Beyond Secure Aggregation: Scalable Multi-Round Secure Collaborative Learning

Umit Yigit Basaran [*1] **Xingyu Lu** [*1] **Başak Güler** [1]

Abstract

Privacy-preserving machine learning (PPML) has achieved exciting breakthroughs for secure collaborative training of machine learning models under formal information-theoretic privacy guarantees. Despite the recent advances, communication bottleneck still remains as a major challenge against scalability to large neural networks. To address this challenge, in this work we introduce the first end-to-end multi-round multi-party neural network training framework with linear communication complexity, under formal information-theoretic privacy guarantees. Our key contribution is a scalable secure computing mechanism for iterative polynomial operations, which incurs only linear communication overhead, significantly improving over the quadratic state-of-the-art, while providing formal end-to-end multi-round information-theoretic privacy guarantees. In doing so, our framework achieves equal adversary tolerance, resilience to user dropouts, and model accuracy as the state-of-the-art, while addressing a key challenge in scalable training.

1. Introduction

Secure multi-party collaborative learning has achieved promising advances for training ML models across multiple data-owners (users), under formal information-theoretic privacy guarantees (Mohassel & Zhang, 2017; Ben-Or et al., 1988; Beerliová-Trubíniová & Hirt, 2008; Al-Rubaie & Chang, 2019; Damgård & Nielsen, 2007; Nikolaenko et al., 2013; Gascón et al., 2017; Mohassel & Rindal, 2018; Wagh et al., 2018). These frameworks typically build on a secure multi-party computing (MPC) primitive known as secret sharing (Shamir, 1979), where each user encodes their local dataset (i.e., the secret) by injecting randomness in the form of random masks, and sends an encoded dataset (i.e., secret share) to every other user. The training computations are then performed on the encoded datasets, as opposed to the true data. At the end of training, i.e., after multiple (global) training rounds, the final model can be decoded by collecting the coded computations (performed on the encoded datasets) from a sufficient number of users. In doing so, no information is revealed on the local datasets, beyond the final model. More recently, information and coding theoretic approaches have demonstrated promising advances in PPML, which utilize a Lagrange interpolation polynomial to encode the datasets, to inject randomness and (computational) redundancy across the computations performed by different users (Yu et al., 2019; So et al., 2020; 2021). Doing so can achieve an order-of-magnitude speedup in training compared to state-of-the-art cryptographic baselines, while providing information-theoretic privacy for the local datasets, and resilience against user dropouts.

A major advantage is their compatibility with complementary differential privacy (DP) techniques, where a discrete noise mechanism can be employed to prevent potential information leakage also from the final model (Dwork et al., 2006; Chaudhuri & Monteleoni, 2009; Abadi et al., 2016; Pathak et al., 2010; McMahan et al., 2018; Rajkumar & Agarwal, 2012; Jayaraman et al., 2018). Several notable works have demonstrated that integrating the two can achieve the best of both worlds; by reducing the amount of noise that should be added to the local computations in distributed settings, achieving better model accuracy for DP, while preventing information leakage from the final model (Chen et al., 2022b;a; Kairouz et al., 2021). Though beyond our current focus, we note that our techniques can be integrated with DP as an interesting future direction.

Communication bottleneck. The main challenge against the scalability of end-to-end information-theoretic learning frameworks is their extensive communication complexity, limiting applications to simpler linear/logistic regression tasks, and preventing scalability beyond $2-4$ users in more complex neural network training. As a result, for neural networks, information-theoretic secure computing approaches are primarily utilized for the gradient aggre-

[*]Equal contribution [1]Department of Electrical and Computer Engineering, University of California Riverside, CA, USA. Correspondence to: Umit Basaran <ubasa001@ucr.edu>, Xingyu Lu <xlu065@ucr.edu>, Basak Guler <bguler@ece.ucr.edu>.

Workshop of Federated Learning and Analytics in Practice, colocated with 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. Copyright 2023 by the author(s).

gation task for large-scale distributed/federated learning, also known as *secure aggregation* (SA), where the aggregated gradient/model is revealed after each (global) training round (Bonawitz et al., 2017; Bell et al., 2020; Zhao & Sun, 2021; So et al., 2022). On the other hand, SA is vulnerable to multi-round privacy attacks; the aggregated models can still reveal extensive information over multiple training rounds, and the privacy protection degrades as the number of training rounds increase (So et al., 2023; Elkordy et al., 2023). In contrast, in this work our focus is on *end-to-end* information-theoretic privacy, where *no intermediate model/gradient can be revealed (even in aggregated form)* beyond the final model after multiple training rounds.

Contributions. To address this challenge, we introduce CLOVER, the first scalable multi-round collaborative neural network training framework under end-to-end multi-round information-theoretic privacy, with linear communication complexity. In doing so, we separate communication into online (data-dependent) and offline (data-agnostic) components. Offline communication is independent from data (e.g., randomness generation), and can be carried out in advance when network load is low. We then offload the communication-intensive operations (with quadratic overhead) to the offline phase, and introduce a coded randomness generation mechanism, using Lagrange interpolation polynomials along with MDS (Maximum Distance Separable) codes, such that the total number of variables communicated is inversely proportional to the number of users, with linear amortized communication complexity. In a network of N users, our framework achieves an $O(N)$ (linear) communication complexity both offline and online (for neural network training), as opposed to the $O(N^2)$ (quadratic) online communication complexity of the state of the art (which, as a result, is limited to simpler logistic/linear regression) (So et al., 2020), while achieving equal adversary and dropout resilience. Our contributions are summarized as:

- We propose the first scalable neural network training framework with end-to-end multi-round information-theoretic privacy, and linear communication overhead.
- Our framework provides formal information-theoretic privacy guarantees, and cuts the communication overhead while achieving equal adversary and dropout resilience as the state-of-the-art.
- Beyond distributed learning, our framework can open up further research in a broad range of privacy-sensitive iterative algorithms, such as for federated analytics.

2. Problem Formulation

Multi-party Neural Network Training.

We consider a network of N users. User i holds a local dataset given by a $d \times m_i$ matrix $\mathbf{X}_i$, where m_i and d denote the number of data points and features, respectively. The corresponding labels are represented by a $c \times m_i$ binary matrix $\mathbf{Y}_i$, where the k^{th} column is the one-hot label vector for data point $k \in [m_i]$, and c is the number of classes. The dataset and labels across the entire network are denoted by $\mathbf{X} \triangleq (\mathbf{X}_1, \ldots, \mathbf{X}_N)$ and $\mathbf{Y} \triangleq (\mathbf{Y}_1, \ldots, \mathbf{Y}_N)$. The communication topology is decentralized (without a central server). We consider a polynomial neural architecture (Chrysos et al., 2021; Kileel et al., 2019) along with quadratic activations and mean-squared error loss. The goal is to learn a model $\mathbf{W}$ to minimize the empirical loss function:

$$\frac{1}{m} \sum_{i \in [m]} \mathcal{L}(\mathbf{W}; \mathbf{x}_i, \mathbf{y}_i) = \frac{1}{m} \sum_{i \in [m]} \| f(\mathbf{x}_i, \mathbf{W}) - \mathbf{y}_i \|_2^2 \quad (1)$$

where $m \triangleq \sum_{i \in [N]} m_i$, and $f(\mathbf{x}_i, \mathbf{W})$ is the output of a feed-forward neural network consisting of L hidden layers, and a final classification layer (layer $L+1$) as shown in Fig. 1, at a single data point $\mathbf{x}_i$ with label $\mathbf{y}_i$ (i^{th} column of $\mathbf{X}$ and $\mathbf{Y}$). The model parameters connecting layer $l-1$ to layer l are given by a matrix $\mathbf{W}^l$ of size $d_l \times d_{l-1}$, where d_l is the number of neurons at layer $l \in [L+1]$, and $\mathbf{W} = (\mathbf{W}^1, \ldots, \mathbf{W}^{L+1})$. Accordingly, $d_0 = d$, and $d_{L+1} = c$. Training is done via gradient descent, where the model is updated iteratively as,

$$\mathbf{W}(t+1) = \mathbf{W}(t) - \frac{\eta}{m} \sum_{i \in [m]} \nabla \mathcal{L}(\mathbf{W}(t); \mathbf{x}_i, \mathbf{y}_i) \quad (2)$$

where $\nabla \mathcal{L}(\mathbf{W}(t); \mathbf{x}_i, \mathbf{y}_i)$ is the gradient for a single data point, $\mathbf{W}(t)$ is the estimated model parameters from training round t. Up to D users may drop out at each training round (e.g., due to poor wireless conditions or unavailability).

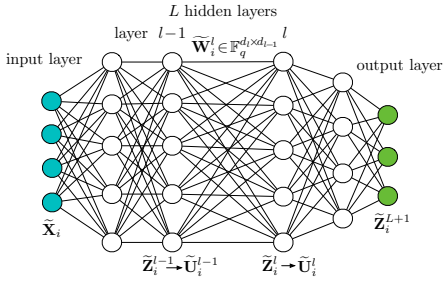


Figure 1. **Neural architecture.** The neural network consists of an input layer, L hidden layers, and an output (classification) layer. The encoded model parameters connecting layer $l-1$ to layer l is given by $\widetilde{\mathbf{W}}_i^l \in \mathbb{F}_p^{d_l \times d_{l-1}}$ for user $i \in [N]$, where d_l is the number of neurons at layer l.

Threat Model. Our focus is on an *honest-but curious* adversary model, in which adversaries follow the protocol but try to obtain further information about the local datasets of honest users. Up to T users are adversarial, who may collude with each other. The set of adversarial and honest users are denoted by $\mathcal{T}$ and $\mathcal{H}$, respectively.

Information-theoretic Privacy. Our goal is end-to-end information-theoretic privacy, where adversaries learn no

information about the local datasets of honest users, beyond the final model. Formally, this can be stated as,

$$I(\{\mathbf{X}_i, \mathbf{Y}_i\}_{[N]\setminus\mathcal{T}}; \mathcal{M}_\mathcal{T} | \{\mathbf{X}_i, \mathbf{Y}_i\}_{i\in\mathcal{T}}, \mathbf{W}(J)) = 0 \quad (3)$$

for any $\mathcal{T}$ such that $|\mathcal{T}| \leq T$, where J is the total number of training rounds, and $\mathcal{M}_\mathcal{T}$ is the collection of all messages received or generated by the adversaries. Similar to (Bonawitz et al., 2017; So et al., 2020; Bell et al., 2020), our framework is bound to finite field operations, where the datasets are represented in a finite field $\mathbb{F}_p$ of integers modulo a large prime p.

Main Problem. To address this challenge, a recent PPML framework for end-to-end (multi-round) information-theoretic privacy (3) is the COPML framework from (So et al., 2020). COPML utilizes Lagrange Coded Computing (LCC) (Yu et al., 2017) to encode the datasets and model, which distributes the training load effectively across the clients, and accelerates the local computations, providing an order of magnitude speed-up compared to well-known secure computing frameworks (Ben-Or et al., 1988; Beerliová-Trubíniová & Hirt, 2008). To do so, the dataset $\mathbf{X}$ is first partitioned into K equal-sized shards, and then encoded using a Lagrange interpolation polynomial, along with T random masks. Each user then obtains an encoded dataset $\widetilde{\mathbf{X}}_i$, whose size is only $(1/K)^{th}$ of the original dataset $\mathbf{X}$. Training is then performed on the *coded datasets*, and the final model $\mathbf{W}(J)$ is decoded by polynomial interpolation, by collecting the computations from $N - D \geq (\deg f)(K + T - 1) + 1$ surviving users, where $\deg f$ quantifies the polynomial degree after J training rounds. Parameter K quantifies the *degree of parallelization*; each user processes $(1/K)^{th}$ of the dataset $\mathbf{X}$, hence as the network size N grows, one can select a larger K for faster training.

On the other hand, $\deg f$ grows exponentially after each multiplication operation (associated with gradient computations). To prevent a degree explosion, users perform an expensive *degree reduction* operation, which requires alternating between Lagrange coding and Shamir's secret sharing. After each local computation (e.g., gradient computations) performed using the Lagrange coded dataset and model, users decode the computations using Shamir's secret sharing, update the model, and then re-encode the model using a new Lagrange polynomial (So et al., 2020). This decode-update-re-encode process incurs a quadratic communication overhead after each training round. As a result, COPML is applied to simpler logistic/linear regression models, as opposed to more complex neural network training. In this work, we ask the following question,

- *Can we train a neural network to solve* (1) *with linear communication complexity, with provable end-to-end information-theoretic privacy guarantees from* (3)*?*

To address this challenge, we propose CLOVER, a scalable multi-party neural network training framework with end-to-end information-theoretic privacy. Our key contribution is a novel coded computing mechanism, termed *Double Lagrange Coding*, for handling iterative multiplicative operations associated with forward/backward propagation. We next describe the individual steps of our framework.

3. The CLOVER Framework

Dataset and Label Encoding. Initially, users agree on $N + K + T$ distinct public parameters $\{\alpha_j\}_{j\in[N]}$, $\{\beta_j\}_{j\in[K+T]}$ from $\mathbb{F}_p$. Each user $i \in [N]$ then partitions its local dataset into K equal-sized shards, $\mathbf{X}_i = (\mathbf{X}_{i1}, \ldots, \mathbf{X}_{iK})$, and generates a Lagrange polynomial of degree $K + T - 1$:

$$u_i(z) \triangleq \sum_{k\in[K]} \mathbf{X}_{ik} \prod_{l\in[K+T]\setminus\{k\}} \frac{z-\beta_l}{\beta_k-\beta_l} + \sum_{k=K+1}^{K+T} \mathbf{V}_{ik} \prod_{l\in[K+T]\setminus\{k\}} \frac{z-\beta_l}{\beta_k-\beta_l} \quad (4)$$

where $u_i(\beta_k) = \mathbf{X}_{ik}$, and $\{\mathbf{V}_{ik}\}_{k=K+1}^{K+T}$ are T uniformly random masks from $\mathbb{F}_p^{d\times\frac{m_i}{K}}$, and sends $\widetilde{\mathbf{X}}_{ij} \triangleq u_i(\alpha_j)$ to user $j \in [N]$. By concatenating the received $\{\widetilde{\mathbf{X}}_{ji}\}_{j\in[N]}$, each user $i \in [N]$ then obtains an encoded dataset:

$$\widetilde{\mathbf{X}}_i \triangleq (\widetilde{\mathbf{X}}_{1i}, \ldots, \widetilde{\mathbf{X}}_{Ni}) \in \mathbb{F}_p^{d\times\frac{m}{K}} \quad (5)$$

The labels $\mathbf{Y}$ are encoded similarly, at the end of which each user $i \in \{N\}$ obtains the encoded labels $\widetilde{\mathbf{Y}}_i \in \mathbb{F}_p^{c\times\frac{m}{K}}$. The goal of dataset and label encoding is two-fold: 1) hide their content against adversaries, 2) reduce the size of data processed during training; each user computes the gradient over the encoded dataset $\widetilde{\mathbf{X}}_i$, whose size is $(1/K)^{th}$ of the original dataset $\mathbf{X}$. As the network size N increases, one can select a larger K, reducing the computation load per-user, speeding up training.

Forward/Backward Propagation. To preserve the privacy of intermediate computations, the model $\mathbf{W}(0) = (\mathbf{W}^1(0), \ldots, \mathbf{W}^{L+1}(0))$ is initialized randomly, also by encoding via a Lagrange polynomial of degree $K + T - 1$, at the end of which each user $i \in [N]$ learns an encoded model $\widetilde{\mathbf{W}}_i^l(0)$ for all $l \in [L+1]$. Using the encoded dataset and labels, users then compute the gradient and update the model. The main challenge, however, is that the polynomial degree doubles with each multiplication operation. Specifically, for forward propagation, each user $i \in [N]$ computes,

$$\widetilde{\mathbf{Z}}_i^l(t) \triangleq \widetilde{\mathbf{W}}_i^l(t)\widetilde{\mathbf{U}}_i^{l-1}(t) \in \mathbb{F}_p^{d_l\times 1}, \quad (6)$$

$$\widetilde{\mathbf{U}}_i^l(t) \triangleq g(\widetilde{\mathbf{Z}}_i^l(t)) \in \mathbb{F}_p^{d_l\times 1} \quad (7)$$

for all $l \in [L+1]$, where $\widetilde{\mathbf{U}}_i^0(t) \triangleq \widetilde{\mathbf{X}}_i$, and $g(\cdot)$ is the quadratic activation function applied element-wise. Note

that $\widetilde{\mathbf{X}}_i$ corresponds to (evaluations of) a degree $K+T-1$ polynomial, on the other hand, the degree of the resulting polynomial in (6) grows exponentially as the number of layers increase, which requires users to reduce the polynomial degree without breaching privacy. To address this, users carry out a degree reduction operation $\phi(\cdot)$,

$$(\widetilde{\mathbf{Z}}_1^l(t),\dots,\widetilde{\mathbf{Z}}_N^l(t)) \leftarrow \phi(\widetilde{\mathbf{Z}}_1^l(t),\dots,\widetilde{\mathbf{Z}}_N^l(t)) \quad (8)$$

which reduces the polynomial degree back to $K+T-1$.

Double Lagrange Coding. Due to space constraints), the details of this degree reduction operation are provided in App. A. At the outset, for each multiplication operation, this mechanism generates two Lagrange polynomials, a higher degree polynomial used to decode a masked version of the true computations (while keeping their true values hidden), and a lower degree polynomial to re-encode them. In doing so, we decouple communication into online and offline phases, and offload the communication-intensive operations to the latter. We then propose an efficient offline randomness generation mechanism where the communication volume is inversely proportional to the number of users, and the overall communication overhead (online and offline) is linear. At the end of (8), each user $i\in[N]$ obtains an updated coded vector $\widetilde{\mathbf{Z}}_i^l(t)$ corresponding to a Lagrange polynomial with degree $K+T-1$. The steps for backpropagation are similar; after each multiplication operation, users carry out a degree reduction operation as in (8), to reduce the degree back to $K+T-1$.

After computing the gradients, users update the model according to (2), as described in App. B. At the end of J rounds, parties collect the coded models $\widetilde{\mathbf{W}}_i(J)$ from any set of $K+T$ users, and decode the final model $\mathbf{W}(J)$ using polynomial interpolation.

4. Theoretical Analysis

We now present the formal guarantees of CLOVER.

Theorem 4.1. *(Information-theoretic privacy) CLOVER guarantees information theoretic privacy:*

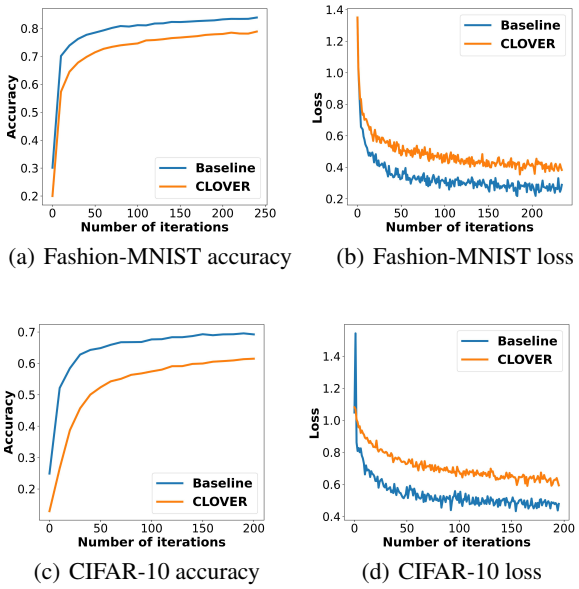
$$I(\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in\mathcal{H}};\mathcal{M}_{\mathcal{T}}|\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in\mathcal{T}},\mathbf{W}(J))=0 \quad (9)$$

against any set $\mathcal{T}$ of $|\mathcal{T}|\le T$ adversaries, where $\mathcal{M}_{\mathcal{T}}$ is the collection of all messages held by adversaries.

Proof. The proof is provided in Appendix C. □

In the sequel, we let $m_i=\bar{m}$ for all $i\in[N]$, to present the complexity explicitly with respect to the number of users.

Theorem 4.2. *(Communication complexity) The per-user communication complexity of CLOVER is $O(\frac{(d+c)N\bar{m}}{K}+\frac{JN\bar{m}}{K}\sum_{l\in[L+1]}d_ld_{l-1})$ in the online phase, and $O(\frac{JN^2\bar{m}}{(N-T)K}\sum_{l\in[L+1]}d_ld_{l-1})$ in the offline phase, respectively. With $T=O(N)$ and $K=\Theta(N)$, the total communication complexity across all N users (including both online and offline phases) is linear in the number of users, which is $O(N(d+c)\bar{m}+NJ\bar{m}\sum_{l\in[L+1]}d_ld_{l-1})$.*



(a) Fashion-MNIST accuracy (b) Fashion-MNIST loss

(c) CIFAR-10 accuracy (d) CIFAR-10 loss

Figure 2. Model accuracy and training loss for the Fashion MNIST and CIFAR-10 datasets, with respect to the baseline neural network with ReLU activation and softmax classification (representing our target accuracy).

The *recovery threshold* is defined as the minimum number of users required to correctly decode the final model.

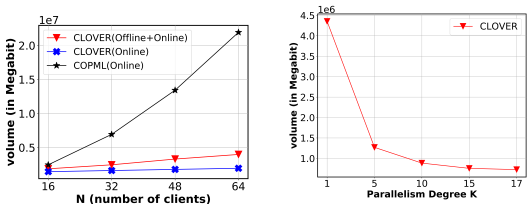
Theorem 4.3. *(Recovery threshold) The recovery threshold of CLOVER is $N\ge D+3(K+T-1)+1$.*

Proof. The recovery threshold is given by the minimum number of computations required for polynomial interpolation, which is $N-D\ge 3(K+T-1)+1$ from Section 3, which is equal to (So et al., 2020). □

5. Experiments

To evaluate the numerical performance, we have implemented a distributed network with $N=64$ users, using the `MPI4Py` Message Passing Interface, and train a two layer neural network for image classification on the Fashion-MNIST and CIFAR-10 datasets. We defer the additional experimental details to App. D.

Performance evaluation. To correctly recover the final model, the number of clients must satisfy the recovery threshold from Thm. 4.3. where the degree of privacy (T) and parallelization (K) are calculated by letting $N=3(K+T-1)+1$ with $T=\lfloor\frac{N}{16}\rfloor$ and $K=\lfloor\frac{N}{8}\rfloor-T$. Similar to (So et al., 2020), the bandwidth and finite field size are set as 40Mbps and $q=2^{26}-5$, respectively.



(a) Communication volume (b) Parallelization gain

Figure 3. (a) Total communication volume (across all clients) for CLOVER and COPML, and (b) Parallelization gain for CLOVER.

Model accuracy. For the baseline target accuracy, we evaluate the model accuracy of the same two-layer neural network trained in the real number domain, by utilizing ReLU activation (Agarap, 2019) and Adam optimizer (Kingma & Ba, 2017). In Fig. 2, we compare the model accuracy of CLOVER, with respect to the baseline for N = 64.

Communication volume. In Fig. 5(a), we compare the total communication volume (across all clients) for CLOVER with respect to COPML (So et al., 2020) for CIFAR-10. We observe that CLOVER can significantly reduce the communication overhead. Finally, in Fig. 5(b), we demonstrate the communication volume of CLOVER with respect to K. We observe that as K increases, the communication volume decreases, This also demonstrates a trade-off between communication overhead, adversary and dropout resilience, as $D + 3(K + T - 1) + 1 \leq N$.

6. Conclusion

This work presents a scalable neural network training framework with strong information-theoretic privacy guarantees. Our framework achieves linear communication complexity, while providing fundamental trade-offs between parallelism gain, adversary, and dropout resilience. Future directions include extending our secure coded computing framework to different neural network structures and adversary models (e.g., active adversaries).

Acknowledgements

This research was sponsored in part by the NSF CAREER Award CCF-2144927, OUSD (R&E)/RT&L under Cooperative Agreement Number W911NF-20-2-0267, and the UCR OASIS Funding Award. The views and conclusions contained in this document are those of the authors.

References

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep learning with differential privacy. In *ACM SIGSAC Conference on Computer and Communications Security*, pp. 308–318, 2016.

Agarap, A. F. Deep learning using rectified linear units (relu), 2019.

Al-Rubaie, M. and Chang, J. M. Privacy-preserving machine learning: Threats and solutions. *IEEE Security & Privacy*, 17(2):49–58, 2019.

Beerliová-Trubíniová, Z. and Hirt, M. Perfectly-secure MPC with linear communication complexity. In *Theory of Cryptography Conference*, pp. 213–230. Springer, 2008.

Bell, J. H., Bonawitz, K. A., Gascón, A., Lepoint, T., and Raykova, M. Secure single-server aggregation with (poly) logarithmic overhead. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1253–1269, 2020.

Ben-Or, M., Goldwasser, S., and Wigderson, A. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing (STOC'88)*, pp. 1–10, 1988.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191, 2017.

Chaudhuri, K. and Monteleoni, C. Privacy-preserving logistic regression. In *Adv. in Neural Inf. Proc. Sys.*, pp. 289–296, 2009.

Chen, W.-N., Choo, C. A. C., Kairouz, P., and Suresh, A. T. The fundamental price of secure aggregation in differentially private federated learning. In *International Conference on Machine Learning*, pp. 3056–3089. PMLR, 2022a.

Chen, W.-N., Ozgur, A., and Kairouz, P. The poisson binomial mechanism for unbiased federated learning with secure aggregation. In *International Conference on Machine Learning*, pp. 3490–3506. PMLR, 2022b.

Chrysos, G. G., Moschoglou, S., Bouritsas, G., Deng, J., Panagakis, Y., and Zafeiriou, S. Deep polynomial neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 44(8):4021–4034, 2021.

Damgård, I. and Nielsen, J. B. Scalable and unconditionally secure multiparty computation. In *Annual International Cryptology Conference*, pp. 572–590. Springer, 2007.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, pp. 265–284. Springer, 2006.

Elkordy, A. R., Zhang, J., Ezzeldin, Y. H., Psounis, K., and Avestimehr, S. How much privacy does federated learning with secure aggregation guarantee? *Proc. Priv. Enhancing Technol. (PETS'23)*, 2023(1):510–526, 2023.

Gascón, A., Schoppmann, P., Balle, B., Raykova, M., Doerner, J., Zahur, S., and Evans, D. Privacy-preserving distributed linear regression on high-dimensional data. *Proceedings on Privacy Enhancing Tech.*, 2017(4):345–364, 2017.

Jayaraman, B., Wang, L., Evans, D., and Gu, Q. Distributed learning without distress: Privacy-preserving empirical risk minimization. *Advances in in Neural Information Processing Systems*, pp. 6346–6357, 2018.

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

Kileel, J., Trager, M., and Bruna, J. On the expressive power of deep polynomial neural networks. *Advances in neural information processing systems*, 32, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization, 2017.

Krizhevsky, A., Nair, V., and Hinton, G. Cifar-10 (canadian institute for advanced research). 2009. URL http://www.cs.toronto.edu/~kriz/cifar.html.

McMahan, H. B., Ramage, D., Talwar, K., and Zhang, L. Learning differentially private recurrent language models. In *Int. Conf. on Learning Representations*, 2018.

Mohassel, P. and Rindal, P. ABY 3: A mixed protocol framework for machine learning. In *ACM SIGSAC Conference on Computer and Communications Security*, pp. 35–52, 2018.

Mohassel, P. and Zhang, Y. SecureML: A system for scalable privacy-preserving machine learning. In *38th IEEE Symposium on Security and Privacy*, pp. 19–38. IEEE, 2017.

Nikolaenko, V., Weinsberg, U., Ioannidis, S., Joye, M., Boneh, D., and Taft, N. Privacy-preserving ridge regression on hundreds of millions of records. In *IEEE Symposium on Security and Privacy*, pp. 334–348, 2013.

Pathak, M., Rane, S., and Raj, B. Multiparty differential privacy via aggregation of locally trained classifiers. In *Advances in Neural Inf. Processing Systems*, pp. 1876–1884, 2010.

Rajkumar, A. and Agarwal, S. A differentially private stochastic gradient descent algorithm for multiparty classification. In *Int. Conf. on Artificial Intelligence and Statistics (AISTATS'12)*, volume 22, pp. 933–941, La Palma, Canary Islands, Apr 2012.

Shamir, A. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition, 2015.

So, J., Güler, B., and Avestimehr, S. A scalable approach for privacy-preserving collaborative machine learning. In *Advances in Neural Information Processing Systems: Annual Conference on Neural Information Processing Systems, NeurIPS*, 2020.

So, J., Güler, B., and Avestimehr, A. S. Codedprivateml: A fast and privacy-preserving framework for distributed machine learning. *IEEE Journal on Selected Areas in Information Theory*, 2(1):441–451, 2021.

So, J., He, C., Yang, C.-S., Li, S., Yu, Q., E Ali, R., Guler, B., and Avestimehr, S. Lightsecagg: a lightweight and versatile design for secure aggregation in federated learning. *Proceedings of Machine Learning and Systems*, 4: 694–720, 2022.

So, J., Ali, R. E., Guler, B., Jiao, J., and Avestimehr, S. Securing secure aggregation: Mitigating multi-round privacy leakage in federated learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Wagh, S., Gupta, D., and Chandran, N. SecureNN: Efficient and private neural network training. Cryptology ePrint Archive, Report 2018/442, 2018.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. URL http://arxiv.org/abs/1708.07747.

Yu, Q., Maddah-Ali, M. A., and Avestimehr, A. S. Coded fourier transform. In *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 494–501. IEEE, 2017.

Yu, Q., Li, S., Raviv, N., Kalan, S. M. M., Soltanolkotabi, M., and Avestimehr, S. A. Lagrange coded computing: Optimal design for resiliency, security, and privacy. In *The 22nd International Conference on Artificial Intelligence and Statistics (AISTATS 2019)*, pp. 1215–1225. PMLR, 2019.

Zhao, Y. and Sun, H. Information theoretic secure aggregation with user dropouts. In *IEEE International Symposium on Information Theory, ISIT'21*, 2021.

A. Double Lagrange Coding

We now introduce the details of our degree reduction primitive, *Double Lagrange Coding* (DLC). This operation builds on two Lagrange polynomials; a higher degree polynomial to decode a masked version of the true gradient computations, and a lower degree Lagrange polynomial to re-encode them. To do so, we utilize an efficient shared randomness generation mechanism using MDS codes, with linear amortized communication complexity.

Consider a polynomial $f(\cdot)$ of degree $\deg(f) = M$ for some $M > K+T-1$, where $f(\beta_1), \dots, f(\beta_K) \in \mathbb{F}_p^{d_1 \times d_2}$ represent the K desired computations (e.g., gradient computations for K data points) for some d_1, d_2, and $f(\alpha_i)$ is the computation performed by user $i \in [N]$. DLC then generates a new (low-degree) Lagrange polynomial $f'(\cdot)$ of degree $K+T-1$, such that $f'(\beta_k) = f(\beta_k)$ for all $k \in [K]$, and $f'(\beta_k) \in \mathbb{F}_p^{d_1 \times d_2}$ are uniformly random vectors for $k \in \{K+1, \dots, K+T\}$. At the end, each user $i \in [N]$ only learns an evaluation point $f'(\alpha_i)$, without learning any information about the true computations $\{f'(\beta_k)\}_{k\in[K]}$. Accordingly, the new (low degree) polynomial preserves the K intended computation results from the old (higher degree) polynomial, without revealing any information about their true values.

To do so, DLC builds on the following offline (data-agnostic) and online (data-dependent) phases. The offline phase is independent from the data, which can be performed in advance when the network load is low. The online phase depends on data, and is performed after training starts.

(Offline) In the offline phase, users first agree on $M+1$ distinct public parameters $\theta_1, \dots, \theta_{M+1} \in \mathbb{F}_p$ such that $\theta_k = \beta_k$ for all $k \in [K]$. Each user $i \in [N]$ then generates $M+1$ uniformly random matrices $\mathbf{R}_{i1}, \dots, \mathbf{R}_{i,M+1}$ of size $\frac{d_1}{N-T} \times d_2$ from $\mathbb{F}_p$, and forms a Lagrange polynomial of degree M,

$$\psi_i(z) = \sum_{k\in[M+1]} \mathbf{R}_{ik} \prod_{l\in[M+1]\setminus\{k\}} \frac{z-\theta_l}{\theta_k-\theta_l} \tag{10}$$

where $\psi_i(\theta_k) = \mathbf{R}_{ik}$ for all $k \in [M+1]$. User i then sends an encoded vector,

$$\widetilde{\mathbf{R}}_{ij} \triangleq \psi_i(\alpha_j) \tag{11}$$

to every other user $j \in [N]$. In addition, user i generates a second (lower degree) Lagrange polynomial of degree $K+T-1$,

$$\begin{aligned}\sigma_i(z) = &\sum_{k\in[K]} \mathbf{R}_{ik} \prod_{l\in[K+T]\setminus\{k\}} \frac{z-\beta_l}{\beta_k-\beta_l} \\ &+ \sum_{k\in\{K+1,\dots,K+T\}} \mathbf{Q}_{ik} \prod_{l\in[K+T]\setminus\{k\}} \frac{z-\beta_l}{\beta_k-\beta_l}\end{aligned} \tag{12}$$

where $\mathbf{Q}_{ik} \in \mathbb{F}_p^{\frac{d_1}{N-T} \times d_2}$ are uniformly random matrices for $k \in \{K+1, \dots, K+T\}$. Next, user i sends an encoded vector,

$$\overline{\mathbf{R}}_{ij} \triangleq \sigma_i(\alpha_j) \tag{13}$$

to every other user $j \in [N]$. After receiving $\{\widetilde{\mathbf{R}}_{ji}, \overline{\mathbf{R}}_{ji}\}_{j\in[N]}$, user $i \in [N]$ combines the received matrices to generate two new Lagrange polynomials,

$$\widetilde{\mathbf{R}}_i \triangleq \Big(\sum_{j\in[N]} \lambda_1^j \widetilde{\mathbf{R}}_{ji}^{\mathrm{T}}, \cdots, \sum_{j\in[N]} \lambda_{N-T}^j \widetilde{\mathbf{R}}_{ji}^{\mathrm{T}} \Big)^{\mathrm{T}} \tag{14}$$

$$= \sum_{k\in[M+1]} \mathbf{R}_k \prod_{l\in[M+1]\setminus\{k\}} \frac{\alpha_i-\theta_l}{\theta_k-\theta_l} \tag{15}$$

$$\overline{\mathbf{R}}_i \triangleq \Big(\sum_{j\in[N]} \lambda_1^j \overline{\mathbf{R}}_{ji}^{\mathrm{T}}, \cdots, \sum_{j\in[N]} \lambda_{N-T}^j \overline{\mathbf{R}}_{ji}^{\mathrm{T}} \Big)^{\mathrm{T}} \tag{16}$$

$$= \sum_{k\in[K]} \mathbf{R}_k \prod_{l\in[K+T]\setminus\{k\}} \frac{\alpha_i-\beta_l}{\beta_k-\beta_l} + \sum_{k\in\{K+1,\dots,K+T\}} \mathbf{Q}_k \prod_{l\in[K+T]\setminus\{k\}} \frac{\alpha_i-\beta_l}{\beta_k-\beta_l} \tag{17}$$

where $\mathbf{R}_k \triangleq (\sum_{j\in[N]} \lambda_1^j \mathbf{R}_{jk}^{\mathrm{T}}, \cdots, \sum_{j\in[N]} \lambda_{N-T}^j \mathbf{R}_{jk}^{\mathrm{T}})^{\mathrm{T}}$, and $\mathbf{Q}_k \triangleq (\sum_{j\in[N]} \lambda_1^j \mathbf{Q}_{jk}^{\mathrm{T}}, \cdots, \sum_{j\in[N]} \lambda_{N-T}^j \mathbf{Q}_{jk}^{\mathrm{T}})^{\mathrm{T}}$. In doing so, the key motivation is to generate high dimensional shared randomness using a lower dimensional random polynomial

generated by each user. Specifically, the dimension of the random vectors generated (and embedded in a Lagrange polynomial) by each user has size $\frac{d_1}{N-T} \times d_2$, whereas the size of the random vectors embedded in the Lagrange polynomials from (14) and (16) both have size $d_1 \times d_2$.

(Online) In the online phase, each user $i \in [N]$ broadcasts $f(\alpha_i) - \widetilde{\mathbf{R}}_i$, which can be viewed as an evaluation of a degree M monomial $h(z) = f(z) - \psi(z)$ at point $z = \alpha_i$, where

$$h(\alpha_i) = f(\alpha_i) - \psi(\alpha_i) = f(\alpha_i) - \widetilde{\mathbf{R}}_i \qquad \forall i \in [N], \tag{18}$$

$$h(\beta_k) = f(\beta_k) - \psi(\beta_k) = f(\beta_k) - \mathbf{R}_k \qquad \forall k \in [K] \tag{19}$$

such that the desired computations $f(\beta_k)$ are hidden by an additive random mask $\mathbf{R}_k = \psi(\beta_k)$, where

$$\psi(z) \triangleq \sum_{k\in[M+1]} \mathbf{R}_k \prod_{l\in[M+1]\setminus\{k\}} \frac{z-\theta_l}{\theta_k-\theta_l} \tag{20}$$

Then, after receiving $h(\alpha_i)$ from a set $\mathcal{I}$ of at least $|\mathcal{I}| \geq M+1$ users, every user can interpolate the polynomial $h(z)$, and compute $h(\beta_k) = f(\beta_k) - \psi(\beta_k)$ for all $k \in [K]$. Finally, each user $i \in [N]$ re-encodes the desired computations $\{f(\beta_k)\}_{k\in[K]}$ with a degree $K+T-1$ Lagrange polynomial,

$$f'(\alpha_i) = \sum_{k\in[K]} h(\beta_k) \prod_{k\in[K+T]\setminus\{k\}} \frac{\alpha_i-\beta_l}{\beta_k-\beta_l} + \overline{\mathbf{R}}_i \tag{21}$$

$$= \sum_{k\in[K]} f(\beta_k) \prod_{k\in[K+T]\setminus\{k\}} \frac{\alpha_i-\beta_l}{\beta_k-\beta_l} + \sum_{k\in\{K+1,\ldots,K+T\}} \mathbf{V}_k \prod_{l\in[K+T]\setminus\{k\}} \frac{\alpha_i-\beta_l}{\beta_k-\beta_l} \tag{22}$$

B. Details of the Model Updating Stage

For updating the model, in principle, one can collect the evaluations from at any set of $3(K+T-1)+1$ users to decode the true gradients using polynomial interpolation, and then sum them up. However, revealing the gradients can breach the privacy of local datasets. To address this, we propose a privacy-preserving model update mechanism with linear communication complexity. The key intuition is to decode a *masked version* of the K true gradients for each *coded gradient* (where the true gradient is hidden by additive random masks), aggregate them, and re-encode them with a Lagrange polynomial, while simultaneously ensuring the cancellation of all additive masks. At the end, each user $i \in [N]$ obtains a coded gradient $\widetilde{\mathbf{G}}_i^l$ that encodes the *sum of all gradients* $\sum_{i\in[m]} \nabla\mathcal{L}(\mathbf{W}(t); \mathbf{x}_i, \mathbf{y}_i)$ from the individual data points with a Lagrange polynomial of degree $K+T-1$, using which each user updates the model as $\widetilde{\mathbf{W}}_i^l \leftarrow \widetilde{\mathbf{W}}_i^l - \frac{\eta}{m}\widetilde{\mathbf{G}}_i^l$ for all $l \in [L+1]$.

Our model update process consists of the following offline and online phases.

(Offline) Users agree on $B \triangleq 3(K+T-1)+1$ public parameters $\{\mu_k\}_{k\in[B]}$ such that $\mu_k = \beta_k$ for $k \in [K]$, and $\{\mu_k\}_{k=K+1}^{B}$ are distinct public parameters. Then, user $i \in [N]$ constructs a Lagrange polynomial of degree $B-1$:

$$\mu_{bi}^l(z) = \sum_{k\in[B]} \mathbf{B}_{bik}^l \prod_{l\in[B]\setminus\{k\}} \frac{\alpha-\mu_l}{\mu_k-\mu_l} \tag{23}$$

for each $l \in [L+1]$ and $b \in [m/K]$, where $\mathbf{B}_{bik}^l \in \mathbb{F}_p^{\frac{d_l}{N-T}\times d_{l-1}}$ are generated independently uniformly at random, and sends an encoded mask $\widetilde{\mathbf{B}}_{bij}^l \triangleq \mu_{bi}^l(\alpha_j)$ to user $j \in [N]$. Upon receiving $\{\widetilde{\mathbf{B}}_{bji}^l\}_{j\in[N],b\in[m/K]}$, user $j \in [N]$ generates a (larger dimensional) encoded mask:

$$\widetilde{\mathbf{B}}_{bi}^l \triangleq \Big(\sum_{j\in[N]} \lambda_1^j (\widetilde{\mathbf{B}}_{bji}^l)^{\mathrm{T}}, \cdots, \sum_{j\in[N]} \lambda_{N-T}^j (\widetilde{\mathbf{B}}_{bji}^l)^{\mathrm{T}} \Big)^{\mathrm{T}} \tag{24}$$

for each $l \in [L+1]$ and $b \in [m/K]$, which can be viewed as an evaluation point of a degree $B-1$ polynomial,

$$\mu_b^l(\alpha) \triangleq \sum_{k\in[B]} \mathbf{B}_{bk}^l \prod_{l\in[B]\setminus\{k\}} \frac{\alpha-\mu_l}{\mu_k-\mu_l} \tag{25}$$

where $\mu_b^l(\alpha_i) = \widetilde{\mathbf{B}}_{bi}^l$ and $\mu_b^l(\beta_k) = \widetilde{\mathbf{B}}_{bk}^l$ such that

$$\mathbf{B}_{bk}^l \triangleq \Big(\sum_{j\in[N]} \lambda_1^j (\mathbf{B}_{bjk}^l)^{\mathrm{T}}, \cdots, \sum_{j\in[N]} \lambda_{N-T}^j (\mathbf{B}_{bjk}^l)^{\mathrm{T}} \Big)^{\mathrm{T}} \tag{26}$$

for all $k \in [B]$. In addition, user $i \in [N]$ constructs a smaller degree (degree $K+T-1$) Lagrange polynomial, by aggregating the locally generated randomness,

$$\tau_i^l(\alpha) \triangleq \sum_{k\in[K]} \Big(\sum_{b\in[m/K]} \sum_{k\in[K]} \mathbf{B}_{bik}^l \Big) \prod_{l\in[K+T]\setminus\{k\}} \frac{\alpha-\beta_l}{\beta_k-\beta_l} + \sum_{k=K+1}^{K+T} \mathbf{E}_{ik} \prod_{l\in[K+T]\setminus\{k\}} \frac{\alpha-\beta_l}{\beta_k-\beta_l} \tag{27}$$

for each $l \in [L+1]$, where $\{\mathbf{E}_{ik}\}_{k=K+1}^{K+T} \in \mathbb{F}_p^{\frac{d_l}{N-T}\times d_{l-1}}$ are generated independently and uniformly at random, and sends $\overline{\mathbf{B}}_{ij}^l \triangleq \tau_i^l(\alpha_j)$ to user $j \in [N]$. After receiving $\{\overline{\mathbf{B}}_{ji}^l\}_{j\in[N]}$, user $i \in [N]$ generates a larger dimensional encoded mask,

$$\overline{\mathbf{B}}_i^l \triangleq \Big(\sum_{j\in[N]} \lambda_1^j (\overline{\mathbf{B}}_{ji}^l)^{\mathrm{T}}, \cdots, \sum_{j\in[N]} \lambda_{N-T}^j (\overline{\mathbf{B}}_{ji}^l)^{\mathrm{T}} \Big)^{\mathrm{T}} \tag{28}$$

for each $l \in [L+1]$, which can be viewed as an evaluation point of a degree $K+T-1$ polynomial,

$$\tau^l(\alpha) \triangleq \sum_{k\in[K]} \mathbf{B}_k^l \prod_{l\in[K+T]\setminus\{k\}} \frac{\alpha-\beta_l}{\beta_k-\beta_l} + \sum_{k=K+1}^{K+T} \mathbf{E}_k^l \prod_{l\in[K+T]\setminus\{k\}} \frac{\alpha-\beta_l}{\beta_k-\beta_l} \tag{29}$$

where $\tau^l(\alpha_i) = \overline{\mathbf{B}}_i^l$, whereas $\tau^l(\beta_k) = \mathbf{B}_k^l$ for $k \in [K]$ and $\tau^l(\beta_k) = \mathbf{E}_k^l$ for $k \in \{K+1, \ldots, K+T\}$, such that

$$\mathbf{B}_k^l \triangleq \sum_{b\in[m/K]} \sum_{k'\in[K]} \mathbf{B}_{bk'}^l \tag{30}$$

$$\mathbf{E}_k^l \triangleq \Big(\sum_{j\in[N]} \lambda_1^j (\mathbf{E}_{jk}^l)^{\mathrm{T}}, \cdots, \sum_{j\in[N]} \lambda_{N-T}^j (\mathbf{E}_{jk}^l)^{\mathrm{T}} \Big)^{\mathrm{T}} \tag{31}$$

(Online) In the online phase, each user $i \in [N]$ broadcasts,

$$\widehat{\mathbf{G}}_{bi}^l = \widetilde{\mathbf{G}}_{bi}^l - \widetilde{\mathbf{B}}_{bi}^l \tag{32}$$

for each $l \in [L+1]$ and $b \in [m/K]$. Note that (32) can be viewed as evaluation points of a degree $B-1$ polynomial $r_b^l(\alpha) - \tau_b^l(\alpha)$ where $\widehat{\mathbf{G}}_{bi}^l = r_b^l(\alpha_i) - \tau_b^l(\alpha_i)$ for $i \in [N]$. As such, after receiving (32) from any set of at least B users, each user can decode the masked gradient $r_b^l(\beta_k) - \mu_b^l(\beta_k) = r_b^l(\beta_k) - \mathbf{B}_{bk}^l$ for all $k \in [K]$ and $b \in [m/K]$, where the true gradient $r_b^l(\beta_k)$ is hidden by the mask $\mathbf{B}_{bk}^l$. Finally, each user $i \in [N]$ locally aggregates and re-encodes the masked gradients by forming a degree $K+T-1$ Lagrange polynomial,

$$\widetilde{\mathbf{G}}_i^l \triangleq \sum_{k\in[K]} \Big(\sum_{b\in[\frac{m}{K}]} \sum_{k'\in[K]} (r_b^l(\beta_{k'}) - \mathbf{B}_{bk}^l) \Big) \prod_{l\in[K+T]} \frac{\alpha_i-\beta_l}{\beta_k-\beta_l} + \overline{\mathbf{B}}_i^l \tag{33}$$

$$= \sum_{k\in[K]} \Big(\sum_{b\in[\frac{m}{K}]} \sum_{k'\in[K]} r_b^l(\beta_{k'}) \Big) \prod_{l\in[K+T]} \frac{\alpha_i-\beta_l}{\beta_k-\beta_l} + \sum_{k=K+1}^{K+T} \mathbf{E}_k^l \prod_{l\in[K+T]\setminus\{k\}} \frac{\alpha_i-\beta_l}{\beta_k-\beta_l} \tag{34}$$

where the additive masks $\mathbf{B}_{bk}^l$ cancel out in (33), hence (34) encodes the *sum gradient* $\sum_{i\in[m]} \nabla\mathcal{L}(\mathbf{W}(t); \mathbf{x}_i, \mathbf{y}_i) = (\sum_{b\in[\frac{m}{K}]} \sum_{k'\in[K]} r_b^l(\beta_{k'})$ over all data points as shown in (2).

C. Information-Theoretic Privacy

Proof. The privacy of the dataset and label encoding follows from (So et al., 2020). In the following, we show that the degree reduction operation $\phi(\cdot)$ preserves information-theoretic privacy for any arbitrary function $f(\cdot)$. From (11), (13), and

(18),

$$I(\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in\mathcal{H}};\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{i\in\mathcal{H},j\in\mathcal{T}},\{f(\alpha_i)-\widetilde{\mathbf{R}}_i\}_{i\in[N]},$$
$$\{\mathbf{R}_{ik}\}_{i\in\mathcal{T},k\in[M+1]},\{\mathbf{Q}_{ik}\}_{i\in\mathcal{T},k\in\{K+1,\dots,K+T\}}|\mathcal{M},\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in\mathcal{T}},\mathbf{W}(J)) \tag{35}$$
$$= I(\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in\mathcal{H}};\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{\substack{i\in\mathcal{H},\\ j\in\mathcal{T}}},\{f(\theta_k)-\mathbf{R}_k\}_{k\in[M+1]},$$
$$\{\mathbf{R}_{ik}\}_{i\in\mathcal{T},k\in[M+1]},\{\mathbf{Q}_{ik}\}_{i\in\mathcal{T},k\in\{K+1,\dots,K+T\}}|\mathcal{M},\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in\mathcal{T}},\mathbf{W}(J)) \tag{36}$$
$$= H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{\substack{i\in\mathcal{H},\\ j\in\mathcal{T}}},\{f(\theta_k)-\mathbf{R}_k\}_{k\in[M+1]},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},\{\mathbf{Q}_{ik}\}_{i\in\mathcal{T},k\in\{K+1,\dots,K+T\}}|\mathcal{M},\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in\mathcal{T}},\mathbf{W}(J))$$
$$- H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{\substack{i\in\mathcal{H},\\ j\in\mathcal{T}}},\{f(\theta_k)-\mathbf{R}_k\}_{k\in[M+1]},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},\{\mathbf{Q}_{ik}\}_{i\in\mathcal{T},k\in\{K+1,\dots,K+T\}}|\mathcal{M},\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in[N]},\mathbf{W}(J)) \tag{37}$$

where $\mathcal{M}\triangleq(\mathcal{M}_{\mathcal{T}}^1,\mathcal{M}_{\mathcal{T}}^2,\mathcal{M}_{\mathcal{T}}^3,\mathcal{M}_{\mathcal{T}}^{4,1},\dots,\mathcal{M}_{\mathcal{T}}^{4,t-1})$ in (35), and (36) follows from the fact that there is a bijective mapping from any $M+1$ evaluation points $\{f(\theta_k)-\mathbf{R}_k\}_{k\in[M+1]}$ to a valid (feasible) set of local computations $\{f(\alpha_i)-\widetilde{\mathbf{R}}_i\}_{i\in[N]}$, since the local computations in (18) correspond to evaluations of a degree M polynomial $h(z)=f(z)-\psi(z)$, which can be uniquely reconstructed from any set of at least $M+1$ evaluation points. For the first term in (37), we find that,

$$H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{i\in\mathcal{H},j\in\mathcal{T}},\{f(\theta_k)-\mathbf{R}_k\}_{k\in[M+1]},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},$$
$$\{\mathbf{Q}_{ik}\}_{i\in\mathcal{T},k\in\{K+1,\dots,K+T\}}|\mathcal{M},\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in\mathcal{T}},\mathbf{W}(J))$$
$$= H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{i\in\mathcal{H},j\in\mathcal{T}},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},$$
$$\Big\{f(\theta_k)-\Big(\sum_{i\in[N]}\lambda_1^{i-1}\mathbf{R}_{ik}^T,\dots,\sum_{i\in[N]}\lambda_{N-T}^{i-1}\mathbf{R}_{ik}^T\Big)^T\Big\}_{k\in[M+1]},$$
$$\{\mathbf{Q}_{ik}\}_{i\in\mathcal{T},k\in\{K+1,\dots,K+T\}}|\mathcal{M},\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in\mathcal{T}},\mathbf{W}(J)) \tag{38}$$
$$= H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{i\in\mathcal{H},j\in\mathcal{T}},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},$$
$$\Big\{f(\theta_k)-\Big(\sum_{i\in[N-T]}\lambda_1^{i-1}\mathbf{R}_{ik}^T,\dots,\sum_{i\in[N-T]}\lambda_{N-T}^{i-1}\mathbf{R}_{ik}^T\Big)^T\Big\}_{k\in[M+1]},$$
$$\{\mathbf{Q}_{ik}\}_{i\in\mathcal{T},k\in\{K+1,\dots,K+T\}}|\mathcal{M},\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in\mathcal{T}},\mathbf{W}(J)) \tag{39}$$
$$\leq (T+D+1)d_l\Big(1+\frac{T}{N-T}\Big)\log p \tag{40}$$

where (39) holds since,

$$(\widetilde{\mathbf{R}}_{1j},\dots,\widetilde{\mathbf{R}}_{N-T,j})=\Big(f(\alpha_j)-\sum_{k\in[D+1]}\big(f(\theta_k)-$$
$$\big(\sum_{i\in[N-T]}\lambda_1^{i-1}\mathbf{R}_{ik}^T,\dots,\sum_{i\in[N-T]}\lambda_{N-T}^{i-1}\mathbf{R}_{ik}^T\big)^T\big)\prod_{l\in[D+1]\setminus\{k\}}\frac{\alpha_j-\theta_l}{\theta_k-\theta_l}\Big)\mathbf{M}^{-1}$$

such that $\mathbf{M}$ is an $(N-T)\times(N-T)$ MDS matrix (hence invertible),

$$\mathbf{M}=\begin{bmatrix}1 & \cdots & 1\\ \lambda_1 & \cdots & \lambda_{N-T}\\ \vdots & \ddots & \vdots\\ \lambda_1^{N-T-1} & \cdots & \lambda_{N-T}^{N-T-1}\end{bmatrix} \tag{41}$$

and that the local computations $\{f(\alpha_j)\}_{j\in\mathcal{T}}$ are already known by the adversaries, i.e., $\{f(\alpha_j)\}_{j\in\mathcal{T}}\in\mathcal{M}$. Finally, (40) holds since conditioning cannot increase entropy, and that entropy is maximized by the uniform distribution. For the second

term in (37), we find that,

$$
\begin{aligned}
&H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{i\in\mathcal{H},j\in\mathcal{T}},\{f(\theta_k)-\mathbf{R}_k\}_{k\in[M+1]},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},\\
&\qquad\{\mathbf{Q}_{ik}\}_{i\in\mathcal{T},k\in\{K+1,\ldots,K+T\}}|\mathcal{M},\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in[N]},\mathbf{W}(J))\\
&\geq H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{i\in\mathcal{H},j\in\mathcal{T}},\{f(\theta_k)-\mathbf{R}_k\}_{k\in[M+1]},\\
&\qquad\{\mathbf{R}_{ik}\}_{i\in\mathcal{T},k\in[M+1]},\{\mathbf{Q}_{ik}\}_{i\in\mathcal{T},k\in\{K+1,\ldots,K+T\}}|\mathcal{M},\\
&\qquad\{\mathbf{X}_i,\mathbf{Y}_i\}_{i\in[N]},\mathbf{W}(J),\{f(\theta_k)\}_{k\in[M+1]}) &(42)\\
&=H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{i\in\mathcal{H},j\in\mathcal{T}},\{\mathbf{R}_k\}_{k\in[M+1]},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},\\
&\qquad\{\mathbf{Q}_{ik}\}_{i\in\mathcal{T},k\in\{K+1,\ldots,K+T\}}) &(43)\\
&=H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{\substack{i\in\mathcal{H},\\ j\in\mathcal{T}}},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},\{\mathbf{Q}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in\{K+1,\ldots,K+T\}}},\\
&\qquad\big\{\big(\sum_{i\in[N]}\lambda_1^{i-1}\mathbf{R}_{ik}^T,\ldots,\sum_{i\in[N]}\lambda_{N-T}^{i-1}\mathbf{R}_{ik}^T\big)^T\big\}_{k\in[M+1]}\big)\\
&=H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{\substack{i\in\mathcal{H},\\ j\in\mathcal{T}}},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},\{\mathbf{Q}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in\{K+1,\ldots,K+T\}}},\\
&\qquad\big\{\big(\sum_{i\in[N-T]}\lambda_1^{i-1}\mathbf{R}_{ik}^T,\ldots,\sum_{i\in[N-T]}\lambda_{N-T}^{i-1}\mathbf{R}_{ik}^T\big)^T\big\}_{k\in[M+1]}\big)\\
&=H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{\substack{i\in\mathcal{H},\\ j\in\mathcal{T}}},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},\{\mathbf{Q}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in\{K+1,\ldots,K+T\}}},\\
&\qquad\{(\mathbf{R}_{1k},\ldots,\mathbf{R}_{N-T,k})\mathbf{M}\}_{k\in[M+1]}) &(44)\\
&=H(\{\widetilde{\mathbf{R}}_{ij},\bar{\mathbf{R}}_{ij}\}_{\substack{i\in\mathcal{H},\\ j\in\mathcal{T}}},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},\{\mathbf{Q}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in\{K+1,\ldots,K+T\}}},\\
&\qquad\{(\mathbf{R}_{1k},\ldots,\mathbf{R}_{N-T,k})\}_{k\in[M+1]}) &(45)\\
&=H(\{\bar{\mathbf{R}}_{ij}\}_{\substack{i\in\mathcal{H},\\ j\in\mathcal{T}}},\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},\{\mathbf{Q}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in\{K+1,\ldots,K+T\}}},\\
&\qquad\{(\mathbf{R}_{1k},\ldots,\mathbf{R}_{N-T,k})\}_{k\in[M+1]}) &(46)\\
&=H(\{\bar{\mathbf{R}}_{ij}\}_{\substack{i\in\mathcal{H},\\ j\in\mathcal{T}}}|\{\mathbf{R}_{ik}\}_{i\in[N-T],k\in[M+1]})\\
&\qquad+H(\{\mathbf{R}_{ik}\}_{i\in[N-T],k\in[M+1]})\\
&\qquad+H(\{\mathbf{R}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in[M+1]}},\{\mathbf{Q}_{ik}\}_{\substack{i\in\mathcal{T},\\ k\in\{K+1,\ldots,K+T\}}}) &(47)\\
&=(D+T+1)d_l\Big(1+\frac{T}{N-T}\Big)\log p &(48)
\end{aligned}
$$

where (42) holds since conditioning cannot increase entropy; (43) follows from the independence of randomness generated; (44) follows from (41); (45) holds since $\mathbf{M}$ is a $(N-T)\times(N-T)$ MDS matrix (hence is invertible); (46) holds since $\{\widetilde{\mathbf{R}}_{ij}\}_{j\in\mathcal{T}}$ can be perfectly reconstructed from $\{\mathbf{R}_{ik}\}_{k\in[M+1]}$ using (10) for all $i\in\mathcal{H}$; (47) follows from the independence of randomness generated; (48) follows from the entropy of uniform random variables, along with,

$$
\begin{aligned}
&H(\{\bar{\mathbf{R}}_{ij}\}_{\substack{i\in\mathcal{H},\\ j\in\mathcal{T}}}|\{\mathbf{R}_{ik}\}_{i\in[N-T],k\in[M+1]})\\
&=\sum_{i\in\mathcal{H}}H(\{\sum_{k\in\{K+1,\ldots,K+T\}}\mathbf{Q}_{ik}\rho_{jk}\}_j\in\mathcal{T}) &(49)\\
&=\sum_{i\in\mathcal{H}}H((\mathbf{Q}_{i,K+1},\ldots,\mathbf{Q}_{i,K+T})\mathbf{\Gamma}) &(50)\\
&=\sum_{i\in\mathcal{H}}H(\mathbf{Q}_{i,K+1},\ldots,\mathbf{Q}_{i,K+T}) &(51)
\end{aligned}
$$

where ρ_{jk} denotes the Lagrange coefficient,

$$\rho_{jk} \triangleq \prod_{l\in[K+T]\setminus\{k\}} \frac{\alpha_j - \beta_l}{\beta_k - \beta_l} \tag{52}$$

for all $j \in [N]$ and $k \in [K+T]$, and $\mathbf{\Gamma}$ is a $(N-T)\times(N-T)$ MDS matrix,

$$\mathbf{\Gamma} \triangleq \begin{bmatrix} \rho_{N-T+1,K+1} & \cdots & \rho_{N,K+1} \\ \vdots & \ddots & \vdots \\ \rho_{N-T+1,K+T} & \cdots & \rho_{N,K+T} \end{bmatrix} \tag{53}$$

hence is invertible. Finally, by combining (48) and (40) with (37),

$$\begin{aligned} 0 \leq & I(\{\mathbf{X}_i, \mathbf{Y}_i\}_{i\in\mathcal{H}}; \{\widetilde{\mathbf{R}}_{ij}, \bar{\mathbf{R}}_{ij}\}_{i\in\mathcal{H},j\in\mathcal{T}}, \{f(\alpha_i) - \widetilde{\mathbf{R}}_i\}_{i\in[N]}, \\ & \{\mathbf{R}_{ik}\}_{i\in\mathcal{T},k\in[M+1]}, \{\mathbf{Q}_{ik}\}_{i\in\mathcal{T},k\in\{K+1,\ldots,K+T\}} | \mathcal{M}, \{\mathbf{X}_i, \mathbf{Y}_i\}_{i\in\mathcal{T}}, \mathbf{W}(J)) \leq 0 \end{aligned} \tag{54}$$

The steps for consecutive degree reduction operations, along with the model update follows the same lines (as they build on the same principles as our degree reduction mechanism), which completes the proof.

□

D. Additional Experimental Details

Setup. Our study focuses on training a fully connected neural network architecture. This architecture consists of an input layer, followed by a hidden layer comprising of 128 neurons, and finally a classification layer. To evaluate the performance of our model, we conduct experiments using two distinct datasets: Fashion MNIST (Xiao et al., 2017), and CIFAR-10 (Krizhevsky et al., 2009). For the CIFAR-10 dataset, we utilize a pre-trained VGG (Simonyan & Zisserman, 2015) model for feature extraction as a part of the neural network implementation. Additionally, we evenly distribute all the datasets across the clients.

Benchmark. For the baseline for target accuracy, we conducted experiments to evaluate the model accuracy of the two-layer neural network architecture trained in the real number domain, by utilizing ReLU activation functions (Agarap, 2019) and Adam optimizer (Kingma & Ba, 2017). Additionally, we implement COPML from (So et al., 2020) to assess the communication overhead, which serves as a benchmark for private computation without revealing the inputs. We note that the gradient computations and model update are the same for both COPML and CLOVER.

Performance evaluation. To correctly recover the final model, the number of clients must satisfy the recovery threshold from Thm. 4.3. where the degree of privacy (T) and parallelization (K) are calculated, by letting $N = 3(K+T-1)+1$ with $T = \lfloor \frac{N}{16} \rfloor$ and $K = \lfloor \frac{N}{8} \rfloor - T$. Similar to (So et al., 2020), the bandwidth and finite field size are set as 40Mbps and $q = 2^{26} - 5$, respectively.