A Backgrounds

A.1 Diffusion Models

A diffusion model is a generative model that gradually adds noise to an input signal $\mathbf{x} = \mathbf{x}_0$ until it is fully destroyed to random noise $\mathbf{x}_T$ and then denoise multiple steps to generate an output signal $\tilde{\mathbf{x}}_0$ with a probability distribution similar to the input. A diffusion process is defined as Gaussian process with Markov chain:

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\mathbf{z}_t, t = 1, ..., T \tag{7}$$

where $\beta_1, ..., \beta_T$ is a fixed variance scheduler which means the quantity of noise for each step t and $\mathbf{z}_t \sim \mathcal{N}(0, I)$. It can be rewritten as,

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t I) \tag{8}$$

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_{t-1}, (1-\bar{\alpha}_t)I) \tag{9}$$

where $\alpha_t \coloneqq 1 - \beta_t$ and $\bar{\alpha}_t \coloneqq \prod_{s=1}^{t} \alpha_s$.

To recover the input signal, we need to learn reverse process, which requires estimating the noise prediction function $\epsilon_\theta(\mathbf{x}_t); t = 1, ..., T$. The parameter θ is optimized by minimizing follows:

$$\mathcal{L}(\theta) = \mathbb{E}_{\epsilon,\mathbf{x},t}[||\epsilon_\theta(\mathbf{x}_t) - \epsilon||_2^2] \tag{10}$$

in which $\epsilon \sim \mathcal{N}(0, I)$. This objective performs denoising score matching over multiple noise scales by t. Leveraging predicted noise ϵ_θ, we can sample $\mathbf{x}_{t-1} \sim p(\mathbf{x}_{t-1}|\mathbf{x}_t)$. The most widely adopted sampling method is Denoising Diffusion Probabilistic Models (DDPM) [47] sampler:

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha}_t}(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}})\epsilon_\theta(\mathbf{x}_t) + \sigma_t\mathbf{z} \tag{11}$$

A.2 Classifier-Free Diffusion Guidance

Classifier-free Diffusion Guidance (CFG) [33] is a simple yet effective conditional diffusion model, avoiding require for a separate classifier. They obtain a combination of a conditional model parameterized with $\epsilon_\theta(\mathbf{x}_t, \mathbf{c})$ and an unconditional model parameterized with $\epsilon_\theta(\mathbf{x}_t) = \epsilon_\theta(\mathbf{x}_t, \mathbf{c} = \varnothing)$, which gives null token to guidance $\mathbf{c}$ in a single network. During training it randomly drop the condition with unconditional probability p_{uncond}. The training process is described in Algorithm 1.

Algorithm 1 Classifier-Free Diffuison Guidance Training

Require: p_{uncond}: probability of unconditional training
Require: $\mathbf{c}$: conditional guidance signal
  repeat
    $(\mathbf{x}, \mathbf{c}) \sim p(\mathbf{x}, \mathbf{c})$
    $\mathbf{c} \to \emptyset$ with probability p_{uncond}
    $\lambda \sim p(\lambda)$
    $\epsilon \sim \mathcal{N}(0, I)$
    $z_\lambda = \alpha_\lambda \mathbf{x} + \sigma_\lambda \epsilon$
    Take gradient step on $\nabla_\theta \left\| \epsilon_\theta(\mathbf{z}_\lambda, \mathbf{c}) - \epsilon \right\|^2$
  until converged

The noise prediction function in sampling phase is modified by linear combination of the conditional and unconditional noise prediction function as follows.

$$\epsilon_t = (1 + w)\epsilon_\theta(\mathbf{z}_t, \mathbf{c}_r) - w\epsilon_\theta(\mathbf{z}_t) \tag{12}$$

Overall sampling process is described in Algorithm 2.

Algorithm 2 Classifier-Free Diffuison Guidance Sampling

Require: w: guidance weight
Require: $\mathbf{c}$: conditional guidance signal
Require: $\lambda_1, ..., \lambda_T$: increasing log SNR sequence with $\lambda_1 = \lambda_{min}, \lambda_T = \lambda_{max}$
  $\mathbf{z}_1 \sim \mathcal{N}(0, I)$
  for $t = 1, ..., T$ **do**
    $\tilde{\epsilon}_t = (1 + w)\epsilon_\theta(\mathbf{z}_t, \mathbf{c}) - w\epsilon_\theta(\mathbf{z}_t)$
    $\tilde{\mathbf{x}}_t = (\mathbf{z}_t - \sigma_{\lambda_t}\tilde{\epsilon}_t)/\sigma_{\lambda_t}$
    $\mathbf{z}_{t+1} \sim \mathcal{N}(\bar{\mu}_{\lambda_{t+1}|\lambda_t}(\mathbf{z}_t, \tilde{\mathbf{x}}_t), (\sigma^2_{\lambda_{t+1}|\lambda_t})^{1-v}(\sigma^2_{\lambda_t|\lambda_{t+1}})^v)$ if $t > 1$ else $\mathbf{z}_{t+1} = \tilde{\mathbf{x}}_t$
  end for
  return $\mathbf{z}_{T+1}$

B Implementation Details

B.1 Classifier-Free Diffusion Guidance Model Training

We follow the U-Net architecture of DDPM [47] with pseudo-class label guidance embedding. Both time embedding and guidance embedding function consist of 2-fully connected layers with SiLU activation function. We use the network parameters and optimization strategy for training Classifier-free Diffusion Guidance model as follows.

Table 5: Hyperparameters of Classifier-free Diffusion Guidance on CIFAR-10

Hyperparameter	**Value**
Base channels	128
Channel multipliers	[1, 2, 2, 2]
Unconditional probability	$p = 0.1$
Time embedding dimension	512
Guidance embedding dimension	512
Dropout	0.1
Diffusion timesteps	1000
beta range	[0.0001, 0.2]
Diffusion noise schedule	Linear
Learning rate	$2e^{-4}$
Batch size	128
Epochs	2500
Optimizer	Adam
EMA decay	0.9999
Sampler	DDPM [47] sampler
Sampling step	1000

B.2 OOD Detection Network Training

Our detection network utilizes ViT_B16 which is pre-trained on ImageNet 21K as the feature extractor $f(\tilde{\mathbf{x}};\theta)$. and we freeze 3-layers of pretrained ViT_B16 and learn 3-additional heads for fine-tuning. We introduce the architecture of each head.

- Binary head g_{bin}: 2-fully connected layers, 256 hidden dimension
- Multi-class head for OOD classification g_{out}: 2-fully connected layers, 256 hidden dimension
- Multi-class head for ID classification g_{in}: 2-fully connected layers, 256 hidden dimension

Our optimization strategy and the hyperparameters for training OOD detection network is described as follows:

Table 6: Hyperparameters of training OOD detection network

Hyperparameter	**Value**
Learning rate	$4e^{-4}$
Batch size	16
Epochs	15
Optimizer	SGD

C Semantic-Discrepant Outlier Samples

C.1 SD Outliers vs Other Generation Method

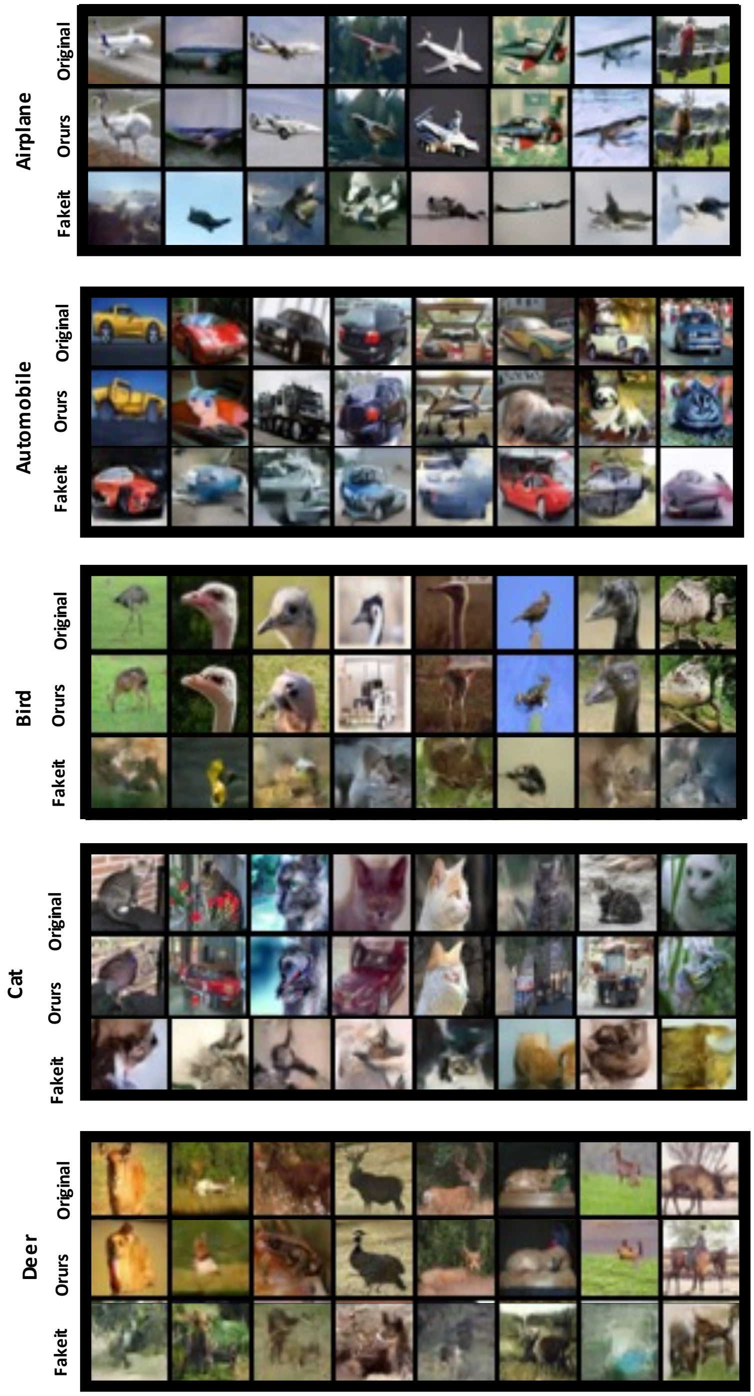

Figure 4: A comparison between the most recently proposed and performance-leading SDE-based method for generating outliers and outliers obtained through our semantic-discrepant sampling.

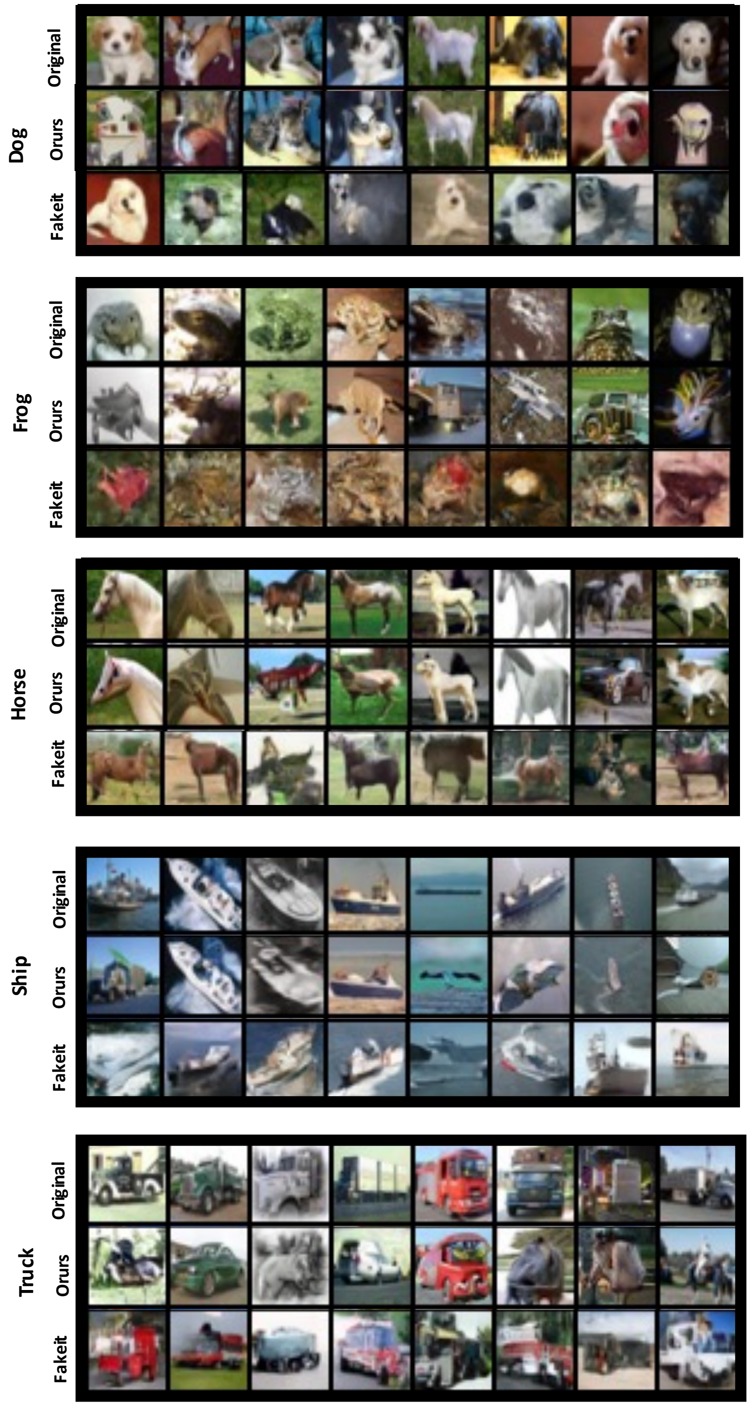

Figure 5: A comparison between the most recently proposed and performance-leading SDE-based method for generating outliers and outliers obtained through our semantic-discrepant sampling.

C.2 Dataset Dependence on Sampling Timesteps

Figure 6: By increasing S, we observed a gradual shift in the sampling condition, where the semantic degradation became increasingly influential. Beyond a certain threshold of S, we found that images similar to the sampling condition could be generated, which could potentially belong to the same ID. Our best detection performance was achieved at S=100.