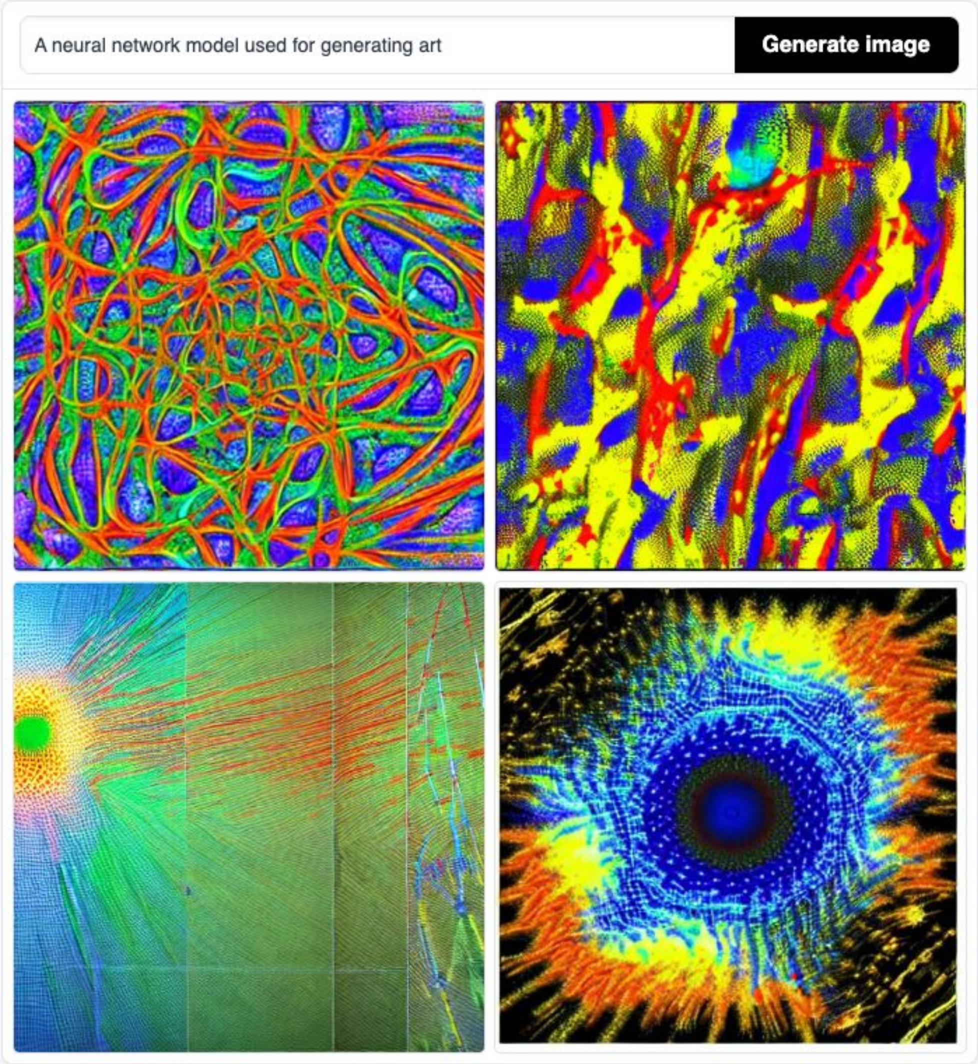
New Generative AI models

(a big thank to Sotiris Anagnostidis for the presentation skeleton)



Overview

- Fundamentals
- GANs
- Autoregressive models
- Diffusion models

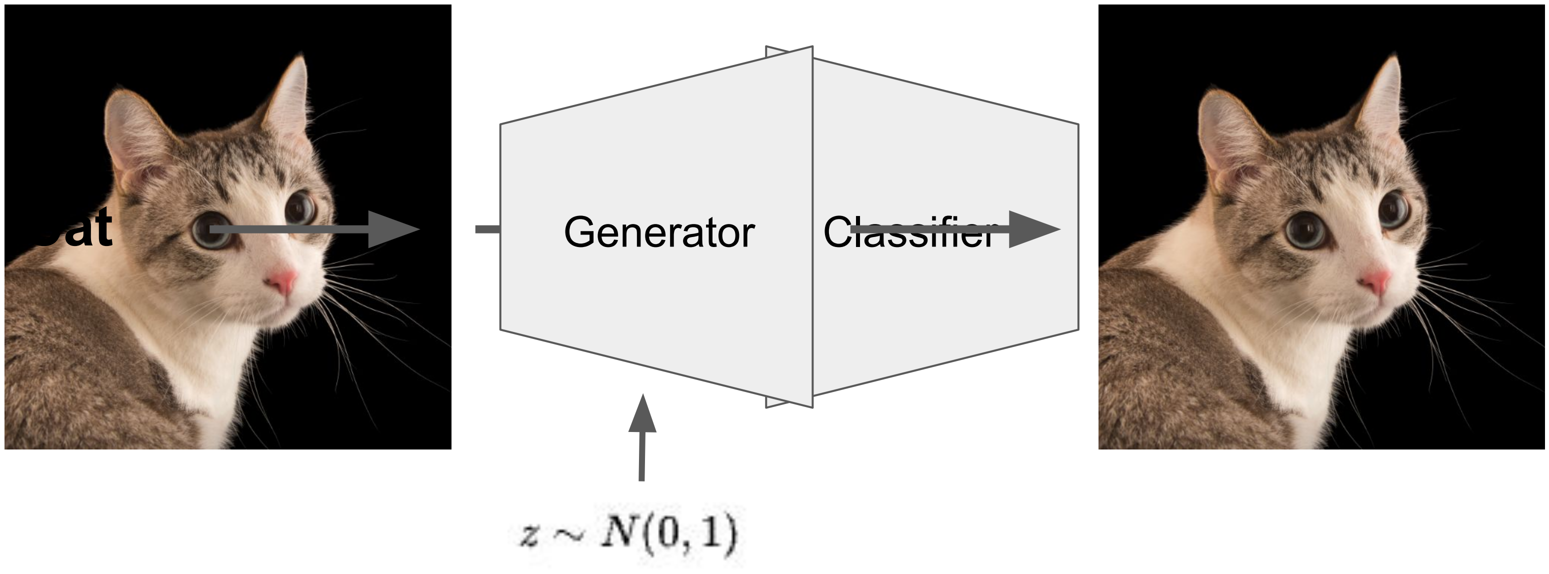
What is a generative model?

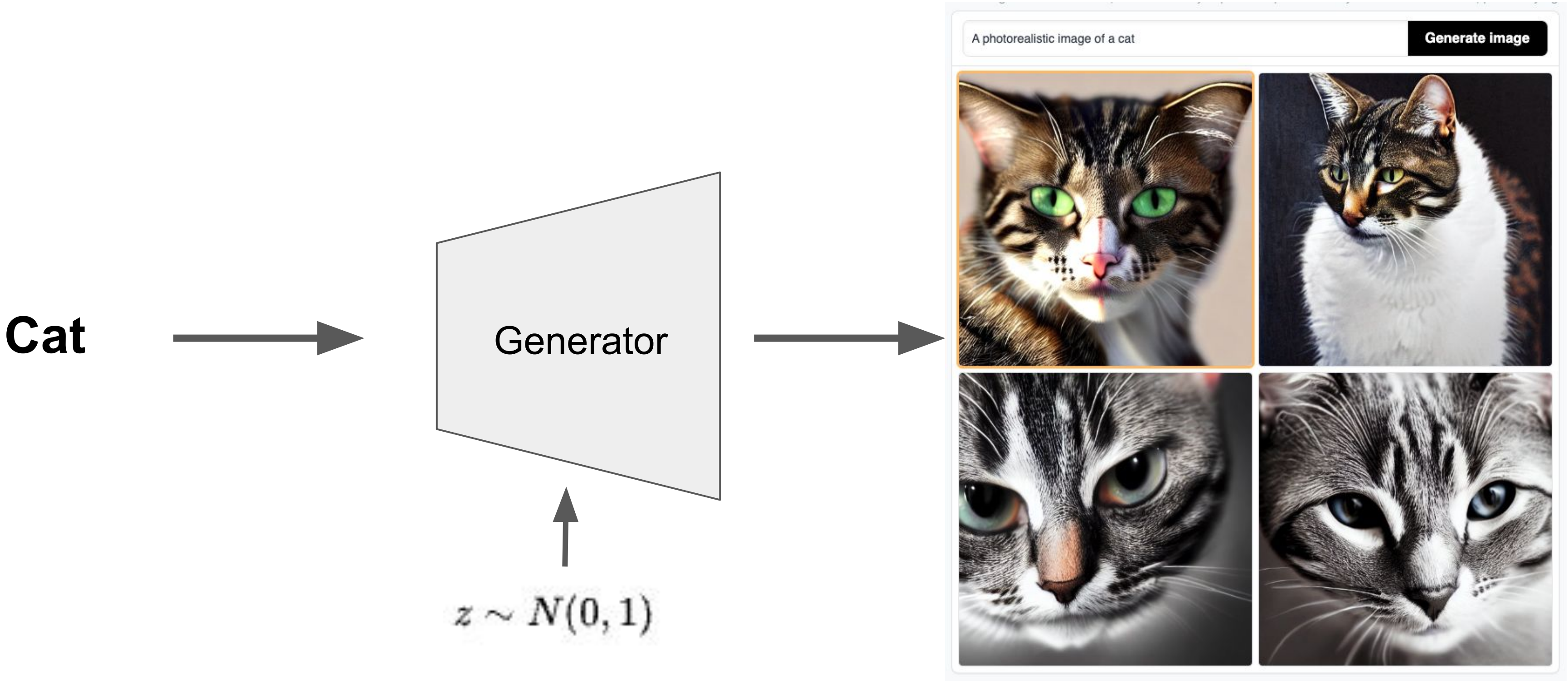
- An algorithm that generates data
- A statistical model of the joint distribution of some data p(x, y, ….)

S

What is a generative model?

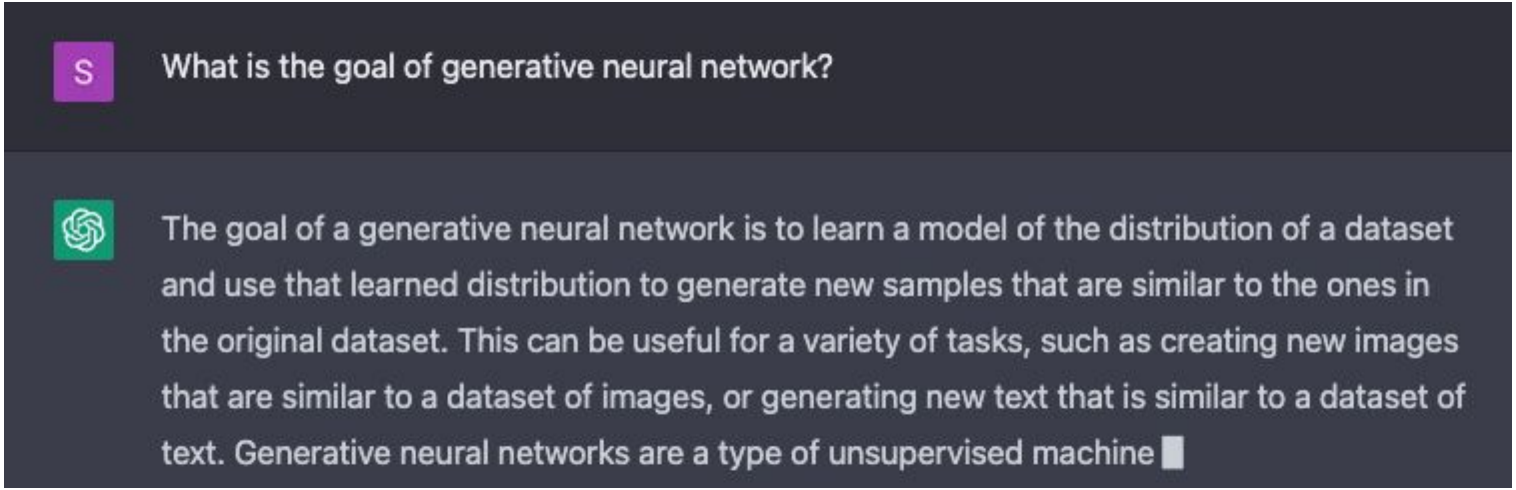
A generative model is a type of machine learning model that is capable of generating new examples that are similar to a training dataset. Generative models are designed to learn the underlying distribution of a dataset and then use this knowledge to generate new examples that belong to the same distribution. These models are typically used in tasks such as image generation, text generation, and audio generation.


Generator
Classifier
$z \sim N(0,1)$

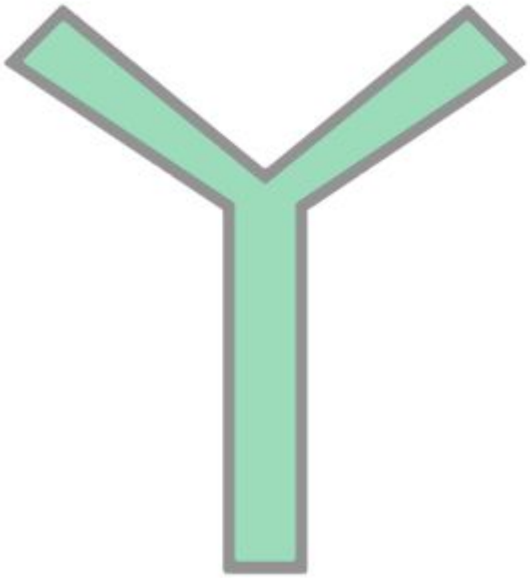

Cat
Generator
$z \sim N(0,1)$
A photorealistic image of a cat
Generate image

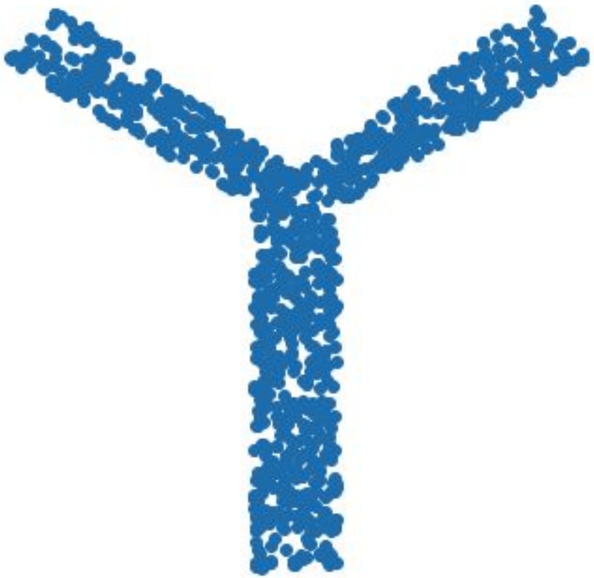
What is the goal of generative modeling?

- Make synthetic data that “looks like” real data
- How to measure “looks like”?
- Has high probability under a density model fit to real data (true density of the data).
- Goal is not to replicate training data (failure mode) but to make new data.

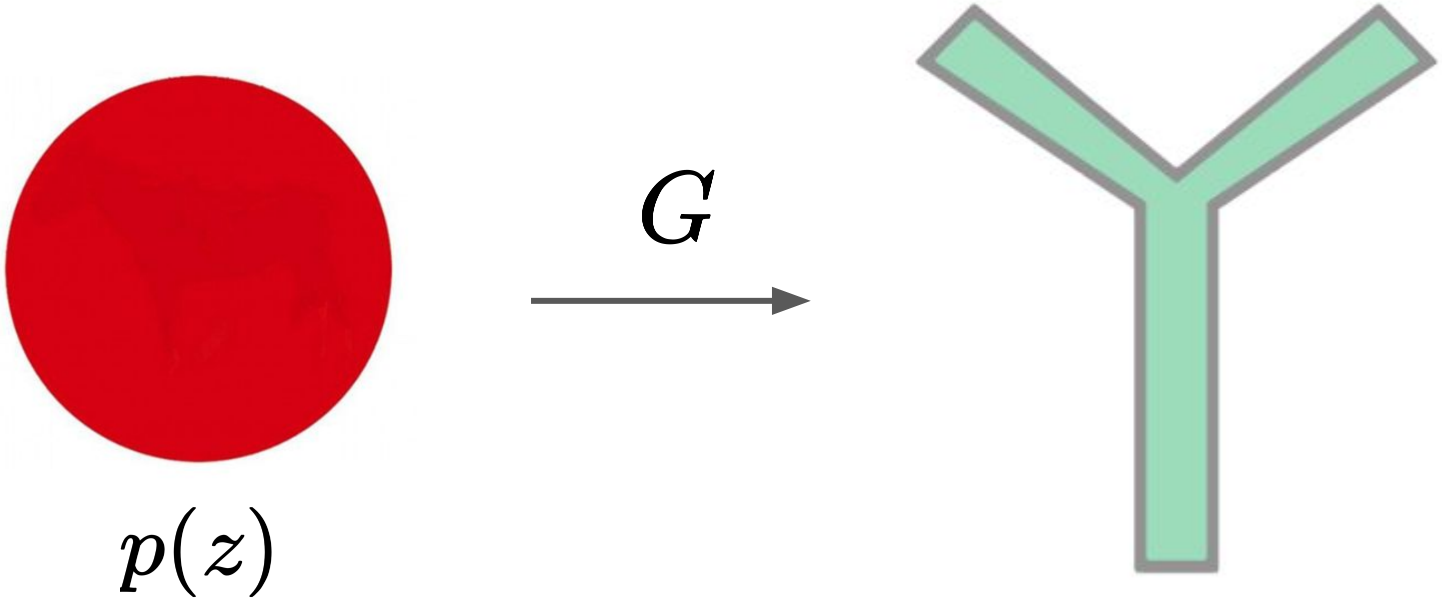


Example

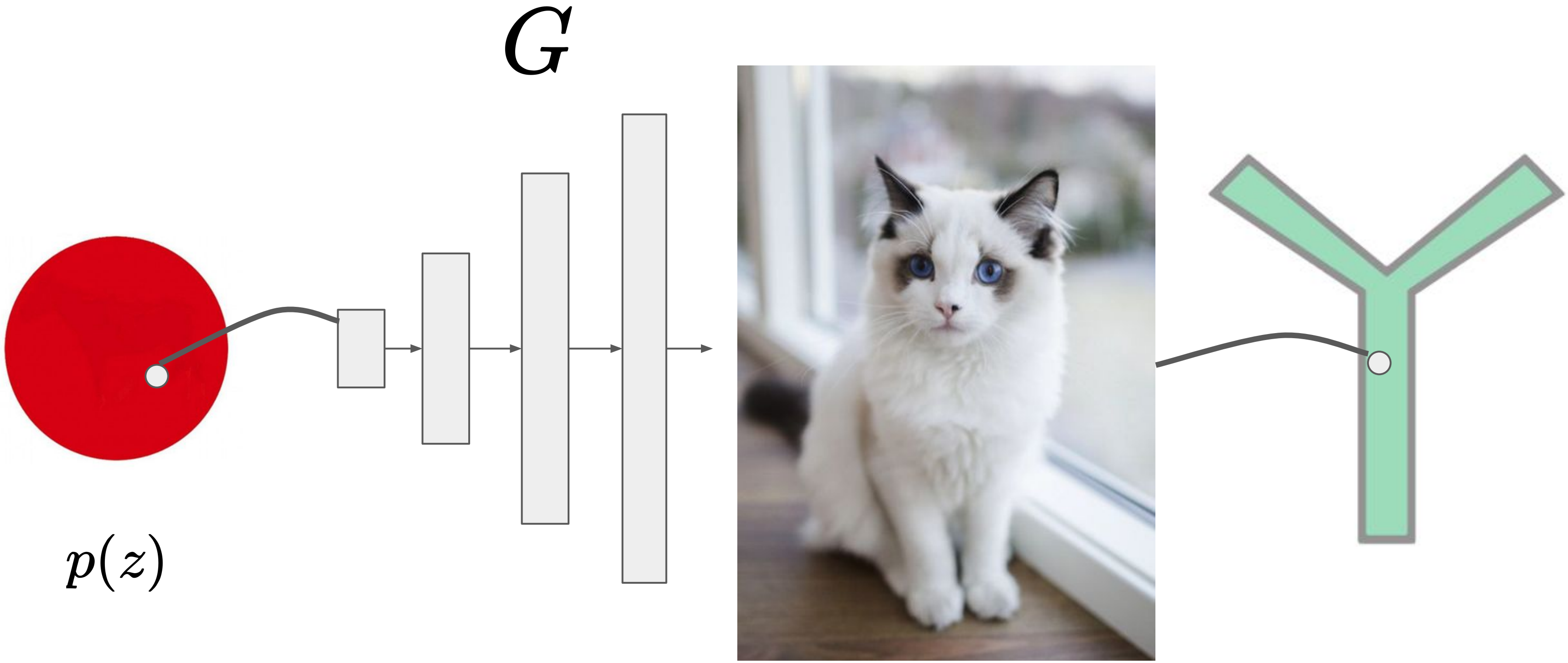




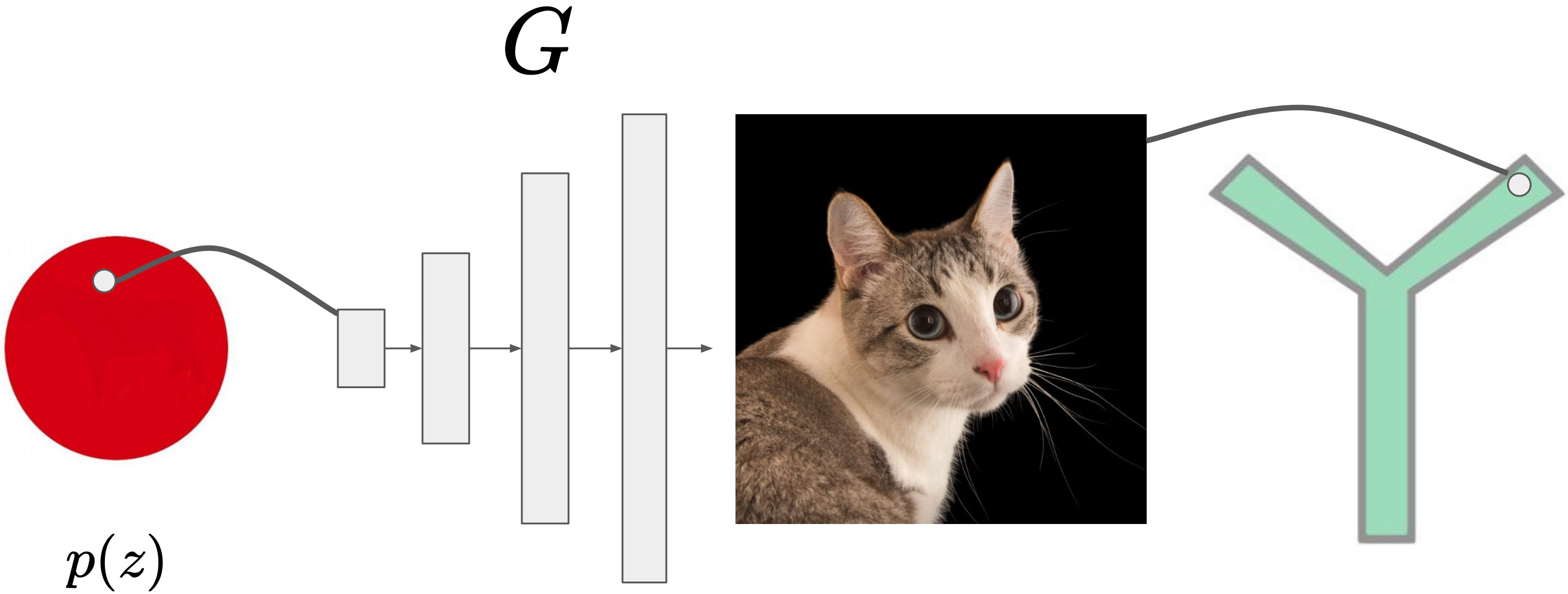
Deep generative models are distribution transformers



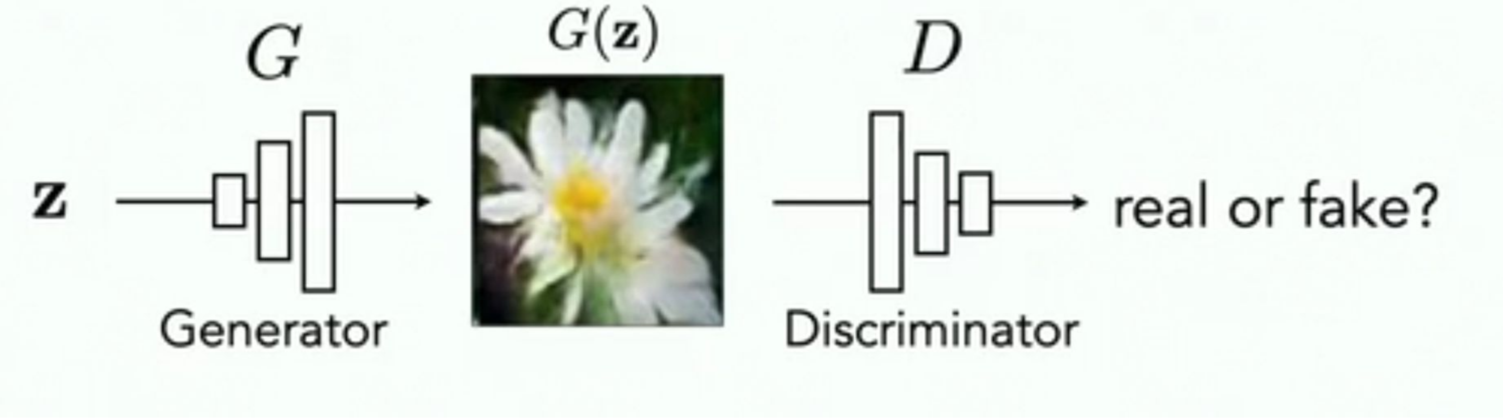
Deep generative models are distribution transformers



Deep generative models are distribution transformers

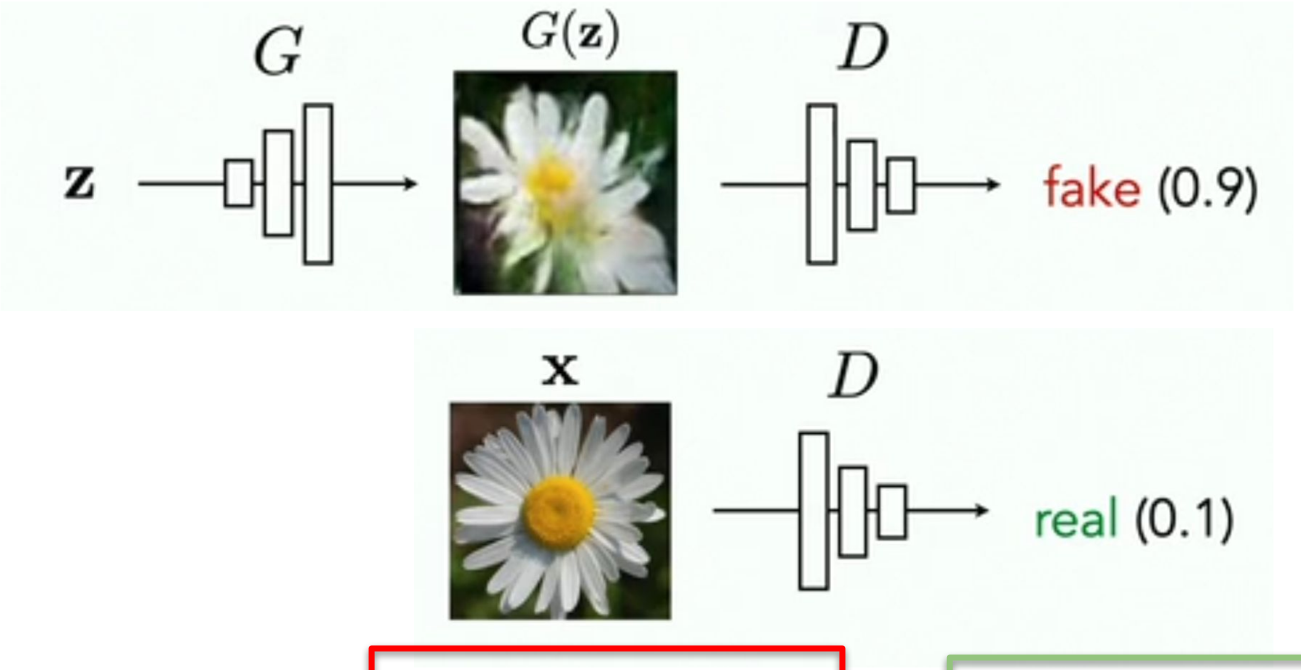


Generative Adversarial Networks (GAN)



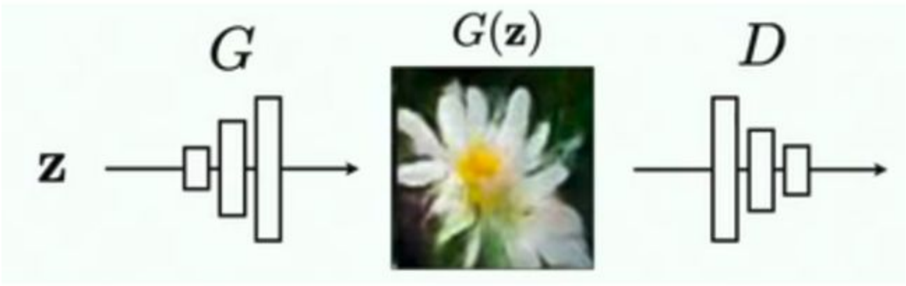
- G tries to synthesize fake images that fool D
- D tries to identify the fakes

Generative Adversarial Networks (GAN)



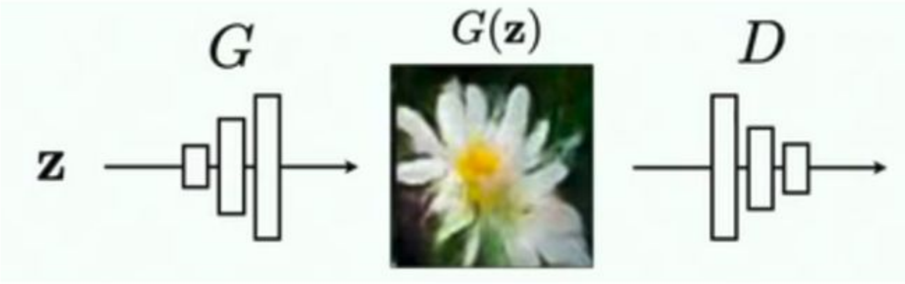
$$\text{argmax}_D \mathbb{E}_{z,x}\left[\boxed{\log D(G(z))} + \boxed{\log(1 - \text{D}(\text{x}))}\right]$$

Generative Adversarial Networks (GAN)



$$\mathrm{arg}\boxed{\mathrm{min}}_G\, \mathbb{E}_{z,x}\left[\log D(G(z)) + \log(1 - D(x))\right]$$

Generative Adversarial Networks (GAN)



$$\text{argmin}_G \, \text{max}_D \, \mathbb{E}_{z,x} \left[\log D(G(z)) + \log(1 - D(x)) \right]$$

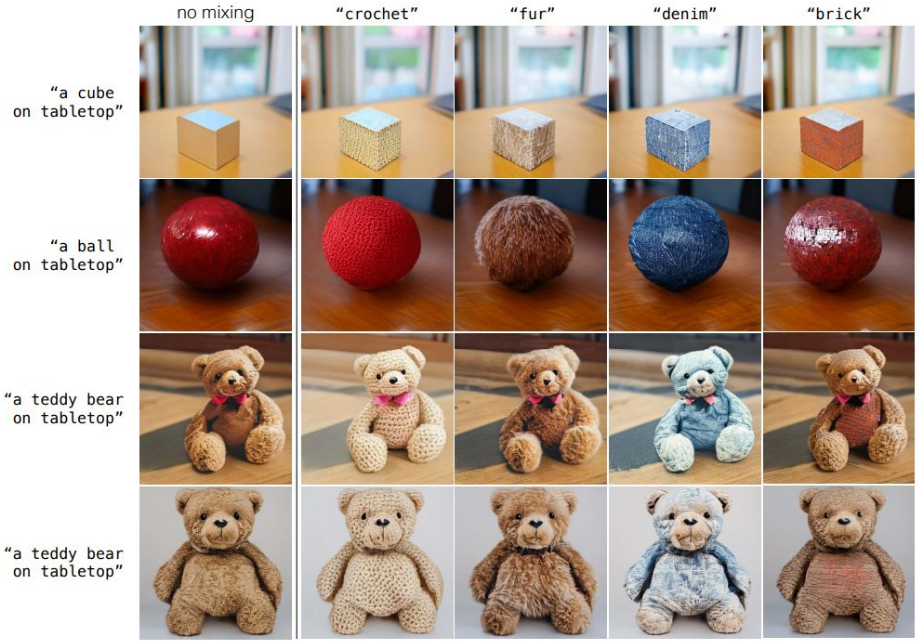
GANs applications

GANs..again!

Scaling up GANs for Text-to-Image Synthesis

Minguk Kang[1,3] Jun-Yan Zhu[2] Richard Zhang[3]
Jaesik Park[1] Eli Shechtman[3] Sylvain Paris[3] Taesung Park[3]

[1]POSTECH [2]Carnegie Mellon University [3]Adobe Research



Autoregressive Models

The cat sits on the ______ $\quad$ Predictor $\longrightarrow$ mat

$$p(X) = p(x_n|x_1,\ldots,x_{n-1})p(x_{n-1}|x_1,\ldots,x_{n-2})\ldots p(x_w|x_1)p(x_1)$$

$$p(\text{The cat sits on the mat})$$

$$p(\text{The})$$

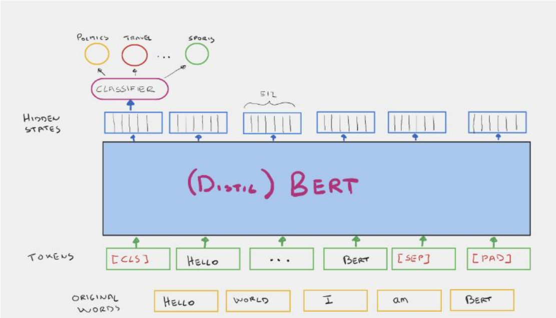
$$p(\text{cat}|\text{The})$$

$$p(\text{sit}|\text{The cat})$$

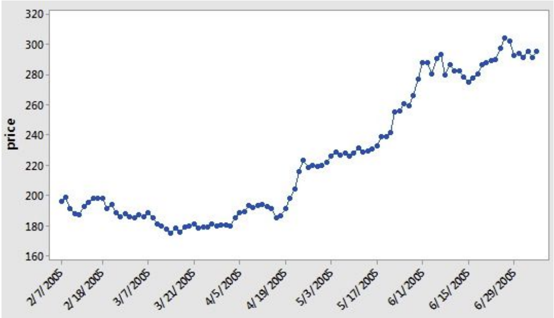
$$p(\text{on}|\text{The cat sits})$$

$$p(\text{the}|\text{The cat sits on })$$

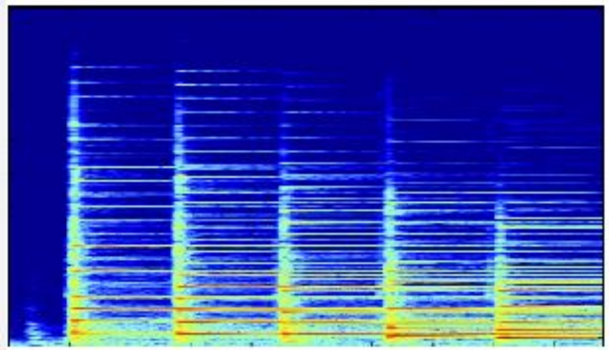
$$p(\text{mat}|\text{The cat sits on the})$$

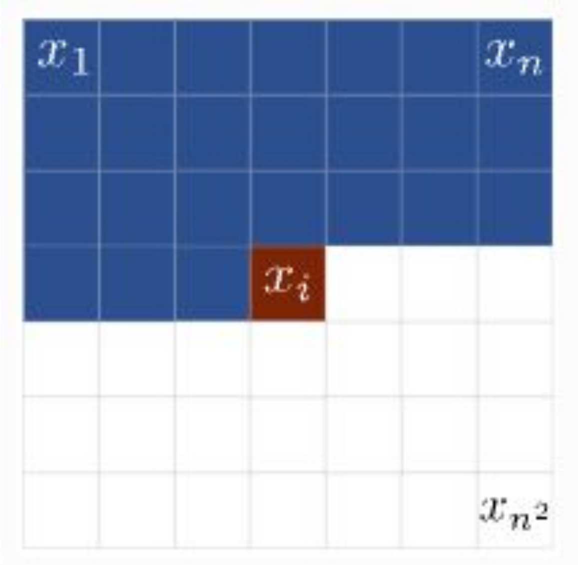


POLITICS
TRAVEL
SPORTS
CLASSIFIER
512
HIDDEN STATES
(DISTIL) BERT
TOKENS
[CLS]
HELLO
...
BERT
[SEP]
[PAD]
ORIGINAL WORDS
HELLO
WORLD
I
am
BERT

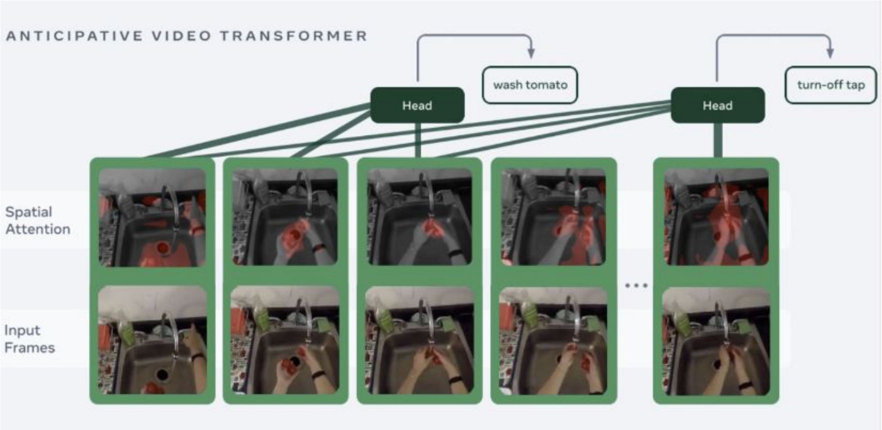


320
300
280
260
240
220
200
180
160
price
2/7/2005
2/18/2005
3/7/2005
3/21/2005
4/5/2005
4/19/2005
5/3/2005
5/17/2005
6/1/2005
6/15/2005
6/29/2005

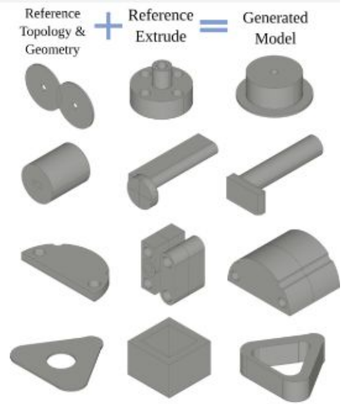




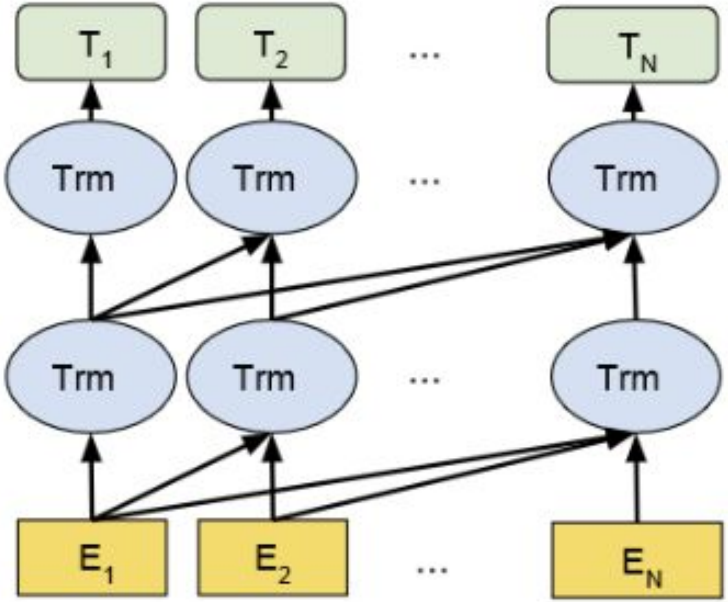
x_1
x_n
x_i
x_{n^2}



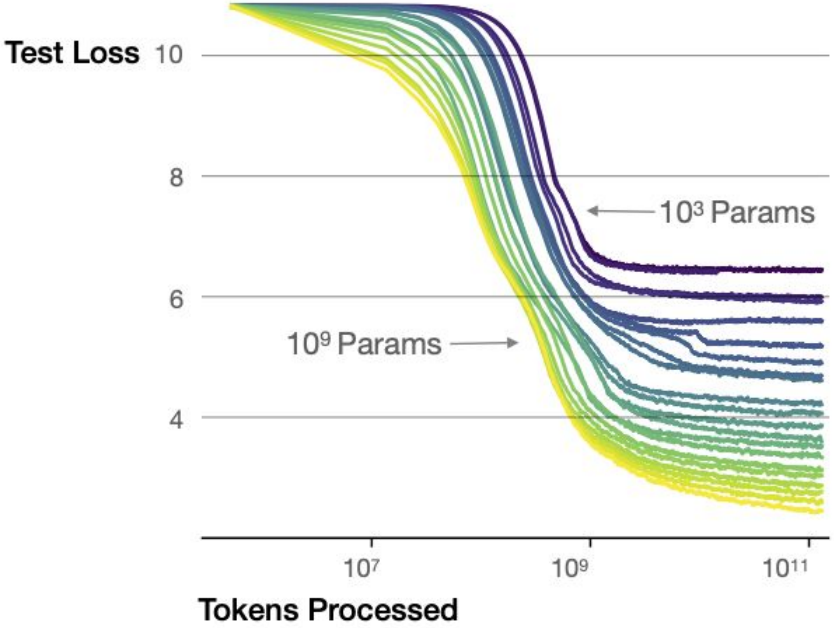
ANTICIPATIVE VIDEO TRANSFORMER
wash tomato
turn-off tap
Head
Head
Spatial Attention
Input Frames
...



Reference Topology & Geometry
+
Reference Extrude
=
Generated Model



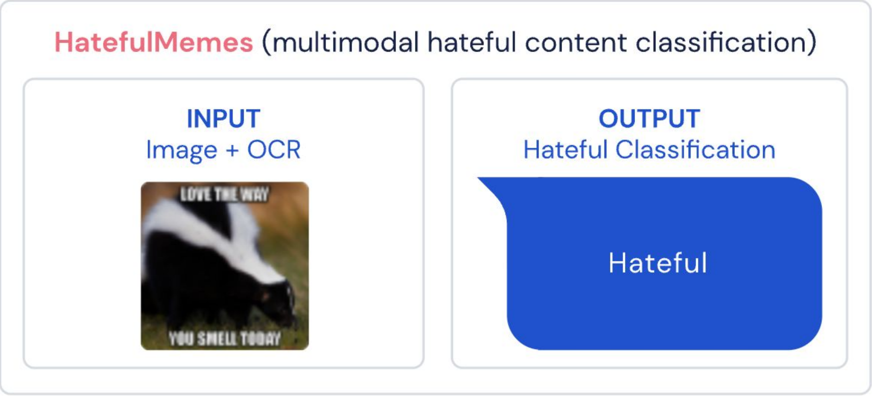
T1
T2
...
TN
Trm
Trm
...
Trm
Trm
Trm
...
Trm
E1
E2
...
EN

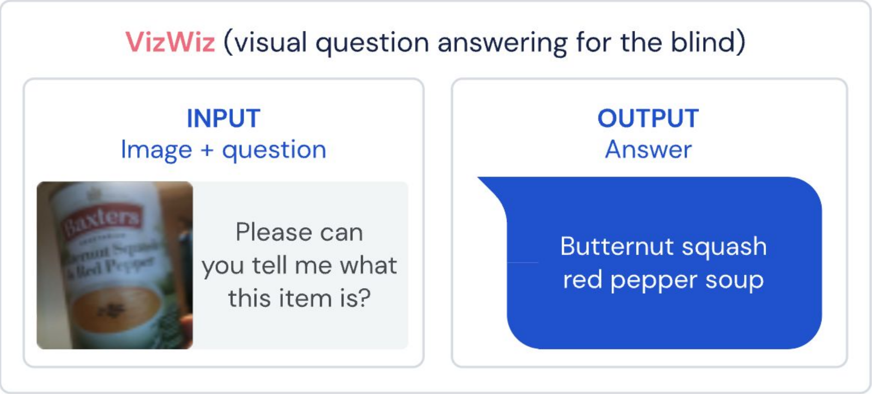


Test Loss
10
8
6
4
10³ Params
10⁹ Params
10⁷
10⁹
10¹¹
Tokens Processed

```
function findHighestNumber(array) {
    var highestNumber = 0;
    for (var i = 0; i < array.length; i++) {
        if (array[i] > highestNumber) {
        highestNumber = array[i];
        }
    }
    return highestNumber;
}
```

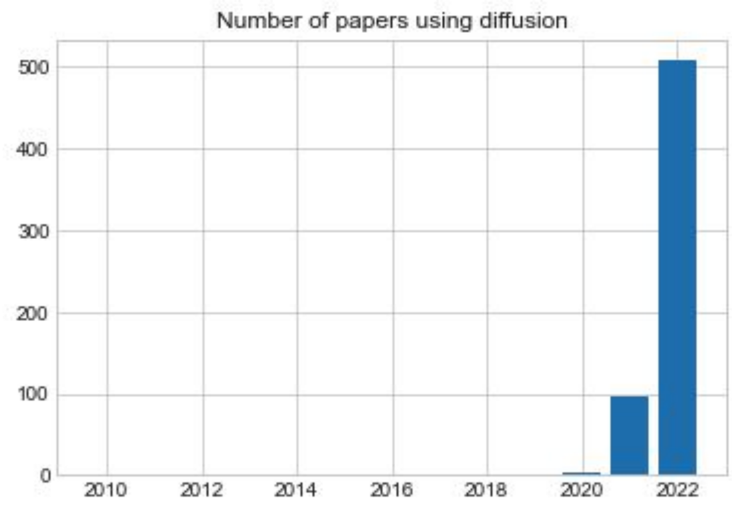
```
theorem amc12b_2020_p6
  (n : ℕ)
  (h₀ : 9 ≤ n) :
  ∃ x : ℕ, (x:ℝ)^2 = (nat.factorial (n + 2) -
    nat.factorial (n + 1)) / nat.factorial n :=
begin
  revert_all,
  intros n hn,
  use n + 1,
  field_simp [nat.factorial_ne_zero, pow_succ'],
  ring_exp
end
```





Diffusion Models

[Awesome Diffusion Models]



How do diffusion models work?

S How do denoising diffusion probabilistic models work?

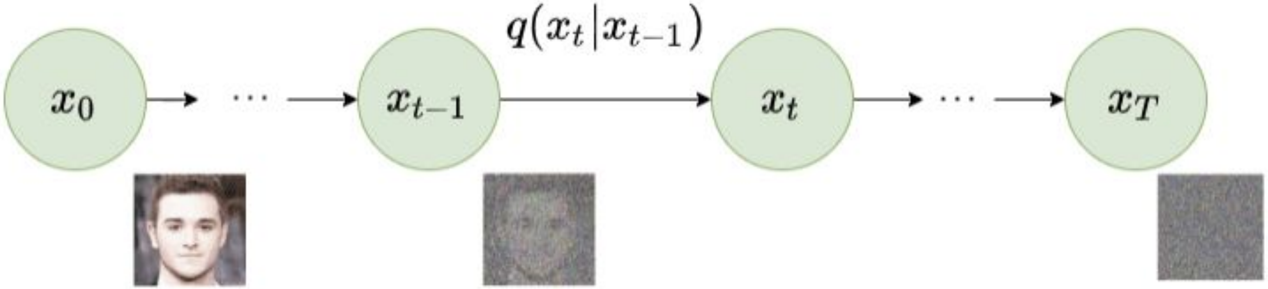
Denoising diffusion probabilistic models are a type of generative model that can be used to generate new data samples that are similar to a training dataset. These models work by using a process called diffusion, which is a way of iteratively refining a set of random noise samples until they resemble the training data.

First, the model is trained on a dataset, and a set of random noise samples are generated. These noise samples are then fed into the model, which uses a series of deterministic and stochastic operations to transform them into samples that are similar to the training data. This process is repeated multiple times, and at each iteration the samples become more and more similar to the training data.

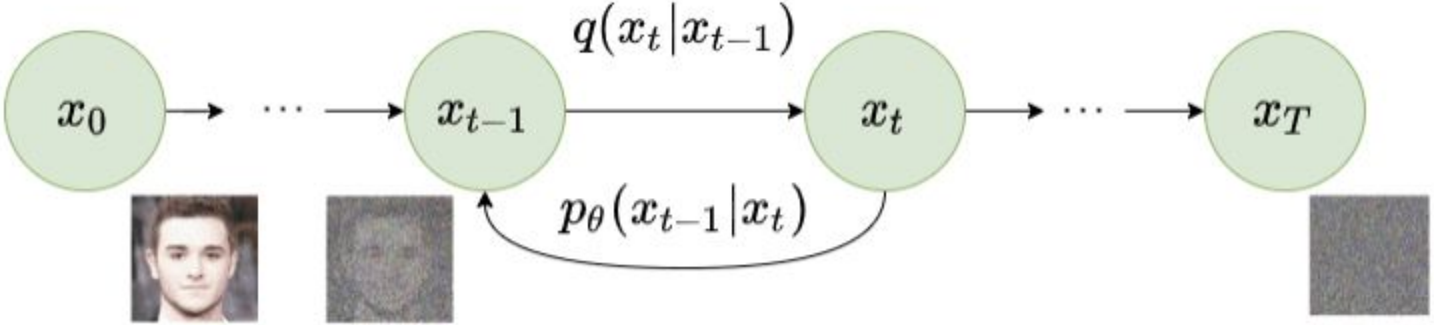
One key aspect of denoising diffusion probabilistic models is that they are designed to be resistant to noise and other forms of corruption in the input data. This allows them to generate high-quality samples even when the input data is noisy or incomplete.

The (vague) details

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



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



[Ho et. al 2020]

Training DDPM

$$\tilde{a}_t = \prod_{s=0}^{t} a_s = \prod_{s=0}^{t} (1-\beta_s)$$

Forward diffusion: $x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon_{t-1} = \sqrt{\tilde{a}_t}x_0 + \sqrt{1-\tilde{a}_t}\epsilon_0$

Reverse diffusion:

1. Sample from an isotropic Gaussian x_T

2. $q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t), \tilde{\beta}_t I)$

$$\tilde{\beta}_t = \frac{1-\tilde{a}_{t-1}}{1-\tilde{a}_t}\beta_t$$

$$\tilde{\mu}_t(x_t) = \frac{1}{\sqrt{a_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\tilde{a}_t}}\boxed{\epsilon}\right) \longrightarrow \epsilon_\theta(x_t, t)$$

3. Train minimizing the Evidence lower bound objective $-\mathbb{E}_{q(x_0)}\log p_\theta(x_0) = \ldots$

4. Or simpler…

$$L = \mathbb{E}_{t\sim[1,T],x_0,\epsilon_t}[D(\epsilon_t - \epsilon_\theta(x_t, t))] = \mathbb{E}_{t\sim[1,T],x_0,\epsilon_t}[D(\epsilon_t - \epsilon_\theta(\sqrt{\tilde{a}_t}x_0 + \sqrt{1-\tilde{a}_t}\epsilon_t, t)]$$

$$D \in L_1, L_2, \ldots$$

Training DDPM

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \mathrm{Uniform}(\{1, \dots, T\})$
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** $\mathbf{x}_0$

Thanks for your attention!

Some References

[CVPR Tutorial on DIffusion Models]

[NERF]

[Lost of blog posts]

[Scaling laws of neural networks]