

February 15, 2021



# Score-based Generative Models using Neural SDEs

**Vikram Voleti**

PhD student, Mila, University of Montreal

Supervisor: Prof. Christopher Pal

# 1. Generative Modeling by Estimating Gradients of the Data Distribution

(<https://arxiv.org/abs/1907.05600>)

## 2. Score-based generative modeling through SDEs

(<https://arxiv.org/abs/2011.13456>)

## 1. Score matching:

$$\frac{1}{2} \mathbb{E}_{p_{\text{data}}} [\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) \|_2^2]$$

Score  
network

## 2. Langevin dynamics:

$$\tilde{\mathbf{x}}_t = \tilde{\mathbf{x}}_{t-1} + \frac{\epsilon}{2} \nabla_{\mathbf{x}} \log p(\tilde{\mathbf{x}}_{t-1}) + \sqrt{\epsilon} \mathbf{z}_t$$

<https://arxiv.org/abs/1907.05600>

## 1. Score matching:

$$\frac{1}{2} \mathbb{E}_{p_{\text{data}}} [\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) \|_2^2]$$

$$\frac{1}{2} \mathbb{E}_{q_{\sigma}(\tilde{\mathbf{x}}|\mathbf{x}) p_{\text{data}}(\mathbf{x})} [\| \mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log q_{\sigma}(\tilde{\mathbf{x}} | \mathbf{x}) \|_2^2]$$

**Denoising score matching**

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})} \left[ \text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})) + \frac{1}{2} \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2 \right]$$

$$\mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_{\text{data}}} \left[ \mathbf{v}^{\top} \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \mathbf{v} + \frac{1}{2} \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2 \right]$$

**Sliced score matching**

<https://arxiv.org/abs/1907.05600>

## Challenges

### Manifold hypothesis

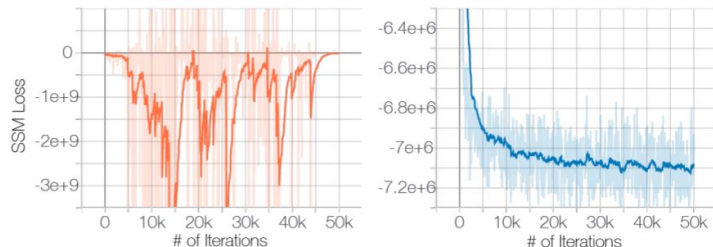


Figure 1: **Left:** Sliced score matching (SSM) loss w.r.t. iterations. No noise is added to data. **Right:** Same but data are perturbed with  $\mathcal{N}(0, 0.0001)$ .

### Inaccurate score estimation

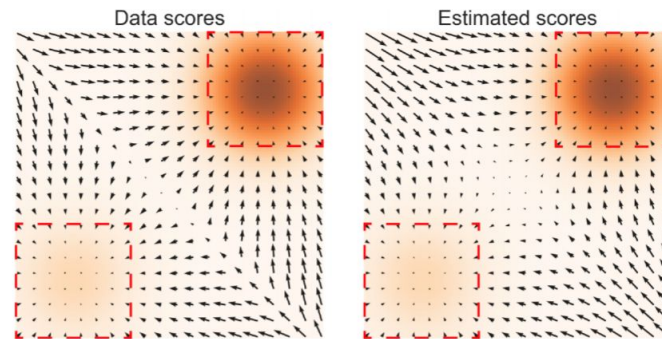


Figure 2: **Left:**  $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$ ; **Right:**  $\mathbf{s}_{\theta}(\mathbf{x})$ . The data density  $p_{\text{data}}(\mathbf{x})$  is encoded using an orange colormap: darker color implies higher density. Red rectangles highlight regions where  $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) \approx \mathbf{s}_{\theta}(\mathbf{x})$ .

<https://arxiv.org/abs/1907.05600>

## Noise Conditional Score Matching (NCSM) (Training)

$$\ell(\boldsymbol{\theta}; \sigma) \triangleq \frac{1}{2} \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{\tilde{\mathbf{x}} \sim \mathcal{N}(\mathbf{x}, \sigma^2 I)} \left[ \left\| \boxed{\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}, \sigma)} + \frac{\tilde{\mathbf{x}} - \mathbf{x}}{\sigma^2} \right\|_2^2 \right]$$

(from Denoising Score Matching)

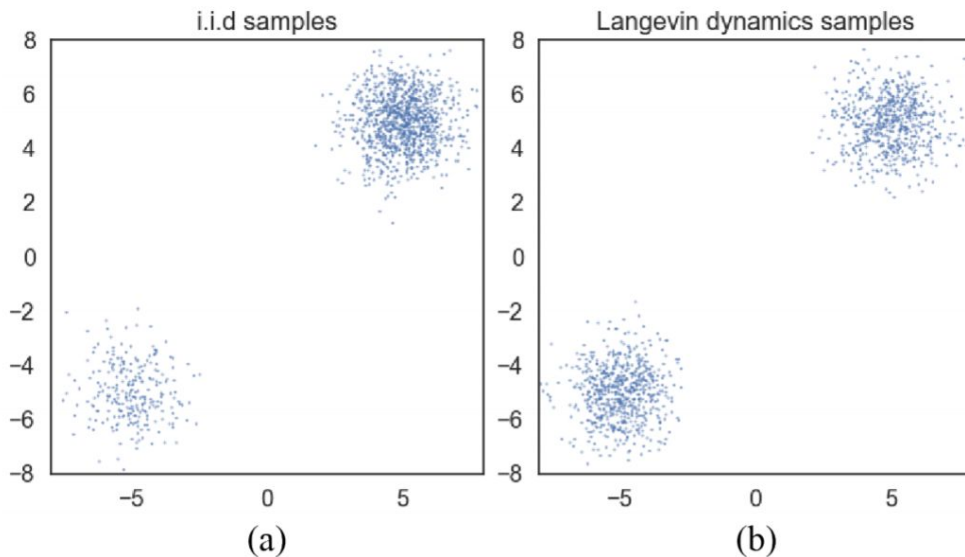
$$\mathcal{L}(\boldsymbol{\theta}; \{\sigma_i\}_{i=1}^L) \triangleq \frac{1}{L} \sum_{i=1}^L \lambda(\sigma_i) \ell(\boldsymbol{\theta}; \sigma_i)$$

$$\mathbf{s}_{\boldsymbol{\theta}^*}(\mathbf{x}, \sigma_i) = \nabla_{\mathbf{x}} \log q_{\sigma_i}(\mathbf{x}) \text{ a.s. for all } i \in \{1, 2, \dots, L\}$$

<https://arxiv.org/abs/1907.05600>

## Challenges

### Slow mixing of Langevin Dynamics



<https://arxiv.org/abs/1907.05600>

## Annealed Langevin Dynamics (sampling)

---

**Algorithm 1** Annealed Langevin dynamics.

---

**Require:**  $\{\sigma_i\}_{i=1}^L, \epsilon, T$ .

Draw a random prior sample.  
For each (progressive lesser) noise level:

Walk through  
Langevin Dynamics  
for T steps

```
1: Initialize  $\tilde{\mathbf{x}}_0$ 
2: for  $i \leftarrow 1$  to  $L$  do
3:    $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2 / \sigma_L^2$   $\triangleright \alpha_i$  is the step size.
4:   for  $t \leftarrow 1$  to  $T$  do
5:     Draw  $\mathbf{z}_t \sim \mathcal{N}(0, I)$ 
6:      $\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \frac{\alpha_i}{2} \mathbf{s}_\theta(\tilde{\mathbf{x}}_{t-1}, \sigma_i) + \sqrt{\alpha_i} \mathbf{z}_t$ 
7:   end for
8:    $\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$ 
9: end for
return  $\tilde{\mathbf{x}}_T$ 
```

---



## Annealed Langevin Dynamics (sampling)

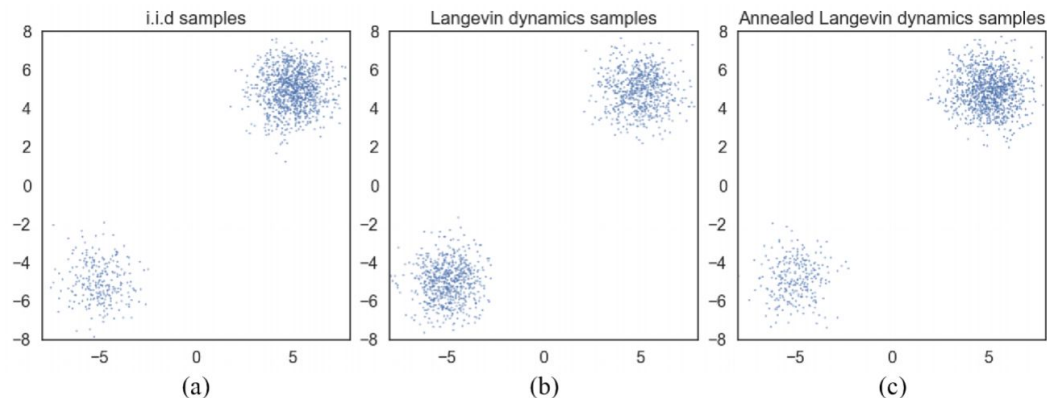


Figure 3: Samples from a mixture of Gaussian with different methods. (a) Exact sampling. (b) Sampling using Langevin dynamics with the exact scores. (c) Sampling using annealed Langevin dynamics with the exact scores. Clearly Langevin dynamics estimate the relative weights between the two modes incorrectly, while annealed Langevin dynamics recover the relative weights faithfully.

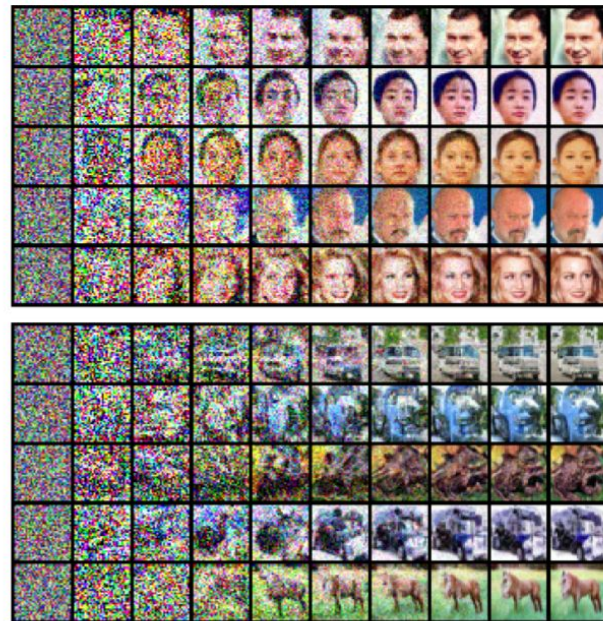


Figure 4: Intermediate samples of annealed Langevin dynamics.

<https://arxiv.org/abs/1907.05600>

1. Generative Modeling by Estimating Gradients of the Data Distribution

(<https://arxiv.org/abs/1907.05600>)

2. **Score-based generative modeling through SDEs** (<https://arxiv.org/abs/2011.13456>)

## Score function (Training)

$$\ell(\boldsymbol{\theta}; \sigma) \triangleq \frac{1}{2} \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{\tilde{\mathbf{x}} \sim \mathcal{N}(\mathbf{x}, \sigma^2 I)} \left[ \left\| \mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}, \sigma) + \frac{\tilde{\mathbf{x}} - \mathbf{x}}{\sigma^2} \right\|_2^2 \right]$$
$$\mathcal{L}(\boldsymbol{\theta}; \{\sigma_i\}_{i=1}^L) \triangleq \frac{1}{L} \sum_{i=1}^L \lambda(\sigma_i) \ell(\boldsymbol{\theta}; \sigma_i)$$

=

$$\boldsymbol{\theta}^* = \arg \min_{\boldsymbol{\theta}} \sum_{i=1}^N \sigma_i^2 \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{p_{\sigma_i}(\tilde{\mathbf{x}}|\mathbf{x})} \left[ \left\| \mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}, \sigma_i) - \nabla_{\tilde{\mathbf{x}}} \log p_{\sigma_i}(\tilde{\mathbf{x}} | \mathbf{x}) \right\|_2^2 \right]$$



continuous

$$\boldsymbol{\theta}^* = \arg \min_{\boldsymbol{\theta}} \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{\mathbf{x}(0)} \mathbb{E}_{\mathbf{x}(t)|\mathbf{x}(0)} \left[ \left\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}(t), t) - \nabla_{\mathbf{x}(t)} \log p_{0t}(\mathbf{x}(t) | \mathbf{x}(0)) \right\|_2^2 \right] \right\}$$

<https://arxiv.org/abs/2011.13456>

## Annealed Langevin Dynamics (sampling)

Reverse process:  $\mathbf{x}_i^m = \mathbf{x}_i^{m-1} + \epsilon_i \mathbf{s}_{\theta^*}(\mathbf{x}_i^{m-1}, \sigma_i) + \sqrt{2\epsilon_i} \mathbf{z}_i^m, \quad m = 1, 2, \dots, M$

Forward (noising) process:

$$\mathbf{x}_i = \mathbf{x}_{i-1} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2} \mathbf{z}_{i-1}, \quad i = 1, \dots, N$$



$$d\mathbf{x} = \sqrt{\frac{d[\sigma^2(t)]}{dt}} d\mathbf{w}$$

<https://arxiv.org/abs/2011.13456>

## Denoising Diffusion Probabilistic Models (DDPM)

<https://arxiv.org/abs/2006.11239>

Forward (noising) process:

$$\mathbf{x}_i = \sqrt{1 - \beta_i} \mathbf{x}_{i-1} + \sqrt{\beta_i} \mathbf{z}_{i-1}, \quad i = 1, \dots, N$$

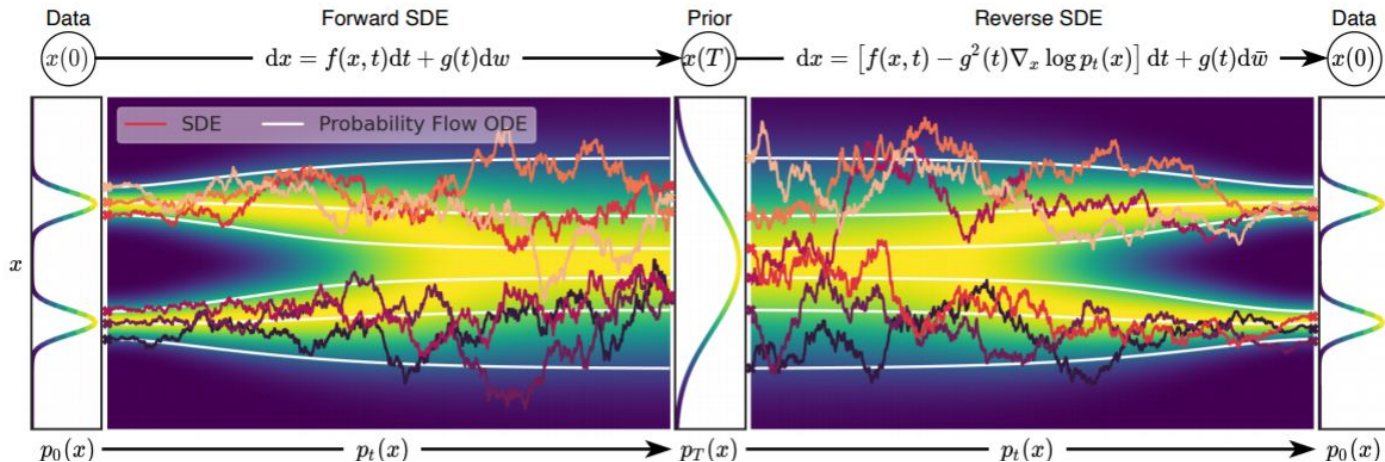


$$d\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x} dt + \sqrt{\beta(t)} d\mathbf{w}$$

<https://arxiv.org/abs/2011.13456>

**Forward (noising) process:**  $dx = f(x, t)dt + g(t)dw$

**Reverse process:**  $dx = [f(x, t) - g(t)^2 \nabla_x \log p_t(x)]dt + g(t)d\bar{w}$



**Figure 2: Overview of score-based generative modeling through SDEs.** We can map data to a noise distribution (the prior) with an SDE (Section 3.1), and reverse this SDE for generative modeling (Section 3.2). We can also reverse the associated probability flow ODE (Section 4.3), which yields a deterministic process that samples from the same distribution as the SDE. Both the reverse-time SDE and probability flow ODE can be obtained by estimating the score  $\nabla_x \log p_t(x)$  (Section 3.3).

<https://arxiv.org/abs/2011.13456>



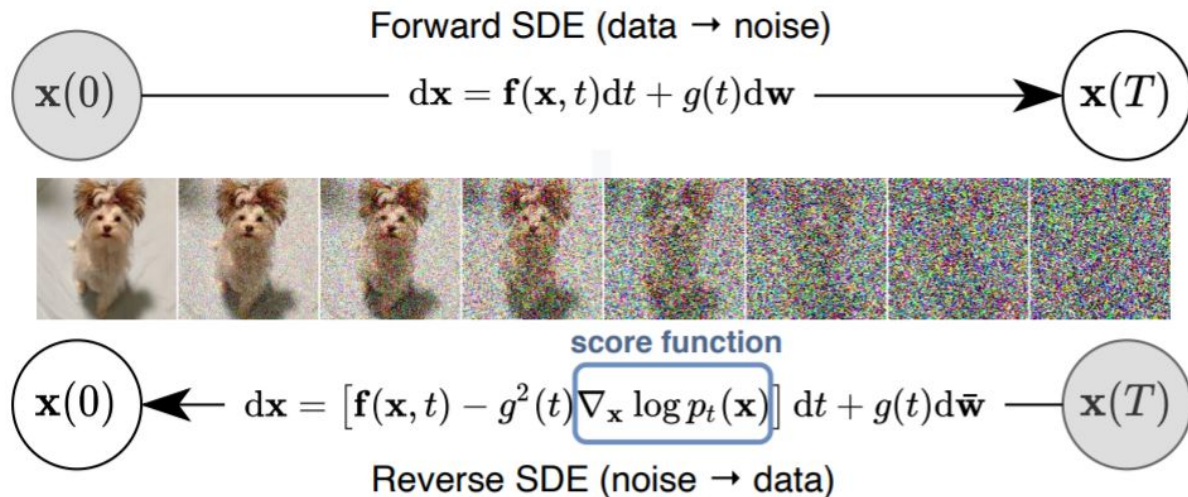


Figure 1: **Solving a reverse-time SDE yields a score-based generative model.** Transforming data to a simple noise distribution can be accomplished with a continuous-time SDE. This SDE can be reversed if we know the score of the distribution at each intermediate time step,  $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ .

## Uniquely identifiable encoding

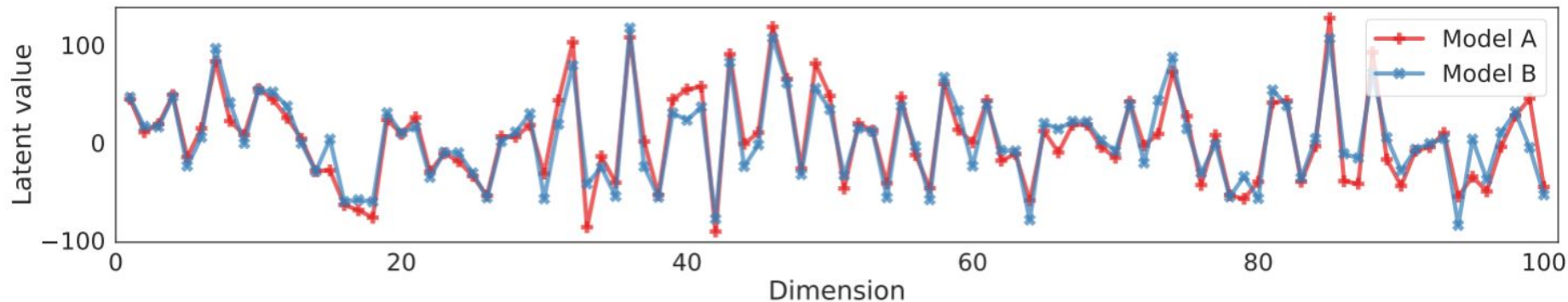


Figure 7: Comparing the first 100 dimensions of the latent code obtained for a random CIFAR-10 image. “Model A” and “Model B” are separately trained with different architectures.



## Conditional generation

$$d\mathbf{x} = \{ \mathbf{f}(\mathbf{x}, t) - g(t)^2 [\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) + \nabla_{\mathbf{x}} \log p_t(\mathbf{y} \mid \mathbf{x})] \} dt + g(t) d\bar{\mathbf{w}}$$

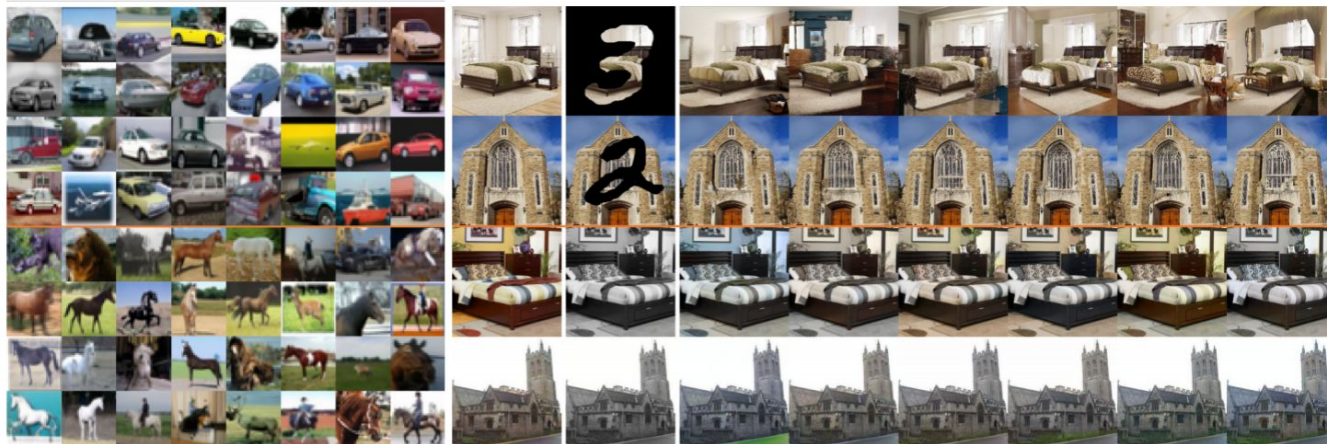


Figure 4: *Left*: Class-conditional samples on  $32 \times 32$  CIFAR-10. Top four rows are automobiles and bottom four rows are horses. *Right*: Inpainting (top two rows) and colorization (bottom two rows) results on  $256 \times 256$  LSUN. First column is the original image, second column is the masked/gray-scale image, remaining columns are sampled image completions or colorizations.

<https://arxiv.org/abs/2011.13456>

**Thank you!**