PRECODE - A Generic Model Extension to Prevent Deep Gradient Leakage

Daniel Scheliga[1]   Patrick Mäder[1,2]

Marco Seeland[1]

[1]Technische Universität Ilmenau, [2]Friedrich Schiller Universität Jena, Germany

{daniel.scheliga,patrick.maeder,marco.seeland}@tu-ilmenau.de

Abstract

Collaborative training of neural networks leverages distributed data by exchanging gradient information between different clients. Although training data entirely resides with the clients, recent work shows that training data can be reconstructed from such exchanged gradient information. To enhance privacy, gradient perturbation techniques have been proposed. However, they come at the cost of reduced model performance, increased convergence time, or increased data demand. In this paper, we introduce PRECODE, a PRivacy EnhanCing mODulE that can be used as generic extension for arbitrary model architectures. We propose a simple yet effective realization of PRECODE using variational modeling. The stochastic sampling induced by variational modeling effectively prevents privacy leakage from gradients and in turn preserves privacy of data owners. We evaluate PRECODE using state of the art gradient inversion attacks on two different model architectures trained on three datasets. In contrast to commonly used defense mechanisms, we find that our proposed modification consistently reduces the attack success rate to 0% while having almost no negative impact on model training and final performance. As a result, PRECODE reveals a promising path towards privacy enhancing model extensions.

1. Introduction

In application domains like medical imaging, data sharing is typically prohibited due to privacy, legal, and technical regulations [17]. Avoiding the need for shared data, collaborative training such as federated learning allows multiple clients to collaboratively compute a common model [20]. This collaborative use of distributed data can result in a severe boost of model performance, while mitigating a lot of systemic privacy risks and costs inherent in model training on centrally aggregated data [11].

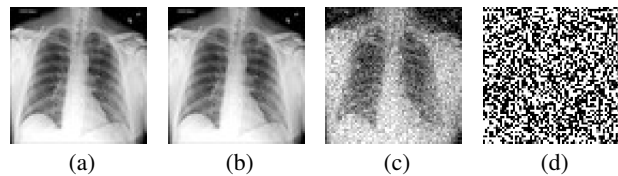Although training data entirely resides with the respec-

Figure 1: **Privacy leakage by reconstructing training data from gradients.** (a) Original image. (b) Reconstructed image. (c) Image reconstructed from perturbed gradient information. (d) Prevented reconstruction using our proposed privacy-enhanced model architecture.

tive clients, recent studies demonstrated the possibility for compromising clients' privacy by reconstructing sensitive information from exchanged model parameters. The most advanced techniques for such privacy leakage are gradient inversion attacks [8, 31, 32, 34] that allow to reconstruct training data from exchanged model weights or gradients. Exemplary reconstructions are shown in Fig. 1. Geiping *et al.* [8] show that the input to any fully connected layer in a neural network can be analytically reconstructed given the model architecture and associated gradients. As basically all state of the art deep learning architectures make use of fully connected layers, the privacy of individual data owners is clearly exposed to such attacks.

Since such privacy leaks are based on exchanged gradients, [31, 34] propose to perturb gradients before sharing. As a side effect an inherent trade-off between privacy and model performance can be observed [10, 31, 34].

To avoid this trade-off, we propose to enhance the architecture of arbitrary models by inserting PRECODE, a *PRivacy EnhanCing mODulE*. The idea of PRECODE is to use variational modeling to disguise the original latent feature space vulnerable to privacy leakage by gradient inversion attacks. We perform a systematic evaluation to investigate the privacy enhancing properties of our proposed model extension. In contrast to gradient perturbation based defense mechanisms, we show that PRECODE has minimal impact on the training process and model performance.
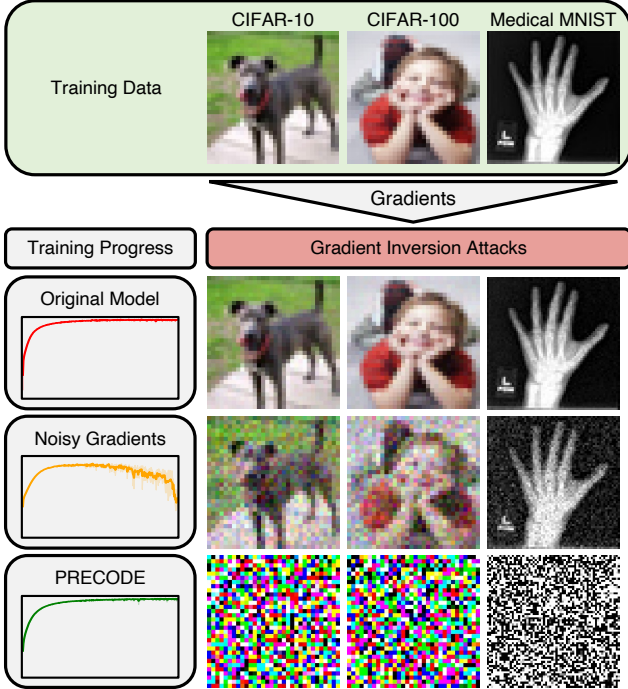
Figure 2: **Summary of the content of this paper.** Neural networks are trained on CIFAR-10, CIFAR-100, and Medical MNIST. During training, gradients are used for inversion attacks aiming to reconstruct original training data. Without defense (Original Model), training data can be entirely reconstructed. Noisy gradients prevent reconstruction to some extent but reduce model performance. Our proposed PRECODE extension entirely prevents reconstruction without impacting the training process.

Fig. 2 displays a visual summary of the content of this paper. Our contributions can be summarized as follows:

- To the best of our knowledge, we are first to propose specific model extensions to prevent privacy leakage.

- We propose a generic privacy enhancing module for defending gradient inversion attacks.

- We perform a systematic empirical evaluation using state of the art gradient inversion attacks on two model architectures trained on three image classification datasets and compare our method with two state of the art defense methods.

2. Related Work

2.1. Gradient Inversion Attacks

Gradient inversion attacks as studied in [8, 31, 32, 34] aim to reconstruct input training data x from their corresponding gradients $\nabla\mathcal{L}_\theta(\hat{y}, y)$, where θ are the parameters of the attacked model F, $\hat{y} = F(x)$ is the models prediction given the input x, y the expected output for x and $\mathcal{L}$ the loss function used to optimize θ.

We follow the threat model described by [8], *i.e.*, an *honest-but-curious* server. In detail, an attacker has insight into the training process, *i.e.*, knowledge of $\mathcal{L}$, F, θ and $\nabla\mathcal{L}_\theta(\hat{y}, y)$, but does not actively interfere with it (*honesty*). Instead, the attacker aims to reconstruct training data (x, y) of another client participating in the training process (*curiosity*). This scenario is particularly relevant for collaborative training processes such as *federated learning* [11, 20, 26], where a shared model is collaboratively trained by participating clients. Data is deliberately not shared, but remains with each respective client. A central aggregator distributes a global model to clients, which in turn respond with model updates, *i.e.*, gradients, after local training. The clients' gradients are then aggregated in order to update the global model. This process is iteratively repeated until convergence or another termination criterion is reached. Apart from star-based connectivity pattern with a central aggregator, other connectivity and aggregation patterns such as peer-to-peer or cluster-based collaborative training have been proposed within the federated learning paradigm [6, 16, 23] Since gradient information must also be exchanged in these approaches, they suffer from similar vulnerabilities.

Using knowledge about $\mathcal{L}$, F, θ and $\nabla\mathcal{L}_\theta(\hat{y}, y)$, gradient inversion attacks can generally be formulated as an optimization problem. As described for *Deep Leakage from Gradients* (DLG) [34], the basic idea is to feed a random dummy image x' into a model F to obtain a dummy gradient $\nabla\mathcal{L}_\theta(F(x'), y')$ by comparison to a randomly initialized dummy label y'. To reconstruct both the original input data x and the associated output y, the Euclidean distance

$$\underset{(x', y')}{\arg\min} ||\nabla\mathcal{L}_\theta(F(x), y) - \nabla\mathcal{L}_\theta(F(x'), y')||^2 \quad (1)$$

between the attacked gradient and the dummy gradient is iteratively minimized using a L-BFGS optimizer [19]. In *improved DLG* (iDLG) [32], Zhao *et al.* found that ground-truth labels y can be analytically reconstructed from gradient information. Without the need to optimize for y' in Eq. 1, optimization becomes faster and more stable. Wei *et al.* propose the *Client Privacy Leakage* (CPL) attack by extending the optimization objective of iDLG by an additive label-matching regularization term

$$\alpha ||F(x') - y||^2 \quad (2)$$

and found it to further stabilize the optimization [31].

Using *inverting gradient attacks* (IGA), Geiping *et al.* [8] further improve the reconstruction process by disentangling gradient magnitude and direction. They propose to minimize cosine distance instead of Euclidean distance between

attacked and dummy gradients in Eq. 1 while using the total variation [24] of the dummy image x' as regularization to constrain the reconstruction to relevant image parts. In addition, they found the Adam optimizer [12] to generally yield more sophisticated reconstruction results compared to the L-BFGS optimizer, especially for deeper models and trained models with overall smaller gradient magnitude.

2.2. Defense Mechanisms

The authors of CPL [31] present a comprehensive analysis of privacy leakage and deduced relevant parameters and potential mitigation strategies. They found that batch size, image resolution, choice of activation functions, and the number of local iterations before gradient exchange can impact privacy leakage. Supporting findings are reported in [8, 22, 32, 33, 34]. Also the training progress was found to impact privacy leakage from gradients, as models trained for more epochs typically yield smaller gradients compared to earlier in the training process [8, 29]. Whereas such parameters and conditions are certainly relevant and could be carefully selected to prevent attacks, they give no guarantee that reconstruction of sensitive data by gradient inversion attacks is not possible [8]. In fact, Geiping *et al.* showcased successful attacks on deep neural networks (ResNet-152) trained for multiple federated averaging steps and even for batches of 100 images [8]. In addition, parameter selection is often controlled by other factors, such as model and/or hardware limitations. In conclusion, data privacy has to be actively controlled by purposeful defense mechanisms.

Cryptographic approaches like homomorphic encryption [3] and secure aggregation schemes [5] potentially allow for the best privacy preservation. However, they come at extreme computational costs and are not applicable in every collaborative learning scenario [34].

Instead of costly cryptographic approaches, most state of the art methods defend against privacy leakage by immediate perturbation of exchanged gradients. Therefore, three approaches are typically utilized: *gradient quantization*, *gradient compression*, and *noisy gradients*.

Gradient quantization is primarily used to reduce communication costs and memory consumption during collaborative training [14, 25]. For quantization, fixed ranges of numerical values are compressed to sets of values, reducing the entropy of the quantized data. As a side effect, the success of inversion attacks on quantized gradients is also reduced. Yet the authors of [34] found that half-precision quantization [27] fails to protect training data. Only low-bit Int-8 quantization was able to defend against inversion attacks, but also caused a 22.6% drop in model accuracy [34].

Gradient compression/sparsification by pruning was also introduced to reduce the communication costs during collaborative training [18]. Here elements of the gradient that carry the least information, *i.e.*, have the smallest

magnitude, are pruned to zero. Like quantization, this reduces the entropy of the gradient and decreases reconstruction success by inversion attacks. Zhu *et al.* found a pruning ratio of at least 20% sufficient to suppress the reconstruction success of the input data [34].

The idea of using noise to limit the information disclosure about individuals was originally introduced in the field of differential privacy [1, 7, 11]. State of the art methods use *noisy gradients* to guarantee a provable degree of privacy to clients participating in collaborative training [5, 17, 21]. In practice, Gaussian or Laplace noise is added to the gradients prior to their exchange. Whereas added noise can suppress gradient inversion attacks, it also negatively impacts both the training process and the final model performance. Wei *et al.* demonstrate that privacy leakage from gradients can be prevented at the cost of a severe drop in model accuracy of up to 9.8% [31]. Such performance drop can be mitigated by an increase of training data [7] which is typically not feasible in practical scenarios.

3. PRECODE - Privacy Enhancing Module

As discussed above, commonly used defense mechanisms rely on gradient perturbations. Although these mechanisms are shown to reduce privacy leakage by gradient inversion attacks, they induce an inherent trade-off between privacy and model performance. Apart from increased model depth, batch size, and image resolution, to the best of our knowledge, no deliberate architectural modifications have been proposed to reduce privacy leakage. Hence, we propose to extend neural network architectures with a PRivacy EnhanCing mODulE (PRECODE). Next, we summarize the key requirements for such module.

3.1. Requirements for PRECODE

Privacy enhancement The PRECODE extension needs to sufficiently disguise training data information within gradients in order to prevent privacy leakage by gradient inversion attacks.

Generic model integrability Targeting a generalized solution, the PRECODE extension should not be limited to certain model architectures but be applicable to arbitrary model architectures. For ease of integration and joint optimization, we also aim for direct integration of the extension into an existing model architecture without requiring further model modifications.

Model performance For such an extension to be practically relevant, it should not harm the final model performance or the training process, *e.g.*, by causing increased convergence times.
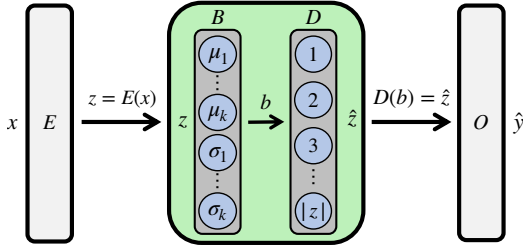
Figure 3: Realization of the PRECODE extension as variational bottleneck.

3.2. Realization using Variational Modeling

Analyzing the criteria described in section 3.1 and inspired by the work of [2, 13], we propose the use of a variational bottleneck (VB) as realization of the PRECODE extension. Aiming to learn a joint distribution between input data and its latent representation, VBs originate from the field of generative modeling [13]. They consist of a probabilistic encoder $B = q(b|z)$ and decoder $D = p(z|b)$. Fig. 3 visualizes the VB and corresponding input and output variables. We use the VB to approximate the distribution of the latent space and obtain a new representation $\hat{z}$ from an approximated prior distribution $p(b)$ by stochastic sampling.

Hence, the privacy enhancing properties of the VB originate from the fact that the returned latent representation $\hat{z}$ is a *stochastic function* of the input representation z. As $\hat{y}$ and $\mathcal{L}(\hat{y}, y)$ are computed based on this stochastic representation, the gradient $\nabla \mathcal{L}(\hat{y}, y)$ does not contain direct information on the input sample x. Furthermore, iterative optimization-based attacks are countered by design, since $z' = E(x')$ is sampled randomly in each optimization step. Small changes in x' cause increased entropy of z', therefore making it difficult for the optimizer to find dummy images x' that minimize the reconstruction loss.

We realize B as a fully connected layer with $2k$ neurons, where k defines the size of the bottleneck representation b. Input to the encoder are the hidden representations $z = E(x)$ computed by forward propagation of an input sample x through all layers E of the base neural network prior to the output layer. B then encodes the representation z into a latent distribution $B(z) = \mathcal{N}(\mu_B, \sigma_B)$ approximated as multivariate Gaussian distribution. The bottleneck features $b \sim q(b|z)$ are fed into the stochastic decoder D, also realized as fully connected layer. The number of neurons in D is equal to the number of features in z. D generates a new representation $\hat{z} = p(z|b)p(b)$ which is then used to calculate the model prediction $\hat{y} = O(\hat{z})$. O corresponds to the layer(s) of the neural network that follow the VB.

For the VB to learn a continuous and complete latent feature space distribution, the loss function $\mathcal{L}_F$ of the base model F is extended by the Kullback-Leibler divergence

(D_{KL}) between $\mathcal{N}(\mu_B, \sigma_B)$ and a standard normal distribution where β controls the weight of the VB loss on the overall loss function:

$$\mathcal{L}(\hat{y}, y) = \mathcal{L}_F(\hat{y}, y) + \beta \cdot D_{KL}(\mathcal{N}(\mu_B, \sigma_B), \mathcal{N}(0, 1)). \quad (3)$$

In order to backpropagate a gradient through the bottleneck layer, the reparameterization technique described in [13] is used. We discuss the impact of hyperparameter selection and placement of the VB in the *supplementary material*. We provide a PyTorch implementation of PRECODE[1].

4. Experimental Setup

This section describes the experimental settings of the systematic empirical studies conducted to evaluate the behaviour of the proposed PRECODE realization on different model architectures, datasets, and attacks, with respect to model performance and privacy of training data.

We evaluate privacy leakage by reconstructing training images using the state of the art gradient inversion attacks IGA [8] and CPL [31]. For comparative evaluation of PRECODE, we also defend inversion attacks using gradient perturbation techniques, *i.e.*, noisy gradients [1] and gradient compression [18]. Gradient quantization is not considered due to the comparatively large negative impact on model performance [34]. Model performance is evaluated by training both fully connected as well as convolutional neural networks for image classification on three different datasets, *i.e.*, CIFAR-10, CIFAR-100, and Medical MNIST.

The CIFAR-10 dataset [15] is composed of $60'000$ colored images with a size of 32×32 pixels. Each image displays one out of ten classes and the amount of images per class is equally distributed. CIFAR-100 [15] represents a harder variant, using same amount of images and same image size but displaying 100 different classes. Both datasets are separated into training and test splits according to the corresponding benchmark protocols. The Medical MNIST dataset [4] comprises six classes of 64×64 sized grayscale images. Whereas one class contains $8'954$ images, each other class is represented by $10'000$ images. The dataset is randomly split into 80% training and 20% test split.

We consider three baseline model architectures with increasing complexity: a shallow MLP (SMLP), deep MLP (DMLP), and LeNet. MLPs are especially considered as Geiping *et al*. theoretically proof that the input to fully connected layers can generally be reconstructed from their gradient [8]. The MLPs are composed of fully connected layers with 1024 neurons and ReLU activation. Whereas the SMLP uses two hidden layers, the DMLP uses four hidden layers. The output layer is a fully connected layer with softmax activation, and the number of neurons is equal to the number of classes within a dataset. In addition to MLPs,

---
[1]https://github.com/SECSY-Group/PRECODE

we use the LeNet implementation from [32] as prototypical architecture for convolutional neural networks. To investigate the privacy enhancing properties of our proposed method, we extend these baseline architectures by placing PRECODE with $k = 256$ and $\beta = 0.001$ between the last feature extracting layer and the final classification layer.

To evaluate the model performance and convergence time we train all randomly initialized models for 300 epochs while minimizing the cross-entropy loss using Adam [12] optimizer with learning rate 0.001, momentum parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and a batch size of 64. For the PRECODE extended models, the loss function is adjusted according to Eq. 3. Training and test accuracy are measured after each epoch as average across all train/test samples, respectively. For experiments with gradient perturbation defense mechanisms, the gradients of the cross-entropy loss are perturbed in each optimization step. All reported metrics are averaged across three runs with different seeds.

For evaluating privacy leakage, we randomly sample a *victim dataset* composed of 128 images from training data. We ensured that each class is represented in the victim dataset. We follow the attack setup as in [31, 32, 34] and simulate a single training step to generate the victim gradient using corresponding victim data. For reconstruction we use the publicly available PyTorch implementation of IGA[2]. We use the default configuration for the attack, *i.e.*, initialization of dummy images from Gaussian distribution, cosine distance based loss function with $\alpha = 10^{-6}$ as total variation regularization weight, Adam optimizer with learning rate 0.01, and step-wise learning rate decay [8]. We found that high quality reconstructions can be generated within $7'000$ iterations. We stopped attacks if there is no decrease in reconstruction loss for $1'200$ iterations.

Following the findings in [8, 31, 32], we assume label information for each reconstruction to be known as it can be analytically reconstructed from gradients of cross-entropy loss functions w.r.t. weights of fully connected layers with softmax activation (cf. section 2.1). Please note that the adjusted loss function of PRECODE might impact such analytical label reconstruction. However, we assume label information also to be known when attacking PRECODE enhanced models. Otherwise our proposed method might have systematic advantages if attacks would need to jointly optimize for images and their labels.

Similar to [8, 31], we compute *mean squared error* (MSE), *peak signal-to-noise ratio* (PSNR) and *structural similarity* (SSIM) [30] between original and reconstructed images to measure the reconstruction quality. Better reconstruction, *i.e.*, higher similarity to original images, is indicated by lower MSE as well as higher PSNR and SSIM values. In addition, we compute the *attack success rate* (ASR) as fraction of the successfully reconstructed images of the
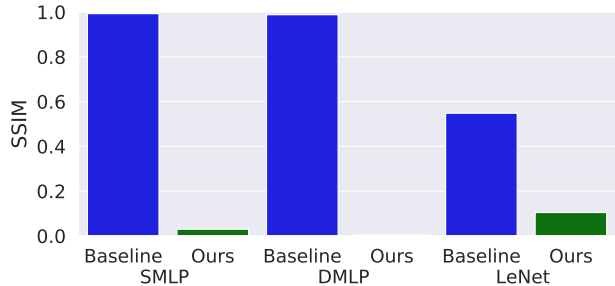
---

[2]https://github.com/JonasGeiping/invertinggradients

Figure 4: SSIM of reconstructed images by IGA on the SMLP, DMLP, and LeNet models for the CIFAR-10 dataset. "Baseline" denotes original models whereas "Ours" denotes extension with PRECODE.

Table 1: Model performance and attack metrics of SMLP, DMLP, and LeNet models trained on CIFAR-10 without and with PRECODE ("Ours"). Arrows indicate direction of improvement. Bold formatting highlights best results.

| Model | | SMLP | | DMLP | | LeNet | |
Defense		Baseline	Ours	Baseline	Ours	Baseline	Ours
Acc_{train} [%]	↑	**95.91**	92.41	**98.51**	98.22	**75.01**	72.01
Acc_{test} [%]	↑	**53.07**	51.75	**54.92**	54.16	**61.19**	58.99
MSE	↑	0.0	**3.95**	0.0	**4.62**	0.49	**2.17**
PSNR	↓	44.13	**6.18**	44.26	**5.51**	15.72	**8.84**
SSIM	↓	0.99	**0.03**	0.99	**0.01**	0.55	**0.1**
ASR [%]	↓	100.0	**0.0**	100.0	**0.0**	42.45	**0.0**

victim dataset [28]. We consider images as being successfully reconstructed if SSIM ≥ 0.6. Using three different seeds, we repeat all inversion attacks on constant sets of 128 victim images per dataset and report the average MSE, PSNR, SSIM, and ASR.

5. Results

5.1. Network Complexity

At first, we investigate the impact of increasing network complexity on the privacy enhancing properties of PRECODE when attacking with IGA. Fig. 4 displays the reconstruction quality as measured by SSIM when training the SMLP, DMLP, and LetNet architectures on CIFAR-10 with and without our proposed extension. Whereas IGA on the baseline MLPs allowed for almost identical reconstructions with SSIM values close to 1, the reconstructions showed negligible similarity after extension with PRECODE. Also the results using LeNet with PRECODE display a massive reduction in SSIM, although SSIM of the baseline models was already reduced compared to the MLPs.

Table 1 summarizes the results obtained on CIFAR-10. Note that the reported accuracies might not be optimal due to potentially suboptimal hyperparameter choices. As can be seen, PRECODE had small impact on the accuracy, *i.e.*, train and test accuracy slightly decreased. Similar to the
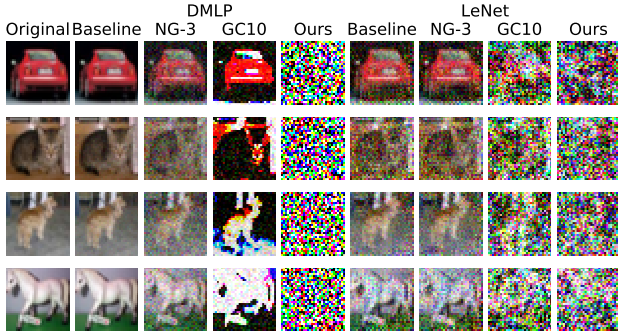
Figure 5: Exemplary reconstruction results on the CIFAR-10 dataset for the baseline DMLP and LeNet models and different defense mechanisms. More reconstruction examples can be found in the *supplementary material*.

Figure 6: Train accuracy on the CIFAR-10 dataset. Line colors define the baseline model and defense mechanisms.

findings in [8], we observe no impact of the network depth on the reconstruction results.

5.2. Defense Mechanisms

Next, we compare PRECODE to other state of the art defense mechanisms, *i.e.*, *noisy gradients* and *gradient compression* by pruning. For noisy gradients, we add Gaussian noise with zero mean and standard deviation σ. We consider two noise levels: $\sigma = 10^{-2}$ (NG-2) and $\sigma = 10^{-3}$ (NG-3). Wei *et al.* report Laplace noise to result in less privacy protection compared to Gaussian noise while having more negative impact on model accuracy [31]. For this reason, we did not consider Laplace noise in our experiments. For gradient compression, referred to as GC in figures and tables, we prune 10% of the gradient values with smallest magnitudes to zero as described in [34].

Fig. 5 displays exemplary reconstruction results on the CIFAR-10 dataset. As can be seen for both baseline models, the training data can be recognizably reconstructed, whereas extension with PRECODE prevents successful reconstruction. Also reconstructions for models using NG-3 and GC show notably reduced similarity to the original images. Yet, for most reconstructed images basic patterns and objects are still recognizable.

The metrics summarized in Table 2 also reflect this behaviour: both NG-3 and GC show increasing MSE and decreasing PSNR, SSIM, and ASR. However, PRECODE clearly outperforms both noisy gradients and gradient compression regarding privacy leakage prevention. In addition, optimization using GC suffered from severe training instabilities that caused unacceptable performance drop. Even with reduced compression rates, we were not able to train models with sufficient accuracy and therefore decided to exclude GC from further experiments. Almost the same observation has been made using noisy gradients with increased noise levels, *i.e.*, $\sigma = 10^{-2}$ (NG-2). NG-2 successfully
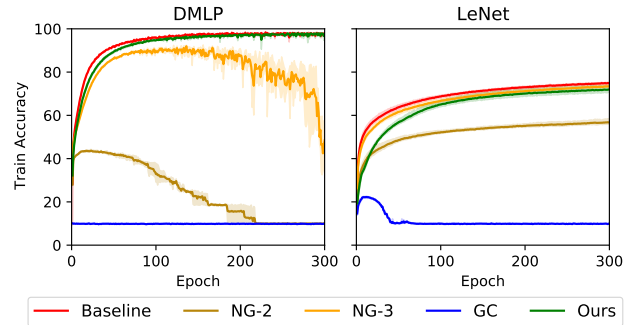
prevents privacy leakage for both architectures but causes a notable drop in model accuracy. Moderate noise levels, *i.e.*, $\sigma = 10^{-3}$ (NG-3), cause less drop in model performance but allowed for larger SSIM. For privacy preservation that is on par with PRECODE, noise levels greater than 10^{-2} would be required. However, this renders models practically irrelevant due to overall bad performance. PRECODE on the other hand barely impacts model performance.

Fig. 6 displays the learning curves in terms of training accuracy for the models trained on CIFAR-10 with and without defense mechanisms. Whereas PRECODE shows no impact on the training process, severe training instabilities and reduced model performance can be observed for the DMLP with NG and GC. NG-3 is the only approach that does not increase convergence time and decrease accuracy for LeNet. However, PRECODE generally provides better protection against IGA, and achieves higher accuracy and faster convergence compared to NG-2 and GC.

5.3. Different Datasets

To extend the empirical evaluation to further datasets, we also consider the CIFAR-100 and Medical MNIST dataset. Results on CIFAR-100 are summarized in Table 3(a). Similar to CIFAR-10, both PRECODE and NG-3 had marginal impact on the test accuracy, whereas increased noise levels of NG-2 resulted in a drop of 10%. Comparing privacy leakage metrics, extending the models with PRECODE resulted in largest MSE and smallest PSNR, SSIM, and ASR.

The results obtained on Medical MNIST, summarized in Table 3(b), also confirm these observations. Compared to CIFAR-10 and CIFAR-100, classification on Medical MNIST generally appears a simpler task as all models achieved 100% accuracy within the first few epochs.

Regarding model training, PRECODE showed the most stable optimization process and fastest model convergence across the privacy-preserving models. The learning curves for both datasets can be found in the *supplementary material*. For some runs, NG-3 showed faster convergence

Table 2: Performance and attack metrics of the DMLP and LeNet models trained on CIFAR-10 with and without the different defense mechanisms. Arrows indicate direction of improvement. Bold formatting highlights best results.

Model Defense		DMLP Baseline	NG-2	NG-3	GC	Ours	LeNet Baseline	NG-2	NG-3	GC	Ours
Acc_{train} [%]	↑	**98.51**	43.83	92.36	10.15	98.22	**75.01**	56.88	73.55	22.29	72.01
Acc_{test} [%]	↑	**54.92**	42.55	51.36	10.04	54.16	61.19	54.36	**62.13**	23.40	58.99
MSE	↑	0.0	2.5	0.17	1.13	**4.62**	0.49	1.59	0.61	1.9	**2.17**
PSNR	↓	44.26	8.11	18.64	12.09	**5.51**	15.72	10.2	14.73	9.36	**8.84**
SSIM	↓	0.99	0.03	0.62	0.42	**0.01**	0.55	0.22	0.49	0.18	**0.1**
ASR [%]	↓	100.0	**0.0**	63.80	3.12	**0.0**	42.45	**0.0**	30.47	**0.0**	**0.0**

Table 3: Model performance and attack metrics of the DMLP and LeNet models trained on (a) CIFAR-100, (b) Medical MNIST, with and without the different defense mechanisms. Arrows indicate direction of improvement. Bold formatting highlights best results.

Model Defense		DMLP Baseline	NG-2	NG-3	Ours	LeNet Baseline	NG-2	NG-3	Ours
Acc_{train} [%]	↑	61.85	14.55	64.78	**80.87**	**63.73**	21.45	48.23	54.42
Acc_{test} [%]	↑	**22.66**	13.44	21.56	21.87	28.92	19.85	**30.22**	27.37
MSE	↑	0.00	2.32	0.15	**4.14**	0.81	1.35	0.76	**1.94**
PSNR	↓	40.23	7.99	18.63	**5.55**	12.69	10.39	13.17	**8.78**
SSIM	↓	0.98	0.03	0.58	**0.01**	0.33	0.20	0.36	**0.10**
ASR [%]	↓	100.00	**0.00**	45.05	**0.00**	2.60	**0.00**	8.33	**0.00**

(a) Result on CIFAR-100.

Model Defense		DMLP Baseline	NG-2	NG-3	Ours	LeNet Baseline	NG-2	NG-3	Ours
Acc_{train} [%]	↑	99.98	**100.00**	99.96	99.89	100.00	99.88	**100.00**	**100.00**
Acc_{test} [%]	↑	99.85	**99.88**	99.82	99.80	**99.96**	99.81	99.92	99.95
MSE	↑	0.00	2.86	0.17	**5.29**	0.81	1.69	1.27	**2.35**
PSNR	↓	38.93	6.57	16.15	**4.30**	12.01	9.07	11.50	**7.55**
SSIM	↓	0.94	0.03	0.43	**0.00**	0.30	0.15	0.26	**0.09**
ASR [%]	↓	99.22	**0.00**	10.94	**0.00**	12.24	**0.00**	12.24	**0.00**

(b) Results on Medical MNIST.

and increased model performance but also increased privacy leakage compared to NG-2 and PRECODE.

5.4. Different Attacks

To show the effectiveness of PRECODE against other attacks we conduct a series of experiments using the CPL attack [31]. Therefore, we adjusted the reconstruction loss function by replacing cosine distance with Euclidean distance as described in section 2.1. When using the originally reported L-BFGS optimizer with learning rate 1 [31], we observed notably increased optimization time and comparably bad reconstruction results even for the baseline models. For this reason, we exchanged L-BFGS with Adam optimizer using parameters as described in section 4.

Table 4 shows the results obtained on CIFAR-10. The overall behavior is similar to the results obtained using IGA. However, compared to IGA (cf. Table 2), we found CPL to generally achieve slightly worse reconstructions.

Table 4: Attack metrics of the DMLP and LeNet models for the CPL attack on CIFAR-10 with and without the different defense mechanisms. Arrows indicate direction of improvement. Bold formatting highlights best results.

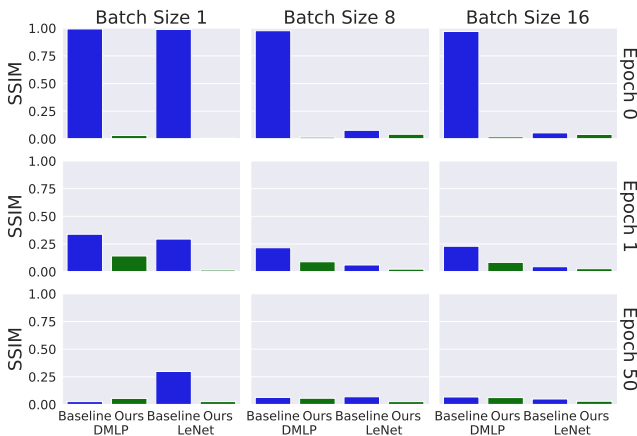| Model Defense		DMLP Baseline	NG-2	NG-3	Ours	LeNet Baseline	NG-2	NG-3	Ours
MSE	↑	0.0	3.01	0.17	**4.07**	0.67	1.52	0.72	**2.51**
PSNR	↓	40.53	7.31	18.57	**6.05**	14.53	10.33	14.15	**8.12**
SSIM	↓	0.99	0.03	0.62	**0.01**	0.47	0.22	0.45	**0.06**
ASR [%]	↓	100.0	**0.0**	64.58	**0.0**	29.17	**0.0**	27.86	**0.0**

Figure 7: SSIM of reconstructed images by IGA on the DMLP, and LeNet models with and without PRECODE for varying victim batch sizes and model training states.

5.5. Batchsizes and Training Progress

Last, we investigate privacy leakage prevention by PRE-CODE over different model training states and batch sizes. All models are attacked after random initialization at the beginning of the training. In addition, we attack the models after 1 and 50 epochs of training. We specifically choose these checkpoints to investigate privacy leakage after each model was trained exactly once on each sample of the training set, as well as when the models are close to convergence (cf. Fig. 6). Regarding batch sizes, we consider a batch size of

1 as baseline, and further investigate models attacked with batches of size 8 and 16. All attacks are performed using the same *victim dataset* as described in section 4. Due to the long computational time needed to execute all reconstruction attacks with these batch sizes, we only consider the baseline models and their extension with PRECODE.

The evaluated reconstruction results are displayed in Fig. 7 in terms of SSIM. During training of the DMLP the victim dataset can be almost entirely reconstructed with SSIM close to 1 independent of the batch size. For LeNet however, increasing batch size caused decreasing SSIM. It can also be observed that SSIM decreases with progressing training state. This can be attributed to lower gradient magnitudes induced by reduced loss as model predictions become more accurate.

Extending the models architectures with PRECODE almost entirely results in decreased SSIM compared to the baseline models. With progressing training using PRECODE, a slight increase in SSIM can be observed. However, all SSIM values remain below 0.15, indicating that no recognizable visual structures can be reconstructed.

6. Discussion

6.1. Key Takeaways

Confirming findings of earlier studies [8, 31, 32, 34], all considered models are prone to privacy leakage by their gradients. Especially MLPs, *i.e.*, models solely constituted of fully connected layers, allow near-identical reconstructions of original training data. To enhance privacy, gradients can be perturbed by noise addition or compression. However, in order to successfully defend against gradient inversion attacks, levels of gradient perturbation that severely impact the optimization process and model performance are required. We conclude that there is an inherent trade-off between model performance and privacy if gradient perturbation methods are used.

Our proposed model extension PRECODE avoids such trade-off. Comparative evaluation using different baseline models and datasets revealed remarkable performance in preventing privacy leakage by IGA and CPL attacks, confirming PRECODE to meet the **privacy enhancement** requirement. By design, PRECODE is a **generic extension** applicable to arbitrary model architectures by **direct integration** into the original model and joint optimization. Yet, PRECODE is a trainable module, *i.e.*, the computational costs increase as the number of trainable model parameters are increased. Realized by fully connected layers, the amount of parameters added by PRECODE grows linearly with the size of the latent features to be stochastically encoded as well as the size of the variational bottleneck. Compared to, *e.g.*, cryptographic methods, this renders negligible increase in computational costs. More importantly,

PRECODE showed no negative effect on the convergence time. Contrary to gradient perturbation at privacy enhancing levels, PRECODE entirely remains within the **model performance** requirement.

6.2. Limitations

Our study is limited as only optimization based gradient inversion attacks are considered. There is no guarantee that other, *e.g.*, analytical attacks [22, 33], are prevented with same efficiency. However, such attacks would need to find sophisticated ways to account for the stochastic sampling, analogous to the reparameterization trick used for backpropagation during variational modeling. In addition, we closely followed related work and investigated only deep learning tasks showcasing very successful attacks on comparably small models, *i.e.*, image classification using MLPs and LeNet. Whereas our proposed extension should also be applicable to other domains such as natural language processing, we currently provide no empirical proof. Regarding model complexity, Geiping *et al.* [8] showed that sufficient parameter tuning and restart attempts even allow to reconstruct images from large batches and more complex models such as residual neural networks [9]. Each attack optimization step then takes orders of magnitudes more time, which prevented systematic evaluation within this study.

7. Conclusion

We introduced PRECODE – PRivacy EnhanCing mODulEs – to defend deep learning models against privacy leakage by gradients exchanged during collaborative learning. Inspired by variational modeling, we realized PRECODE as variational bottleneck prior to the output layer of original models. We systematically evaluated our extension using fully connected and convolutional architectures on three image classification datasets and two state of the art gradient inversion attacks. To prevent privacy leakage, commonly used gradient perturbation methods require perturbation levels that cause unacceptable drop in model performance. Compared to these methods, PRECODE showed consistently better prevention of privacy leakage, while having almost no impact on model performance. We conclude that our proposed extension represents a promising alternative to commonly used defense mechanisms. In future work we plan to evaluate other model architectures, datasets, and tasks, for deeper investigation of the generic applicability of our approach.

8. Acknowledgments

References

[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] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016.

[3] Yoshinori Aono, Takuya Hayashi, Lihua Wang, Shiho Moriai, et al. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 13(5):1333–1345, 2017.

[4] A. P. Lozano (apolanco3225). Medical mnist classification. https://github.com/apolanco3225/Medical-MNIST-Classification, 2017.

[5] 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.

[6] Moming Duan, Duo Liu, Xianzhang Chen, Renping Liu, Yujuan Tan, and Liang Liang. Self-balancing federated learning with global imbalanced data in mobile systems. *IEEE Transactions on Parallel and Distributed Systems*, 32(1):59–71, 2020.

[7] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[8] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients - how easy is it to break privacy in federated learning? In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 16937–16947. Curran Associates, Inc., 2020.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

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

[11] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *In International Conference on Learning Representations (ICLR)*, May 2015.

[13] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[14] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[15] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[16] Jin-woo Lee, Jaehoon Oh, Sungsu Lim, Se-Young Yun, and Jae-Gil Lee. Tornadoaggregate: Accurate and scalable federated learning via the ring-based architecture. *arXiv preprint arXiv:2012.03214*, 2020.

[17] Wenqi Li, Fausto Milletarì, Daguang Xu, Nicola Rieke, Jonny Hancox, Wentao Zhu, Maximilian Baust, Yan Cheng, Sébastien Ourselin, M Jorge Cardoso, et al. Privacy-preserving federated brain tumour segmentation. In *International Workshop on Machine Learning in Medical Imaging*, pages 133–141. Springer, 2019.

[18] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887*, 2017.

[19] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

[20] 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.

[21] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018.

[22] Xudong Pan, Mi Zhang, Yifan Yan, Jiaming Zhu, and Min Yang. Theory-oriented deep leakage from gradients via linear equation solver. *arXiv preprint arXiv:2010.13356*, 2020.

[23] Abhijit Guha Roy, Shayan Siddiqui, Sebastian Pölsterl, Nassir Navab, and Christian Wachinger. Braintorrent: A peer-to-peer environment for decentralized federated learning. *arXiv preprint arXiv:1905.06731*, 2019.

[24] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[25] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems*, 31(9):3400–3413, 2019.

[26] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321, 2015.

[27] Giuseppe Tagliavini, Stefan Mach, Davide Rossi, Andrea Marongiu, and Luca Benini. A transprecision floating-point platform for ultra-low power computing. In *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1051–1056. IEEE, 2018.

[28] Isabel Wagner and David Eckhoff. Technical privacy metrics: a systematic survey. *ACM Computing Surveys (CSUR)*, 51(3):1–38, 2018.

[29] Yijue Wang, Jieren Deng, Dan Guo, Chenghong Wang, Xi-anrui Meng, Hang Liu, Caiwen Ding, and Sanguthevar Ra-jasekaran. Sapag: A self-adaptive privacy attack from gradi-ents. *arXiv preprint arXiv:2009.06228*, 2020.

[30] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Si-moncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[31] Wenqi Wei, Ling Liu, Margaret Loper, Ka-Ho Chow, Mehmet Emre Gursoy, Stacey Truex, and Yanzhao Wu. A framework for evaluating gradient leakage attacks in feder-ated learning. *arXiv preprint arXiv:2004.10397*, 2020.

[32] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*, 2020.

[33] Junyi Zhu and Matthew Blaschko. R-gap: Recursive gradient attack on privacy. *Proceedings ICLR 2021*, 2021.

[34] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, pages 14774–14784, 2019.