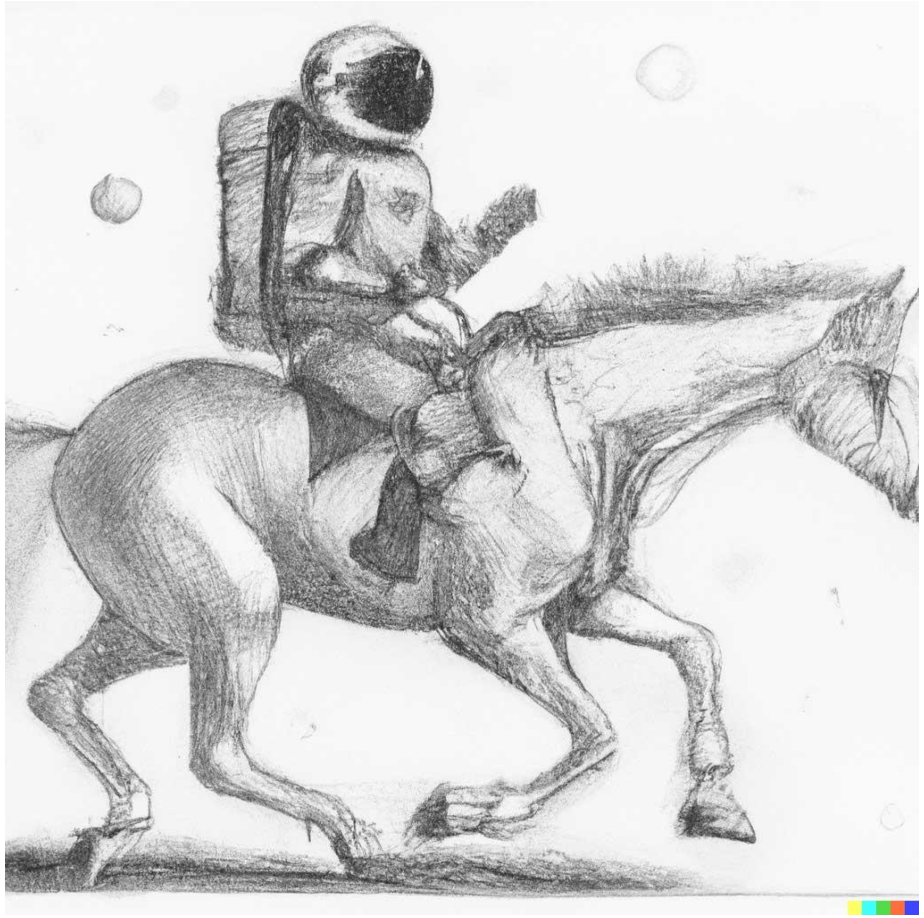
Generative Models & Determining Trained

Presenter: Kim Seung Hwan (overnap@khu.ac.kr)

“An astronaut riding a horse in a photorealistic style”



“An astronaut riding a horse as a pencil drawing”

OpenAI DALL-E 2

Goal

- Determining if a model has trained on some data
- This is important in two ways as DNNs are becoming commercialized
    1. Copyright aspect; even more important in generative models
    2. Metric aspect; the model's training efficiency, generalization performance, and the value of new data for it
- It seems worth, but is it possible?

Goal

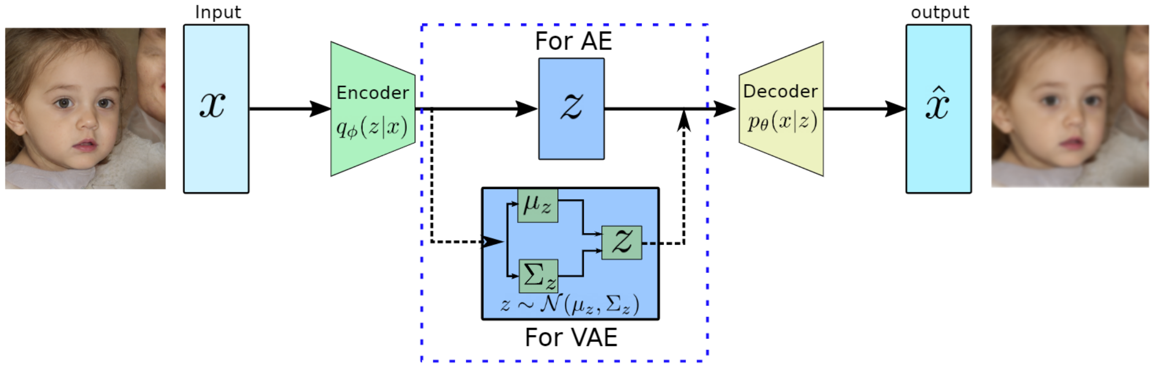
- My thoughts:
- In supervised learning, a model is forced to represent training data
- The model is optimized for the training data, not the true distribution
- The ideal generalization is difficult to achieve empirically
- It will be able to determine under appropriate constraints (but so what?)

Progress

- Studying generative models, especially VAE and Diffusion Model
- Naïve experiments and analysis attempts on simplified problem

VAE

- **V**ariational **A**uto **E**ncoder
- Let the latent of AE be indeterministic
- This will be some distribution (e.g. Gaussian RV)
- For encoder $q(z|x)$ and decoder $p(x|z)$, we want to know $p(z|x)$
- Indirectly perform MLE by applying variational inference ideas



VAE

$$
\begin{aligned}
\log p(\boldsymbol{x}) &= \log p(\boldsymbol{x}) \int q_\phi(\boldsymbol{z}|\boldsymbol{x}) dz && \left(\text{Multiply by } 1 = \int q_\phi(\boldsymbol{z}|\boldsymbol{x}) d\boldsymbol{z}\right) && (9)\\
&= \int q_\phi(\boldsymbol{z}|\boldsymbol{x}) (\log p(\boldsymbol{x})) dz && (\text{Bring evidence into integral}) && (10)\\
&= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} [\log p(\boldsymbol{x})] && (\text{Definition of Expectation}) && (11)\\
&= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})}\right] && (\text{Apply Equation 2}) && (12)\\
&= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x},\boldsymbol{z}) q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p(\boldsymbol{z}|\boldsymbol{x}) q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] && \left(\text{Multiply by } 1 = \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right) && (13)\\
&= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p(\boldsymbol{z}|\boldsymbol{x})}\right] && (\text{Split the Expectation}) && (14)\\
&= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] + D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p(\boldsymbol{z}|\boldsymbol{x})) && (\text{Definition of KL Divergence}) && (15)\\
&\geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] \leftarrow \text{ELBO} && (\text{KL Divergence always} \geq 0) && (16)
\end{aligned}
$$

VAE

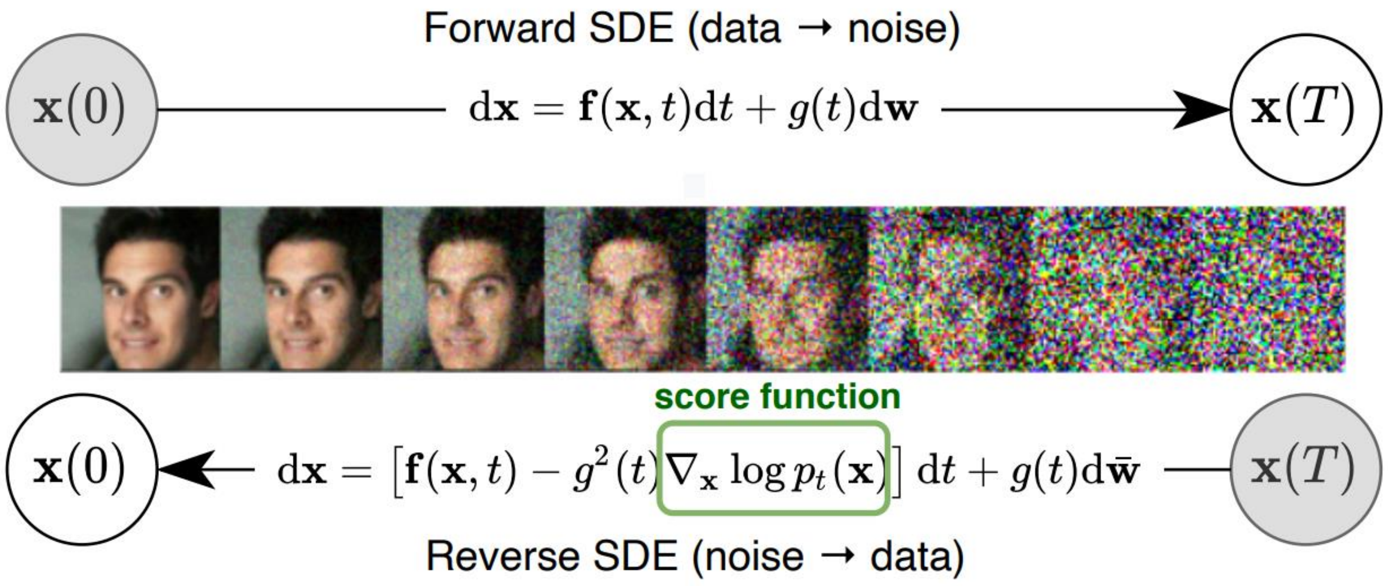
- "Increasing ELBO achieves MLE and posterior matching simultaneously"?
- What does MLE mean? Is it right to only grow the likelihood on training data?
- May fail to reduce true posterior matching term:
- When increasing ELBO, decreasing p.m. term is only guaranteed when θ is fixed!

$$\underset{\theta}{\underline{\log p(\boldsymbol{x})}} = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] + \underbrace{D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p(\boldsymbol{z}|\boldsymbol{x}))}_{\text{true posterior matching term}}\ \phi$$

$$\geq \underbrace{\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]}_{\text{ELBO}} = \underbrace{\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\right]}_{\text{reconstruction term}} - \underbrace{D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p(\boldsymbol{z}))}_{\text{prior matching term}}$$

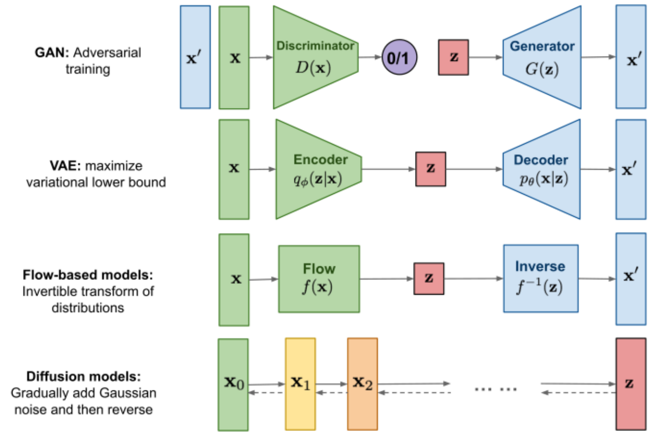
Diffusion Model

- Adding small Gaussians to a sample to make it a full Gaussian
- It requires a lot of steps



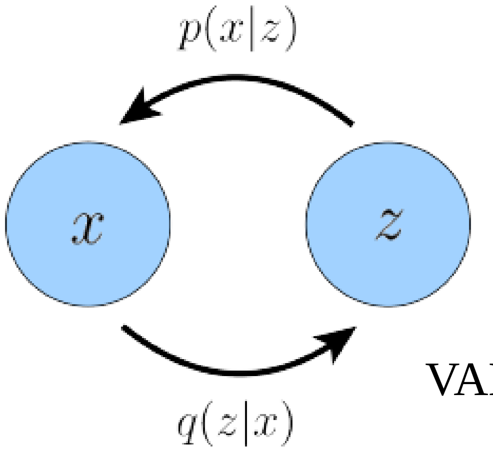
Diffusion Model

- Generative models convert data x to latent data z and then reconstruct them back to x or something similar
- Diffusion model is one of the score-based generative model
- The score is the gradient-log of the true data distribution $\nabla_x \log p(x)$
- There is a lot of math involved, but it is hard to know the meaning…
- Let us approach it from a different perspective and then come back

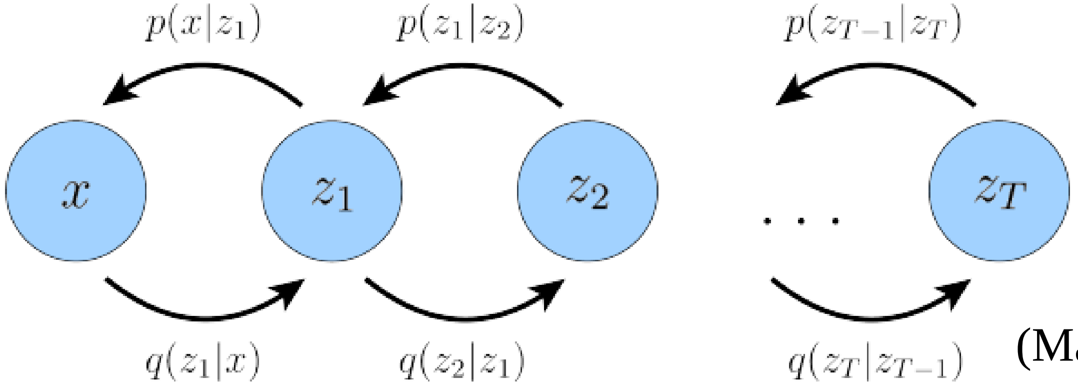


Diffusion Model

- A hierarchical structure in which several VAEs are connected
- This is called HVAE and is a generalization of VAE
- Like DNNs, HVAEs are likely to have more representation ability



VAE



(Markovian) HVAE

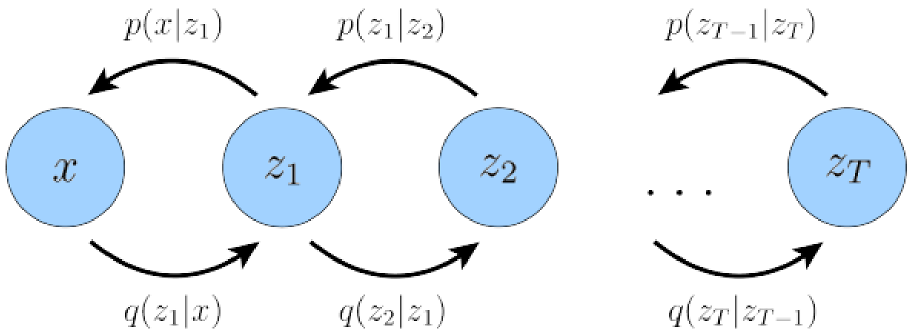
Diffusion Model

- But it is too free to model; Let us set each VAE to be a Markov chain
- Then, given the original data $x = z_0$, for each time t and latent z_t, it can be modeled with encoders $q(z_t|z_{t-1})$ and decoders $p(z_{t-1}|z_t)$
- Model training can also be done with ELBO in the same way as VAE!

$$p(\boldsymbol{x}, \boldsymbol{z}_{1:T}) = p(\boldsymbol{z}_T) p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}_1) \prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)$$

$$q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x}) = q_{\boldsymbol{\phi}}(\boldsymbol{z}_1|\boldsymbol{x}) \prod_{t=2}^{T} q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{z}_{t-1})$$

'cause it's Markov chain!



$$\begin{aligned}
\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}, \boldsymbol{z}_{1:T}) d\boldsymbol{z}_{1:T} \\
&= \log \int \frac{p(\boldsymbol{x}, \boldsymbol{z}_{1:T}) q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} d\boldsymbol{z}_{1:T} \\
&= \log \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \left[\frac{p(\boldsymbol{x}, \boldsymbol{z}_{1:T})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \right] \\
&\geq \underbrace{\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z}_{1:T})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \right]}_{\text{ELBO}}
\end{aligned}$$

Diffusion Model

- If $T = 1$, it is equivalent to ELBO in vanilla VAE
- The third term is very dominant in training cost!
- Because it has to be calculated T times and every KL divergence must be optimized simultaneously
- How to be solve?

$$
\begin{aligned}
\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}_{0:T}) d\boldsymbol{x}_{1:T} \\
&= \log \int \frac{p(\boldsymbol{x}_{0:T}) q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} d\boldsymbol{x}_{1:T} \\
&= \log \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \right] \\
&\geq \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \right] \\
&= \underbrace{\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1) \right]}_{\text{reconstruction term}} - \underbrace{\mathbb{E}_{q(\boldsymbol{x}_{T-1}|\boldsymbol{x}_0)} \left[D_{\mathrm{KL}}(q(\boldsymbol{x}_T|\boldsymbol{x}_{T-1}) \parallel p(\boldsymbol{x}_T)) \right]}_{\text{prior matching term}} - \sum_{t=1}^{T-1} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t+1}|\boldsymbol{x}_0)} \left[D_{\mathrm{KL}}(q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) \parallel p_\theta(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})) \right]}_{\text{consistency term}} \\
&= \underbrace{\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1) \right]}_{\text{reconstruction term}} - \underbrace{D_{\mathrm{KL}}(q(\boldsymbol{x}_T|\boldsymbol{x}_0) \parallel p(\boldsymbol{x}_T))}_{\text{prior matching term}} - \sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)} \left[D_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)) \right]}_{\text{denoising matching term}}
\end{aligned}
$$

Diffusion Model

- Diffusion model solves this by imposing strong constraints
  1. The latent dimension = the data dimension
  2. Every latent encoder is a linear Gaussian model
  3. The latent in final step T must be a standard Gaussian
- This cleans up ELBO because encoders $q(z_t|z_{t-1})$ are parameterless
- An important intuition is that the sum of a Gaussian RV is another Gaussian RV
- i.e., $X \sim N(\mu_1, {\sigma_1}^2), Y \sim N(\mu_2, {\sigma_2}^2) \Rightarrow X + Y \sim N(\mu_1 + \mu_2, {\sigma_1}^2 + {\sigma_2}^2)$

Diffusion Model

- Diffusion model solves this by imposing strong constraints
    1. The latent dimension = the data dimension
    2. Every latent encoder is a linear Gaussian model
    3. The latent in final step T must be a standard Gaussian
- This cleans up ELBO because encoders $q(z_t|z_{t-1})$ are parameterless
- Let $q(z_t|z_{t-1}) = N\left(z_t; \sqrt{\alpha_t}, (1-\alpha_t)I\right)$ for every t
- So $x_t = \sqrt{\alpha_t}x_{t-1} + \sqrt{1-\alpha_t}\epsilon$ where $\epsilon \sim N(\epsilon; 0, I)$

Diffusion Model

$$
\begin{aligned}
\boldsymbol{x}_t &= \sqrt{\alpha_t}\boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1}^* \\
&= \sqrt{\alpha_t}\left(\sqrt{\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}_{t-2}^*\right) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1}^* \\
&= \sqrt{\alpha_t\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\alpha_t-\alpha_t\alpha_{t-1}}\boldsymbol{\epsilon}_{t-2}^* + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1}^* \\
&= \sqrt{\alpha_t\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\sqrt{\alpha_t-\alpha_t\alpha_{t-1}}^2 + \sqrt{1-\alpha_t}^2}\boldsymbol{\epsilon}_{t-2} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\alpha_t-\alpha_t\alpha_{t-1}+1-\alpha_t}\boldsymbol{\epsilon}_{t-2} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\boldsymbol{\epsilon}_{t-2} \\
&= \ldots \\
&= \sqrt{\prod_{i=1}^{t}\alpha_i}\boldsymbol{x}_0 + \sqrt{1-\prod_{i=1}^{t}\alpha_i}\boldsymbol{\epsilon}_0 \\
&= \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0 \\
&\sim \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\mathbf{I})
\end{aligned}
$$

- An important intuition is that the sum of a Gaussian RV is another Gaussian RV
- i.e., $X \sim N(\mu_1, {\sigma_1}^2), Y \sim N(\mu_2, {\sigma_2}^2) \Rightarrow X + Y \sim N(\mu_1 + \mu_2, {\sigma_1}^2 + {\sigma_2}^2)$

Diffusion Model

$$\underbrace{\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1)\right]}_{\text{reconstruction term}} - \cancel{\underbrace{D_{\mathrm{KL}}(q(\boldsymbol{x}_T|\boldsymbol{x}_0)\parallel p(\boldsymbol{x}_T))}_{\text{prior matching term}}} - \sum_{t=2}^{T}\underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[D_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0)\parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t))\right]}_{\text{denoising matching term}}$$

$$\begin{aligned}
q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0) &= \frac{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{x}_0)q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)}{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\\
&= \frac{\mathcal{N}(\boldsymbol{x}_t;\sqrt{\alpha_t}\boldsymbol{x}_{t-1},(1-\alpha_t)\mathbf{I})\mathcal{N}(\boldsymbol{x}_{t-1};\sqrt{\bar\alpha_{t-1}}\boldsymbol{x}_0,(1-\bar\alpha_{t-1})\mathbf{I})}{\mathcal{N}(\boldsymbol{x}_t;\sqrt{\bar\alpha_t}\boldsymbol{x}_0,(1-\bar\alpha_t)\mathbf{I})}\\
&\propto \mathcal{N}(\boldsymbol{x}_{t-1};\underbrace{\frac{\sqrt{\alpha_t}(1-\bar\alpha_{t-1})\boldsymbol{x}_t+\sqrt{\bar\alpha_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar\alpha_t}}_{\mu_q(\boldsymbol{x}_t,\boldsymbol{x}_0)},\underbrace{\frac{(1-\alpha_t)(1-\bar\alpha_{t-1})}{1-\bar\alpha_t}\mathbf{I}}_{\boldsymbol{\Sigma}_q(t)})
\end{aligned}$$

Diffusion Model

$$\overset{?}{\underbrace{\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1)\right]}_{\text{reconstruction term}}} - \cancel{\underbrace{D_{\mathrm{KL}}(q(\boldsymbol{x}_T|\boldsymbol{x}_0)\parallel p(\boldsymbol{x}_T))}_{\text{prior matching term}}} - \boxed{\sum_{t=2}^{T}\underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[D_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0)\parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t))\right]}_{\text{denoising matching term}}} \ *$$

$$\boldsymbol{\mu}_q(\boldsymbol{x}_t,\boldsymbol{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t}$$

$$\underbrace{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{I}}_{\boldsymbol{\Sigma}_q(t)}$$

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)}{1-\bar{\alpha}_t}$$

$$\begin{aligned}
&\arg\min_{\boldsymbol{\theta}} D_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0)\parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t))\\
&= \arg\min_{\boldsymbol{\theta}} D_{\mathrm{KL}}(\mathcal{N}(\boldsymbol{x}_{t-1};\boldsymbol{\mu}_q,\boldsymbol{\Sigma}_q(t))\parallel \mathcal{N}(\boldsymbol{x}_{t-1};\boldsymbol{\mu}_{\boldsymbol{\theta}},\boldsymbol{\Sigma}_q(t)))\\
&= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)}\frac{\bar{\alpha}_{t-1}(1-\alpha_t)^2}{(1-\bar{\alpha}_t)^2}\left[\left\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\boldsymbol{x}_0\right\|_2^2\right]
\end{aligned}$$

- Set $p(x_{t-1}|x_t)$ to be Gaussian to reduce KL divergence
- Since the variance is only a function of t, we just take it

Diffusion Model

$$\frac{1}{2\sigma_q^2(t)} \frac{\bar{\alpha}_{t-1}(1-\alpha_t)^2}{(1-\bar{\alpha}_t)^2}\left[\left\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\right\|_2^2\right]$$

$$= \frac{1}{2}\left(\frac{\bar{\alpha}_{t-1}}{1-\bar{\alpha}_{t-1}} - \frac{\bar{\alpha}_t}{1-\bar{\alpha}_t}\right)\left[\left\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\right\|_2^2\right]$$

$$= \frac{1}{2}\left(\mathrm{SNR}(t-1) - \mathrm{SNR}(t)\right)\left[\left\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\right\|_2^2\right]$$

$$\mathrm{SNR}(t) = \frac{\bar{\alpha}_t}{1-\bar{\alpha}_t} = \frac{\mu^2}{\sigma^2}$$

- Finding the appropriate hyperparameter α can be done with NN
- But this is a bit questionable

$$\mathrm{SNR}(t) = \exp(-\omega_{\boldsymbol{\eta}}(t))$$

$$\frac{\bar{\alpha}_t}{1-\bar{\alpha}_t} = \exp(-\omega_{\boldsymbol{\eta}}(t))$$

$$\therefore \bar{\alpha}_t = \mathrm{sigmoid}(-\omega_{\boldsymbol{\eta}}(t))$$

$$\therefore 1 - \bar{\alpha}_t = \mathrm{sigmoid}(\omega_{\boldsymbol{\eta}}(t))$$

Diffusion Model

- Expression manipulation never ends…
- We can view the optimization problem of diffusion model in 3 aspects:
  - Training original data!

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \frac{\bar{\alpha}_{t-1}(1-\alpha_t)^2}{(1-\bar{\alpha}_t)^2}\left[\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\|_2^2\right]$$

  - Training the noise!

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{(1-\bar{\alpha}_t)\alpha_t}\left[\|\boldsymbol{\epsilon}_0 - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)\|_2^2\right]$$

  - Training the score!

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{\alpha_t}\left[\|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \nabla \log p(\boldsymbol{x}_t)\|_2^2\right]$$

$\because$ $$\boldsymbol{x}_0 = \frac{\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0}{\sqrt{\bar{\alpha}_t}}$$

This is the 'score'!

"commonly used to correct sample bias"

$\because$ 1 $$\mathbb{E}\left[\boldsymbol{\mu}_z | \boldsymbol{z}\right] = \boldsymbol{z} + \boldsymbol{\Sigma}_z \boxed{\nabla_{\boldsymbol{z}} \log p(\boldsymbol{z})}$$ Tweedie's Formula

2. $$\boldsymbol{x}_0 = \frac{\boldsymbol{x}_t + (1-\bar{\alpha}_t)\nabla \log p(\boldsymbol{x}_t)}{\sqrt{\bar{\alpha}_t}} = \frac{\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0}{\sqrt{\bar{\alpha}_t}}$$

$\Rightarrow$ $$\nabla \log p(\boldsymbol{x}_t) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_0$$

Score-based Generative Model

- The score is the gradient-log of the true data distribution $\nabla_x \log p(x)$
- That is, for arbitrarily flexible and parameterizable function f called the energy function, arbitrarily flexible probability distribution:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{Z_{\boldsymbol{\theta}}} e^{-f_{\boldsymbol{\theta}}(\boldsymbol{x})}$$

- But normalizing constant Z can be difficult to compute tractably
- This can be fixed by setting 'the score' to gradient-log and finding the score instead

$$\begin{aligned}
\nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) &= \nabla_{\boldsymbol{x}} \log\left(\frac{1}{Z_{\boldsymbol{\theta}}} e^{-f_{\boldsymbol{\theta}}(\boldsymbol{x})}\right) \\
&= \nabla_{\boldsymbol{x}} \log \frac{1}{Z_{\boldsymbol{\theta}}} + \nabla_{\boldsymbol{x}} \log e^{-f_{\boldsymbol{\theta}}(\boldsymbol{x})} \\
&= -\nabla_{\boldsymbol{x}} f_{\boldsymbol{\theta}}(\boldsymbol{x}) \\
&\approx \boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x})
\end{aligned}$$

Score-based Generative Model

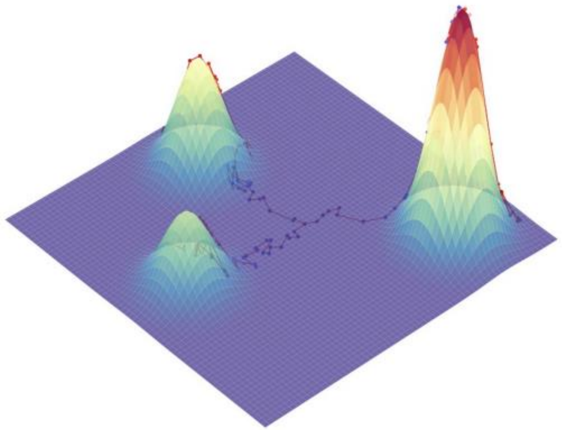
- Express the score as a DNN by optimizing the Fisher Divergence

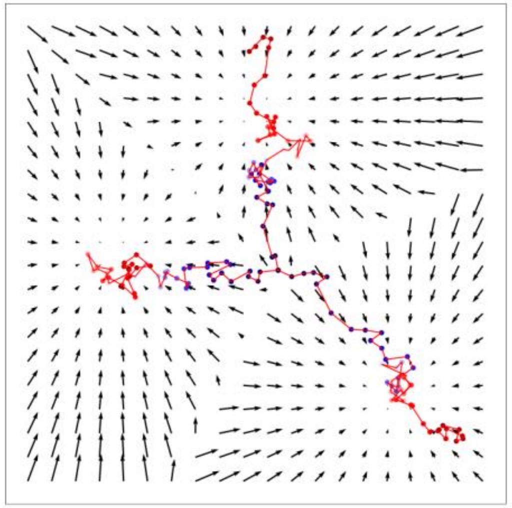
$$\mathbb{E}_{p(\boldsymbol{x})}\left[\|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x})-\nabla\log p(\boldsymbol{x})\|_2^2\right]$$ …and this is the vanilla score matching

- The score means the direction the log-likelihood increases
- The Score-based generative model borrows the idea of Langevin dynamics, a molecular system model that can represent molecular diffusion

$$M\ddot{\mathbf{X}}=-\nabla U(\mathbf{X})-\gamma M\dot{\mathbf{X}}+\sqrt{2M\gamma k_B T}\,\mathbf{R}(t)$$

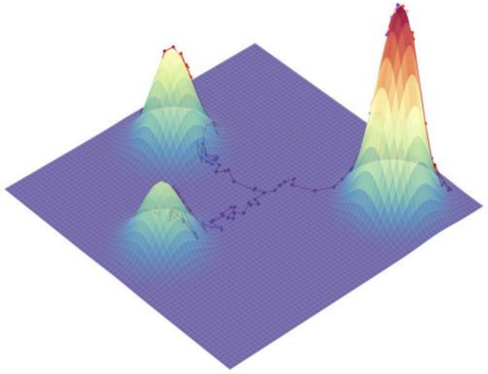
$$\boldsymbol{x}_{i+1}\leftarrow\boldsymbol{x}_i+c\nabla\log p(\boldsymbol{x}_i)+\sqrt{2c}\boldsymbol{\epsilon},\quad i=0,1,...,K$$

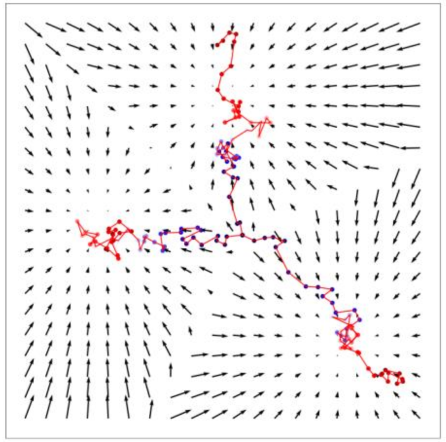


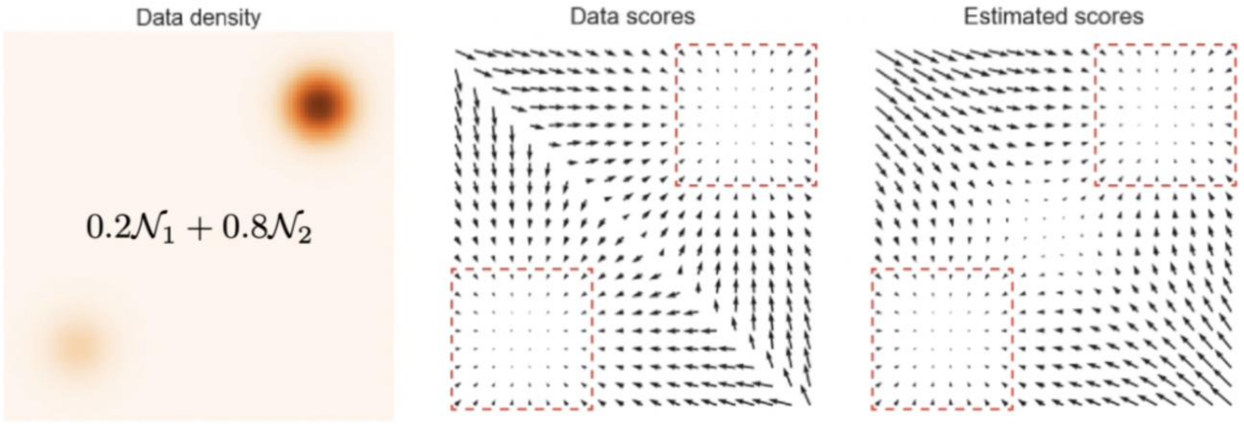


Score-based Generative Model

- It can estimate an intractable distribution $p_{\theta}(\boldsymbol{x}) = \frac{1}{Z_{\theta}} e^{-f_{\theta}(\boldsymbol{x})}$
- But it is very computationally expensive and has various problems:
  1. Ill-defined when x lies on a low-dimensional manifold
  2. Not be accurate in low density regions
  3. Density is not reflected well (e.g., mixture model) ???



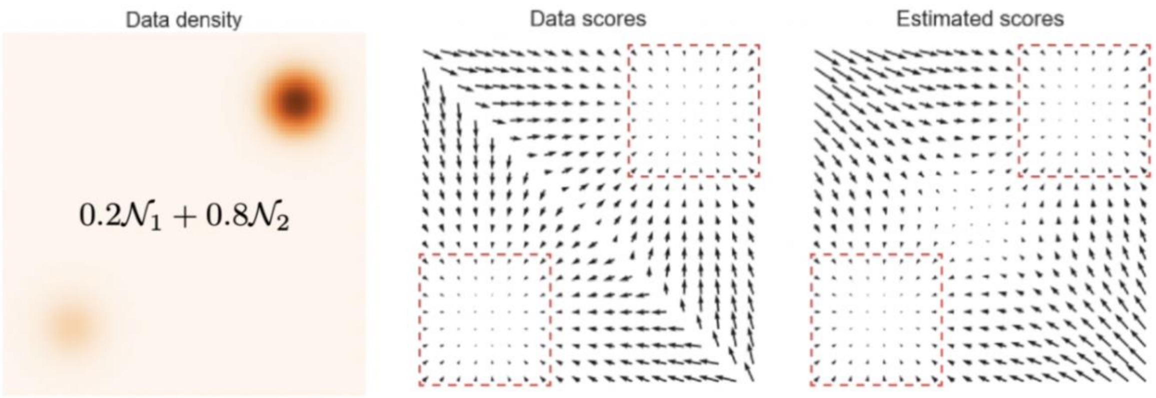




Score-based Generative Model

- This can be handled by adding Gaussians of different strengths
- The problem of computational cost is solved by estimating the scores of distributions made from noising samples
- Since Gaussian is defined in all spaces, it solves many problems

1. Ill-defined when x lies on a low-dimensional manifold OK
2. Not be accurate in low density regions somewhat ok
3. Density is not reflected well (e.g., mixture model) ???



Diffusion Model

- Therefore, if we optimize all the Fisher Divergence at each noise level,

$$\arg\min_{\boldsymbol{\theta}} \sum_{t=1}^{T} \lambda(t) \mathbb{E}_{p_{\sigma_t}(\boldsymbol{x}_t)} \left[\| \boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}, t) - \nabla \log p_{\sigma_t}(\boldsymbol{x}_t) \|_2^2 \right]$$

- A similar expression emerges when solved from the HVAE perspective
- Furthermore, if we modify Langevin dynamics sampling in terms of simulated annealing:
  - Initialize from some fixed prior (e.g., uniform, gaussian)
  - Running Langevin dynamics for each $t = T, T-1, T-2$ ...
  - The starting point of each step is the ending point of the previous one
- It can be perfectly modeled as a Markovian HVAE!

Diffusion Model

- In conclusion, the diffusion model can be interpreted as an
  1. Hierarchical VAE with strong constraints to deal with the computational cost
  2. Denoising score-based generative model with a clear, comprehensive explanation and great performance
- These are not separate but complementary to each other
- Approaching from different perspectives always gives good ideas!
- I hope this helped you get a rough understanding

Diffusion Model

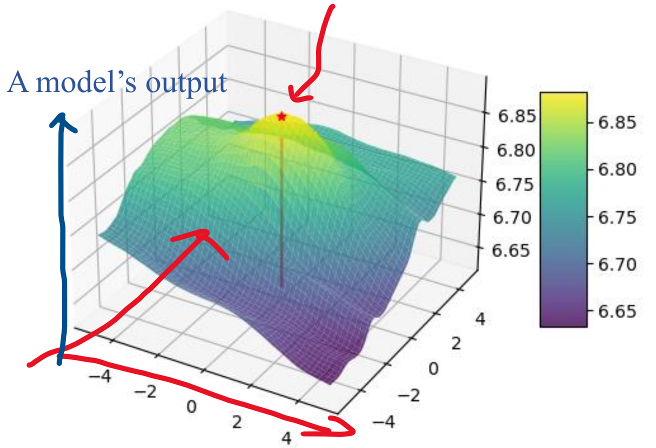
- Diffusion models still have very important topics like SDE, but these are not fully understood. (too HARD for me!!!)
- Simple to understand, it seems to be an explanation that unifies the continuous Langevin dynamics process and the discrete DNN process
- The guidance (conditional one) is also possible (e.g., Image-Text)
- However, there are many other stories about these…
- For more details on this presentation, SDE, etc., see below:
  - Luo et al. "Understanding Diffusion Models: A Unified Perspective" [2208.11970]
  - Song et al. "Generative Modeling by Estimating Gradients of the Data Distribution" [1907.05600] (NCSN)
  - Ho et al. "Denoising Diffusion Probabilistic Models" [2006.11239] (DDPM)

Experiments and Analysis

- This is one of the results of the experiment in the last presentation
- Simplifying the situation as a classification problem, let us determine whether the model has learned from that data
- Are there any significance features that only trained data have?

- Looking at the picture on the right
- Can we say that the data has been trained?

The trained data (I thought it’s significant)

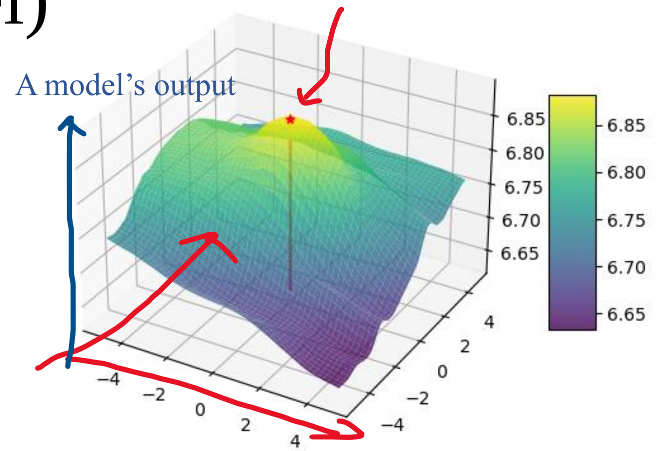


A plane in the data space (no dimensionality reduction!)

Experiments and Analysis

- At first glance, it seems trained (or maybe not)
- Can we set a specific metric?
- My thoughts: Find a feature that is significant compared to other points in the L_p norm hypersphere centered on the data
- A good candidate is logit (the output of the model)

The trained data (I thought it's significant)

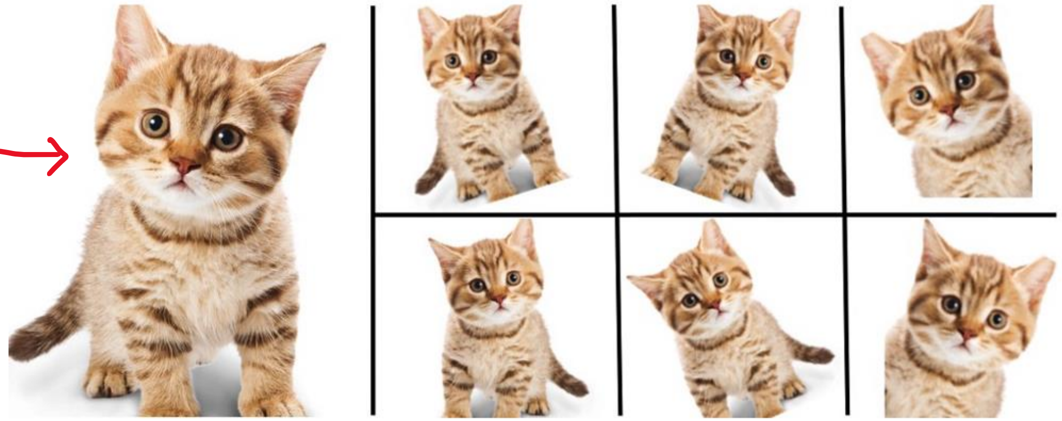


A plane in the data space (no dimensionality reduction!)

Experiments and Analysis

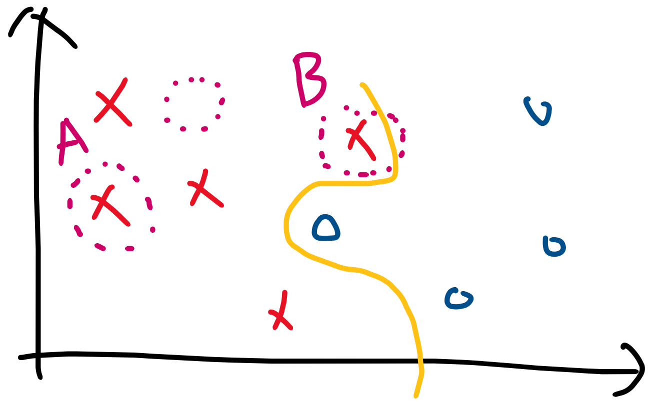
- e.g., if the output of the data itself is noticeably higher than the output of Gaussian noises centered on that data, it has been trained
- Since the model is supervised by training data, the trained data itself will produce a higher output than its vicinity
- But…

If only this cat was trained,
the output of it would be higher than the other cats
(this can be seen as overfitting)



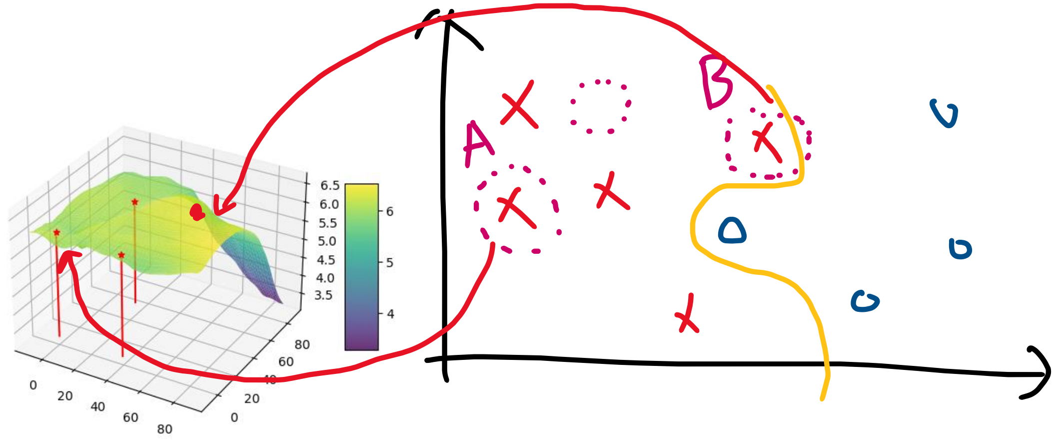
Experiments and Analysis

- Let us consider a simple binary classification problem
- According to the hypersphere hypothesis, both A and B should exhibit significant values (of metric) compared to their surroundings
- But B will be more significant than A, since B is close to the d.b.
- A might even have values same as $C\ldots$

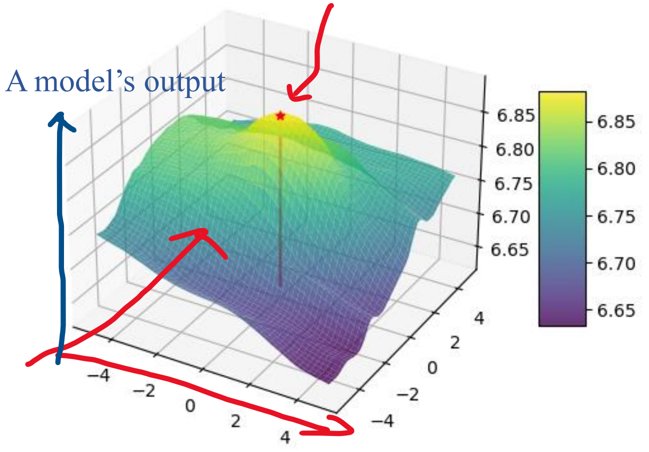


Experiments and Analysis

- Significant values can be found
- But we do not know if it is caused by just the supervision by that one sample or by others
- No matter how special it is, there is no guarantee that this really happened because of itself



The trained data (I thought it's significant)



A plane in the data space (no dimensionality reduction!)

Experiments and Analysis

- However, I believe that the model has prints of the supervision
- In my opinion, the output of the input side of the model with less abstraction is more likely to be affected by the supervision
- But I doubt that this is a good direction
- As we move towards the input side of the model, less meaning is given, and more information is required…

Experiments and Analysis

- I might change the goal
- Even if I know all parameters of the model, it is difficult to determine
- If the constraint is not a single piece of data, but a small set of data, I can approach it from an adversarial attack perspective
- But it is more meaningful to determine for just one sample

Thank you for listening

Presenter: Kim Seung Hwan (overnap@khu.ac.kr)