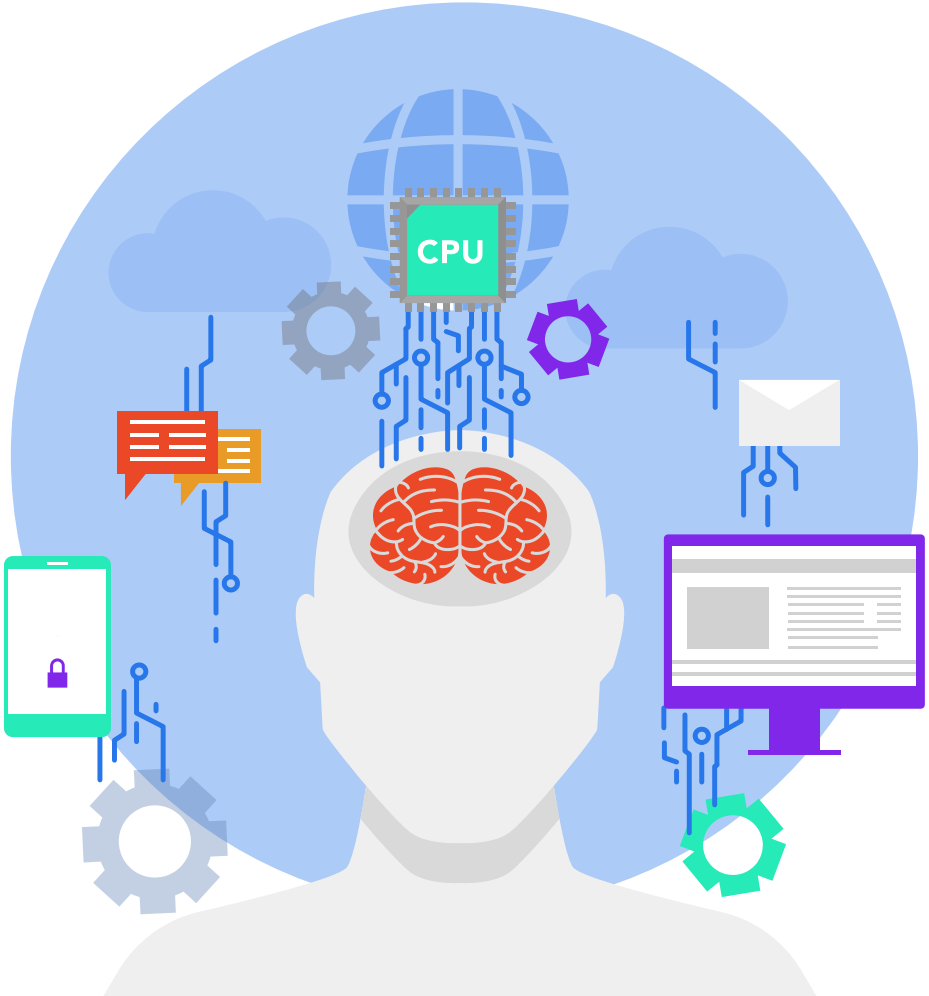
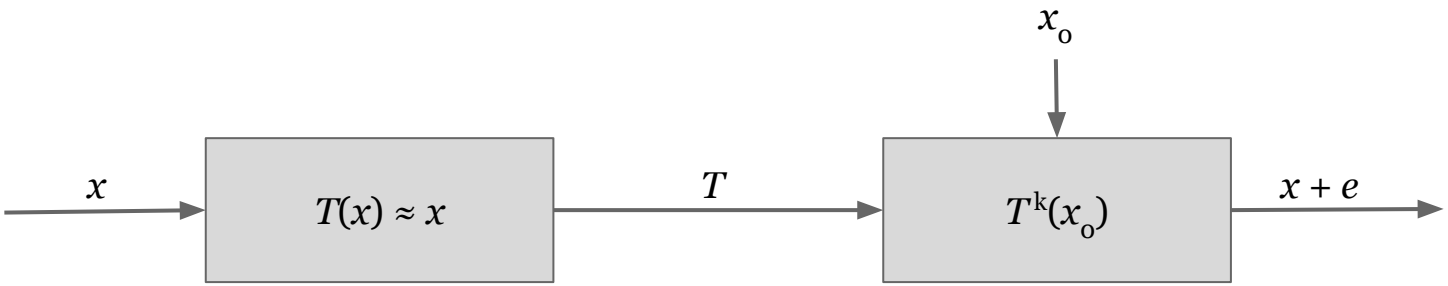


Advanced Image Processing

Alexander Kovalenko

Fractal Image Compression

- A very good blogpost: https://pvigier.github.io/2018/05/14/fractal-image-compression.html
- O. Zmeskal et al, Entropy of fractal systems, 2013, https://doi.org/10.1016/j.camwa.2013.01.017
- Y. Fisher, Fractal Image Compression, 1994, https://doi.org/10.1142/S0218348X94000442



$$x = \lim_{k \to \infty} T^k (x_0) + e$$

Fractal Image Compression

Original Lena image

Self-similar portions of the image

Fractal Image Compression

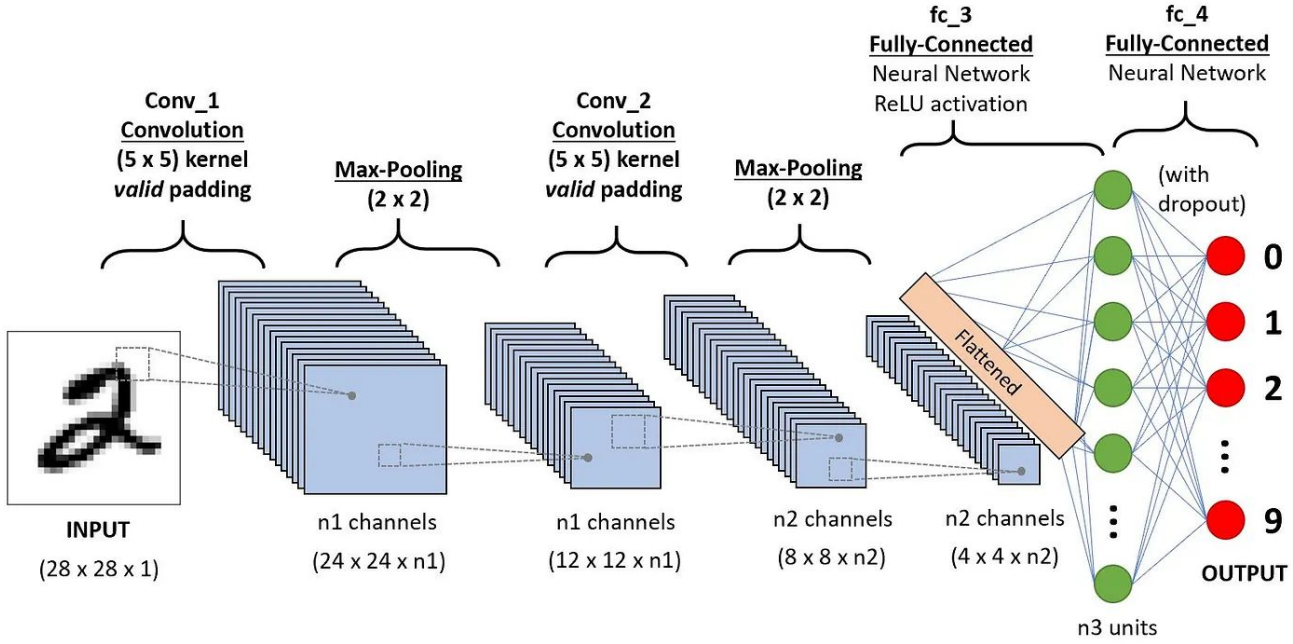
Original Lena image (184,320 bytes)

FIF-max. quality (30,368)
comp. ratio: **6.07:1**

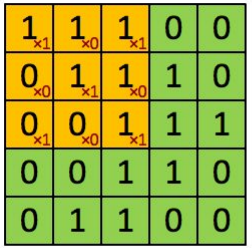
JPEG-min. quality (8,247)
comp. ratio: **22.35:1**

FIF-min. quality (3,924)
comp. ratio: **46.97:1**

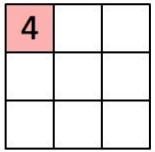
A Quick Recap on CNNs



Y LeCun, L Bottou, Y Bengio, P Haffner.
Proceedings of the IEEE 86 (11), 1998

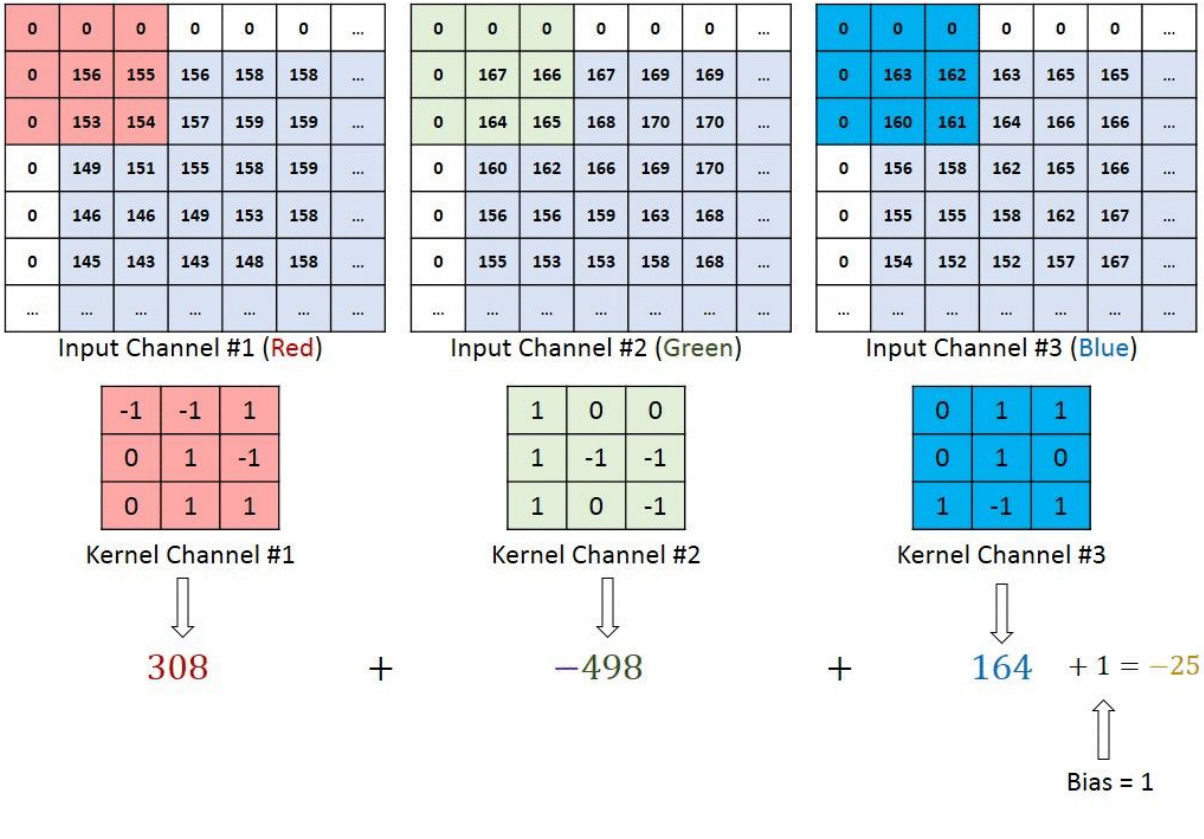


Image



Convolved Feature

Quick recap on CNNs



Generative vs. Discriminative

Generative Models

- Attempt to model the underlying distribution of the data
- Learn how the data was generated

- Can be used to generate new data points similar to the original data
- Useful when there is a need to generate new data points

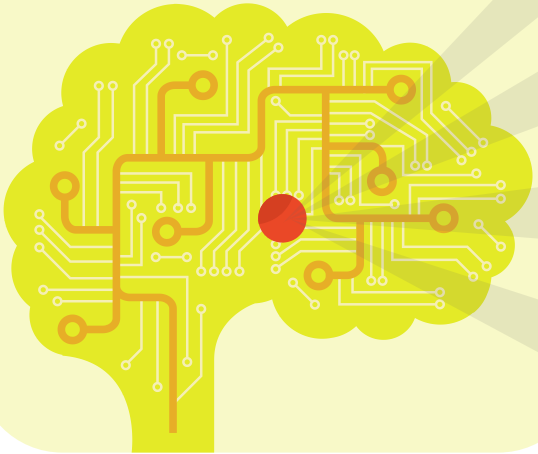
Vs

Discriminative Models

- Focus on learning the decision boundary
- Do not try to model the distribution of the data
- Directly learn the mapping from input to output labels
- Useful when the goal is to classify new data points

Generative Deep Learning Models

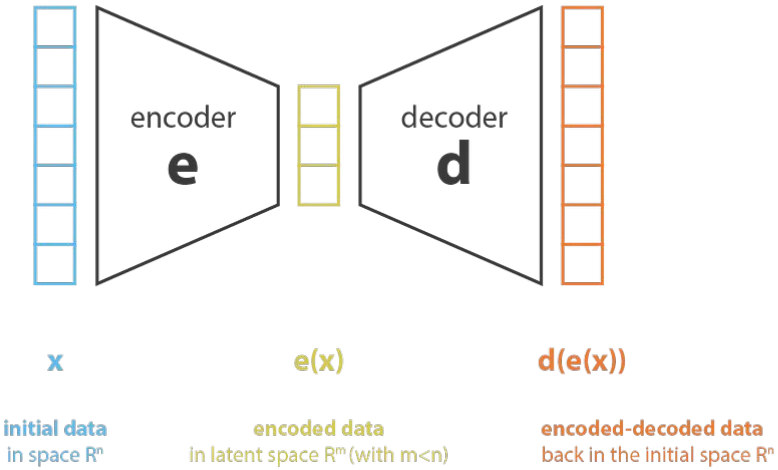
GNN for Image Processing



- **VAE** — CVAE, VampPrior VAE...
- **GAN** — DCGAN, CGAN, CycleGAN...
- **Transformer** — TransGAN, MaskGIT...
- **Diffusion** — DiffAE, stable Diff., Paella...

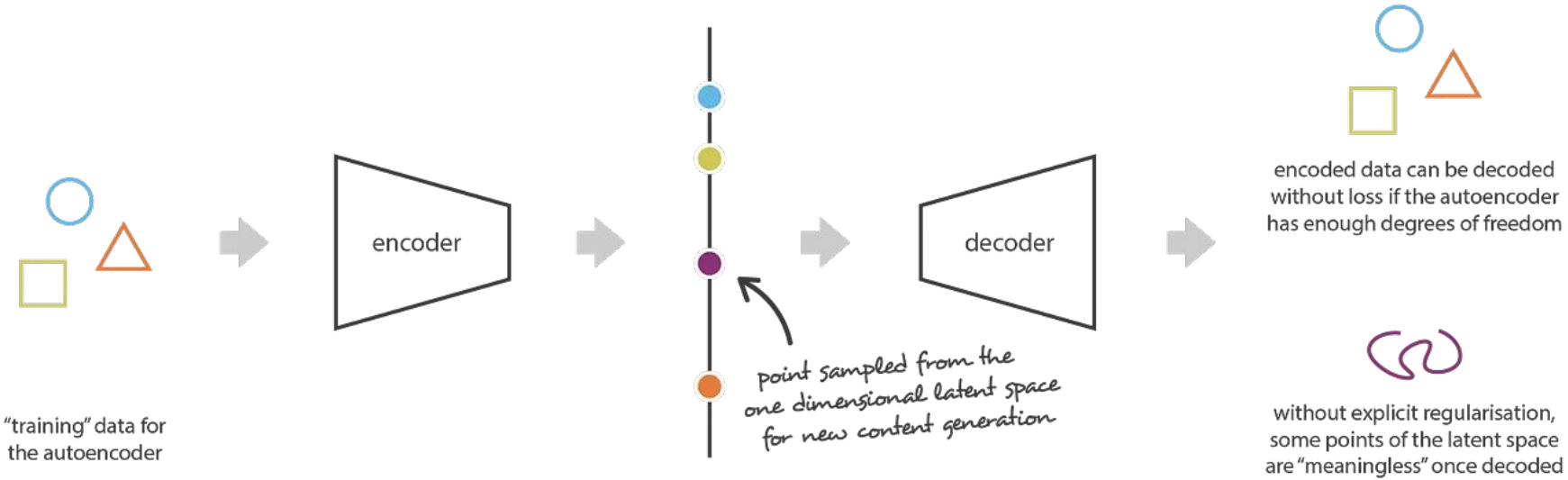
Autoencoders vs. Variational Autoencoders

Encoder-Decoder Concept:



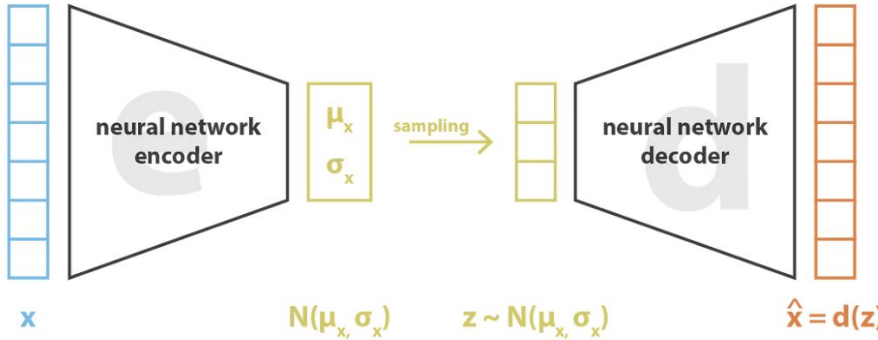
https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Autoencoders vs. Variational Autoencoders



https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Autoencoders vs. Variational Autoencoders



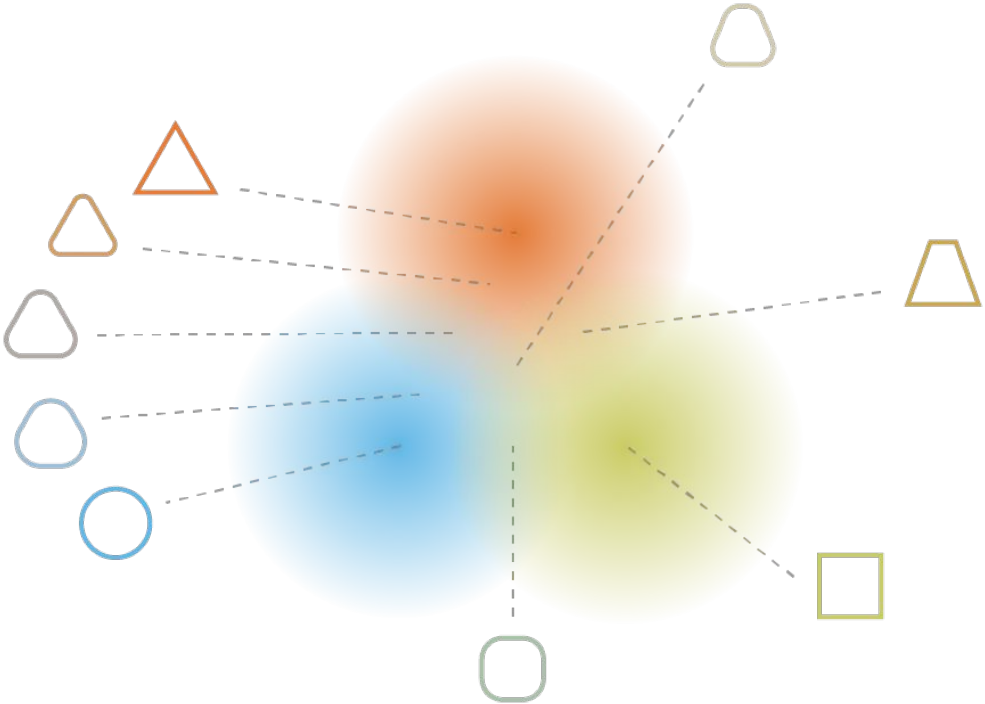
$$\text{loss} = ||x - \hat{x}||^2 + KL[N(\mu_x, \sigma_x), N(0, I)] = ||x - d(z)||^2 + KL[N(\mu_x, \sigma_x), N(0, I)]$$

https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Autoencoders vs. Variational Autoencoders

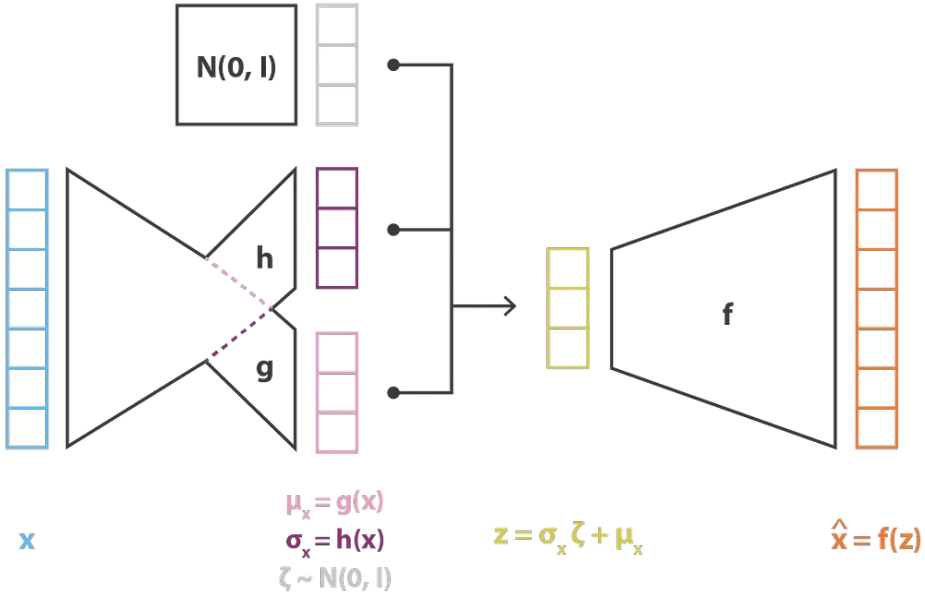
https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Autoencoders vs. Variational Autoencoders



https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

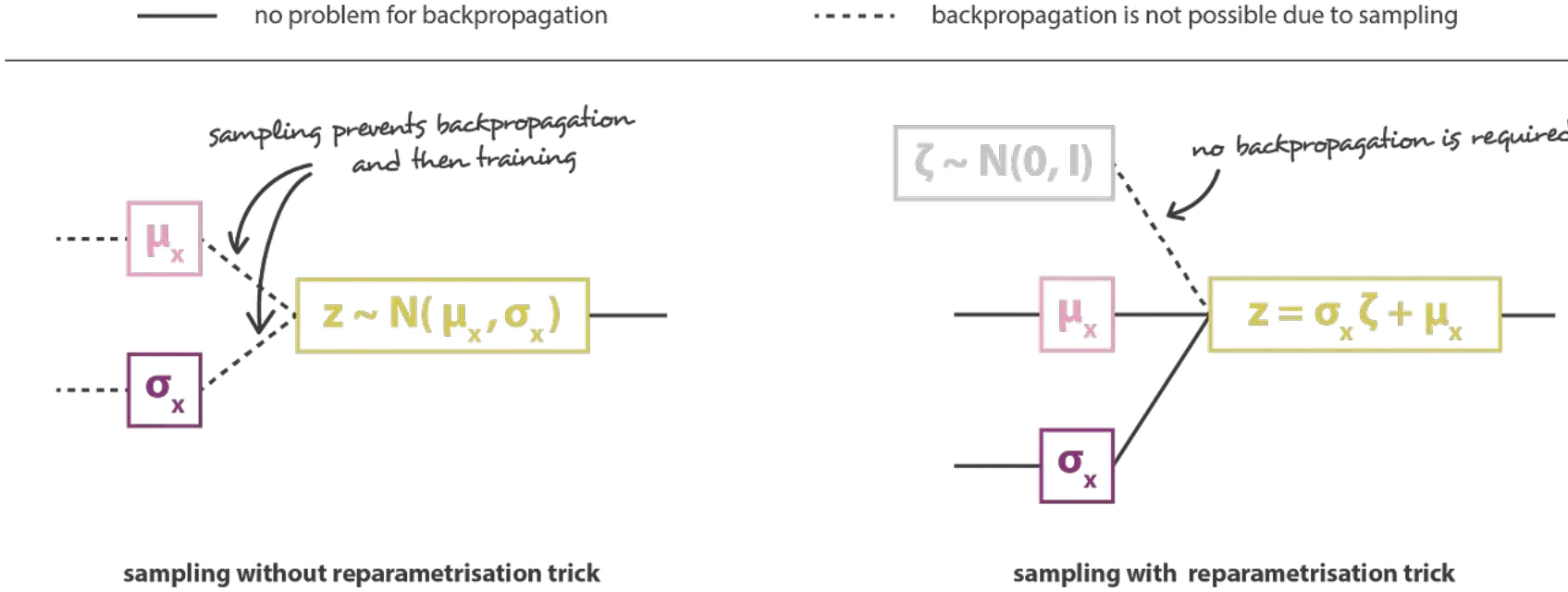
Autoencoders vs. Variational Autoencoders



$$\text{loss} = C \,||\, x - \hat{x} \,||^2 + KL[\, N(\mu_x, \sigma_x), N(0, I) \,] = C \,||\, x - f(z) \,||^2 + KL[\, N(g(x), h(x)), N(0, I) \,]$$

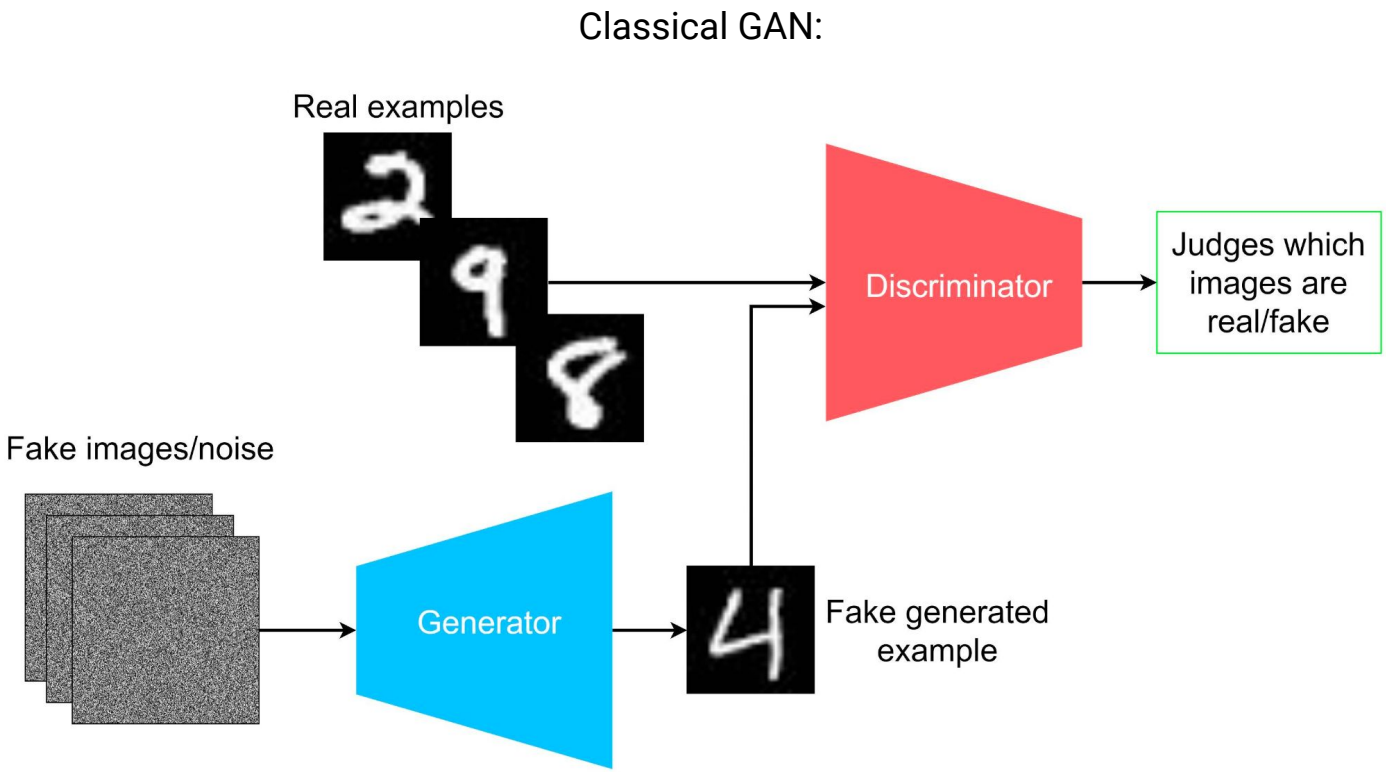
https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Autoencoders vs. Variational Autoencoders



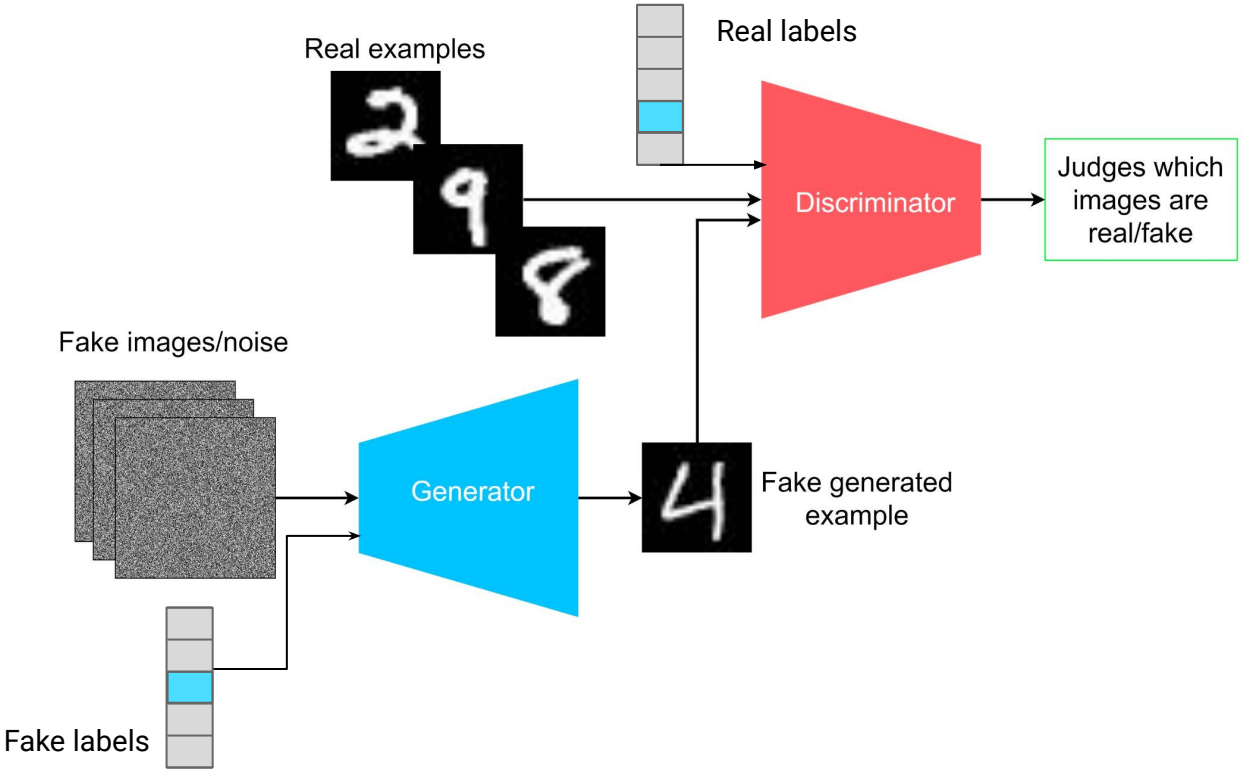
https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Generative Adversarial Networks

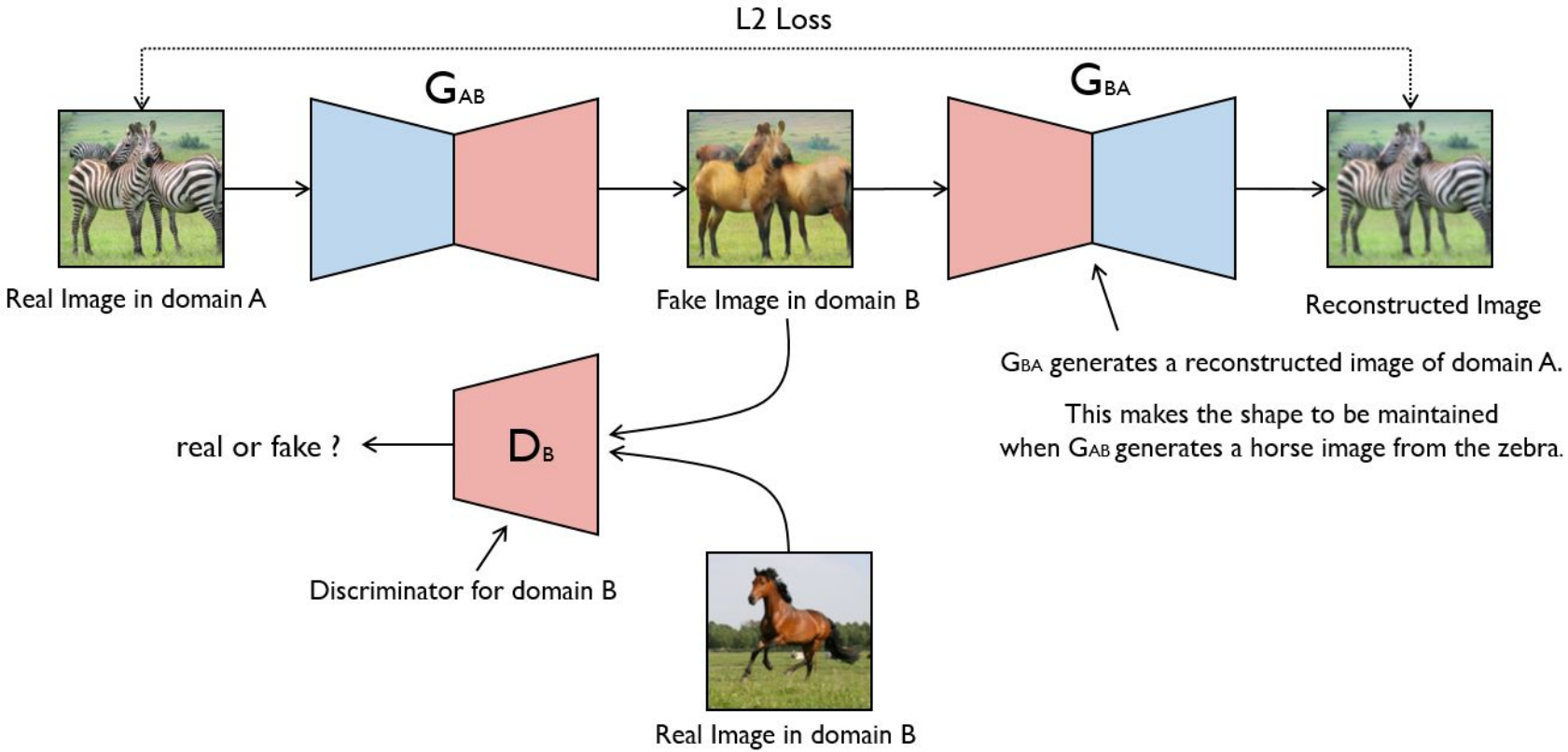


Generative Adversarial Networks

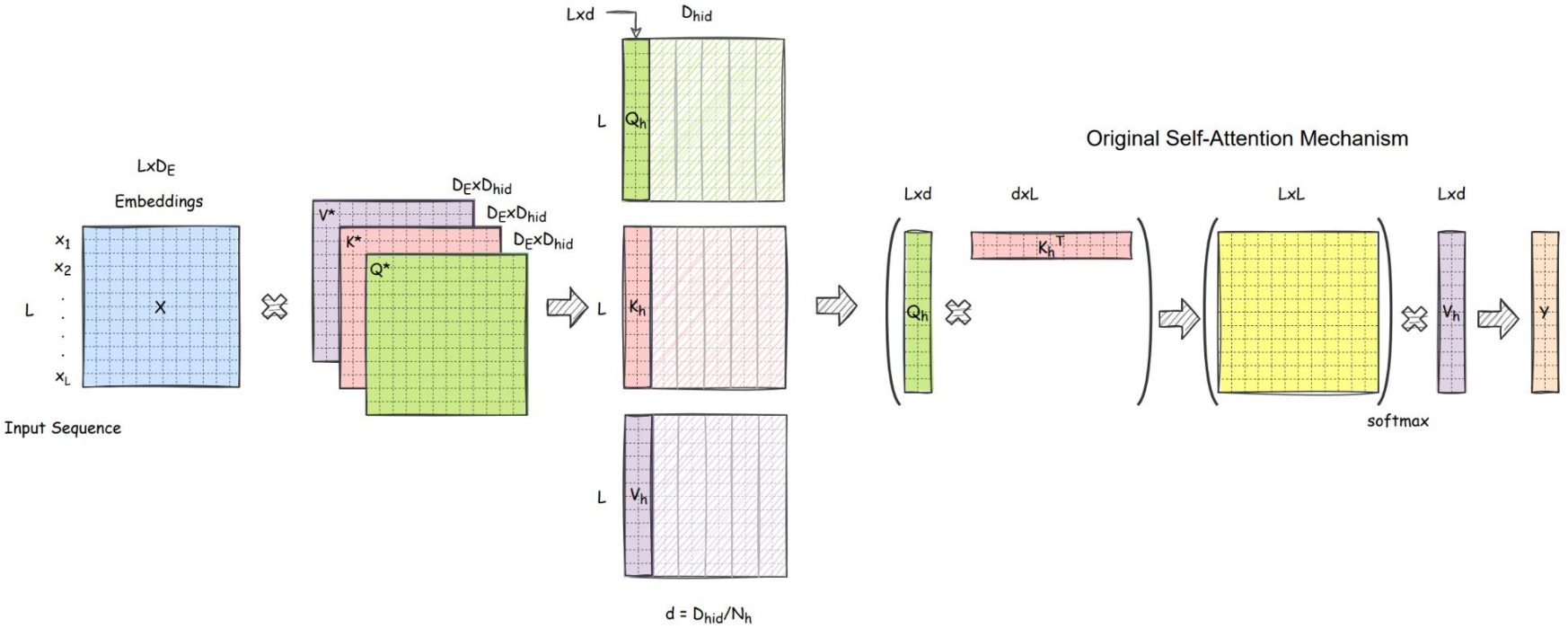
Conditional GAN:



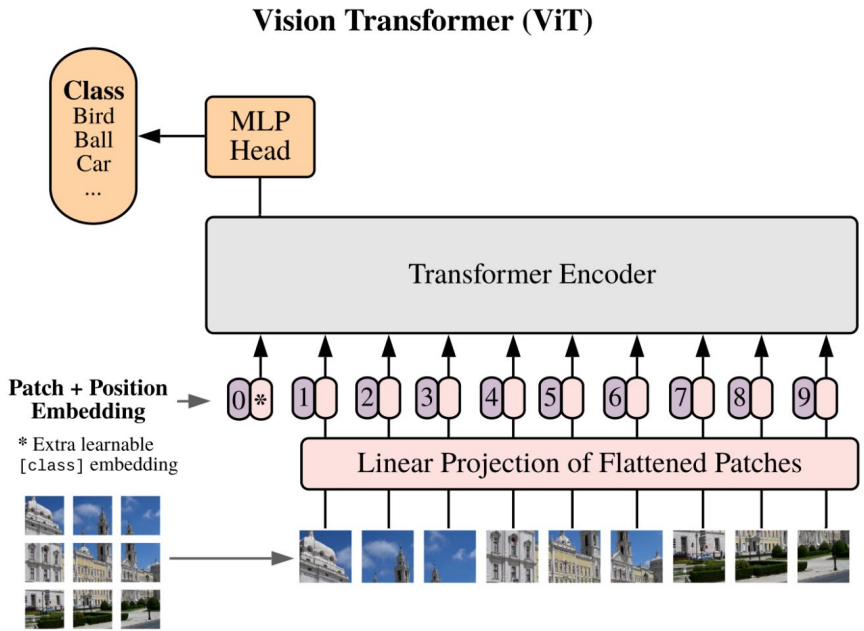
Generative Adversarial Networks

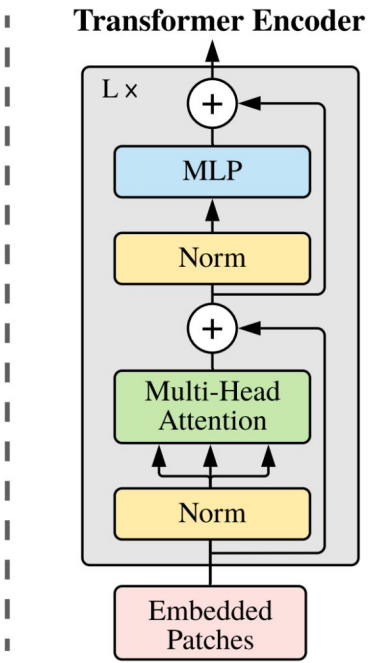


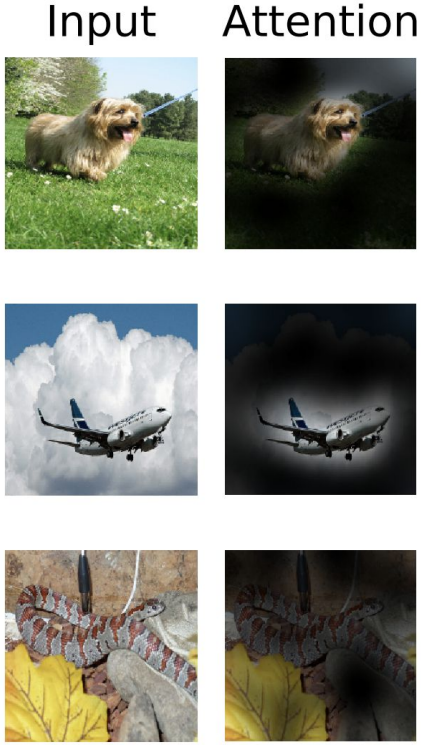
Vision Transformer



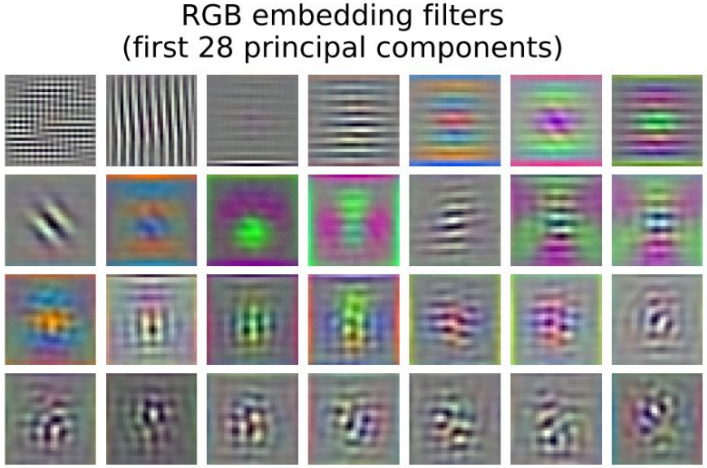
Vision Transformer

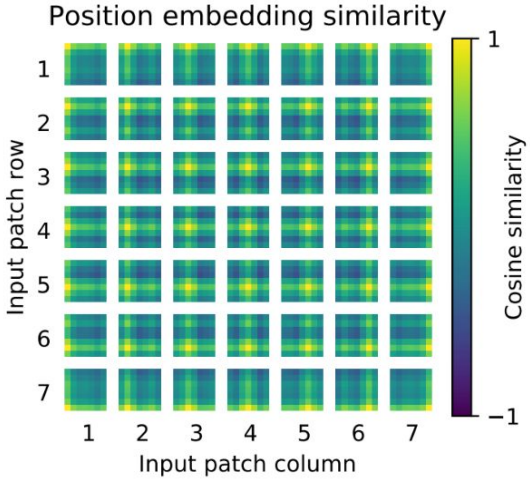


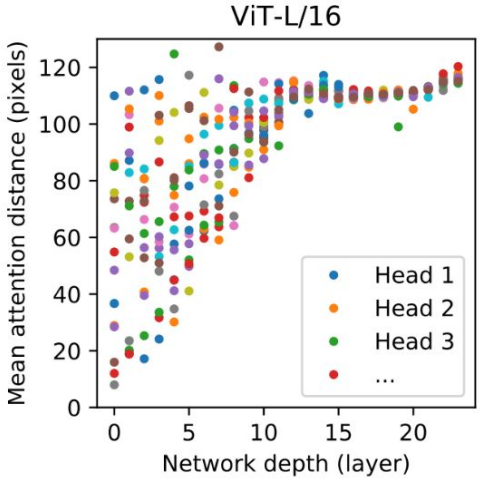




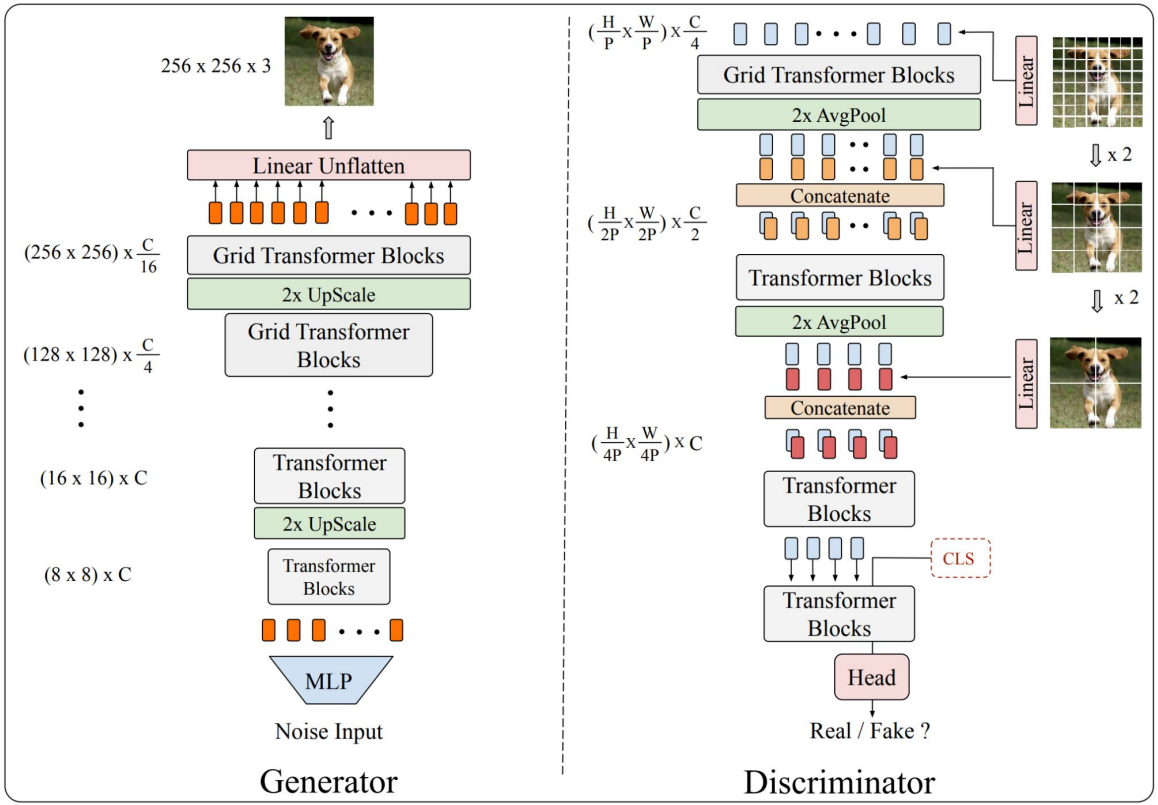
Vision Transformer





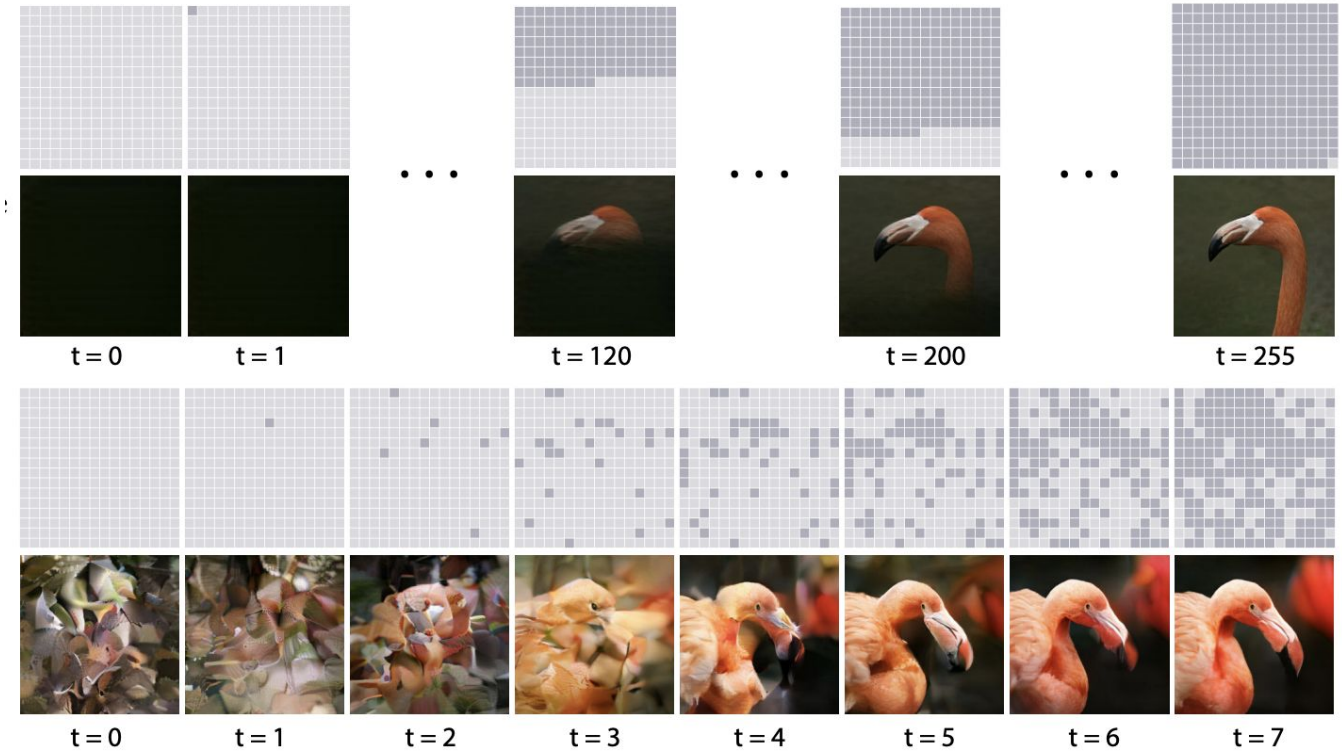


TransGAN



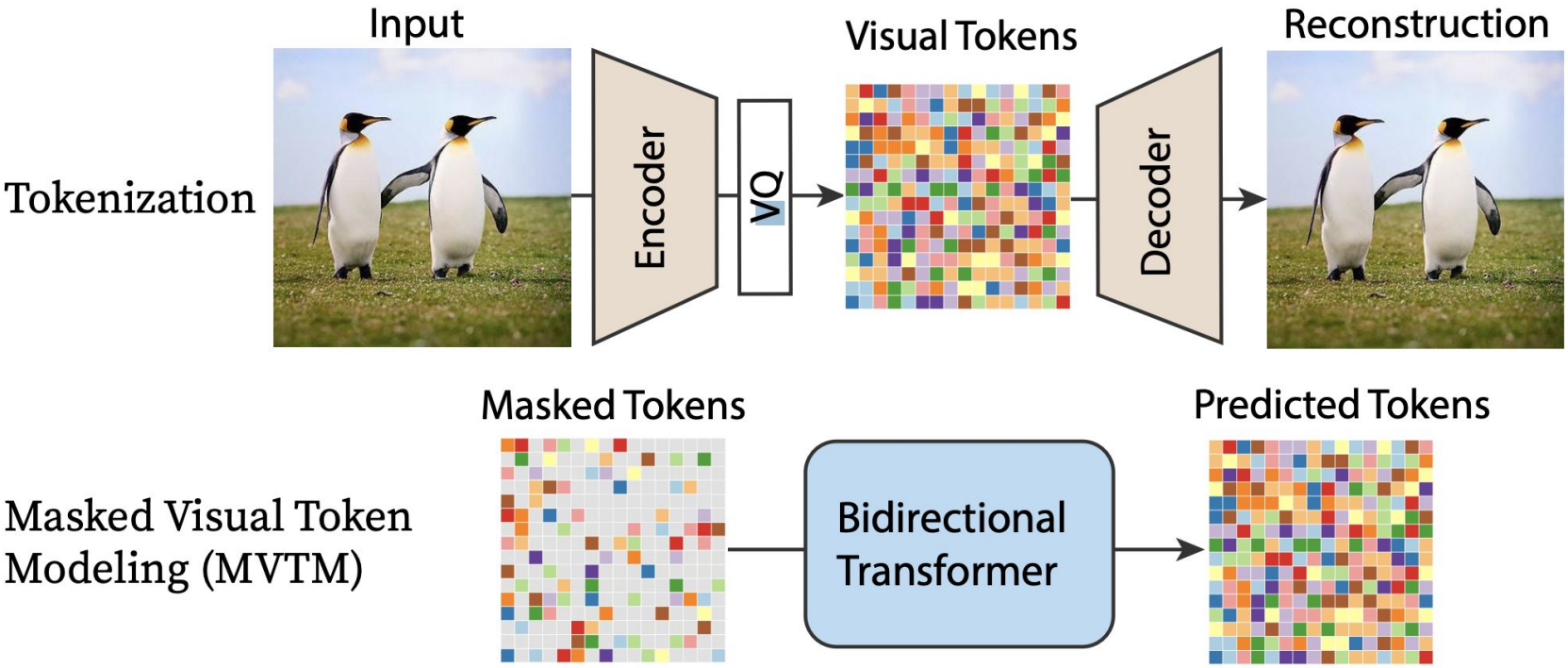
https://arxiv.org/pdf/2102.07074.pdf

MaskGIT: Masked Generative Image Transformer



https://arxiv.org/pdf/2202.04200.pdf

MaskGIT: Masked Generative Image Transformer



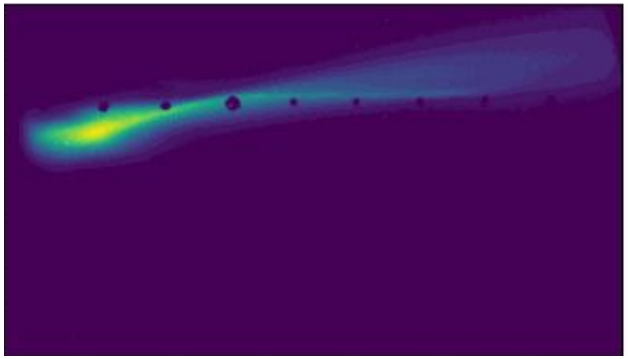
https://arxiv.org/pdf/2202.04200.pdf

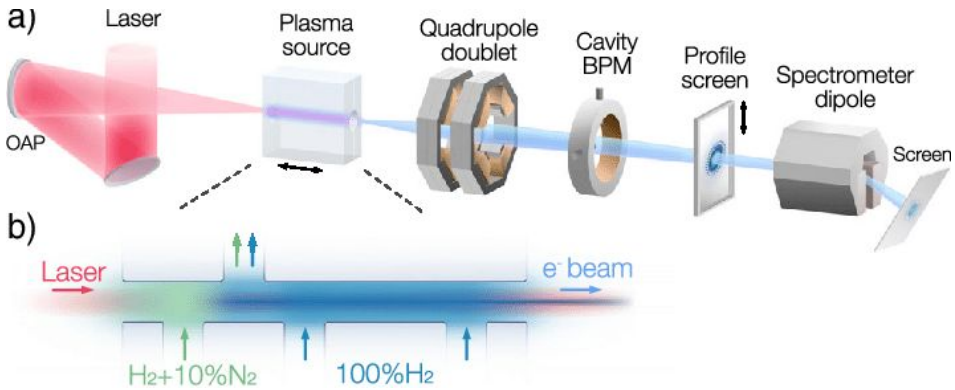
Diffusion Models

Key papers:

- J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics, **2015**, https://arxiv.org/pdf/1503.03585.pdf (https://github.com/Sohl-Dickstein/Diffusion-Probabilistic-Models)
- Y. Song, S. Ermon, Generative Modeling by Estimating Gradients of the Data Distribution, **2020**, https://arxiv.org/pdf/1907.05600.pdf
- J. Ho, A. Jain, P. Abbeel, Denoising Diffusion Probabilistic Models, **2020**, https://arxiv.org/pdf/2006.11239.pdf
- A. Nichol, P. Dhariwal, Improved Denoising Diffusion Probabilistic Models, 2021, https://arxiv.org/pdf/2102.09672.pdf (https://github.com/openai/improved-diffusion)
- P. Dhariwal, A. Nichol, Diffusion Models Beat GANs on Image Synthesis, https://arxiv.org/pdf/2105.05233.pdf, (https://github.com/openai/guided-diffusion)

Commercial Break!

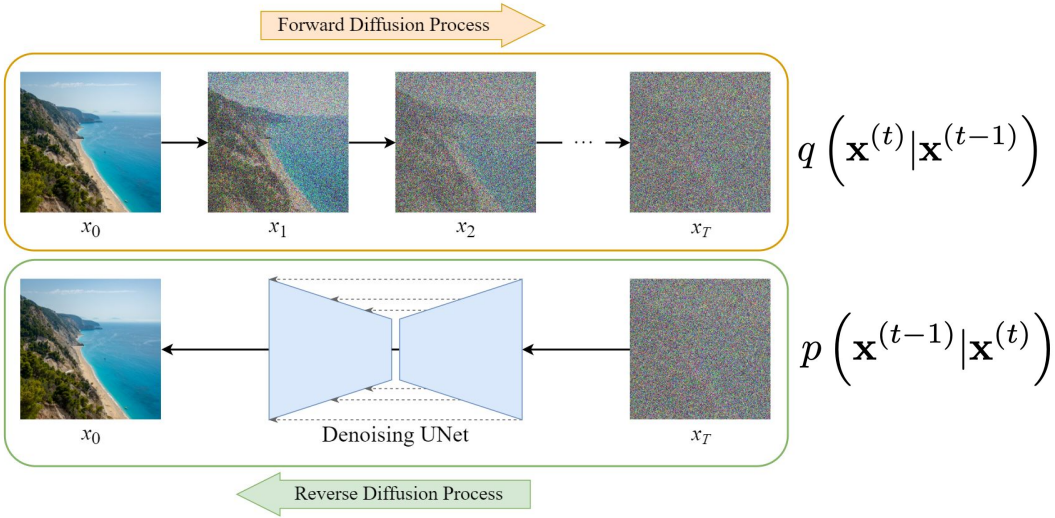




Diffusion Models

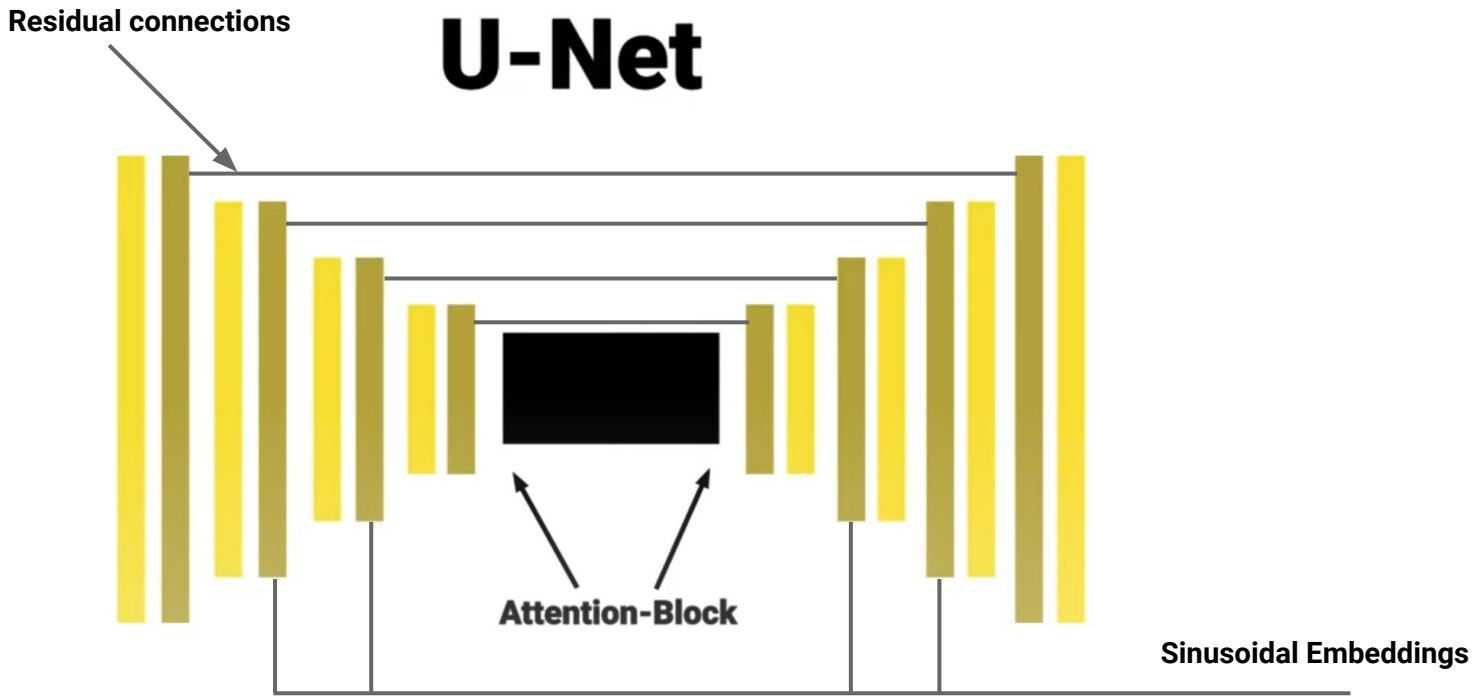
- To create samples using score-based models, we must first take into account a **diffusion process that gradually transforms** the data into **random noise**. By reversing this diffusion process, we can generate samples with scores.

- A diffusion process can be likened to the movement of a particle in a fluid, akin to **Brownian motion**. The particle moves unpredictably due to chance collisions with other particles along its path.



https://medium.com/@steinsfu/diffusion-model-clearly-explained-cd331bd41166

Diffusion Models



Forward Diffusion

$$q\left(\mathbf{x}^{(0\cdots T)}\right) = q\left(\mathbf{x}^{(0)}\right)\prod_{t=1}^{T} q\left(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}\right)$$

Data distribution at steps 0…T

$$q\left(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}\right) = T_\pi\left(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)};\beta_t\right)$$

Markov diffusion kernel

Diffusion rate

Gaussian/Binomial diffusion into a distribution with identity-covariance

Forward Diffusion

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t, \sqrt{1-\beta}x_{t-1}, \beta_t I)$$

Forward Diffusion

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t, \sqrt{1-\beta}x_{t-1}, \beta_t I)$$

Variance

Mean

Normal distribution

x_0 x_1

Forward Diffusion

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t, \sqrt{1-\beta}x_{t-1}, \beta_t I)$$

$$\beta = 0.001...0.02$$

$$\alpha_t = \prod_{s=1}^{t} \alpha_s$$

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t, \sqrt{1-\beta_t}x_{t-1}, \beta_t I) =$$
$$= \sqrt{1-b}x_{t-1} + \sqrt{\beta_t}\epsilon = \sqrt{\alpha_t}x_{t-1} + \sqrt{1-\alpha_t}\epsilon$$

Forward Diffusion

$$q(x_t|x_{t-2}) = \sqrt{\alpha_t \alpha_{t-1}} x_{t-2} + \sqrt{1 - \alpha_t \alpha_{t-1}} \epsilon$$

$$q(x_t|x_0) = \sqrt{\alpha_t \alpha_{t-1} \dots \alpha_1 \alpha_0} x_0 + \sqrt{1 - \alpha_t \alpha_{t-1} \dots \alpha_1 \alpha_0} \epsilon =$$

$$= \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon$$

Reverse Diffusion

$$p\left(\mathbf{x}^{(0\cdots T)}\right) = p\left(\mathbf{x}^{(T)}\right) \prod_{t=1}^{T} p\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}\right)$$

Data distribution at steps 0…T

$$q\left(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}\right) = T_\pi\left(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)};\beta_t\right)$$

Markov diffusion kernel

Diffusion rate

Gaussian/Binomial diffusion into a Gaussian/Binomial distribution with identity-covariance

Reverse Diffusion

$$p(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

Variance (fixed)

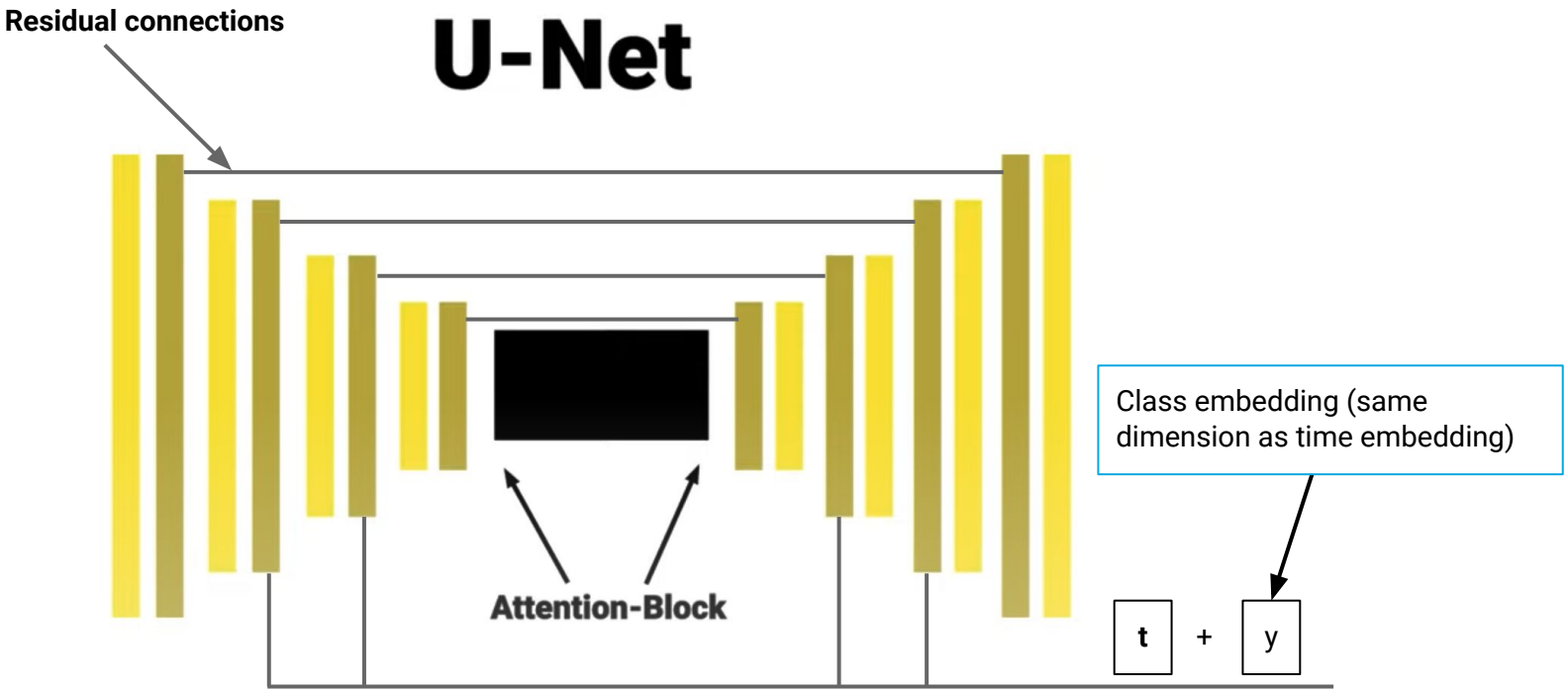
Mean

$$L = -\log(p_\theta(x_0))$$

The full math can be found in: https://arxiv.org/pdf/1503.03585.pdf, Appendix B, or in the blogpost: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/
However, the authors have finally given up on it and approximated the formula:

$$L_{simple} = \mathbb{E}_{t,x_0,\epsilon}[||\epsilon - \epsilon_\theta(x_t, t)||^2]$$

Guided Diffusion



Guided Diffusion

- J. Ho & T. Salimans, Classifier Free Diffusion Guidance, **2022**, https://arxiv.org/pdf/2207.12598.pdf

Given the differentiable discriminative model that estimates:

$$p(y \mid x)$$

we can obtain:

$$\nabla_x \log p(y \mid x)$$

Guided diffusion by scaling the conditioning term:

$$\nabla_x \log p_\gamma(x \mid y) = \nabla_x \log p(x) + \gamma \nabla_x \log p(y \mid x)$$

More of a good stuff here: https://sander.ai/2022/05/26/guidance.html#fn:cf

Guided Diffusion

$$p(y \mid x) = \frac{p(x \mid y) \cdot p(y)}{p(x)}$$

$$\implies \log p(y \mid x) = \log p(x \mid y) + \log p(y) - \log p(x)$$

$$\implies \nabla_x \log p(y \mid x) = \nabla_x \log p(x \mid y) - \nabla_x \log p(x).$$

Substitute into the previous formula for guidance:

$$\nabla_x \log p_\gamma(x \mid y) = \nabla_x \log p(x) + \gamma \left(\nabla_x \log p(x \mid y) - \nabla_x \log p(x) \right),$$

Or, equivalently:

$$\nabla_x \log p_\gamma(x \mid y) = (1 - \gamma) \nabla_x \log p(x) + \gamma \nabla_x \log p(x \mid y)$$

More of a good stuff here: https://sander.ai/2022/05/26/guidance.html#fn:cf

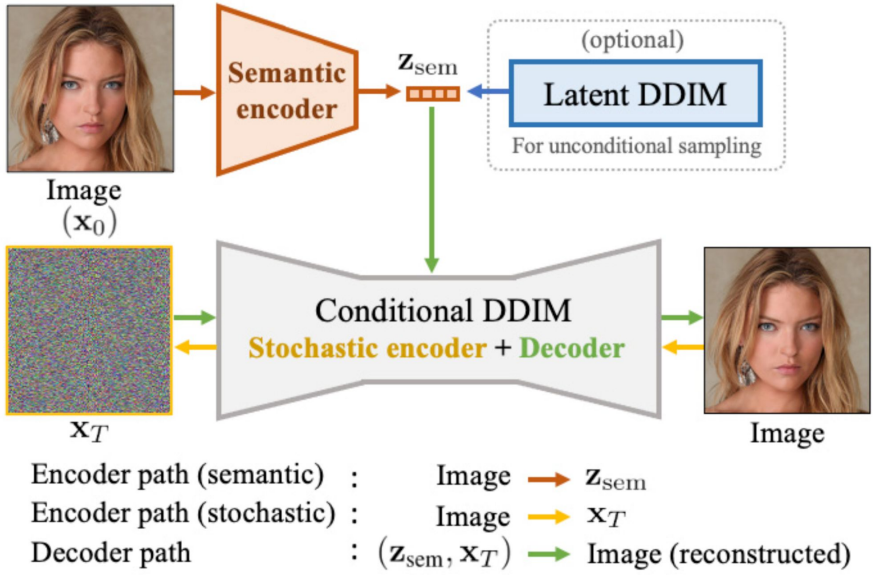
Guided Diffusion

γ = 1

γ = 3

Diffusion Autoencoders

- K. Preechakul, N. Chatthee, S. Wizadwongsa, S. Suwajanakorn, Diffusion Autoencoders: Toward a Meaningful and Decodable Representation, https://arxiv.org/abs/2111.15640, (https://github.com/phizaz/diffae)



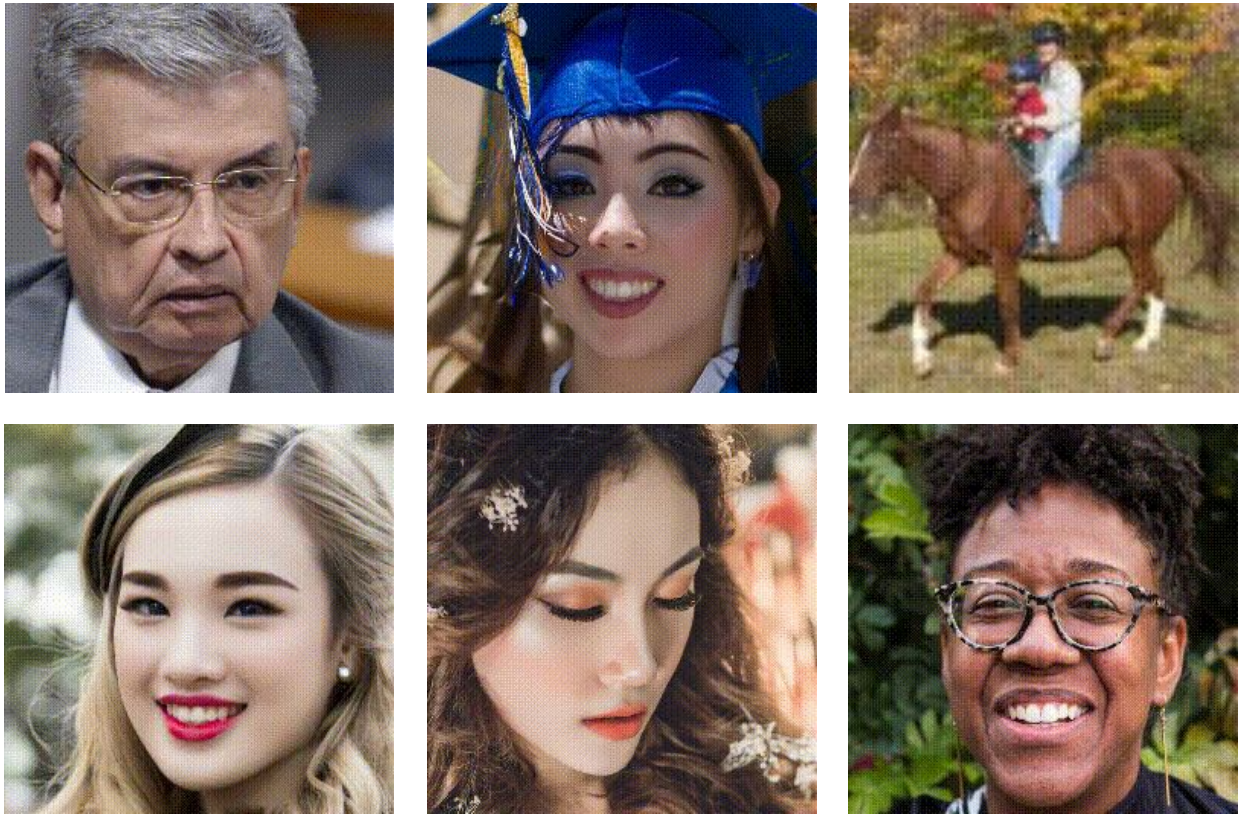
https://diff-ae.github.io/

Diffusion Autoencoders

https://diff-ae.github.io/

https://diff-ae.github.io/

Diffusion Autoencoders

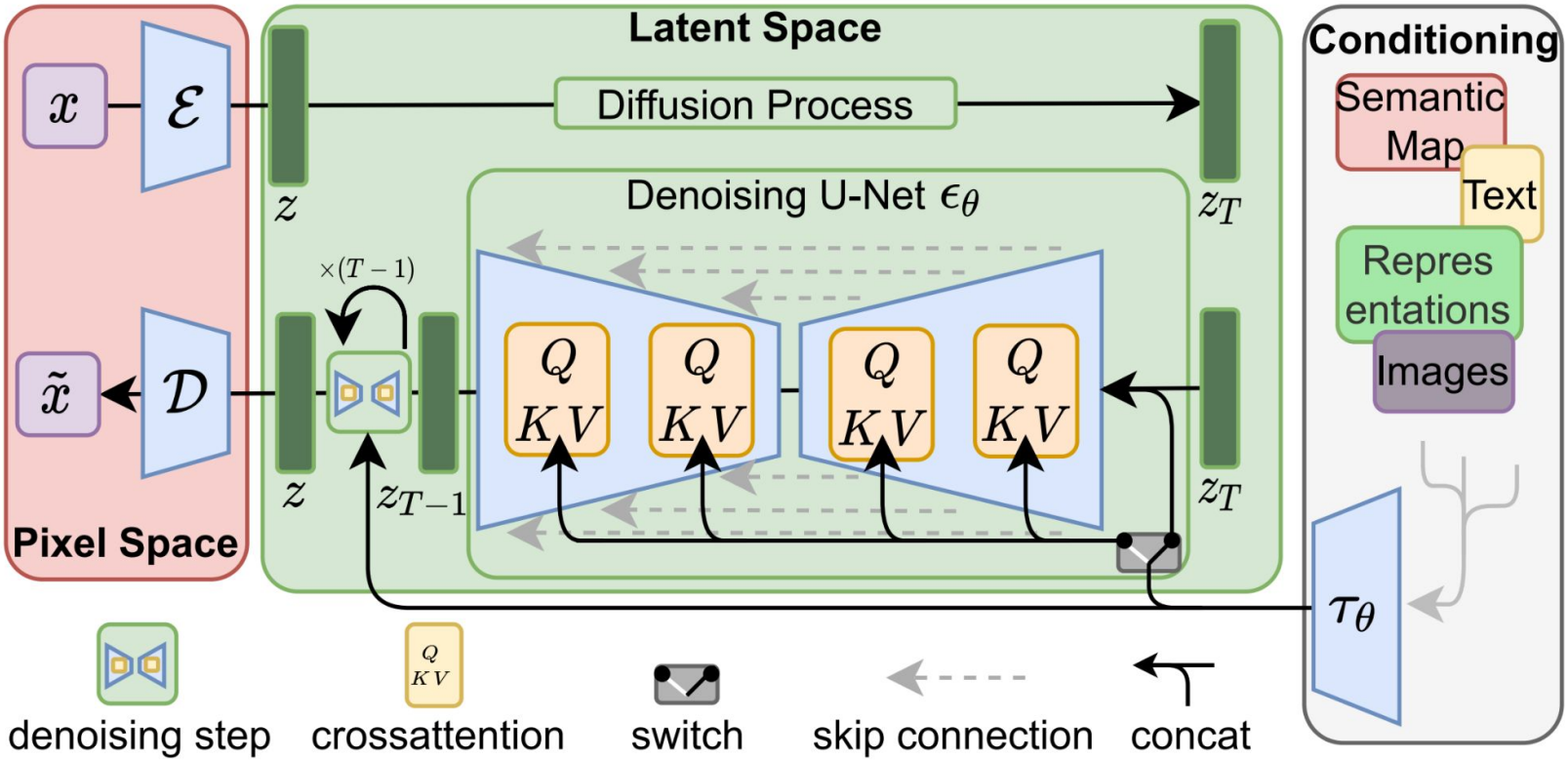


Class	#Positives	z_{sem}	$\mathcal{W}$
5_o_Clock_Shadow	1318	**0.96**	0.94
Arched_Eyebrows	3262	**0.88**	0.86
Attractive	5183	**0.90**	0.86
Bags_Under_Eyes	2564	**0.89**	0.85
Bald	229	0.99	0.99
Bangs	1601	**0.98**	0.95
Big_Lips	3247	**0.73**	0.68
Big_Nose	2813	**0.88**	0.85
Black_Hair	1989	**0.96**	0.93
Blond_Hair	1546	**0.99**	0.97
Blurry	34	**0.90**	0.82
Brown_Hair	2087	**0.89**	0.81
Bushy_Eyebrows	1682	**0.93**	0.85
Chubby	622	**0.95**	0.93
Double_Chin	530	**0.95**	0.94
Eyeglasses	416	1.00	0.98
Goatee	688	**0.98**	0.96
Gray_Hair	395	**0.98**	0.97
Heavy_Makeup	4143	**0.97**	0.95
High_Cheekbones	4160	**0.95**	0.91
Male	3273	1.00	1.00
Mouth_Slightly_Open	4195	**0.98**	0.94
Mustache	502	**0.97**	0.94
Narrow_Eyes	998	**0.86**	0.77
No_Beard	7335	**0.99**	0.97
Oval_Face	1872	**0.77**	0.71
Pale_Skin	434	**0.96**	0.94
Pointy_Nose	2855	**0.74**	0.70
Receding_Hairline	777	**0.94**	0.89
Rosy_Cheeks	1003	**0.96**	0.92
Sideburns	747	**0.99**	0.97
Smiling	4175	**0.99**	0.96
Straight_Hair	1975	**0.84**	0.77
Wavy_Hair	3197	**0.90**	0.87
Wearing_Earrings	2310	**0.92**	0.86
Wearing_Hat	325	**0.99**	0.96
Wearing_Lipstick	5064	**0.98**	0.97
Wearing_Necklace	1501	**0.79**	0.75
Wearing_Necktie	636	**0.96**	0.95
Young	6978	**0.94**	0.91
Weighted average		**0.92**	0.89
Macro average		**0.93**	0.89

Stable Diffusion

- R. Rombach, A. Blattmann, D. Lorenz, P. Esser, B. Ommer, High-Resolution Image Synthesis with Latent Diffusion Models, **2022**, https://arxiv.org/pdf/2112.10752.pdf
  https://github.com/CompVis/latent-diffusion

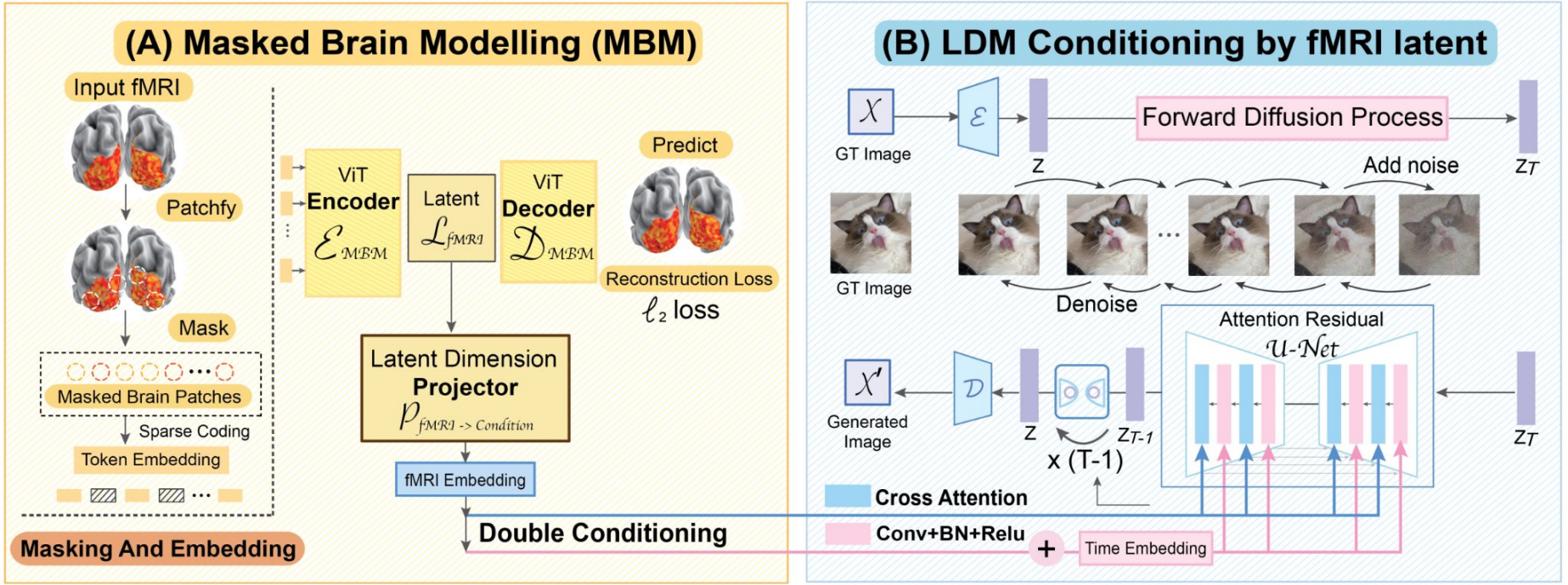
Stable Diffusion



https://arxiv.org/pdf/2112.10752.pdf

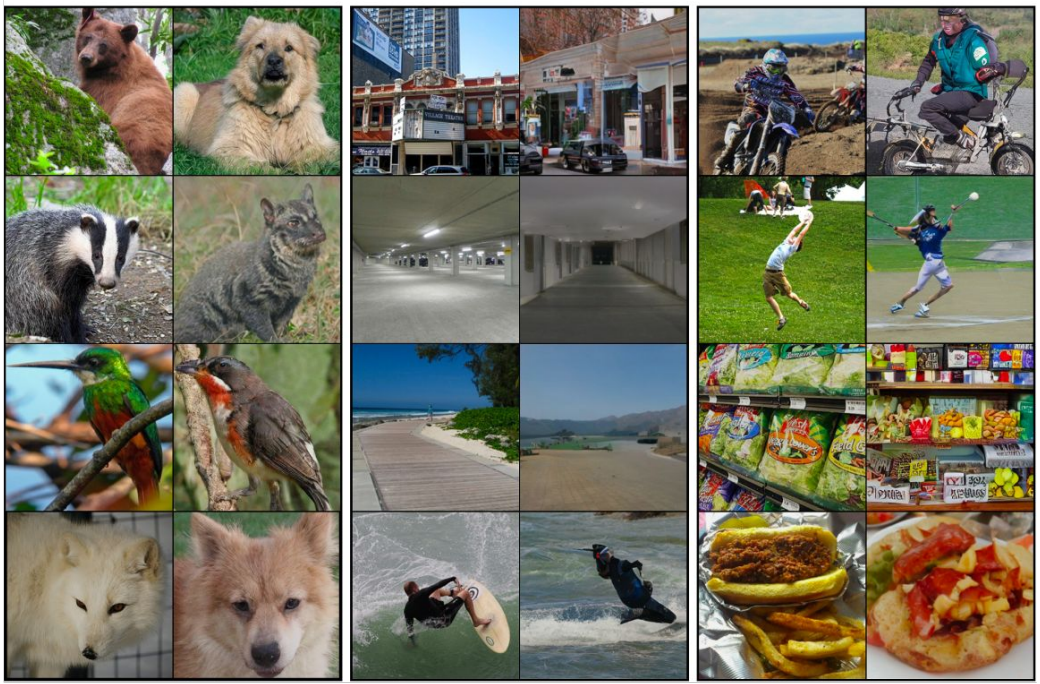
Seeing Beyond the Brain

- J. Chen, et al., Seeing Beyond the Brain: Conditional Diffusion Model with Sparse Masked Modeling for Vision Decoding, **2022**, https://mind-vis.github.io/



https://arxiv.org/abs/2211.06956

Seeing Beyond the Brain



https://arxiv.org/abs/2211.06956

Thank you for your attention