# Trainig Data Attribution for Diffusion Models

# Key Question

What is the *influence* of a piece of training data over a given generated sample?

# Key Question (Reformulated)

If the model *had not been trained* on this piece of training data, *how different* would the model output look?

# Strategy

Given: a pretrained model, exogenous noise (input), sample generated by the exogenous noise, a piece of training data

1. Unlearn the piece of training data from the model
2. Generate a *counterfactual sample* using the exogenous noise
3. Compare the original sample and the counterfactual sample

# Countefactual Sampling

Recall that (roughly speaking) DDPM generates an image by recursively sampling:

$$x_T = z_T \tag{1}$$

$$x_{t-1} = \mu_\theta(x_t, t) + z_{t-1} \tag{2}$$

where $z_t \sim \mathcal{N}(0, I)$. Hence by keeping track of $(z_t)$, one can generate a counterfactual sample by

$$\tilde{x}_T = z_T \tag{3}$$

$$\tilde{x}_{t-1} = \mu_{\tilde{\theta}}(\tilde{x}_t, t) + z_{t-1}, \tag{4}$$

to measure only the effect of change in parameters on the generated samples.

# Challenges

1. Unlearning training data
   - Retrain the model from scratch $\rightarrow$ Expensive
   - Use approximation methods $\rightarrow$ Hard to assess the accuracy in the context of diffusion models
2. Generation of counterfactuals
   - Especially expensive for diffusion models
   - E.g. evaluate counterfactuals for *all* training data samples

# Encoded Ensembles

- Diffusion model $f(x, t, \theta)$, where $x$ is the input, $t$ is the time, and $\theta$ is the trainable parameters
- $X$: total training data, $\mathcal{X} = 2^X$
- $\mathcal{A}(S, r)$: training procedure that takes a set of training samples and the exogenous noise, and outputs the trained parameters

Denote an ensemble of diffusion models $f_e$ as

$$f_e(x, t) = \mathop{\mathbb{E}}_{S \sim \mathcal{X}} \left[ \mathop{\mathbb{E}}_{r \sim R} \left[ f\left(x, t, \mathcal{A}(S, r)\right)\right]\right]. \tag{5}$$

Then if we only consider models trained with subsets that do not contain some point $\tilde{x}$, we have

$$f_e^{-\tilde{x}}(x, t) = \frac{1}{\Pr(\tilde{x} \in S' \sim \mathcal{X})} \mathop{\mathbb{E}}_{S \sim \mathcal{X}} \left[ \mathop{\mathbb{E}}_{r \sim R} \left[ f(x, t, \mathcal{A}(S, r) \mathbf{1}_{\tilde{x} \notin S}\right]\right]. \tag{6}$$

# Encoded Ensembles

How to form an ensemble:

1. Assign a unique $n$ length bit vector with fixed hamming weight $h$ (exactly $h$ elements are nonzero) to each training sample.
2. Create $n$ training subsets where $i$th subset contains exactly the training samples whose $i$th element of assigned bit vector is 1.
3. Form *encoded ensemble* $\hat{f}_e$ using $n$ models that are independently trained with $n$ training subsets.

# Encoded Ensembles

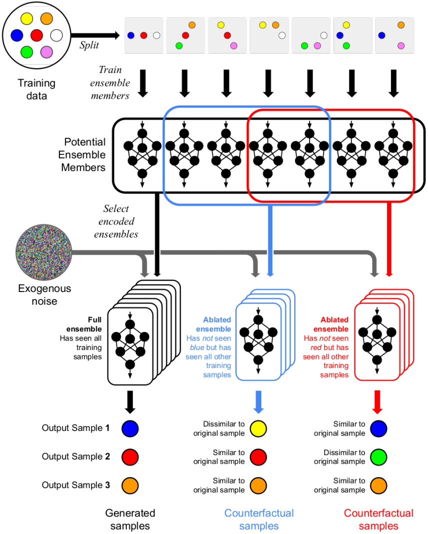


Figure 1: Encoded ensembles

# Encoded Ensembles

**Advantages**

- Only need $\mathcal{O}(\log(|X|)^{1+\epsilon})$ models for any arbitrary nonzero $\epsilon$.
- For each sample in the training data, there exists at least one model in the ensemble that has not seen it during training.
- (With mild condition), as $n$ grows, the encoded ensemble $\hat{f}_e$ is a unbiased estimator of $f_e$.
- Similarly as $n$ grows, $\hat{f}_e^{-\tilde{x}}$ is a unbiased estimator of $f_e^{-\tilde{x}}$.

# Encoded Ensembles

Does ensemble generate coherent output?

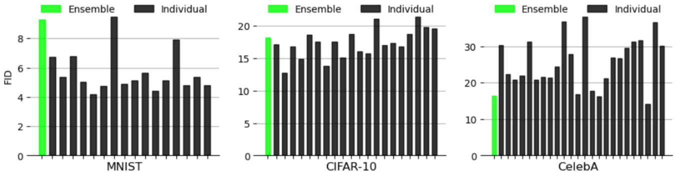


Figure 2: FID scores of ensemble and individual models

# Encoded Ensembles

Does ensemble converges?

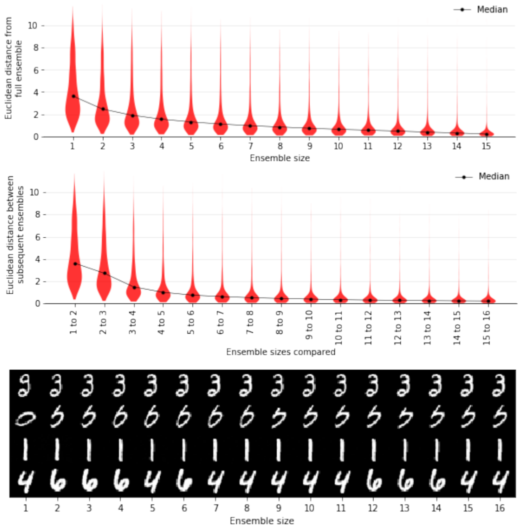


Figure 3: Ensemble converges as size increases

# Jacobian Approximation

Let $\theta_1, \cdots, \theta_n$ be the parameters for the models of the given encoded ensemble $\hat{f}_e$, and let $v$ be a $n$-dimensional vector with non-negative entries. Then define an operation $\cdot$ by

$$\left(\hat{f}_e \cdot v\right)(x,t) = \sum_{i=1}^{n} f(x,t,\theta_i) v_i. \tag{7}$$

Note that for $u_0 = (1/n, \cdots, 1/n)$, we have

$$\hat{f}_e \cdot u_0 = \hat{f}_e. \tag{8}$$

Also for a given point $\tilde{x}$, we can define $u_{-\tilde{x}}$ such that

$$\hat{f}_e \cdot u_{-\tilde{x}} = \hat{f}_e^{-\tilde{x}}. \tag{9}$$

# Jacobian Approximation

Now define $y(v, \epsilon)$ as the sample generated by an exogenous noise $\epsilon$, and denoiser $\hat{f}_e \cdot v$. Then for given $v$ and fixed exogenous noise $\epsilon$, using the first-order Taylor expansion around $u_0$, we have

$$y(v, \epsilon) = y(u_0, \epsilon) + \frac{\partial y(x, \epsilon)}{\partial x}\Big|_{x=u_0} (v - u_0) + \mathcal{O}(\|v - u_0\|^2). \quad (10)$$

Hence we can approximate the counterfactual sample by

$$y(u_{-\tilde{x}}, \epsilon) \simeq y(u_0, \epsilon) + \frac{\partial y(x, \epsilon)}{\partial x}\Big|_{x=u_0} (u_{-\tilde{x}} - u_0). \quad (11)$$

# Jacobian Approximation

How well does influecne measured by Jacobian approximation aligns with true influence?

- Last step predicted noise: At last step, drop the outputs of all ablated models
- Individual models: Negate residuals (individually generated image - original image) of all ablated models

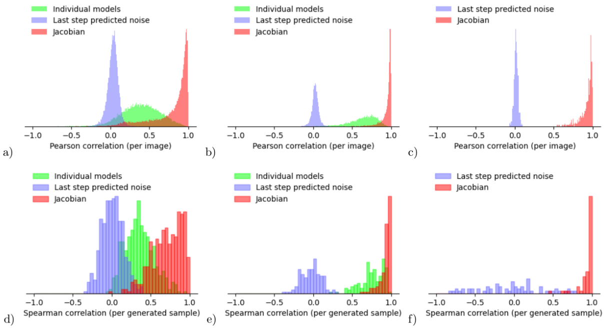


Figure 4: Correlation with true influence

# Experiments

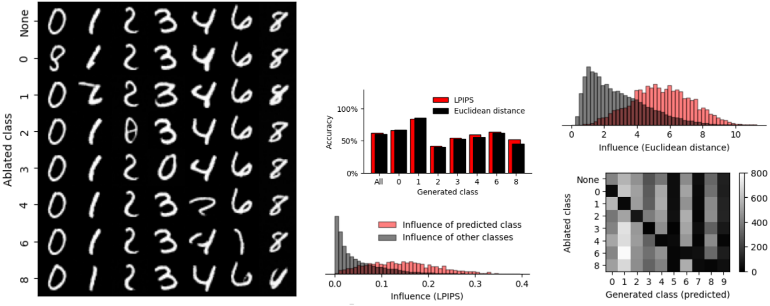


Figure 5: Training data attribution

# Experiments


Figure 6: Top10 most influential training samples

# Thank You

Q & A