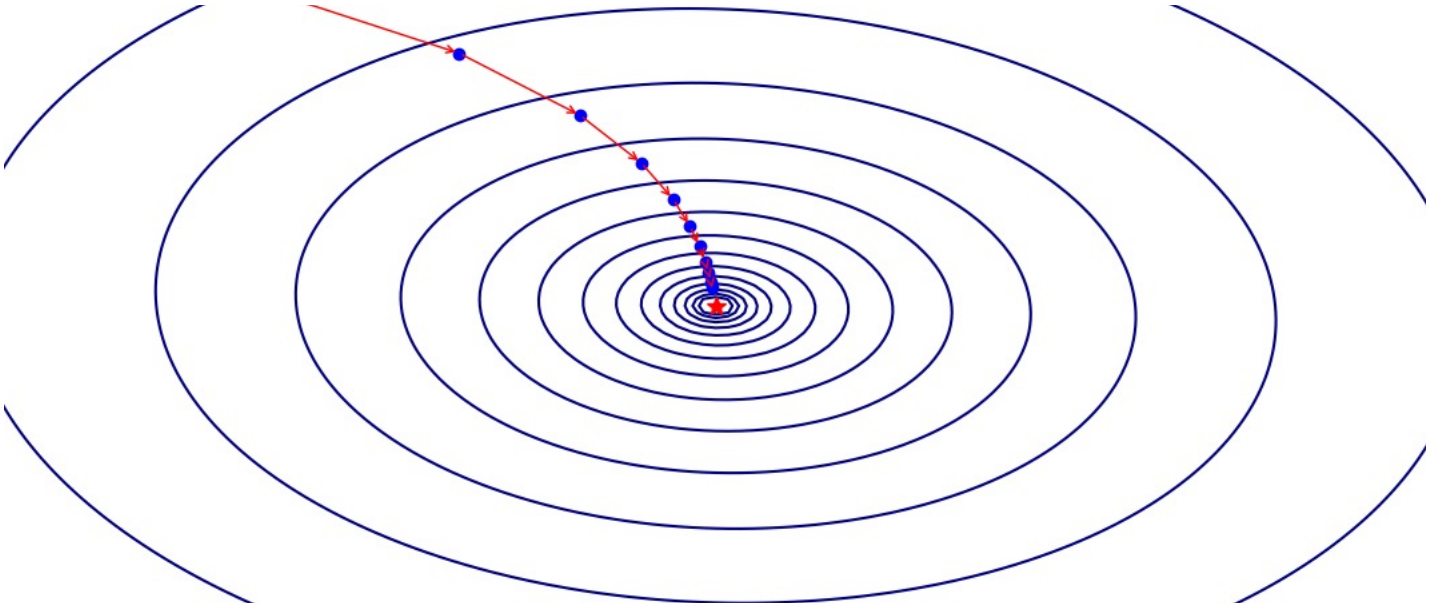


# Advanced Topics in Deep Learning

Summer Semester 2024

6. Generative Models II

10.07.2024

Prof. Dr. Vasileios Belagiannis

Chair of Multimedia Communications and Signal Processing

Friedrich-Alexander-Universität
Technische Fakultät

# Course Topics



1. Interpretability.
2. Attention and Transformers.
3. Self-supervised Learning I.
4. Self-supervised Learning II.
5. Similarity Learning.
6. **Generative Models.**
7. Model Compression.
8. Transfer learning, domain adaptation, few-shot learning.
9. Uncertainty Estimation.
10. Geometric Deep Learning.
11. Recap and Q&A.

- The exam will be written.
- We will have an exam preparation test.

Acknowledgements

- Special thanks Arij Bouazizi, Julia Hornauer, Julian Wiederer, Adrian Holzbock and Youssef Dawoud for contributing to the lecture preparation.

# Recap



- Generative models with Deep Neural Networks.
- Generative Adversarial Networks.
- Auto-Encoders.
- Variational Auto-Encoders.

# Today's Agenda and Objectives



- Generative Models.
- Diffusion Models.
- Forward Process.
- Reverse Process.
- Sampling.
- Conditional Diffusion Models.

# Generative Models



- We focus on generative models where no labels are provided, similar to unsupervised learning and self-learning.
  - Consider the training set $D = \{\{\boldsymbol{x_1}\}, \ldots, \{\boldsymbol{x_n}\}\}$. Each sample can be, for instance, an image.
  - The generative model $p_\theta(\boldsymbol{x})$ will learn to form images based on the training set $D$, which expresses the data distribution $p_{data}(\boldsymbol{x})$.
- By learning the data distribution based on the training set, the generative model can used for sampling, e.g., sample new images.

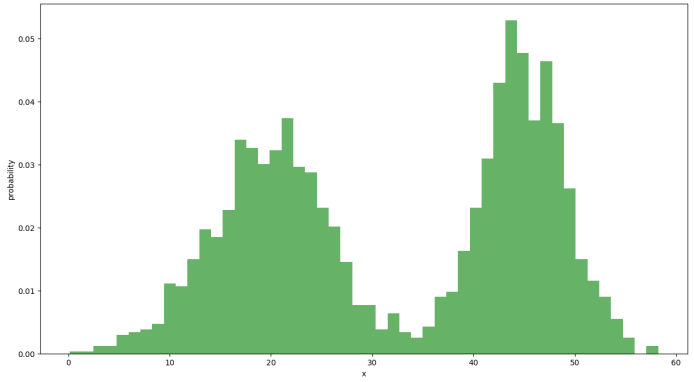
*Generated with DiffusionBee.*

# Generative Model Training

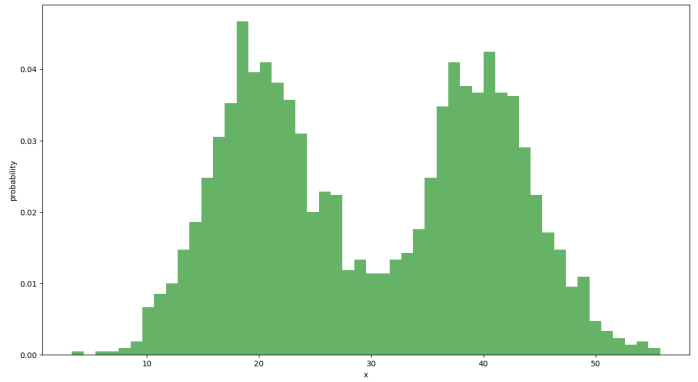


- We have seen two approaches to training the generative model $p_\theta(\boldsymbol{x})$ to match the data distribution $p_{data}(\boldsymbol{x})$.
  - **Maximize** the expected log-likelihood
  - **Minimize** the divergence between $p_\theta(\boldsymbol{x})$ and $p_{data}(\boldsymbol{x})$.
- Generative Adversarial Networks (GANs)
  - Minimize the Jensen-Shannon divergence.
- Variational Autoencoders (VAEs)
  - Maximize the log-likelihood of the ELBO.

Training Data / Data distribution $p_{data}(\boldsymbol{x})$




Learned Distribution $p_\theta(\boldsymbol{x})$

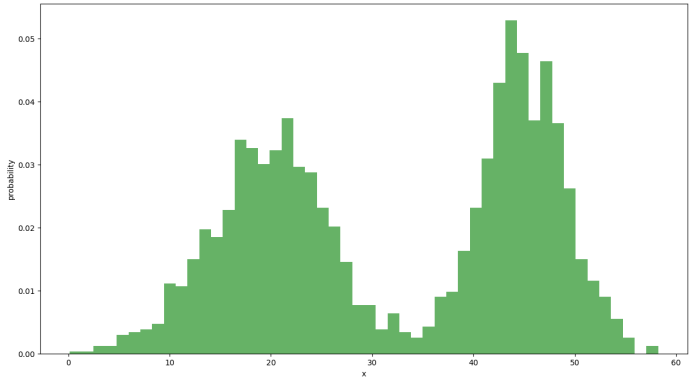


# Generative Model Training

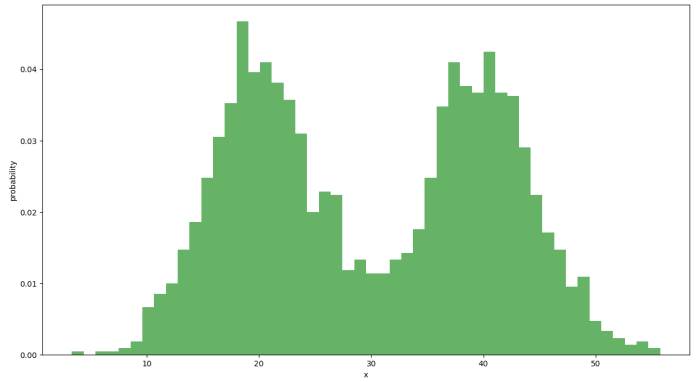


- Generative Adversarial Networks (GANs)
  - Minimize the *Jensen-Shannon* divergence.
  - $min_G max_D \ \mathbb{E}_{x\sim p_{data}(x)}[Log\,(D(x))] + \mathbb{E}_{z\sim p_z(z)}[Log\left(1-D\left(G(z)\right)\right)]$
- Variational Autoencoders (VAEs)
  - Maximize the log-likelihood of the ELBO.
  - $\mathcal{L}(\theta,\varphi;x^i) = -D_{KL}\left(q_\phi(z|x^i)||p_\theta(z)\right) + \mathbb{E}_{q_\phi(z|x^i)}[logp_\theta(x^i|z)].$

Training Data / Data distribution $p_{data}(\boldsymbol{x})$



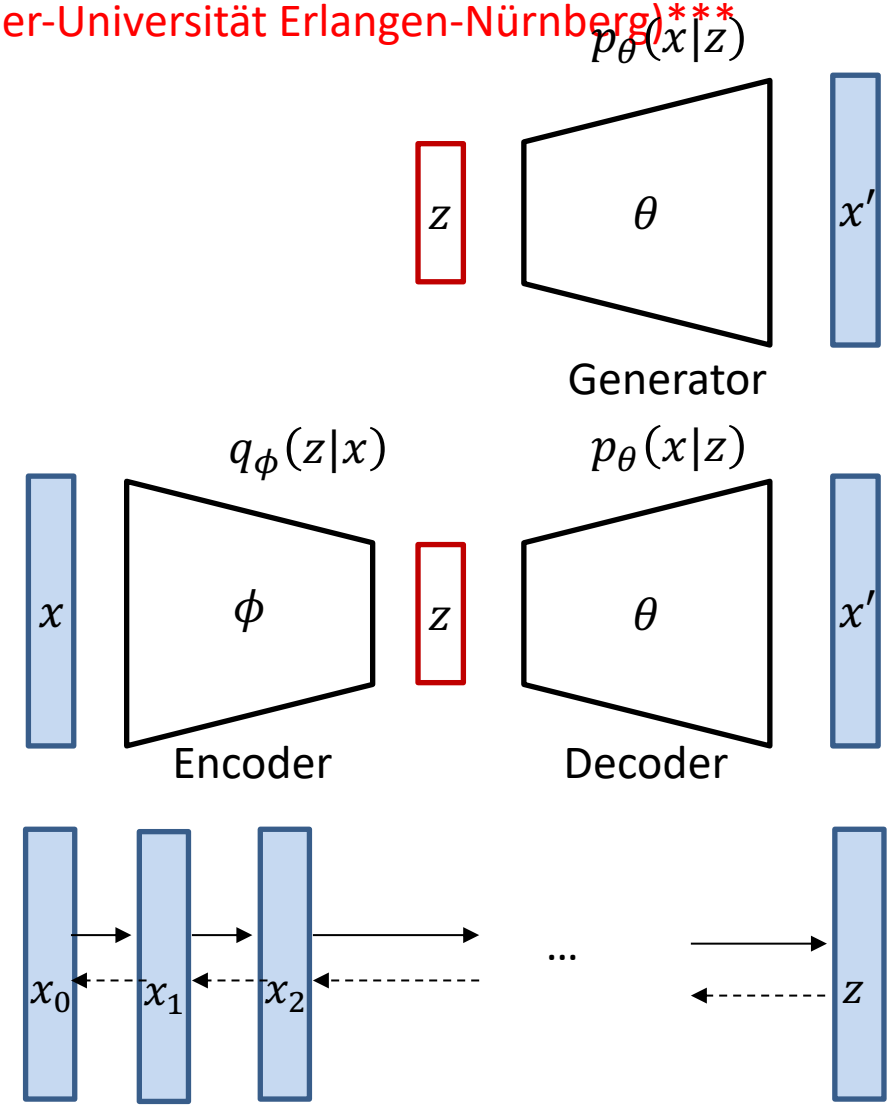

Learned Distribution $p_\theta(\boldsymbol{x})$



# Generative Model Sampling



- How do we generate a new sample?
- Both GANs and VAEs sample **noise** $z$ (based on a prior distribution) in the latent space to create a new sample in a **single** step.
- Diffusion models start with a fully noisy sample and **gradually (in multiple steps)** denoise it to an image.
  - We deal with noisy on the sample space, e.g., image space. It's possible to have noise in the latent space as well (latent diffusion).



# Diffusion Models



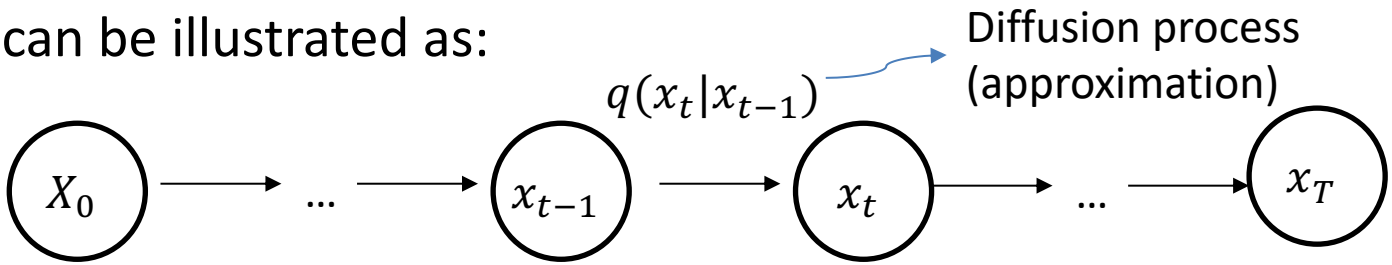
- The motivation is to start from a based distribution, like a Gaussian distribution and gradually (in multiple steps) convert it to the target data distribution, like an image.
  - This process is like a **Markov** chain. Each step in the transition from noise to image is a step in the chain.
  - Moreover, it is **assumed** that the current step depends only on the previous one (Markov property).
  - By making the Markov chain assumption, we can define the diffusion process as a sequence of Gaussian transitions. In the end, it is assumed to have an isotropic Gaussian distribution given a large number of steps.
- Diffusion models contain the step-wise* functionality not only during sampling but also during training.
- A diffusion model consists of:
  - **Forward** process (image to noise transition).
  - **Reverse** process (noise to image transition).
  - **Sampling** (noise to image transition).
- The forward and reverse processed are required to perform **training**.

*Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." Advances in neural information processing systems 33 (2020): 6840-6851.

# Forward Process



- Consider the sample $X_0$ that is noise free, e.g. an image.
- In the **forward** process, we gradually add small amounts of **Gaussian** noise to the image. Each steps makes the image $x_t$ noisier until we end with noise $x_T$.
- The process of adding noise is repeated for $T$ steps.
- Ideally, we will end up with pure noise (isotropic Gaussian distribution) after a large number of iterations.
- The process can be illustrated as:



# Forward Process Derivation



- For each time step, the image noising is given by:
  - $x_t = \sqrt{1-\beta_t} x_{t-1} + \sqrt{\beta_t}\epsilon.$
  - With $\epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and $\beta_t$ the noise variance.
  - The noise variance $\beta_1 \dots \beta_T$ is from 0 to 1. It tells us the amount of noise to add at the current step.
- Based on the update step $x_t$, we can now write the sampling as a **Gaussian** distribution:
  - $q(x_t|x_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t} x_{t-1}, \beta_t \boldsymbol{I}).$
  - $\beta_t$ corresponds to the current variance schedule.
  - $\sqrt{1-\beta_t} x_{t-1}$ is the mean value.
  - To derive the above, consider that:
    - $\mathcal{N}\left(\sqrt{1-\beta_t} x_{t-1}, \beta_t \boldsymbol{I}\right) = \sqrt{1-\beta_t} x_{t-1} + \beta_t \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$

# Forward Process Derivation (Cont.)

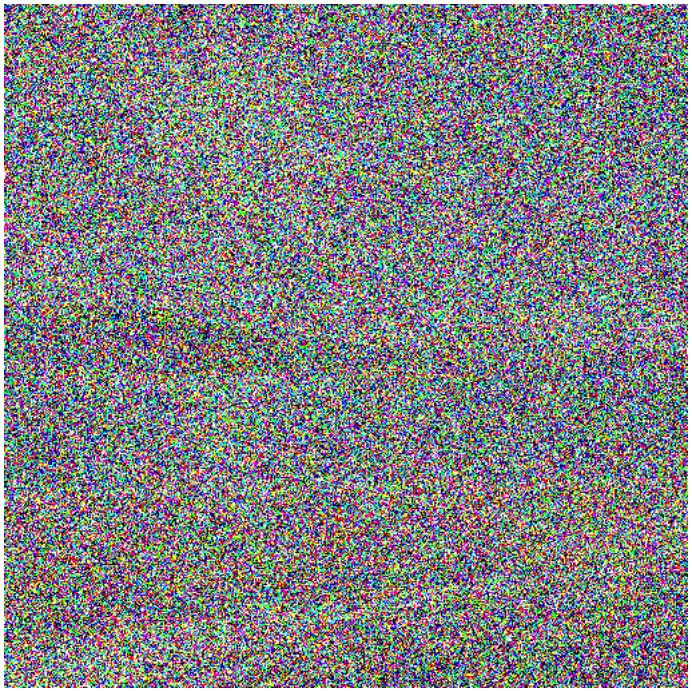


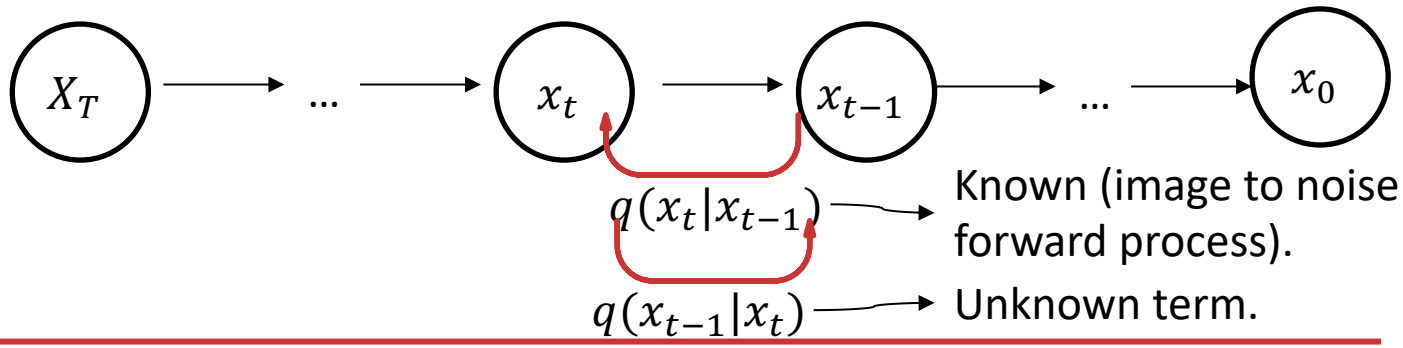
- For each time step, the image noising is given by:
  - $x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon$.
  - With $\epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and $\beta_t$ the noise variance.
- We can rewrite $x_t$ as:
  - $x_t = \sqrt{a_t}x_{t-1} + \sqrt{1-a_t}\epsilon$.
  - We define $a_t = 1 - \beta_t$ with $a_t \in (0,1)$.
  - Also define $\bar{a}_t = \prod_{i=1}^{t} \alpha_i$.
- The forward process is a recurrence. It can be written as:
  - $x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon = \sqrt{a_t}x_{t-1} + \sqrt{1-a_t}\epsilon = \sqrt{a_t}\left(\sqrt{a_{t-1}}x_{t-2} + \sqrt{1-a_{t-1}}\epsilon\right) + \sqrt{1-a_t}\epsilon = \cdots = \sqrt{a_t \dots a_1}x_0 + \sqrt{1-a_t \dots a_1}\epsilon = \sqrt{\bar{a}_t}x_0 + \sqrt{1-\bar{a}_t}\epsilon$.
  - We can thus **always** express $x_t$ w.r.t. $x_0$ as $q(x_t|x_0) = \mathcal{N}(\sqrt{\bar{a}_t}x_0, (1-\bar{a}_t)\boldsymbol{I})$. Also, we have $q(x_t|x_{t-1}) = \mathcal{N}(\sqrt{a_t}x_{t-1}, 1-a_t\boldsymbol{I})$.

# Reverse Process



- The noising steps cannot help us to create an image. However, the opposite approach (denoising) can lead to the **formation** of an image from Gaussian noise in several steps.
  - This is called the reverse process.
- Our goal is to learn with the diffusion model to go from the noise to the each using a multiple step process $q(x_{t-1}|x_t)$.
- We generate data from random noise.
- The process can be illustrated as:

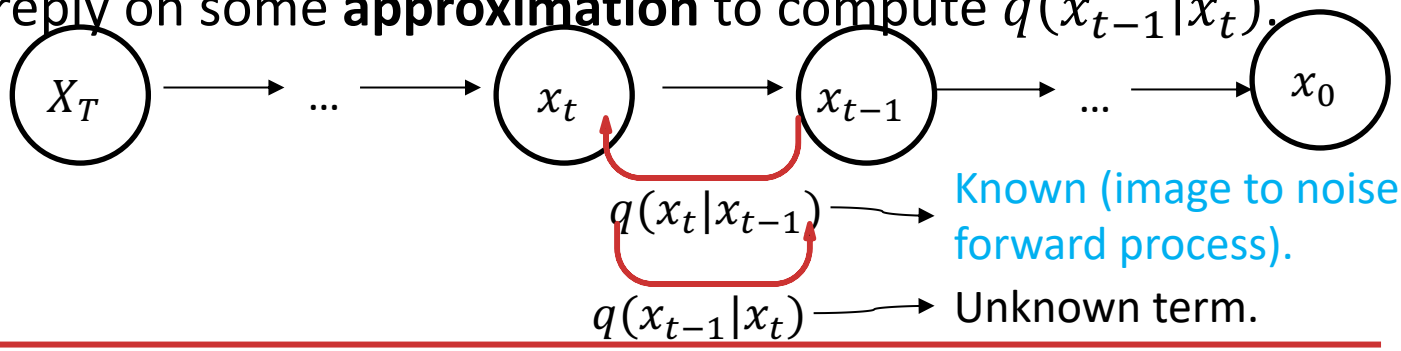




# Reverse Process



- The reverse prices $q(x_{t-1}|x_t)$ is clear unknown. For that reason, we consider first the Bayer rule:
  - $f(a|b) = \frac{f(a,b)}{f(b)} = \frac{f(a)f(b|a)}{f(b)}$.
- $q(x_{t-1}|x_t) = \frac{q(x_{t-1},x_t)}{q(x_t)} = \frac{q(x_{t-1})q(x_t|x_{t-1})}{q(x_t)} = q(x_t|x_{t-1})\frac{q(x_{t-1})}{q(x_t)}$
- $q(x_t)$ is computationally intractable because we must compute $q(x_t) = \int q(x_t|x_{t-1})\, q(x_{t-1})dx$.
- $q(x_t|x_{t-1})$ is a Gaussian distribution that we know (image to noise).
- We have to reply on some **approximation** to compute $q(x_{t-1}|x_t)$.



# Reverse Process



- We can assume forward process steps with very **small** noise schedule $\beta_t$.
  - Based on this assumption, we further assume that the reverse process can be a Gaussian distribution similar to the forward process.
- The reverse process approximation is a Gaussian with unknown mean and variance.
  - We rely on a neural network to **learn** the Gaussian parameters for the reverse process.
  - Our neural network model is defined as $p_\theta = (x_{t-1}|x_t) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t); \Sigma_\theta(x_t, t)\right)$.
- Our goal in training the neural network is to **maximize** the **log-likelihood** of the generated data using the reverse process.
  - Our objective is similar to VAEs training.
  - Recall that for VAEs, we have:
    - $\mathcal{L}(\theta, \varphi; x^i) = -D_{KL}\left(q_\phi(z|x^i)||p_\theta(z)\right) + \mathbb{E}_{q_\phi(z|x^i)}\left[logp_\theta(x^i|z)\right] \rightarrow$
    - $\mathbb{E}_{q_\phi(z|x^i)}\left[logp_\theta(x^i|z)\right] - D_{KL}\left(q_\phi(z|x^i)||p_\theta(z)\right) \leq logp_\theta(x)$
    - We can use the same objective for the diffusion process as well.

# Learning the Reverse Process



- We want to learn the reverse process $p_\theta(x_{t-1}|x_t)$ from the available forward process $q(x_t|x_{t-1})$.
- We will use the KL divergence as objective function where we will consider all time steps $T$. We have thus:
  - $D_{KL}\big(q(x_{1:T}|x_0)||p_\theta(x_{1:T}|x_0)\big)$.
- Similar to VAEs, we apply variational inference and optimize an evidence lower bound on the likelihood $p_\theta(x_0)$ of the sample $x_0$.
  - $logp_\theta(x_0) \geq logp_\theta(x_0) - D_{KL}\big(q(x_{1:T}|x_0)||p_\theta(x_{1:T}|x_0)\big) = logp_\theta(x_0) - \mathbb{E}_{x_{1:T}\sim q(x_{1:T}|x_0)}\left[log\frac{q(x_{1:T}|x_0)}{\frac{p_\theta(x_{1:T}|x_0)}{p_\theta(x_0)}}\right] = logp_\theta(x_0) - \mathbb{E}_q\left[log\frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})} + logp_\theta(x_0)\right] = -\mathbb{E}_q\left[log\frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})}\right]$.

# Learning the Reverse Process (Cont.)



- Now we have $log p_\theta(x_0) \geq -\mathbb{E}_q\left[log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})}\right]$.
  - Note that $q$ is fixed and we optimise only for $p_\theta$.
  - $-\mathbb{E}_q\left[log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})}\right]$
- Next, we would like to transform the above bound (ELBO) into a sum of simpler terms.
  - Consider the variational lower bound loss $L_{VLB} = \mathbb{E}_q\left[log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})}\right] = \mathbb{E}_q\left[log \frac{\prod_{t=1}^{T} q(x_t|x_{t-1})}{p_\theta(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)}\right] = \cdots = \mathbb{E}_q\left[D_{KL}\left(q(x_T|x_0)||p_\theta(x_T)\right)\right] + \sum_{t=2}^{T} D_{KL}\left(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)\right) - log\, p_\theta(x_0|x_1)$.
  - We have three terms in $L_{VLB}$.
    - We can write it as $L_{VLB} = L_T + L_{T-1} + \cdots + L_0$

# Learning the Reverse Process (Cont.)



- $L_T = D_{KL}\big(q(x_T|x_0)||p_\theta(x_T)\big)$ corresponds to the prior loss and compares the final $x_T$ with $x_0$.
- The reconstruction term $L_0 = -log\ p_\theta(x_0|x_1)$is the probability of the true $x_0$ given the current prediction $x_1$.
- The diffusion loss is $L_t = D_{KL}\big(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)\big) = D_{KL}\big(q(x_t|x_{t+1}, x_0)||p_\theta(x_t|x_{t+1})\big)$ for $1 \leq t \leq T-1$.
  - It measures the similarity of the learned reverse process $p_\theta(x_t|x_{t+1})$ with the real reverse process $q(x_t|x_{t+1}, x_0)||$.
  - This part of the the variational lower bound loss $L_{VLB}$ is important.

# Diffusion Loss



- The diffusion loss is:
  - $L_t = D_{KL}\big(q(x_t|x_{t+1}, x_0)||p_\theta(x_t|x_{t+1})\big)$ for $1 \leq t \leq T-1$.
- We know that:
  - $q(x_t|x_0) = \mathcal{N}(\sqrt{\bar{a}_t}x_0, (1-\bar{a}_t)\boldsymbol{I})$ with $\bar{a}_t = \prod_{i=1}^{t} \alpha_i$.
  - $q(x_t|x_{t-1}) = \mathcal{N}\big(\sqrt{a_t}x_{t-1}, 1-a_t\boldsymbol{I}\big)$ with $a_t = 1-\beta_t$.
- Using the Bayes' rule, it can be shown that:
  - $q(x_t|x_t, x_0) = \mathcal{N}\big(\tilde{\mu}(x_t, x_0), \tilde{\beta}_t\boldsymbol{I}\big)$ for some $\tilde{\mu}$ and $\tilde{\beta}$.
- We also assume that:
  - $p(x_{t-1}|x_t) = \mathcal{N}\big(\mu_\theta\,(x_t), \tilde{\beta}_t\boldsymbol{I}\big)$.
  - $\mu_\theta\,(\cdot)$ corresponds to a neural network.
- Based on $\mu_\theta\,(x_t)$ and $\tilde{\mu}(x_t, x_0)$, the **diffusion** loss becomes to **minimize** their different while assuming the same variance $\tilde{\beta}_t$.

# Diffusion Loss (Cont.)



- The term $q(x_t|x_t, x_0) = \mathcal{N}\left(\tilde{\mu}(x_t, x_0), \tilde{\beta}_t \boldsymbol{I}\right)$ can be further worked by considering $\tilde{\mu}(x_t, x_0)$ to that:
  - $\tilde{\mu}_t = \frac{1}{\sqrt{a_t}}\left(x_t - \frac{1-a_t}{\sqrt{1-\bar{a}_t}}\epsilon_t\right)$.
  - $\epsilon_t$ now corresponds to the noise added to $x_0$ and resulted in $x_t$.
- We can now our model as:
  - $\mu_\theta(x_t, t) = \frac{1}{\sqrt{a_t}}\left(x_t - \frac{1-a_t}{\sqrt{1-\bar{a}_t}}\epsilon_\theta(x_t, \tau)\right)$.
  - $\epsilon_\theta(x_t, t)$ corresponds to the neural network model of the noise $\epsilon_t$.
- With this model and parametrization, minimizing the **diffusion** loss $L_t$ refers to predicting the noise $\epsilon_t$ using the neural network model $\epsilon_\theta(\cdot)$and then removing it from $x_t$.

# Diffusion Loss (Cont.)

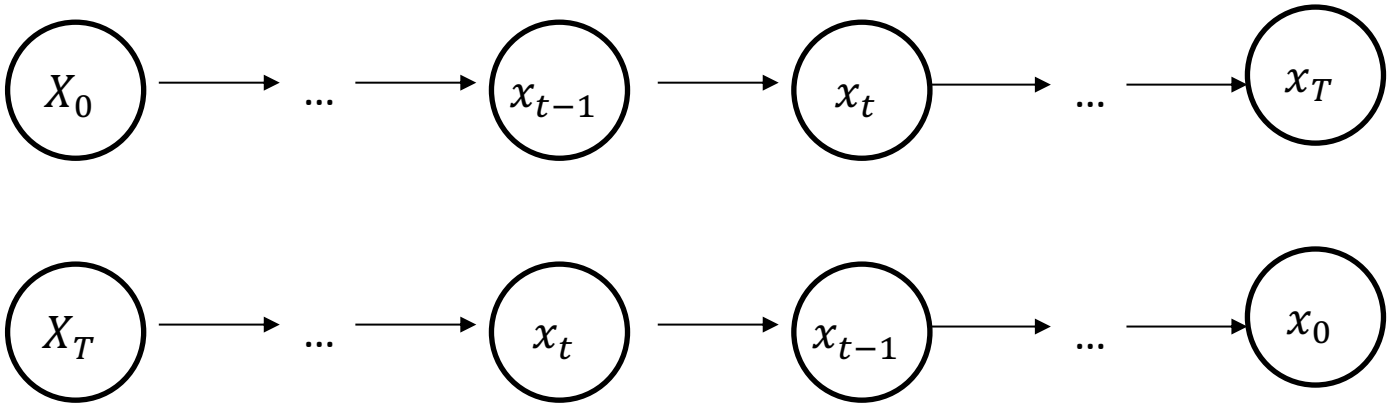


- Finally, we can write the **diffusion** loss $L_t$ as:
  - $L_t = \mathbb{E}_{x_0,\epsilon}[C_1 \cdot \|\tilde{\mu}(x_t, x_0) - \mu_\theta(x_t, t)\|^2] = \mathbb{E}_{x_0,\epsilon}[C_2 \cdot \|\epsilon_t - \epsilon_\theta(x_t, t)\|^2] = \mathbb{E}_{x_0,\epsilon}\left[C_2 \cdot \left\|\epsilon_t - \epsilon_\theta(\sqrt{\bar{a}_t}x_0 + \sqrt{1-\bar{a}_t}\epsilon_t, t)\right\|^2\right]$.
  - With $x_t = \sqrt{\bar{a}_t}x_0 + \sqrt{1-\bar{a}_t}\epsilon_t$.
  - $C_1$, $C_2$ are positive constants that we can ignore.
- So, **optimizing** the diffusion loss is given as:
  - Sample $x_0$.
  - Sample a time step $t, \dots, T$.
  - Sample noise $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.
  - Generate noisy $x_t = \sqrt{\bar{a}_t}x_0 + \sqrt{1-\bar{a}_t}\epsilon_t$.
  - Compute loss $L_t = \|\epsilon_t - \epsilon_\theta(x_t, t)\|^2$.
  - Perform gradient descent on $L_t$.
- Training takes place until convergence.

# Diffusion Loss Interpretation



- During the forward process, we can randomly pick up a time step and add noise $\epsilon_t$ to our noisy sample (image).
- In the reverse process, we take a noisy sample $x_t$ and denoise it using the neural network $\epsilon_\theta(\cdot)$. However, the neural network does not directly remove the noise from the sample. Instead, it produces the noise that we need to remove from the sample.



# Sample Generation



- Diffusion models are commonly used to generate images, videos and graphs, among other things.
- The most common architecture for images is to rely on a U-Net* style architecture.

*Generated with DiffusionBee.*

*Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18 (pp. 234-241). Springer International Publishing.

# Sample Generation (Cont.)



- Diffusion models are commonly used to generate images, videos and graphs, among other things.
- The most common architecture for images is to rely on a U-Net* style architecture.

*Generated with DiffusionBee.*

*Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18 (pp. 234-241). Springer International Publishing.

# Comparison to other Generative Models



- 2 main metrics are used in the literature to evaluate the performance of generative models
- SOTA comparison regarding the Inception Score(IS) & Frechet Inception Distance(FID)
  - FID: Calculates the statistical difference between real and generated images' feature representations extracted from a pre-trained Inception-v3 model. A lower FID score suggests better quality and diversity of the generated images compared to the real ones.

  -> lower FID is better.
  - IS: Measures the quality by classification confidence and measure the diversity. Additionally measures image diversity by the spread of predicted classes across generated images.

  -> higher IS is better.

# Comparison to other Generative Models (Cont.)



- Results on standard benchmarks for image generation.

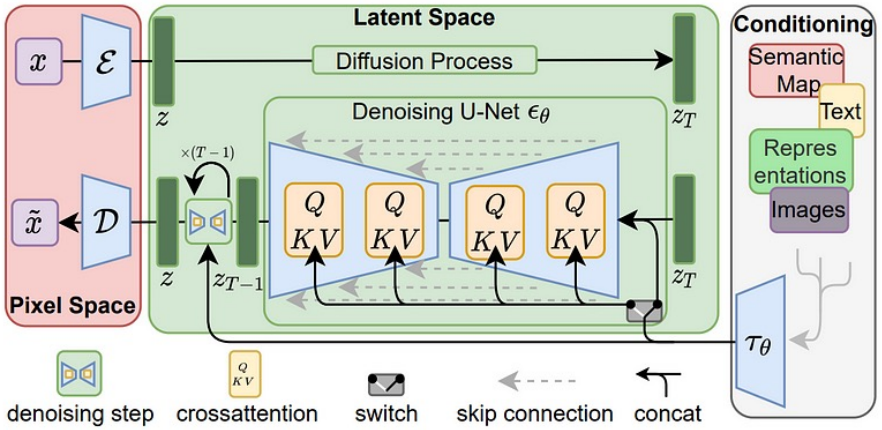
| | Model | IS↑ | FID↓ |
|---|---|---|---|
| Diffusion models | Denoising Diffusion GAN (ours), T=4 | 9.63 | 3.75 |
| | DDPM (Ho et al., 2020) | 9.46 | 3.21 |
| | NCSN (Song & Ermon, 2019) | 8.87 | 25.3 |
| | Adversarial DSM (Jolicoeur-Martineau et al., 2021b) | - | 6.10 |
| | Likelihood SDE (Song et al., 2021b) | - | 2.87 |
| | Score SDE (VE) (Song et al., 2021c) | 9.89 | 2.20 |
| | Score SDE (VP) (Song et al., 2021c) | 9.68 | 2.41 |
| | Probability Flow (VP) (Song et al., 2021c) | 9.83 | 3.08 |
| | LSGM (Vahdat et al., 2021) | 9.87 | 2.10 |
| | DDIM, T=50 (Song et al., 2021a) | 8.78 | 4.67 |
| | FastDDPM, T=50 (Kong & Ping, 2021) | 8.98 | 3.41 |
| | Recovery EBM (Gao et al., 2021) | 8.30 | 9.58 |
| | Improved DDPM (Nichol & Dhariwal, 2021) | - | 2.90 |
| | VDM (Kingma et al., 2021) | - | 4.00 |
| | UDM (Kim et al., 2021) | 10.1 | 2.33 |
| | D3PMs (Austin et al., 2021) | 8.56 | 7.34 |
| | Gotta Go Fast (Jolicoeur-Martineau et al., 2021a) | - | 2.44 |
| | DDPM Distillation (Luhman & Luhman, 2021) | 8.36 | 9.36 |
| GANs | SNGAN (Miyato et al., 2018) | 8.22 | 21.7 |
| | SNGAN+DGflow (Ansari et al., 2021) | 9.35 | 9.62 |
| | AutoGAN (Gong et al., 2019) | 8.60 | 12.4 |
| | TransGAN (Jiang et al., 2021) | 9.02 | 9.26 |
| | StyleGAN2 w/o ADA (Karras et al., 2020a) | 9.18 | 8.32 |
| | StyleGAN2 w/ ADA (Karras et al., 2020a) | 9.83 | 2.92 |
| | StyleGAN2 w/ Diffaug (Zhao et al., 2020) | 9.40 | 5.79 |
| VAEs etc. | Glow (Kingma & Dhariwal, 2018) | 3.92 | 48.9 |
| | PixelCNN (Oord et al., 2016b) | 4.60 | 65.9 |
| | NVAE (Vahdat & Kautz, 2020) | 7.18 | 23.5 |
| | IGEBM (Du & Mordatch, 2019) | 6.02 | 40.6 |
| | VAEBM (Xiao et al., 2021) | 8.43 | 12.2 |

Source: Xiao, Z., Kreis, K., & Vahdat, A. (2021). Tackling the generative learning trilemma with denoising diffusion gans. *arXiv preprint arXiv:2112.07804*

# Latent Diffusion Models



- Latent diffusion modes (LDMs):
  - Pixel space diffusion models have huge drawbacks:
    - Sampling speed is very slow
    - Training on high-resolution images is resource intensive
  - LDMs* fix these issues with a simple idea:
    1. Encode everything into a latent space with the help of a pretrained VQGAN encoder $\epsilon$.
    2. perform diffusion in the latent space of the VQGAN.
    → diffusion is applied in lower dimension
    3. decode back to pixel space with the VQGAN decoder $D$




* Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 10684-10695).

# Denoising Diffusion Implicit Models



- Latent diffusion modes (LDMs):
  - Pixel space diffusion models have huge drawbacks
  - LDMs fix the computation issue with encoding to a latent space

  Sampling speed is still problematic!
  - Change sampling strategy, e.g., deterministic sampling (DDIM)*
  - Trade sampling speed for quality with Denoising Diffusion Implicit Models (DDIM):
    - Using only $T = 100$ steps.
    - DDIM reaches only slightly.
    - Higher FID score than DDPM.
    - When using $T = 1000$.

| | CIFAR10 ($32 \times 32$) | | | | |
|---|---|---|---|---|---|
| | 10 | 20 | 50 | 100 | 1000 |
| DDIM | **13.36** | **6.84** | **4.67** | **4.16** | 4.04 |
| | 14.04 | 7.11 | 4.77 | 4.25 | 4.09 |
| | 16.66 | 8.35 | 5.25 | 4.46 | 4.29 |
| | 41.07 | 18.36 | 8.01 | 5.78 | 4.73 |
| DDPM | 367.43 | 133.37 | 32.72 | 9.99 | **3.17** |

FID scores for both DDPMs and DDIMs on CIFAR10 data.

* Song, J., Meng, C., & Ermon, S. (2020). Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

# Conditional Sampling & Guidance



- We can condition* the sample generation on additional information, e.g., image generation conditioned on a sentence.
  - Example prompt: “a university class full of students writing an exam, Realistic, HD”.

*Generated with DiffusionBee.*

* Dhariwal, Prafulla, and Alexander Nichol. "Diffusion models beat gans on image synthesis." *Advances in neural information processing systems* 34 (2021): 8780-8794.

# Study Material

***Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)***


- *Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." Advances in neural information processing systems 33 (2020): 6840-6851.*
- *Song, Yang, et al. "Score-based generative modeling through stochastic differential equations." arXiv preprint arXiv:2011.13456 (2020).*
- *Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., & Ganguli, S. (2015, June). Deep unsupervised learning using nonequilibrium thermodynamics. In International conference on machine learning (pp. 2256-2265). PMLR.*
- *Song, J., Meng, C., & Ermon, S. (2020). Denoising diffusion implicit models. arXiv preprint arXiv:2010.02502.*

# Next Lecture



*Model Compression*