

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

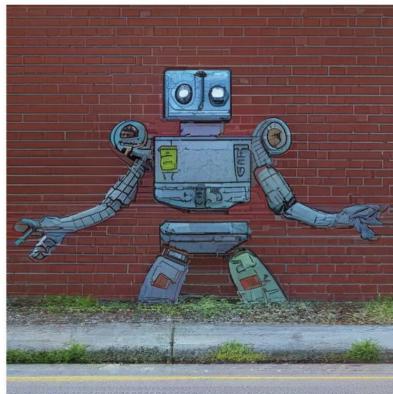
Overview

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 - 10.45	Break	
11.00 - 11.45	Sadeep Jayasumana	MarkovGen: Structured Prediction for fast text-to-image generation
11:45 - 12:30	Ameesh Makadia	Recent trends in text-to-3D 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 the wall, grass is growing out of cracks in the concrete.



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



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

Text to video Generation



Lumiere



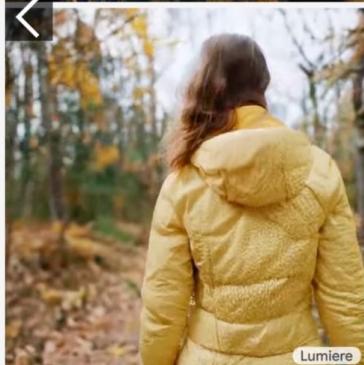
Lumiere



Lumiere



Lumiere



Lumiere



Lumiere



Lumiere



Lumiere



Text-to-3D Generation

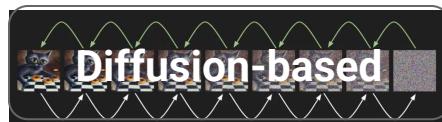


<https://dreamfusion3d.github.io/>

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.



Transformers and Diffusion models



Elon Musk
@elonmusk

Subscribe

...

Who should be President in 2032?

Transformers

77.4%

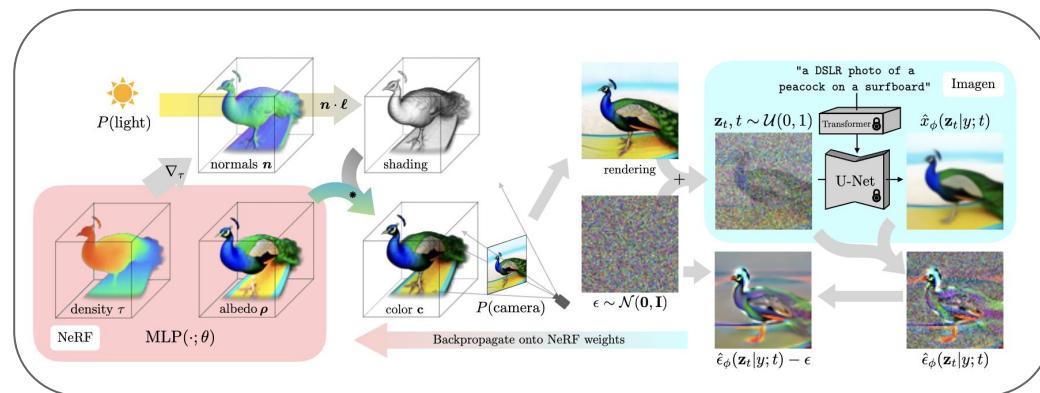
Diffusion

22.6%

1,178,197 votes · Final results

t2i models are centerpieces of many generative models

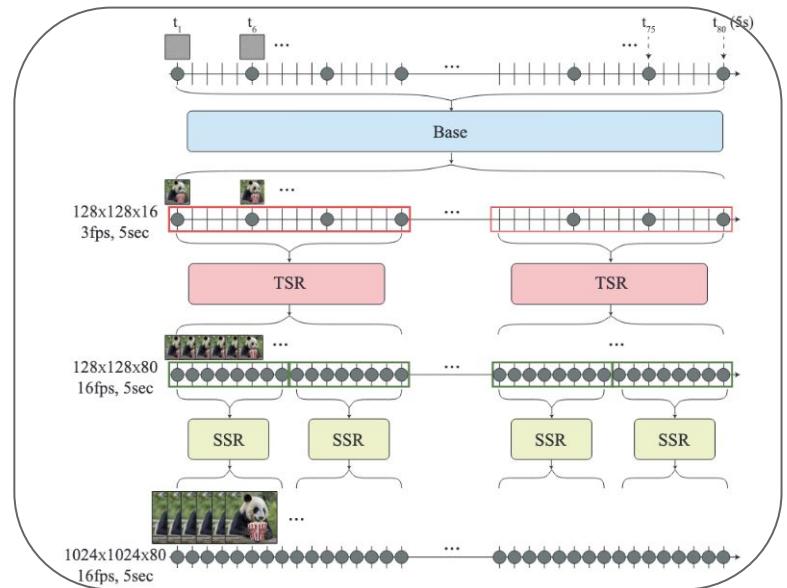
Text-to-3D



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

<https://dreamfusion3d.github.io/>

Text-to-Video



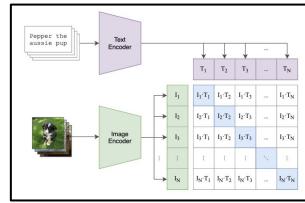
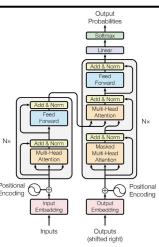
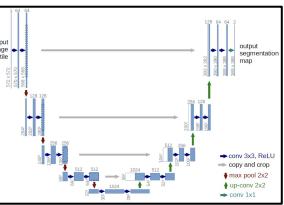
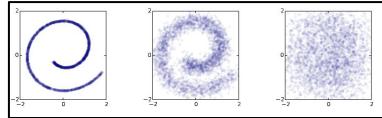
Generate distinct keyframes using text-to-image model, followed by temporal and spatial super-resolution models.

Bar-Tal et al. Lumiere, 2024

Pieces of the Text-to-Image Puzzle

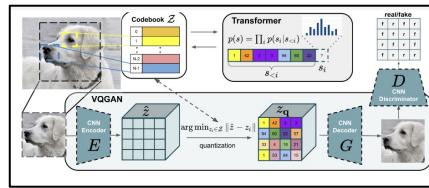
2015

Diffusion



2021

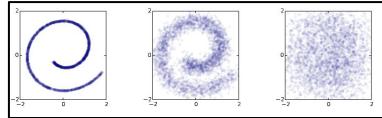
DALL-E



Pieces of the Text-to-Image Puzzle

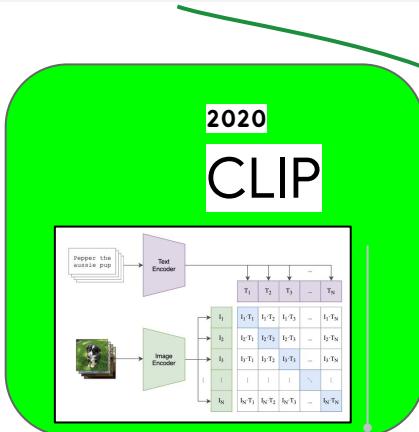
2015

Diffusion



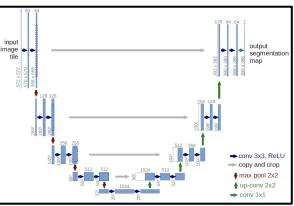
2020

CLIP



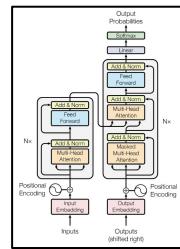
2021

DALL-E

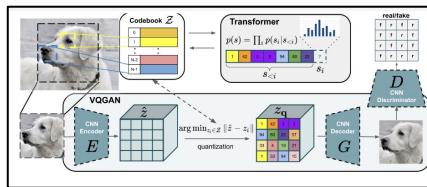


UNet

2015



Transformers
2017



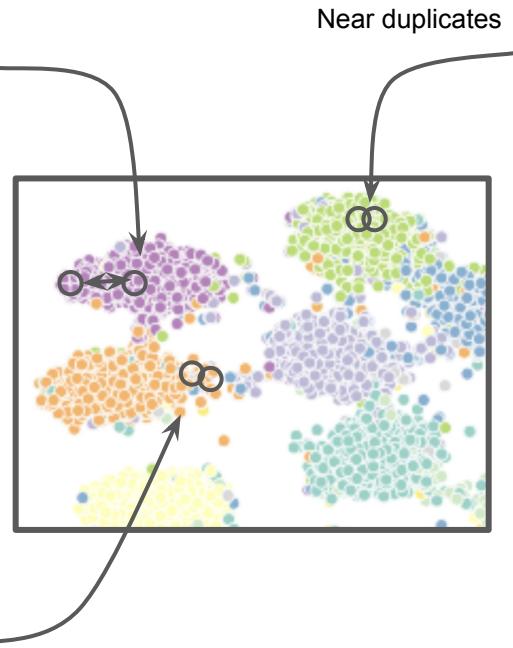
VQGAN

2021

Image features with similar objects are close



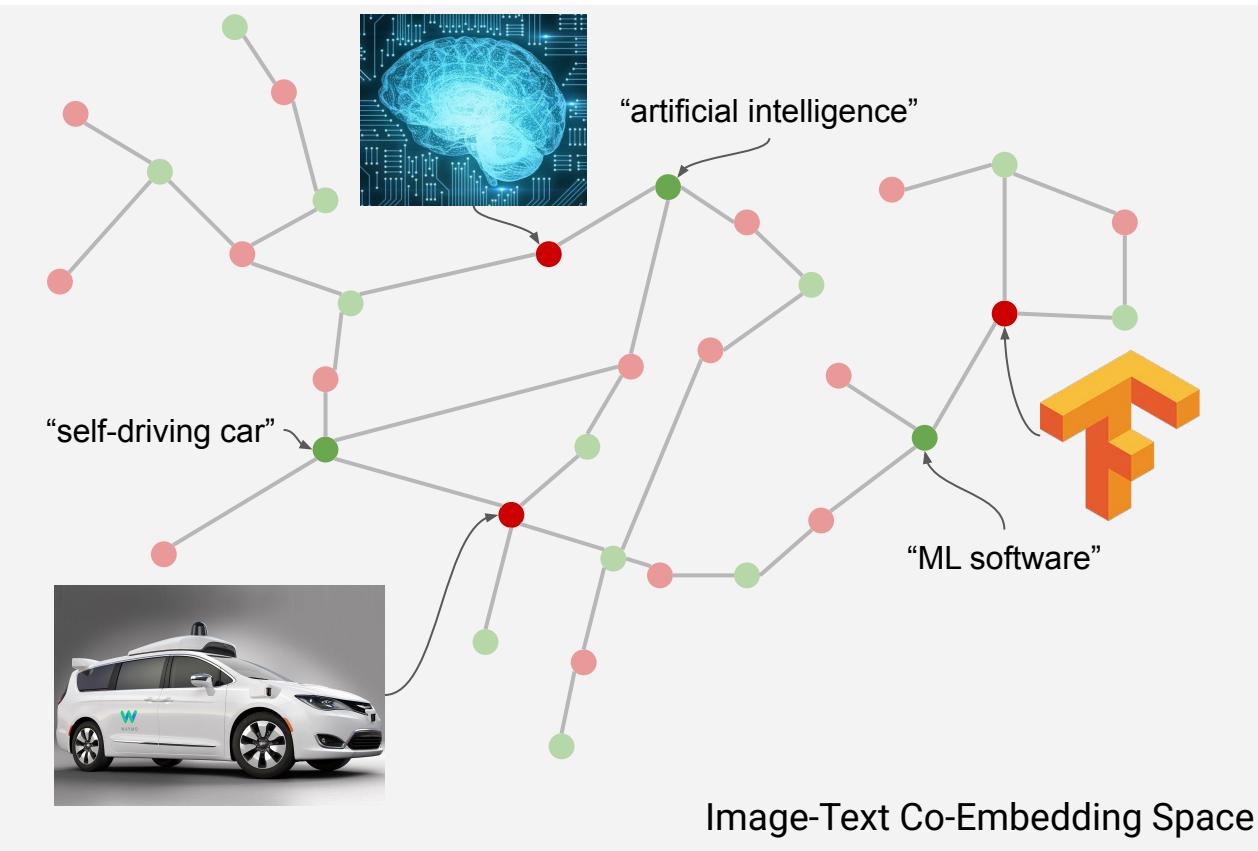
(red wolf, maned wolf,
Canis rufus, *Canis niger*) (timber wolf, grey wolf,
Canis lupus, *Canis lupus*)



Sussex spaniel

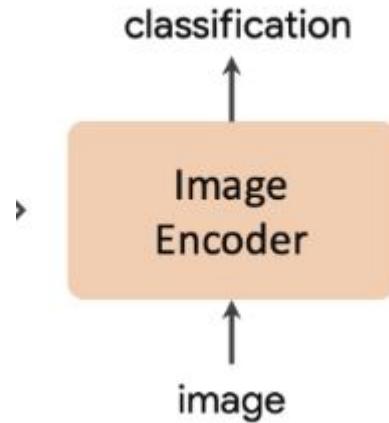
Features corresponding to images containing same semantic objects are close to each other in the embedding space.

Image-Text Co-Embedding Spaces

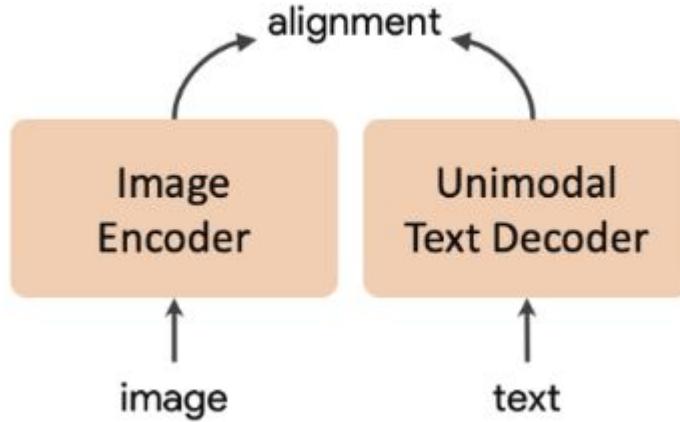


Bipartite mapping between image and text embeddings

Single tower vs. two tower models



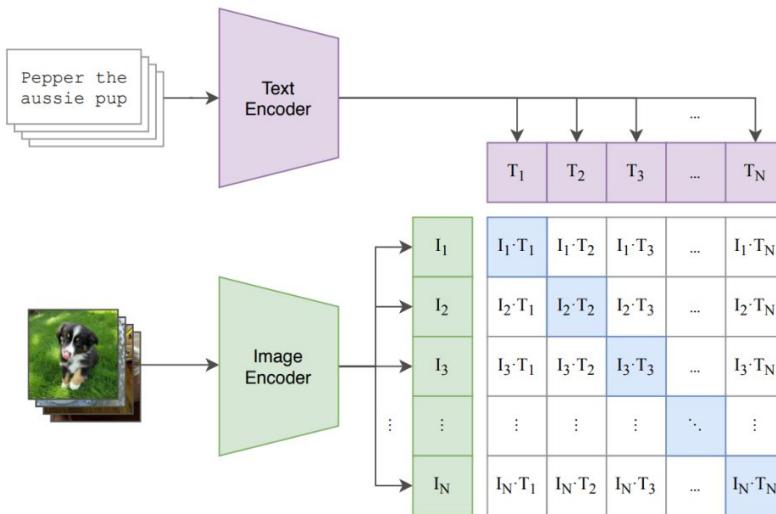
Single-tower classification with ResNets or ViTs trained on a chosen set of labels such as in ImageNet.



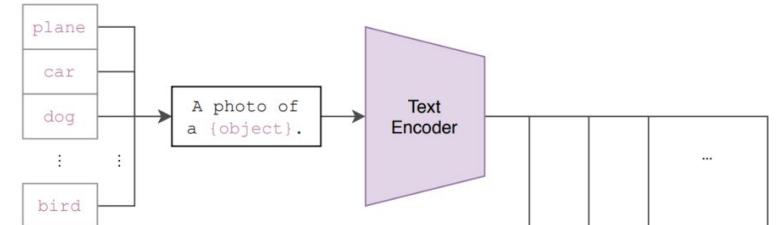
Learning two tower models allows us to use zero-shot classification methods on different classes.

CLIP/ALIGN

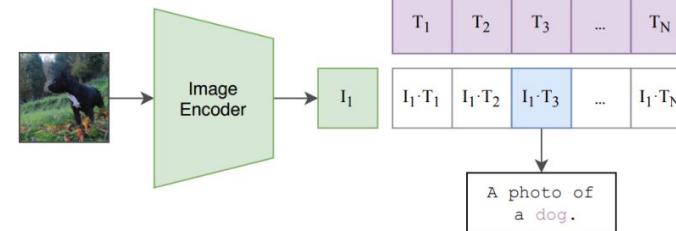
(1) Contrastive pre-training



(2) Create dataset classifier from label text



(3) Use for zero-shot prediction



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

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

x_i, y_i Image and Text normalized embeddings

Text-Image Coembedding References

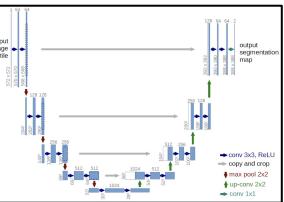
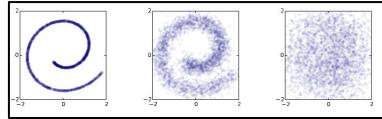
[CLIP] [Learning Transferable Visual Models From Natural Language Supervision](#), 2021.

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

Pieces of the Text-to-Image Puzzle

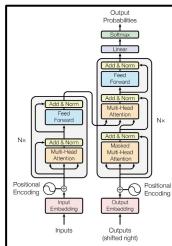
2015

Diffusion

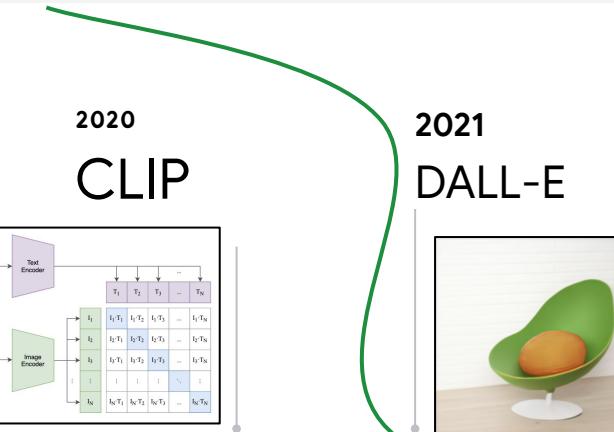


UNet

2015



Transformers
2017

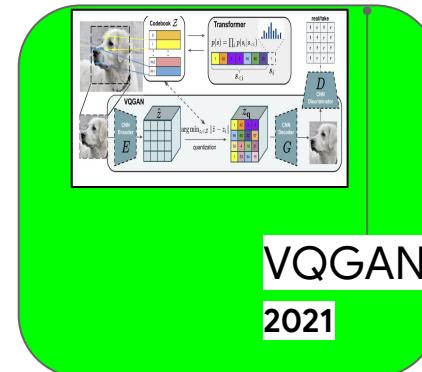


2020

CLIP

2021

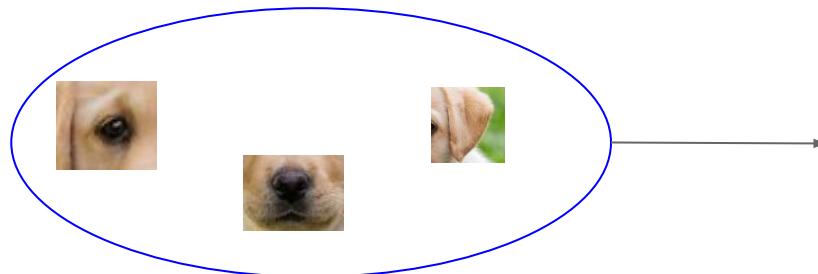
DALL-E



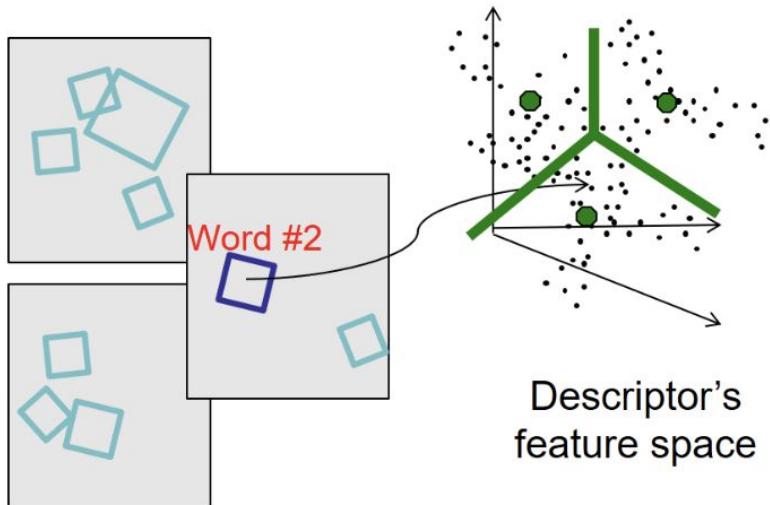
VQGAN
2021

Background: Visual Words

Individual parts of an object reveal a lot of information.

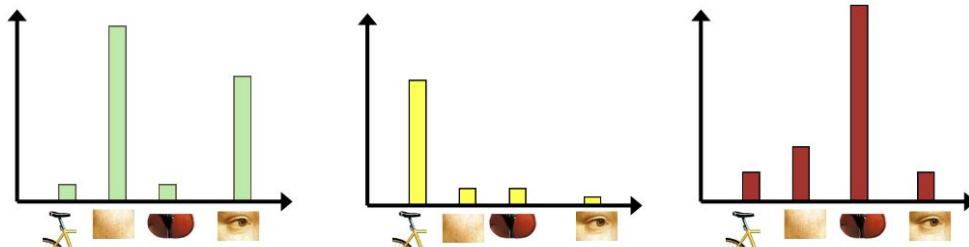


Background: Visual words



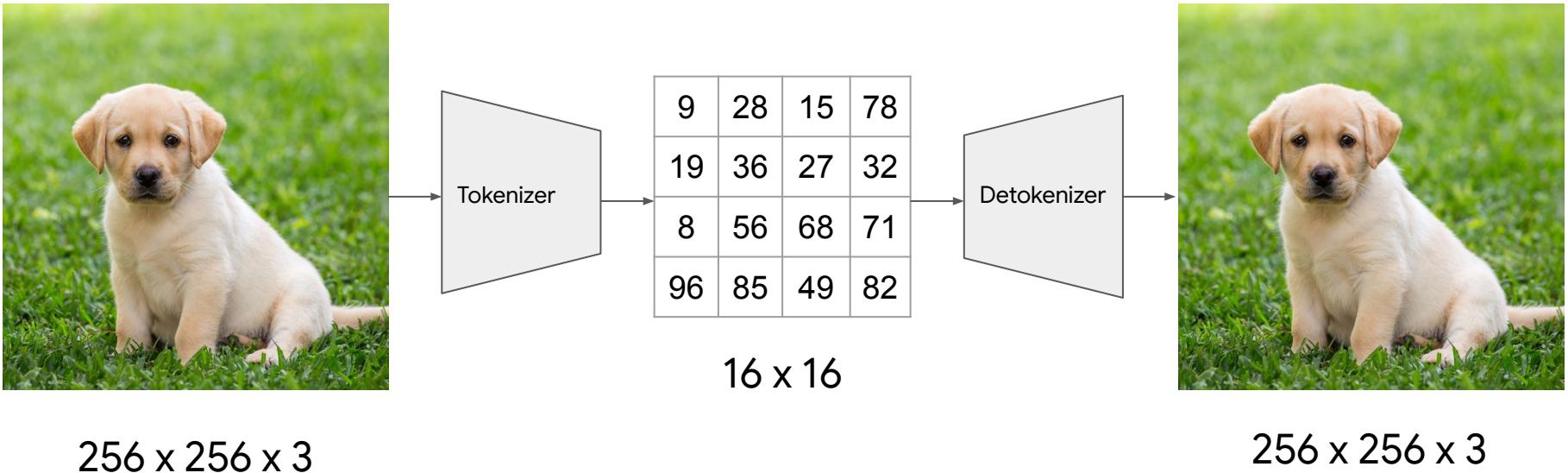
- Quantize via clustering, let cluster centers be the prototype “words”
- Determine which word to assign to each new image region by finding the closest cluster center.

Background: Visual words



Source: Kristen Grauman

Image Tokenization

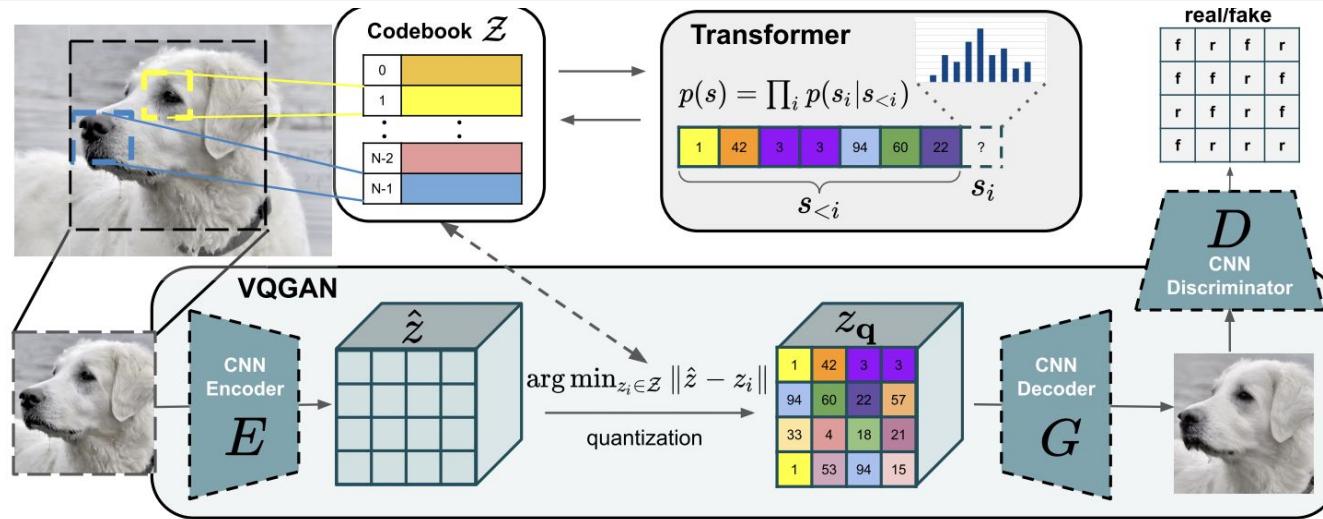


Key Idea in VQGAN

- 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](#)

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



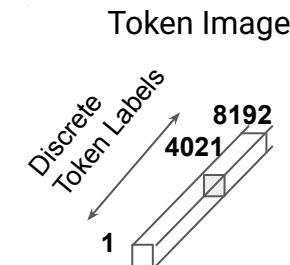
$$\mathcal{Z} = \{z_k\}_{k=1}^K \subset \mathbb{R}^{n_z}$$

Discrete codebook consisting of K vectors

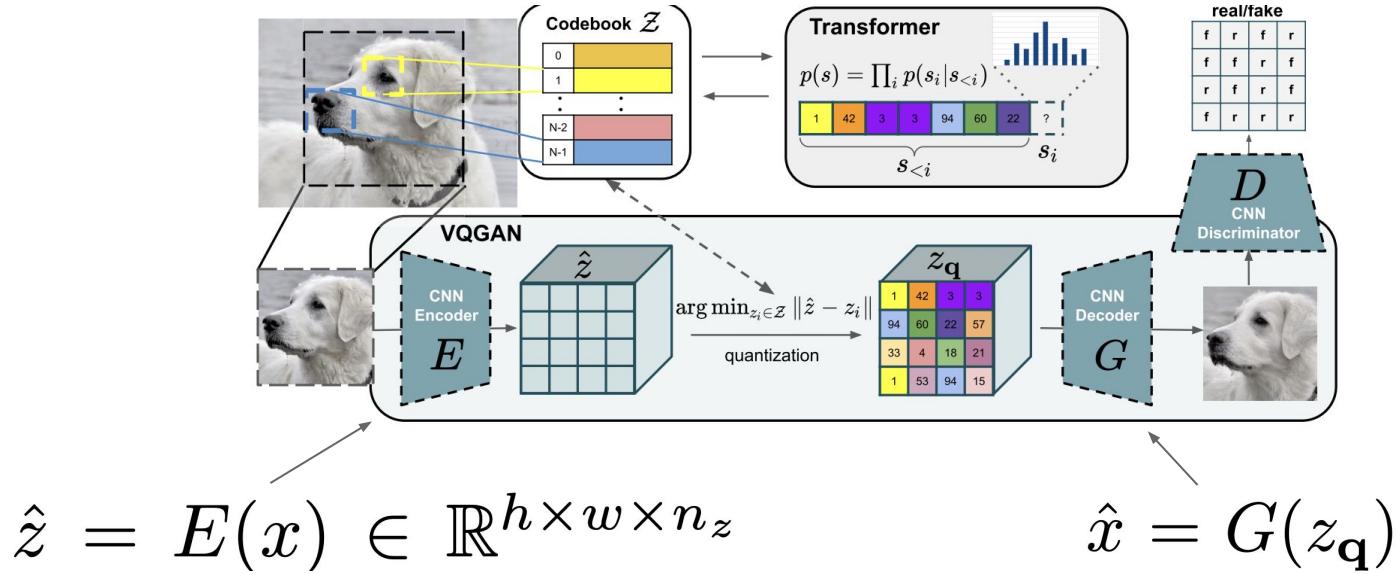
$$z_{\mathbf{q}} \in \mathbb{R}^{h \times w \times n_z}$$

Image represented with codebook entries

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



Codebook



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

Learning the codebook

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

$$\begin{aligned}\mathcal{L}_{\text{vQ}}(E, G, \mathcal{Z}) = & \|x - \hat{x}\|^2 + \boxed{\|\text{sg}[E(x)] - z_{\mathbf{q}}\|_2^2} \\ & + \|\text{sg}[z_{\mathbf{q}}] - E(x)\|_2^2\end{aligned}$$

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.

Learning a perceptually rich codebook

GAN Loss:

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

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

$$\mathcal{Q}^* = \arg \min_{E, G, \mathcal{Z}} \max_D \mathbb{E}_{x \sim p(x)} \left[\mathcal{L}_{\text{VQ}}(E, G, \mathcal{Z}) + \lambda \mathcal{L}_{\text{GAN}}(\{E, G, \mathcal{Z}\}, D) \right]$$

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

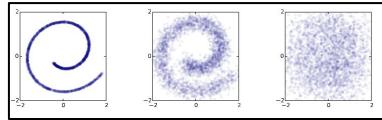
Feature Codebook References

- [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

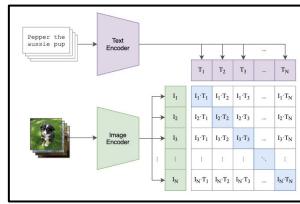
2015

Diffusion



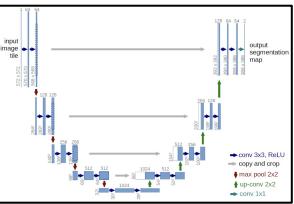
2020

CLIP



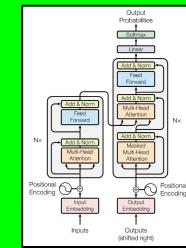
2021

DALL-E

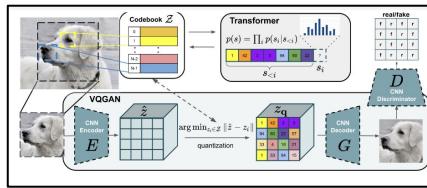


UNet

2015

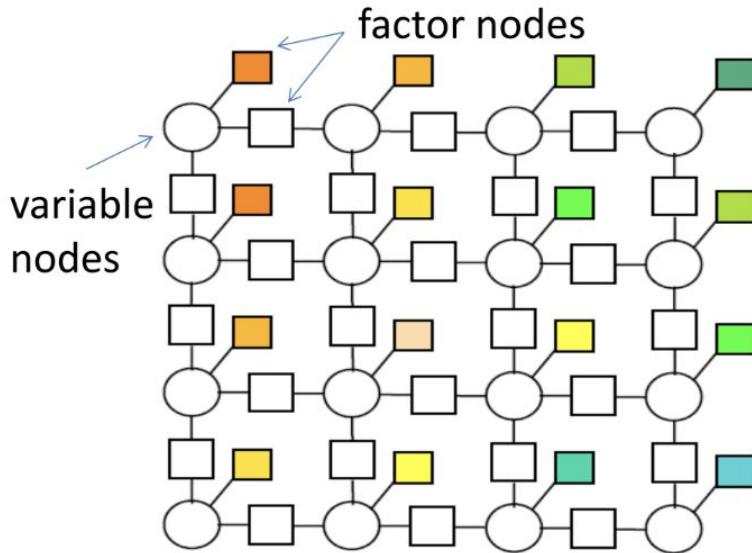


Transformers
2017



VQGAN
2021

Markov Random Fields (MRFs)



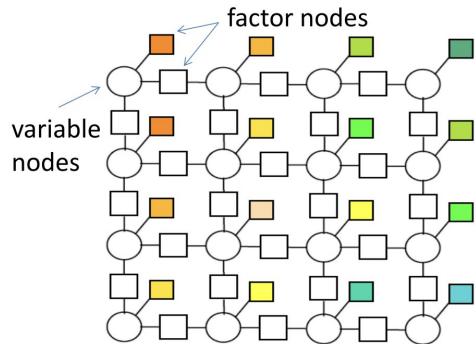
Goal: find most probable interpretation of scene

Minimize an energy function:

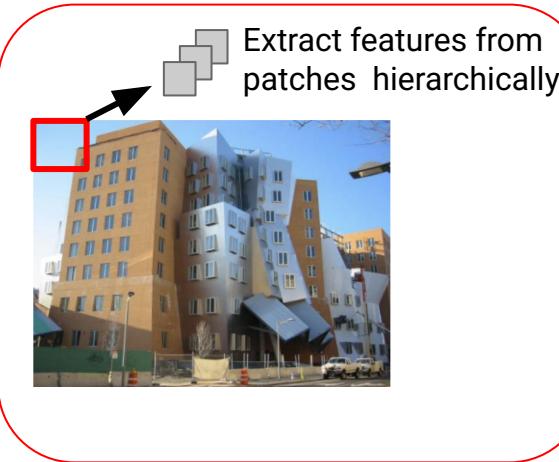
$$E(\mathbf{x}) = \text{unary_cost} + \text{pairwise_cost}$$

- Solve 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!

Long-range interaction!



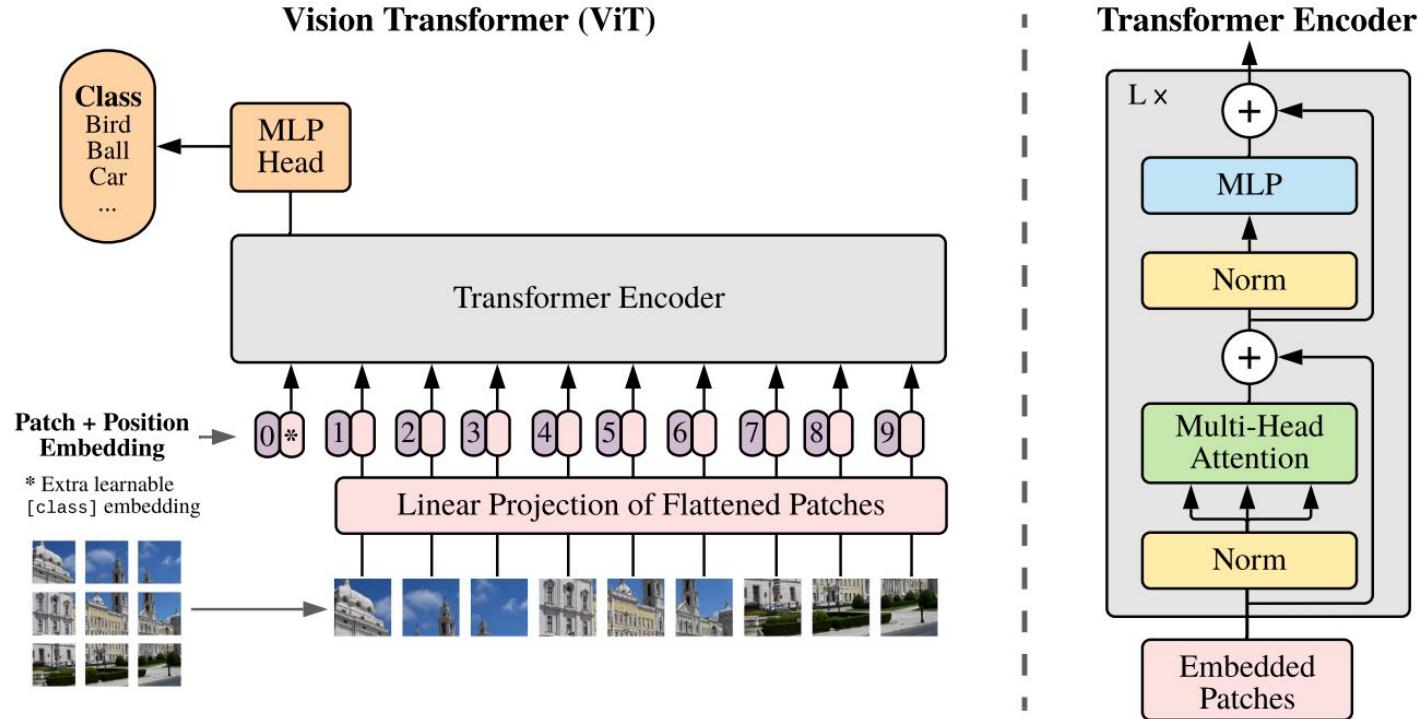
Transformers allow long range
interactions!

Graphcuts
1999

AlexNet
2012

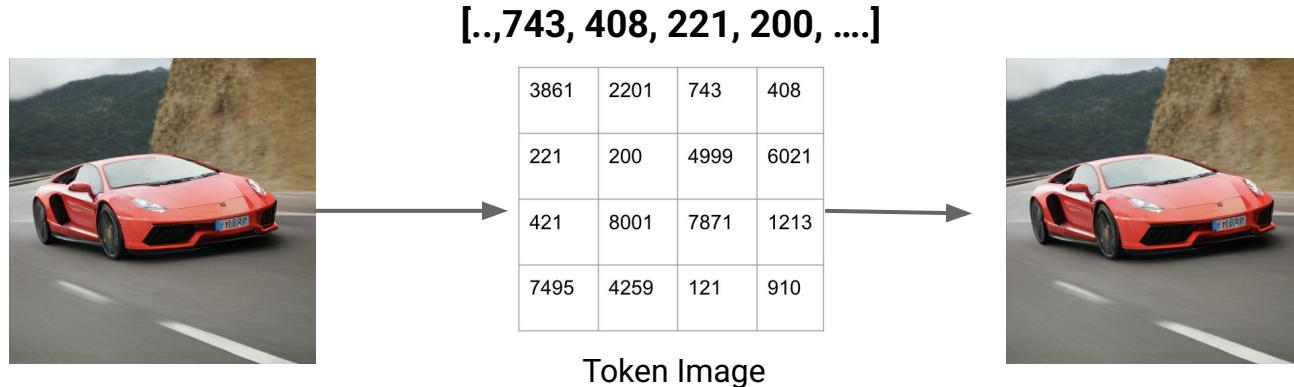
Transformers
2017

Vision Transformer



Conditioned Synthesis using Transformers

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



- 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

row major			
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

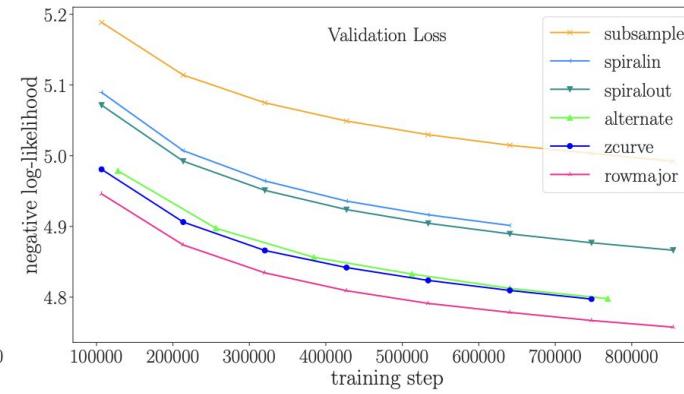
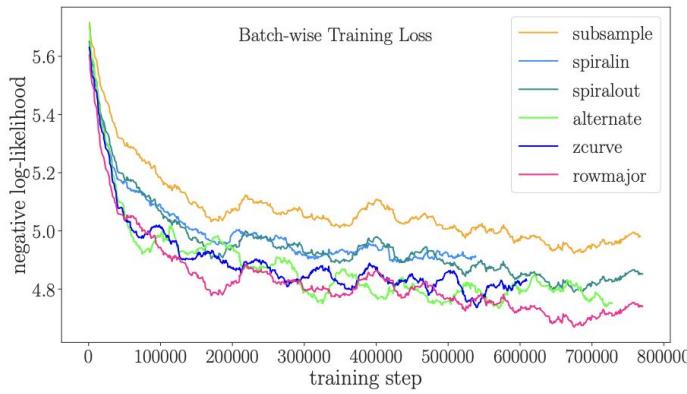
spiral in			
0	11	10	9
1	12	15	8
2	13	14	7
3	4	5	6

spiral out			
15	4	5	6
14	3	0	7
13	2	1	8
12	11	10	9

z-curve			
0	1	4	5
2	3	6	7
8	9	12	13
10	11	14	15

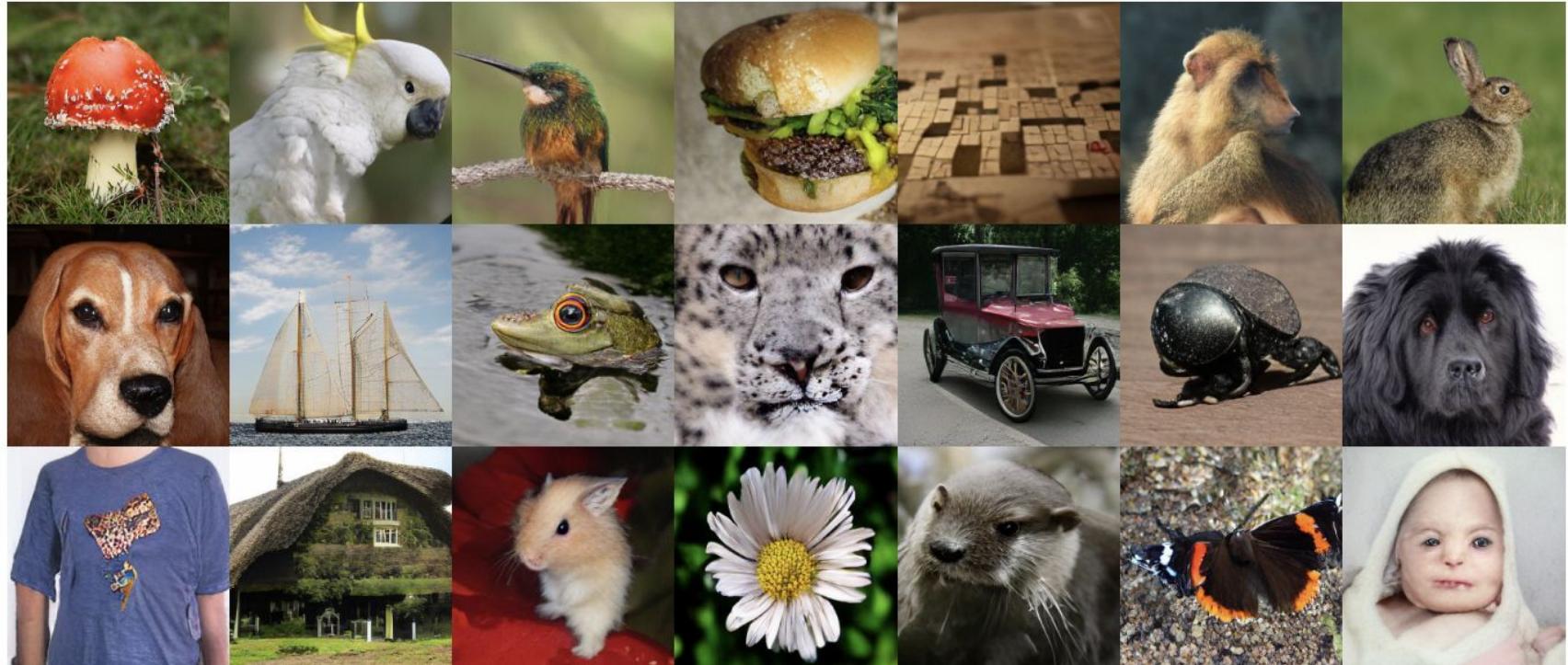
subsample			
0	4	1	5
8	12	9	13
2	6	3	7
10	14	11	15

alternate			
0	1	2	3
7	6	5	4
8	9	10	11
15	14	13	12



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

Class conditioned Image Synthesis



256x256 images conditioned on ImageNet

Conditioned Image Synthesis



Depth -> Image

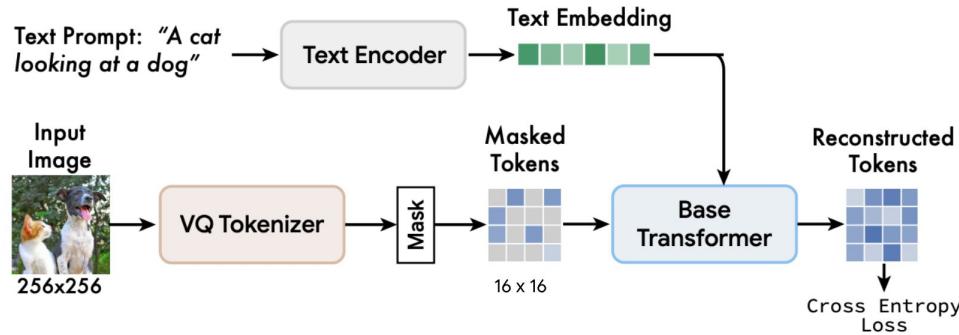


Low res. -> High res.
(Superresolution)

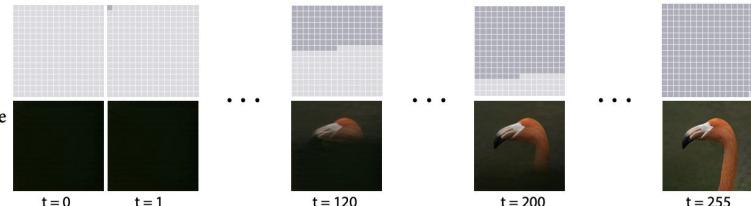
Semantic -> Image

Edge -> Image

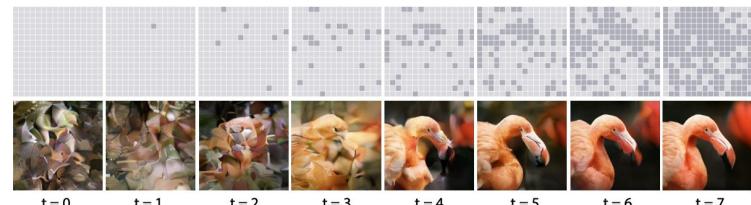
Efficient Text-to-Image Generation using Muse



Sequential
Decoding
with Autoregressive
Transformers



Scheduled
Parallel
Decoding
with MaskGIT



MarkovGen: MRFs to speedup Muse

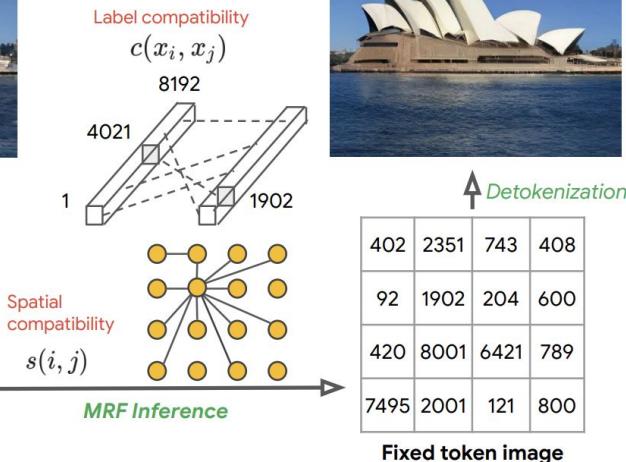
$$E(\mathbf{x}) = \text{unary_cost} + \text{pairwise_cost}$$



↑ Detokenization

4021	2351	743	408
221	1902	4999	600
420	8001	6421	1213
7495	2001	121	900

Imperfect token Image



MRF: Model Formulation

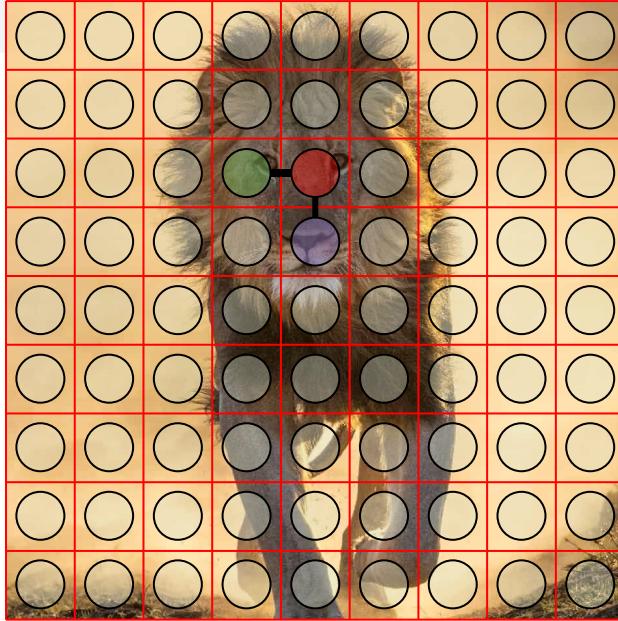
$$E(\mathbf{x}) = \text{unary_cost} + \text{pairwise_cost}$$

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.



$$\text{cost}(X_i = l) = -\text{logit}_i(l)$$

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

Speedup over Muse without quality loss.

Full Muse: All steps



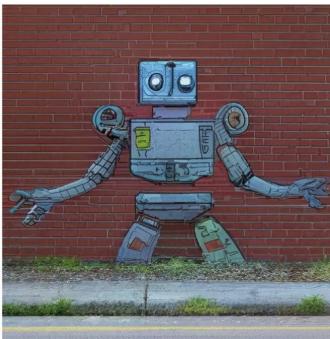
Early Exit Muse: Fewer steps
1.5x faster



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

Full Muse: All steps



MarkovGen: Fewer steps + MRF
1.5x faster

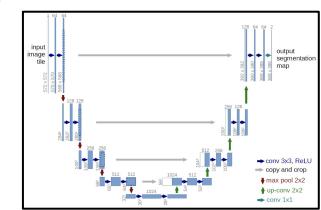
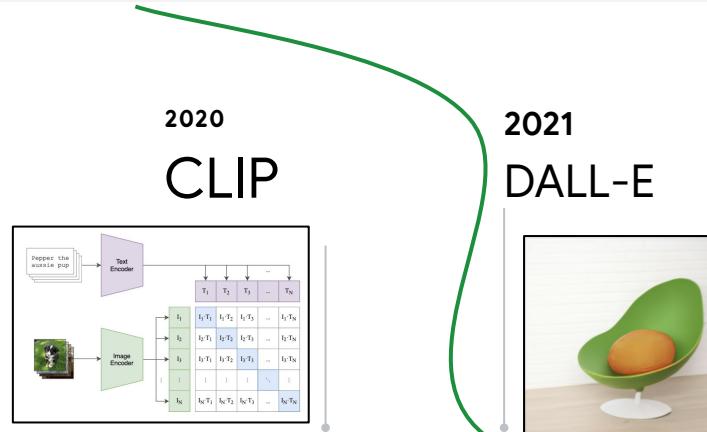
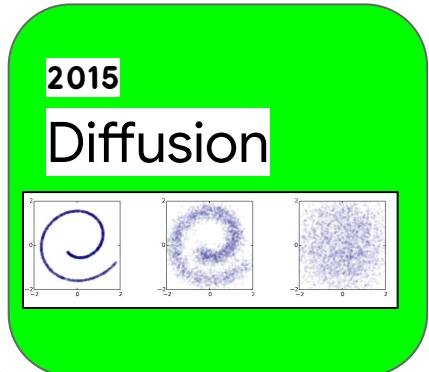


A robot cooking in the kitchen

MRF and Transformers References

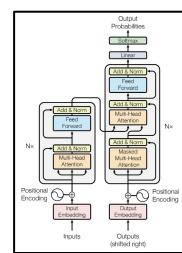
- Masked generative image transformer. In: CVPR (2022)
- Muse:Text-to-image generation via masked generative transformers. ICML (2023)
- Markovgen: Structured prediction for efficient text-to-image generation (2023)
- Hierarchical text-conditional image generation with clip latents. preprint (2022)
- 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

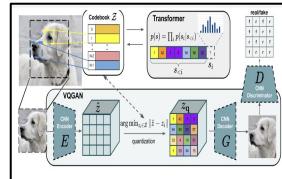


UNet

2015



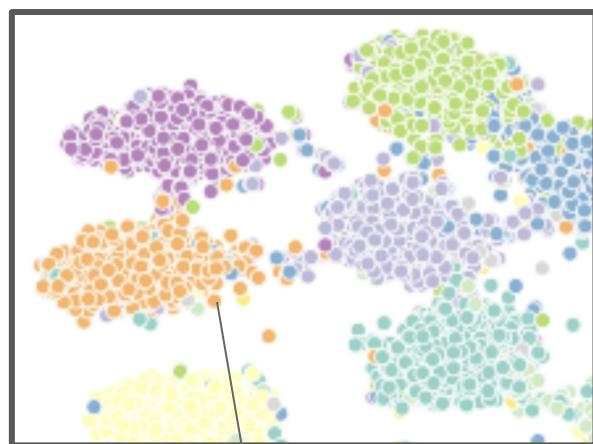
Transformers
2017



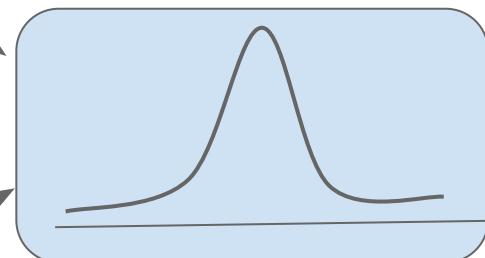
VQGAN

2021

Basic idea -> Diffusion Model



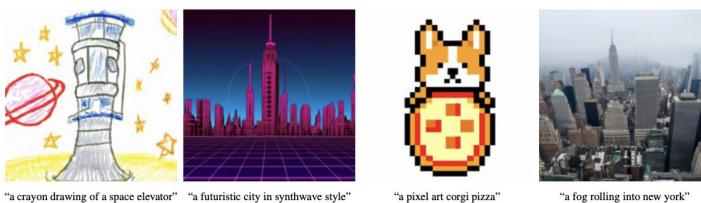
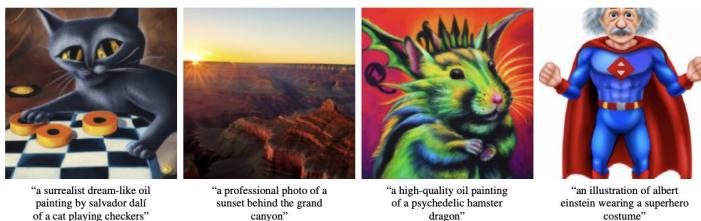
Forward diffusion



Reverse diffusion

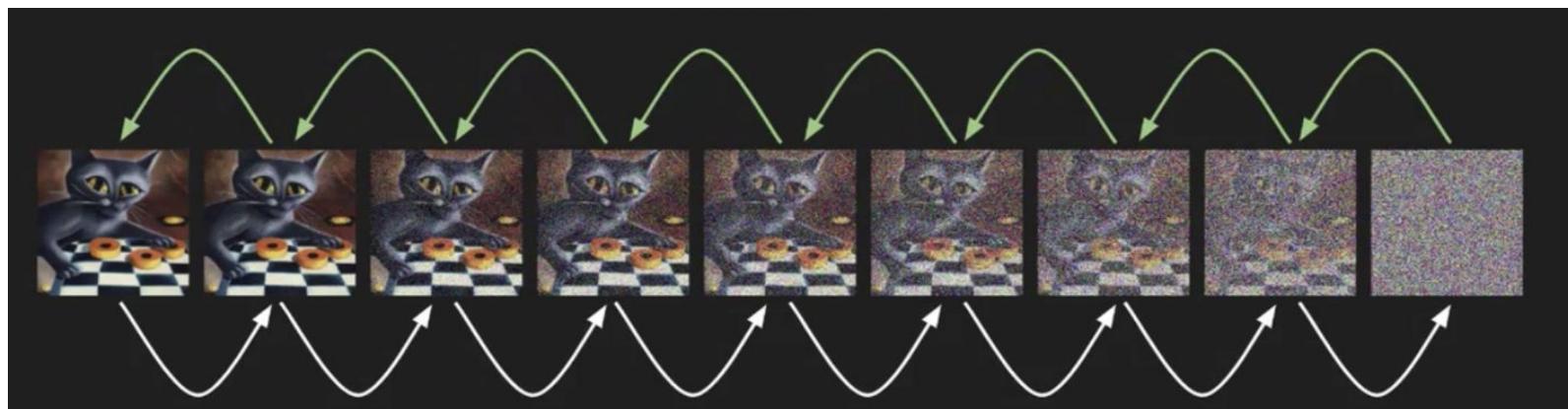


Diffusion Models



Background: Diffusion models

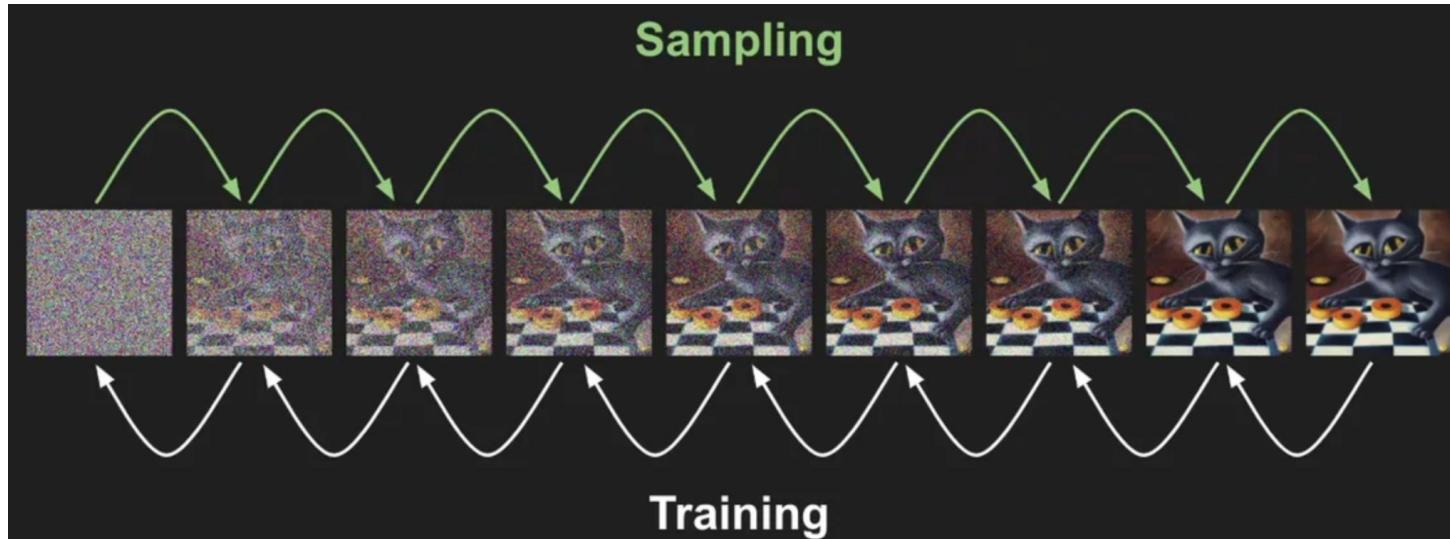
“Systematically and slowly destroy structure in a data distribution through an iterative forward diffusion process.



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]

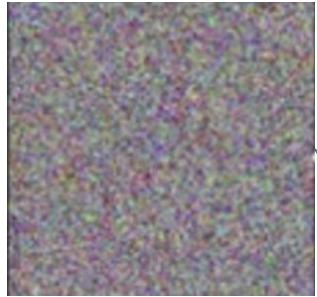
Background: Diffusion models



- 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: $\|\epsilon - \text{pred}\|^2$



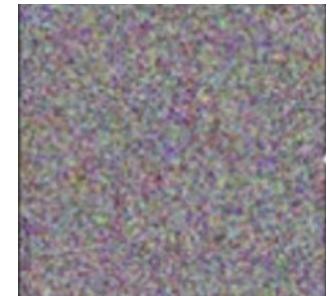
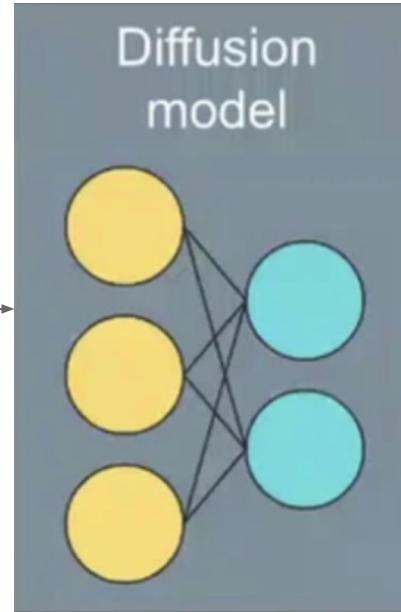
noise ϵ



image x_0



Noised image x_t



pred

Training Diffusion models



Sample an image from the data distribution

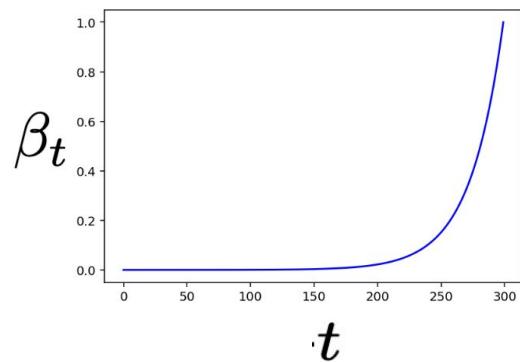
Training Diffusion models



Sample an image from the data distribution

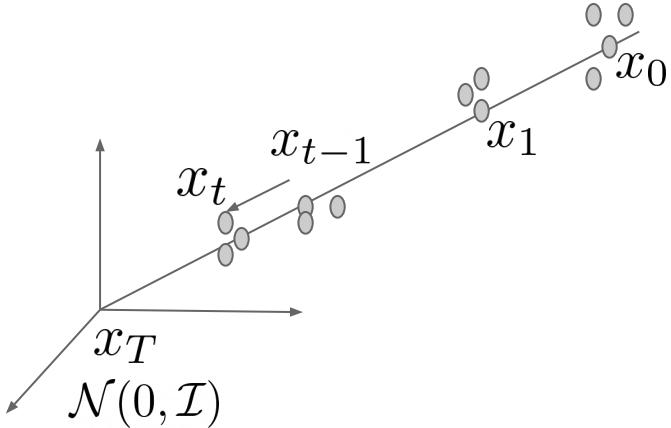
Markov chain of latent variables by progressively adding Gaussian noise.

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



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 $\xrightarrow{\quad}$

Sample an
image from the
data distribution

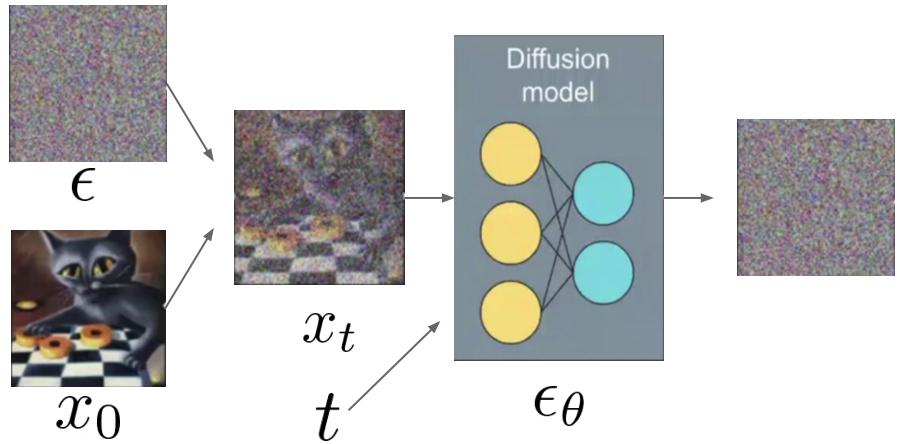
Markov chain of latent variables by progressively adding Gaussian noise.

$$\alpha_t := 1 - \beta_t \quad \bar{\alpha}_t := \prod_{s=1}^t \hat{\alpha}_s$$

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

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

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

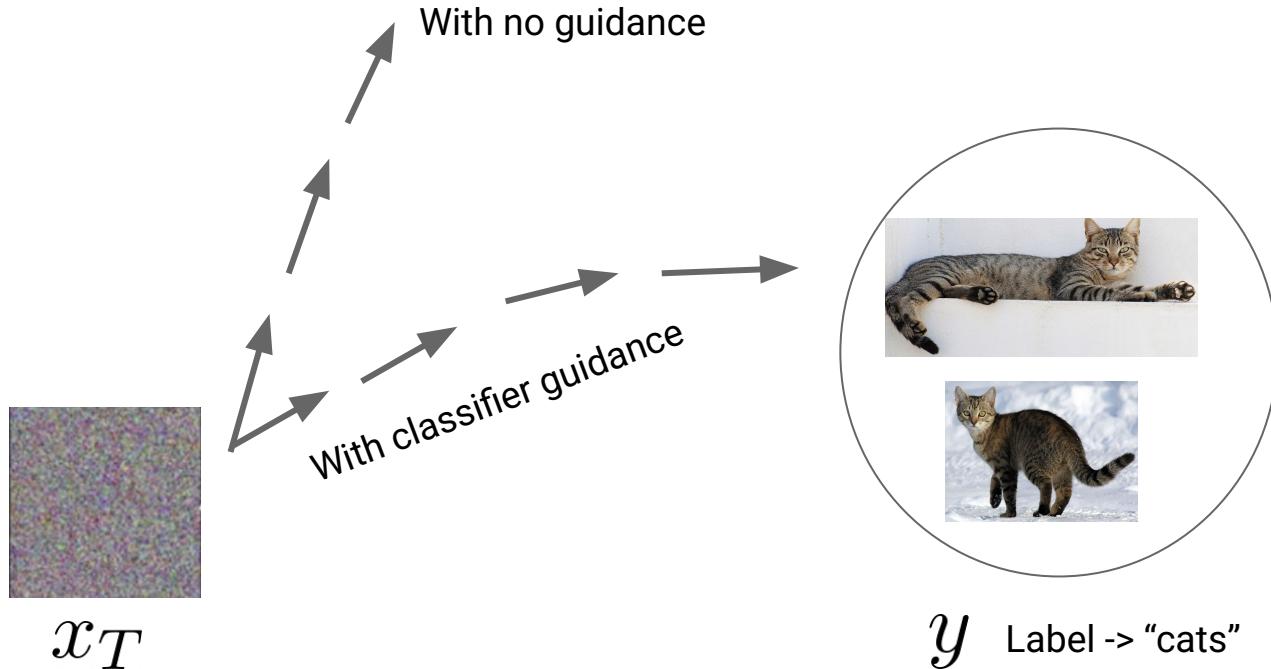
```
1: repeat
2:    $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ 
3:    $t \sim \text{Uniform}(\{1, \dots, T\})$ 
4:    $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
5:   Take gradient descent step on
      
$$\nabla_{\theta} \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta}(\sqrt{\bar{\alpha}_t}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

6: until converged
```

Algorithm 2 Sampling

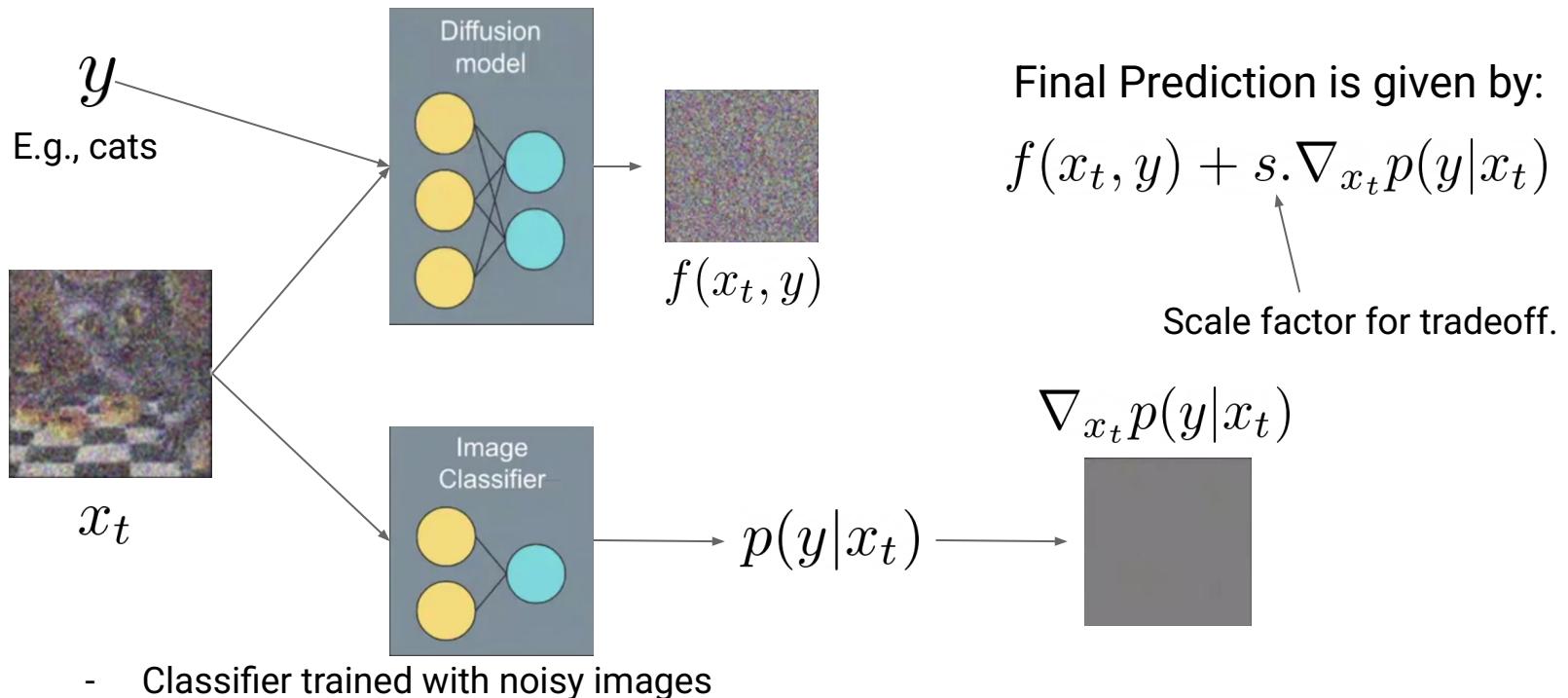
```
1:  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
2: for  $t = T, \dots, 1$  do
3:    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  if  $t > 1$ , else  $\mathbf{z} = \mathbf{0}$ 
4:    $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1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

Classifier Guidance

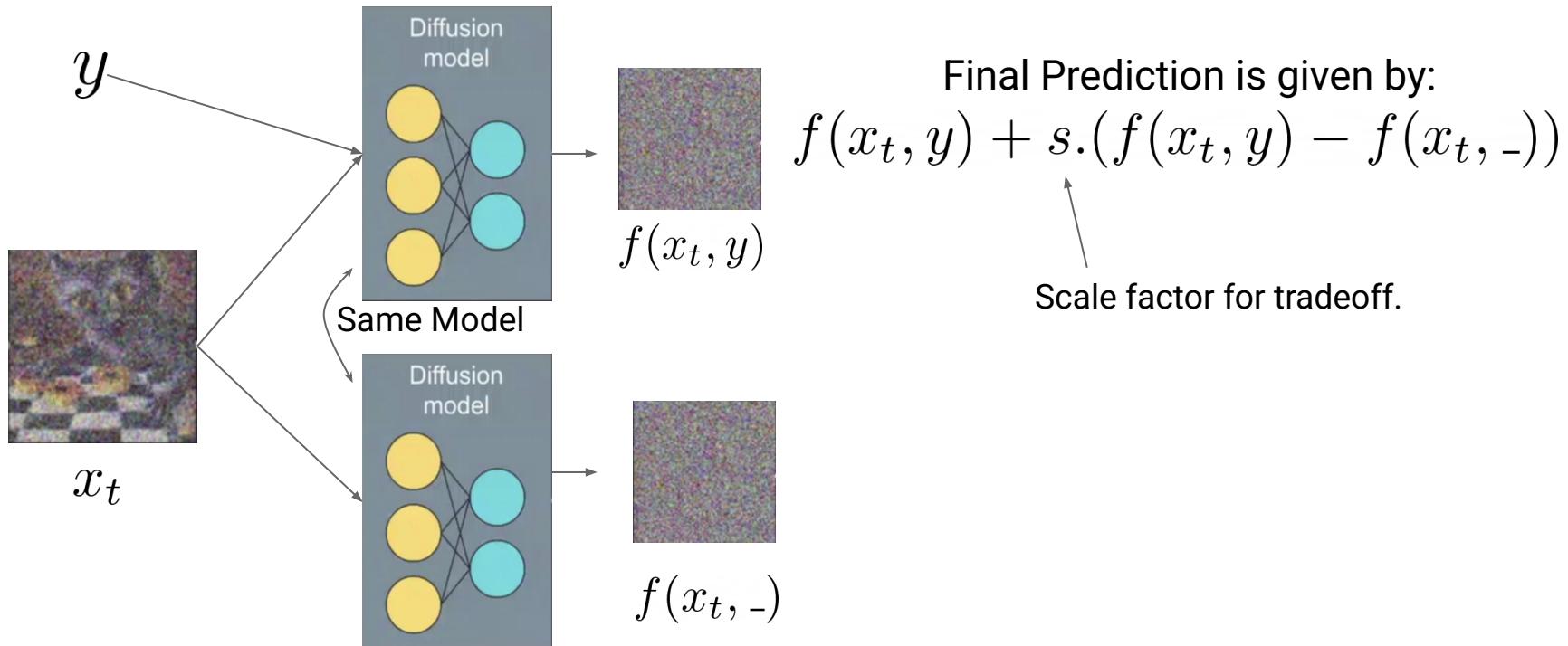


[Diffusion Models Beats GANs on Image Synthesis (Dhariwal & Nichol 2021)]

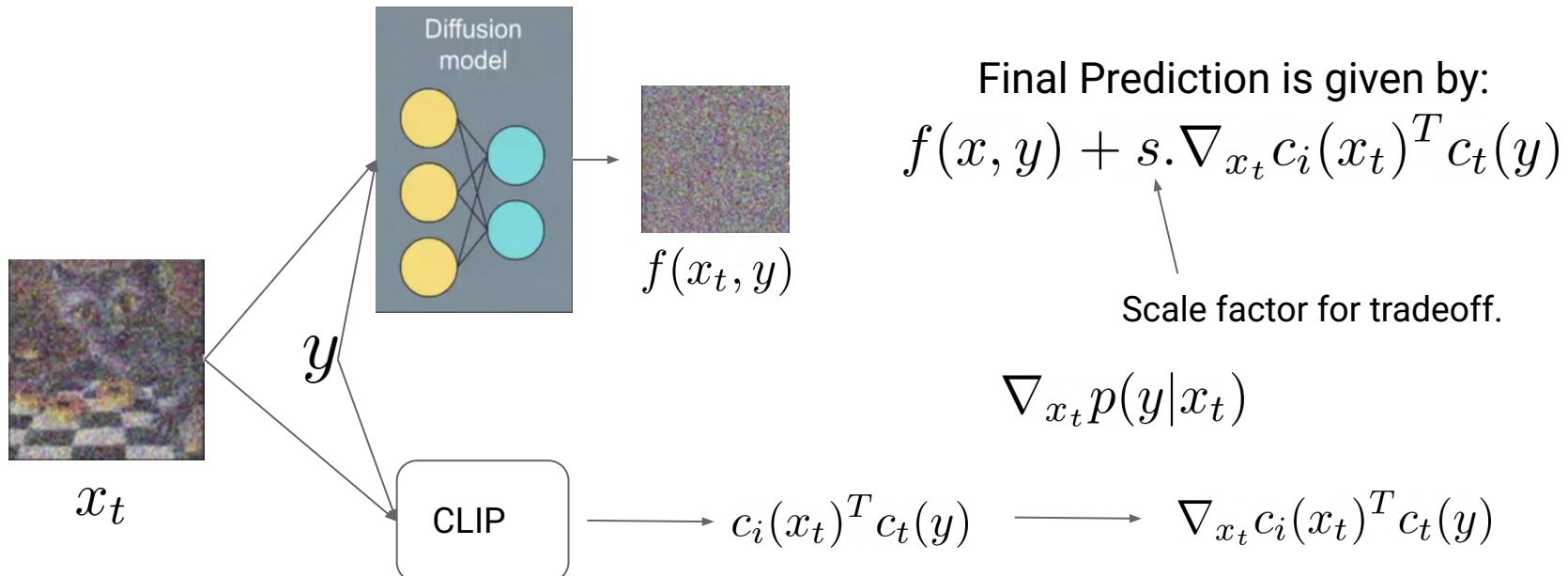
Classifier Guidance



Classifier-Free guidance



CLIP Guidance



CLIP verses classifier-free guidance

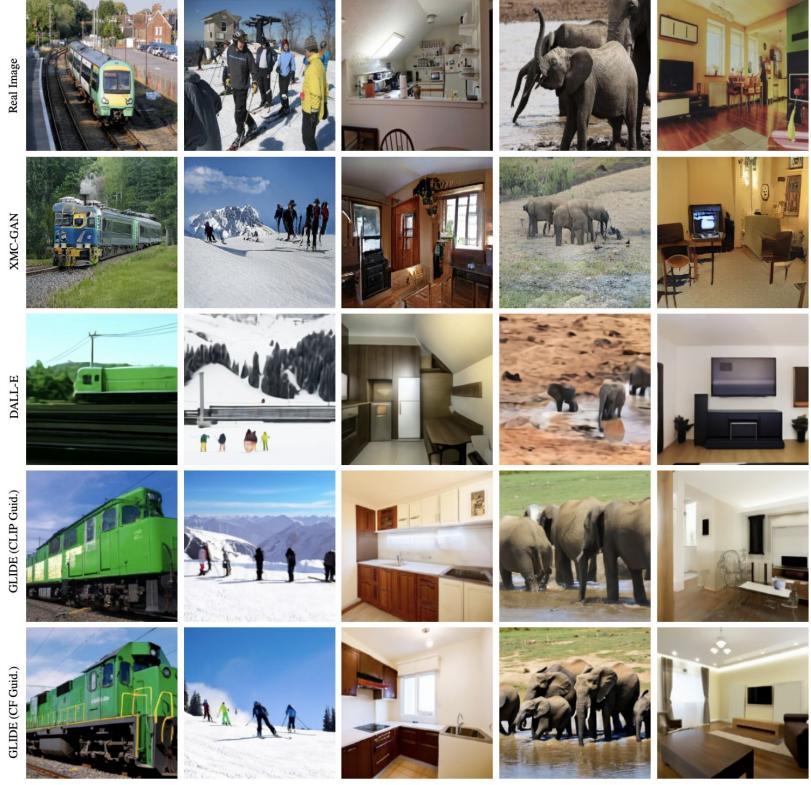


CLIP Guidance



Classifier-Free Guidance

Comparison



"a green train is coming down the tracks"

"a group of skiers are preparing to ski down a mountain."

"a small kitchen with a low ceiling"

"a group of elephants walking in muddy water."

"a living area with a television and a table"

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)

Discussion

- 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
- Going forward, it is extremely important to cut costs of these inference algorithms
 - Hinted the use of parallel decoding and MRF methods for cutting down the costs
 - More detailed algorithms will be presented by Dilip and Sadeep
- Progress in generation hinges on evaluation methods
 - Shobhita will present new evaluation methods