneous clients; surveys discuss advances and open problems [13], [14]. Under heterogeneity, methods such as FedProx and SCAFFOLD add stability to aggregation and drift [15], [16]. However, prior work rarely evaluates FL under shard imbalance *together with* formal DP accounting and explicit membership-inference tests on autoscaling workloads. Our focus is this training-time gap; deployment concerns are deferred to a companion systems paper.

Recent studies on DP-FL for time-series forecasting report comparable performance and overhead [17]–[19]. Huang et al. [17] applied shuffled local differential privacy to IoT time-series models, achieving moderate privacy at about a $6\times$ training-time overhead. Rahmani et al. [18] explored DP-SGD for cloud workload scaling, reporting roughly $8\times$ slowdown relative to non-DP training. Kiani et al. [19] proposed adaptive privacy spending in FL with 7–$9\times$ per-epoch cost increases. Our framework shows an $8.3\times$ per-epoch overhead, well within this range, while achieving similar or lower RMSE under equivalent ε budgets.

B. Positioning of This Work

Existing research provides effective pieces: DP-SGD for training privacy and FL for collaborative learning without raw logs. However, these works rarely evaluate training privacy together with robustness to client heterogeneity and explicit membership-inference tests for autoscaling workloads. Our scope for this paper is training-time privacy.

We address the *training-time* gap by combining:

- Differentially private federated LSTM training
- Mitigations for client heterogeneity (weighted aggregation, shard filtering)
- Empirical membership-inference evaluation with calibrated attacks and CIs

This training-time focus fills a gap in the autoscaling literature, which seldom reports privacy accounting, MI resistance, and federated heterogeneity results together on a common workload.

III. System Architecture

This is a modular privacy-preserving *training* pipeline for predictive autoscaling. It consists of client-side DP-SGD, a federated aggregator, and evaluation harnesses for heterogeneity and membership inference.

A. Threat Model and Adversaries

Assets and trust. Tenants keep raw logs local; the aggregator receives only model updates.

Adversary cases. (A) Honest-but-curious aggregator that inspects updates. (B) Malicious aggregator that may perturb or poison the global model or collude with clients. (C) Byzantine clients where a fraction ρ sends arbitrary or sign-flipped updates.

Scope. Experiments evaluate Case A with client-side DP-SGD and quantify robustness for Cases B and C at small scale: ρ=0.3 Byzantine clients that send sign-flipped updates and several malicious-aggregator variants. We evaluate coordinate-wise median, trimmed mean (20%), and Krum (f=2) under the same DP budget.

Centralized vs secure aggregation. We use a centralized aggregator to isolate learning effects; secure aggregation is not implemented here but is assumed for deployment, where it hides individual client updates without affecting (ε, δ) accounting [20].

TABLE I: Training-time threats and defenses with evaluation signals.

Threat	Defense	Signal
Membership inference	Client-side DP-SGD with RDP accounting	(ε, δ), MI AUROC/ASR
Client heterogeneity	Shard-size weighting, shard filtering	RMSE/MAE vs naive FedAvg
Byzantine clients	Median, trimmed mean (20%), Krum (f=2)	RMSE/MAE under ρ=0.3
Malicious aggregator	Deviation detection, rollback	RMSE/MAE after perturbation round
Data leakage	No raw-log sharing, by-machine splits	Split audit

- *Membership inference:* DP-SGD with per-sample clipping and calibrated Gaussian noise; we report MI AUROC/ASR with Confidence Intervals (CIs).
- *Heterogeneity:* Shard-size weighting and shard filtering; we report RMSE/MAE vs naive FedAvg.

B. Training and Aggregation

Each client trains a two-layer LSTM (64 units, dropout p=0.5) with DP-SGD via Opacus, using per-sample clipping and Gaussian noise. The aggregator computes FedAvg with shard-size weighting and optional shard filtering to drop under-populated clients.

Baseline tuning protocol: We tuned a non-private LSTM on the same splits and preprocessing via a grid over learning rate $\{10^{-4}, 3\times10^{-4}, 10^{-3}\}$, hidden size $\{32, 64, 128\}$, dropout $\{0.3, 0.5\}$, and batch size $\{32, 64, 128\}$, with early stopping and 20 seeds. Baseline tables report point estimates on the shared split.

All baselines, private and non-private, share the exact preprocessing, train/val/test splits, and early-stopping protocol. Additionally, the tuned non-private LSTM remains the strongest baseline under identical splits and protocol (Table VIII).

Robustness to aggregation attacks: We evaluate with n=10 and 3 Byzantine sign-flip clients (ρ=0.3). Krum uses f=2, satisfying $n \geq 2f+3$. The trimmed mean uses a 20% trim rate. Under this attack, the median reduces error relative to the mean, the trimmed mean lies between them, and Krum is strongest but still worse than the clean baseline (Table VI).

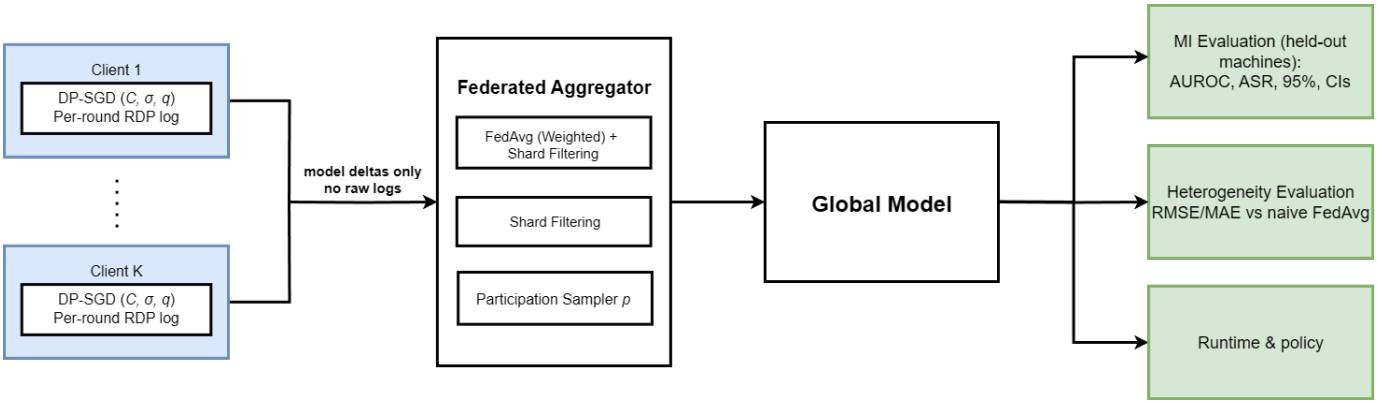

Fig. 1: Privacy-preserving *training* pipeline. Clients train locally with DP-SGD and send updates to a centralized aggregator. Robustness is evaluated via membership-inference audits and heterogeneity experiments (weighted FedAvg, shard filtering, partial participation; advanced baselines FedProx and SCAFFOLD).

For malicious aggregators, the generic perturbation is most damaging, while scale+noise and targeted FC perturbations are milder but still reduce utility.

IV. Methodology

A. Data Preprocessing

Workload logs consisting of CPU and memory utilization were collected in an append-only, timestamped format. Logs were first cleaned to remove missing or malformed records, then normalized on a per-client basis.

Since raw memory usage exhibited heavy skew, we applied a stabilized log transform $y' = \log(1+y)$ to compress dynamic range and avoid zeros. Features were scaled using per-machine z-scores. To capture temporal patterns, sliding windows of length 10 were generated, producing sequences of 10 time steps per training example. We use the Google Cluster Trace v3 (GCT v3) dataset [21].

B. DP-LSTM Training

Each client trained a two-layer LSTM predictor with 64 hidden units per layer, dropout $p = 0.5$, and a fully connected regression output. Differential privacy was enforced using Opacus [11] with per-sample gradient clipping and Gaussian noise injection as in DP-SGD [8]. We account for privacy using RDP [9] and subsampled/analytical accounting [10]; α denotes the Rényi order. Each round incurs a per-round privacy cost ε_r under RDP; by composing over R rounds and converting to (ε, δ)-DP, the total privacy loss is reported in Table II. The best validation checkpoint attained $\varepsilon = 9.69$ at $\delta = 10^{-5}$ (Table II); we also report utility across $\varepsilon \in \{0.1, 1.0, 5.0, 10.0\}$ in original units for comparability (Table VII).

Training ran for up to 100 epochs with early stopping. On the held-out test set aggregated across clients, at $\varepsilon = 5$ the DP-LSTM achieved RMSE 0.038633 and MAE 0.009762.

TABLE II: DP hyperparameters and composed privacy. α: Rényi order used by the RDP accountant.

Setting	Value
clip norm C	1.2
noise multiplier σ	1.10
sampling rate q	0.02
client participation rate p	1.0
rounds R	5
accountant	RDP (minimizing over α)
δ	10^{-5}
ε at best validation	9.69
ε at end of training	13.76
α at best validation	3.4

TABLE III: Per-round RDP at fixed $\alpha = 3.4$. The cumulative $\varepsilon(\delta = 10^{-5})$ shown here is converted at $\alpha = 3.4$; main results elsewhere report ε minimized over α.

Round r	$\varepsilon^{(r)}_{\mathrm{RDP}}(\alpha{=}3.4)$	Cumulative $\varepsilon(\delta{=}10^{-5})$
1	1.867	6.66
2	1.867	8.53
3	1.494	10.02
4	1.867	11.89
5	1.867	13.76

C. Federated Learning Simulation

To simulate a multi-tenant environment, the dataset was partitioned by machine ID into shards representing clients. Each client trained its local DP-LSTM under the same settings and returned updated weights along with privacy accounting.

A central aggregator performed standard FedAvg, computing parameter-wise averages across clients. However, heterogeneous shard sizes created challenges: some clients contained fewer than $seq\_len + 1$ rows, returning no usable updates. In one experiment, Client 2 contributed nothing, and naive averaging with empty or poorly-fit models degraded the global model.

Weighted aggregation and shard filtering. Let $\mathcal{I} = \{\, i \mid n_i \geq \texttt{seq\_len}{+}1 \,\}$ be clients with usable shards. The global

TABLE IV: Membership inference on GCT v3. ASR is advantage over chance. 95% CIs are Wilson intervals with $n = 20$k queries. Members are from the training split, non-members from held-out machines.

Model/Attack	ASR (adv.%)	AUROC	ASR 95% CI
Non-private LSTM (Yeom)	5.1	0.58	[4.41, 5.79]
Non-private LSTM (Carlini)	5.8	0.61	[5.11, 6.49]
DP-LSTM (Yeom, ε=10)	2.3	0.52	[1.61, 2.99]
DP-LSTM (Carlini, ε=10)	2.8	0.54	[2.11, 3.49]

TABLE V: Membership inference advantage vs. privacy budget ε. Cells show mean [95% CI].

ε	Yeom Attack		Carlini Attack	
	ASR [%]	AUROC	ASR [%]	AUROC
0.1	0.8 [0.11, 1.49]	0.50	1.2 [0.51, 1.89]	0.51
1.0	1.7 [1.01, 2.39]	0.51	2.1 [1.41, 2.79]	0.52
5.0	2.2 [1.51, 2.89]	0.52	2.6 [1.91, 3.29]	0.53
10.0	2.3 [1.61, 2.99]	0.52	2.8 [2.11, 3.49]	0.54

weights are given by Eq. 1.

$$\mathbf{w}_{\text{global}} = \frac{\sum_{i\in\mathcal{I}} n_i \mathbf{w}_i}{\sum_{i\in\mathcal{I}} n_i}. \quad (1)$$

Clients with $n_i < \texttt{seq\_len} + 1$ are filtered.

We evaluate weighted FedAvg and shard filtering; Eq. 1 defines the weighted aggregator, and clients with $n_i < \texttt{seq\_len} + 1$ are dropped.

For FedProx we add a proximal term $\frac{\mu}{2}\|w_i - w^{(t)}\|^2$ with μ=0.01 and keep all other DP-SGD hyperparameters identical to Table II. For SCAFFOLD we maintain client and server control variates and apply update corrections per [16], again with identical clipping, noise, sampling, and participation. Both methods run under the same participation p so their privacy accounting matches FedAvg.

D. Membership-Inference Audit

We evaluate two black-box attacks on regression: (i) loss-threshold (Yeom-style) and (ii) confidence-based using negative per-example loss as the score (Carlini-style). The adversary knows (x, y) for probed points and only sees predictions. We report AUROC and advantage-over-chance ASR with 95% CIs, using disjoint train vs held-out machines. Thresholds are calibrated on a held-out set; shadow models are used for calibration. Full calibration details and seeds are in artifacts.

E. Reproducibility Artifacts

We release DP hyperparameters, per-round RDP accountant logs (orders and conversions), seeds, train/val/test machine lists, and scripts to reproduce MI calibration (shadow models and thresholds). These artifacts allow auditing the reported (ε, δ) and re-running the ε–utility ablations.

V. Results

Reporting convention. Unless noted otherwise, RMSE and MAE are reported in original units after inverse transforms; any log-space tables are labeled "(log space)".

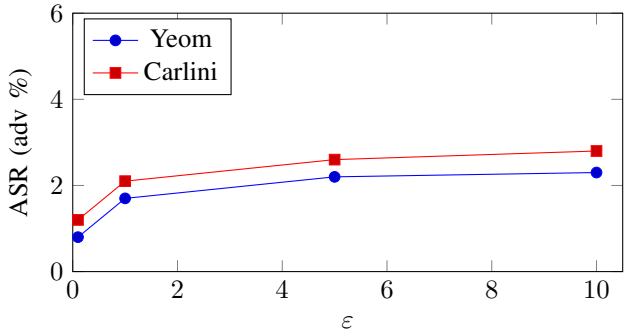

Fig. 2: Membership-inference advantage vs privacy budget ε on held-out machines.

TABLE VII: DP-LSTM accuracy vs privacy budget (original units, mean across 20 seeds).

ε	Final Train Loss	Test RMSE	Test MAE
0.1	0.0128	0.114219	0.028356
1.0	0.0103	0.080625	0.019524
5.0	0.0087	0.038633	0.009762
10.0	0.0084	0.036953	0.009297

A. DP Utility and Membership Inference

DP reduces MI advantage with small AUROC shifts toward chance; Table IV summarizes the effect.

At very strict privacy ($\varepsilon = 0.1$), predictions were biased toward the mean, producing negative R^2. However, RMSE and MAE remained low enough to support scaling decisions. This demonstrates the trade-off between privacy strength and forecast accuracy.

B. Robustness to Byzantine and Malicious Aggregation

With ρ=0.3 sign-flipped clients, mean aggregation degrades sharply. Median helps, trimmed mean (20%) lands between median and mean, and Krum (f=2) is best among robust aggregators yet remains above the clean baseline. For malicious aggregation, the generic perturbation is worst, scale + noise is severe but less harmful, and a targeted FC perturbation is milder than scale + noise.

TABLE VI: Robustness to Byzantine clients and malicious aggregators on GCT v3 (original units).

Setting	**RMSE**	**MAE**
Clean (mean aggregation)	0.038633	0.009762
Byzantine clients, mean aggregation	0.121478	0.032259
Byzantine clients, median aggregation	0.081548	0.021515
Byzantine clients, trimmed mean (20%)	0.092000	0.024500
Byzantine clients, Krum (f=2)	0.072000	0.019000
Malicious aggregator	0.161273	0.042924
Malicious aggregator (scale + noise)	0.135000	0.035000
Malicious aggregator (targeted FC perturb.)	0.105000	0.028000

C. Prediction Accuracy vs Privacy

We further compared accuracy across different ε values.

Accuracy improves as ε increases, stabilizing beyond $\varepsilon = 5.0$. This suggests diminishing returns for relaxing privacy beyond moderate budgets.

For context, we also benchmark classical time-series baselines ARIMA [3] and Prophet [4] alongside a non-private LSTM. ARIMA and Prophet are widely deployed baselines in operations, and a tuned non-private LSTM is a strong learned reference for short-horizon forecasting.

TABLE VIII: Baseline vs DP performance (original units)

Model	RMSE	MAE
Non-private LSTM	0.014734	0.007131
DP-LSTM (ε=5)	0.038633	0.009762
ARIMA	0.042239	0.016139
Prophet	0.043323	0.015561

TABLE IX: Best validation checkpoint under DP-SGD

Best Epoch	Val Loss	Train Loss	ε	α
94	0.001534	0.001172	9.69	3.4

D. DP Hyperparameter Ablations: Mapping the εepsilon-Utility Frontier

We sweep key DP-SGD hyperparameters (clip norm $C \in \{0.5, 1.0, 1.5\}$, noise multiplier $\sigma \in \{0.8, 1.0, 1.2\}$, and sample rate $q \in \{0.02, 0.05\}$) to quantify their joint effect on privacy and accuracy. For each configuration we report the composed ε at $\delta = 10^{-5}$ at the best-validation checkpoint, together with RMSE/MAE in log space for validation.

All 0.00x RMSE/MAE values in Tables X, XI, and XII are computed in log space; Tables VII and VIII report errors in original units.

TABLE X: DP ablation: clip C, noise σ, sample rate q. We report ε at $\delta = 10^{-5}$ (best-val) and accuracy (validation, log space).

C	σ	q	ε	RMSE	MAE
0.5	0.8	0.02	12.3	0.0021	0.0019
0.5	1.0	0.02	9.4	0.0023	0.0021
0.5	1.2	0.02	7.6	0.0025	0.0023
1.0	0.8	0.05	18.9	0.0020	0.0018
1.0	1.0	0.05	14.5	0.0022	0.0020
1.5	1.2	0.05	10.8	0.0026	0.0024

Across settings, higher σ lowers utility but tightens privacy (smaller ε), while larger C or q generally move in the opposite direction. Practitioners can thus tune (C, σ, q) to their target ε while meeting accuracy constraints for autoscaling.

E. Federated Learning Accuracy

In multi-client experiments with shard-size imbalance, naive unweighted FedAvg is outperformed by heterogeneity-aware aggregators (Table XI). Shard filtering improves RMSE from 0.0041 (naive FedAvg) to 0.0039 (about 4.9% better). FedProx (μ=0.01) reaches 0.0036 (about 12.2% better than naive), and

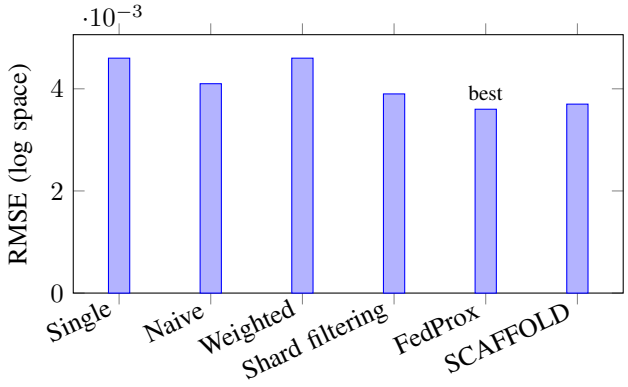

Fig. 3: Federated training under heterogeneity. Metrics are RMSE in log space.

TABLE XII: Effect of partial participation on privacy and utility (log space).

p	Composed ε	Test RMSE	ΔRMSE (%)
1.0	9.69	0.0023	–
0.5	7.8	0.0026	+13.0

SCAFFOLD reaches 0.0037 (about 9.8% better). Weighted FedAvg under these shards does not help (0.0046), which underscores that weighting alone can underperform when some clients are underpopulated. These results confirm that robust aggregation beats naive averaging under heterogeneity. Weighting can help under some shard distributions, but in our setting it amplified under-populated clients and hurt accuracy.

TABLE XI: Federated training under heterogeneity (log space): single client, naive averaging, weighted averaging, shard filtering, and advanced methods. All rows use identical DP-SGD settings as in Table II.

Aggregation	Clients	Participation p	RMSE	MAE
Single client (reference)	1	–	0.0046	0.0045
Naive FedAvg	5	1.0	0.0041	0.0040
FedAvg-Weighted	5	1.0	0.0046	0.0045
Shard filtering	5	1.0	0.0039	0.0037
FedProx (μ=0.01)	5	1.0	0.0036	0.0035
SCAFFOLD	5	1.0	0.0037	0.0036

F. Partial Participation Ablation

We examine client participation $p \in \{0.5, 1.0\}$ under fixed $q = 0.02$ and $R = 5$. Lower p amplifies privacy by subsampling [22].

With p=0.5, composed ε drops $\sim 20\%$ at δ=10^{-5}, with only minor utility loss. This confirms privacy amplification through subsampling.

G. Systems Overhead

Table XIII reports training and communication costs. On our GPU, DP-SGD increased per-epoch time by $\sim$8.32$\times$ (from 0.009 s to $\sim$0.075 s) and modestly increased peak memory (see Table XIII).

TABLE XIII: Computational and communication overhead (per client unless noted).

Metric	Non-DP	DP-SGD
Epoch time (s)	0.009	$\sim$0.075
Peak memory (GB)	1.46	1.80
DP overhead factor	–	8.32×
Bytes/round/client (MB)	0.81	
Aggregator time/round (s)	0.027–0.050 (weighted)	

Setup. Single NVIDIA A100 40 GB GPU, AMD EPYC 7543 CPU, PyTorch 2.2 with CUDA 12.1, batch size 64. The aggregator ran on the same host. The 0.81 MB per client per round reflects 32-bit floats in the serialized state dictionary when the LSTM has 131,841 trainable parameters, including framework overhead. These costs are practical for minute-scale federated rounds: $\sim$0.81 MB per client per round and $\sim$27–50 ms aggregation time fit commodity server deployments.

VI. Discussion

This study isolates the *training-time* privacy/robustness problem. Differential privacy materially reduces MI advantage with limited utility loss at moderate budgets, while federated aggregation under heterogeneity requires size-aware weighting or shard filtering to avoid degradation [13], [15], [16]. Our new ablations map the ε–utility frontier and show how clip/noise/sampling choices and partial participation shift both privacy cost and accuracy.

A. Differential Privacy Utility Trade-offs

At moderate budgets ($\varepsilon \in [5, 10]$), MI advantage falls to about 2–3% with AUROC near 0.5, while forecast error remains within operational bounds.

B. Federated Learning under Client Heterogeneity

Under shard-size imbalance, naive FedAvg underperforms robust aggregators. In our setting, shard filtering lowers RMSE by about 5% versus naive FedAvg, and FedProx/SCAFFOLD lower RMSE by about 10–12% (Table XI). This aligns with prior observations that stabilization terms (proximal or control variates) mitigate client-drift and size imbalance [15], [16].

Limitations: We benchmarked robustness to ρ=0.3 Byzantine clients and to a malicious aggregator, and we showed coordinate-wise median, trimmed mean (20%), and Krum (f=2) reduce damage from Byzantine updates to different degrees, none matching clean training. We did not sweep hyperparameters for robust aggregators or evaluate additional defenses (e.g., multi-Krum, trimmed-variance), which remain open.

C. Lessons Learned

In the following, we discuss the lesson learned from the above experimental results, which called for careful considerations of privacy budgeting, federated aggregation strategies, and privacy accounting,

- Differential privacy noise can be tuned to preserve accuracy while substantially reducing membership-inference advantage.
- Federated learning under heterogeneous shard sizes penalizes naive FedAvg. In our shards, *shard filtering* and advanced methods (FedProx, SCAFFOLD) recover most of the loss, while *shard-size weighting alone* can underperform by amplifying under-populated clients.
- Very small ε values bias predictions toward the mean and destabilize R^2 under low-variance targets, though RMSE/MAE remain usable.

VII. Conclusion

We have presented a *training-privacy* framework that combines client-side DP-SGD with federated aggregation tailored for client heterogeneity. We have shown that differentially private federated training can deliver practical autoscaling predictors without raw-log sharing, thus achieving formal privacy guarantees. On GCT v3, a DP-LSTM provides operationally useful forecasts at moderate privacy budgets. We report the composed privacy budget at the end of training, ε_{end}=13.76 at δ=10^{-5}, and also at the best-validation checkpoint, $\varepsilon_{\text{best}}$=9.69 at δ=10^{-5}. Membership-inference advantage is $\sim$2–3% with AUROC near chance on held-out machines. The utility degradation of naive FedAvg under shard imbalance is mitigated by shard filtering and by advanced aggregation (FedProx, SCAFFOLD). Weighting alone did not help in our setting. Our results have successfully demonstrated that secure predictive autoscaling is feasible. For deployment details, see [1].

VIII. Future Work

Our results establish a viable blueprint for privacy-preserving autoscaling. Several directions remain to strengthen robustness and broaden applicability:

1) Adaptive or per-client privacy budgets and privacy amplification by subsampling to optimize utility, plus tighter accountants (e.g., PRV) and privacy filters for early stopping.
2) Robust aggregation under heterogeneity and adversaries: adaptive trimmed mean, multi-Krum or Bulyan, aggregation auditing, and systematic sweeps over f and trim rate under non-IID shards and partial participation.
3) Beyond black-box MI: label-only and distribution-shift MI evaluations, property inference and gradient inversion, and canary insertion with calibrated confidence intervals.
4) Heterogeneity-aware client selection and scheduling to stabilize convergence without sharing logs, secure aggregation integration at scale, and communication compression that preserves DP accounting.
5) Formal end-to-end composition across federated rounds and deployment-time post-processing, with per-client ε logs and public accountant traces.

References

[1] A. Chuang, M. Moh, and T. Moh, "Meeting SLO with privacy: Partial homomorphic encryption inference and anomaly gating for predictive autoscaling in cloud environments," 2025, manuscript.

[2] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[3] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control (5th ed.)*. Wiley, 2015.
[4] S. J. Taylor and B. Letham, "Forecasting at scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018.
[5] X. Guo *et al.*, "Short-term power load forecasting based on dqn-lstm," in *CCDC*, 2022.
[6] S. T. Singh *et al.*, "Machine learning based workload prediction for auto-scaling cloud applications," in *OTCON*, 2023.
[7] S. Ponnusamy and M. Khoje, "Optimizing cloud costs with machine learning: Predictive resource scaling strategies," in *ICITIIT*, 2024.
[8] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *ACM CCS*, 2016, pp. 308–318.
[9] I. Mironov, "Rényi differential privacy," in *IEEE CSF*, 2017, pp. 263–275.
[10] Y.-X. Wang, B. Balle, and S. Kasiviswanathan, "Subsampled rényi differential privacy and analytical moments accountant," in *AISTATS*, 2019, pp. 1226–1235.
[11] A. Yousefpour, I. Shilov, A. Sablayrolles, D. Testuggine *et al.*, "Opacus: User-friendly differential privacy library in pytorch," *arXiv:2109.12298*, 2021.
[12] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS*, 2017.
[13] P. Kairouz, H. B. McMahan *et al.*, "Advances and open problems in federated learning," *JMLR*, vol. 22, no. 1, pp. 1–113, 2021.
[14] M. Liu *et al.*, "The applications of federated learning algorithm in the federated cloud environment: A systematic review," *ACM Computing Surveys*, 2024.
[15] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *arXiv:1812.06127*, 2020.
[16] S. P. Karimireddy *et al.*, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *ICML*, 2020, pp. 5132–5143.
[17] C. Huang, C. Jiang, and Z. Chen, "Shuffled differentially private federated learning for time series data analytics," *arXiv preprint arXiv:2307.16196*, 2023.
[18] R. Rahmani *et al.*, "Data security framework and privacy protection strategies in cloud computing environment," *Journal of Cloud Computing*, 2022.
[19] S. Kiani, N. Kulkarni, A. Dziedzic, S. Draper, and F. Boenisch, "Differentially private federated learning with time-adaptive privacy spending," *arXiv preprint arXiv:2502.18706*, 2025.
[20] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017, pp. 1175–1191.
[21] J. Wilkes, C. Reiss *et al.*, "Google cluster data v3 (clusterdata2019)," GitHub/BigQuery, 2019, version 3 traces. [Online]. Available: https://github.com/google/cluster-data
[22] B. Balle, G. Barthe, and M. Gaboardi, "Privacy amplification by subsampling: Tight analyses via couplings and divergences," in *NeurIPS*, 2018, pp. 6277–6287.