Meeting 4

08.01.2025

- **Control Over/Conditional Diffusion Model**
  - Recall to vanilla DDPM
  - Score-based approach to DDPM

- **Control Over/Conditional Diffusion Model**
  - Fine-tuning (skipping this)
  - Conditioning (during training)
  - Guidance (During Sampling)
    - Classifier-Guidance
    - Classifier-Free Guidance
  - Cascading DDPM

- **Towards Modern Models**
  - Dreambooth

Recall to training and sampling procedure of vanilla DDPM (= unconditional)

(Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models.)

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

There are architectural improvements to this model, which I will not explain today.
Assume for later slides:

- variance is now a learned parameter
- noise schedule is more sophisticated

Score-based approach to DDPM (still unconditional)

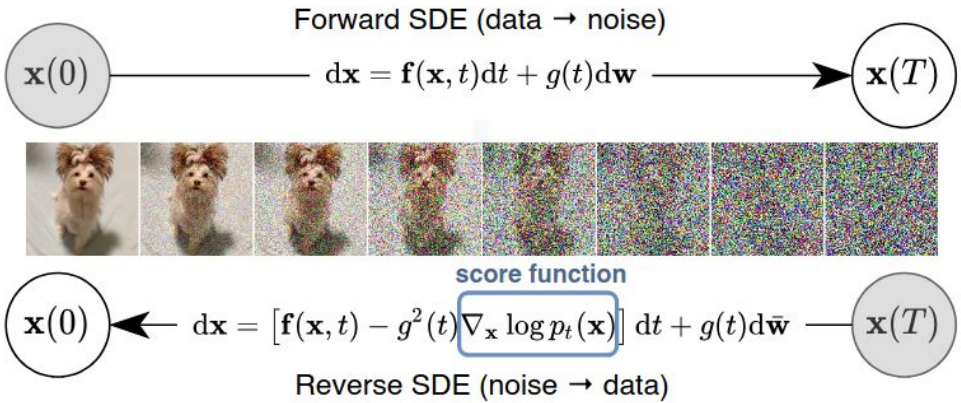
(Song, Yang, et al. "Score-based generative modeling through stochastic differential equations." 2021)

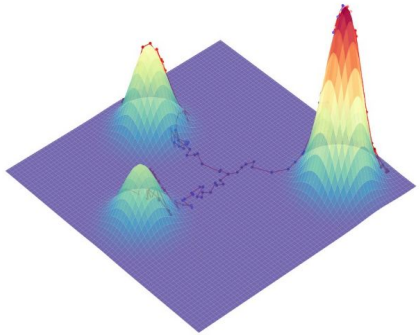
Model could predict these, the first two is prominent in literature:

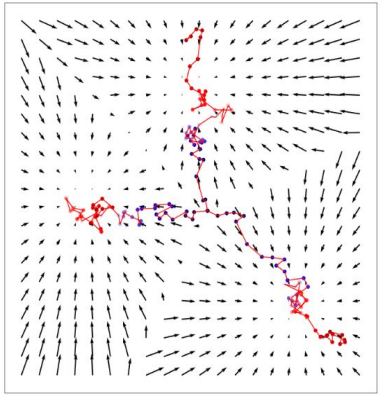
1. Noise level
2. Score function
3. Original Image / Mean of the local distribution that the original image is sampled from

➔ Score-based approach (not-studied, only intuition)

The intuition here could be helpful for understanding conditional models.







Control Over/Conditional Diffusion Model

- Fine-tuning (skipping this)
- Conditioning (during training)
- Guidance (During Sampling)
  - Classifier-Guidance
  - Classifier-Free Guidance
- Cascading DDPM

Conditional DDPM
Class Conditioning during training

Not much to talk here. Only a class embedding is learned during the training.
(Below algorithm is from another paper, notations are different)
There is also a code example (I understood it better when looked at the code)

Algorithm 1 Training a denoising model f_θ

1: **repeat**
2: $(\boldsymbol{x}, \boldsymbol{y}_0) \sim p(\boldsymbol{x}, \boldsymbol{y})$ *x is the class label*
3: $\gamma \sim p(\gamma)$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take a gradient descent step on
$$\nabla_\theta \left\| f_\theta(\boldsymbol{x}, \sqrt{\gamma}\boldsymbol{y}_0 + \sqrt{1-\gamma}\boldsymbol{\epsilon}, \gamma) - \boldsymbol{\epsilon} \right\|_p^p$$
6: **until** converged

Algorithm 2 Inference in T iterative refinement steps

1: $\boldsymbol{y}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\boldsymbol{z} = \mathbf{0}$
4: $\boldsymbol{y}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\boldsymbol{y}_t - \frac{1-\alpha_t}{\sqrt{1-\gamma_t}} f_\theta(\boldsymbol{x}, \boldsymbol{y}_t, \gamma_t) \right) + \sqrt{1-\alpha_t}\boldsymbol{z}$
5: **end for**
6: **return** $\boldsymbol{y}_0$

```python
class ClassConditionedUnet(nn.Module):
    def __init__(self, num_classes=10, class_emb_size=4):
        super().__init__()

        # The embedding layer will map the class label to a vector of size class_emb_size
        self.class_emb = nn.Embedding(num_classes, class_emb_size)

        # Self.model is an unconditional UNet with extra input channels to accept the conditioning inform
        self.model = UNet2DModel(
            sample_size=28,  # the target image resolution
            in_channels=1 + class_emb_size,  # Additional input channels for class cond.
            out_channels=1,  # the number of output channels
            layers_per_block=2,  # how many ResNet layers to use per UNet block
            block_out_channels=(32, 64, 64),
            down_block_types=(
                "DownBlock2D",  # a regular ResNet downsampling block
                "AttnDownBlock2D",  # a ResNet downsampling block with spatial self-attention
                "AttnDownBlock2D",
            ),
            up_block_types=(
                "AttnUpBlock2D",
                "AttnUpBlock2D",  # a ResNet upsampling block with spatial self-attention
                "UpBlock2D",  # a regular ResNet upsampling block
            ),
        )
```

➔ Classifier Guidance

(Dhariwal, P., & Nichol, A. Diffusion models beat gans on image synthesis. ,2021)

Main idea: A classifier can be trained on noisy images and the gradient of the probability could be used in the sampling (reverse) process. Here is the derivation from the paper:

$$p_{\theta,\phi}(x_t|x_{t+1}, y) = Z p_\theta(x_t|x_{t+1}) p_\phi(y|x_t)$$

θ=Diffusion Model, Φ=Classifier

$$p_\theta(x_t|x_{t+1}) = \mathcal{N}(\mu, \Sigma) \quad \textit{(Vanilla Reverse Process)}$$

$$\log p_\theta(x_t|x_{t+1}) = -\frac{1}{2}(x_t - \mu)^T \Sigma^{-1}(x_t - \mu) + C$$

Taylor approximation for the classifier (some assumptions made such that this is reasonable)

$$\begin{aligned}\log p_\phi(y|x_t) &\approx \log p_\phi(y|x_t)|_{x_t=\mu} + (x_t - \mu)\nabla_{x_t} \log p_\phi(y|x_t)|_{x_t=\mu} \\ &= (x_t - \mu)g + C_1 \quad \textit{(Classifier)}\end{aligned}$$

Here, $g = \nabla_{x_t} \log p_\phi(y|x_t)|_{x_t=\mu}$, and C_1 is a constant. This gives

Guided Reverse Process:

$$\begin{aligned}\log(p_\theta(x_t|x_{t+1})p_\phi(y|x_t)) &\approx -\frac{1}{2}(x_t - \mu)^T \Sigma^{-1}(x_t - \mu) + (x_t - \mu)g + C_2 \\ &= -\frac{1}{2}(x_t - \mu - \Sigma g)^T \Sigma^{-1}(x_t - \mu - \Sigma g) + \frac{1}{2}g^T \Sigma g + C_2 \\ &= -\frac{1}{2}(x_t - \mu - \Sigma g)^T \Sigma^{-1}(x_t - \mu - \Sigma g) + C_3 \\ &= \log p(z) + C_4, z \sim \mathcal{N}(\mu + \Sigma g, \Sigma)\end{aligned}$$

Classifier Guidance

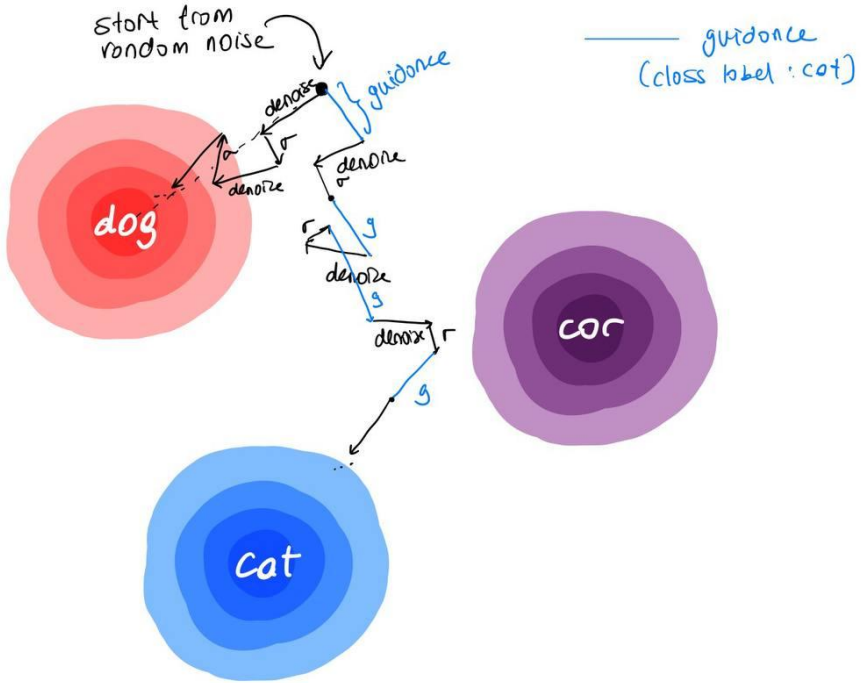
(Dhariwal, P., & Nichol, A. Diffusion models beat gans on image synthesis. ,2021)

Gradient scale **s** added to control the guidance effect

Algorithm 1 Classifier guided diffusion sampling, given a diffusion model $(\mu_\theta(x_t), \Sigma_\theta(x_t))$, classifier $p_\phi(y|x_t)$, and gradient scale s.

Input: class label y, gradient scale s
$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$
for all t from T to 1 **do**
  $\mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$
  $x_{t-1} \leftarrow$ sample from $\mathcal{N}(\mu + s\Sigma \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$
end for
return x_0

Guidance Visualized

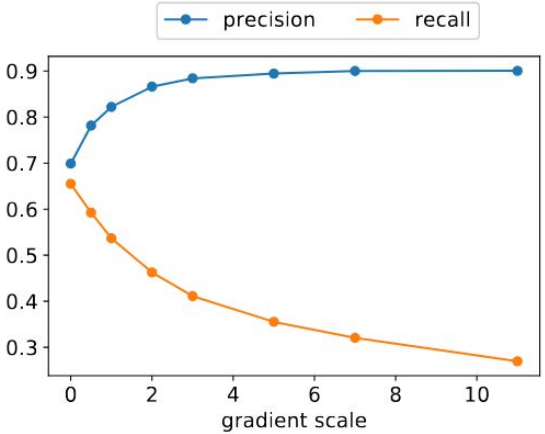


Classifier Guidance

(Dhariwal, P., & Nichol, A. Diffusion models beat gans on image synthesis. ,2021)

Guidance and conditional generation together, also the effect of guidance.
Moreover, what is lost when guidance is used and comparison to GAN

Conditional	Guidance	Scale	FID	sFID	IS	Precision	Recall
✗	✗		26.21	**6.35**	39.70	0.61	0.63
✗	✓	1.0	33.03	6.99	32.92	0.56	**0.65**
✗	✓	10.0	**12.00**	10.40	**95.41**	**0.76**	0.44
✓	✗		10.94	6.02	100.98	0.69	**0.63**
✓	✓	1.0	**4.59**	**5.25**	186.70	0.82	0.52
✓	✓	10.0	9.11	10.93	**283.92**	**0.88**	0.32



Recall: Measure of **diversity** (diffusion models are already diverse in generating samples, whereas GAN falls short)
Precision: Measure of **fidelity**. With the help guidance, the fidelity of GAN achieved.

What the authors used as a classifier?: The model was downsampling trunk of UNet model, training data was noise added version of ImageNet.
More advanced classifiers could be used = **CLIP**

➔ Classifier-Free Guidance

(Ho, Jonathan, and Tim Salimans. "Classifier-free diffusion guidance." ,2022)

Main idea: Instead of training a different classifier, use the gradients of the conditional diffusion model. The claim is not creating the sota model, achieve the control of guidance without a classifier.
We will have *one model*, but *two modes*: one conditional, one unconditional.
Here notations again scramble, $\mathbf{z}$ is the latent (noised images) and $\mathbf{c}$ is the condition.

$$\tilde{\boldsymbol{\epsilon}}_\theta(\mathbf{z}_\lambda, \mathbf{c}) = (1+w)\boldsymbol{\epsilon}_\theta(\mathbf{z}_\lambda, \mathbf{c}) - w\boldsymbol{\epsilon}_\theta(\mathbf{z}_\lambda)$$

Algorithm 1 Joint training a diffusion model with classifi

Require: p_{uncond}: probability of unconditional training
1: **repeat**
2: $(\mathbf{x}, \mathbf{c}) \sim p(\mathbf{x}, \mathbf{c})$
3: $\mathbf{c} \leftarrow \varnothing$ with probability p_{uncond}
4: $\lambda \sim p(\lambda)$
5: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
6: $\mathbf{z}_\lambda = \alpha_\lambda \mathbf{x} + \sigma_\lambda \boldsymbol{\epsilon}$
7: Take gradient step on $\nabla_\theta \left\| \boldsymbol{\epsilon}_\theta(\mathbf{z}_\lambda, \mathbf{c}) - \boldsymbol{\epsilon} \right\|^2$
8: **until** converged

Algorithm 2 Conditional sampling with classifier-free guidance

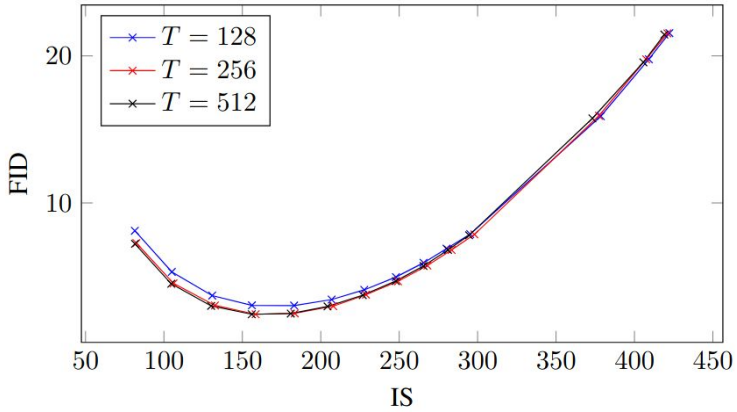
Require: w: guidance strength
Require: $\mathbf{c}$: conditioning information for conditional sampling
Require: $\lambda_1, \ldots, \lambda_T$: increasing log SNR sequence with $\lambda_1 = \lambda_{\min}, \lambda_T = \lambda_{\max}$
1: $\mathbf{z}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = 1, \ldots, T$ **do**
▷ Form the classifier-free guided score at log SNR λ_t
3: $\tilde{\boldsymbol{\epsilon}}_t = (1+w)\boldsymbol{\epsilon}_\theta(\mathbf{z}_t, \mathbf{c}) - w\boldsymbol{\epsilon}_\theta(\mathbf{z}_t)$
▷ Sampling step (could be replaced by another sampler, e.g. DDIM)
4: $\tilde{\mathbf{x}}_t = (\mathbf{z}_t - \sigma_{\lambda_t}\tilde{\boldsymbol{\epsilon}}_t)/\alpha_{\lambda_t}$
5: $\mathbf{z}_{t+1} \sim \mathcal{N}(\tilde{\boldsymbol{\mu}}_{\lambda_{t+1}|\lambda_t}(\mathbf{z}_t, \tilde{\mathbf{x}}_t), (\tilde{\sigma}^2_{\lambda_{t+1}|\lambda_t})^{1-v}(\sigma^2_{\lambda_t|\lambda_{t+1}})^v)$ if $t < T$ else $\mathbf{z}_{t+1} = \tilde{\mathbf{x}}_t$
6: **end for**
7: **return** $\mathbf{z}_{T+1}$

Classifier-Free Guidance

(Ho, Jonathan, and Tim Salimans. "Classifier-free diffusion guidance." ,2022)

FID (Fréchet Inception Distance, lower better) and **IS (Inspection Score, higher better)** are two different evaluation metrics for generative models.
IS only access the quality of generated samples, FID also penalize low diversity.
The curve on the right represent trade of between FID and IS, just like classifier guidance.

Model	FID (↓)	IS (↑)
BigGAN-deep, max IS (Brock et al., 2019)	25	253
BigGAN-deep (Brock et al., 2019)	5.7	124.5
CDM (Ho et al., 2021)	3.52	128.8
LOGAN (Wu et al., 2019)	3.36	148.2
ADM-G (Dhariwal & Nichol, 2021)	2.97	-
Ours	$T = 128/256/1024$	
$w = 0.0$	8.11 / 7.27 / 7.22	81.46 / 82.45 / 81.54
$w = 0.1$	5.31 / 4.53 / 4.5	105.01 / 106.12 / 104.67
$w = 0.2$	3.7 / 3.03 / 3	130.79 / 132.54 / 130.09
$w = 0.3$	3.04 / **2.43** / **2.43**	156.09 / 158.47 / 156
$w = 0.4$	3.02 / 2.49 / 2.48	183.01 / 183.41 / 180.88
$w = 0.5$	3.43 / 2.98 / 2.96	206.94 / 207.98 / 204.31
$w = 0.6$	4.09 / 3.76 / 3.73	227.72 / 228.83 / 226.76
$w = 0.7$	4.96 / 4.67 / 4.69	247.92 / 249.25 / 247.89
$w = 0.8$	5.93 / 5.74 / 5.71	265.54 / 267.99 / 265.52
$w = 0.9$	6.89 / 6.8 / 6.81	280.19 / 283.41 / 281.14
$w = 1.0$	7.88 / 7.86 / 7.8	295.29 / 297.98 / 294.56
$w = 2.0$	15.9 / 15.93 / 15.75	378.56 / 377.37 / 373.18
$w = 3.0$	19.77 / 19.77 / 19.56	409.16 / 407.44 / 405.68
$w = 4.0$	21.55 / 21.53 / 21.45	**422.29** / 421.03 / 419.06

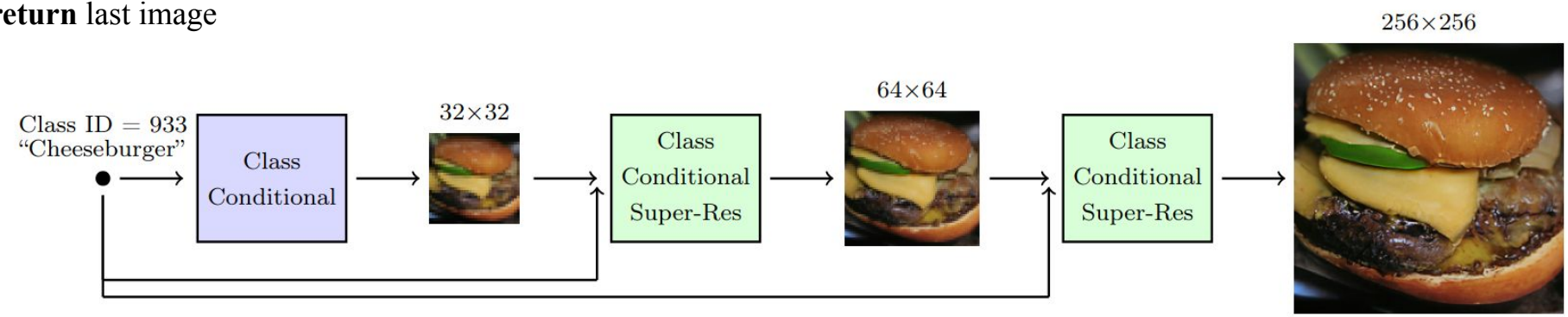


➔ Cascaded Diffusion Models

(Ho, Jonathan, et al. "Cascaded diffusion models for high fidelity image generation.,2022;
previous work: Saharia, Chitwan, et al. "Image super-resolution via iterative refinement.",2022)

Main idea: Using one than one diffusion models in a cascading manner. In the cascading connections, there will be an conditioning on images.

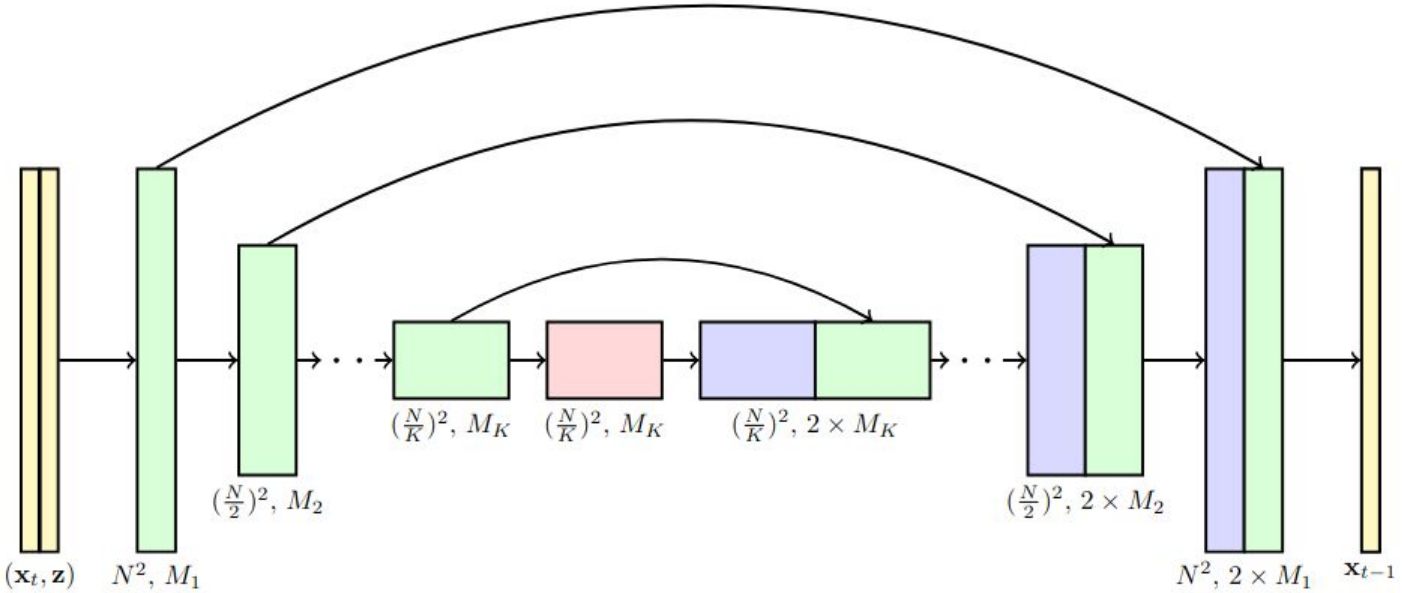
start:
sample an image from an conditional diffusion model from pure noise
repat:
Upscale the generated image and give it to the super-resolution model together with class label
SR model takes the upscaled image concatenated with noise channel-wise
SR model produce an high resolution image
until desired resolution
return last image



➔ Cascaded Diffusion Models

(Ho, Jonathan, et al. "Cascaded diffusion models for high fidelity image generation.,2022;
previous work: Saharia, Chitwan, et al. "Image super-resolution via iterative refinement.",2022)

The UNet model used in the super-resolution blocks:
Note: with this model, they achieve the best FID score on 64^2,128^2,256^2 generation tasks.



Towards Modern Models

(only mention on the main idea)

- CLIP as a classifier guidance : GLIDE, DiffusionCLIP, StyleCLIP (GAN)
- Dreambooth - fine-tuning technique

➔ CLIP

(Radford, Alec, et al. "Learning transferable visual models from natural language supervision." 2021)

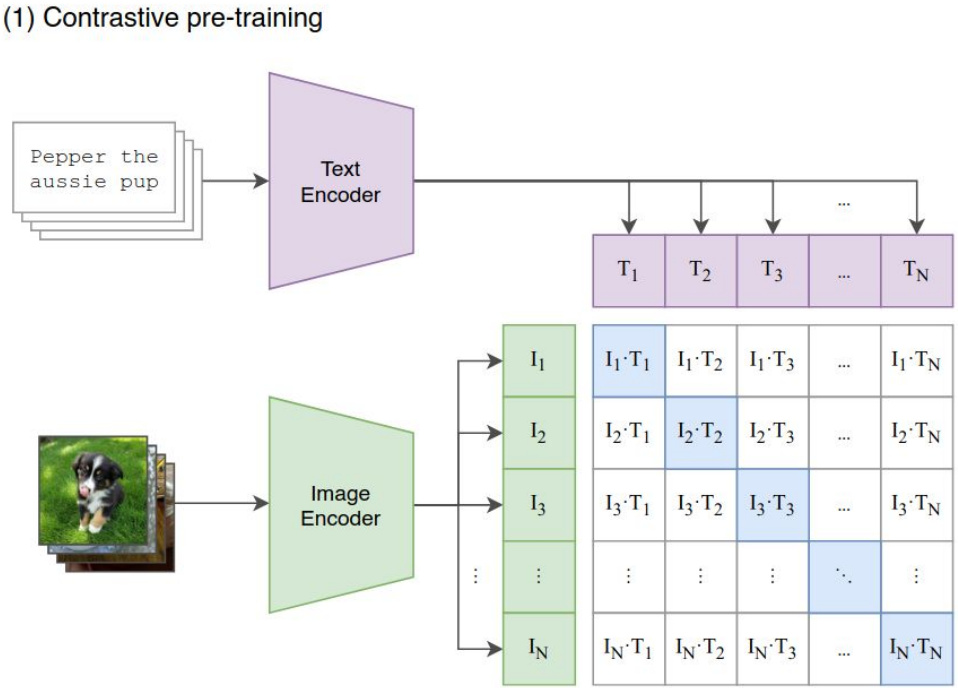
Main Idea: Create an joint space both for the image and text (caption) embeddings such that a given image and the corresponded caption goes to the same embedding.

Text decoder is an transformer.
Image encoder could be ResNet or ViT (vision transformer).

*order error

Prompt	Benzerlik Olasılığı
Mountain	0.0495
Snow	0.0408
Sky	0.0029
Scenery	0.0065
Car	0.0002
Tree	0.0001
Dog	0.0001
Cat	0.0000

GLIDE is the first example using CLIP guidance for image generation. They train CLIP additionally with noise images.
(Nichol, Alex, et al. "Glide: Towards photorealistic image generation and editing with text-guided diffusion models." 2021.)



There are many many models and improvements after the text-encoder guidance:

- DALL-E 2 (OpenAI)
- Imagen (Google)
- Stable Diffusion

All of them utilizes text-encoders like CLIP, BERT and T5. All of the ongoing studies converged to VLMs.

➔ DreamBooth

(Ruiz, Nataniel, et al. "Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation." , 2023.)

Main idea: Train a CLI

- Find a rare token:
  - That is very rarely used in a language model such that this token has weak prior. Let it be [token] = [V]
- Fine-tune with label “a [V] [class noun]”:
  - [class noun] helps the model to use the prior on the class.
- With a loss to avoid (mode collapse / overfit / language drift).
  - They call this loss class-specific prior preservation loss. It is simple yet very effective.

DreamBooth

(Ruiz, Nataniel, et al. "Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation." , 2023.)

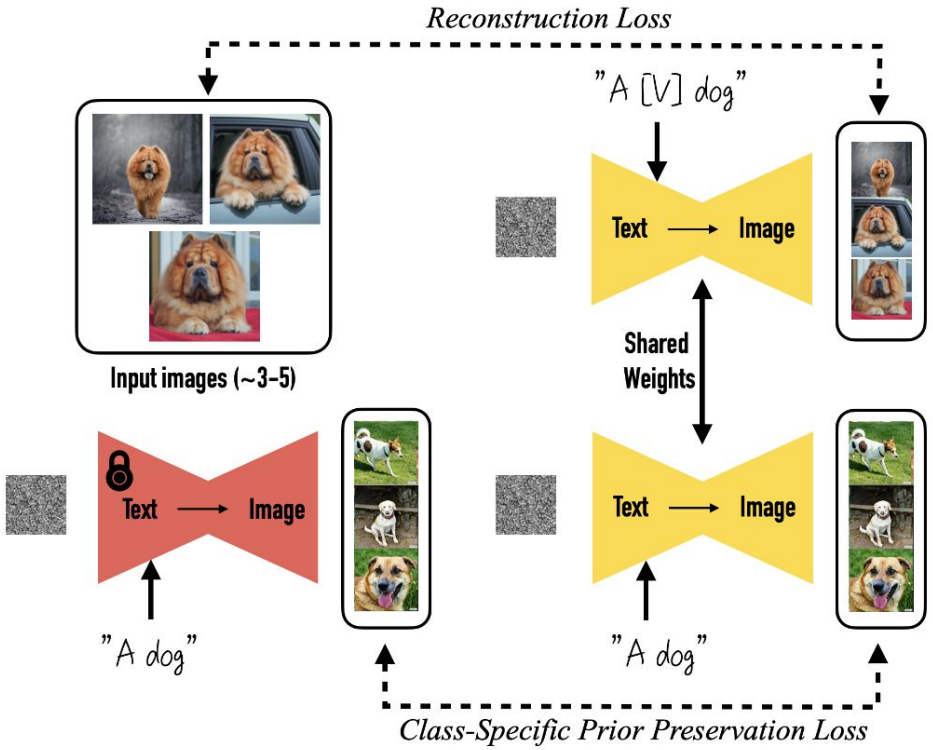
Reconstruction loss +
class-specific prior preservation loss:

$$\mathbb{E}_{\mathbf{x},\mathbf{c},\boldsymbol{\epsilon},\boldsymbol{\epsilon}',t}[w_t \|\hat{\mathbf{x}}_\theta(\alpha_t \mathbf{x} + \sigma_t \boldsymbol{\epsilon}, \mathbf{c}) - \mathbf{x}\|_2^2 + \lambda w_{t'} \|\hat{\mathbf{x}}_\theta(\alpha_{t'} \mathbf{x}_{\text{pr}} + \sigma_{t'} \boldsymbol{\epsilon}', \mathbf{c}_{\text{pr}}) - \mathbf{x}_{\text{pr}}\|_2^2]$$

ω_t is the error schedule (weighted error calculation for different time stamps, not related to this work)
λ is the scale of prior reservation.

Let the pre-trained model generate images of “a [class noun]”
Fine-tune the same model simultaneously. Generate with:
“a [V] [class noun]” and use reconstruction loss
“a [class noun] and use prior preservation loss

The pre-trained and fine-tune models generations could not compared one-to-one. The average cosine similarities of their embedding (ViT) is used for comparison.



Next:

- Architectural improvements
- Image to image editing (mask a patch with noise and re-sample)
- More on VLMs
- Add-it
- Synthetic data generation using diffusion models
    - Synthetic Data for Computer Vision - CVPR workshop 2024/2025