7 Appendix

7.1 Score matching details

We found that using score matching [32] does not work reliably when the data's structure lies on a low-dimensional manifold (e.g., natural images). We found that applying randomized smoothing [56], which adds Gaussian noise to the image for robust training, helps stabilize score matching as it smoothens the density function. Randomized smoothing also makes the bound tighter. We observed that adding a reasonable amount of noise (*e.g.,* standard deviation of 0.25, which was originally used by Cohen et al. [56]) works well in general, but adding only small noise (standard deviation of 0.01) does not. We show both results in Section 3.3.1.

7.2 Hyperparameters

Attacks For attacks in Section 3.3.1, 3.3.2, and 4.2, we used the following hyperparameters. For the optimizer-based attack for Gaussian synthetic input, we used Adam with lr=10^{-3}, and λ=0.1–100 for the regularizer. For the optimizer-based attack for NCF-MLP and DistilBert, we used Adam with lr=0.1. For the DNN-based attack for MNIST and CIFAR-10 (Figure 2, 3, 5), we used a modified DNN from Li et al. [20], which uses a series of convolution (Conv) and convolution transpose (ConvT) layers interspersed with leaky ReLU of slope 0.2. All the models were trained for 100 epochs using Adam with lr=10^{-3}. Below summarizes the architecture parameters. For DNN-based attacks in Section 3.3.2, we put a sigmoid at the end. For the attack in Section 4.2, we do not.

Table 3: DNN attacker architectures used in the paper. Output channel dimension (c_{out}), kernel size (k), stride (s), and output padding (op) are specified. Input padding was 1 for all layers.

Dataset + encoder	Architecture
MNIST + Conv	3×Conv(c_{out}=16, k=3, s=1) + ConvT(c_{out}=32, k=3, s=1, op=0) + ConvT(c_{out}=1, k=3, s=1, op=0)
CIFAR-10 + split-early	3×Conv(c_{out}=64, k=3, s=1) + ConvT(c_{out}=128, k=3, s=1, op=0) + ConvT(c_{out}=3, k=3, s=1, op=0)
CIFAR-10 + split-middle	3×Conv(c_{out}=128, k=3, s=1) + ConvT(c_{out}=128, k=3, s=2, op=1) + ConvT(c_{out}=3, k=3, s=2, op=1)
CIFAR-10 + split-late	3×Conv(c_{out}=256, k=3, s=1) + 2×ConvT(c_{out}=256, k=3, s=2, op=1) + ConvT(c_{out}=3, k=3, s=2, op=1)

Split inference Below are the hyperparameters for the models used in Section 4.2. For ResNet-18, we used an implementation tuned for CIFAR-10 dataset from [70], with ReLU replaced with GELU and max pooling replaced with average pooling. We used the default hyperparameters from the repository except for the following: bs=128, lr=0.1, and weight_decay=5×10^{-4}. For NCF-MLP, we used an embedding dimension of 32 and MLP layers of output size [64, 32, 16, 1]. We trained NCF-MLP with Nesterov SGD with momentum=0.9, lr=0.1, and batch size of 128 for a single epoch. We assumed 5-star ratings as click and others as non-click. For DistilBert, we used Adam optimizer with a batch size of 16, lr=2×10^{-5}, β_1=0.9, β_2=0.999, and $\epsilon = 10^{-8}$. We swept the compression layer channel dimension among 2, 4, 8, 16, and the SNR regularizer λ between 10^{-3} and 100.

Training Below are the hyperparameters for the models evaluated in Section 5.2. We used the same model and hyperparameters with split inference for training the encoder with the pretraining dataset. Then, we freeze the layers up to block 4 and trained the rest for 10 epochs with CIFAR-10, with lr=10^{-3} and keeping other hyperparameters the same.

Execution Environment All the evaluation was done on a single A100 GPU. The training and evaluation of each model ranged roughly from less than an hour (ResNet-18 with split-early, NCF-MLP) to 3–7 hours (ResNet-18 with split-late, DistilBert).

7.3 van Trees inequality

Below, we restate the van Trees Inequality [67], which we use to prove Theorem 5.

Theorem 2 (Multivariate van Trees inequality)**.** *Let $(\mathcal{X}, \mathcal{F}, P_\theta : \theta \in \Theta)$ be a family of distributions on a sample space $\mathcal{X}$ dominated by μ. Let $p(\mathbf{x}|\theta)$ denote the density of $X \sim P_\theta$ and $\mathcal{I}_\mathbf{x}(\theta)$ denotes its FIM. Let $\theta \in \Theta$ follows a probability distribution π with a density $\lambda_\pi(\theta)$ with respect to Lebesgue measure. Suppose that λ_π and $p(\mathbf{x}|\theta)$ are absolutely μ-almost surely continuous and λ_π converges to 0 and the endpoints of Θ. Let ψ be an absolutely continuous function of θ, and ψ_n an arbitrary estimator of $\psi(\theta)$. Assume regularity conditions from Section 2.2 is met. If we make n observations $\{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}\}$, then:*

$$\int_\Theta \mathbb{E}_\theta[||\psi_n - \psi(\theta)||_2^2]\lambda_\pi(\theta)d\theta \geq \frac{(\int \operatorname{div}\psi(\theta)\lambda_\pi(\theta)d\theta)^2}{n\int \operatorname{Tr}(\mathcal{I}_\mathbf{x}(\theta))\lambda_\pi(\theta)d\theta + \operatorname{Tr}(\mathcal{J}(\lambda_\pi))},$$

where div *is the divergence operator.*

7.4 Proof of Corollary 1

Proof. Let ψ be an identity transformation $\psi(\theta) = \theta$. For the setup in Corollary 1, $n = 1$ and $\operatorname{div}(\mathbf{x}) = d$, so the multivariate van Trees inequality from Theorem 2 reduces to:

$$\mathbb{E}_\pi\mathbb{E}_\theta[||\hat{\mathbf{x}} - \mathbf{x}||_2^2/d] \geq \frac{d}{\mathbb{E}_\pi[\operatorname{Tr}(\mathcal{I}_\mathbf{e}(\mathbf{x}))] + \operatorname{Tr}(\mathcal{J}(f_\pi))} = \frac{1}{\mathbb{E}_\pi[\operatorname{dFIL}(\mathbf{x})] + \operatorname{Tr}(\mathcal{J}(f_\pi))/d}$$

□

7.5 Comparison with differential privacy

Differential privacy [42] is not well-suited for instance encoding, as we discuss in Section 2.1. We formulate and compare a DP-based instance encoding and compare it with our dFIL-based instance encoding in a split inference setup (Section 4) to show that DP-based instance encoding indeed does not work well.

To formulate DP for instance encoding, we define an adjacent set $\mathcal{D}$ and $\mathcal{D}'$ as two differing inputs. A randomized method $\mathcal{A}$ is (α, ϵ)-Rényi differentially private (RDP) if $D_\alpha(\mathcal{A}(\mathcal{D})||\mathcal{A}(\mathcal{D}')) \leq \epsilon$ for $D_\alpha(P||Q) = \frac{1}{\alpha-1}\log \mathbb{E}_{x\sim Q}[(\frac{P(x)}{Q(x)})^\alpha]$. As DP provides a different privacy guarantee with dFIL, we use the theorem from Guo et al. [28] to derive an MSE lower bound using DP's privacy metric for an unbiased attacker. Assuming a reconstruction attack $\hat{\mathbf{x}} = \operatorname{Att}(\mathbf{e})$ that reconstructs $\mathbf{x}$ from the encoding $\mathbf{e} = \operatorname{Enc}(\mathbf{x})$, repurposing the theorem Guo et al. [28] gives:

$$\mathbb{E}[||\hat{\mathbf{x}} - \mathbf{x}||_2^2/d] \geq \frac{\Sigma_{i=1}^d \operatorname{diam}_i(\mathcal{X})^2/4d}{e^\epsilon - 1} \tag{6}$$

for a (2, ϵ)-RDP Enc, where $\mathcal{X}$ is the input data space. We can construct a (2, ϵ)-RDP encoder Enc_{RDP} from a deterministic encoder Enc_D by scaling and clipping the encoding adding Gaussian noise, or $\operatorname{Enc}_{RDP} = \operatorname{Enc}_D(\mathbf{x})/\max(1, \frac{||\operatorname{Enc}_D(\mathbf{x})||_2}{C}) + \mathcal{N}(0, \sigma^2)$, similarly to [42]. The noise to be added is $\sigma = \frac{(2C)^2}{\epsilon}$ [71]. Equation 6 for DP is comparable to Equation 2 for dFIL, and we use the two equations to compare DP and dFIL parameters. We use Equation 2 because [28] does not discuss the bound against biased attackers.

We evaluate both encoders for split inference using CIFAR-10 dataset and ResNet-18. We split the model after block 4 (split-middle from Section 4.2.1) and did not add any optimizations discussed in Section 4 for simplicity. For the DP-based encoder, we retrain the encoder with scaling and clipping so that the baseline accuracy without noise does not degrade. We ran both models without standardizing the input, which makes $\operatorname{diam}_i(\mathcal{X}) = 1$ for all i.

Table 4 compares the test accuracy achieved when targeting the same MSE bound for an unbiased attacker using dFIL and DP, respectively. The result clearly shows that DP degrades the accuracy much more for similar privacy levels (same unbiased MSE bound), becoming impractical very quickly. DP suffers from low utility because DP is agnostic with the input and the model, assuming a worst-case input and model weights. Our dFIL-based bound uses the information of the input and model weights in its calculation of the bound and can get a tighter bound.

Table 4: Test accuracy when targeting the same MSE bound.

Unbiased MSE bound	1e-5	1e-4	1e-3	1e-2
dFIL-based	**93.09%**	**93.11%**	**92.52%**	**87.52%**
DP-based	64.64%	56.68%	46.46%	33%

7.6 Attack based on a diffusion model

We additionally designed a powerful, diffusion model-based reconstruction attacker to study the privacy of dFIL against the best-effort attacker, motivated from the fact that recently developed diffusion models [72] are excellent denoisers. During training of a diffusion model, (1) a particularly-designed noise is added to an image, and (2) a DNN is trained to predict and remove the noise [72]. The first part can be thought of as an instance encoder (that is purposely made easy to invert), and we can calculate its dFIL. The second part can be thought of as a reconstruction attacker. As the noising and denoising are specifically designed for the denoising to work well, we expect a mature, pretrained diffusion model to give a very good attack quality. We used DDPM [72] pretrained with CIFAR-10 from Google [73].

Figure 7 shows the result. The first column of each row shows the original image, and other columns show the reconstruction of our DDPM-based attacker with different dFIL. We scale and show dFIL with the same scale as Figure 5, as DDPM works with a different normalized image that produces dFIL at a different scale. Our new attack provided an interesting result: the attack was able to reconstruct conceptually similar images with the original image even when pixel-by-pixel reconstruction was prohibited by high $1/\operatorname{dFIL}$. For example, 7th row at $1/\operatorname{dFIL} = 49.5$ (6th column) successfully reconstructed a white car with a red background, although the reconstruction MSE was high and the design of the reconstructed car was nothing like the original image. The result shows that high-level information of the image (*e.g.*, the color of the car/background, the orientation of the car, *etc.*) can still be preserved after encoding with a relatively high $1/\operatorname{dFIL}$, which is why it is possible to perform downstream training/inference with a privately-encoded data without revealing the original data.

7.7 Additional figures and tables

Table 5: The reconstruction quality of an input is highly correlated with dFIL. Correct parts in bold.

1/dFIL	Reconstructed text (from split-early)
10^{-5}	**it's a charming and often affecting journey.**
1	**it's** cones **charming**ound **often affecting journey** closure
10	grounds yuki cum sign recklessound fanuche pm stunt

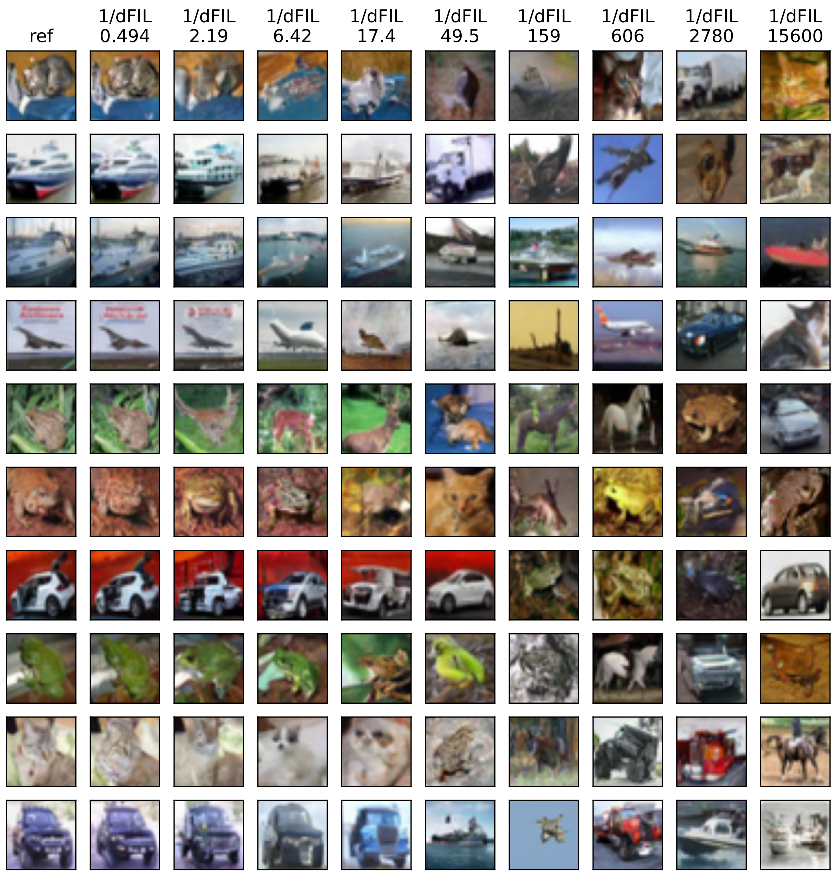
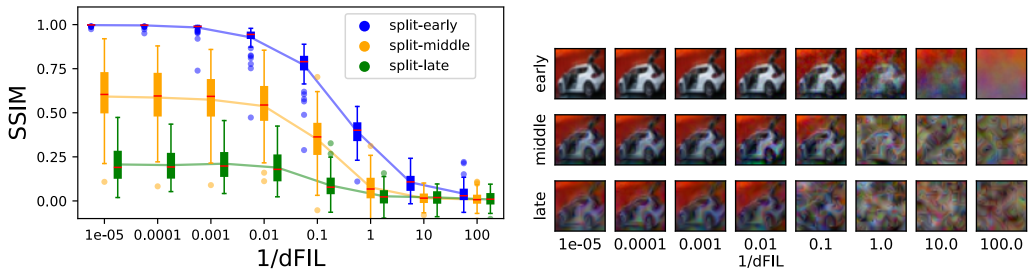


Figure 7: Results from DDPM [72]-based reconstruction attack. High $1/\,\mathrm{dFIL}$ prevents exact pixel-to-pixel reconstruction, but images that share some high-level features with the original image can be generated unless $1/\,\mathrm{dFIL}$ is not too high.



(a) 1/dFIL vs. SSIM

(b) 1/dFIL vs. reconstructed image quality

Figure 8: Optimizer-based attack with total variation (TV) prior [34] against our split inference system in Section 4. The trend is very similar to Figure 5.

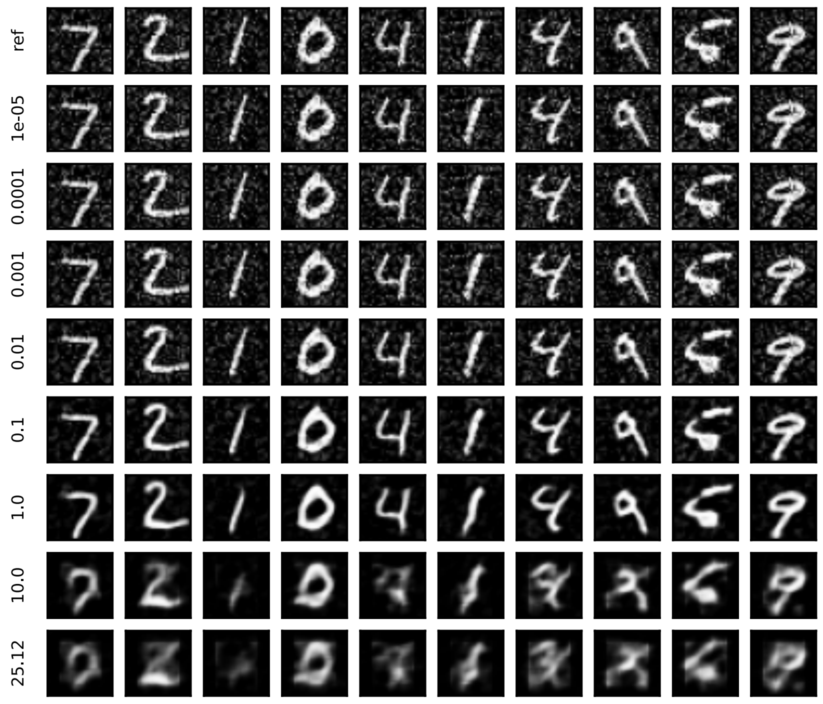


Figure 9: More reconstruction result of Figure 2.

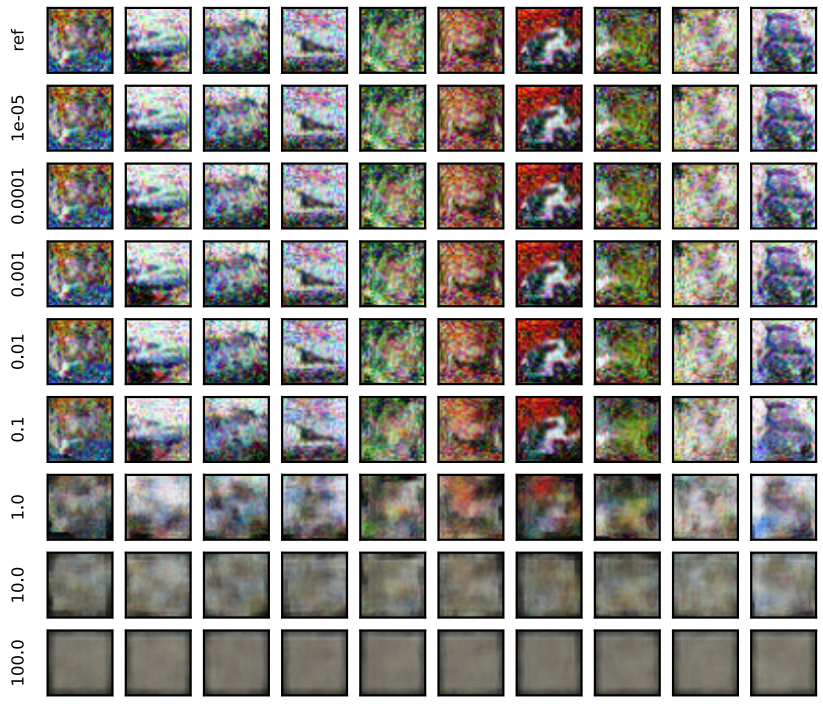


Figure 10: More reconstruction result of Figure 3.

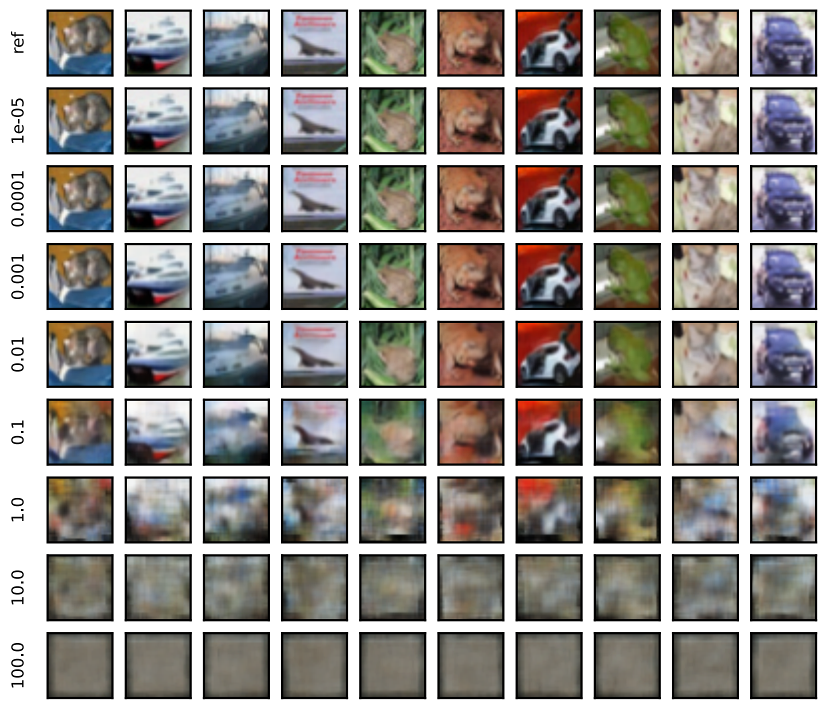
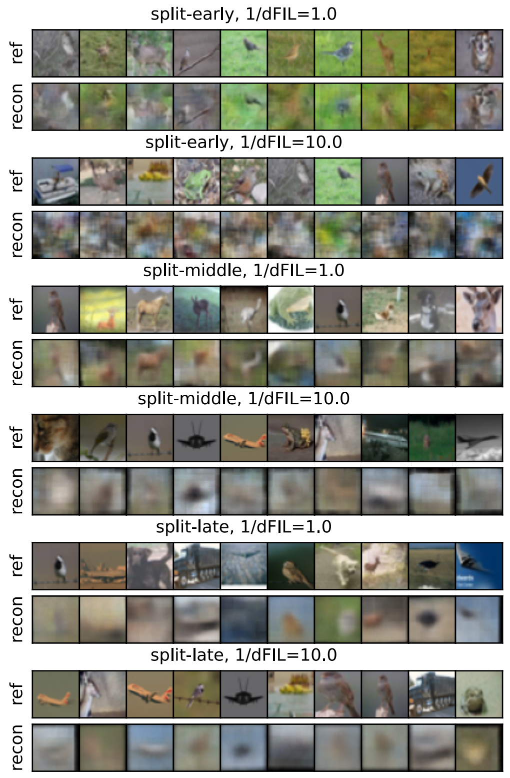
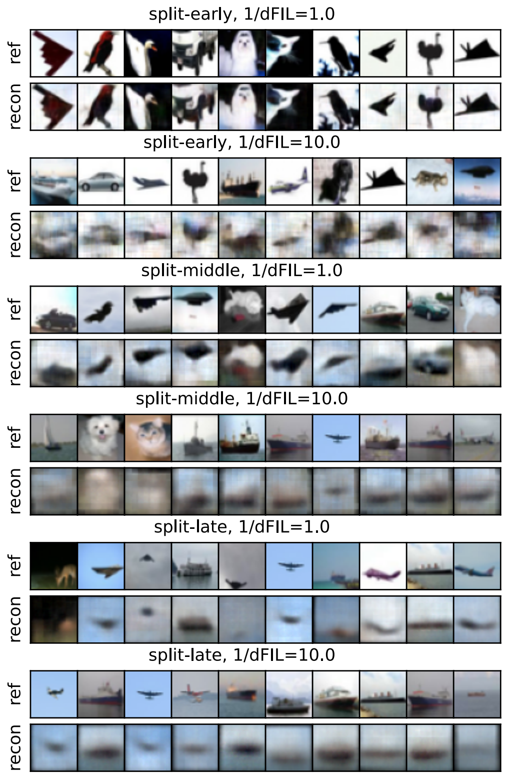


Figure 11: More reconstruction result of Figure 4.



(a) Reconstructions with the worst SSIM values.



(b) Reconstructions with the best SSIM values.

Figure 12: Ten reconstructions with the best and worst SSIM values for various setups. The result is an extension of Figure 5. Images with a simple shape and high color contrast tend to be reconstructed more easily, which matches our intuition. Omitting results from $1/\,\mathrm{dFIL} = 100$, as no meaningful reconstruction was possible.

Figure 13: Ten images with the lowest and highest dFIL values, for split-middle setup in Figure 5. Images with high dFIL tend to have a simpler shape and a high color contrast, potentially being easier to reconstruct. Individual dFIL value of each samples can potentially be used to detect data that are more leaky.