Cornerstones of the Text-to-Pixels Journey

Srikumar Ramalingam

Google Research, NYC
Adjunct Faculty, University of Utah

https://efficient-genai.github.io/

Tutorial Speakers

Richard Hartley
Australian National University

Sadeep Jayasumana
Octave, Ex-Google

Ameesh Makadia
Google Research

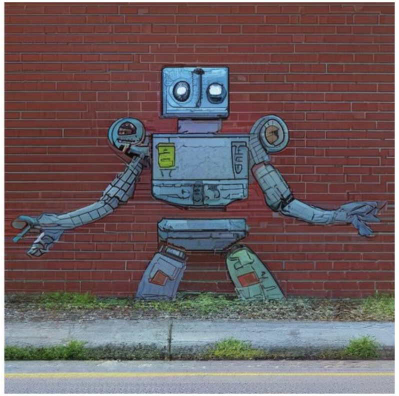
Srikumar Ramalingam
Google Research

Overview

Time	Speaker	Title
9:00 - 9.45	Richard Hartley	Mathematics of Diffusion Models
9:45 - 10.30	Srikumar Ramalingam	Cornerstones of the Text-to-Pixels Journey
10.30 - 11.00	Break	
11.00 - 11:45	Sadeep Jayasumana	MarkovGen: Structured Prediction for fast text-to-image generation
11:45 - 12:30	Ameesh Makadia	Latent representations for efficient text-to-image and text-to-video generation

Text-to-Image Generation

A robot cooking in the kitchen.



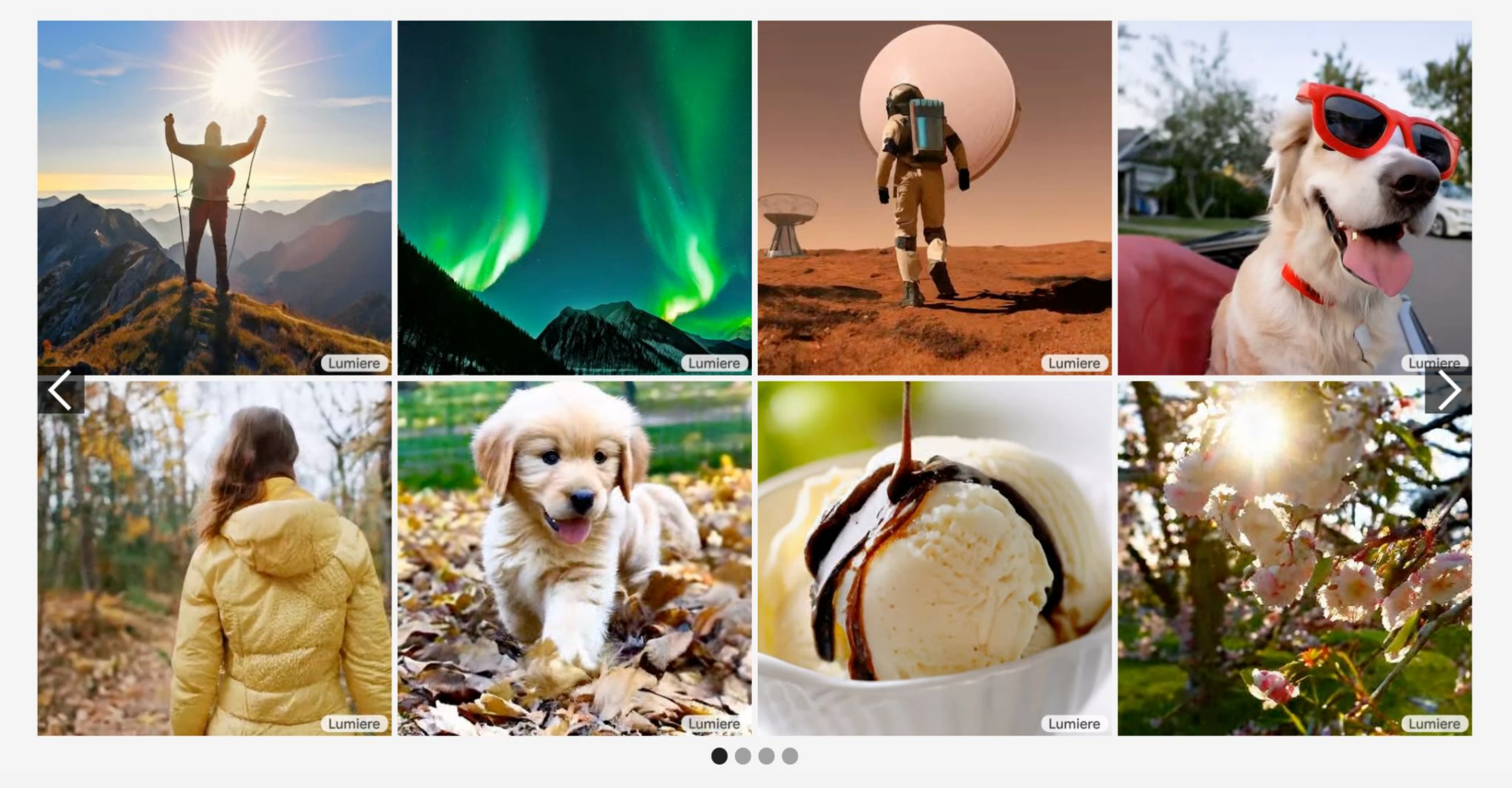
A robot painted as graffiti on a brick wall. a sidewalk is in front of wall, grass is growing out of cracks in the concrete.

A raccoon wearing formal clothes, wearing a tophat. The raccoon is holding a garbage bag.

A hyper-realistic concept art of an alien pyramid landscape, inspired by ArtStation artists.

Jayasumana et al. Markovgen 2023.

Text to video Generation



https://lumiere-video.github.io/
Bar-Tal et al. Lumiere, 2024

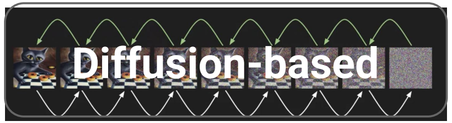
Text-to-3D Generation

https://dreamfusion3d.github.io/

Ben Poole, Ajay Jain, Ben Mildenhall, Jon Barron

Text-to-Image backbone

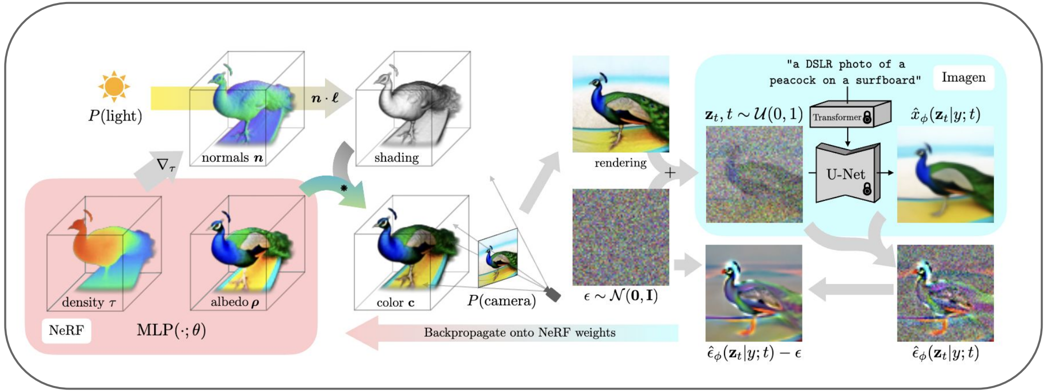
Three-quarters front view of a blue 1977 Corvette coming around a curve in a mountain road and looking over a green valley on a cloudy day.



Generation of images, 3D, videos, or other entities based on a conditional input predominantly use diffusion-based or auto-regressive methods.

Centerpieces of many generative models

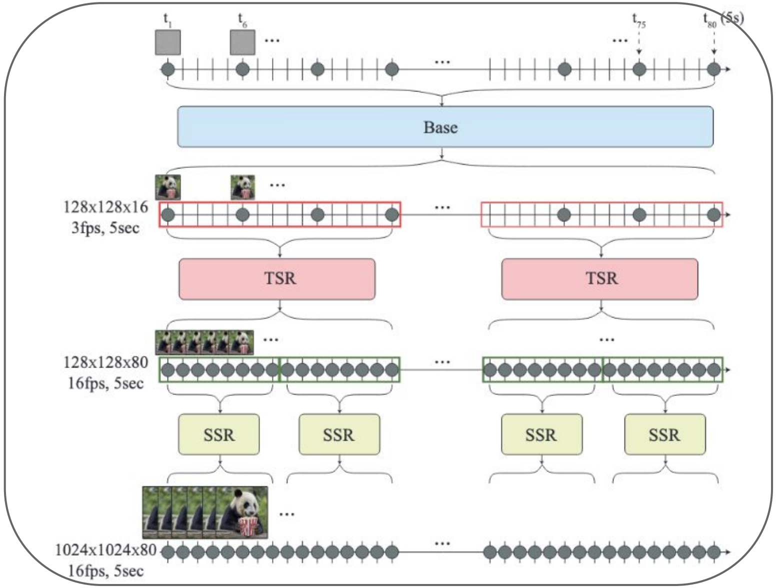
Text-to-3D



Use text-to-image and NeRF models as building blocks to generate 3D from text.

https://dreamfusion3d.github.io/

Text-to-Video



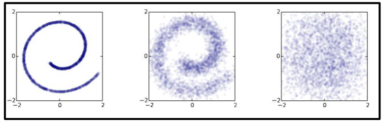
Generate distinct keyframes using text-to-image model, followed by temporal and spatial super-resolution models.

Bar-Tal et al. Lumiere, 2024

Pieces of the Text-to-Image Puzzle

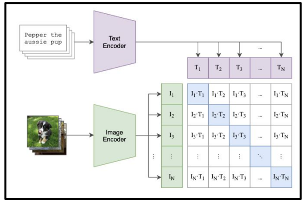
2015

Diffusion



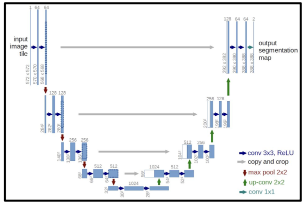
2020

CLIP



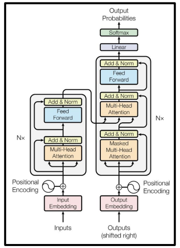
2021

DALL-E



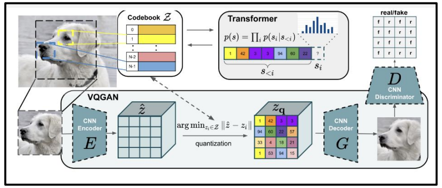
UNet

2015



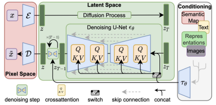
Transformers

2017



VQGAN

2021



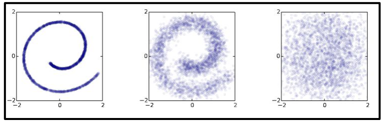
LDM

2021

Pieces of the Text-to-Image Puzzle

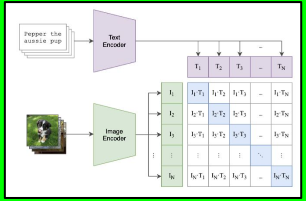
2015

Diffusion



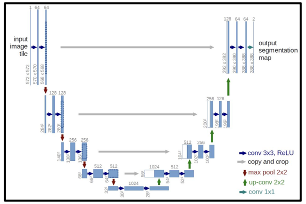
2020

CLIP



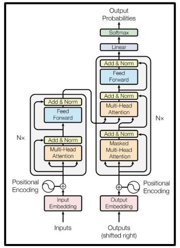
2021

DALL-E



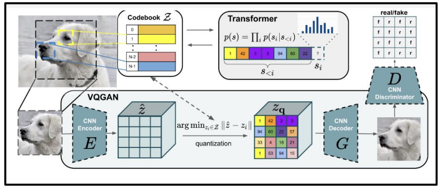
UNet

2015



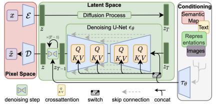
Transformers

2017



VQGAN

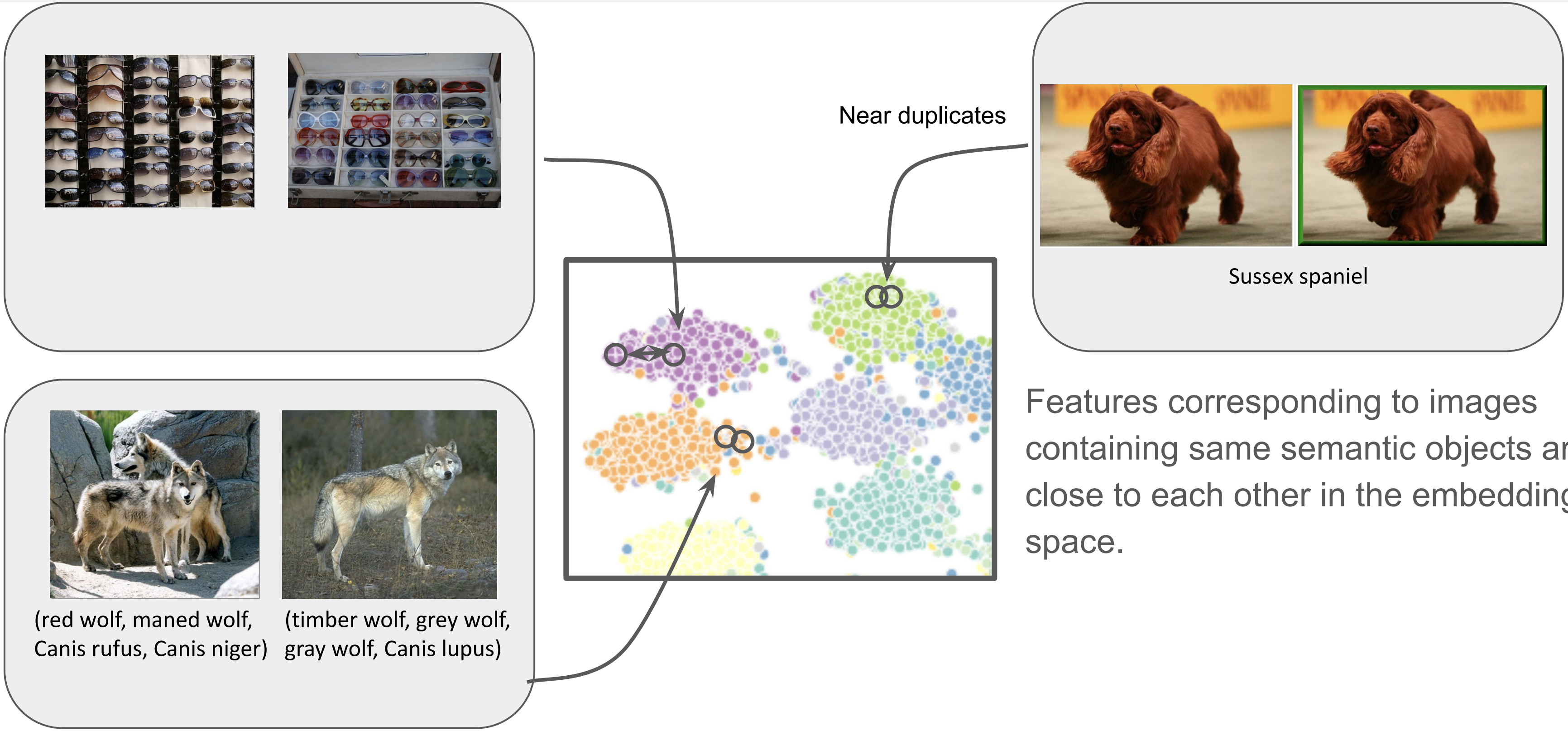
2021



LDM

2021

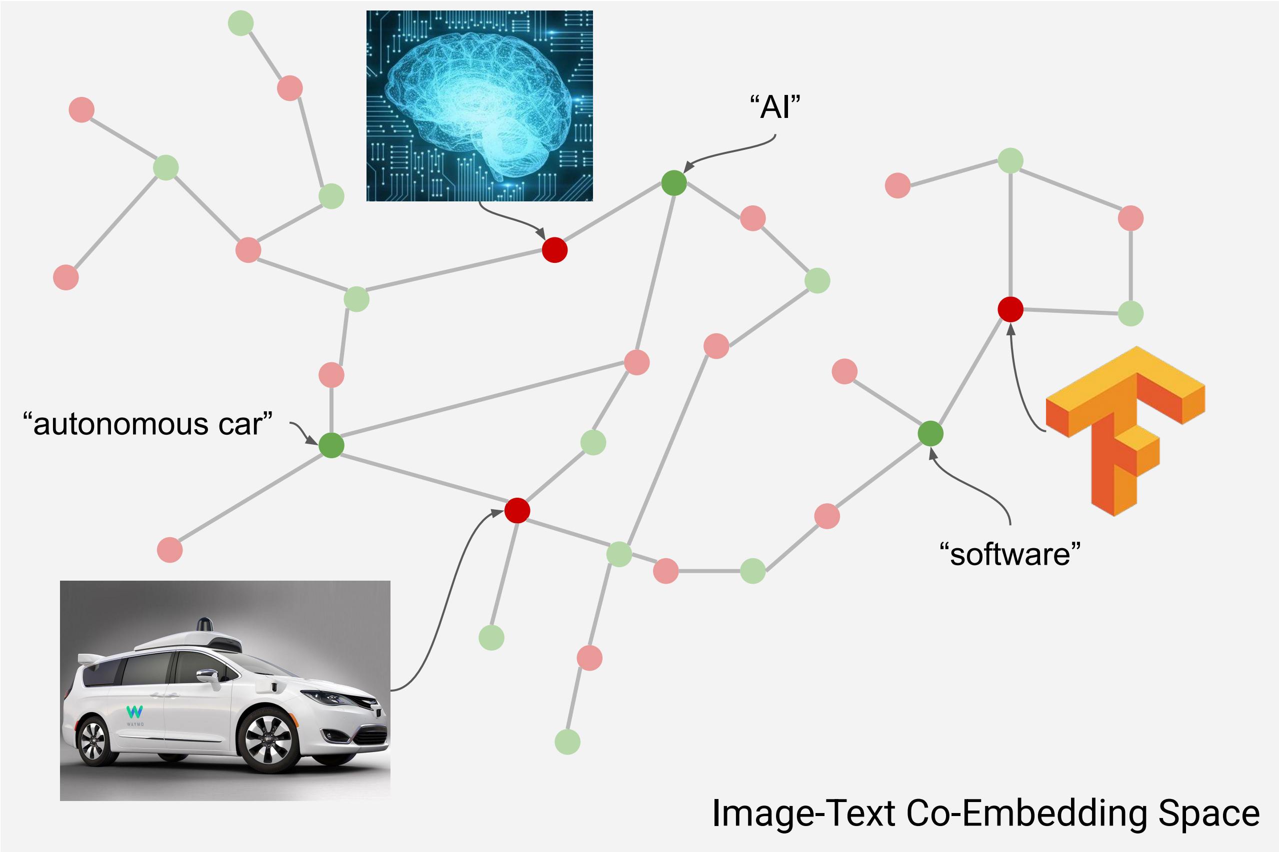
Image features with similar objects are close



Features corresponding to images containing same semantic objects are close to each other in the embedding space.

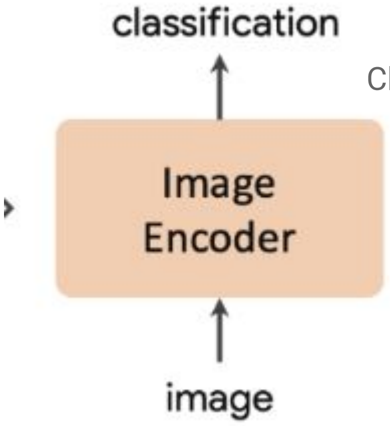
ImageNet training

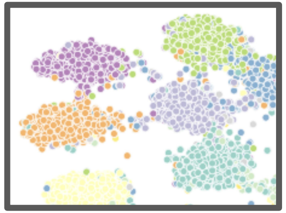
Image-Text Co-Embedding Spaces



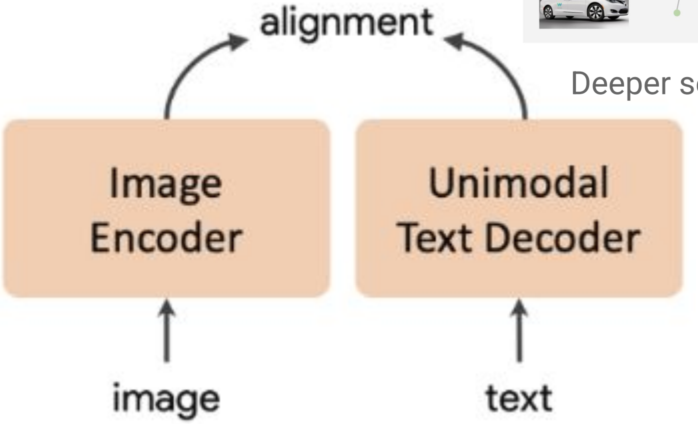
- Bipartite mapping between image and text embeddings
- Building a map where we can associate text address to physical entities, etc.

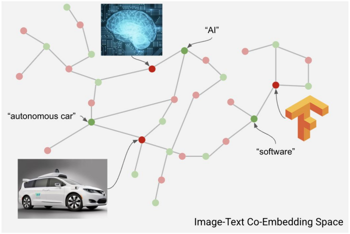
Single tower vs. two tower models





Clusters for image labels



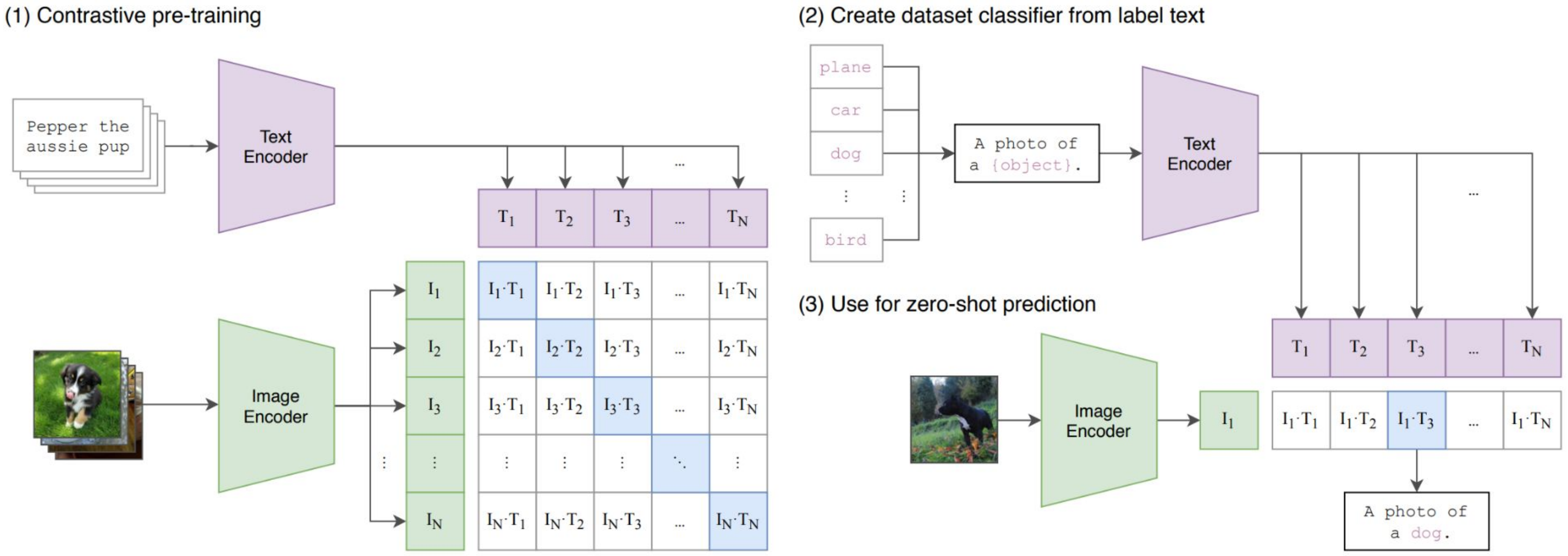


Deeper semantics

Single-tower classification with ResNets or ViTs trained on a chosen set of labels such as in ImageNet.

Learning two tower models allows us to use zero-shot classification methods on different classes.

CLIP/ALIGN



$$L_{i2t} = -\frac{1}{N}\sum_{i}^{N}\log\frac{\exp(x_i^\top y_i/\sigma)}{\sum_{j=1}^{N}\exp(x_i^\top y_j/\sigma)} \qquad L_{t2i} = -\frac{1}{N}\sum_{i}^{N}\log\frac{\exp(y_i^\top x_i/\sigma)}{\sum_{j=1}^{N}\exp(y_i^\top x_j/\sigma)}$$

x_i, y_i - Image and Text normalized embeddings

InfoNCE - Sohn, K. Improved deep metric learning with multi-class n-pair loss objective. NeurIPS, 2016.

Text-Image Co-embedding References

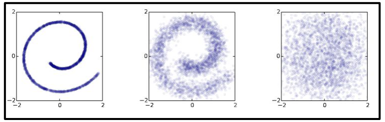
[CLIP] Learning Transferable Visual Models From Natural Language Supervision, 2021.

[ALIGN] Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision, 2021.

Pieces of the Text-to-Image Puzzle

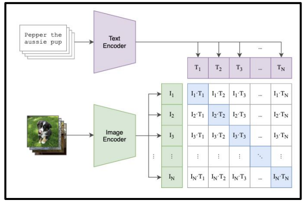
2015

Diffusion



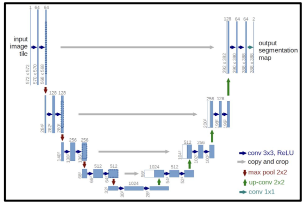
2020

CLIP



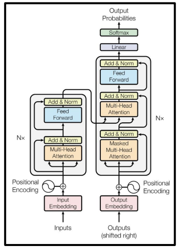
2021

DALL-E



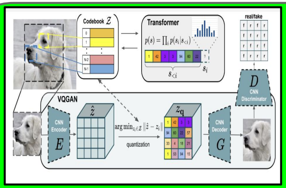
UNet

2015



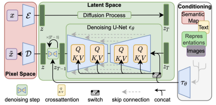
Transformers

2017



VQGAN

2021

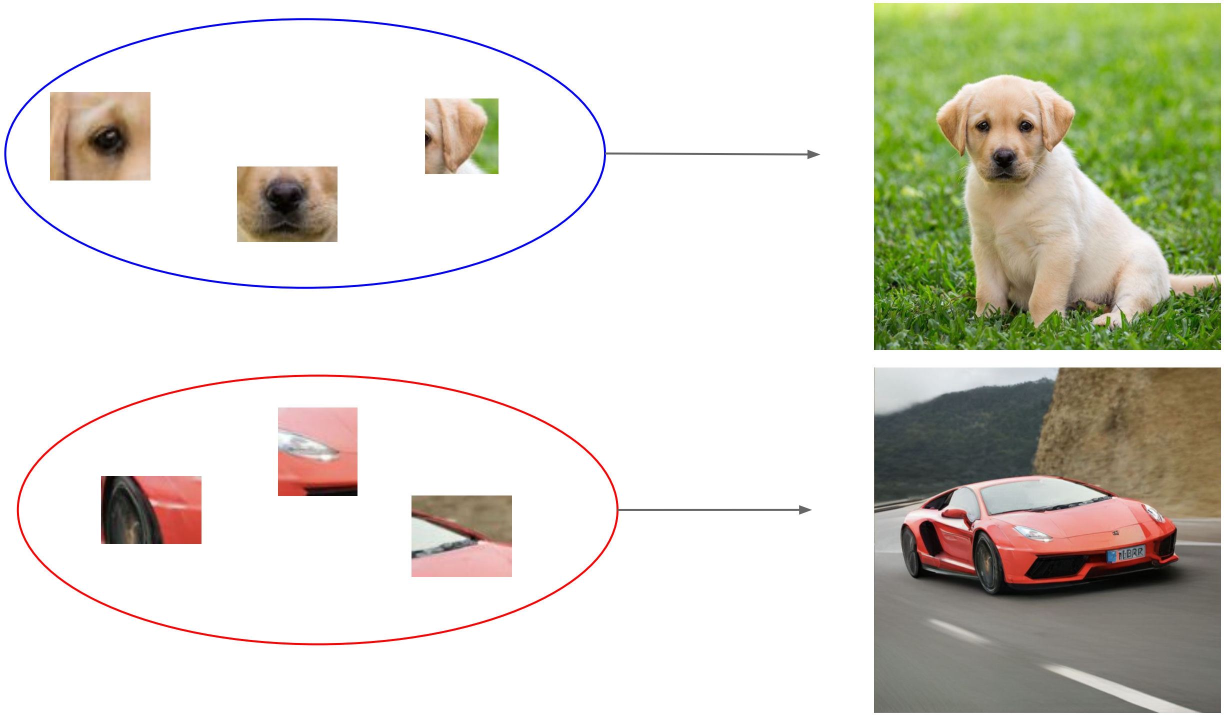


LDM

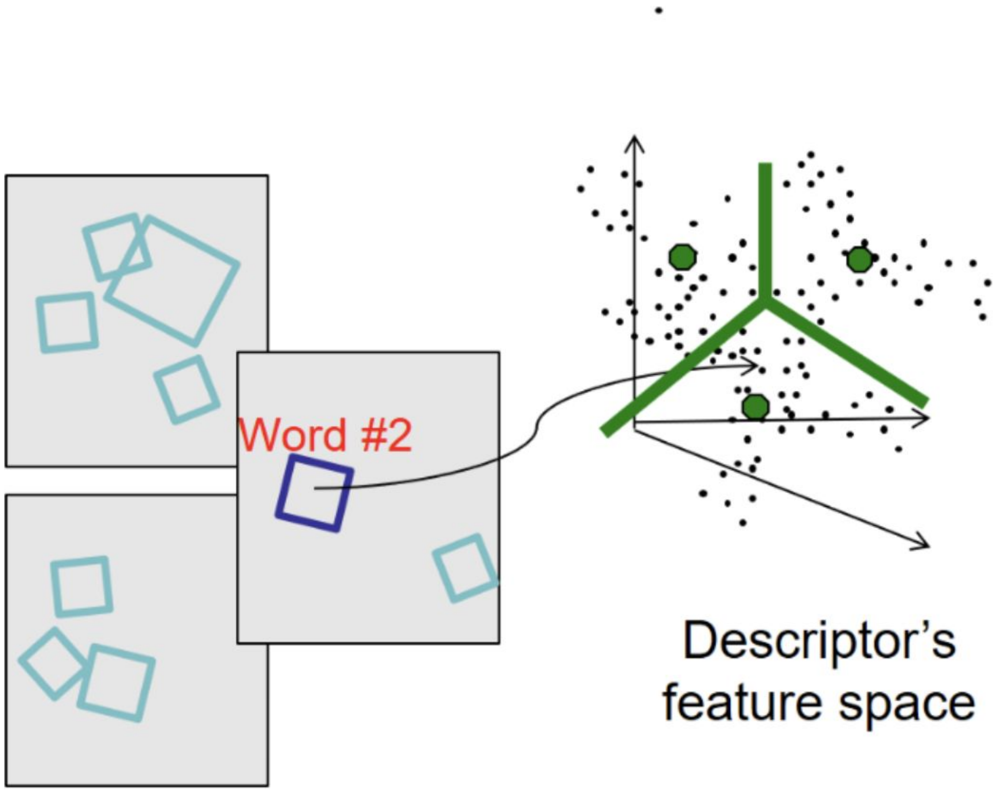
2021

Background: Visual Words

Individual parts of an object reveal a lot of information.



Background: Visual words



- Quantize via clustering, let cluster centers be prototype "words"
- Determine which word to assign to each new image region by finding the closest cluster center.

Source: David Nister

Background: Visual words

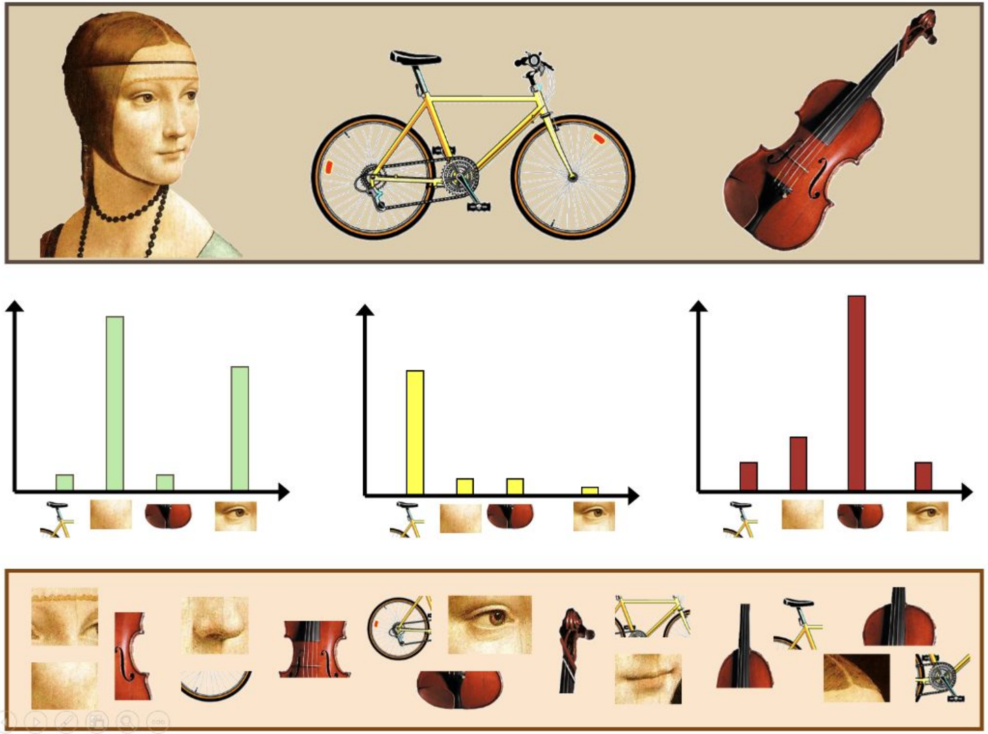
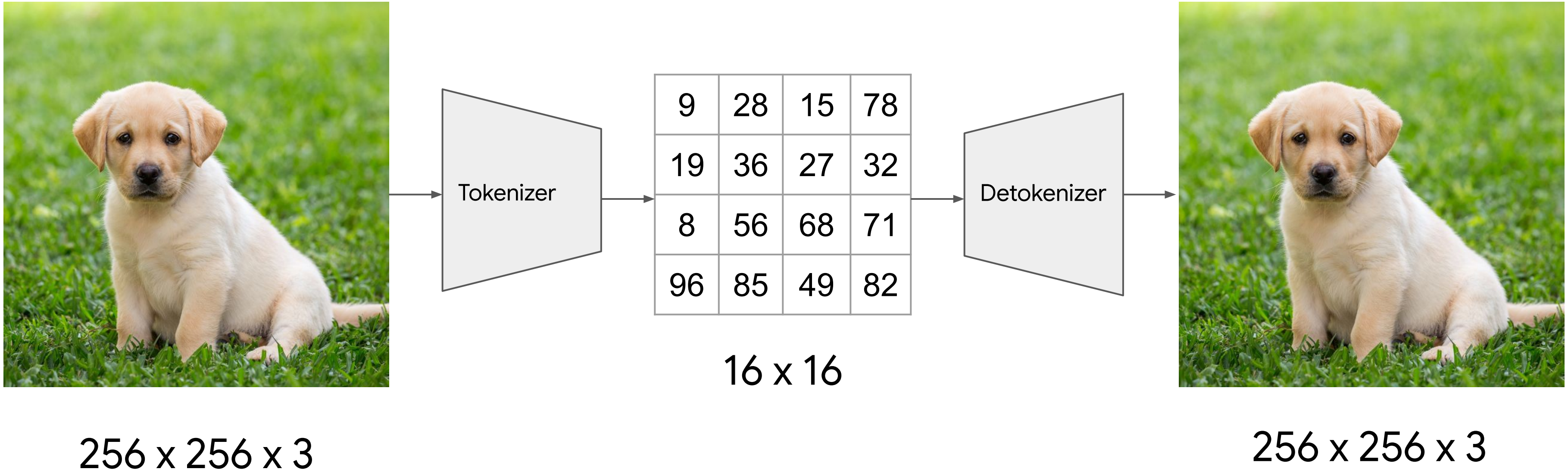


Image search and image generation are closely related problems.- if you can describe an image accurately you can generate it as well ..

Source: Kristen Grauman

Image Tokenization

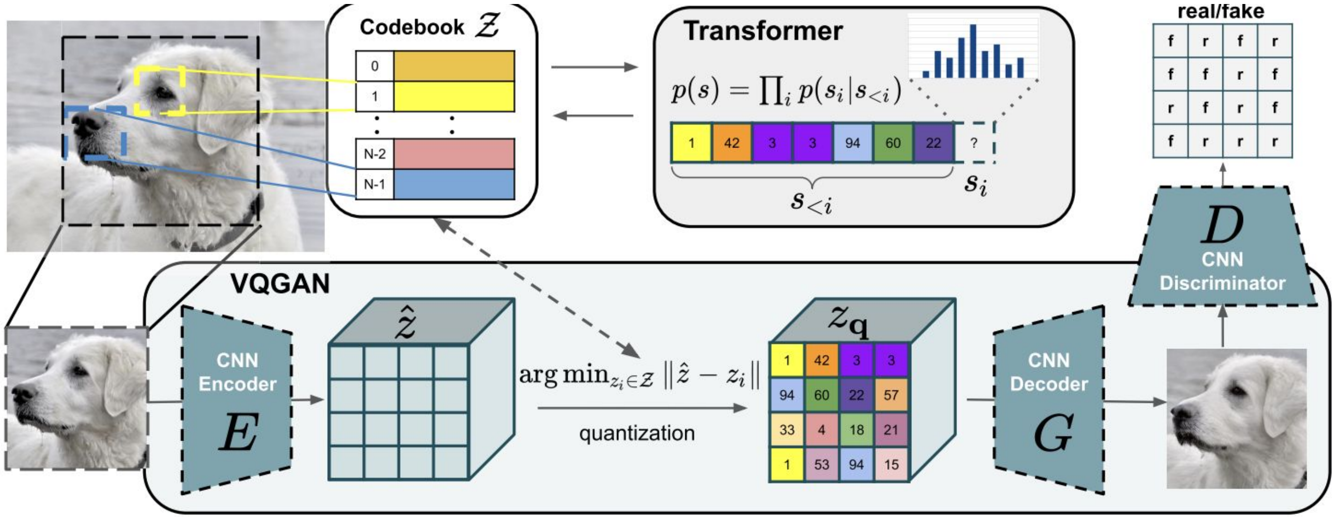


Key Idea in VQGAN

- Use for CNNs for learning local features and transformers for long range interactions
  - CNNs are used to learn a codebook of context-rich visual parts.
  - Transformers are used to model the long range interactions among the individual visual parts.
- Efficient image generation backbone that allows conditional inputs (similar to ControlNet).
- Default choice in Latent diffusion, MUSE, Parti, Paella, etc.

Taming transformers for high-resolution image synthesis,
Patrick Esser*, Robin Rombach*, Björn Ommer, 2020.

Overview of VQGAN



Two stage training:

- Learn the encoder, decoder, and codebook.
- Learn the transformer to synthesize images with conditional inputs.

Codebook

$$x \in \mathbb{R}^{H \times W \times 3}$$

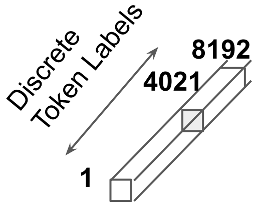
Input Image

3861	2201	743	408
221	200	4999	6021
421	8001	7871	1213
7495	4259	121	910

Token Image

$$\mathcal{Z} = \{z_k\}_{k=1}^{K} \subset \mathbb{R}^{n_z}$$

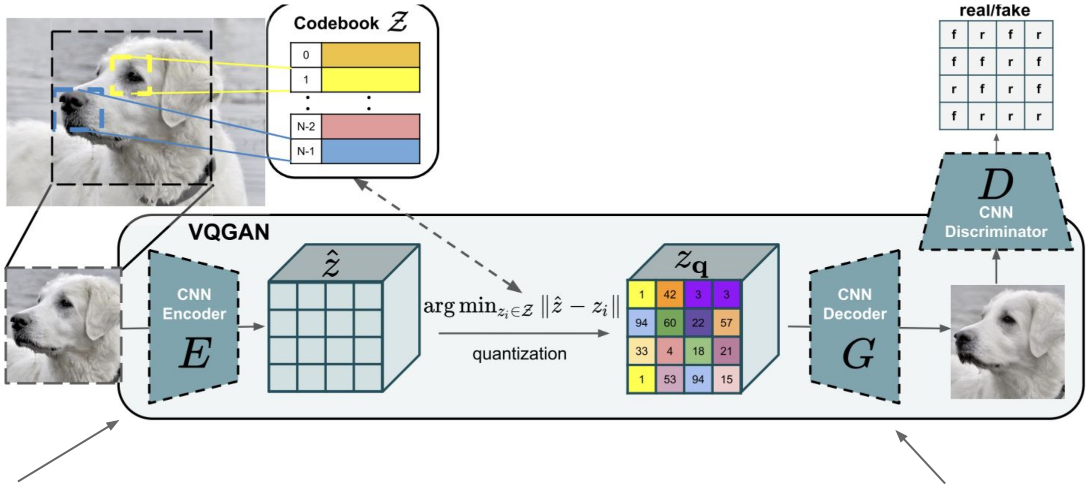
Discrete codebook consisting of K vectors



$$z_{\mathbf{q}} \in \mathbb{R}^{h \times w \times n_z}$$

Image represented with codebook entries

Codebook



$$\hat{z} = E(x) \in \mathbb{R}^{h \times w \times n_z}$$

$$\hat{x} = G(z_{\mathbf{q}})$$

$$z_{\mathbf{q}} = \mathbf{q}(\hat{z}) := \left(\underset{z_k \in \mathcal{Z}}{\arg\min} \|\hat{z}_{ij} - z_k\| \right) \in \mathbb{R}^{h \times w \times n_z}$$

Learning the codebook

$$\hat{x} = G(z_{\mathbf{q}}) = G\left(\mathbf{q}(E(x))\right)$$

$$\mathcal{L}_{\mathrm{VQ}}(E, G, \mathcal{Z}) = \|x - \hat{x}\|^2 + \boxed{\|\mathrm{sg}[E(x)] - z_{\mathbf{q}}\|_2^2} + \|\mathrm{sg}[z_{\mathbf{q}}] - E(x)\|_2^2$$

Move the codebook vectors closer to the frozen encoder vectors, and vice versa.

- Reconstruction loss optimizes the encoder and decoder.
- L2 loss to move the encoder outputs towards the codebook entries and another L2 loss to move codebook entries towards the encoder outputs.

[Aaron van den Oord et al. Neural Discrete Representation Learning, 2017]

Learning a perceptually rich codebook

Discriminator wants to maximize this, while the generator wants to minimize this.

GAN Loss:

$$\mathcal{L}_{\mathrm{GAN}}(\{E, G, \mathcal{Z}\}, D) = \boxed{[\log D(x) + \log(1 - D(\hat{x}))]}$$

$$\mathcal{Q}^* = \underset{E,G,\mathcal{Z}}{\arg\min} \max_{D} \mathbb{E}_{x \sim p(x)} \Big[\mathcal{L}_{\mathrm{VQ}}(E, G, \mathcal{Z}) + \lambda \mathcal{L}_{\mathrm{GAN}}(\{E, G, \mathcal{Z}\}, D) \Big]$$

- Learn the encoder, decoder, and codebook with a perceptual and GAN loss.

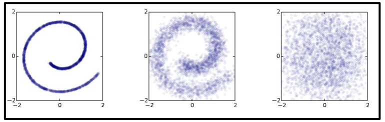
Feature Codebook References

- [VQGAN]: Taming Transformers for High-Resolution Image Synthesis, 2020.
- [VQVAE]: Neural Discrete Representation Learning, 2018.
- Video Google: A Text Retrieval Approach to Object Matching in Videos, 2003.

Pieces of the Text-to-Image Puzzle

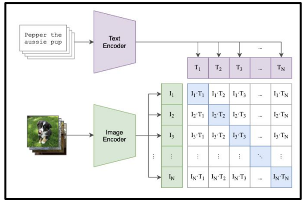
2015

Diffusion



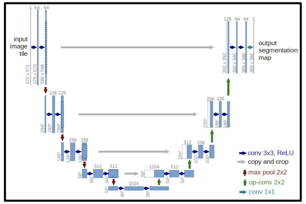
2020

CLIP



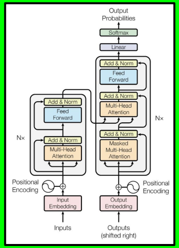
2021

DALL-E



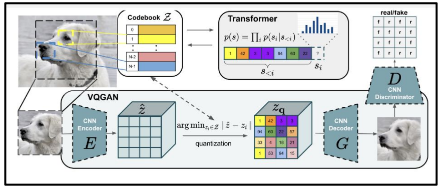
UNet

2015



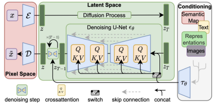
Transformers

2017



VQGAN

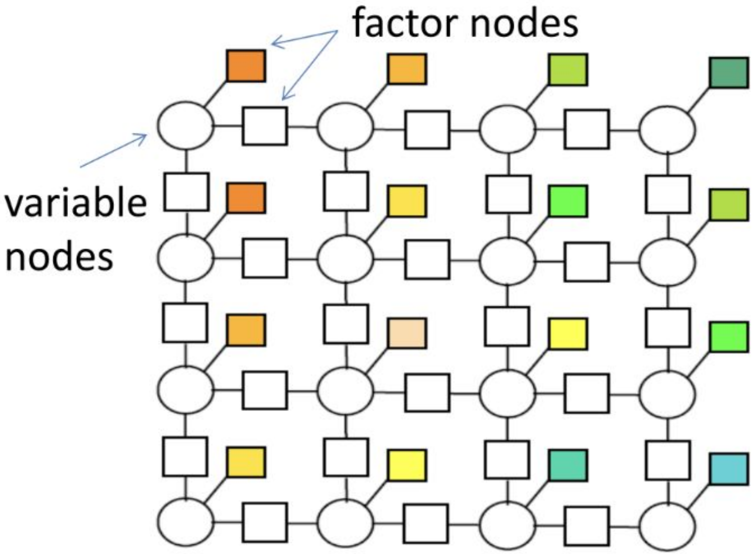
2021



LDM

2021

Markov Random Fields (MRFs)



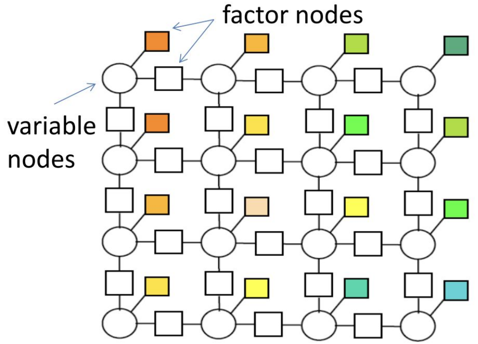
Goal: find most probable interpretation of scene

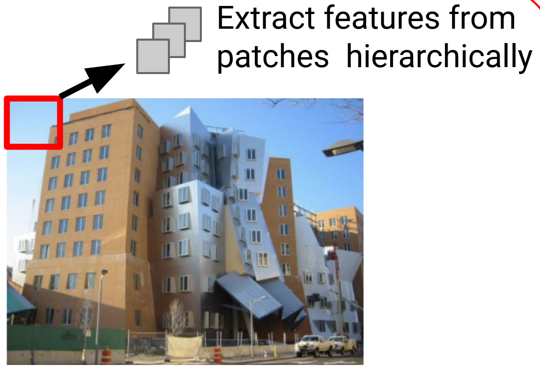
Minimize an energy function:

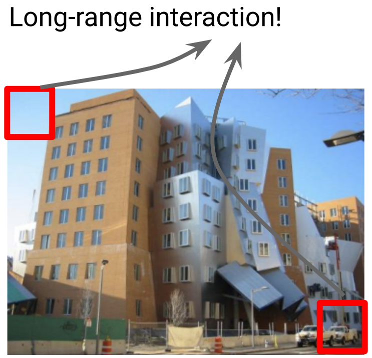
$$E(\mathbf{x}) = \mathrm{unary\_cost} + \mathrm{pairwise\_cost}$$

- Solve using using graph cuts or BP

Model Hierarchy (MRFs -> CNNs -> Transformers)







MRFs with 4 or 8-neighborhood were solved efficiently using graph cuts and belief propagation.

CNNs are very good at extracting local features!

Transformers allow long range interactions!

Graphcuts

1999

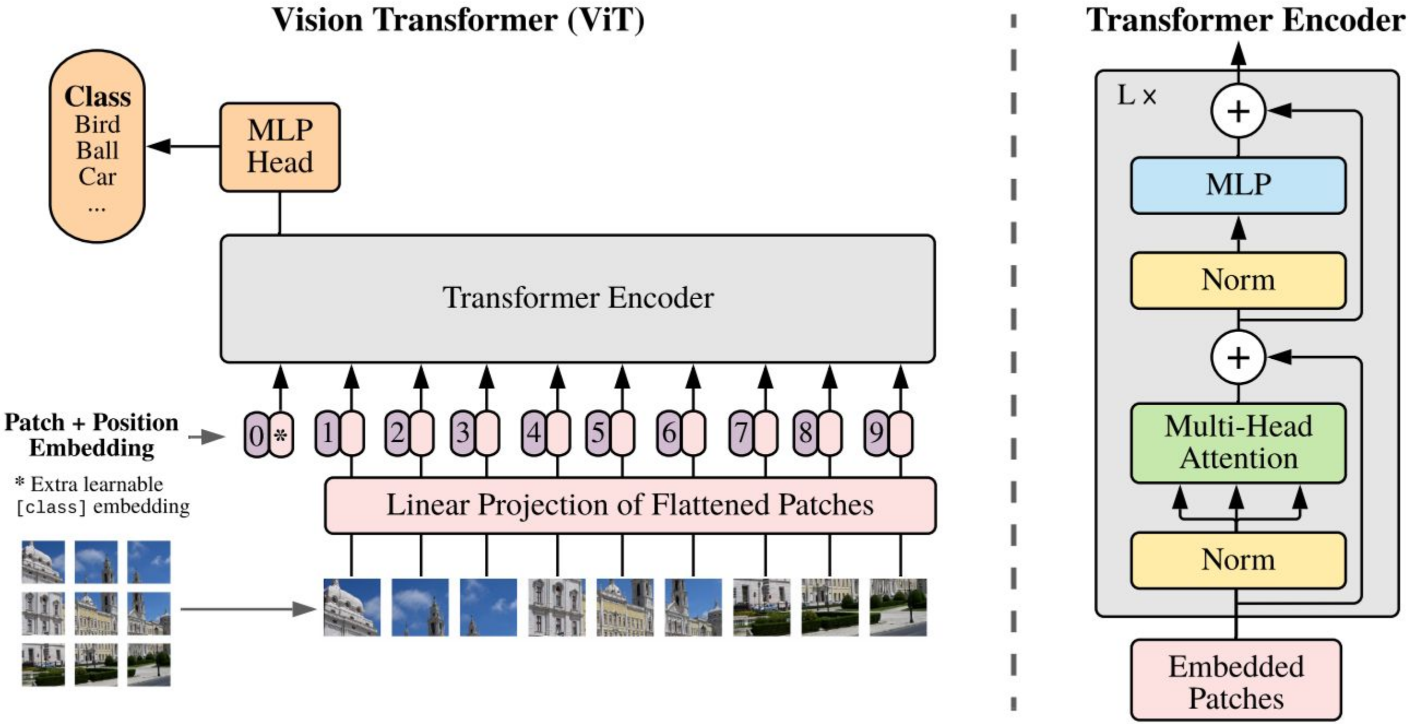
AlexNet

2012

Transformers

2017

Vision Transformer



Conditioned Synthesis using Transformers

With the encoder, decoder, and codebook, we can treat the image synthesis problem as sequence prediction problem.

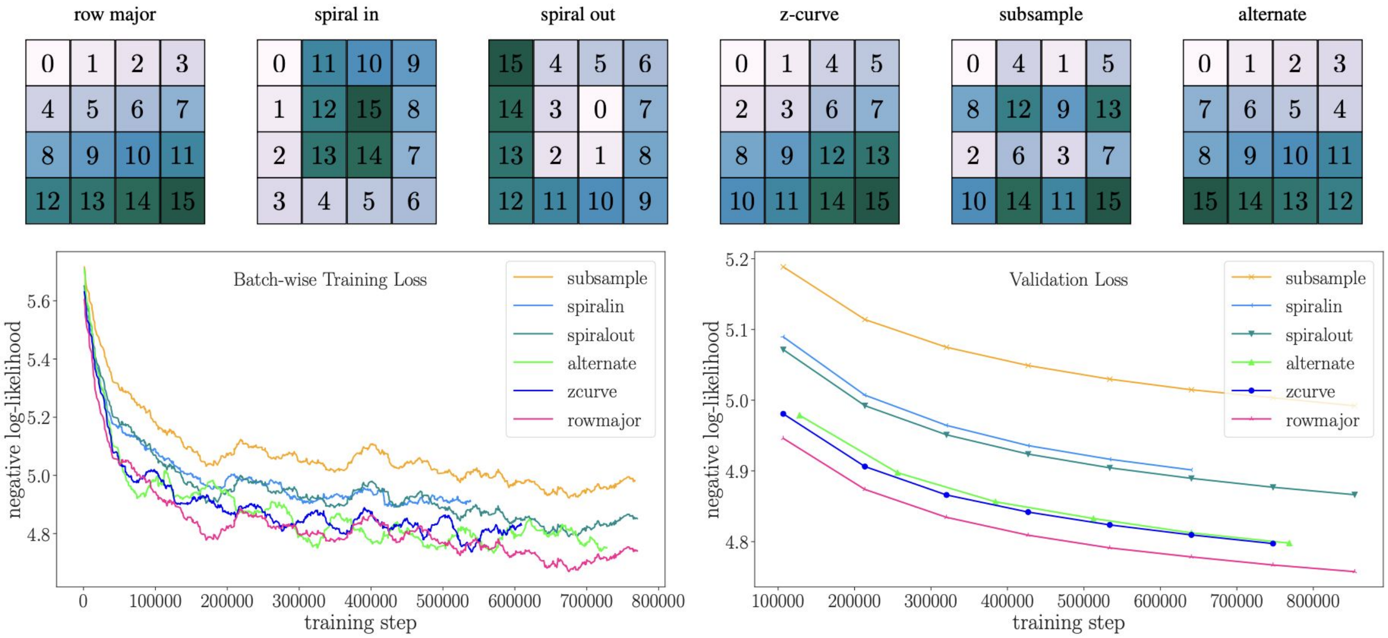
[..,743, 408, 221, 200, ...]

3861	2201	743	408
221	200	4999	6021
421	8001	7871	1213
7495	4259	121	910

Token Image

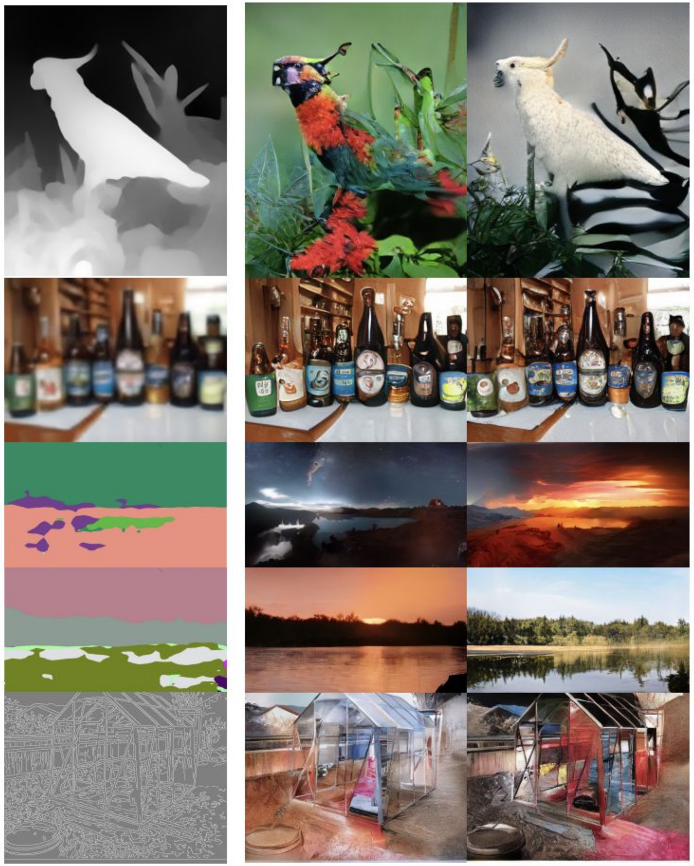
- Based on some ordering, the token prediction can be achieved auto-regressively by feeding the previous tokens.
- To provide conditional inputs, we can learn another codebook if it has spatial extent to generate token indices for conditions.

Different ordering of tokens for image synthesis



- The ordering is trivial for language tasks, whereas there is no easy way to fix the ordering for images.

Conditioned Image Synthesis



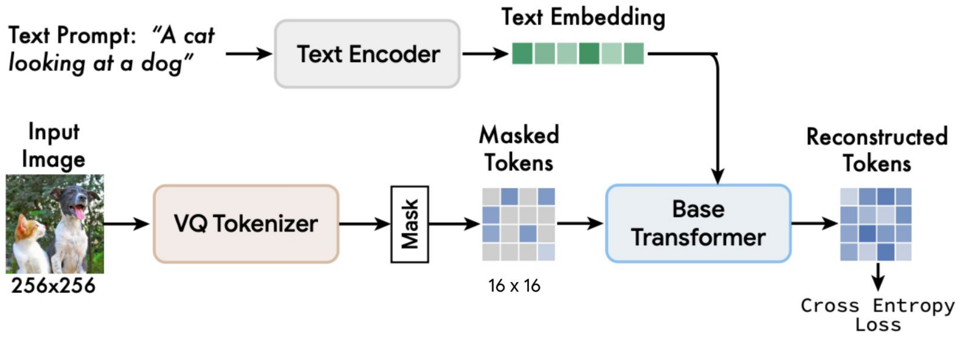
Depth -> Image

Low res. -> High res.
(Superresolution)

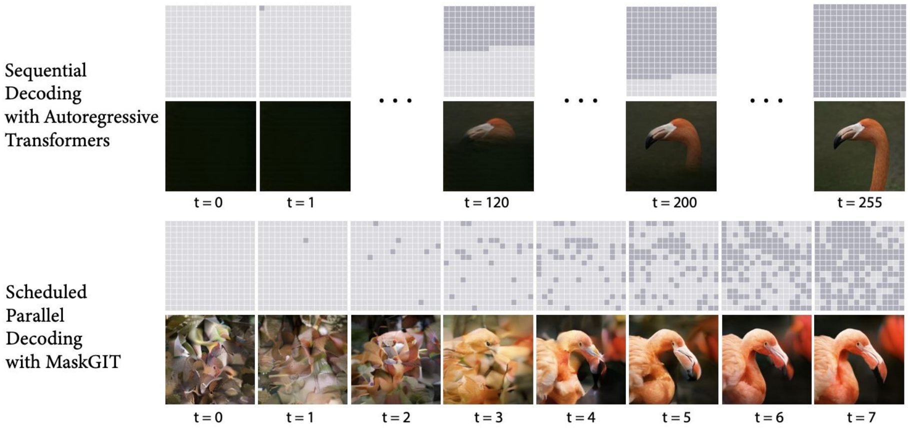
Semantic -> Image

Edge -> Image

Efficient Text-to-Image Generation using Muse



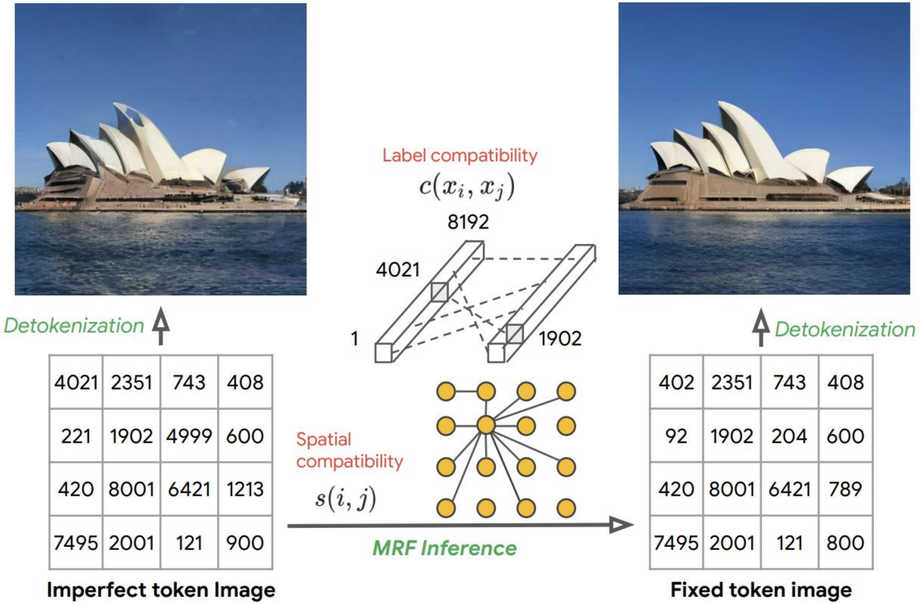
The Muse 3B model is 10x faster than Parti/Imagen 3B on TPUv4.



Chang et al. Muse: Text-To-Image Generation via Masked Generative Transformers, 2023.

MarkovGen: MRFs to speedup Muse

$$E(\mathbf{x}) = \text{unary\_cost} + \text{pairwise\_cost}$$



Jayasumana et al. **MarkovGen: Structured Prediction for Efficient Text-to-Image Generation, 2024.**

MRF: Model Formulation

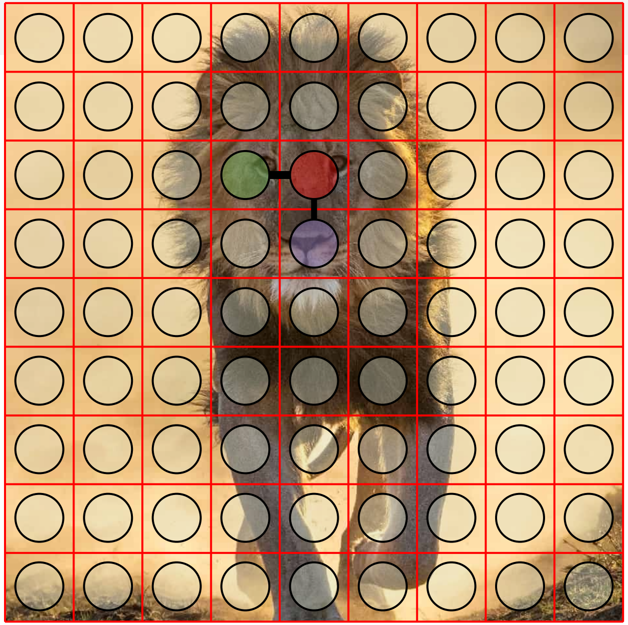
$$E(\mathbf{x}) = \text{unary\_cost} + \text{pairwise\_cost}$$

Unary Cost

- $\text{cost}(X_i = l) = ?$
- You pay a penalty if your label doesn't agree with the classifier.

Pairwise cost

- $\text{cost}(X_i = l', X_j = l'') = ?$
- You pay a penalty if you assign *"incompatible"* labels to two *"neighboring"* tokens.



$$\text{cost}(X_i = l) = -\,\text{logit}_\text{i}(l)$$

$$\text{cost}(X_i = l', X_j = l'') = -c(l', l'')s(i, j)$$

Speedup over Muse without quality loss.

Full Muse: All steps | Early Exit Muse: Fewer steps 1.5x faster | MarkovGen: Fewer steps + MRF 1.5x faster

A robot cooking in the kitchen

A robot painted as graffiti on a brick wall. a sidewalk is in front of the wall, and grass is growing out of cracks in the concrete.

Model	Time (ms)
Muse base (single step)	10.40
Muse super-resolution (single step)	24.00
MRF inference on base	0.29
MRF inference on super-resolution	0.29
T5-XXL inference	0.30
Detokenizer	0.15
Muse	442.05
MarkovGen (ours)	281.03

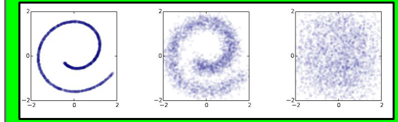
MRF and Transformers References

- Masked generative image transformer. In: CVPR (2022)
- Muse:Text-to-image generation via masked generative transformers. ICML (2023)
- Markovgen: Structured prediction for efficient text-to-image generation (2023)
- Hierarchical text-conditional image generation with clip latents. preprint (2022)
- Photorealistic text-to-image diffusion models with deep language understanding. preprint (2022),
- Scaling autoregressive models for content-rich text-to-image generation. In: ICML (2022)

Pieces of the Text-to-Image Puzzle

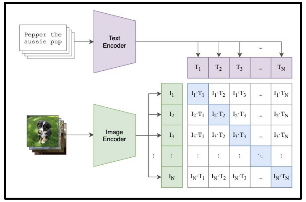
2015

Diffusion



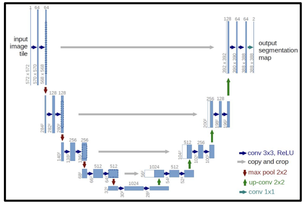
2020

CLIP



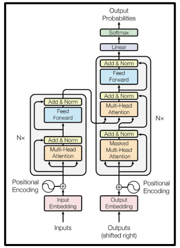
2021

DALL-E



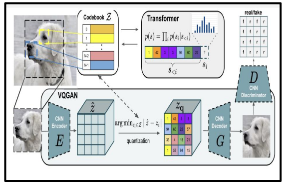
UNet

2015



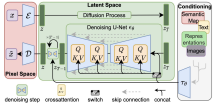
Transformers

2017



VQGAN

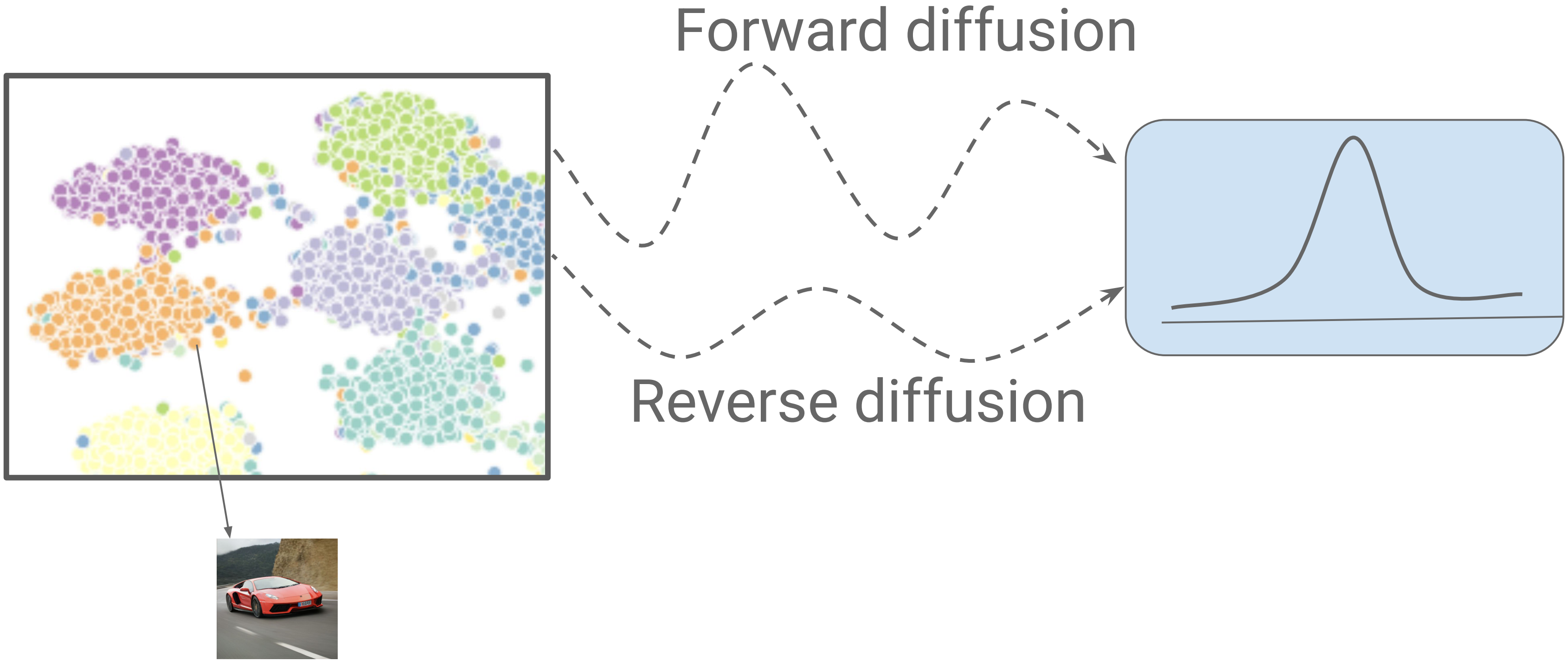
2021



LDM

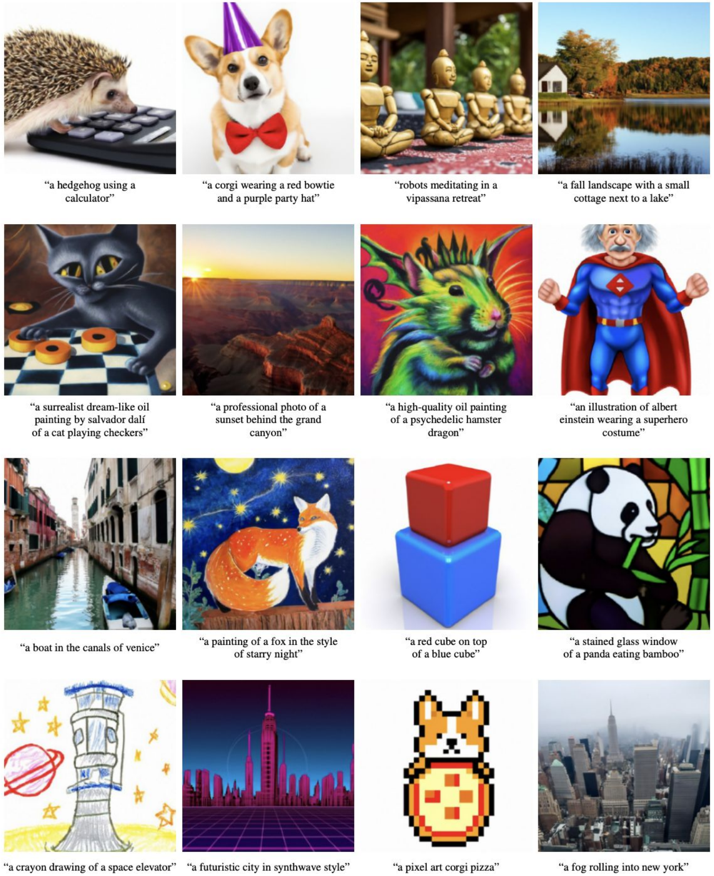
2021

Basic idea -> Diffusion Model



[Deep unsupervised learning of nonlinear thermodynamics, Sohl-Dickstein et al. 2015]

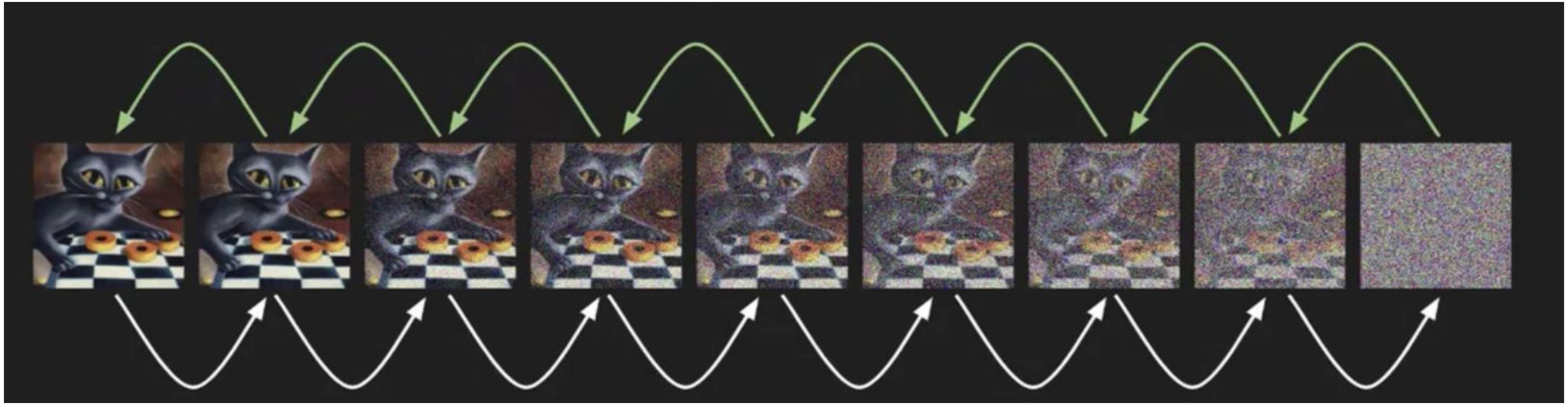
Diffusion Models



[Nichol et al. GLIDE 2021]

Background: Diffusion models

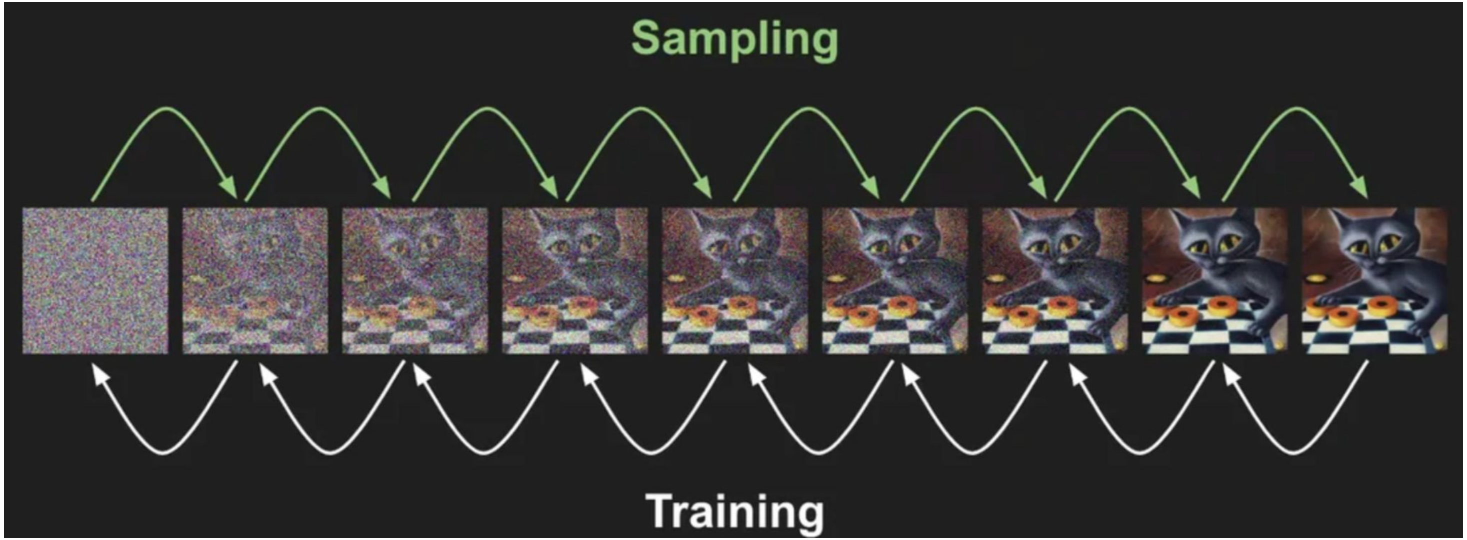
“Systematically and slowly destroy structure in a data distribution through an iterative forward diffusion process.



We then learn a reverse diffusion process that restores structure in data, yielding a highly flexible and tractable generative model of the data.”

[Deep unsupervised learning of nonlinear thermodynamics, Sohl-Dickstein et al. 2015]

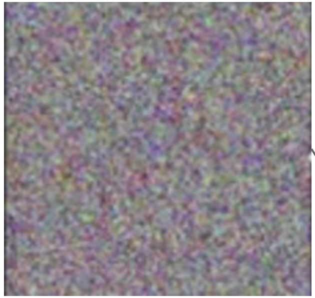
Background: Diffusion models



- While training we start with clean images from the dataset, add noise and try to predict the added noise.
- While sampling, we start with noise and iteratively denoise the image to generate an image.

Diffusion model

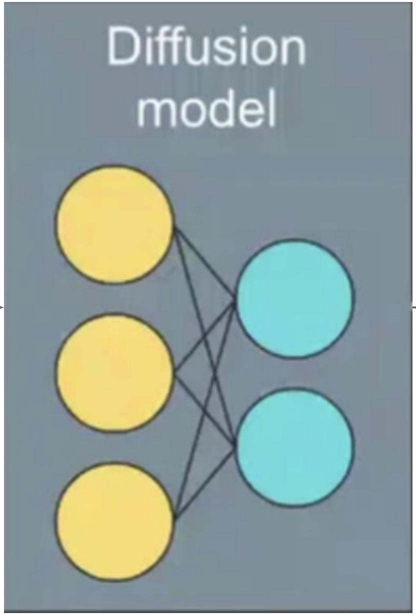
Mean squared error loss: $||\epsilon - pred||^2$

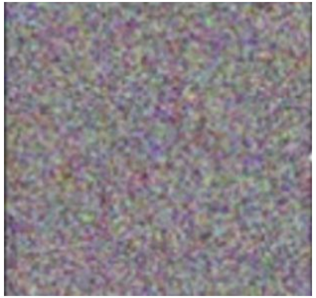


noise ϵ

image x_0

Noised image x_t





$pred$

[Nichol et al. GLIDE 2021]

Training Diffusion models

$x_0 \sim q(x_0)$ x_1 x_2 x_T

Markov chain of latent variables by progressively adding Gaussian noise.

Sample an image from the data distribution

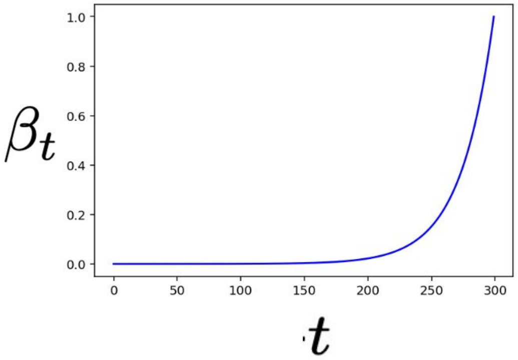
Training Diffusion models

$x_0 \sim q(x_0)$ x_1 x_2 x_T

Sample an image from the data distribution

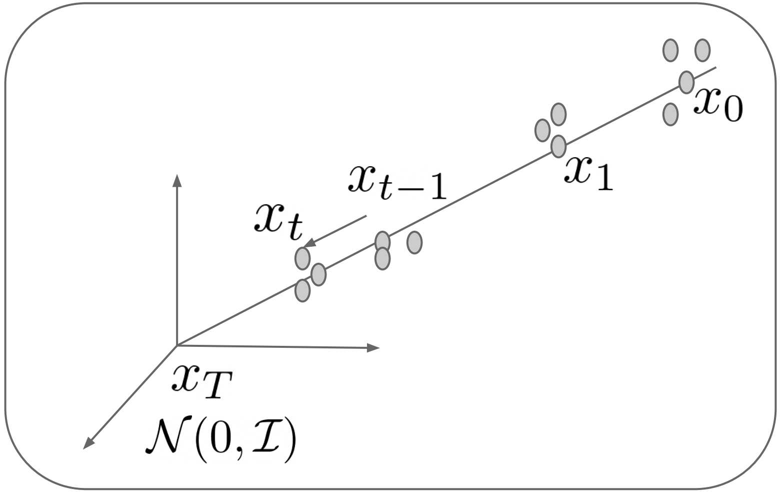
Markov chain of latent variables by progressively adding Gaussian noise.

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$



Training diffusion models

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$



- We are somewhat shrinking the mean and moving it towards the 0.
- If the total noise added is large enough, and if each step adds small enough noise, then x_T can be approximated by $\mathcal{N}(0, \mathcal{I})$.

Training Diffusion Models

$x_0 \sim q(x_0)$ x_1 x_2

Sample an image from the data distribution

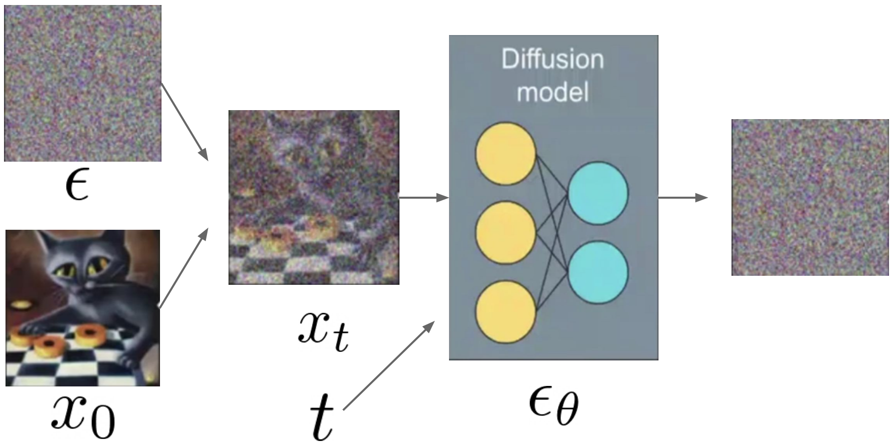
Markov chain of latent variables by progressively adding Gaussian noise.

$$\alpha_t := 1 - \beta_t \qquad \bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$$

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I})$$

$$(1 - \alpha_t) < 1, \sqrt{\alpha} < 1$$

Loss Function



$$L_{simple} = E_{t\sim[1,T],x_0\sim q(x_0),\epsilon\sim\mathcal{N}(0,I)}[||\epsilon - \epsilon_\theta(x_t, t)||^2]$$

Sampling and Training pseudocode

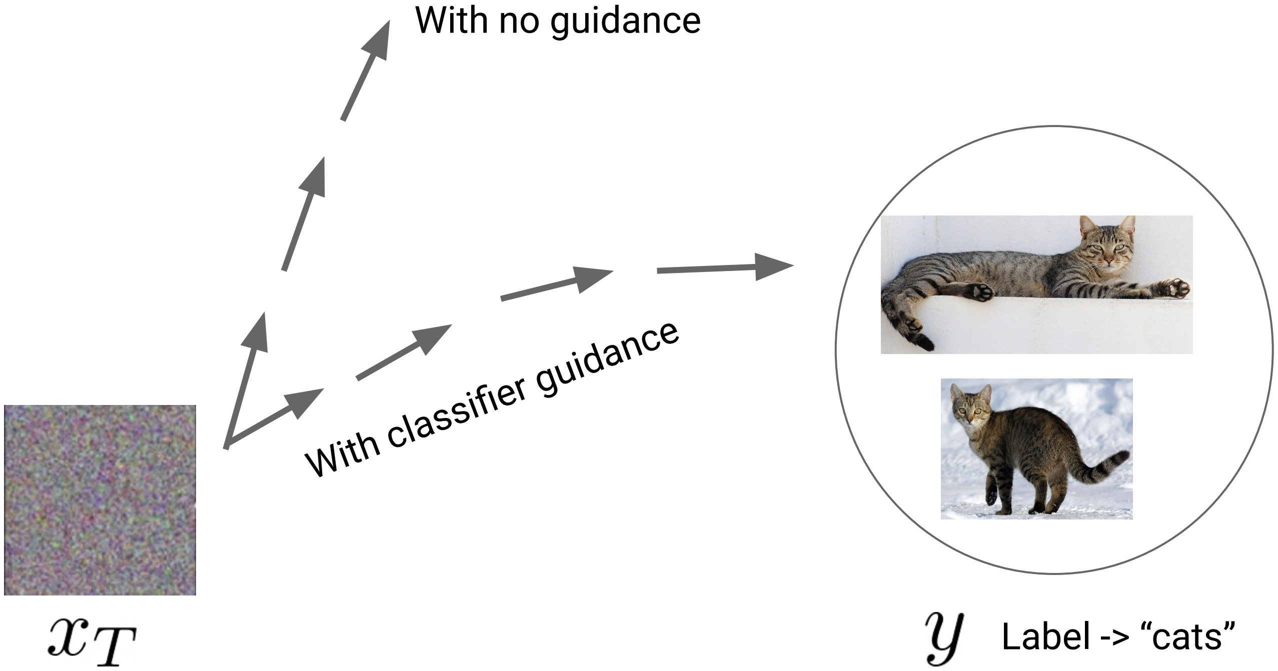
Algorithm 1 Training

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

Algorithm 2 Sampling

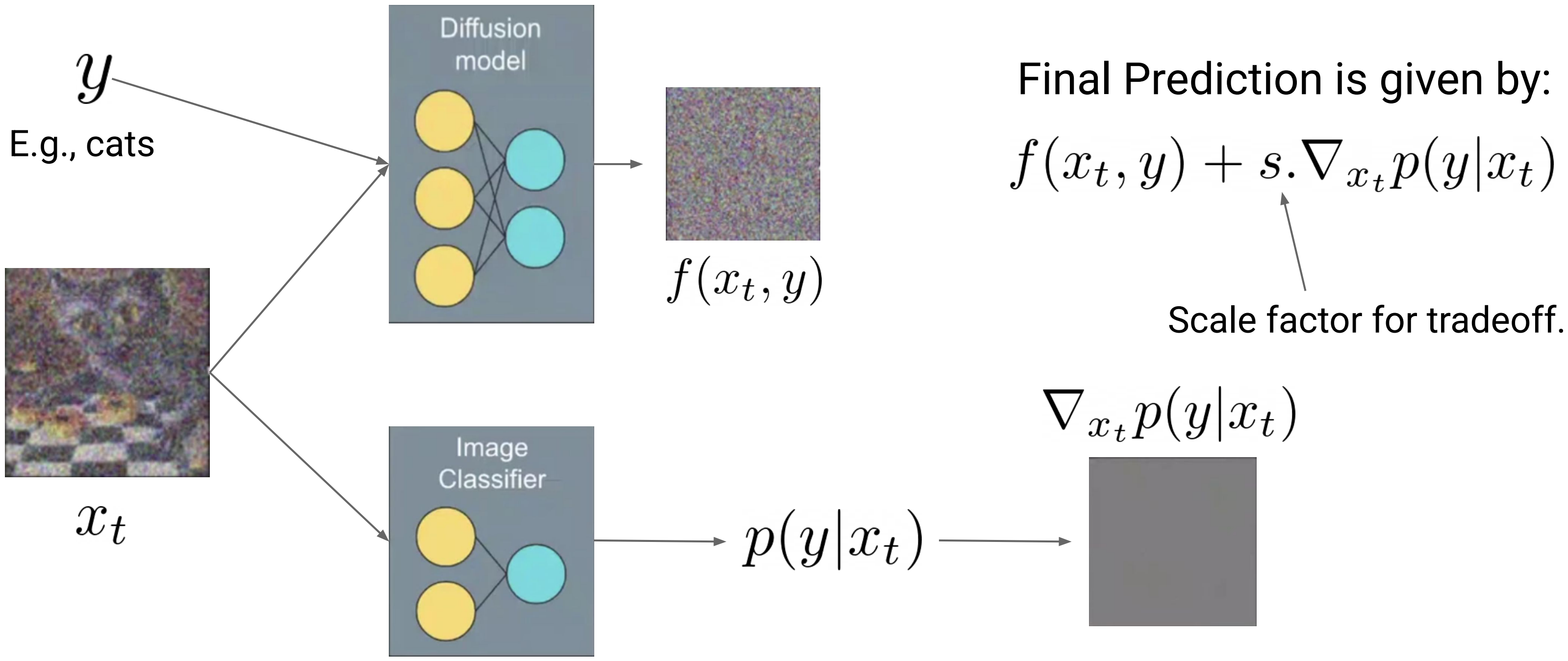
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ **do**
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,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

Classifier Guidance



[Diffusion Models Beats GANs on Image Synthesis (Dhariwal & Nichol 2021)]

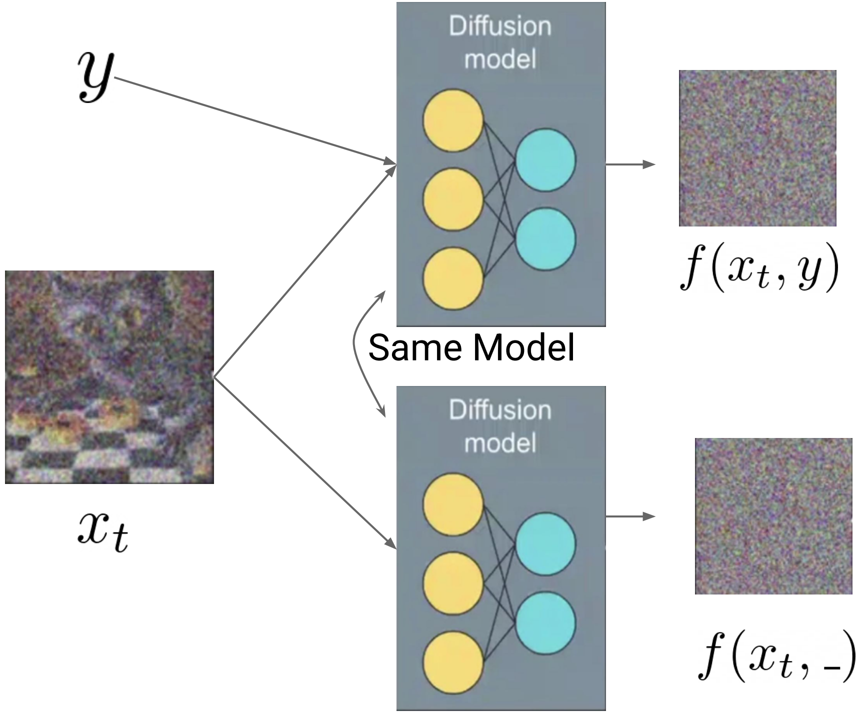
Classifier Guidance



- Classifier trained with noisy images

[Diffusion Models Beats GANs on Image Synthesis (Dhariwal & Nichol 2021)]

Classifier-Free guidance



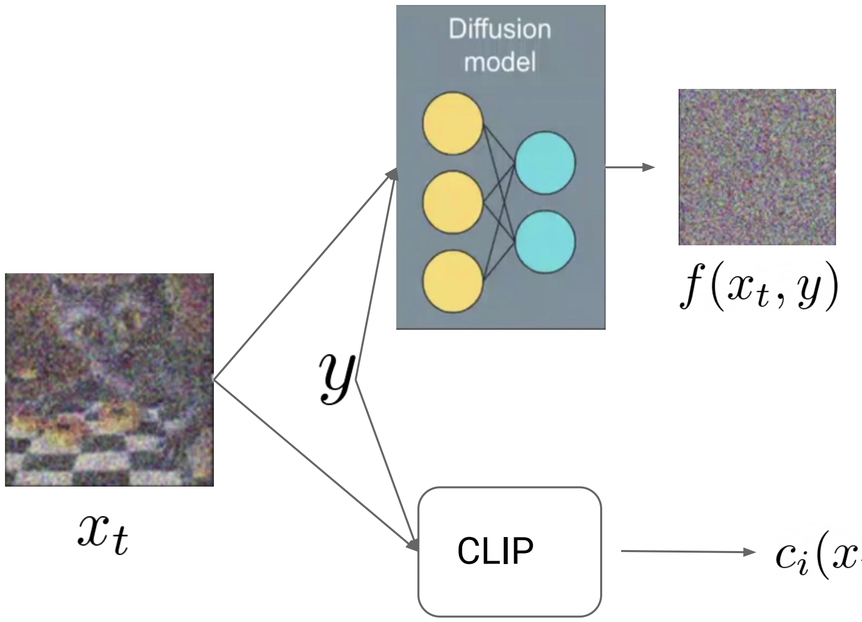
Final Prediction is given by:

$$f(x_t, y) + s.(f(x_t, y) - f(x_t, \_))$$

Scale factor for tradeoff.

[Classifier-Free Diffusion Guidance (Ho & Salimans 2021)]

CLIP Guidance



Final Prediction is given by:

$$f(x, y) + s.\nabla_{x_t} c_i(x_t)^T c_t(y)$$

Scale factor for tradeoff.

$$\nabla_{x_t} p(y|x_t)$$

$$c_i(x_t)^T c_t(y) \longrightarrow \nabla_{x_t} c_i(x_t)^T c_t(y)$$

- CLIP trained with noisy images

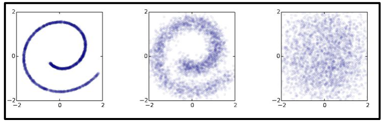
References for Diffusion Models

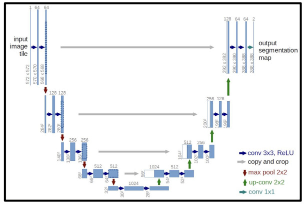
- Deep unsupervised learning of nonlinear thermodynamics, (Sohl-Dickstein et al. 2015).
- Denoising Diffusion Probabilistic Models (Ho et al. 2020)
- Diffusion Models Beats GANs on Image Synthesis, (Dhariwal & Nichol 2021)
- Classifier-Free Diffusion Guidance (Ho & Salimans 2021)
- Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding
- Improved Denoising Diffusion Probabilistic Models (Nichol & Dhariwal 2021)
- GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models (Ramesh et al. 2022)
- Understanding Diffusion Models: A Unified Perspective (Luo et al 2022)

Pieces of the Text-to-Image Puzzle

2015

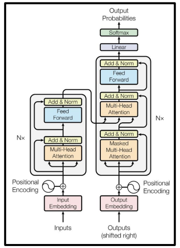
Diffusion





UNet

2015

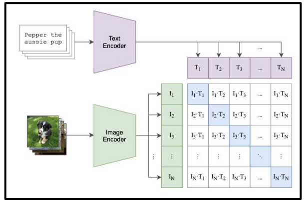


Transformers

2017

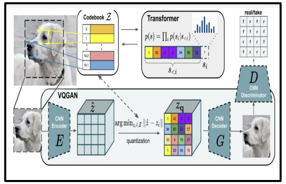
2020

CLIP



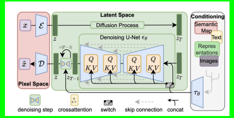
2021

DALL-E



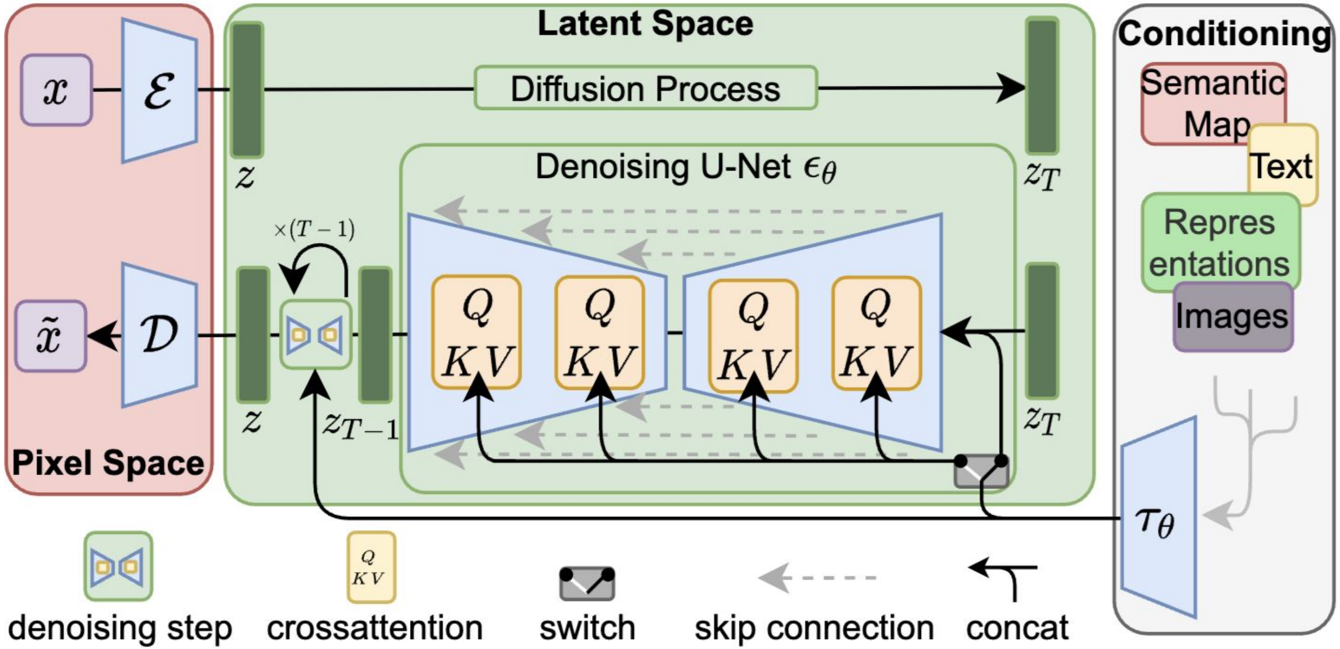
VQGAN

2021

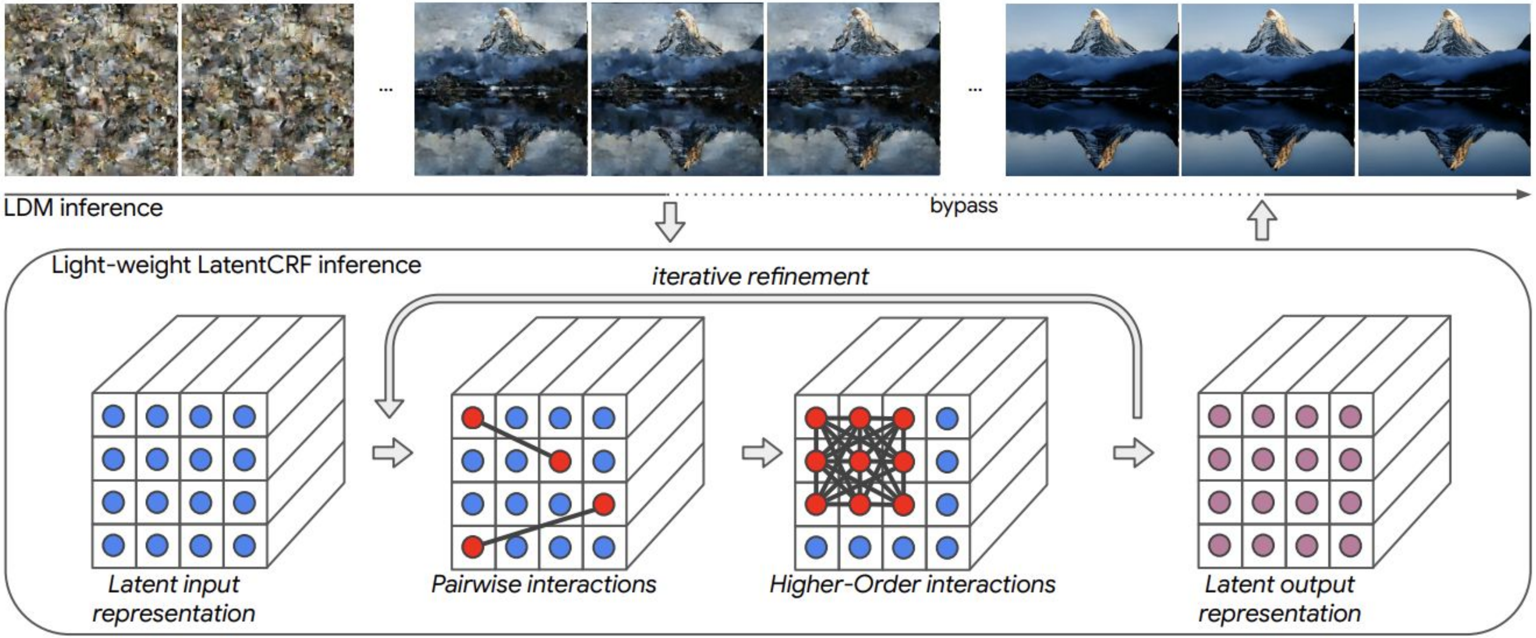


LDM **2021**

Latent diffusion models



LatentCRF for efficient inference



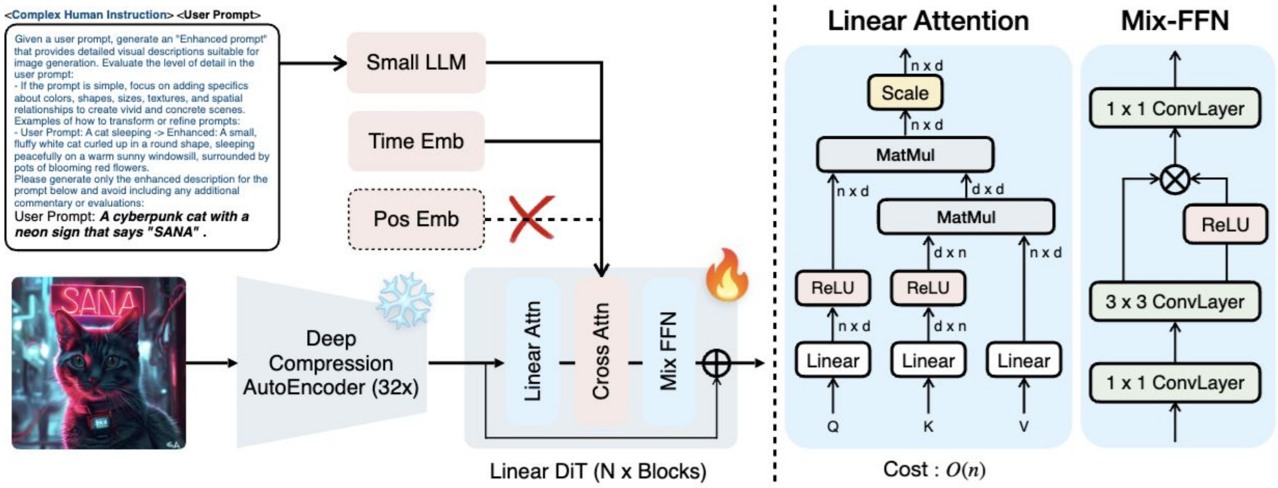
We replace several LDM inference iterations with CRF inference. Our LatentCRF modifies latent vectors with pairwise and higher-order interactions to better align with the distribution of natural image latents.

Ranasinghe et al. LatentCRF

LatentCRF for diffusion models

[Ranasinghe et al. LatentCRF]

SANA: Efficient High-Resolution Image Synthesis with Linear Diffusion Transformer



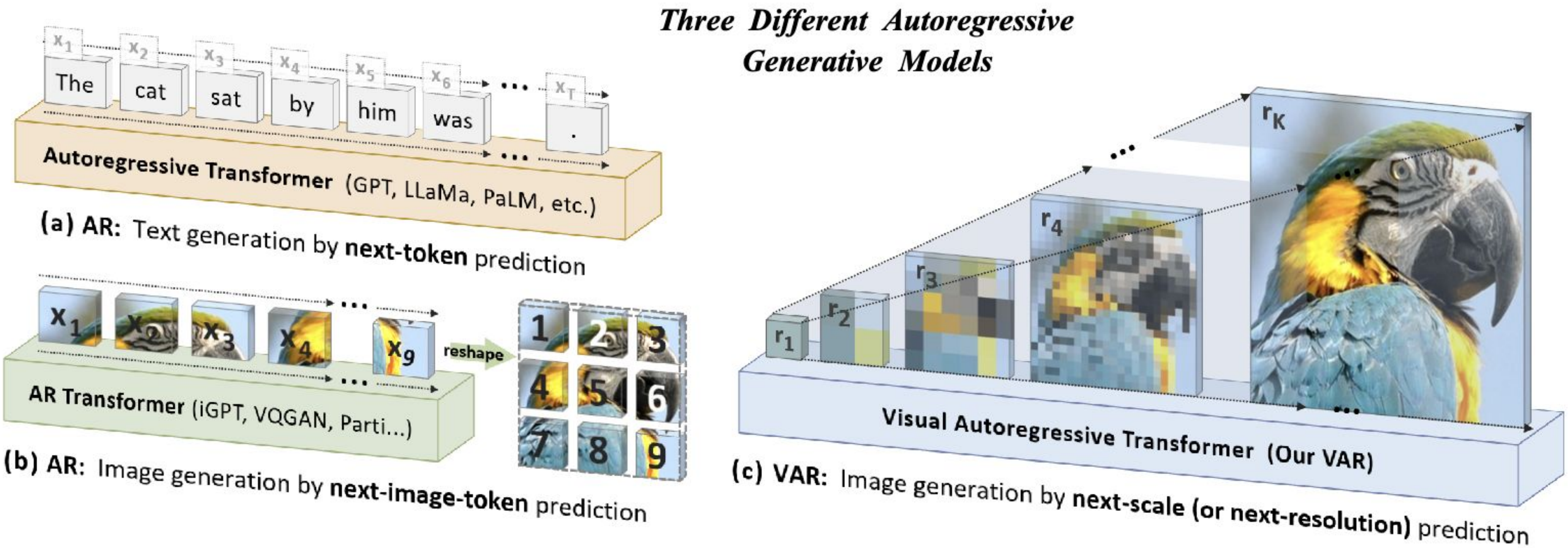
(a). Architecture overview of our Sana. (b). Linear DiT Module.

(a). Generation results from our 1.6B models

- 32x deep compression autoencoder (instead of 8x).
- Complex human instructino (CHI)
- Linear attention and Mix-FFN
- VLMs for image captioning

Xie et al. ICLR 2025.

Autoregressive modeling vs. VAR



[Tian et al. Visual Autoregressive Modeling: Scalable Image Generation via Next-Scale Prediction, 2024]

Discussion

- Image representation is the key - pixels verses latent vectors/tokenization
- Diffusion models have been shown very effectively via UNets or transformers.
- Larger datasets and GPU/TPU usage led to visually stunning generation results.
    - From 1.2M ImageNet to 5B Laion dataset
    - Hundreds of GPU hours for training
  - It is extremely important to cut costs of these inference algorithms
    - Sadeep will be talking about MarkovGen and CMMD
    - Ameesh will be presenting spectral autoencoders and techniques for efficient video generation.