See through Gradients: Image Batch Recovery via GradInversion

Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M. Alvarez, Jan Kautz, Pavlo Molchanov

NVIDIA

{dannyy, amallya, avahdat, josea, jkautz, pmolchanov}@nvidia.com

Abstract

Training deep neural networks requires gradient estimation from data batches to update parameters. Gradients per parameter are averaged over a set of data and this has been presumed to be safe for privacy-preserving training in joint, collaborative, and federated learning applications. Prior work only showed the possibility of recovering input data given gradients under very restrictive conditions – a single input point, or a network with no non-linearities, or a small 32×32 px input batch. Therefore, averaging gradients over larger batches was thought to be safe. In this work, we introduce GradInversion, using which input images from a larger batch ($8 - 48$ images) can also be recovered for large networks such as ResNets (50 layers), on complex datasets such as ImageNet (1000 classes, 224×224 px). We formulate an optimization task that converts random noise into natural images, matching gradients while regularizing image fidelity. We also propose an algorithm for target class label recovery given gradients. We further propose a group consistency regularization framework, where multiple agents starting from different random seeds work together to find an enhanced reconstruction of the original data batch. We show that gradients encode a surprisingly large amount of information, such that all the individual images can be recovered with high fidelity via GradInversion, even for complex datasets, deep networks, and large batch sizes.

1. Introduction

Sharing weight updates or gradients during training is the central idea behind collaborative, distributed, and federated learning of deep networks [1, 22, 24, 25, 28]. In the basic setting of federated stochastic gradient descent, each device learns on local data, and shares gradients to update a global model. Alleviating the need to transmit training data offers several key advantages. This keeps user data private, allaying concerns related to user privacy, security, and other proprietary concerns. Further, this eliminates the need to store, transfer, and manage possibly large datasets. With this framework, one can train a model on medical data without access to any individual's data [3, 32], or perception model
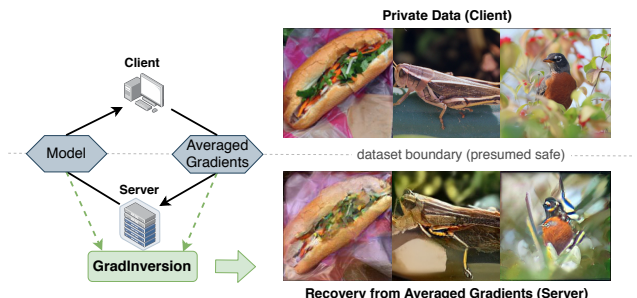
(a) Inverting averaged gradients to recover original image batches
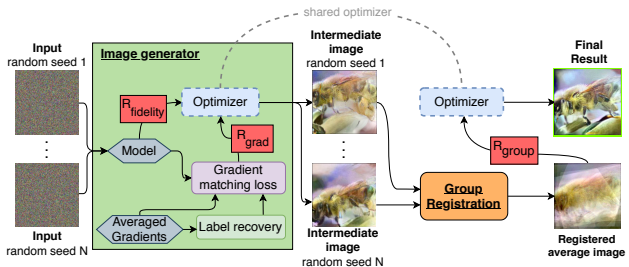
(b) Overview of our proposed GradInversion method

Figure 1: We propose (a) GradInversion to recover hidden training image batches with high fidelity via inverting *averaged* gradients. GradInversion formulates (b) an optimization process that transforms noise to input images (Sec. 3.1). It starts with label restoration from the gradient of the fully connected layer (Sec. 3.2), then optimizes inputs to match target gradients under fidelity regularization (Sec. 3.3) and registration-based group consistency regularization (Sec. 3.4) to improve reconstruction quality. This enables recovery of 224×224 pixel ImageNet samples from ResNet-50 batch gradients, which was previously impossible (please zoom into examples above. More in Sec. 4).

for autonomous driving without invasive data collection [41].

While this setting might seem safe at first glance, a few recent works have begun to question the central premise of federated learning - is it possible for gradients to leak private information of the training data? Effectively serving as a "proxy" of the training data, the link between gradients to the data in fact offers potential for retrieving information: from revealing the positional distribution of original data [33, 44], to even enabling pixel-level detailed image reconstruction from gradients [13, 53, 55]. Despite remarkable progress, in-

verting an original image through gradient matching remains a very challenging task – successful reconstruction of images of high resolution for complex datasets such as ImageNet [9] has remained elusive for batch sizes larger than one.

Emerging research on network inversion techniques offers insights into this task. Network inversion enables noise-to-image conversion via back-propagating gradients on appropriate loss functions to the learnable inputs. Initial solutions were limited to shallow networks and low-resolution synthesis [11, 39], or creating an artistic effect [37]. However, the field has rapidly evolved, enabling high-fidelity, high-resolution image synthesis on ImageNet from commonly trained classifiers, making downstream tasks data-free for pruning, quantization, continual learning, knowledge transfer, *etc*. [5, 17, 42, 48]. Among these, DeepInversion [48] yields state-of-the-art results on image synthesis for ImageNet. It enables the synthesis of realistic data from a vanilla pretrained ResNet-50 [19] classifier by regularizing feature distributions through batch normalization (BN) priors.

Building upon DeepInversion [48], we delve into the problem of batch recovery via gradient inversion. We formulate the task as the optimization of the input data such that the gradients on that data match the ones provided by the client, while ensuring realism of the input data. However, since the gradient is also a function of the ground-truth label, one of the main challenges is to identify the ground-truth label for each data point in the batch. To tackle this, we propose a one-shot batch label restoration algorithm that uses gradients from the last fully connected layer.

Our goal is to recover the exact images that the client possesses. By starting from noisy inputs generated by different random seeds, multiple optimization processes are likely to converge to different minimas. Due to the inherently spatially-invariant nature of convolutional neural networks (CNNs), these resulting images share spatial information but differ in the exact location and arrangement. To allow for improved convergence towards the ground truth images, we compute a registered mean image from all candidates and introduce a group consistency regularization term on every optimization process to reduce deviation. We find that the proposed approach and group consistency regularization provide superior better image recovery compared to prior optimization approaches [13, 55].

Our non-learning based image recovery method recovers more specific details of the hidden input data when compared to the state-of-the-art generative adversarial networks (GAN), such as BigGAN [4]. More importantly, we demonstrate that a full recovery of individual images of 224×224 px resolution with high fidelity and visual details, by inverting gradients of the batch, is now made feasible even up to batch size of 48 images.

Our main contributions are as follows:

- We introduce GradInversion to recover hidden original images from random noise via optimization given batch-averaged gradients.
- We propose a label restoration method to recover ground truth labels using final fully connected layer gradients.
- We introduce a group consistency regularization term, based on multi-seed optimization and image registration, to improve reconstruction quality.
- We demonstrate that a full recovery of detailed individual images from batch-averaged gradients is now feasible for deep networks such as ResNet-50.
- We introduce a new *Image Identifiability Precision* metric to measure the ease of inversion over varying batch sizes, and identify samples vulnerable to inversion.

2. Related Work

Image synthesis. GANs [16, 23, 36, 38, 50] have delivered state-of-the-art results for generative image modeling, *e.g.*, BigGAN-deep on ImageNet [4]. Training a GAN's generator, however, requires access to original data. Multiple works have also looked into training GANs given only a pretrained model [6, 34], but result in images that lack details or perceptual similarities to original data.

Prior work in security studies image synthesis from a pretrained single network. The *model inversion* attack by Fredrikson *et al*. [11] optimizes inputs to obtain class images using gradients from the target model. Follow-up works [20, 45, 47] scale to new threat scenarios, but remain limited to shallow networks. The *Secret Revealer* [52] exploits priors from auxiliary datasets and trains GANs to guide inversion, scales the attack to modern architectures, but on the datasets with less diverse samples, *e.g.*, MNIST and face recognition.

Though originally aiming at understanding network properties, visualization techniques offer another viable option to generate images from networks. Mahendran *et al*. [31] explore inversion, activation maximization, and caricaturization to synthesize "natural pre-images" from a trained network [30, 31]. Nguyen *et al*. use global generative priors to help invert trained networks [39] for images, followed by Plug & Play [38] that boosts up image diversity and quality via latent priors. These methods still rely on auxiliary dataset information, feature embedding, or altered training.

Recent efforts focus on image generation from a pretrained network without any auxiliary information. Deep-Dream [37] by Mordvintsev *et al*. hints on "dreaming" new visual features onto images leveraging gradients on inputs, extendable towards noise-to-image conversion. Saturkar *et al*. [42] extended the approach to more realistic images. The more recent extensions [5, 48] significantly improved state-of-the-art performance on image synthesis from off-the-shelf classifiers, without auxiliary information nor additional training but relying on BN statistics.

Gradient-based inversion. There have been early attempts

to invert gradients in pursuit of proxy information of the original data, *e.g.*, the existence of certain training samples [33, 44] or sample properties [21, 44] of the dataset. These methods primarily focus on very shallow networks.

A more challenging task aims at reconstructing the exact images from gradients. The early attempt by Phong *et al.* [26] brought theoretical insights on this task by showing provable reconstruction feasibility on single neuron or single layer networks. Wang *et al.* [46] empirically inverted out single image representations from gradients of a 4-layer network. Along the same line, Zhu *et al.* [55] pushed gradient inversion towards deeper architectures by jointly optimizing for "pseudo" labels and inputs to match target gradients. The method leads to accurate reconstruction up to pixel-level, while remains limited to continuous models (*e.g.*, ones with sigmoid instead of ReLU) without any strides, and scales up to low-resolution CIFAR datasets. Zhao *et al.* [53] extend the approach with a label restoration step, hence improving speed of single image reconstruction. The very recent work by Geiping *et al.* [13] for the first time pushed the boundary towards ImageNet-level gradient inversion - it reconstructs single images from gradients. Despite remarkable progress, the field struggles on ImageNet for any batch size larger than one, when gradients get averaged.

3. GradInversion

In this section, we explain GradInversion in detail. We first frame the problem of input reconstruction from gradients as an optimization process. Then, we explain our batch label restoration method, followed by the auxiliary losses used to ensure realism and group consistency regularization.

3.1. Objective Function

Given a network with weights $\mathbf{W}$ and a batch-averaged gradient $\Delta\mathbf{W}$ calculated from a ground truth batch with images $\mathbf{x}^*$ and labels $\mathbf{y}^*$, our optimization solves for

$$\hat{\mathbf{x}}^* = \underset{\hat{\mathbf{x}}}{\arg\min}\ \mathcal{L}_{\text{grad}}(\hat{\mathbf{x}}; \mathbf{W}, \Delta\mathbf{W}) + \mathcal{R}_{\text{aux}}(\hat{\mathbf{x}}), \qquad (1)$$

where $\hat{\mathbf{x}} \in \mathbb{R}^{K \times C \times H \times W}$ (K, C, H, W being the batch size, number of color channels, height, width) is a "synthetic" input batch, initialized as random noise and optimized towards the ground truth $\mathbf{x}^*$. $\mathcal{L}_{\text{grad}}(\cdot)$ enforces matching of the gradients of this synthetic data (on the original loss for a network with weights $\mathbf{W}$) with the provided gradients $\Delta\mathbf{W}$. This is augmented by auxiliary regularization $\mathcal{R}_{\text{aux}}(\cdot)$ based on image fidelity and group consistency regularization,

$$\mathcal{R}_{\text{aux}}(\hat{\mathbf{x}}) = \mathcal{R}_{\text{fidelity}}(\hat{\mathbf{x}}) + \mathcal{R}_{\text{group}}(\hat{\mathbf{x}}). \qquad (2)$$

We next elaborate on each term individually. For gradient matching, we minimize the ℓ_2 distances between gradients on the synthesized images $\hat{\mathbf{x}}$ and the ground truth gradient:

$$\mathcal{L}_{\text{grad}}(\hat{\mathbf{x}}; \mathbf{W}, \Delta\mathbf{W}) = \alpha_{\text{G}} \sum_l ||\nabla_{\mathbf{W}^{(l)}} \mathcal{L}(\hat{\mathbf{x}}, \hat{\mathbf{y}}) - \Delta\mathbf{W}^{(l)}||_2, \quad (3)$$

where $\Delta\mathbf{W}^{(l)} = \nabla_{\mathbf{W}^{(l)}} \mathcal{L}(\mathbf{x}^*, \mathbf{y}^*)$ refers to ground truth gradient at layer l, and the summation, scaled by α_{G}, runs over all layers. One key yet missing component here is the $\hat{\mathbf{y}}$ that initiates the backpropagation. We next explain an effective algorithm for restoring batch-wise label from the gradients of the fully connected classification layer.

3.2. Batch Label Restoration

Considering the cross-entropy loss for the classification task, the ground truth gradient of $\mathbf{x}^* = [x_1, x_2, ..., x_K]$ of batch size K can be decomposed into:

$$\nabla_{\mathbf{W}} \mathcal{L}(\mathbf{x}^*, \mathbf{y}^*) = \frac{1}{K} \sum_k \nabla_{\mathbf{W}} \mathcal{L}(x_k, y_k), \qquad (4)$$

where x_k, y_k denotes an original image/label pair. For each image x_k, the gradient w.r.t. the network final logits z at index n is $\nabla_{z_{n,k}} \mathcal{L}(x_k, y_k) = p_{k,n} - y_{k,n}$, where $p_{k,n}$ is the post-softmax probability in range (0, 1), and $y_{k,n}$ is the binary presentation of y_k at index n among N total classes. Consequentially, this leaves $\text{sign}\big(\nabla_{z_{n,k}} \mathcal{L}(x_k, y_k)\big)$ negative *iff* $n = n_k^*$ at the ground truth index, and positive otherwise. However, we do not have access to $\nabla_{z_{n,k}} \mathcal{L}(x_k, y_k)$ as gradients are only given w.r.t. the model parameters.

Denote the parameters of the final fully connected classification layer by $\mathbf{W}^{(\text{FC})} \in \mathbb{R}^{M \times N}$ with M being the number of embedded features, and N being the number of target classes. Define $\Delta\mathbf{W}^{(\text{FC})}_{m,n,k} := \nabla_{w_{m,n}} \mathcal{L}(x_k, y_k)$ as the gradient of the training loss for image x_k w.r.t. $\mathbf{W}^{(\text{FC})}_{m,n}$, connecting feature m to logits n. We are only given the average of the tensor $\Delta\mathbf{W}^{(\text{FC})}$ along the batch dimension k. Using the chain rule we have:

$$\Delta\mathbf{W}^{(\text{FC})}_{m,n,k} = \nabla_{z_{n,k}} \mathcal{L}(x_k, y_k) \times \frac{\partial z_{n,k}}{\partial w_{m,n}}. \qquad (5)$$

Note that $\frac{\partial z_{n,k}}{\partial w_{m,n}} = o_{m,k}$ where $o_{m,k}$ is the mth input of the fully connected layer, and is also the mth output of previous layer. If the previous layer has commonly used activation functions such as ReLU or sigmoid, $o_{m,k}$ is always non-negative. This hints on target label existence via signs of a new informative indicator:

$$\mathbf{S}_{n,k} := \sum_m \Delta\mathbf{W}^{(\text{FC})}_{m,n,k} = \sum_m \underbrace{\nabla_{z_{n,k}} \mathcal{L}(x_k, y_k)}_{\text{neg. \textit{iff} } n = n_k^*} \times \underbrace{o_{m,k}}_{\text{non-neg.}}, \qquad (6)$$

where $\mathbf{S} = \{\mathbf{S}_{n,k}\}$ is an $N \times K$-matrix, constructed by summing the tensor $\Delta\mathbf{W}^{(\text{FC})}$ along the feature dimension m. Interestingly, $\mathbf{S}$ contains negative values for the ground truth label of each instance. Thus, the kth column of $\mathbf{S}$ can be used to restore the ground truth label for the kth image by simply identifying the index of the negative entry. Zhao *et al.* [53] explored this rule for single image label restoration. However, we do not have access to $\mathbf{S}$ in our

multi-sample batch setup as the given gradients are averaged over all images.

Motivated by this, we define the N-dimensional batch-level vector $\mathbf{s} = \{\mathbf{s}_n\}$ by averaging $\mathbf{S}$ along its columns:

$$\mathbf{s}_n := \frac{1}{K}\sum_k \mathbf{S}_{n,k} = \sum_m \underbrace{\left(\frac{1}{K}\sum_k \Delta\mathbf{W}^{(\text{FC})}_{m,n,k}\right)}_{\text{given in } \Delta\mathbf{W}^{(\text{FC})}}. \qquad (7)$$

The appealing property of $\mathbf{s}$ is that it can be computed easily from the given gradient for the fully connected layer by summing it along the feature dimension m as shown on the right hand side of Eqn. 7.

As noted above, each column in $\mathbf{S}$ is a vector, containing a single negative peak at the label index and positive otherwise. Since the vector $\mathbf{s}$ is a linear super-position of $\mathbf{S}$'s columns, from all individual images x_k's in the batch, this information can be lost during summation. However, we empirically observe that the encoded positions often possess larger magnitudes $|\mathbf{S}_{n_k^*,k}| \gg |\mathbf{S}_{n \neq n_k^*,k}|$. This leaves a negative sign mostly intact when the summation brings in positive values from other images.

To further enable a more robust propagation of negative signs, we utilize column-wise minimum values, instead of summation along the feature dimension for the $\mathbf{s}$ calculation: to have a sum along the feature dimension be negative, at least one of its positions has to be negative, but not vice versa. This further boosts up the label restoration accuracy, especially when the batch size is large. Thus, we formulate the final label restoration algorithm for batch size K as:

$$\hat{\mathbf{y}} = \arg \text{sort}\left(\min_m \nabla_{\mathbf{W}^{(\text{FC})}_{m,n}} \mathcal{L}(\mathbf{x}^*, \mathbf{y}^*)\right)[: K], \qquad (8)$$

with m corresponding to the feature embedding dimension before the fully connected layer. The resulting $\hat{\mathbf{y}}$ supports Eqn. 3 in subsequent $\hat{\mathbf{x}}$ optimization in pursuit for $\mathbf{x}^*$. One limitation of the proposed method is that it assumes non-repeating labels in the batch, which generally holds for a randomly sampled batch of size K that is much smaller than the number of classes at 1000 for ImageNet.

Even with correct $\mathbf{y}^*$, finding the global minima for $\mathcal{L}_{\text{grad}}(\cdot)$ remains challenging. The task is under-constrained, suffers from information loss due to non-linearity and pooling layers, and has only one correct solution [13, 55]. We next introduce $\mathcal{R}_{\text{aux}}(\cdot)$ based on fidelity and group consistency regularization to assist with this optimization.

3.3. Fidelity (Realism) Regularization

We use the strong prior proposed in DeepInversion [48] to guide the optimization towards natural images. Specifically, we add $\mathcal{R}_{\text{fidelity}}(\cdot)$ to the loss function to steer $\hat{\mathbf{x}}$ away from unrealistic images with no discernible visual information:

$$\mathcal{R}_{\text{fidelity}}(\hat{\mathbf{x}}) = \alpha_{\text{tv}}\mathcal{R}_{\text{TV}}(\hat{\mathbf{x}}) + \alpha_{\ell_2}\mathcal{R}_{\ell_2}(\hat{\mathbf{x}}) + \alpha_{\text{BN}}\mathcal{R}_{\text{BN}}(\hat{\mathbf{x}}), \quad (9)$$

| original | results of independent optimization processes |

Figure 2: Reconstruction variation in single-path optimization, focusing on one target from a batch of size 8. Optimizations follow the exact same loss hyperparameters, given only varying random seeds for pixel-wise initialization of $\hat{\mathbf{x}}$.

where $\mathcal{R}_{\text{TV}}$ and $\mathcal{R}_{\ell_2}$ denote standard image priors [30, 37, 40] that penalize the total variance and ℓ_2 norm of $\hat{\mathbf{x}}$, resp., with scaling factors $\alpha_{\text{tv}}, \alpha_{\ell_2}$. The key insight of DeepInversion resides in exploiting a strong prior in BN statistics:

$$\mathcal{R}_{\text{BN}}(\hat{\mathbf{x}}) = \sum_l || \mu_l(\hat{\mathbf{x}}) - \text{BN}_l(\text{mean})||_2 + \\ \sum_l || \sigma_l^2(\hat{\mathbf{x}}) - \text{BN}_l(\text{variance})||_2, \qquad (10)$$

where $\mu_l(\hat{\mathbf{x}})$ and $\sigma_l^2(\hat{\mathbf{x}})$ are the batch-wise mean and variance estimates of feature maps corresponding to the lth convolutional layer. By enforcing valid intermediate distributions at all levels, $\mathcal{R}_{\text{fidelity}}(\cdot)$ yields convergence towards realistic-looking solutions.

3.4. Group Consistency Regularization

An additional challenge of gradient-based inversion lies in the exact localization of the target object, due to translational invariance of CNNs. Unlike an ideal scenario where optimization converges to one ground truth, we observe that when repeating the optimization with different seeds, e.g., as in Fig. 2, each optimization process unveils a local minimum that allocates semantically correct image features at all levels, but differs from others – images shift around the ground truth, focusing on slightly different details. During the forward pass, the existence of pooling layers, strided convolutions, and zero-padding, jointly causes spatial equivariance among the restored images, as also observed by Geiping et al. [13]. A combination of the restored images from varying seeds, however, hints at the potential for a better restoration of the final image closer to the ground truth.

We introduce a group consistency regularization term that exploits multiple seeds simultaneously in a joint optimization manner, as shown in Fig 3. Intuitively, a joint exploration with multiple paths can expand and enlarge the search space during gradient descent. However, we have to regularize them to prevent too much divergence, at least in the final stages, given the search for a single target. We optimize each input using the target Eqn. 1. To facilitate information exchange, we regularize all the inputs simultaneously with a new group consistency regularization term:

$$\mathcal{R}_{\text{group}}(\hat{\mathbf{x}}, \hat{\mathbf{x}}_{g \in G}) = \alpha_{\text{group}}||\hat{\mathbf{x}} - \mathbb{E}(\hat{\mathbf{x}}_{g \in G})||_2, \qquad (11)$$

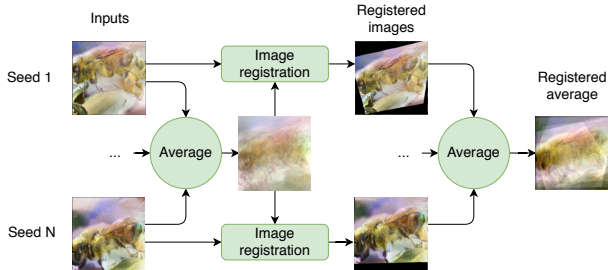that jointly considers all the image candidates across all the

Figure 3: Overview of group consistency regularization.

seeds, and penalizes any candidate $\hat{\mathbf{x}}_g$ once it deviates away from the "consensus" image $\mathbb{E}(\hat{\mathbf{x}}_{g \in G})$ of the group.

One quick and intuitive option for $\mathbb{E}(\hat{\mathbf{x}}_{g \in G})$ is pixel-wise averaging. Though "lazy" as it seems, pixel-wise mean already leads to visual improvements by mixing the information and feedback from all seeds in the group, as we will show later. To further explore the underlying transformations across seeds and create better consensus, we add in image registration to improve $\mathbb{E}(\hat{\mathbf{x}}_{g \in G})$:

$$\mathbb{E}(\hat{\mathbf{x}}_{g \in G}) = \frac{1}{|G|} \sum_g \mathbf{F}_{\hat{\mathbf{x}}_g \to \frac{1}{|G|} \sum_g \hat{\mathbf{x}}_g}(\hat{\mathbf{x}}_g). \quad (12)$$

This leads to our final group consistency regularization shown in Fig. 3. We (i) first compute the pixel-wise mean within the candidate set of size $|G|$ as a coarse registration target, (ii) register each individual image towards the target via $\mathbf{F}(\cdot)$, and (iii) obtain the post-registered mean as the target for regularization. We use RANSAC-flow [43] for $\mathbf{F}(\cdot)$. As we will show later, group consistency regularization enables consistent improvements in recovery across various evaluation metrics, further closing the gap between reconstructed and original batches.

3.5. The Final Update

Using all the above losses, we update the input in an iterative manner. To further encourage exploration and diversity, we add pixel-wise random Gaussian noise in each update, inspired by the Langevin updates in energy-based models [10, 12, 15]. Our final optimization steps are:

$$\begin{aligned}
\Delta_{\hat{\mathbf{x}}^{(t)}} &\leftarrow \nabla_{\hat{\mathbf{x}}}\left(\mathcal{L}_{\mathrm{grad}}(\hat{\mathbf{x}}^{(t-1)}, \nabla \mathbf{W}) + \mathcal{R}_{\mathrm{aux}}(\hat{\mathbf{x}}^{(t-1)})\right) \\
\eta &\leftarrow \mathcal{N}(0, \mathcal{I}) \\
\hat{\mathbf{x}}^{(t)} &\leftarrow \hat{\mathbf{x}}^{(t-1)} + \lambda(t)\Delta_{\hat{\mathbf{x}}^{(t)}} + \lambda(t)\alpha_n \eta
\end{aligned}$$

where $\Delta_{\hat{\mathbf{x}}^{(t)}}$ corresponds to an optimizer update, η denotes randomly sampled noise to encourage exploration, $\lambda(t)$ is the learning rate, and α_n re-scales the finally added noise.

4. Experiments

We evaluate our method for the classification task on the large-scale 1000-class ImageNet ILSVRC 2012 dataset [9] at 224×224 pixels. We first perform a number of ablations to

evaluate the contribution of each component of our method. Then, we show the success of GradInversion and compare with prior art. Finally, we increase the batch size to explore the limits of gradient inversion.

Implementation details. In all cases, image pixels are initialized $i.i.d.$ from Gaussian noise of $\mu = 0$ and $\sigma = 1$. We primarily focus on the ResNet-50 architecture for the classification task, pre-trained with MOCO V2 and fine-tuned only the classification layer, achieving 71.0% top-1 accuracy on ImageNet[1]. We observe that stronger feature extraction leads to better restoration as compared to the default pre-trained PyTorch model, and shallower network architectures (ResNet-18). We use Adam for optimization with a 0.1 learning rate with cosine learning rate decay, and 50 iterations as warm up. We use $\alpha_{\mathrm{tv}} = 1 \cdot 10^{-4}, \alpha_{\ell_2} = 1 \cdot 10^{-6}, \alpha_{\mathrm{BN}} = 0.1, \alpha_{\mathrm{G}} = 0.001, \alpha_{\mathrm{group}} = 0.01, \alpha_n = 0.2$ as loss scaling constants. For feature distribution regularization, we primarily focus on the case when BN statistics of the target batch is jointly provided with the gradients as commonly required in distributed learning for global BN updates [29, 49, 54]. We also analyze regularization towards network BN means and variances - averaged over dataset, they offer proxy for single batch statistics. We synthesize image batches of resolution 224×224 using NVIDIA V100 GPUs and automatic-mixed precision (AMP) [35] acceleration. Optimization of each batch consumes 20K optimization iterations.

Evaluation metrics. We present visual comparisons of images obtained under different settings and evaluate three quantitative metrics for image similarity. To account for pixel-wise mismatch, we compute: (i) the cosine similarity in $\mathrm{FFT}_{\mathrm{2D}}$ frequency response, (ii) post-registration PSNR, and (iii) LPIPS perceptual similarity score [51] between reconstruction and original images.

4.1. Ablation Studies

4.1.1 Label restoration

We first restore labels from the gradients of the fully connected layer. Table 1 summarizes the averaged label restoration accuracy on ImageNet training and validation sets, given 10K randomly drawn samples divided into varying batch sizes. In a zero-shot method, GradInversion restores original labels accurately, improving upon prior art [53].

4.1.2 Batch reconstruction

We next gradually add each proposed loss to the optimization process. Here we focus on a batch of 8 images for algorithm ablations before expanding towards a larger batch size. We summarize results in Table 5 and discuss insights next:

Adding $\mathcal{L}_{grad}$**.** We find ℓ_2 loss outperforms cosine similarity [13] for gradient matching - see Appendix for details.

---

[1]Based on https://github.com/facebookresearch/moco. MOCO V2 (Chen *et al.* [8]) enhances MOCO (He *et al.* [18]) with SimCLR (Chen *et al.* [7]) and reports ImageNet top-1 accuracy at 71.1% [8].

Batch	Label Restoration Accuracy (%)			
	Training Set		Validation Set	
size	[53][†]	ours	[53][†]	ours
1	100.0	100.0	100.0	100.0
8	95.89	99.56	96.08	99.47
32	89.88	99.29	90.32	99.19
64	84.51	98.79	82.27	98.21
96	80.53	97.88	82.13	98.11

Table 1: Average restoration accuracy over 10K random samples of different batch size from the ImageNet training/validation sets without label repeats. [†]: the original method [53] only works for single image - we extend it by adopting its sum rule for Eqn. 7 and then show improvements.

Obj. Function	$\mathcal{L}_{grad}(\hat{x}^*; \nabla W)$	Distance to Original Images		
		FFT$_{2D}$ ↓	PSNR ↑	LPIPS ↓
$\mathcal{N}(0, \mathcal{I})$	8.625	0.706	9.964	1.351
$\mathcal{L}_{grad}$	4.190	0.404	10.753	0.919
+ $\mathcal{R}_{fidelity}$	3.206	0.279	12.058	0.655
+ $\mathcal{R}_{group.lazy}$	2.918	0.233	12.261	0.578
+ $\mathcal{R}_{group.reg}$	**2.685**	**0.175**	**12.929**	**0.484**

x* $\mathcal{L}_{grad}$ $+\mathcal{R}_{fidelity}$ $+\mathcal{R}_{group.lazy}$ $+\mathcal{R}_{group.reg}$
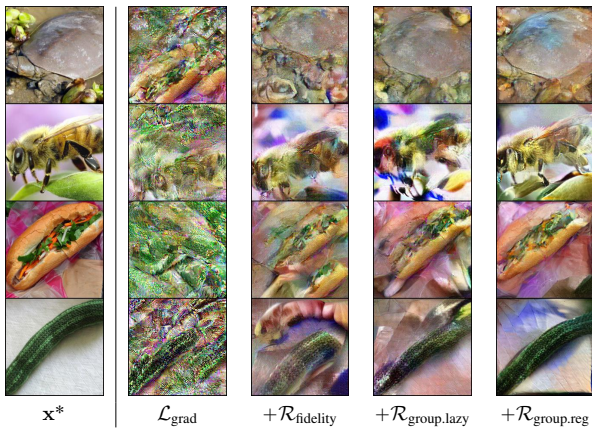
Table 2: Ablation study when each proposed loss to optimization objective function - quantitative (up) and qualitative (bottom) comparison. Original batch contains 8 samples - we show 4 samples visually here amid space limit, see Appendix for entire batch.

Model	Distance to Original Images		
	FFT$_{2D}$ ↓	PSNR ↑	LPIPS ↓
ResNet-50 (MOCO V2)	0.175	12.929	0.484
ResNet-50 (standard)	0.204	11.771	0.584
ResNet-18 (standard)	0.218	10.729	0.693

Table 3: Reconstruction under varying feature extraction strength.

Reconstructed images remain noisy. Partial original features emerge, but leak among images within the batch.

Adding $\mathcal{R}_{fidelity}$. Adding fidelity regularization immediately improves image quality. Conditioned on image prior, gradient inversion starts to allocate visual details towards individual images, enabling both visual and quantitative improvements in Table 5.

Adding $\mathcal{R}_{group}$. Group consistency regularization further improves reconstruction. For this analysis, we use 8 random seeds, each determining a Gaussian initialization of inputs and the associated pixel-wise perturbations. All seeds are

jointly optimized, compatible with standard multi-node training pipeline that supports synchronization only for $\mathbb{E}(\hat{x}_{g \in G})$ computation. For better insights, we next compare the choice of (i) "lazy" pixel-wise mean, and (ii) registration-augmented mean as the regularization target.

a) Lazy regularization. We observe "lazy" pixel-wise mean as regularization target already brings in performance improvements. Though not yet accommodating for inter-seed variation, pixel-wise mean hints on correct "perceived" positions of the target objects. Objects start to emerge at correct positions with improved orientations.

b) Registration enhancement. We then add in registration to exploit consensus among candidates. We start registration after 5K initial optimization iterations to allow for sufficient feature emergence, then iterate every 100 iterations. Ideally, each candidate shall be registered to its original image for the best spatial adjustment. While given no such access, registration to pixel-wise mean turns out to be effective. The final registration-based regularization helps close the remaining gap - it improves all evaluation metrics in Table 5. At this stage, GradInversion accurately allocates detailed original contents to *individual* images, from *averaged* gradients.

Inverting different networks. We observe that gradients from a stronger feature extractor leak more information - see a quick comparison in Table 3. Self-supervised pretraining of ResNet-50 leads to the best image reconstruction, when compared to a standard training recipe of the same ResNet-50 architecture, and a weaker ResNet-18. We continue our analysis with ResNet-50 MOCO V2 to study the limits of batch reconstruction under gradient inversion.

4.2. Comparison with the state-of-the-art

We next compare with prior art on the batch size of 8 images with 224x224px. We summarize both qualitative (Fig. 4) and quantitative results (Table 4). We compare with three viable methods for image synthesis:

(i) Gradient inversion [13, 55]: We first compare with prior model inversion methods for gradient matching: (i) deep gradient leakage method by Zhu *et al.* [55] and (ii) federated gradient inversion by Geiping *et al.* [13]. We first extend both techniques towards ImageNet batch restoration following the authors' public open-sourced repository [14, 56]. For an additional fair comparison, we also compare with both methods at batch size one in Fig. 5, and show notable fidelity and localization improvements.

(ii) DeepInversion [48]: We also analyze performance improvements over the baseline DeepInversion method that synthesizes images conditioned on ground-truth labels.

(iii) GAN latent space projection [23]: We finally compare with the GAN-based latent code optimization method. We applied latent code projection as in StyleGAN2 [23] for BigGAN-deep generator [4] at resolution 256×256. Given no access to original images for projection loss [23], we
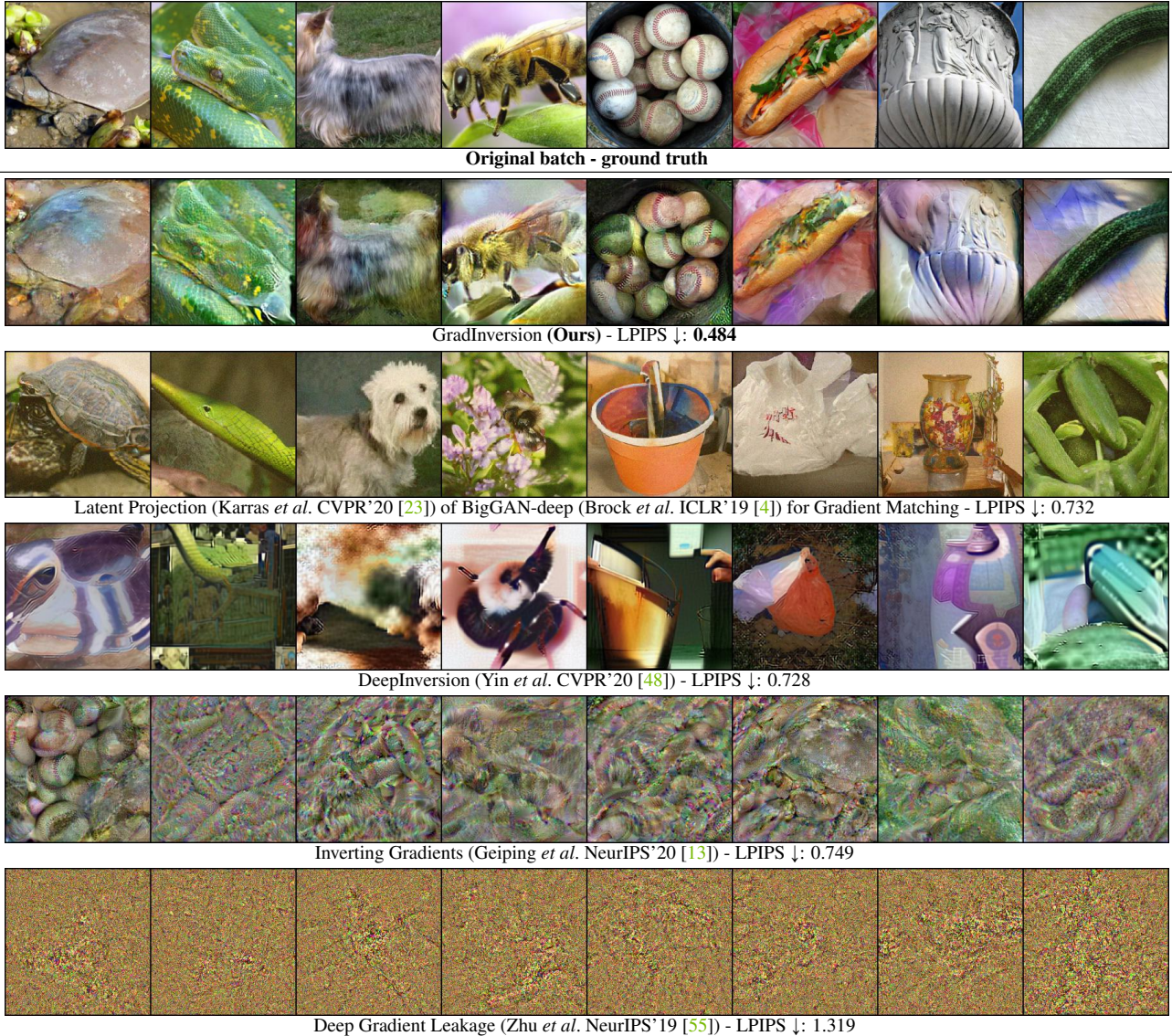
Original batch - ground truth

GradInversion (**Ours**) - LPIPS ↓: **0.484**

Latent Projection (Karras *et al.* CVPR'20 [23]) of BigGAN-deep (Brock *et al.* ICLR'19 [4]) for Gradient Matching - LPIPS ↓: 0.732

DeepInversion (Yin *et al.* CVPR'20 [48]) - LPIPS ↓: 0.728

Inverting Gradients (Geiping *et al.* NeurIPS'20 [13]) - LPIPS ↓: 0.749

Deep Gradient Leakage (Zhu *et al.* NeurIPS'19 [55]) - LPIPS ↓: 1.319

Figure 4: ImageNet batch gradient inversion for ResNet-50 visual comparison with state-of-the-art methods. GradInversion labels rearranged in ascending order to match ground truth after label restoration at 100% accuracy. Best viewed in color.

base the target loss on ℓ_2 distances between synthesized and ground truth gradients.

GradInversion outperforms prior art both visually (Fig. 4) and numerically (Table 4). Without label restoration, a joint optimizing to seek for image-label pairs [55] struggles to converge on ImageNet, as also observed by [13] even at batch size one. Total variation prior and magnitude-invariant loss as in [13] help improve reconstruction, but remain too weak to guide optimization towards ground truth. The Deep-Inversion [48] baseline improves image fidelity as expected, but inverts images with little observable links to the original batch. Projection onto BigGAN's latent space offers a balance between image fidelity and restored details, but falls short under a weaker guidance from original gradients rather than original images, as projection to latent space is

NP-hard [27] and misses visual details [23].

4.3. Effect of scaling up the batch size

We next increase the batch size. Our current analysis scales up to batch size 48 using a 32GB NVIDIA V100 GPU. As shown in Fig. 6, the amount of recoverable image content gradually decreases as batch size increases. As expected, more averaging of gradient information in a batch better protects privacy of an individual image. Surprisingly, GradInversion still unveils a decent amount of original visual information at batch size 48, and sometimes a viable complete reconstruction, as shown in Fig. 7.

Image Identifiablity Precision (IIP). We formulate a new score that measures the amount of "image-specific" features revealed by gradient inversion. Intuitively, this measures

Method	Requirements		Distance to Original Images		
	y*	GAN	FFT$_{2D}$ ↓	PSNR ↑	LPIPS ↓
Noise $\mathcal{N}(0, I)$ (starting point)	-	-	0.706	9.964	1.351
Latent projection (Karras *et al.* CVPR'20 [23]) of BigGAN-deep (Brock *et al.* ICLR'19 [4])	✓	✓	0.275	10.149	0.722
DeepInversion (Yin *et al.* CVPR'20 [48])	✓	-	0.238	10.131	0.728
Inverting Gradients (Geiping *et al.* NeurIPS'20 [13])	✓	-	0.355	11.703	0.749
Deep Gradient Leakage (Zhu *et al.* NeurIPS'19 [55])	-	-	0.602	10.252	1.319
GradInversion - BN$_{approx.}$ **(ours)**	-	-	0.232	11.235	0.633
GradInversion - BN$_{exact}$ **(ours)**	-	-	**0.175**	**12.929**	**0.484**

Table 4: Comparison of GradInversion with state-of-the-art methods for ResNet-50 gradient inversion on ImageNet1K. BN$_{approx.}$ denotes regularizing towards BN statistics in the network learnt from the original dataset; BN$_{exact}$ denotes the BN statistics of target batch shared (or leaked) in distributed setup for global BN updates, *e.g.*, Synchronized Batch Normalization [49].
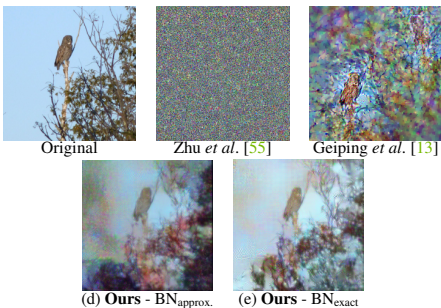

Figure 5: Comparison with prior art on ResNet-50 (ImageNet) gradient inversion at batch size 1 for a "challenging" sample from [13].
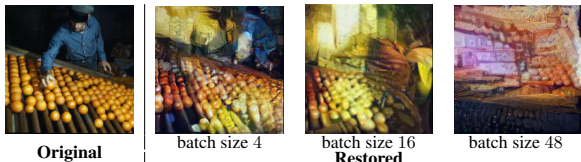

Figure 6: Reduced amount of restored original visual features as batch size increases.

how easy it is to identify a particular image, given only its reconstruction, among all its similar peers in the original dataset. Quantifiably, we calculate the fraction of exact matches between an original image and the nearest neighbor to its reconstruction. The resulting metric, referred to as *Image Identifiability Precision (IIP)*, evaluates gradient inversion strength across varying batch sizes. Fig. 8 plots the IIP curve for GradInversion. As expected, reconstruction efficacy gradually decreases as batch size increases, as also seen in Fig. 6. We make a surprising observation that many samples ($\sim 28\%$) can be correctly identified even after averaging gradient from 48 images.

The Vulnerable Population. We empirically observe a positive correlation between reconstruction efficacy and gradient magnitude. Delving deeper into this observation, we identify a new set of images that are more "vulnerable" to leak information under GradInversion. To this end, we identify one image per ImageNet class category whose gradient ℓ_2 norm is the largest within that class folder. When compared to random images in Fig. 8, batches sampled from such "vulnerable population" increases the IIP by large margins, nearly doubled at batch size 48. This advocates for careful attention
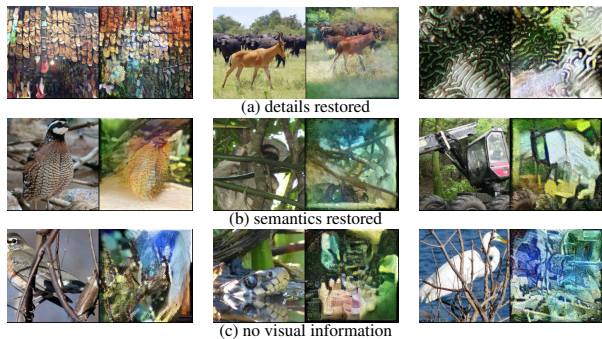

Figure 7: Varying level of information leakage at batch size 48 on ImageNet validation set. Each block containing a pair of (left) original sample and its (right) reconstruction by GradInversion.
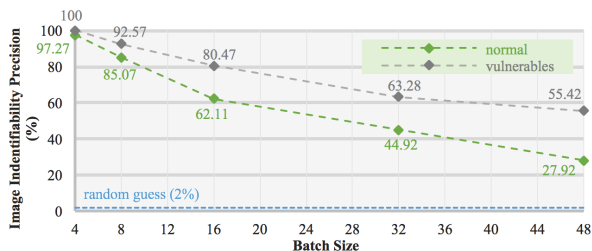

Figure 8: The Image Identifiability Precision (IIP) curve of GradInversion on ImageNet validation set, as a function of increasing batch size. Each point averaged per 256 randomly selected samples of varying batch sizes (240 samples for batch size 48). Nearest neighbors measured in avgpool feature space cosine similarity.

to such vulnerable samples before gradient sharing.

Conclusions

We introduced GradInversion to reconstruct individual images in a batch, given averaged gradients. We showed that the assumption of privacy when sharing gradients from deep networks on complex datasets even at large batch sizes, does not hold. This offers new insights into the development of privacy-preserving deep learning frameworks.

It can also be fruitful to study the underlying mechanism of information transfer that enables original data recovery from gradients. We hope that future work can study vulnerabilities of aggregation-based federated learning [2], as well as further strengthen them to prevent inversion.

References

[1] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan, et al. Towards federated learning at scale: System design. *SysML*, 2019.

[2] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth. Practical secure aggregation for privacy-preserving machine learning. In *CCS*, 2017.

[3] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi. Federated learning of predictive models from federated electronic health records. *IJMI*.

[4] A. Brock, J. Donahue, and K. Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019.

[5] Y. Cai, Z. Yao, Z. Dong, A. Gholami, M. W. Mahoney, and K. Keutzer. ZeroQ: A novel zero shot quantization framework. In *CVPR*, 2020.

[6] H. Chen, Y. Wang, C. Xu, Z. Yang, C. Liu, B. Shi, C. Xu, C. Xu, and Q. Tian. Data-free learning of student networks. In *ICCV*, 2019.

[7] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[8] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[10] Y. Du and I. Mordatch. Implicit generation and modeling with energy based models. In *NeurIPS*, 2019.

[11] M. Fredrikson, S. Jha, and T. Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *CCS*, 2015.

[12] R. Gao, Y. Lu, J. Zhou, S.-C. Zhu, and Y. Nian Wu. Learning generative convnets via multi-grid modeling and sampling. In *CVPR*, 2018.

[13] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller. Inverting gradients–How easy is it to break privacy in federated learning? In *NeurIPS*, 2020.

[14] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller. Inverting Gradients Github Source Code. https://github.com/JonasGeiping/invertinggradients, 2020. [Online; accessed 3-Nov-2020].

[15] W. Grathwohl, K.-C. Wang, J.-H. Jacobsen, D. Duvenaud, M. Norouzi, and K. Swersky. Your classifier is secretly an energy based model and you should treat it like one. *ICLR*, 2020.

[16] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of Wasserstein GANs. In *NeurIPS*, 2017.

[17] M. Haroush, I. Hubara, E. Hoffer, and D. Soudry. The knowledge within: Methods for data-free model compression. In *CVPR*, 2020.

[18] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[20] Z. He, T. Zhang, and R. B. Lee. Model inversion attacks against collaborative inference. In *ACSAC*, 2019.

[21] B. Hitaj, G. Ateniese, and F. Perez-Cruz. Deep models under the GAN: Information leakage from collaborative deep learning. In *CCS*, 2017.

[22] F. N. Iandola, M. W. Moskewicz, K. Ashraf, and K. Keutzer. Firecaffe: Near-linear acceleration of deep neural network training on compute clusters. In *CVPR*, 2016.

[23] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of StyleGAN. In *CVPR*, 2020.

[24] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

[25] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[26] T. Le, Y. Aono, T. Hayashi, et al. Privacy-preserving deep learning: Revisited and enhanced. In *ICATIS*, pages 100–110, 2017.

[27] Q. Lei, A. Jalal, I. S. Dhillon, and A. G. Dimakis. Inverting deep generative models, one layer at a time. In *NeurIPS*, 2019.

[28] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su. Scaling distributed machine learning with the parameter server. In *OSDI*, pages 583–598, 2014.

[29] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia. Path aggregation network for instance segmentation. In *CVPR*, 2018.

[30] A. Mahendran and A. Vedaldi. Understanding deep image representations by inverting them. In *CVPR*, 2015.

[31] A. Mahendran and A. Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *IJCV*, 2016.

[32] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*.

[33] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *IEEE Symp. Security and Privacy (SP)*.

[34] P. Micaelli and A. J. Storkey. Zero-shot knowledge transfer via adversarial belief matching. In *NeurIPS*, 2019.

[35] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, and H. Wu. Mixed precision training. In *ICLR*, 2018.

[36] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.

[37] A. Mordvintsev, C. Olah, and M. Tyka. Inceptionism: Going deeper into neural networks. https://research.googleblog.com/2015/06/

`inceptionism-going-deeper-into-neural.html`, 2015.

[38] A. Nguyen, J. Clune, Y. Bengio, A. Dosovitskiy, and J. Yosinski. Plug & play generative networks: Conditional iterative generation of images in latent space. In *CVPR*, 2017.

[39] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In *NeurIPS*, 2016.

[40] A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, 2015.

[41] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah. Federated learning for ultra-reliable low-latency V2V communications. In *GLOBECOM*, 2018.

[42] S. Santurkar, A. Ilyas, D. Tsipras, L. Engstrom, B. Tran, and A. Madry. Image synthesis with a single (robust) classifier. In *NeurIPS*, 2019.

[43] X. Shen, F. Darmon, A. A. Efros, and M. Aubry. RANSAC-Flow: Generic two-stage image alignment. In *ECCV*, 2020.

[44] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symp. Security and Privacy (SP)*.

[45] Y. Wang, C. Si, and X. Wu. Regression model fitting under differential privacy and model inversion attack. In *IJCAI*, 2015.

[46] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *INFOCOM*, 2019.

[47] Z. Yang, E.-C. Chang, and Z. Liang. Adversarial neural network inversion via auxiliary knowledge alignment. *arXiv preprint arXiv:1902.08552*, 2019.

[48] H. Yin, P. Molchanov, J. M. Alvarez, Z. Li, A. Mallya, D. Hoiem, N. K. Jha, and J. Kautz. Dreaming to distill: Data-free knowledge transfer via DeepInversion. In *CVPR*.

[49] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal. Context encoding for semantic segmentation. In *CVPR*, 2018.

[50] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. Self-attention generative adversarial networks. In *ICML*, 2019.

[51] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[52] Y. Zhang, R. Jia, H. Pei, W. Wang, B. Li, and D. Song. The secret revealer: Generative model-inversion attacks against deep neural networks. In *CVPR*, 2020.

[53] B. Zhao, K. R. Mopuri, and H. Bilen. iDLG: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*, 2020.

[54] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017.

[55] L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. In *NeurIPS*, 2019.

[56] L. Zhu, Z. Liu, and S. Han. Deep Leakage From Gradients Github Source Code. `https://github.com/mit-han-lab/dlg`, 2019. [Online; accessed 3-Nov-2020].