Improving Model Inversion Attacks Using Additional Backbones

Christopher Roßbach
FAU-Erlangen-Nürnberg

September 5, 2025

Abstract

Model inversion attacks aim to infer sensitive information about input or training data from the outputs or representations of machine learning models. In this work we explore model inversion in the form of reconstructing images from their feature embeddings in a known latent space. We explore the effectiveness of optimization based methods in different latent spaces. Using the discovered findings, we propose a method that leverages the embeddings of multiple reconstruction candidates in foreign latent spaces to improve reconstruction quality.

1 Introduction

Model inversion attacks pose significant risks by reconstructing private information about either training data or model inputs. This report investigates techniques that try to reconstruct the full model input from a latent representation. We limit ourselves to the task of image reconstruction from feature embeddings given by a pretrained vision backbone. We assume a white-box setting where the attacker has access to the model architecture and parameters as well as the feature embedding of the target image.

We aim to approximate the private image x^* owned by the victim from its embedding $y^* = f(x^*)$ where f is a pretrained known backbone network and y^* is the embedding known to the attacker (either leaked or purposely shared).

2 Related Work

Brief overview of previous research on model inversion attacks and defense mechanisms.

3 Methodology

We try to reconstruct the private image from its embedding through an optimization process. The problem of reconstructing the private image from its embedding is a strongly ill posed which forces us to incorporate additional constraints and priors to the optimization process. The optimization problem can be formulated as

$$\min_{\hat{x}} d_f(f(\hat{x}), y^*) + \mathcal{R}(\hat{x}),$$

where $\mathcal{R}$ is a regularization term that encourages the reconstruction to be smooth and natural and is further described in Section 3.3. d_f is a distance function that is dependent on the used backbone f. For a ResNet backbone we use the mean squared error (MSE) as the distance function while using a cosine based distance for a clip encoder.

3.1 The Iterative Base Method

We employ an iterative optimization method to progressively refine the reconstruction. We generally optimize in the image space while resorting to a lower dimensional image space at the beginning of the reconstruction. Successive upscaling encourages the reconstruction of coarse structures before refining details. We take the resolution steps used in [2] and find adding another low resolution step of 32 to be beneficial, resulting in a scale sequence of $n \in \{32, 64, 128, 224\}$ where upscaling is performed at steps 400, 900 and 1800.

At each iteration, we update the reconstruction x in the possibly smaller space $\mathbb{R}^{3\times n\times n}$ from which we retrieve the full size reconstruction proposal $\mathbb{R}^{3\times N\times N} \ni \hat{x} = S_N(x)$ by using a differentiable scaling function $S_k : \mathbb{R}^{3\times n\times n} \to \mathbb{R}^{3\times k\times k}$. The next reconstruction x is determined by

$$x^{(t)} = S_{n_t}\left(x^{(t-1)} - \eta \frac{\nabla\mathcal{L}(x^{(t-1)}, y^*)}{\|\nabla\mathcal{L}(x^{(t-1)}, y^*)\|_2}\right), \qquad (1)$$

where t denotes the iteration step, n_t the resolution at step t, η is the learning rate, and $\mathcal{L}$ the loss function. This process is repeated for 3000 iterations after which no notable improvement was noticed in practice. Normally we perform multiple reconstructions in parallel, because the result depends on the initialization $x^{(0)}$. We denote such a set of equally constructed reconstructions as $X^{(t)} = \{x_0^{(t)}, x_1^{(t)}, \dots, x_{m-1}^{(t)}\}$ where m is the number of parallel reconstructions.

For now, the loss function $\mathcal{L}$ is defined as

$$\mathcal{L}(x, y^*) = \mathcal{L}_f(x, y^*) + \mathcal{R}(S_N(x)), \tag{2}$$

with

$$\mathcal{L}_f(x, y^*) = d_f(f(A(S_N(x))), y^*), \tag{3}$$

where A is a randomized augmentation function as described in Section 3.2, and d_f is a distance function that is dependent on the used backbone f and $\mathcal{R}$ is a regularization term as described in Section 3.3.

3.2 Augmentations

As proposed in [1] we employ a set of differentiable augmentations to improve the robustness of the reconstruction. We expect the embedding to be stable under (slight) scaling, rotation and translation. We incorporate these augmentations into the optimization process as a prior by applying a (different) set of augmentations at each step of the optimization. The transformation is thereby chosen randomly from $(\text{scale}, \text{rotate}, \text{translate}_h, \text{translate}_v) \in [0.7, 1.5] \times [-30, 30] \times [-0.1, 0.1] \times [-0.1, 0.1]$.

3.3 Regularization

For regularization we use a term based on total variation as proposed in [3] with the adjustment of using a L_1 distance:

$$\mathcal{R}(x) = g_{\alpha,\beta}\left(\frac{\mathcal{R}_{TV}(x)}{(n-1)^2}\right)$$

with

$$\mathcal{R}_{TV}(x) = \sum_{i,j} |x_{i,j} - x_{i+1,j}| + |x_{i,j} - x_{i,j+1}|.$$

This term is used to penalize sharp edges in the reconstructed images. Here n refers to the width of the square image and $g_{\alpha,\beta}$ is a penalty function that balances the contribution of the total variation term and is defined as:

$$g_{\alpha,\beta}(x) = \text{relu}(x - \alpha) + \beta \cdot \text{relu}(x - \alpha)^2.$$

This quadratic dead zone penalty function allows for punishment-free variation in the interval $[-\alpha, \alpha]$ and (for big enough β) limits the total variation to be close to α. The use of this penalty function makes sure, that the this regularization term does not loose importance when adding additional terms to the objective function of the optimization. A parameter selection of $\alpha = 0.3$ and $\beta = 10$ was found to yield good results in practice.

3.4 Impact of Inverted Model

The reconstruction method described in Section 3.1 yields visibly different results when using different backbones f. It is to note, that the reconstruction for different backbones does not strictly yield better or worse reconstructions, but rather reconstruction with qualitatively different details. This effect can be seen in figure 1.

For the picture of the safe, we note that the reconstruction on a ResNet backbone gives a picture containg structural properties of a safe, while the reconstruction on the clip embedding seems to reproduce the text contained in the lower right corner of original picture. For the dog picture, the reconstruction based on the clip embedding yields a better representation of the dog's features compared to the ResNet based reconstruction, while the ResNet reconstruction contains patterns of the background. In the reconstruction of the fish picture the ResNet based reconstruction focuses on the fish's shape and the hands while the clip based reconstruction reconstructs the face of the person holding the fish.

From this observation we conclude, that a improvement in reconstruction quality can be achieved by combining different backbones in the reconstruction process while maintaining the assumption to have access to a single embedding.

3.5 Average of Foreign Embeddings

Another important observation is that if we have two different backbones f and h, a private image x^* and a slightly modified version x_p^* of the private image for which $f(x^*) \approx f(x_p^*)$ then we also observe $h(x^*) \approx h(x_p^*)$. That is, if we modify the private image only slightly, the embeddings produced by different backbones remain consistent. However, this behaviour does not apply to reconstructions: for a reconstruction $\hat{x}_f$ obtained by procedure described in 3.1 with respect to the backbone f that suffices $f(\hat{x}_f) \approx f(x^*)$ we observe strong differences in the embedding under h, $d_h(h(\hat{x}_f), h(x^*)) \gg d_f(f(\hat{x}_f), f(x^*)) \approx 0$. The same is true for two different reconstructions $\hat{x}_{f,0}$ and $\hat{x}_{f,1}$: while $f(\hat{x}_{f,0}) \approx f(\hat{x}_{f,1})$ we observe $h(\hat{x}_{f,0}) \not\approx h(\hat{x}_{f,1})$.

Furthermore, if we have a set of reconstruction $X_f = \{\hat{x}_{f,0}, \hat{x}_{f,1}, \dots\}$ we observe that that the average embedding under h of the reconstructions X_f

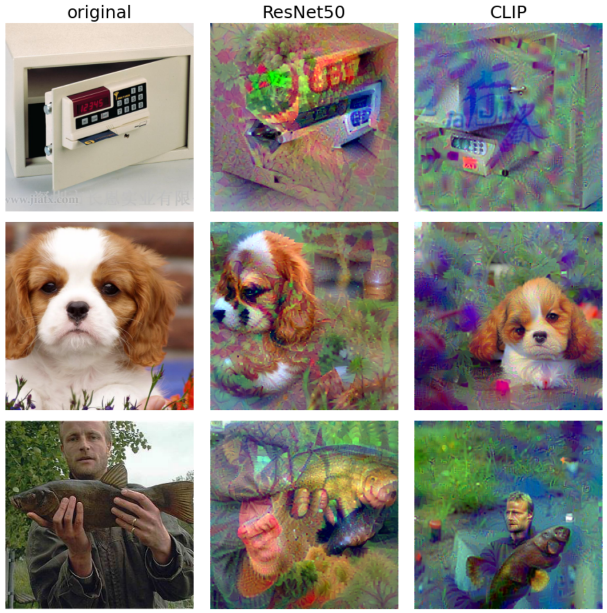

Figure 1: Comparison of reconstructions on ResNet and CLIP backbones.

is closer to the embedding und h of the private image then the average distance of the reconstructions:

$$d_h\left(\overline{h(X_f)}, h(x^*)\right) < \overline{d_h(h(X_f), h(x^*))}.$$

We often observe the even stronger statement, that the averaged embedding is closer to the embedding of the private image then every single reconstruction:

$$\forall \hat{x}_f \in X_f : d_h\left(\overline{h(X_f)}, h(x^*)\right) < d_h(h(\hat{x}_f), h(x^*)).$$

This effect can be seen in figure 2.

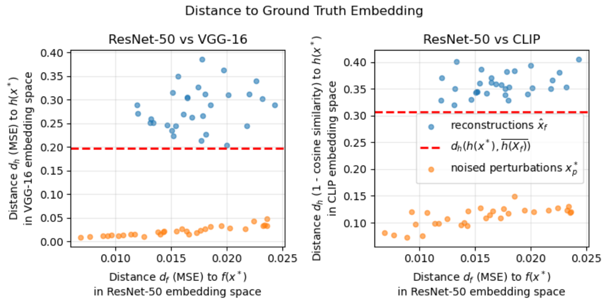

Figure 2: Comparison of embedding distances for noised images, reconstructions and averaged reconstruction embeddings for f being a ResNet backbone, and h being a VGG backbone (left) and h being a CLIP backbone (right).

That leads to the intuitive assumption the that embeddings under a foreign backbone h scatter around the embedding of the private image $h(x^*)$ and do not deviate in a specific direction. This idea is illustrated in figure 3.

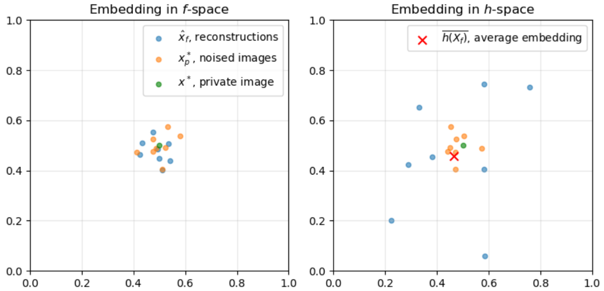

Figure 3: Illustration of the imagined scattering behavior of embeddings under a backbone used for reconstruction f (left) and a foreign backbone h (right). The average embedding in the foreign space is closer to the embedding of the private image than the embeddings of the reconstructions.

3.6 Combining Multiple Backbones

To potentially make use of this we explored multiple methods to incorporate the average in the foreign embedding space during the reconstruction process.

Since most of the used networks apply a ReLu activation as their last non-linearity, simply calculating the arithmetic average in each coordinate leads to a huge shift in the shape of the distribution of embeddings. To reduce that shift, we can perform the averaging in the latent space before applying the last ReLu activation. The effect of this can bee seen in figure 4. We perform the experiments on both, the truncated and the original backbones.

We incorporate the average embedding in a foreign space by extending loss function from equation 2:

$$\mathcal{L}(x, y^*, y_h) = \mathcal{L}_f(x, y^*) + \mathcal{L}_h(x, y_h) + \mathcal{R}(S_N(x)) \quad (4)$$

where $\mathcal{L}_h$ is defined as in equation 3 and y_h is a element in the foreign space. We will choose y_h to either be $\overline{h(X^{(t)})}$, the average embedding of the current reconstructions in the foreign space, or $\overline{h(X_f)}$, the average embedding of reconstructions of previously completed reconstruction run with the loss function from equation 2.

4 Experiments

We conducted a series of experiments to evaluate the effectiveness of the proposed method. We experimented with different choices of the foreign backbone h and the construction of y_h.

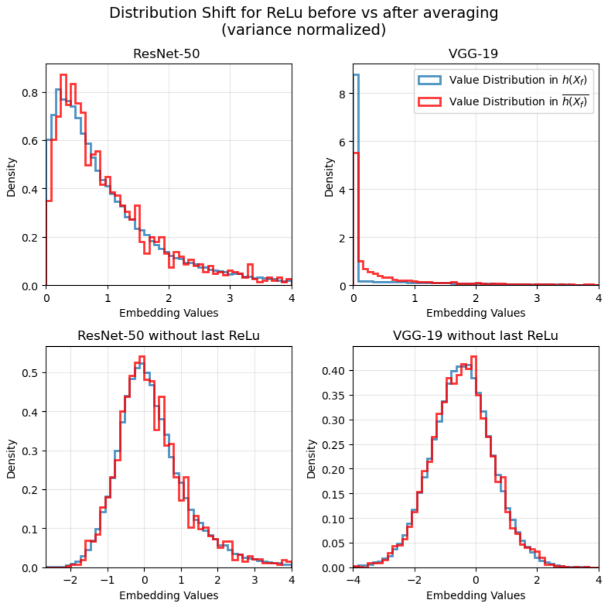

Figure 4: Distribution of embeddings in different backbones. Top: Value distribution under original backbone. Bottom: Value distribution under modified backbone with the last ReLu removed.

For the presented experiments we fixed f to be a ResNet-50, pretrained on ImageNet without its classification layer. For h we explored different ResNet backbones, ResNet-18, ResNet-34, ResNet-50, ResNet-101 and ResNet-152 (referred to as rn18, rn34, rn50, rn101, rn152), VGG-16 and VGG-19 (referred to as vgg16 and vgg19) and a CLIP ViT-L/14 (referred to as clip). For all of them (except clip) we used a model pretrained on ImageNet without its classification layer and also performed the experiments on the original and the version without the last ReLU activation (denoted by a -nrl suffix). For the CLIP model, we used an OpenAI pretrained version as presented in [4]. As metrics for the evaluation of the reconstruction quality we used the structural similarity index measure (SSIM) [8], the learned perceptual image patch similarity (LPIPS) [7] as well as the cosine similarity based distance to the ground truth CLIP embedding of the image ($\mathcal{L}_{\text{clip}}(\hat{x}, \text{clip}(x))$). The SSIM value is supplemented by LPIPS, which is used to measure perceptual similarity, as a human viewer would perceive it and by the CLIP loss, which measures how well the reconstruction captures the semantic content of the original image.The results can be seen in table 1.

The calculation of y_h was performed by using $m = 16$ parallel reconstructions and in the two different ways described in Section 3.6: We choose y_h to be the average of the current iteration ($y_h^{\text{avg}} = \overline{h(X^{(t)})}$) or to be a reference vector obtained by independently and previously completed run ($y_h^{\text{ref}} = \overline{h(X_f)}$).

5 Discussion

Interpretation of results, limitations, and potential improvements.

5.1 Possible Extensions

This idea can be extended in multiple directions without diverging from the main concept. Possible extension in direction of how h is chosen include:

1. The identity mapping $h(x) = x$ would correspond to averaging the reconstructions in pixel space. That could encourage the reconstructions mechanism to maintain common features in pixel space that were already discovered.

2. Choosing h as an autoencoder trained on a large corpus of images could encourage the reconstructions to maintain features that are common in natural images.

3. If the attacker has a suspicion about the distribution of the private image, h could be chosen as a backbone trained on a similar distribution.

4. If the attacker is only interested in the reconstruction of certain parts of the image, h could be chosen as a backbone trained for that specific task (e.g. face recognition).

5. In the same manner the backbone f could be appended by a task specific head c to encourage the reconstruction of certain features and h can be chosen to be $c \circ f$.

6. If x^* stems from a different distribution than the data used to train f, the effect of using foreign backbones h may increase. Especially if h is trained on a distribution similar to the one of x^*.

Possible extensions in direction of how y_h is calculated include:

1. Instead of using the average embedding in the foreign space, one could use a more robust statistic like the median or a trimmed mean.

2. One could also use a clustering algorithm to identify groups of similar reconstructions and use the centroid of the largest cluster as y_h.

3. For location sensitive embeddings h (e.g. $h = \mathrm{id}$) using image registration method like RANSAC-Flow [5] before averaging could be beneficial as this method already was shown to be effective for other reconstruction scenarios as in [6].

Extensions towards adjusting the loss and distance functions include:

1. Another extension would be to adjust the distance function d_h to punish certain components differently depending on the agreement of the reconstructions in the foreign space in that components.

2. Combining multiple losses $\mathcal{L}_{h_1}, \mathcal{L}_{h_2}, \ldots$ for different backbones $h_1, h_2, \ldots$ is a simple but not yet explored extension.

6 Conclusion

Summary of findings and suggestions for future work.

References

[1] Amin Ghiasi, Hamid Kazemi, Steven Reich, Chen Zhu, Micah Goldblum, and Tom Goldstein. Plug-In Inversion: Model-Agnostic Inversion for Vision with Data Augmentations.

[2] Hamid Kazemi, Atoosa Chegini, Jonas Geiping, Soheil Feizi, and Tom Goldstein. What do we learn from inverting CLIP models?

[3] Aravindh Mahendran and Andrea Vedaldi. Understanding Deep Image Representations by Inverting Them. pages 5188–5196.

[4] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8748–8763. PMLR.

[5] Xi Shen, François Darmon, Alexei A. Efros, and Mathieu Aubry. RANSAC-Flow: Generic Two-Stage Image Alignment. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 618–637. Springer International Publishing.

[6] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M. Alvarez, Jan Kautz, and Pavlo Molchanov. See through Gradients: Image Batch Recovery via GradInversion. pages 16332–16341. IEEE.

[7] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. pages 586–595.

[8] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: From error visibility to structural similarity. 13(4):600–612.

Type	ReLU	h	SSIM ↑	LPIPS VGG ↓	LPIPS Alex ↓	CLIP cos sim ↑	# Runs
baseline	–	none	0.120 (+0.000)	0.770 (+0.000)	0.785 (+0.000)	0.671 (+0.000)	4
avg	no	rn18	0.171 (+0.052)	0.822 (+0.052)	0.811 (+0.026)	0.589 (-0.081)	8
avg	yes	rn18	0.171 (+0.052)	0.800 (+0.030)	0.802 (+0.017)	0.600 (-0.070)	8
ref	no	rn18	0.088 (-0.032)	0.814 (+0.043)	0.829 (+0.044)	0.647 (-0.024)	1
ref	yes	rn18	0.085 (-0.035)	0.803 (+0.033)	0.826 (+0.041)	0.641 (-0.030)	1
avg	no	rn34	0.171 (+0.051)	0.804 (+0.034)	0.801 (+0.017)	0.570 (-0.100)	8
avg	yes	rn34	0.171 (+0.052)	0.795 (+0.024)	0.796 (+0.011)	0.577 (-0.094)	8
ref	no	rn34	0.084 (-0.036)	0.792 (+0.022)	0.826 (+0.041)	0.661 (-0.010)	1
ref	yes	rn34	0.086 (-0.034)	0.788 (+0.018)	0.819 (+0.034)	0.659 (-0.011)	1
avg	no	rn50	0.174 (+0.054)	0.770 (+0.000)	0.794 (+0.010)	0.627 (-0.044)	8
avg	yes	rn50	0.174 (+0.054)	0.762 (-0.008)	0.789 (+0.005)	0.627 (-0.043)	8
ref	no	rn50	0.089 (-0.031)	0.765 (-0.005)	0.814 (+0.029)	0.654 (-0.017)	1
ref	yes	rn50	0.090 (-0.030)	0.761 (-0.009)	0.806 (+0.021)	0.643 (-0.028)	1
avg	no	rn101	0.171 (+0.051)	0.768 (-0.002)	0.780 (-0.005)	0.618 (-0.052)	8
avg	yes	rn101	0.172 (+0.052)	0.760 (-0.010)	0.776 (-0.009)	0.625 (-0.046)	8
ref	no	rn101	0.087 (-0.033)	0.770 (-0.000)	0.822 (+0.037)	0.651 (-0.019)	1
ref	yes	rn101	0.089 (-0.031)	0.764 (-0.007)	0.809 (+0.025)	0.658 (-0.012)	1
avg	no	rn152	0.174 (+0.054)	0.765 (-0.005)	0.782 (-0.003)	0.611 (-0.060)	8
avg	yes	rn152	0.173 (+0.053)	0.761 (-0.009)	0.780 (-0.005)	0.624 (-0.047)	8
ref	no	rn152	0.090 (-0.030)	0.755 (-0.015)	0.804 (+0.019)	0.668 (-0.002)	1
ref	yes	rn152	0.091 (-0.029)	0.758 (-0.012)	0.803 (+0.019)	0.646 (-0.025)	1
avg	no	vgg16	0.180 (+0.060)	0.870 (+0.100)	0.850 (+0.065)	0.560 (-0.110)	8
avg	yes	vgg16	0.175 (+0.055)	0.780 (+0.010)	0.789 (+0.004)	0.615 (-0.055)	8
ref	no	vgg16	0.094 (-0.026)	0.808 (+0.038)	0.835 (+0.050)	0.623 (-0.048)	1
ref	yes	vgg16	0.093 (-0.027)	0.778 (+0.008)	0.807 (+0.022)	0.641 (-0.030)	1
avg	no	vgg19	0.170 (+0.050)	0.848 (+0.078)	0.864 (+0.080)	0.555 (-0.116)	8
avg	yes	vgg19	0.174 (+0.054)	0.764 (-0.006)	0.779 (-0.005)	0.616 (-0.055)	8
ref	no	vgg19	0.083 (-0.037)	0.831 (+0.061)	0.837 (+0.053)	0.617 (-0.053)	1
ref	yes	vgg19	0.087 (-0.033)	0.792 (+0.022)	0.817 (+0.032)	0.632 (-0.039)	1
avg	no	clip	0.168 (+0.048)	0.764 (-0.006)	0.796 (+0.012)	0.572 (-0.099)	8
avg	yes	clip	0.167 (+0.048)	0.764 (-0.006)	0.797 (+0.012)	0.569 (-0.102)	8
ref	no	clip	0.098 (-0.022)	0.814 (+0.044)	0.843 (+0.058)	0.693 (+0.023)	2
ref	yes	clip	0.103 (-0.017)	0.794 (+0.024)	0.793 (+0.008)	0.678 (+0.007)	2

Table 1: Comparison of reconstruction quality for different choices of y_h. SSIM and CLIP cos sim: higher is better. LPIPS: lower is better. Each reported value is the average over 8 different images chosen randomly from the imagenet validation set. For each of the images 16 reconstructions were performed and their metrics were averaged.