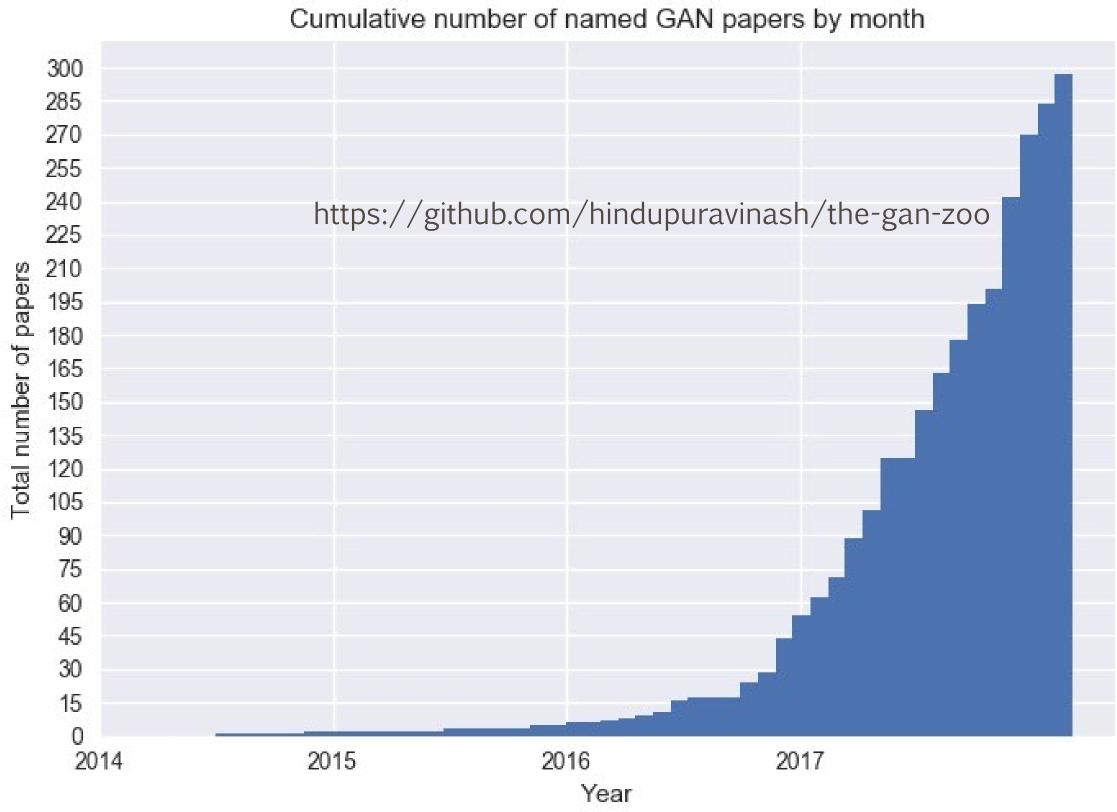
GENERATIVE MODELS

Chih-Chung Hsu (許志仲)
Institute of Data Science
National Cheng Kung University
https://cchsu.info

All Kinds of GAN …

- GAN
- ACGAN
- BGAN
- CGAN
- DCGAN
- EBGAN
- fGAN
- GoGAN
- ⋮



https://github.com/hindupuravinash/the-gan-zoo

Mihaela Rosca, Balaji Lakshminarayanan, David Warde-Farley, Shakir Mohamed, "Variational Approaches for Auto-Encoding Generative Adversarial Networks", arXiv, 2017

---

[2] We use the Greek α prefix for α-GAN, as AEGAN and most other Latin prefixes seem to have been taken https://deephunt.in/the-gan-zoo-79597dc8c347.

partial credit by CS311n

Supervised vs. Unsupervised

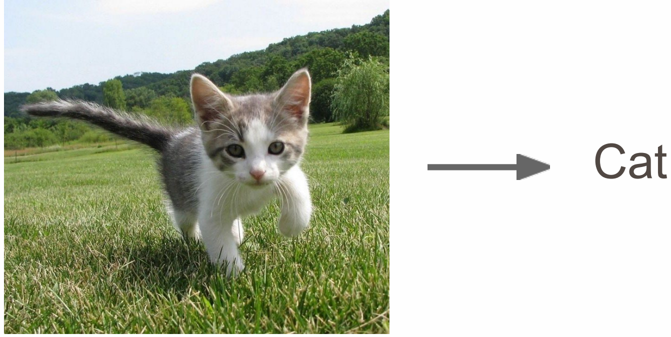
- Supervised learning:
  - We have correct answer to learn!!
    - Exactly learning the proper parameters!!
  - There are several state-of-the-art methods!!
- Unsupervised learning:
  - We don’t have the correct answer
    - Only “guess”
  - Need to have a lot of strategies to fine-tune the estimated answer
  - Not good enough algorithm so far

partial credit by CS311n

Supervised vs Unsupervised Learning

- Supervised Learning
- Data: (x, y)
- x is data, y is label

- Goal: Learn a function to map x -> y
- Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.



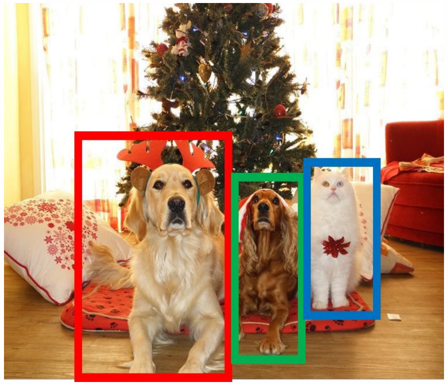
Classification

partial credit by CS311n

Supervised vs Unsupervised Learning

- Supervised Learning
- Data: (x, y)
- x is data, y is label

- Goal: Learn a function to map x -> y
- Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.



DOG, DOG, CAT

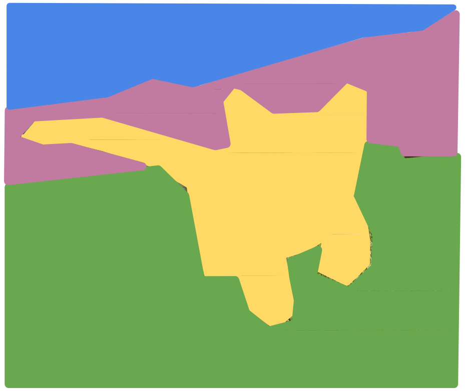
Object Detection

partial credit by CS311n

Supervised vs Unsupervised Learning

- Supervised Learning
- Data: (x, y)
- x is data, y is label

- Goal: Learn a function to map x -> y
- Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.



GRASS, **CAT**, **TREE**, **SKY**

Semantic Segmentation

partial credit by CS311n

Supervised vs Unsupervised Learning

- Supervised Learning
- Data: (x, y)
- x is data, y is label

- Goal: Learn a function to map x -> y
- Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

A cat sitting on a suitcase on the floor

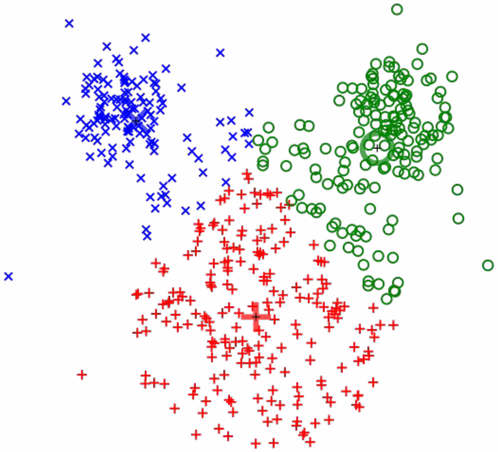
Image captioning

partial credit by CS311n

Supervised vs Unsupervised Learning

- Supervised Learning
- Data: (x, y)
- x is data, y is label

- Goal: Learn a function to map x -> y
- Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.



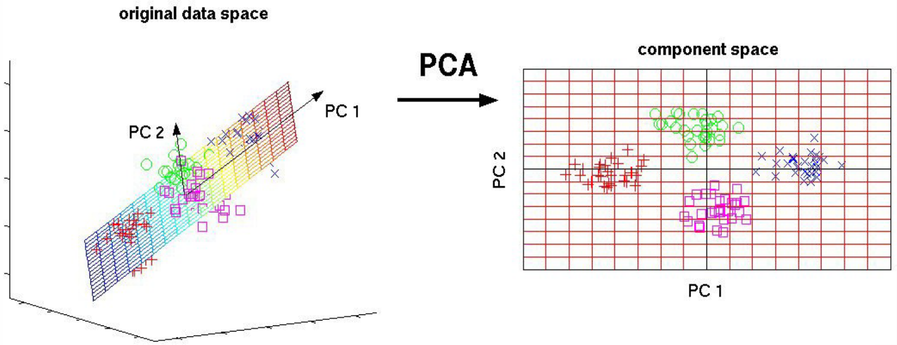
K-means clustering

partial credit by CS311n

Supervised vs Unsupervised Learning

- Supervised Learning
- Data: (x, y)
- x is data, y is label

- Goal: Learn a function to map x -> y
- Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.



3-d

2-d

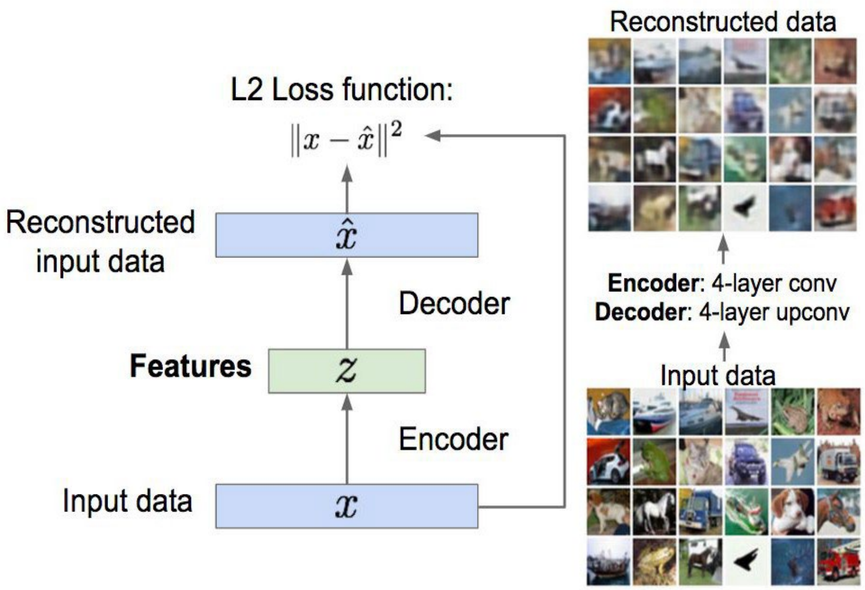
Principal Component Analysis (Dimensionality reduction)

partial credit by CS311n

Supervised vs Unsupervised Learning

- Supervised Learning
- Data: (x, y)
- x is data, y is label

- Goal: Learn a function to map x -> y
- Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.



Autoencoders (Feature learning)

partial credit by CS311n

Supervised vs Unsupervised Learning

- Supervised Learning
- Data: (x, y)
- x is data, y is label

- Goal: Learn a function to map x -> y
- Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

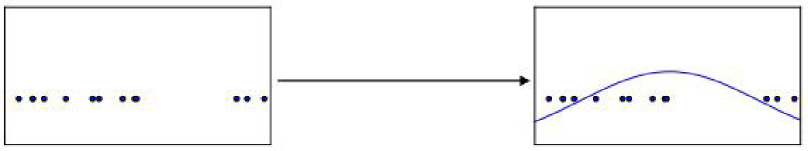
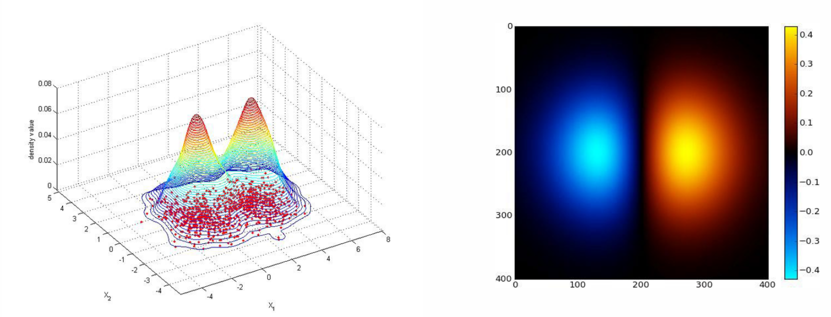


Figure copyright Ian Goodfellow, 2016. Reproduced with permission.

1-d density estimation



2-d density estimation

partial credit by CS311n

Supervised vs Unsupervised Learning

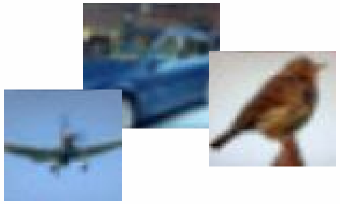
- Supervised Learning
- Data: (x, y)
- x is data, y is label
- Goal: Learn a function to map x -> y
- Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

- Unsupervised Learning
- Data: x ← Training data is cheap
- Just data, no labels!
- Goal: Learn some underlying hidden structure of the data
- Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

Solve unsupervised learning
=> understand structure of visual world

partial credit by CS311n

Generative Models



Training data ~ $p_{data}(x)$

Generated samples ~ $p_{model}(x)$

Want to learn $p_{model}(x)$ similar to $p_{data}(x)$

Given training data, generate new samples from same distribution

Addresses density estimation, a core problem in unsupervised learning

Several flavors:

- Explicit density estimation: explicitly define and solve for $p_{model}(x)$
- Implicit density estimation: learn model that can sample from $p_{model}(x)$ w/o explicitly defining it

partial credit by CS311n

Why Generative Models?

- Realistic samples for artwork, super-resolution, colorization, etc.

- Generative models of time-series data can be used for simulation and planning (reinforcement learning applications!)
- Training generative models can also enable inference of latent representations that can be useful as general features

Figures from L-R are copyright: (1) Alec Radford et al. 2016; (2) Phillip Isola et al. 2017. Reproduced with authors permission (3) BAIR Blog.

Unsupervised Learning

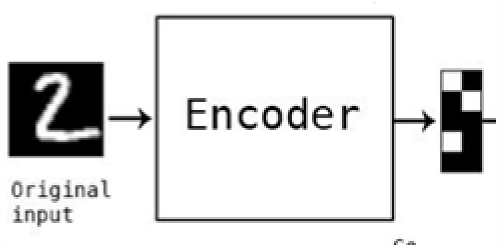
- More challenging than supervised learning :
  - No label or curriculum → self learning

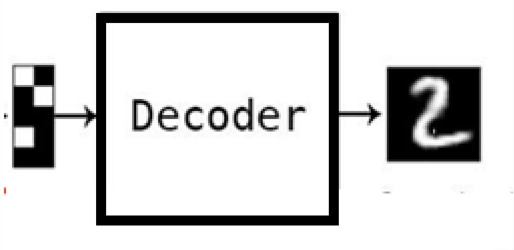
- Traditional solutions:
  - Clustering
  - Linear / nonlinear dimensionality reduction
    - PCA vs. Manifold learning
- Some NN solutions :
  - Boltzmann machine
  - Auto-encoder or Variational Inference
  - Generative Adversarial Network

partial credit by CS311n

Unsupervised learning vs. Generative model

- Unsupervised learning
  - z=f(x)
- Generative model
  - x=g(z)
- It is …
  - P(z|x) vs. P(x|z)
  - An encoder vs. a decoder
    - Encoder: Feature extraction / Dimensionality reduction
    - Decoder: Generator / Upsampling
    - P(z|x) = P(x, z) / P(x) ➔ P(x) Intractable (ELBO)
    - P(x|z) = P(x, z) / P(z) ➔ P(z) is prior





P(x,z) is necessary!!

partial credit by CS311n

Some background first: Autoencoders

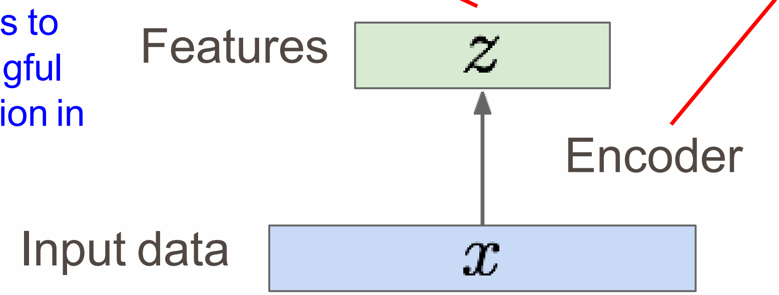
Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

z usually smaller than **x** (dimensionality reduction)

Q: Why dimensionality reduction?

A: Want features to capture meaningful factors of variation in data

Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN



Features z

Encoder

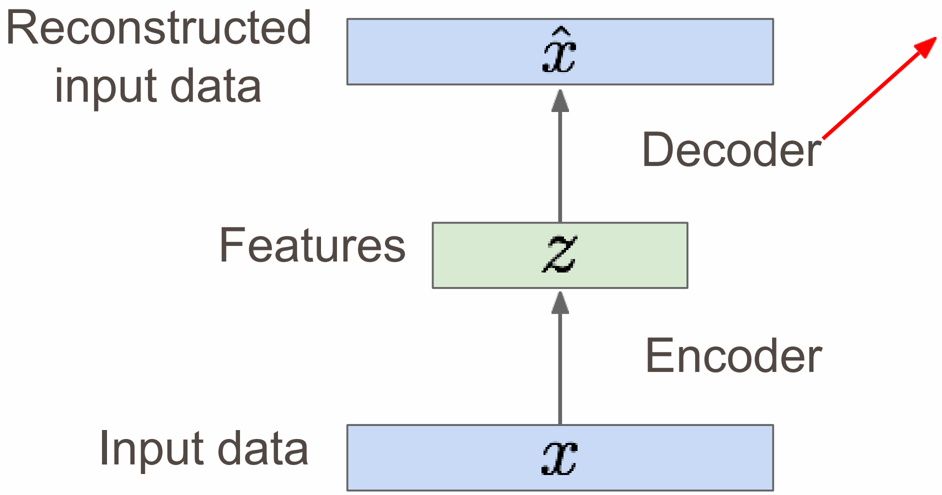
Input data x

partial credit by CS311n

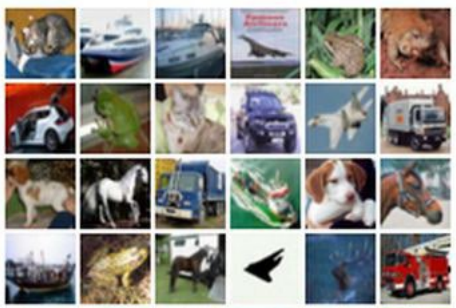
Some background first: Autoencoders

How to learn this feature representation?

Train such that features can be used to reconstruct original data "Autoencoding" - encoding itself



Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN (upconv)

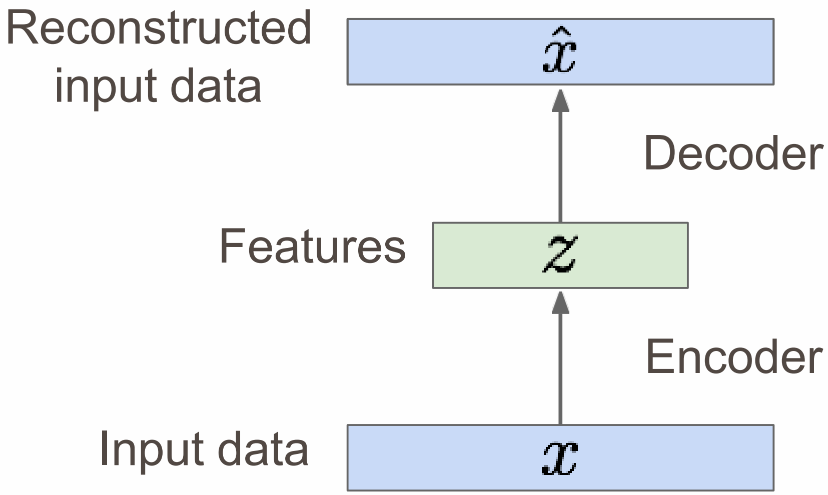


partial credit by CS311n

Some background first: Autoencoders

How to learn this feature representation?

Train such that features can be used to reconstruct original data "Autoencoding" - encoding itself



Reconstructed input data

$\hat{x}$

Decoder

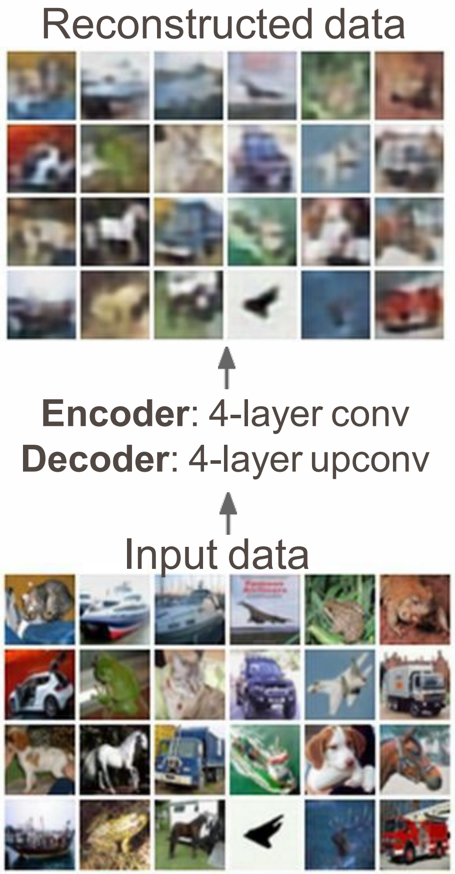
Features

z

Encoder

Input data

x



Reconstructed data

Encoder: 4-layer conv

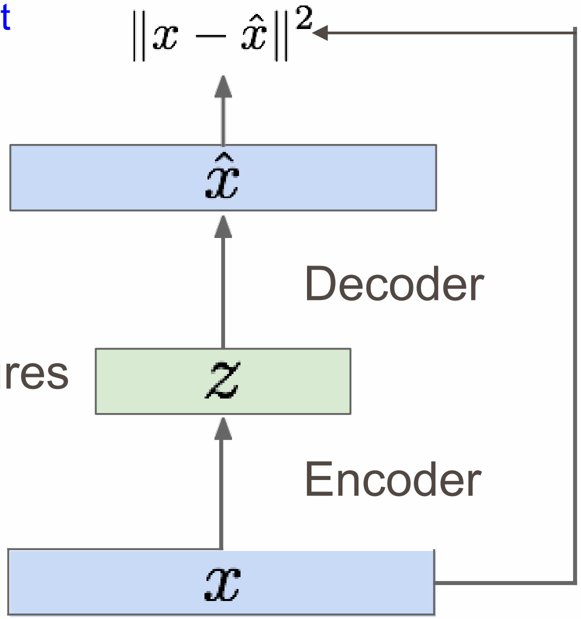
Decoder: 4-layer upconv

Input data

partial credit by CS311n

Some background first: Autoencoders

Train such that features can be used to reconstruct original data



L2 Loss function:

$$\|x - \hat{x}\|^2$$

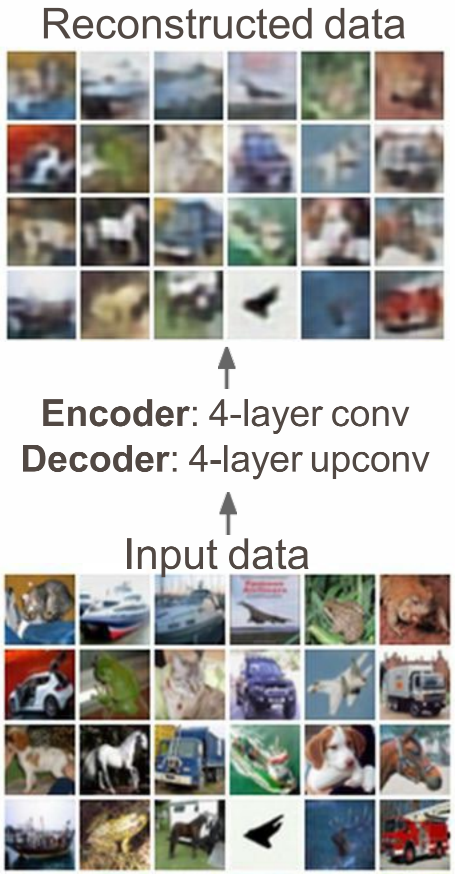
Reconstructed input data $\hat{x}$

Decoder

Features z

Encoder

Input data x



Reconstructed data

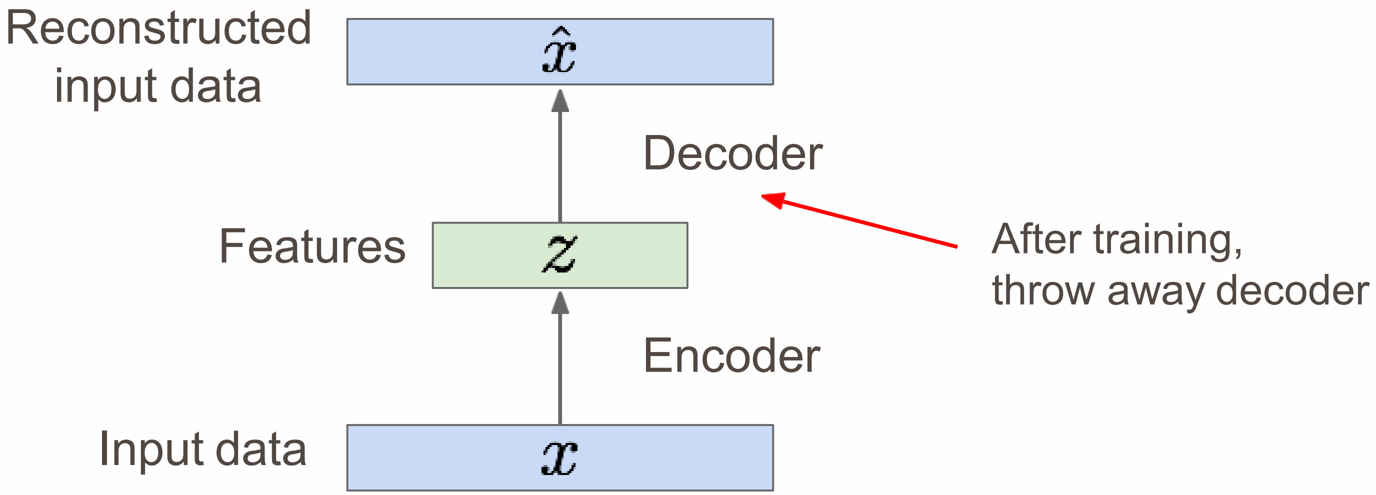
Encoder: 4-layer conv

Decoder: 4-layer upconv

Input data

partial credit by CS311n

Some background first: Autoencoders

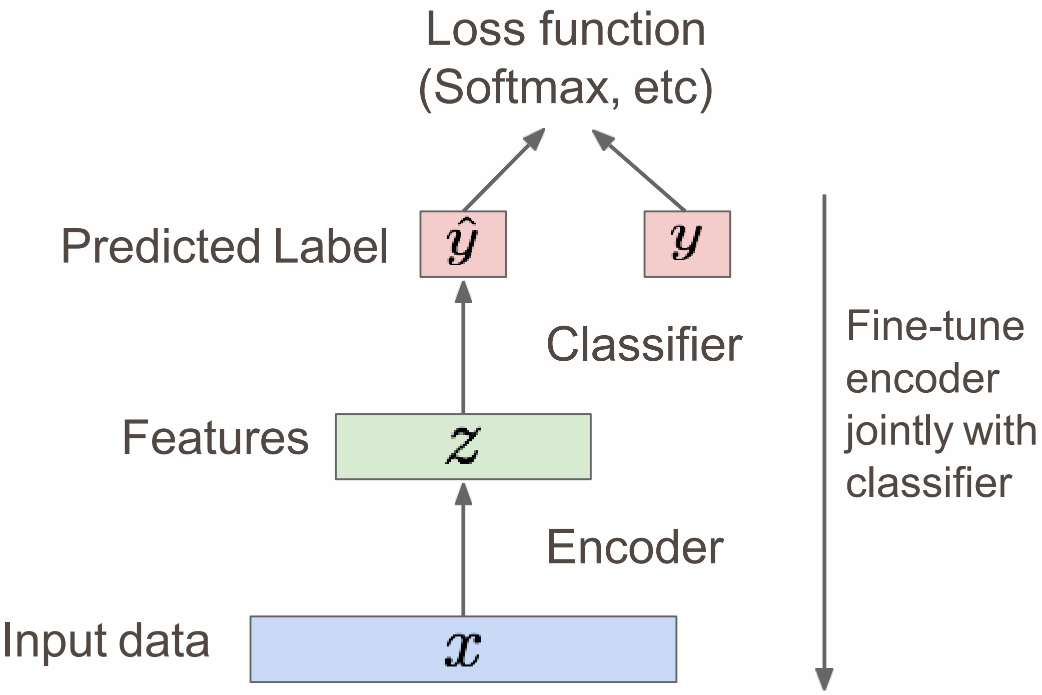


After training, throw away decoder

partial credit by CS311n

Some background first: Autoencoders

Encoder can be used to initialize a **supervised** model



Loss function (Softmax, etc)

Predicted Label $\hat{y}$

y

Classifier

Features z

Encoder

Input data x

Fine-tune encoder jointly with classifier

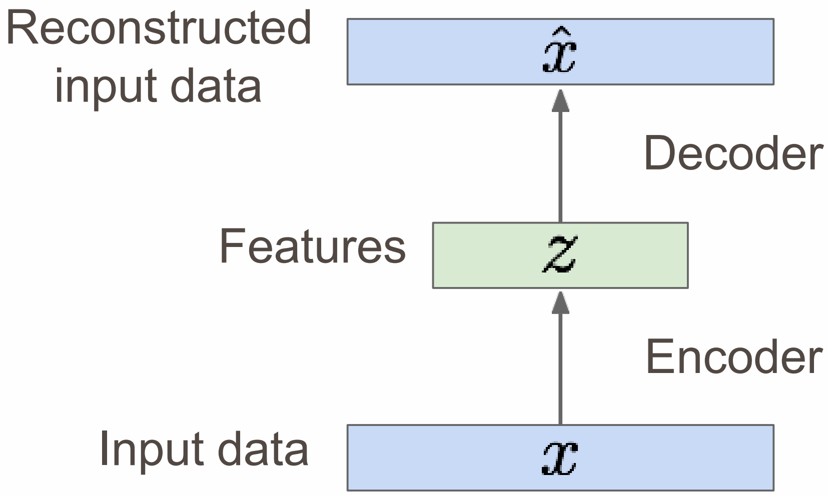
bird plane

dog deer truck

Train for final task (sometimes with small data)

partial credit by CS311n

Some background first: Autoencoders



Autoencoders can reconstruct data, and can learn features to initialize a supervised model

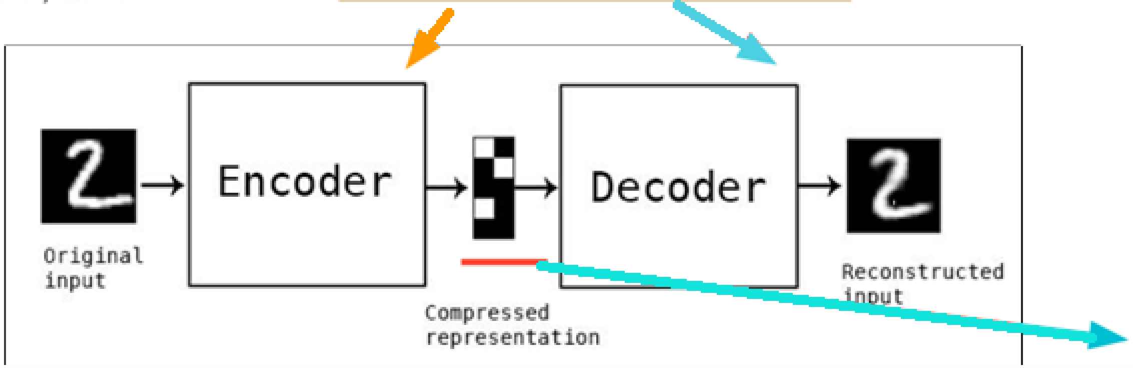
Features capture factors of variation in training data. Can we generate new images from an autoencoder?

partial credit by CS311n

Unsupervised Deep Learning: AutoEncoder

- With no answer "data"
  - Use "Reconstruction" to learn!!
  - A good representation should keep the information well (reconstruction error)
  - Deep + nonlinearity might help enhance the representation power

$$\min_{\mathbf{w}_1,\mathbf{w}_2} \|\mathbf{x}_i - g(f(\mathbf{x}_i;\mathbf{w}_1);\mathbf{w}_2)\|_2^2$$



Learnt representation

Credit: LeCun

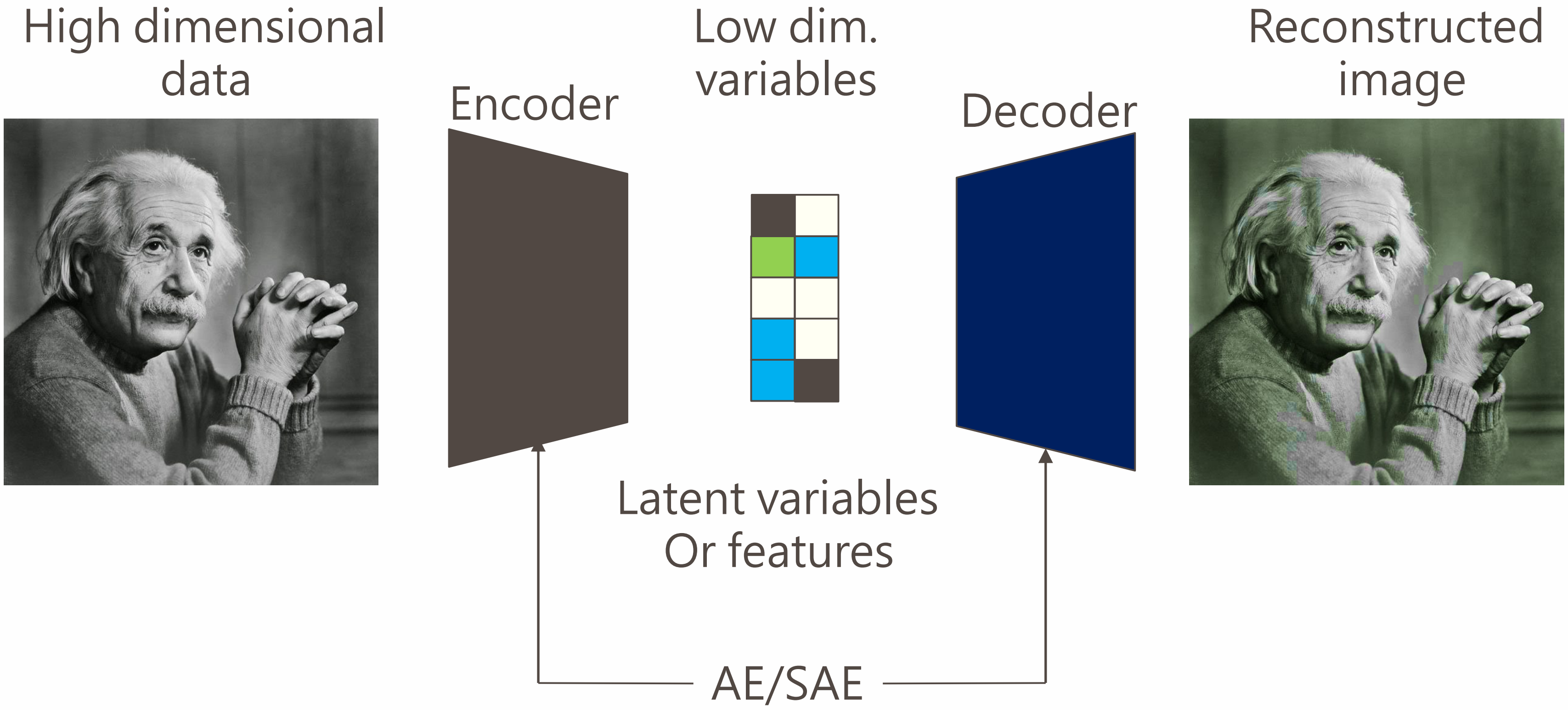
partial credit by CS311n

Deep Version of AutoEncoder

- Stacked autoencoder (SAE)
- Similar to AE but deeper
- Use CNN/Fully connected layers

partial credit by CS311n

What Exactly AE is?



High dimensional data

Encoder

Low dim. variables

Decoder

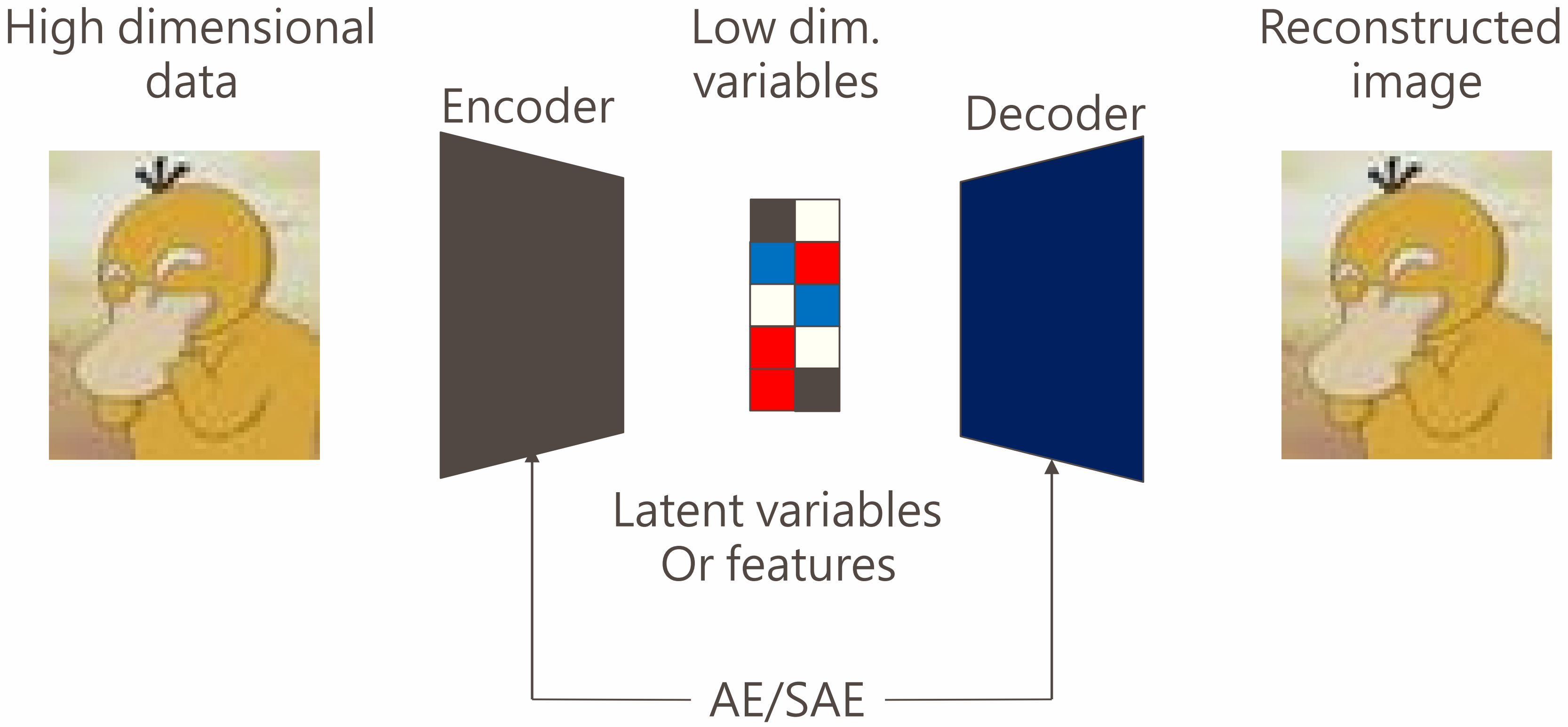
Reconstructed image

Latent variables Or features

AE/SAE

partial credit by CS311n

What Exactly AE is? (cont.)



High dimensional data

Encoder

Low dim. variables

Decoder

Reconstructed image

Latent variables Or features

AE/SAE

partial credit by CS311n

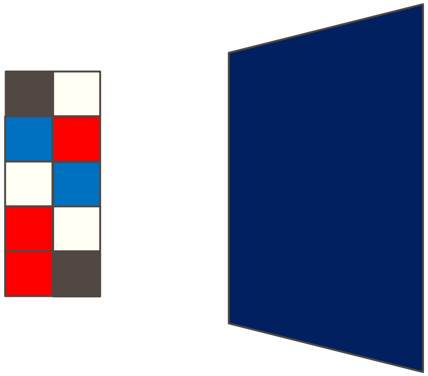
What Exactly AE is? (cont.)

Low dim.
variables 1

Decoder

Reconstructed
image 1

For example
0.666 = 可達鴨

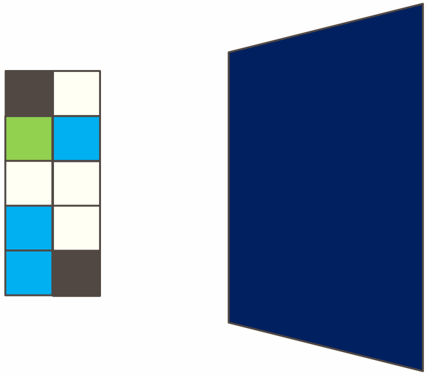


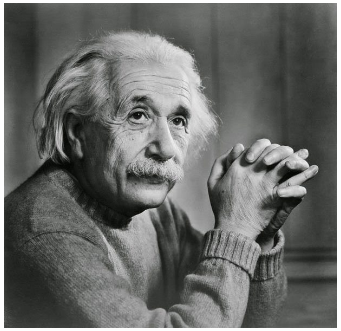
Low dim.
Variables 2

Decoder

Reconstructed
image 2

For example
0.747 = 愛因斯坦

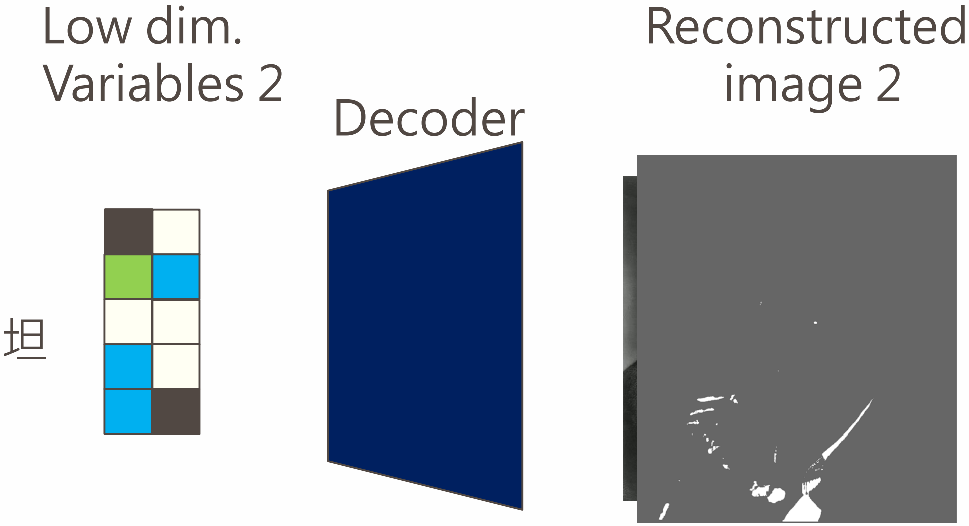




partial credit by CS311n

Problem in SAE/AE

- One feature corresponds to one reconstructed image!
  - Feature is generated from Encoder….
    - Such AE/SAE cannot be used to generate arbitrary images



For example
0.74 = 愛因斯坦
0.746 = ??

partial credit by CS311n

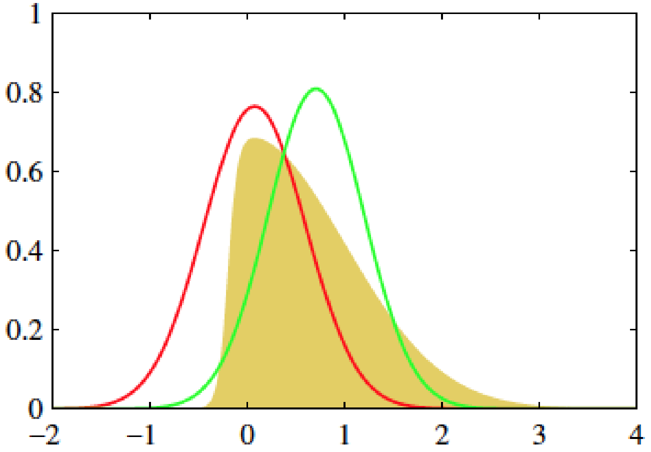
Improved AutoEncoder

- Variational autoencoder - VAE
- Kingma et al, “Auto-Encoding Variational Bayes”, 2013.
  - Generative Model + Stacked Autoencoder
  - Based on Variational approximation
- From AE to VAE
  - Since the feature (latent variable) is not continuous
    - Explicit feature is required for generating an image
  - MODELING feature instead!!

partial credit by CS311n

Variational Inference

- Target: p
  - Hard to find their distribution
- We assumed that it likes to Gaussian distributions, green and red lines *q*
  - Which one closes to the real distribution p, then we choose that q as the solution
    - Minimize distance between distributions!!



partial credit by CS311n

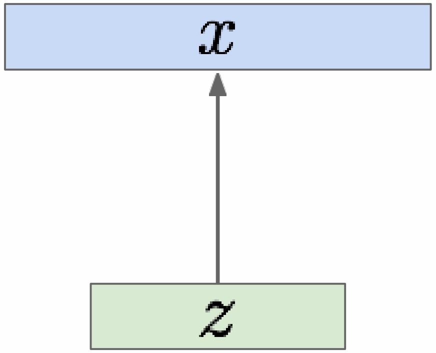
Variational Autoencoders

Probabilistic spin on autoencoders - will let us sample from the model to generate data!

Assume training data $\{x^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation **z**

Sample from
true conditional
$p_{\theta^*}(x \mid z^{(i)})$

Sample from
true prior
$p_{\theta^*}(z)$



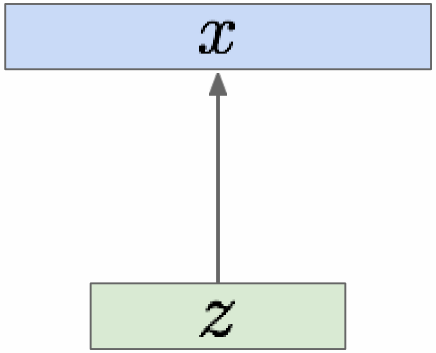
Intuition (remember from autoencoders!): **x** is an image, **z** is latent factors used to generate **x:** attributes, orientation, etc.

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

partial credit by CS311n

Variational Autoencoders

We want to estimate the true parameters θ^* of this generative model.



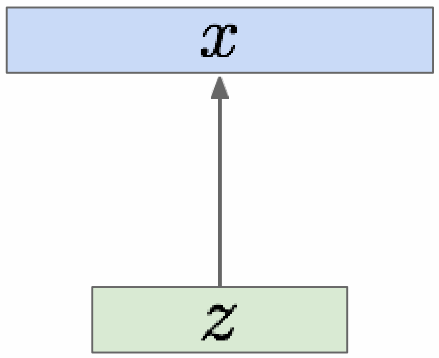
Sample from true conditional $p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior $p_{\theta^*}(z)$

Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

partial credit by CS311n

Variational Autoencoders



Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior

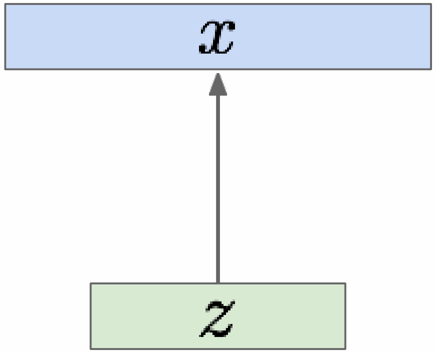
$p_{\theta^*}(z)$

We want to estimate the true parameters θ^* of this generative model.

How should we represent this model?

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

Variational Autoencoders



Sample from
true conditional
$p_{\theta^*}(x \mid z^{(i)})$

Sample from
true prior
$p_{\theta^*}(z)$

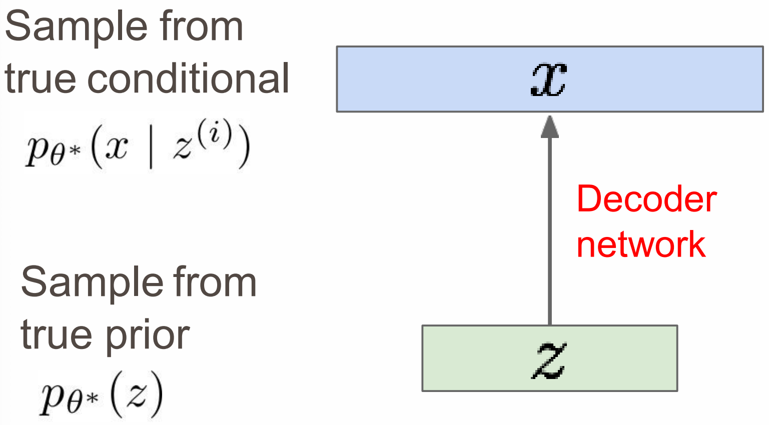
We want to estimate the true parameters θ^* of this generative model.

How should we represent this model?

Choose prior p(z) to be simple, e.g. Gaussian. Reasonable for latent attributes, e.g. pose, how much smile.

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

Variational Autoencoders



We want to estimate the true parameters θ^* of this generative model.

How should we represent this model?

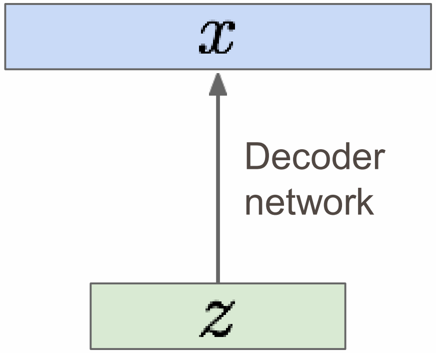
Choose prior p(z) to be simple, e.g. Gaussian.

Conditional p(x|z) is complex (generates image) => represent with neural network

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

partial credit by CS311n

Variational Autoencoders



Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior

$p_{\theta^*}(z)$

We want to estimate the true parameters θ^* of this generative model.

How to train the model?

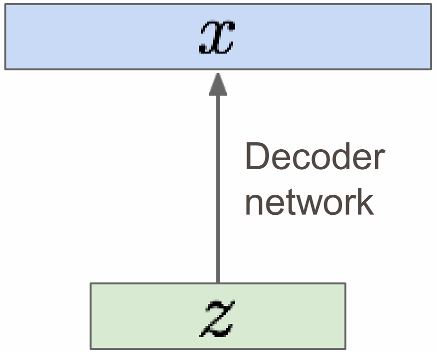
Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

partial credit by CS311n

Variational Autoencoders

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$



x

Decoder network

z

Sample from true prior

$p_{\theta^*}(z)$

We want to estimate the true parameters θ^* of this generative model.

How to train the model?

Strategy for training generative models from FVBNs (fully visible belief networks, Deep Belief nets). Learn model parameters to maximize likelihood of training data

$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

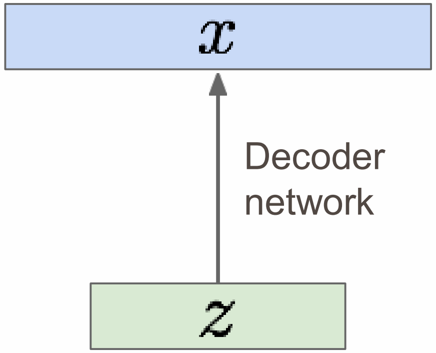
partial credit by CS311n

Variational Autoencoders

We want to estimate the true parameters θ^* of this generative model.

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$



Sample from true prior

$p_{\theta^*}(z)$

How to train the model?

Strategy for training generative models from FVBNs. Learn model parameters to maximize likelihood of training data

$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

Now with latent z

Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

partial credit by CS311n

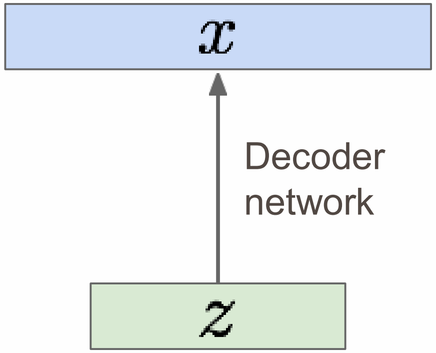
Variational Autoencoders

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior

$p_{\theta^*}(z)$



We want to estimate the true parameters θ^* of this generative model.

How to train the model?

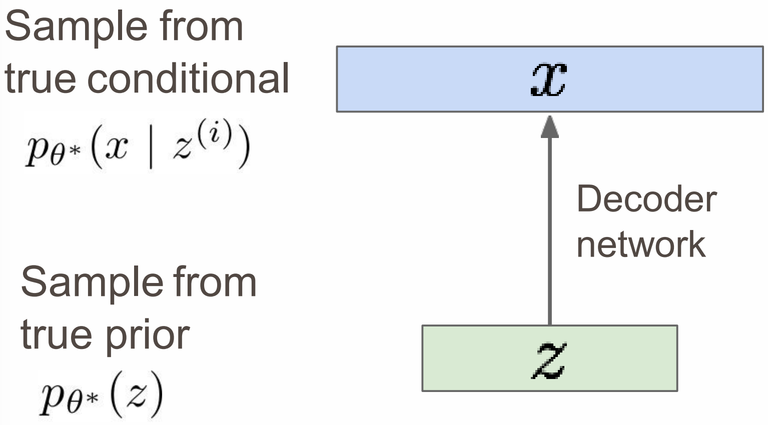
Strategy for training generative models from FVBNs, Learn model parameters to maximize likelihood of training data

$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

Q: What is the problem with this?

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

Variational Autoencoders


Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior

$p_{\theta^*}(z)$

We want to estimate the true parameters θ^* of this generative model.

How to train the model?

Remember strategy for training generative models from FVBNs. Learn model parameters to maximize likelihood of training data

$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

Q: What is the problem with this?

Intractable!

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

$$p(x) = \frac{p(x,z)}{p(z|x)} \Rightarrow p(z|x) = \frac{p(x,z)}{\boxed{p(x)}}$$

Given p(x)

partial credit by CS311n

Variational Autoencoders: Intractability

Data likelihood: $$p_\theta(x) = \overbrace{\int}^{\text{Intractable to compute p(x|z) for every z}} \underbrace{p_\theta(z)}_{\text{Gaussian prior}}\underbrace{p_\theta(x|z)}_{\text{NN decoder}}dz$$

$$p_\theta(z|x) = p_\theta(x|z)p_\theta(z)/\underbrace{p_\theta(x)}_{\text{Intractable ...}}$$

- Posterior density also intractable:

Solution: In addition to decoder network modeling $p_\theta(x|z)$, define additional encoder network $q_\phi(z|x)$ that approximates $p_\theta(x|z)$,

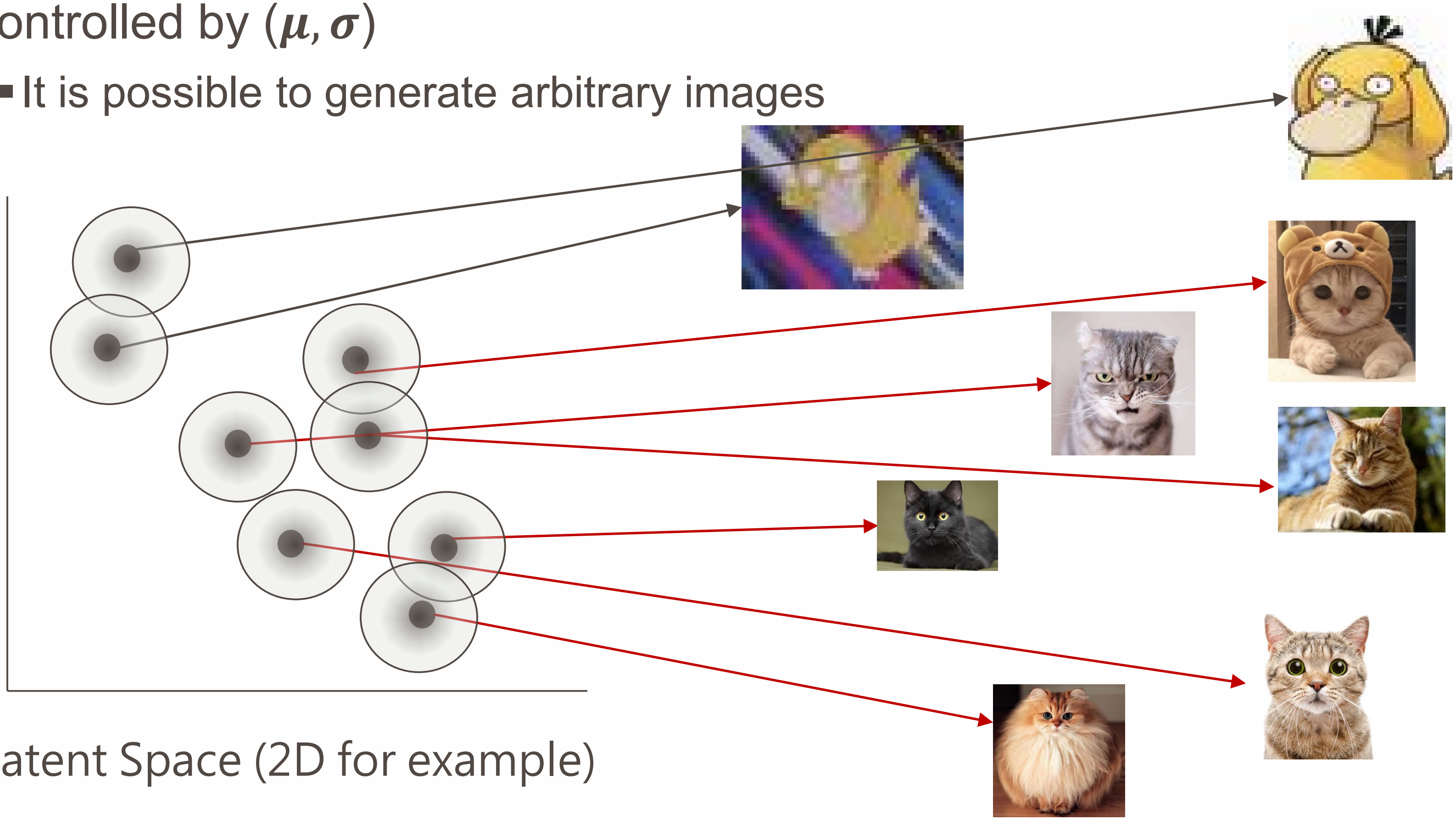
- Will see that this allows us to derive a lower bound on the data likelihood that is tractable, which we can optimize

Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

partial credit by CS311n

From AE to VAE

- Modeling: Assume the feature is sampled from Gaussian controlled by $(\boldsymbol{\mu}, \boldsymbol{\sigma})$
  - It is possible to generate arbitrary images



partial credit by CS311n

From AE to VAE

- In this way, loss function can be defined as
  - $L_{data} = \|X - \bar{X}\|_2^2, \text{where } \bar{X}$ is the reconstructed image
  - $L_{latent} = \mathrm{KL}(\mathrm{P}|\mathrm{Q})$ ➔ KL(Latent variables, Gaussian)
  - $L = L_{data} + L_{latent}$

- Difficult to optimize L
  - The distribution of latent variables is unknown & uncontrollable.
- Solution:
  - Force latent variable to be a parameters of a specified distribution: Encoder ➔ $(\boldsymbol{\mu}, \boldsymbol{\sigma})$

partial credit by CS311n

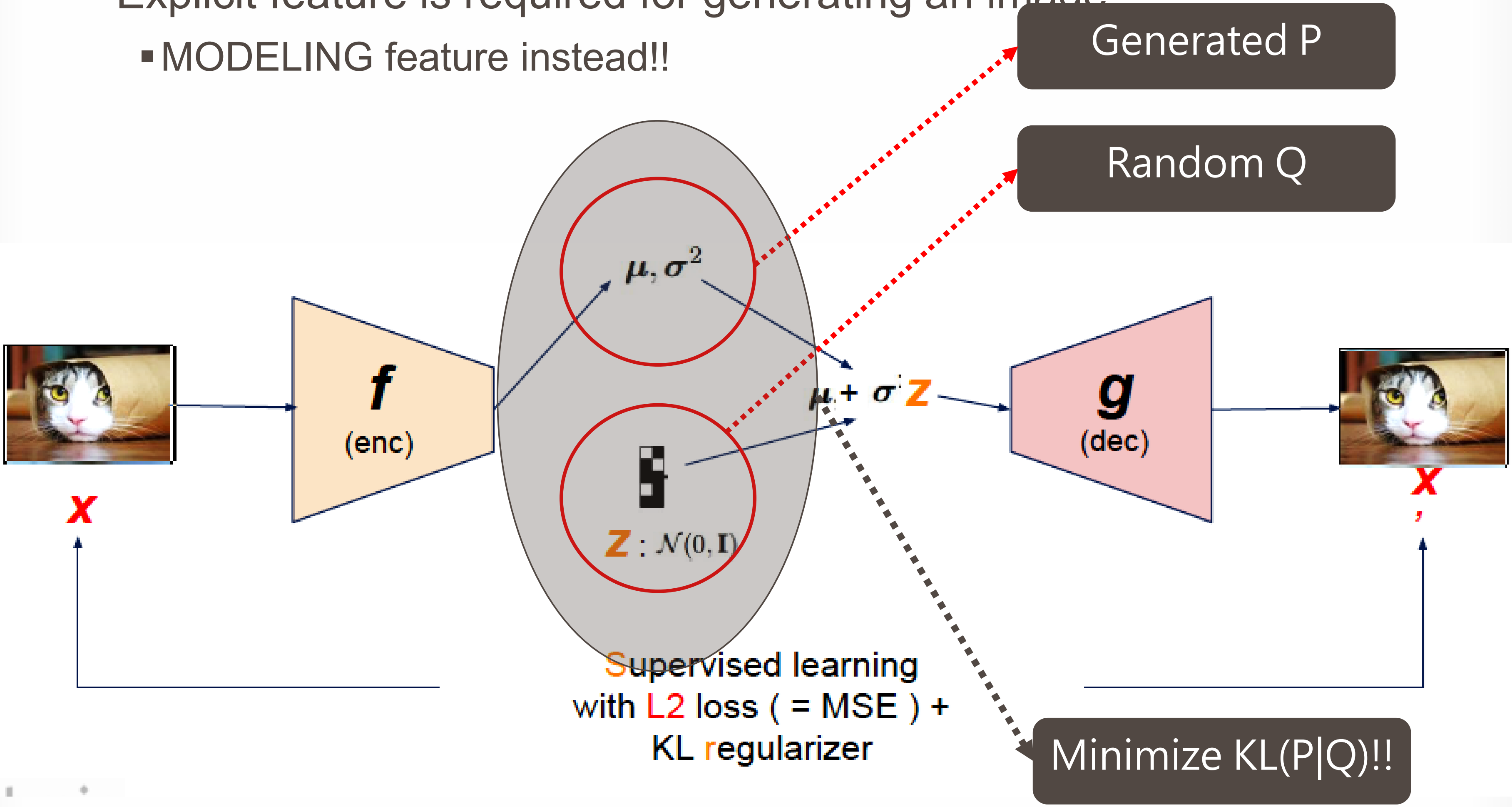
Variational AE (VAE)

- Minimize KL(P|Q)!!
  - ➔ Variational inference!!
- Recall that
  - P(z|x) = P(x, z) / P(x) ➔ P(x) Intractable (ELBO)
  - Approximation solution
    - Use q(z|θ) to approximate P(z|x)
    - Variational inference!
- Shortcoming
  - Blurred images will be generated (no guarantee its quality)

partial credit by CS311n

From AE to VAE

- Explicit feature is required for generating an image
  - MODELING feature instead!!



Generated P

Random Q

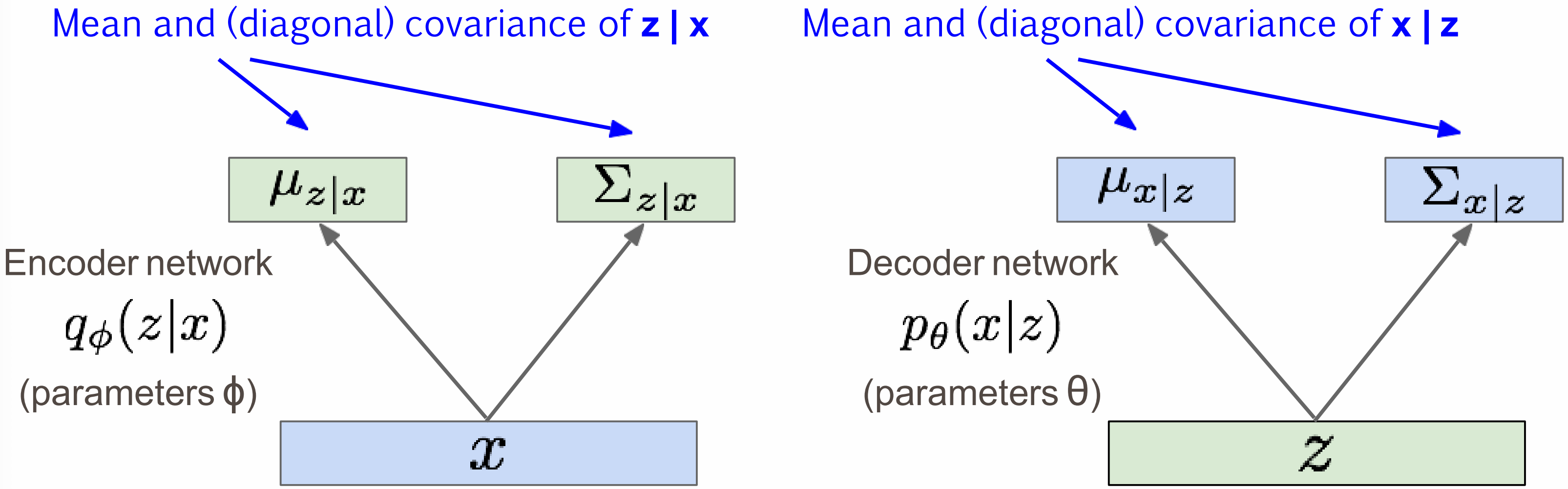
Minimize KL(P|Q)!!

Supervised learning with L2 loss (= MSE) + KL regularizer

partial credit by CS311n

Variational Autoencoders

Since we're modeling probabilistic generation of data, encoder and decoder networks are probabilistic



Mean and (diagonal) covariance of **z | x**

Mean and (diagonal) covariance of **x | z**

$\mu_{z|x}$ $\Sigma_{z|x}$

Encoder network

$q_\phi(z|x)$

(parameters ϕ)

x

$\mu_{x|z}$ $\Sigma_{x|z}$

Decoder network

$p_\theta(x|z)$

(parameters θ)

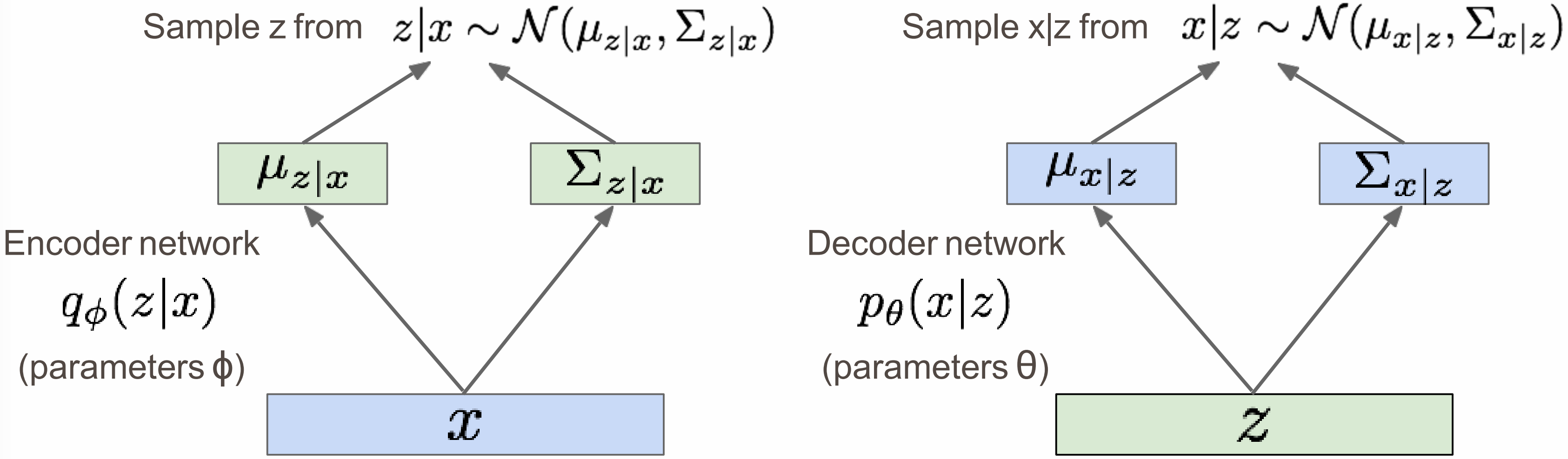
z

Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

partial credit by CS311n

Variational Autoencoders

Since we're modeling probabilistic generation of data, encoder and decoder networks are probabilistic



Sample z from $z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$

$\mu_{z|x}$ $\Sigma_{z|x}$

Encoder network

$q_\phi(z|x)$

(parameters ϕ)

x

Sample x|z from $x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$

$\mu_{x|z}$ $\Sigma_{x|z}$

Decoder network

$p_\theta(x|z)$

(parameters θ)

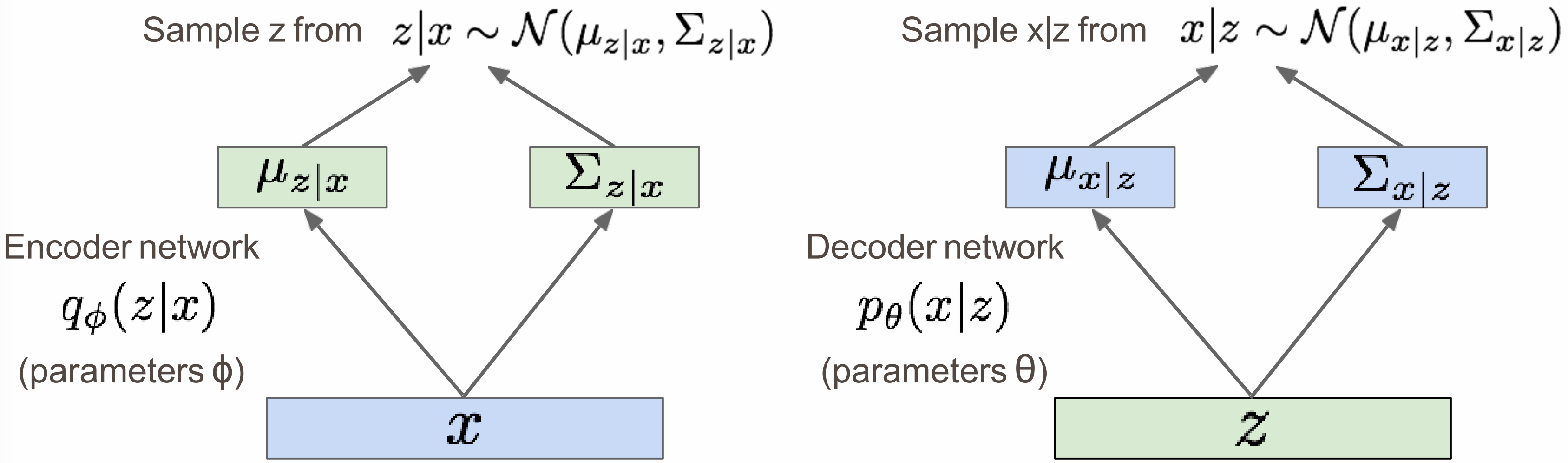
z

Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

partial credit by CS311n

Variational Autoencoders

Since we're modeling probabilistic generation of data, encoder and decoder networks are probabilistic



Sample z from $z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$

$\mu_{z|x}$ $\Sigma_{z|x}$

Encoder network

$q_\phi(z|x)$

(parameters ϕ)

x

Sample x|z from $x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$

$\mu_{x|z}$ $\Sigma_{x|z}$

Decoder network

$p_\theta(x|z)$

(parameters θ)

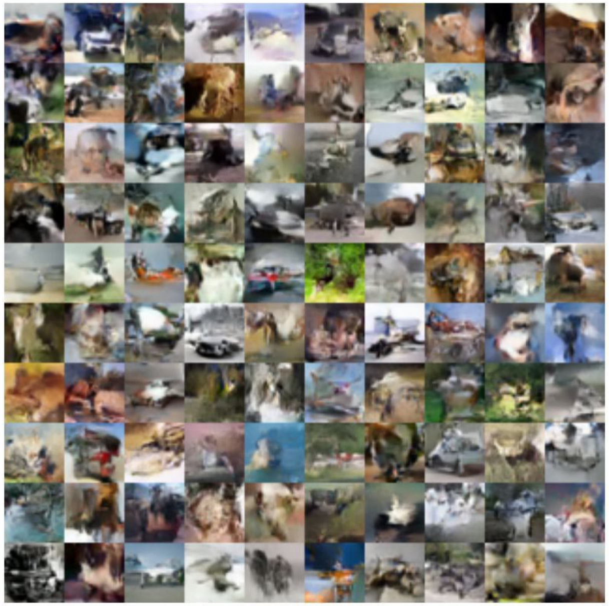
z

Encoder and decoder networks also called "recognition"/"inference" and "generation" networks

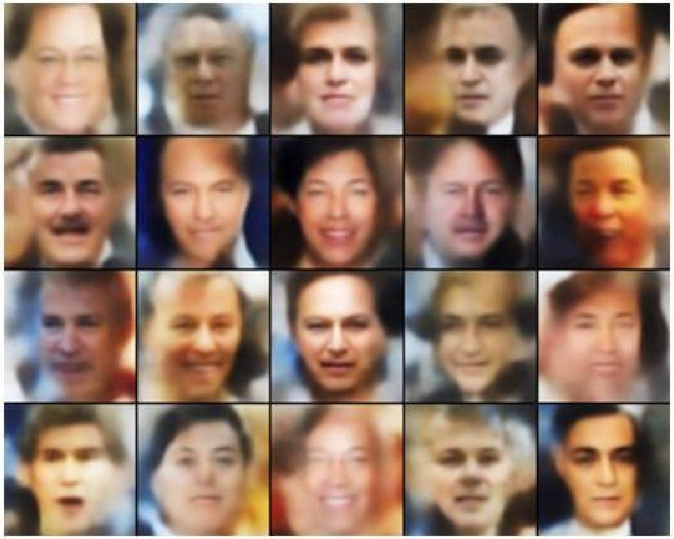
Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

partial credit by CS311n

Variational Autoencoders: Generating Data!



32x32 CIFAR-10



Labeled Faces in the Wild

Figures copyright (L) Dirk Kingma et al. 2016; (R) Anders Larsen et al. 2017. Reproduced with permission.

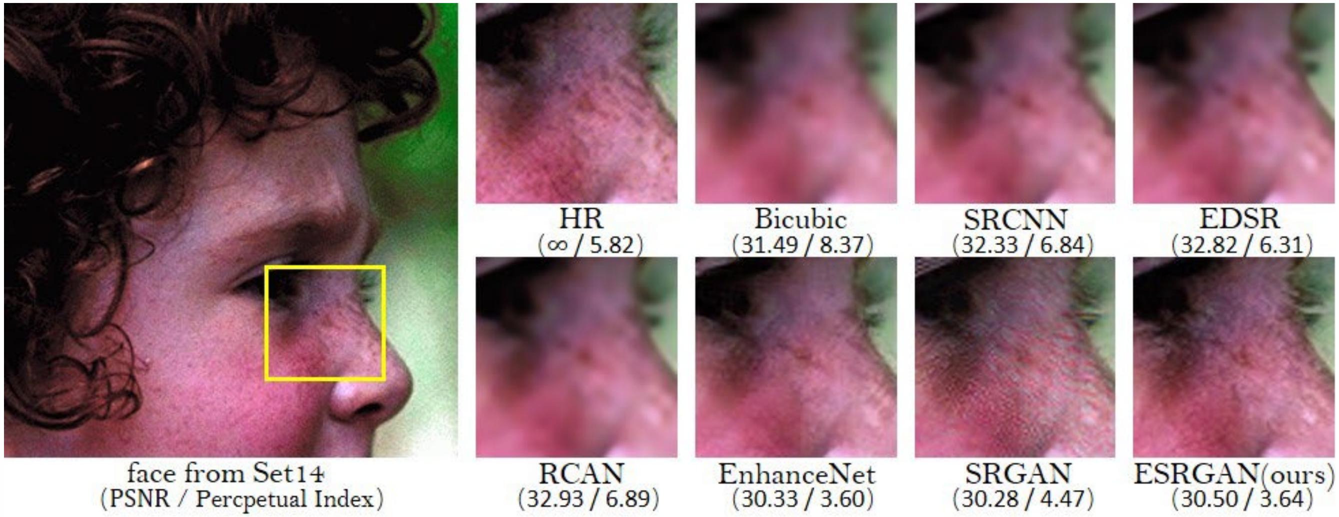
Variational Autoencoders

- Probabilistic spin to traditional autoencoders => allows generating data
- Defines an intractable density => derive and optimize a (variational) lower bound
- Pros:
  - Principled approach to generative models
  - Allows inference of q(z|x), can be useful feature representation for other tasks
- Cons:
  - Maximizes lower bound of likelihood: okay, but not as good evaluation as PixelRNN/PixelCNN
  - Samples blurrier and lower quality compared to state-of-the-art (GANs)
- Active areas of research:
  - More flexible approximations, e.g. richer approximate posterior instead of diagonal Gaussian, e.g., Gaussian Mixture Models (GMMs)
  - Incorporating structure in latent variables, e.g., Categorical Distributions

partial credit by CS311n

But Why?

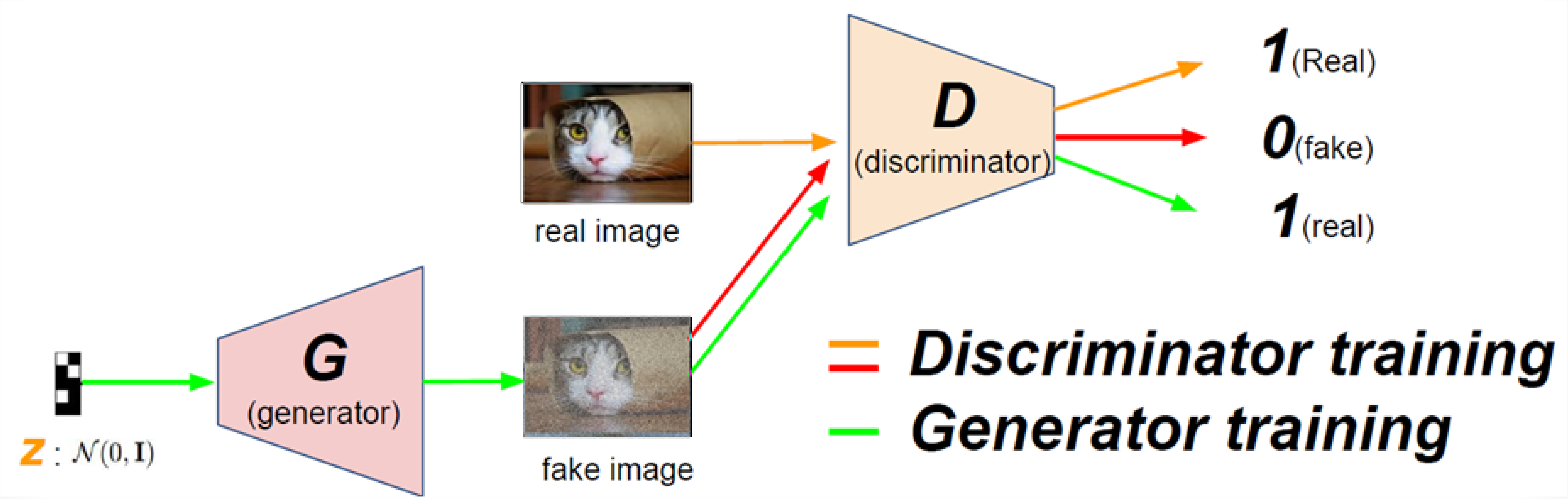
- Recall that we use “pixel” to measure the quality
  - So what?
  - It is well-known that there is no promising metric that can reflect the truly perceptual quality (visual quality)
- Example
  - PSNR
  - Some perceptual index were proposed to resolve this issue.



partial credit by CS311n

Unsupervised Deep Learning

- How to generate an image with good quality?
  - Generative adversarial network (GAN)



Goodfellow, Ian, et al. "Generative adversarial nets." *Advances in neural information processing systems*. 2014.

partial credit by CS311n

Why Generative Models?

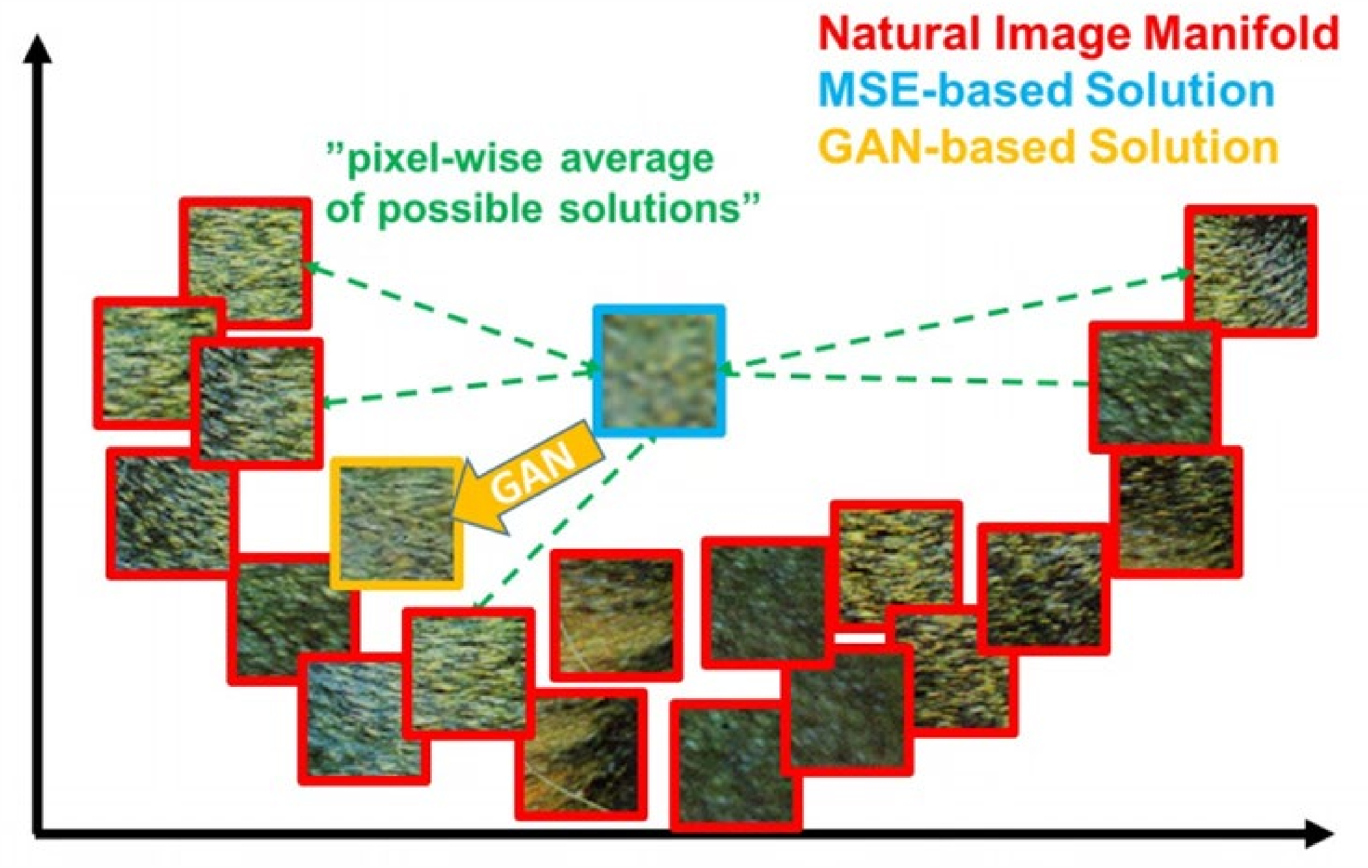
- Excellent test of our ability to use high-dimensional, complicated probability distributions
- Simulate possible futures for planning or simulated RL
- Missing data
  - Semi-supervised learning
- Multi-modal outputs
- Realistic generation tasks

(Goodfellow 2016)

partial credit by CS311n

Generating an Image using GAN

- Learn and predict P(x|z)



Ledig et al. CVPR 2017

partial credit by CS311n

Training Procedure

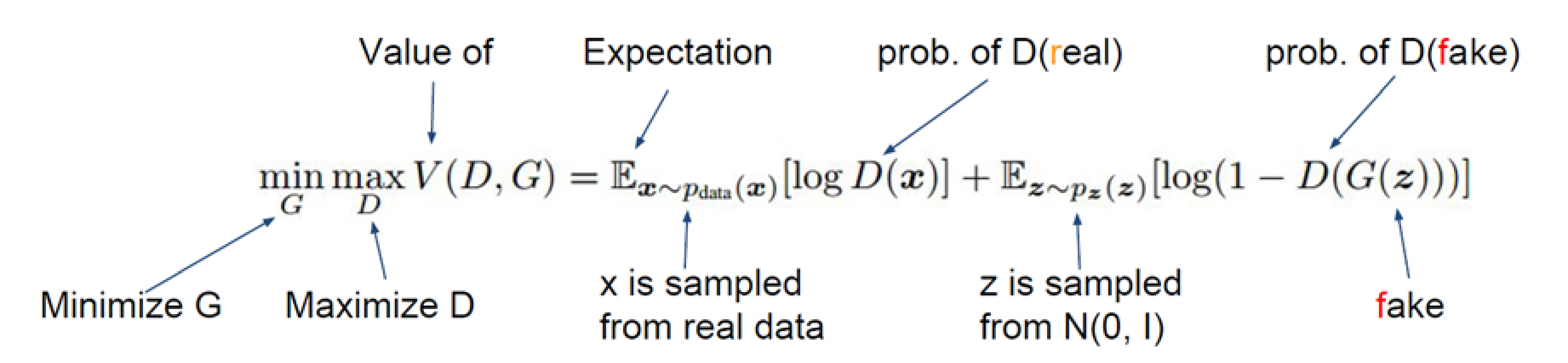
- Use SGD-like algorithm of choice (Adam) on two mini-batches simultaneously:
    - A mini-batch of training examples
    - A mini-batch of generated samples
- Optional: run k steps of one player for every step of the other player.

(Goodfellow 2016)

partial credit by CS311n

The Cost Function of GAN

- Notation



Value of

Expectation

prob. of D(real)

prob. of D(fake)

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Minimize G

Maximize D

x is sampled
from real data

z is sampled
from N(0, I)

fake

Ian Goodfellow et al., “Generative Adversarial Nets”, NIPS 2014

partial credit by CS311n

Training GANs: Two-player game

- Generator network: try to fool the discriminator by generating real-looking images
- Discriminator network: try to distinguish between real and fake images
- Train jointly in minimax game
- Minimax objective function:

Discriminator outputs likelihood in (0,1) of real image

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Discriminator output for real data x

Discriminator output for generated fake data G(z)

- Discriminator (θ_d) wants to **maximize objective** such that D(x) is close to 1 (real) and D(G(z)) is close to 0 (fake)
- Generator (θ_g) wants to **minimize objective** such that D(G(z)) is close to 1 (discriminator is fooled into thinking generated G(z) is real)

partial credit by CS311n

Training GANs: Two-player game

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

1. **Gradient ascent** on discriminator

$$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. **Gradient descent** on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

partial credit by CS311n

Training GANs: Two-player game

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x\sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z\sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))\right]$$

Alternate between:

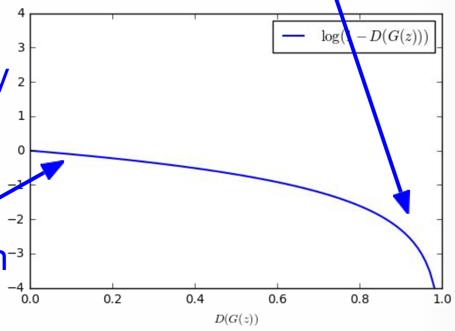
1. **Gradient ascent** on discriminator

$$\max_{\theta_d} \left[\mathbb{E}_{x\sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z\sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))\right]$$

2. **Gradient descent** on generator

$$\min_{\theta_g} \mathbb{E}_{z\sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

In practice, optimizing this generator objective does not work well!



Gradient signal dominated by region where sample is already good

When sample is likely fake, want to learn from it to improve generator. But gradient in this region is relatively flat!

partial credit by CS311n

Training GANs: Two-player game

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x\sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z\sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

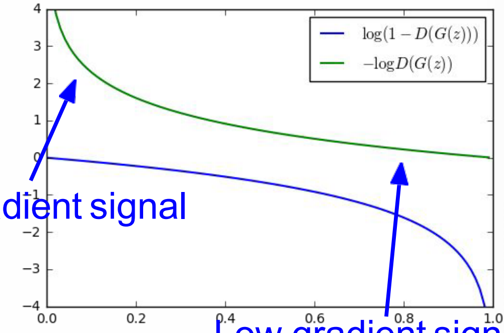
1. **Gradient ascent** on discriminator

$$\max_{\theta_d} \left[\mathbb{E}_{x\sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z\sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. **Instead: Gradient ascent** on generator, different objective

$$\max_{\theta_g} \mathbb{E}_{z\sim p(z)} \log(D_{\theta_d}(G_{\theta_g}(z)))$$

Instead of minimizing likelihood of discriminator being correct, now maximize likelihood of discriminator being wrong.
Same objective of fooling discriminator, but now higher gradient signal for bad samples => works much better! Standard in practice.



High gradient signal

Low gradient signal

partial credit by CS311n

Training GANs: Two-player game

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

1. **Gradient ascent** on discriminator

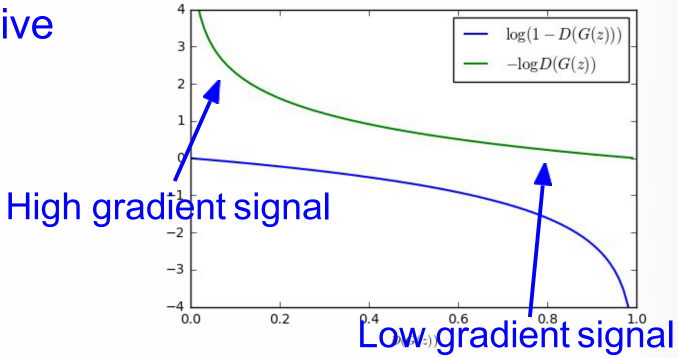
$$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. **Instead: Gradient ascent** on generator, different objective

$$\max_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(D_{\theta_d}(G_{\theta_g}(z)))$$

Instead of minimizing likelihood of discriminator being correct, now maximize likelihood of discriminator being wrong.
Same objective of fooling discriminator, but now higher gradient signal for bad samples => works much better! Standard in practice.

Aside: Jointly training two networks is challenging, can be unstable. Choosing objectives with better loss landscapes helps training, is an active area of research.



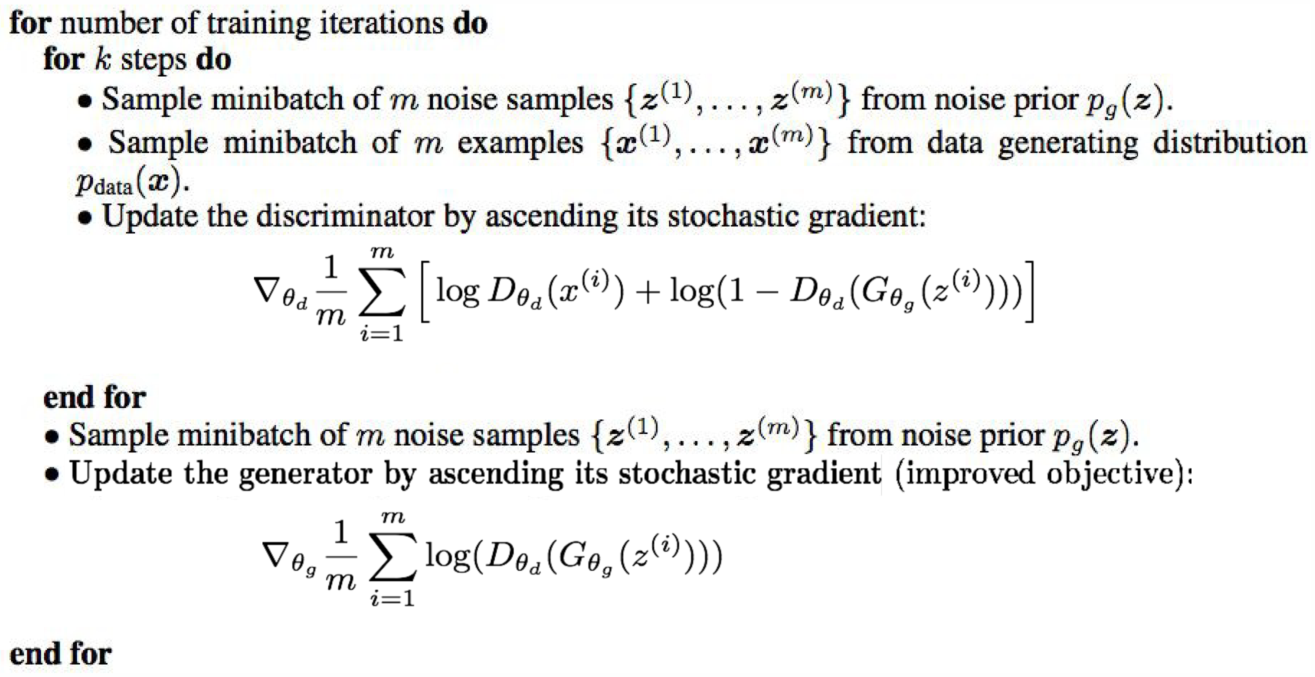
High gradient signal

Low gradient signal

partial credit by CS311n

Training GANs: Two-player game

Putting it together: GAN training algorithm



```
for number of training iterations do
    for $k$ steps do
        • Sample minibatch of $m$ noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
        • Sample minibatch of $m$ examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
        • Update the discriminator by ascending its stochastic gradient:
            $\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[ \log D_{\theta_d}(x^{(i)}) + \log(1 - D_{\theta_d}(G_{\theta_g}(z^{(i)}))) \right]$
    end for
    • Sample minibatch of $m$ noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
    • Update the generator by ascending its stochastic gradient (improved objective):
        $\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log(D_{\theta_d}(G_{\theta_g}(z^{(i)})))$
end for
```

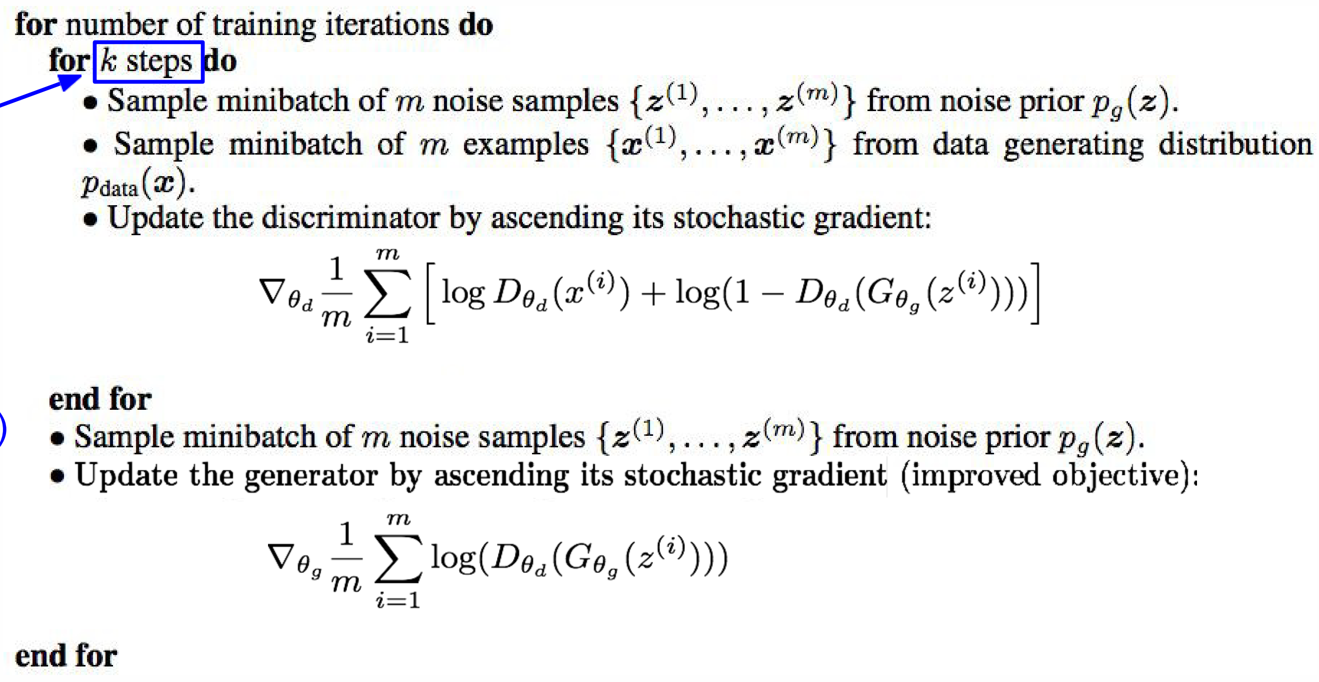
partial credit by CS311n

Training GANs: Two-player game

Putting it together: GAN training algorithm

Some find k=1 more stable, others use k > 1, no best rule.

Recent work (e.g. Wasserstein GAN) alleviates this problem, better stability!

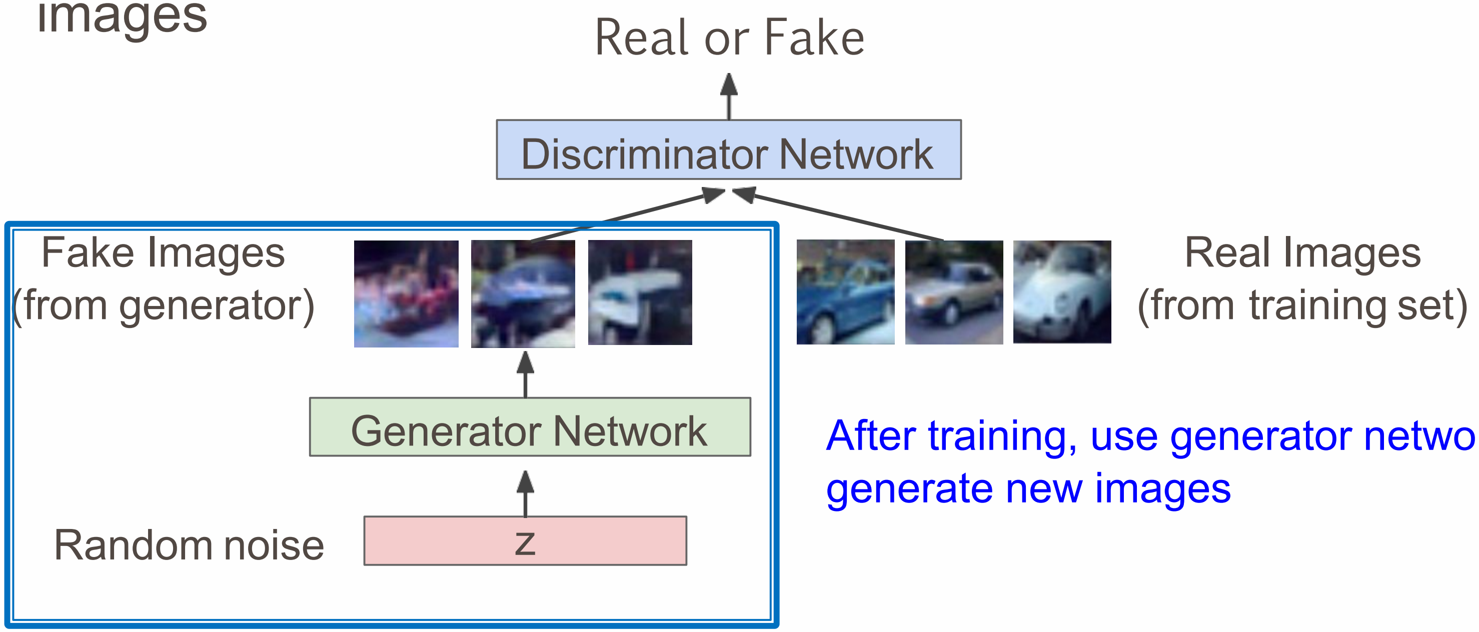


```
for number of training iterations do
    for $k$ steps do
        • Sample minibatch of $m$ noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
        • Sample minibatch of $m$ examples $\{x^{(1)}, \dots, x^{(m)}\}$ from data generating distribution $p_{\text{data}}(x)$.
        • Update the discriminator by ascending its stochastic gradient:
            $$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[ \log D_{\theta_d}(x^{(i)}) + \log(1 - D_{\theta_d}(G_{\theta_g}(z^{(i)}))) \right]$$
    end for
    • Sample minibatch of $m$ noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
    • Update the generator by ascending its stochastic gradient (improved objective):
        $$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log(D_{\theta_d}(G_{\theta_g}(z^{(i)})))$$
end for
```

partial credit by CS311n

Training GANs: Two-player game

- Generator network: try to fool the discriminator by generating real-looking images
- Discriminator network: try to distinguish between real and fake images


Real or Fake

Discriminator Network

Fake Images
(from generator)

Real Images
(from training set)

Generator Network

Random noise

z

After training, use generator network to generate new images

partial credit by CS311n

Problems in GANs

- No guarantee to equilibrium
  - Mode collapsing
    - All smoothing results
  - Oscillation
    - May never converge
  - No indicator when to finish
- All generative models
  - Evaluation metrics (predefined)
  - Robust but difficult to train
  - Diversity testing is required

partial credit by CS311n

GAN’s Ways

- Theatrical analysis of the nature of the GANs
  - WGAN
    - Wasserstein GAN (Replace KL with Wasserstein)
    - Solved the issue when there is no overlapping between distributions of generated & ground truth samples
  - BEGAN
  - WGAN-GP
  - RAGAN
  - …etc
- Applications
  - Based on a state-of-the-art GAN and fine-tune it.

partial credit by CS311n

Improved GAN: DCGAN

- Radford et al, Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, 2015
  - Tricks for gradient flow
    - Max pooling → Strided convolution or average pooling
    - Use LeakyReLU instead of ReLU
  - Other tricks
    - Use batch normal both generator and discriminator
    - Use Adam optimizer (lr = 0.0002, a = 0.9, b=0.5)

Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." *arXiv preprint arXiv:1511.06434* (2015).

partial credit by CS311n

DCGAN

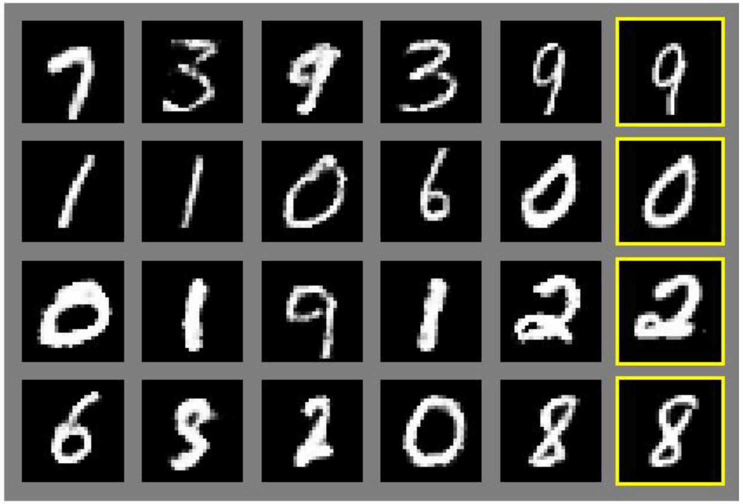
- Convert max-pooling layers to convolution layers
- Convert fully connected layers to global average pooling layers in the discriminator
- Use batch normalization layers in the generator and the discriminator
- Use leaky-ReLU activation functions in the discriminator
- Other tricks
  - Use Adam optimizer (lr = 0.0002, a = 0.9, b=0.5)

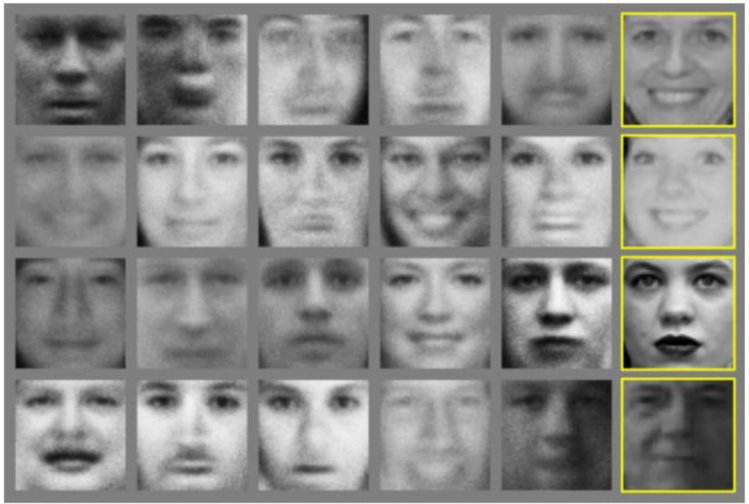
Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." *arXiv preprint arXiv:1511.06434* (2015).

partial credit by CS311n

Generative Adversarial Nets

Generated samples



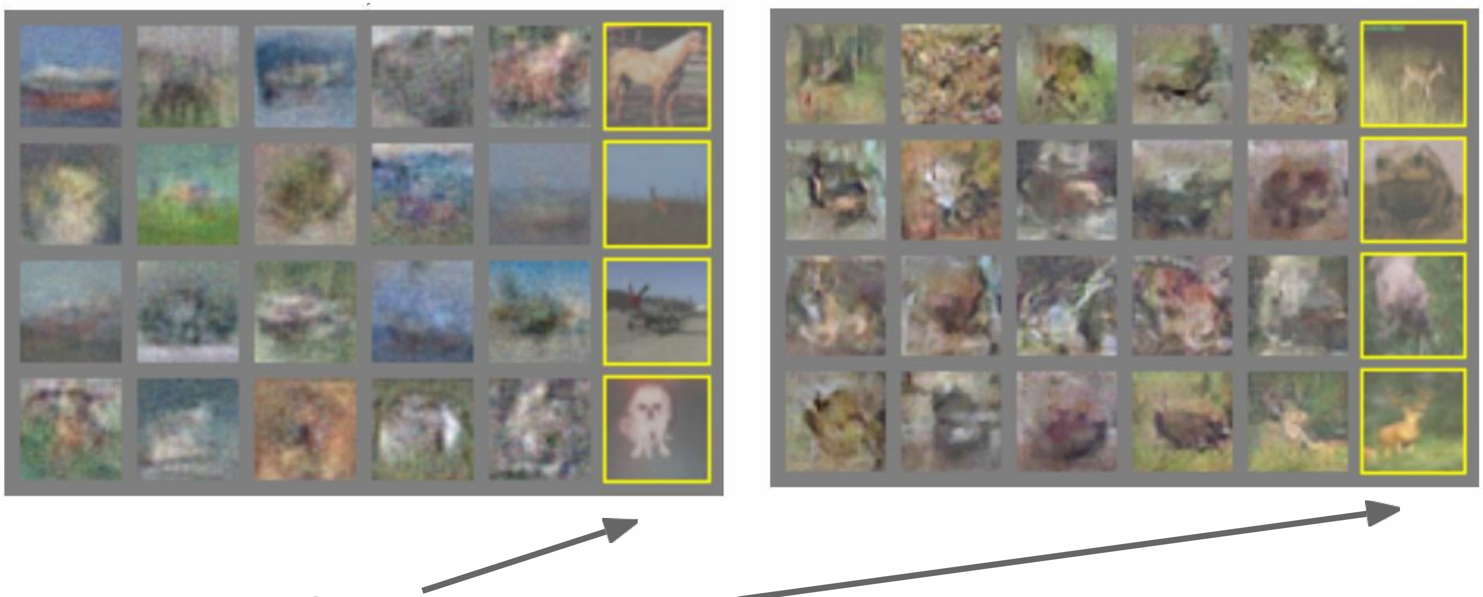


Nearest neighbor from training set

Figures copyright Ian Goodfellow et al., 2014. Reproduced with permission.

Generative Adversarial Nets

Generated samples (CIFAR-10)

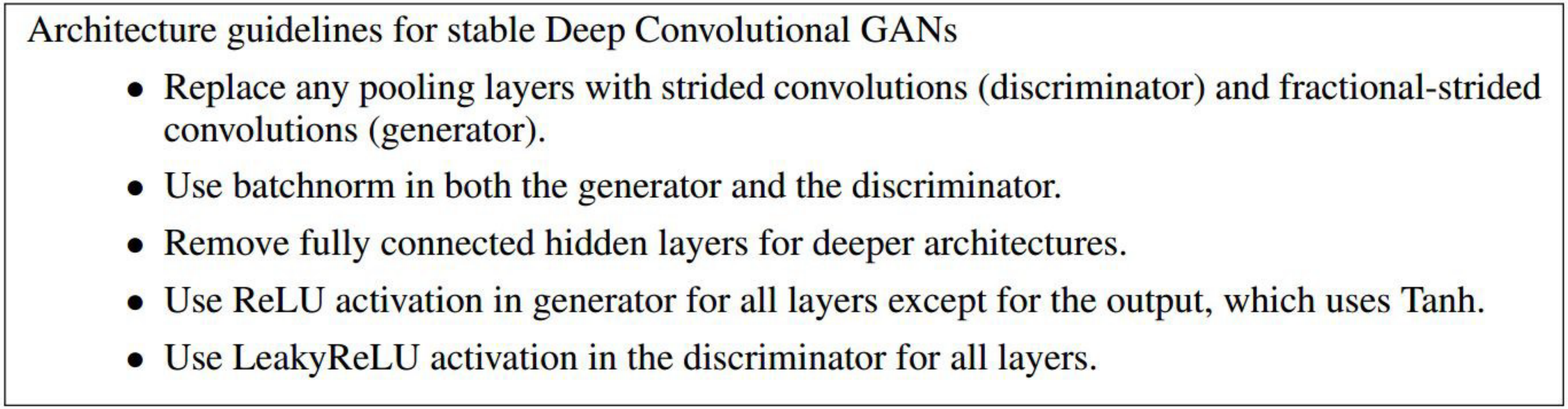


Nearest neighbor from training set

Figures copyright Ian Goodfellow et al., 2014. Reproduced with permission.

Generative Adversarial Nets: Convolutional Architectures

Generator is an upsampling network with fractionally-strided convolutions
Discriminator is a convolutional network



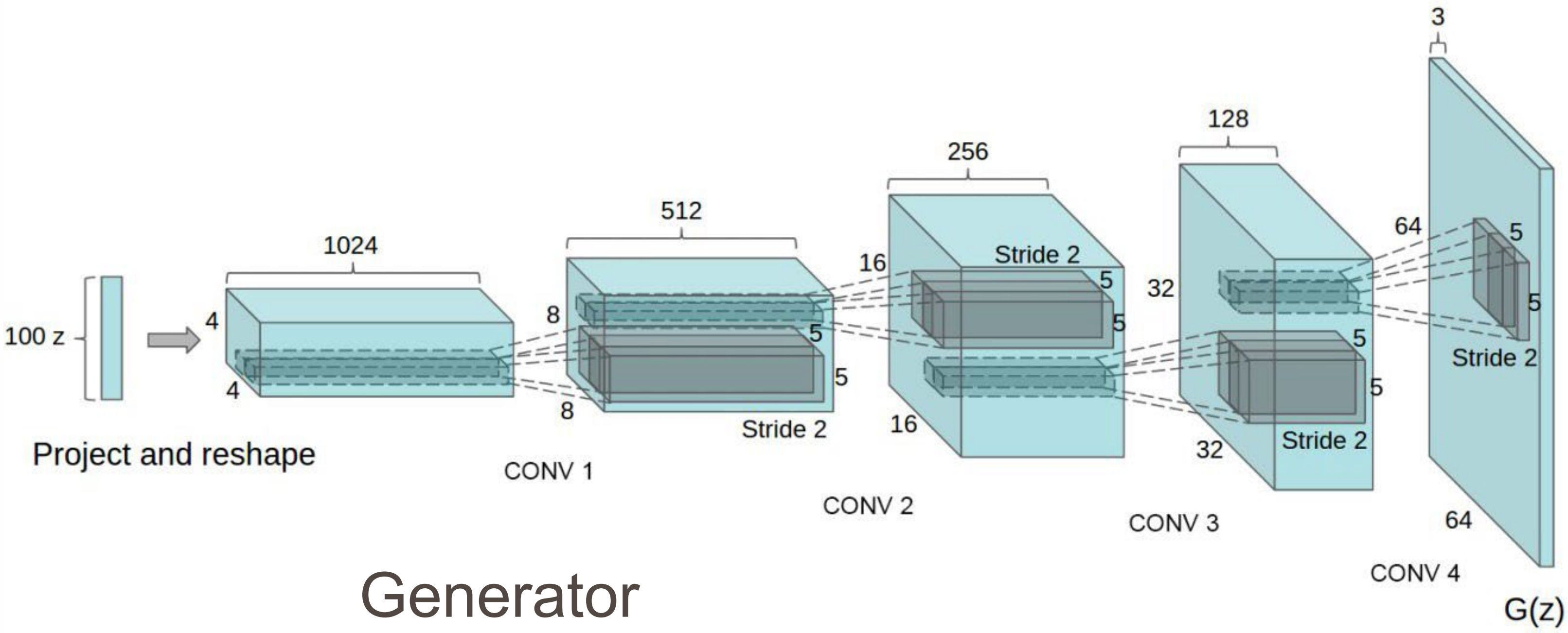
Architecture guidelines for stable Deep Convolutional GANs

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

Radford et al, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

partial credit by CS311n

Generative Adversarial Nets: Convolutional Architectures

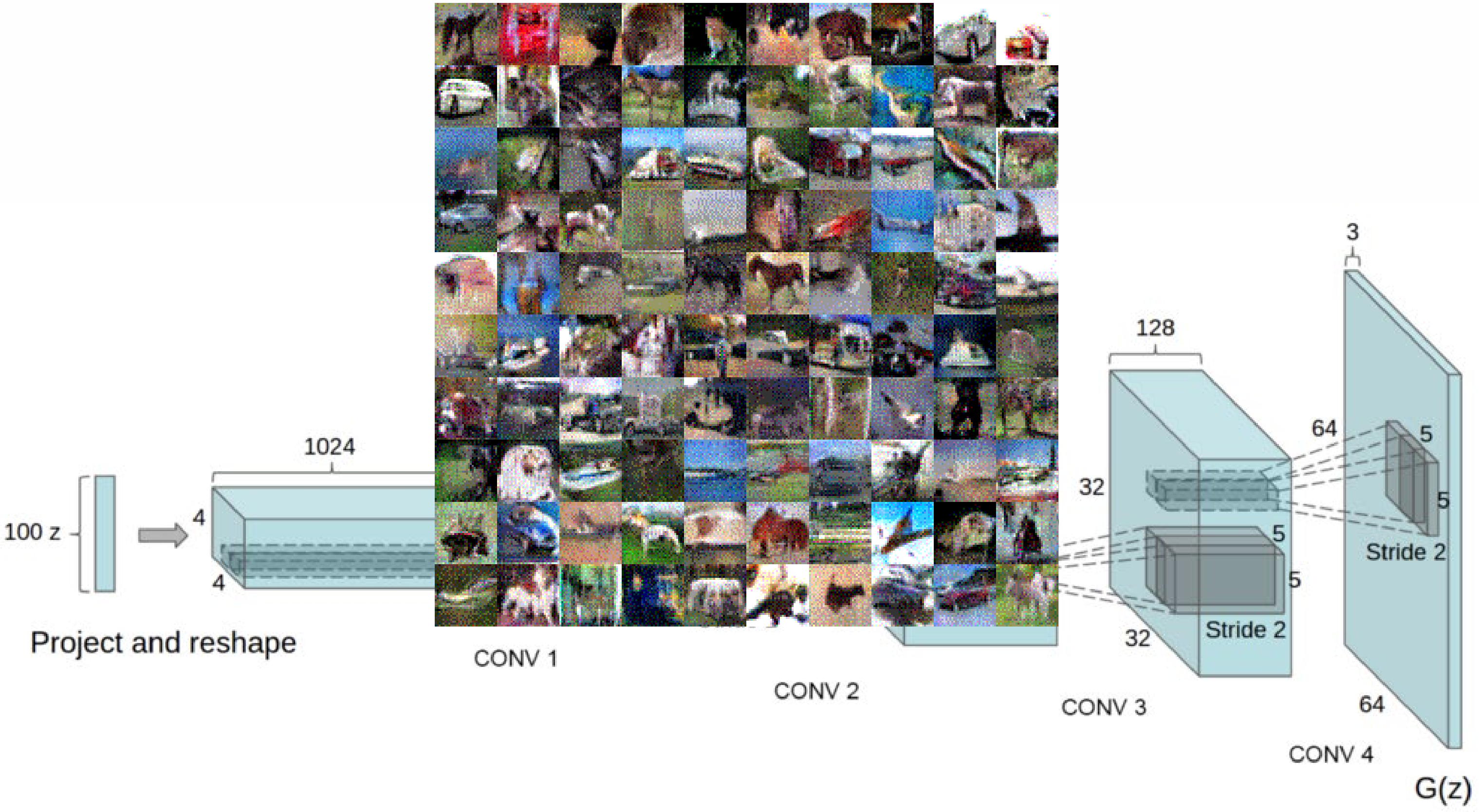


Generator

Radford et al, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", ICLR 2016

partial credit by CS311n

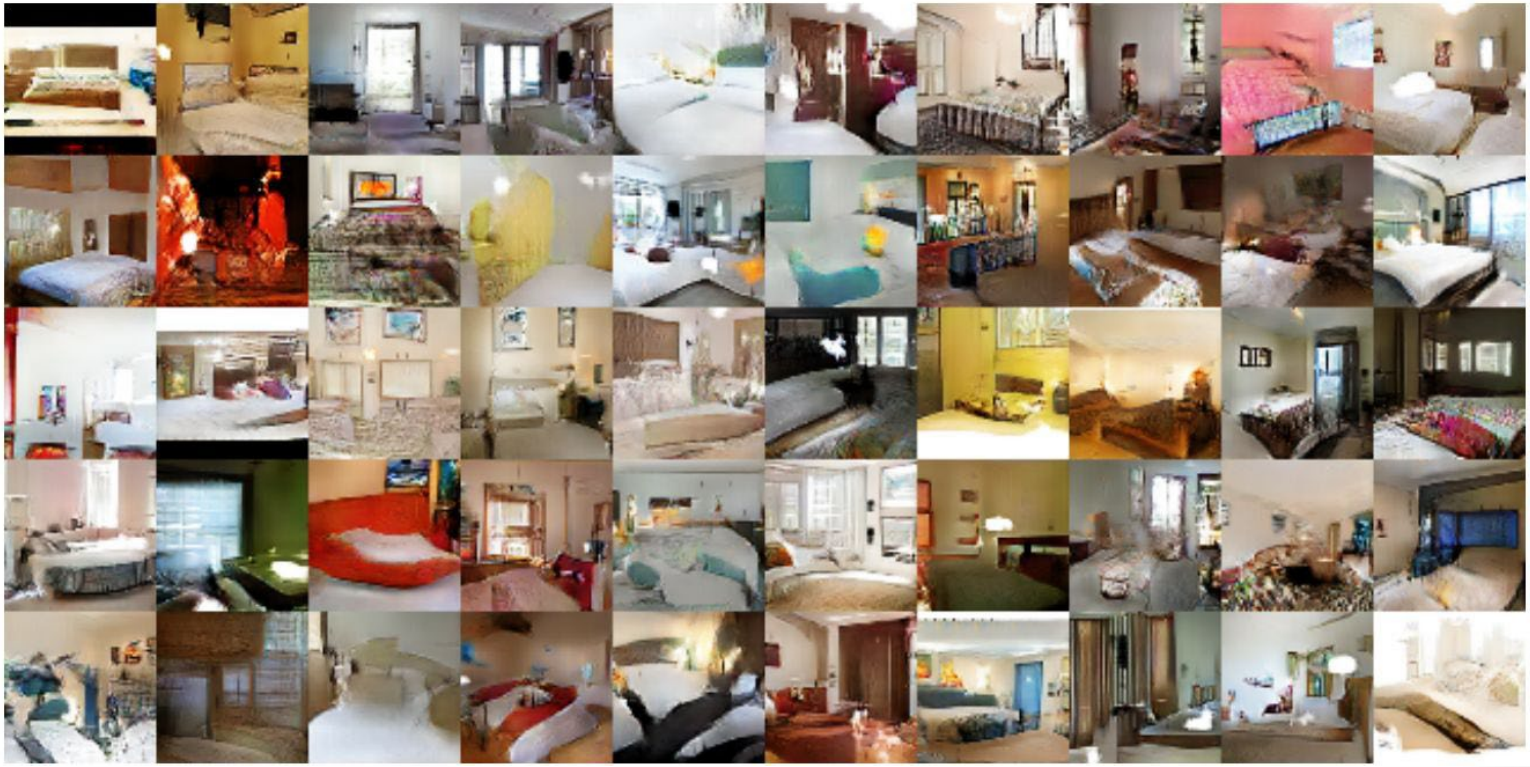
DCGAN: Generate the images with Deep Convolutional GAN



partial credit by CS311n

Generative Adversarial Nets: Convolutional Architectures

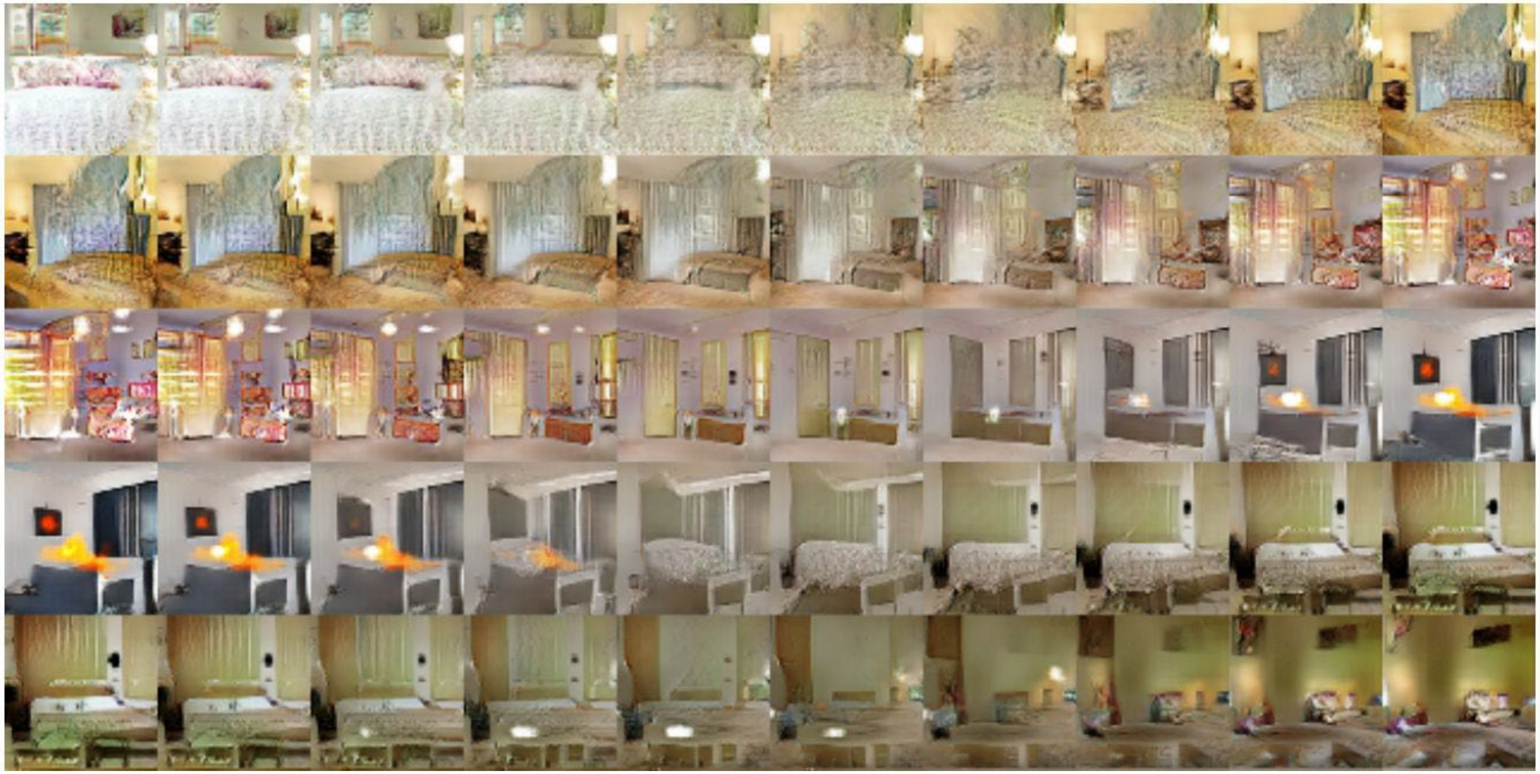
Samples from the model look much better!



Radford et al, ICLR 2016

partial credit by CS311n

Generative Adversarial Nets: Convolutional Architectures



Interpolating between random points in laten space

Radford et al, ICLR 2016

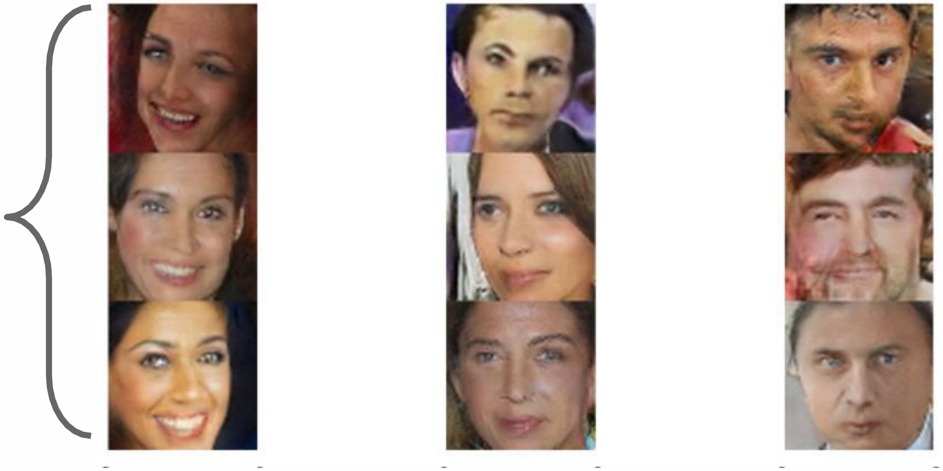
partial credit by CS311n

Generative Adversarial Nets: Interpretable Vector Math

Radford et al, ICLR 2016

Smiling woman Neutral woman Neutral man

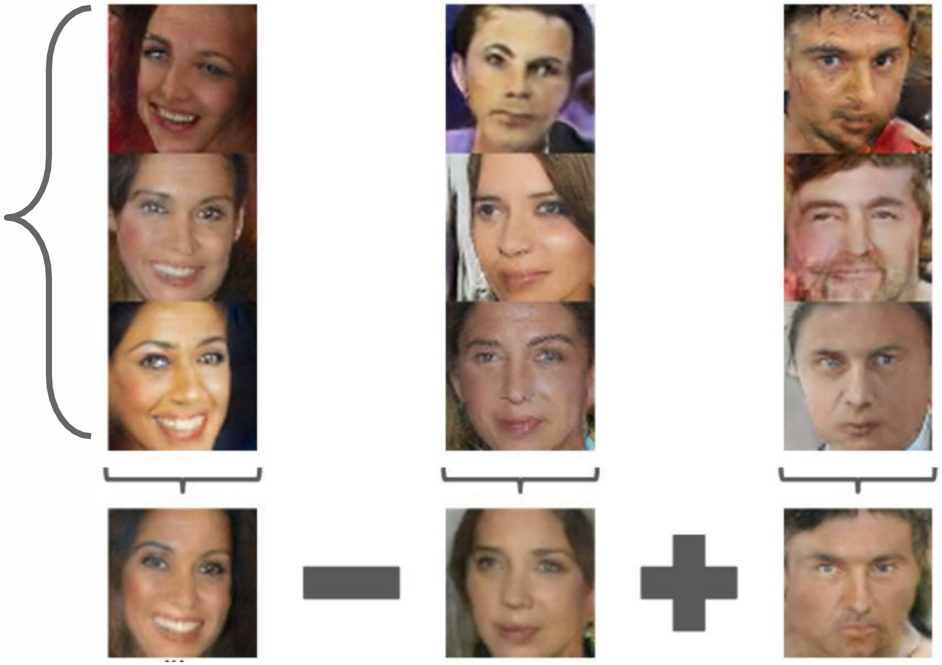
Samples from the model



partial credit by CS311n

Generative Adversarial Nets: Interpretable Vector Math

Radford et al, ICLR 2016



Smiling woman Neutral woman Neutral man

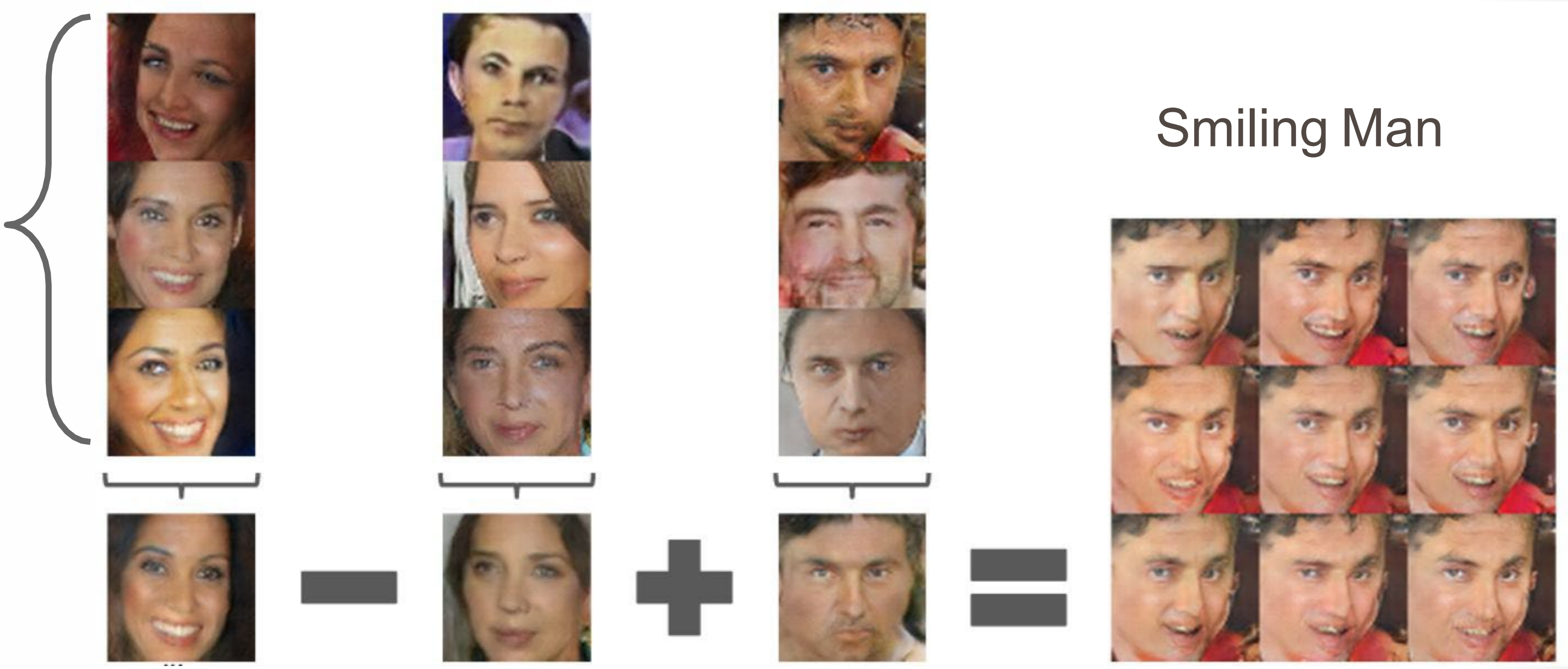
Samples from the model

Average Z vectors, do arithmetic

partial credit by CS311n

Generative Adversarial Nets: Interpretable Vector Math

Radford et al, ICLR 2016



Smiling woman Neutral woman Neutral man

Samples from the model

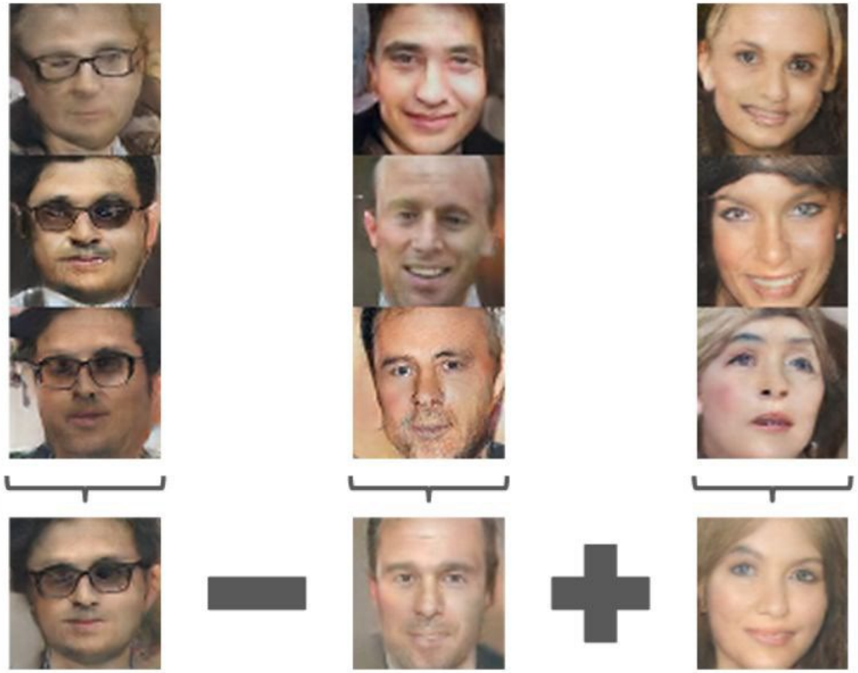
Average Z vectors, do arithmetic

Smiling Man

partial credit by CS311n

Generative Adversarial Nets: Interpretable Vector Math

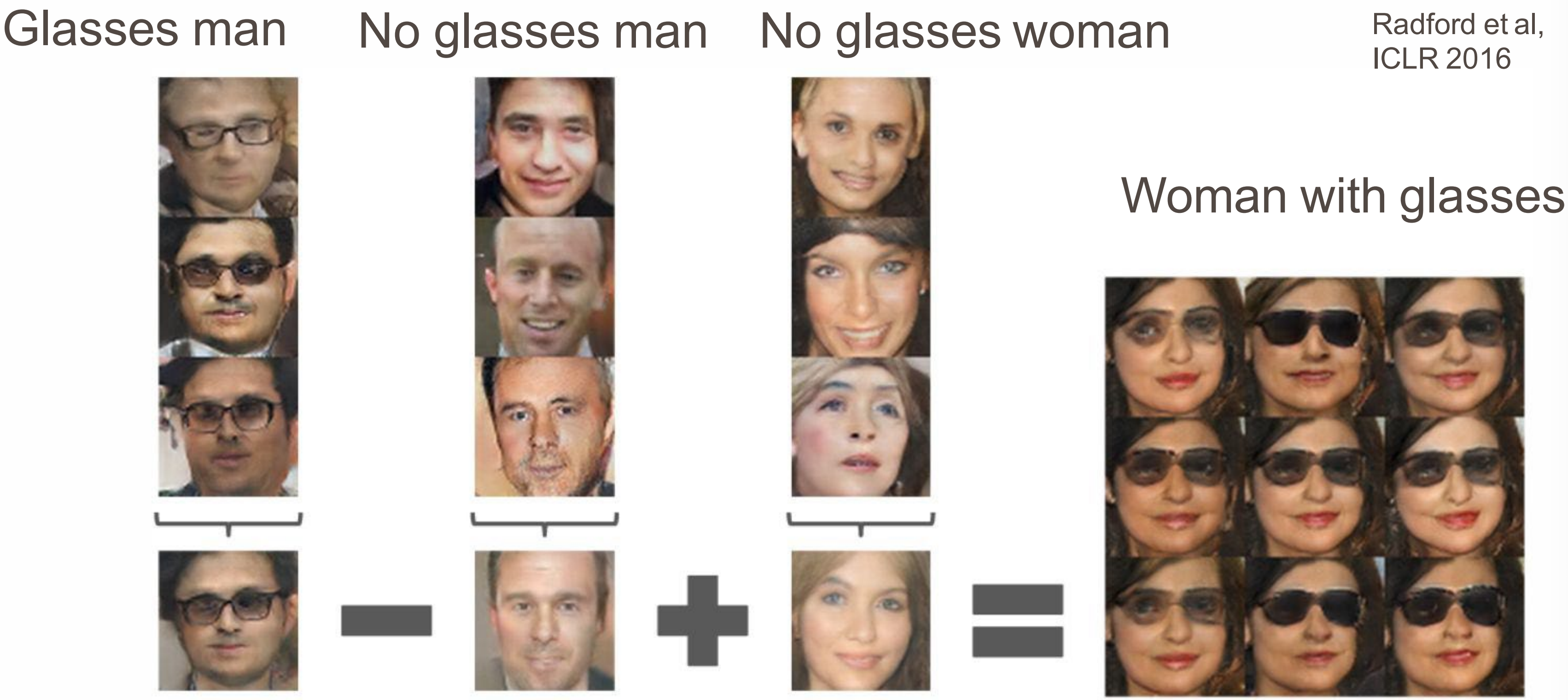
Glasses man No glasses man No glasses woman



Radford et al,
ICLR 2016

partial credit by CS311n

Generative Adversarial Nets: Interpretable Vector Math



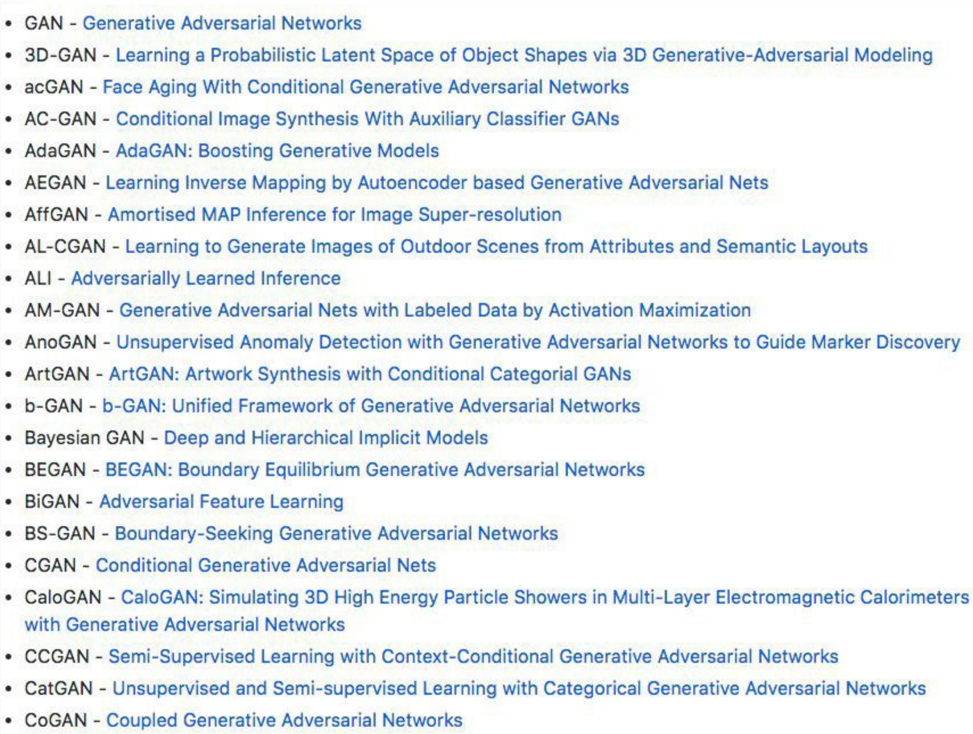
Radford et al, ICLR 2016

partial credit by CS311n

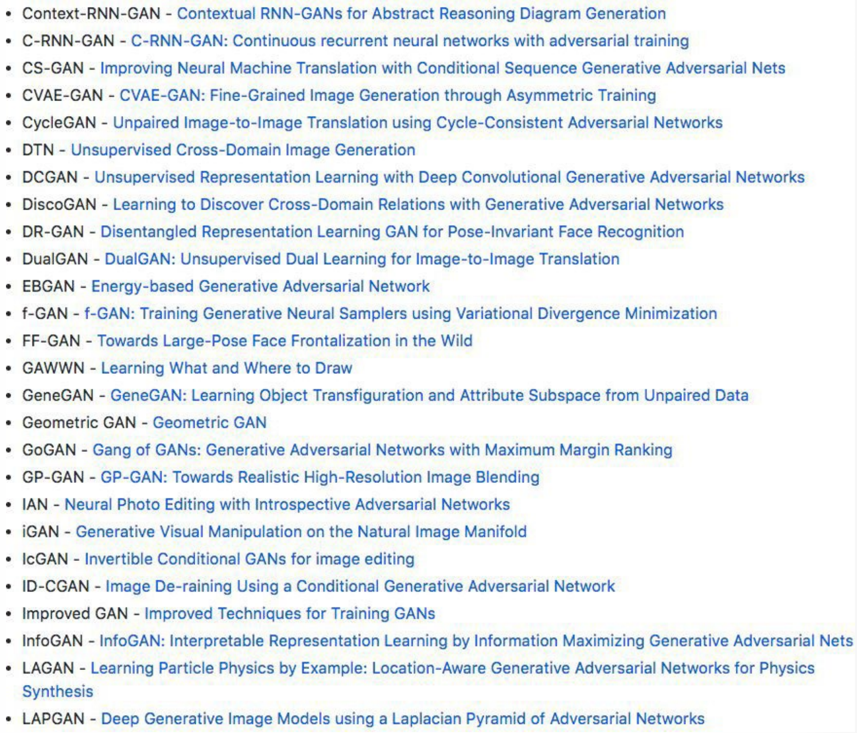
https://github.com/soumith/ganhacks

and tricks for trainings GANs

"The GAN Zoo"



- GAN - Generative Adversarial Networks
- 3D-GAN - Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling
- acGAN - Face Aging With Conditional Generative Adversarial Networks
- AC-GAN - Conditional Image Synthesis With Auxiliary Classifier GANs
- AdaGAN - AdaGAN: Boosting Generative Models
- AEGAN - Learning Inverse Mapping by Autoencoder based Generative Adversarial Nets
- AffGAN - Amortised MAP Inference for Image Super-resolution
- AL-CGAN - Learning to Generate Images of Outdoor Scenes from Attributes and Semantic Layouts
- ALI - Adversarially Learned Inference
- AM-GAN - Generative Adversarial Nets with Labeled Data by Activation Maximization
- AnoGAN - Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery
- ArtGAN - ArtGAN: Artwork Synthesis with Conditional Categorial GANs
- b-GAN - b-GAN: Unified Framework of Generative Adversarial Networks
- Bayesian GAN - Deep and Hierarchical Implicit Models
- BEGAN - BEGAN: Boundary Equilibrium Generative Adversarial Networks
- BiGAN - Adversarial Feature Learning
- BS-GAN - Boundary-Seeking Generative Adversarial Networks
- CGAN - Conditional Generative Adversarial Nets
- CaloGAN - CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks
- CCGAN - Semi-Supervised Learning with Context-Conditional Generative Adversarial Networks
- CatGAN - Unsupervised and Semi-supervised Learning with Categorical Generative Adversarial Networks
- CoGAN - Coupled Generative Adversarial Networks



- Context-RNN-GAN - Contextual RNN-GANs for Abstract Reasoning Diagram Generation
- C-RNN-GAN - C-RNN-GAN: Continuous recurrent neural networks with adversarial training
- CS-GAN - Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets
- CVAE-GAN - CVAE-GAN: Fine-Grained Image Generation through Asymmetric Training
- CycleGAN - Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks
- DTN - Unsupervised Cross-Domain Image Generation
- DCGAN - Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks
- DiscoGAN - Learning to Discover Cross-Domain Relations with Generative Adversarial Networks
- DR-GAN - Disentangled Representation Learning GAN for Pose-Invariant Face Recognition
- DualGAN - DualGAN: Unsupervised Dual Learning for Image-to-Image Translation
- EBGAN - Energy-based Generative Adversarial Network
- f-GAN - f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization
- FF-GAN - Towards Large-Pose Face Frontalization in the Wild
- GAWWN - Learning What and Where to Draw
- GeneGAN - GeneGAN: Learning Object Transfiguration and Attribute Subspace from Unpaired Data
- Geometric GAN - Geometric GAN
- GoGAN - Gang of GANs: Generative Adversarial Networks with Maximum Margin Ranking
- GP-GAN - GP-GAN: Towards Realistic High-Resolution Image Blending
- IAN - Neural Photo Editing with Introspective Adversarial Networks
- iGAN - Generative Visual Manipulation on the Natural Image Manifold
- IcGAN - Invertible Conditional GANs for image editing
- ID-CGAN - Image De-raining Using a Conditional Generative Adversarial Network
- Improved GAN - Improved Techniques for Training GANs
- InfoGAN - InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets
- LAGAN - Learning Particle Physics by Example: Location-Aware Generative Adversarial Networks for Physics Synthesis
- LAPGAN - Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks

https://github.com/hindupuravinash/the-gan-zoo

partial credit by CS311n

https://github.com/soumith/ganhacks

- 3D-ED-GAN - Shape Inpainting using 3D Generative Adversarial Network and Recurrent Convolutional Networks
- 3D-GAN - Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling (github)
- 3D-IWGAN - Improved Adversarial Systems for 3D Object Generation and Reconstruction (github)
- 3D-PhysNet - 3D-PhysNet: Learning the Intuitive Physics of Non-Rigid Object Deformations
- 3D-RecGAN - 3D Object Reconstruction from a Single Depth View with Adversarial Learning (github)
- ABC-GAN - ABC-GAN: Adaptive Blur and Control for improved training stability of Generative Adversarial Networks (github)
- ABC-GAN - GANs for LIFE: Generative Adversarial Networks for Likelihood Free Inference
- AC-GAN - Conditional Image Synthesis With Auxiliary Classifier GANs
- acGAN - Face Aging With Conditional Generative Adversarial Networks
- ACGAN - Coverless Information Hiding Based on Generative adversarial networks
- acGAN - On-line Adaptative Curriculum Learning for GANs
- ACtuAL - ACtuAL: Actor-Critic Under Adversarial Learning
- AdaGAN - AdaGAN: Boosting Generative Models
- Adaptive GAN - Customizing an Adversarial Example Generator with Class-Conditional GANs
- AdvEntuRe - AdvEntuRe: Adversarial Training for Textual Entailment with Knowledge-Guided Examples
- AdvGAN - Generating adversarial examples with adversarial networks
- AE-GAN - AE-GAN: adversarial eliminating with GAN
- AE-OT - Latent Space Optimal Transport for Generative Models
- AEGAN - Learning Inverse Mapping by Autoencoder based Generative Adversarial Nets
- AF-DCGAN - AF-DCGAN: Amplitude Feature Deep Convolutional GAN for Fingerprint Construction in Indoor Localization System
- AffGAN - Amortised MAP Inference for Image Super-resolution
- AIM - Generating Informative and Diverse Conversational Responses via Adversarial Information Maximization
- AL-CGAN - Learning to Generate Images of Outdoor Scenes from Attributes and Semantic Layouts
- ALI - Adversarially Learned Inference (github)
- AlignGAN - AlignGAN: Learning to Align Cross-Domain Images with Conditional Generative Adversarial Networks
- AlphaGAN - AlphaGAN: Generative adversarial networks for natural image matting
- AM-GAN - Activation Maximization Generative Adversarial Nets
- AmbientGAN - AmbientGAN: Generative models from lossy measurements (github)
- AMC-GAN - Video Prediction with Appearance and Motion Conditions
- AnoGAN - Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery
- APD - Adversarial Distillation of Bayesian Neural Network Posteriors
- APE-GAN - APE-GAN: Adversarial Perturbation Elimination with GAN
- ARAE - Adversarially Regularized Autoencoders for Generating Discrete Structures (github)
- ARDA - Adversarial Representation Learning for Domain Adaptation
- ARIGAN - ARIGAN: Synthetic Arabidopsis Plants using Generative Adversarial Network
- ArtGAN - ArtGAN: Artwork Synthesis with Conditional Categorial GANs
- ASDL-GAN - Automatic Steganographic Distortion Learning Using a Generative Adversarial Network
- ATA-GAN - Attention-Aware Generative Adversarial Networks (ATA-GANs)
- Attention-GAN - Attention-GAN for Object Transfiguration in Wild Images
- AttGAN - Arbitrary Facial Attribute Editing: Only Change What You Want (github)
- AttnGAN - AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks (github)
- AVID - AVID: Adversarial Visual Irregularity Detection
- B-DCGAN - B-DCGAN:Evaluation of Binarized DCGAN for FPGA
- b-GAN - Generative Adversarial Nets from a Density Ratio Estimation Perspective
- BAGAN - BAGAN: Data Augmentation with Balancing GAN
- Bayesian GAN - Deep and Hierarchical Implicit Models
- Bayesian GAN - Bayesian GAN (github)
- BCGAN - Bayesian Conditional Generative Adverserial Networks
- BCGAN - Bidirectional Conditional Generative Adversarial networks
- BEAM - Boltzmann Encoded Adversarial Machines
- BEGAN - BEGAN: Boundary Equilibrium Generative Adversarial Networks
- BEGAN-CS - Escaping from Collapsing Modes in a Constrained Space
- Bellman GAN - Distributional Multivariate Policy Evaluation and Exploration with the Bellman GAN
- BGAN - Binary Generative Adversarial Networks for Image Retrieval (github)
- Bi-GAN - Autonomously and Simultaneously Refining Deep Neural Network Parameters by a Bi-Generative Adversarial Network Aided Genetic Algorithm
- BicycleGAN - Toward Multimodal Image-to-Image Translation (github)
- BiGAN - Adversarial Feature Learning
- BinGAN - BinGAN: Learning Compact Binary Descriptors with a Regularized GAN
- BourGAN - BourGAN: Generative Networks with Metric Embeddings
- BranchGAN - Branched Generative Adversarial Networks for Multi-Scale Image Manifold Learning
- BRE - Improving GAN Training via Binarized Representation Entropy (BRE) Regularization (github)
- BridgeGAN - Generative Adversarial Frontal View to Bird View Synthesis
- BS-GAN - Boundary-Seeking Generative Adversarial Networks
- BubGAN - BubGAN: Bubble Generative Adversarial Networks for Synthesizing Realistic Bubbly Flow Images
- BWGAN - Banach Wasserstein GAN
- C-GAN - Face Aging with Contextual Generative Adversarial Nets
- C-RNN-GAN - C-RNN-GAN: Continuous recurrent neural networks with adversarial training (github)
- CA-GAN - Composition-aided Sketch-realistic Portrait Generation
- CaloGAN - CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks (github)
- CAN - CAN: Creative Adversarial Networks, Generating Art by Learning About Styles and Deviating from Style Norms
- CapsGAN - CapsGAN: Using Dynamic Routing for Generative Adversarial Networks
- CapsuleGAN - CapsuleGAN: Generative Adversarial Capsule Network
- CatGAN - Unsupervised and Semi-supervised Learning with Categorical Generative Adversarial Networks
- CatGAN - CatGAN: Coupled Adversarial Transfer for Domain Generation
- CausalGAN - CausalGAN: Learning Causal Implicit Generative Models with Adversarial Training
- CC-GAN - Semi-Supervised Learning with Context-Conditional Generative Adversarial Networks (github)
- cd-GAN - Conditional Image-to-Image Translation
- CDcGAN - Simultaneously Color-Depth Super-Resolution with Conditional Generative Adversarial Network
- CE-GAN - Deep Learning for Imbalance Data Classification using Class Expert Generative Adversarial Network
- CFG-GAN - Composite Functional Gradient Learning of Generative Adversarial Models
- CGAN - Conditional Generative Adversarial Nets
- CGAN - Controllable Generative Adversarial Network
- Chekhov GAN - An Online Learning Approach to Generative Adversarial Networks
- ciGAN - Conditional Infilling GANs for Data Augmentation in Mammogram Classification
- CinCGAN - Unsupervised Image Super-Resolution using Cycle-in-Cycle Generative Adversarial Networks
- CipherGAN - Unsupervised Cipher Cracking Using Discrete GANs
- ClusterGAN - ClusterGAN : Latent Space Clustering in Generative Adversarial Networks
- CM-GAN - CM-GANs: Cross-modal Generative Adversarial Networks for Common Representation Learning
- CoAtt-GAN - Are You Talking to Me? Reasoned Visual Dialog Generation through Adversarial Learning
- CoGAN - Coupled Generative Adversarial Networks
- ComboGAN - ComboGAN: Unrestrained Scalability for Image Domain Translation (github)
- ConceptGAN - Learning Compositional Visual Concepts with Mutual Consistency
- Conditional cycleGAN - Conditional CycleGAN for Attribute Guided Face Image Generation
- constrast-GAN - Generative Semantic Manipulation with Contrasting GAN
- Context-RNN-GAN - Contextual RNN-GANs for Abstract Reasoning Diagram Generation
- CorrGAN - Correlated discrete data generation using adversarial training
- Coulomb GAN - Coulomb GANs: Provably Optimal Nash Equilibria via Potential Fields
- Cover-GAN - Generative Steganography with Kerckhoffs' Principle based on Generative Adversarial Networks
- cowboy - Defending Against Adversarial Attacks by Leveraging an Entire GAN
- CR-GAN - CR-GAN: Learning Complete Representations for Multi-view Generation
- Cramèr GAN - The Cramer Distance as a Solution to Biased Wasserstein Gradients
- Cross-GAN - Crossing Generative Adversarial Networks for Cross-View Person Re-identification
- crVAE-GAN - Channel-Recurrent Variational Autoencoders
- CS-GAN - Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets
- CSG - Speech-Driven Expressive Talking Lips with Conditional Sequential Generative Adversarial Networks
- CT-GAN - CT-GAN: Conditional Transformation Generative Adversarial Network for Image Attribute Modification
- CVAE-GAN - CVAE-GAN: Fine-Grained Image Generation through Asymmetric Training
- CycleGAN - Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks (github)
- D-GAN - Differential Generative Adversarial Networks: Synthesizing Non-linear Facial Variations with Limited Number of Training Data
- D-WCGAN - I-vector Transformation Using Conditional Generative Adversarial Networks for Short Utterance Speaker Verification
- D2GAN - Dual Discriminator Generative Adversarial Nets
- D2IA-GAN - Tagging like Humans: Diverse and Distinct Image Annotation
- DA-GAN - DA-GAN: Instance-level Image Translation by Deep Attention Generative Adversarial Networks (with Supplementary Materials)
- DADA - DADA: Deep Adversarial Data Augmentation for Extremely Low Data Regime Classification
- DAGAN - Data Augmentation Generative Adversarial Networks
- DAN - Distributional Adversarial Networks
- DBLRGAN - Adversarial Spatio-Temporal Learning for Video Deblurring
- DCGAN - Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks (github)
- DE-GAN - Generative Adversarial Networks with Decoder-Encoder Output Noise
- DeblurGAN - DeblurGAN: Blind Motion Deblurring Using Conditional Adversarial Networks (github)
- DeepFD - Learning to Detect Fake Face Images in the Wild
- Defense-GAN - Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models (github)
- Defo-Net - Defo-Net: Learning Body Deformation using Generative Adversarial Networks
- DeliGAN - DeLiGAN : Generative Adversarial Networks for Diverse and Limited Data (github)
- DF-GAN - Learning Disentangling and Fusing Networks for Face Completion Under Structured Occlusions
- DialogWAE - DialogWAE: Multimodal Response Generation with Conditional Wasserstein Auto-Encoder
- DiscoGAN - Learning to Discover Cross-Domain Relations with Generative Adversarial Networks
- DistanceGAN - One-Sided Unsupervised Domain Mapping
- DM-GAN - Dual Motion GAN for Future-Flow Embedded Video Prediction
- DMGAN - Disconnected Manifold Learning for Generative Adversarial Networks
- DNA-GAN - DNA-GAN: Learning Disentangled Representations from Multi-Attribute Images
- DOPING - DOPING: Generative Data Augmentation for Unsupervised Anomaly Detection with GAN
- dp-GAN - Differentially Private Releasing via Deep Generative Model
- DP-GAN - DP-GAN: Diversity-Promoting Generative Adversarial Network for Generating Informative and Diversified Text
- DPGAN - Differentially Private Generative Adversarial Network
- DR-GAN - Representation Learning by Rotating Your Faces
- DRAGAN - How to Train Your DRAGAN (github)
- Dropout-GAN - Dropout-GAN: Learning from a Dynamic Ensemble of Discriminators
- DRPAN - Discriminative Region Proposal Adversarial Networks for High-Quality Image-to-Image Translation
- DSH-GAN - Deep Semantic Hashing with Generative Adversarial Networks
- DSP-GAN - Depth Structure Preserving Scene Image Generation
- DTLC-GAN - Generative Adversarial Image Synthesis with Decision Tree Latent Controller
- DTN - Unsupervised Cross-Domain Image Generation
- DTR-GAN - DTR-GAN: Dilated Temporal Relational Adversarial Network for Video Summarization
- DualGAN - DualGAN: Unsupervised Dual Learning for Image-to-Image Translation
- Dualing GAN - Dualing GANs
- DVGAN - Human Motion Modeling using DVGANs
- Dynamics Transfer GAN - Dynamics Transfer GAN: Generating Video by Transferring Arbitrary Temporal Dynamics from a Source Video to a Single Target Image
- E-GAN - Evolutionary Generative Adversarial Networks
- EAR - Generative Model for Heterogeneous Inference
- EBGAN - Energy-based Generative Adversarial Network
- ecGAN - eCommerceGAN : A Generative Adversarial Network for E-commerce
- ED//GAN - Stabilizing Training of Generative Adversarial Networks through Regularization
- Editable GAN - Editable Generative Adversarial Networks: Generating and Editing Faces Simultaneously
- EGAN - Enhanced Experience Replay Generation for Efficient Reinforcement Learning
- EL-GAN - EL-GAN: Embedding Loss Driven Generative Adversarial Networks for Lane Detection
- ELEGANT - ELEGANT: Exchanging Latent Encodings with GAN for Transferring Multiple Face Attributes
- EnergyWGAN - Energy-relaxed Wassertein GANs (EnergyWGAN): Towards More Stable and High Resolution Image Generation
- ESRGAN - ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks
- ExGAN - Eye In-Painting with Exemplar Generative Adversarial Networks
- ExposureGAN - Exposure: A White-Box Photo Post-Processing Framework (github)
- ExprGAN - ExprGAN: Facial Expression Editing with Controllable Expression Intensity
- f-CLSWGAN - Feature Generating Networks for Zero-Shot Learning
- f-GAN - f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization
- FairGAN - FairGAN: Fairness-aware Generative Adversarial Networks
- Fairness GAN - Fairness GAN
- FakeGAN - Detecting Deceptive Reviews using Generative Adversarial Networks
- FBGAN - Feedback GAN (FBGAN) for DNA: a Novel Feedback-Loop Architecture for Optimizing Protein Functions
- FBGAN - Featurized Bidirectional GAN: Adversarial Defense via Adversarially Learned Semantic Inference
- FC-GAN - Fast-converging Conditional Generative Adversarial Networks for Image Synthesis
- FF-GAN - Towards Large-Pose Face Frontalization in the Wild
- FGGAN - Adversarial Learning for Fine-grained Image Search
- Fictitious GAN - Fictitious GAN: Training GANs with Historical Models
- FIGAN - Frame Interpolation with Multi-Scale Deep Loss Functions and Generative Adversarial Networks
- Fila-GAN - Synthesizing Filamentary Structured Images with GANs
- First Order GAN - First Order Generative Adversarial Networks (github)
- Fisher GAN - Fisher GAN
- Flow-GAN - Flow-GAN: Bridging implicit and prescribed learning in generative models
- FrankenGAN - rankenGAN: Guided Detail Synthesis for Building Mass-Models Using Style-Synchonized GANs

partial credit by CS311n

2017: Explosion of GANs

Better training and generation

LSGAN, Zhu 2017.

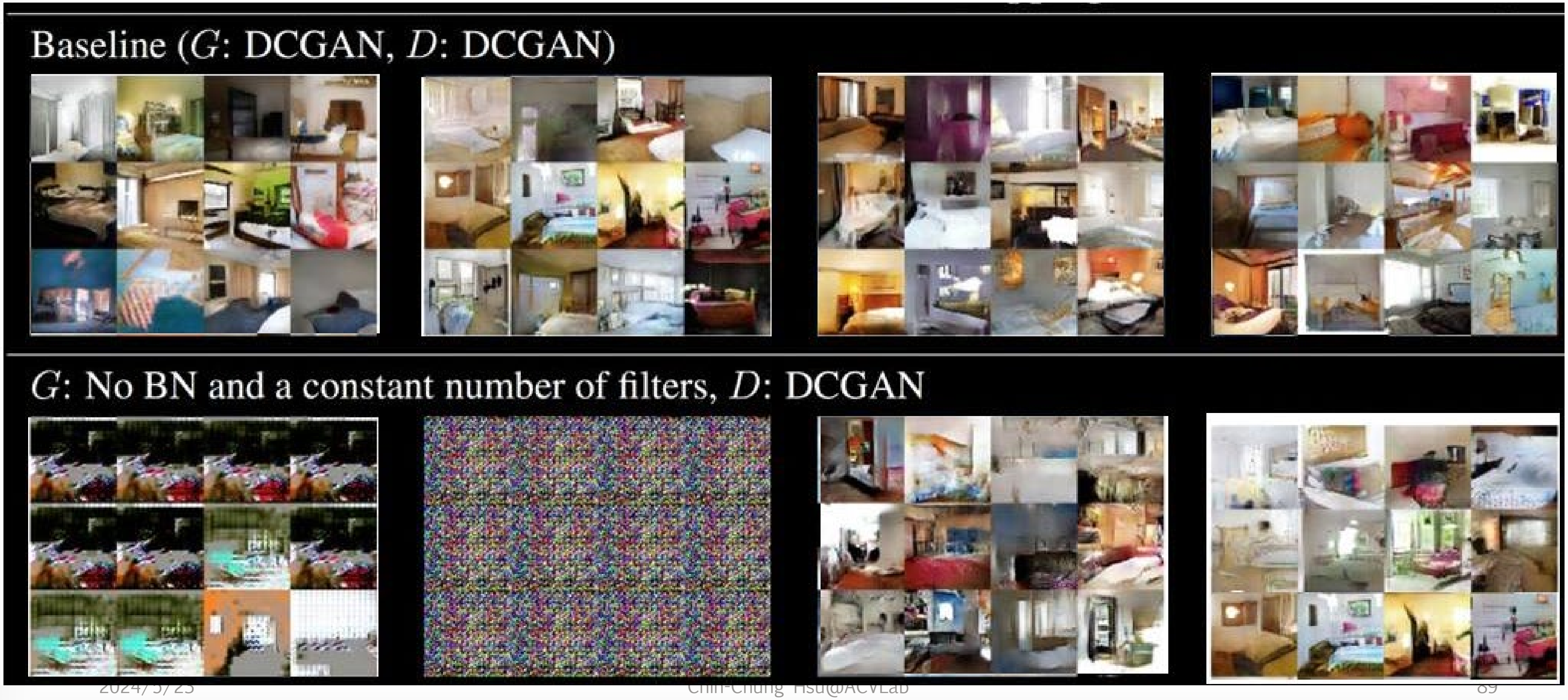
Wasserstein GAN, Arjovsky 2017. Improved Wasserstein GAN, Gulrajani 2017.

Progressive GAN, Karras 2018.

partial credit by CS311n

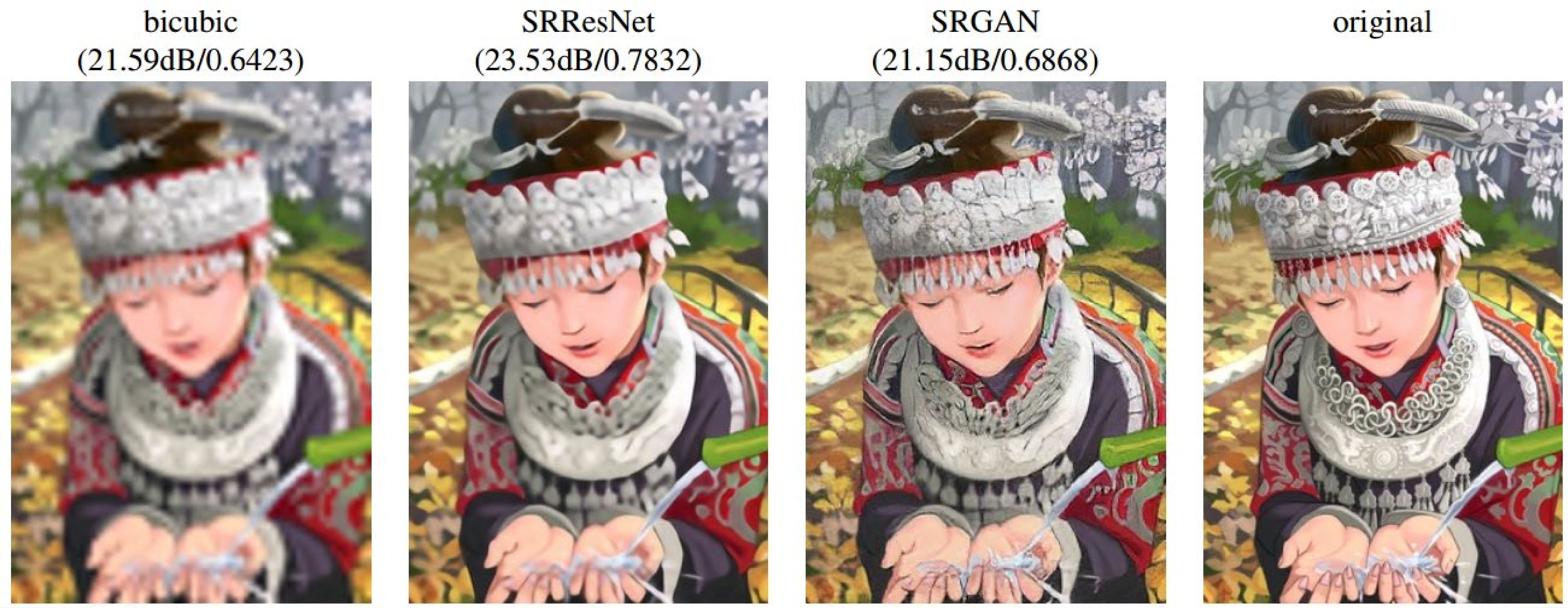
Improved Versions of GAN

- There are more than 100 improved GANs/Applications since 2014!!
  - A hot topic in deep learning


Baseline (*G*: DCGAN, *D*: DCGAN)
G: No BN and a constant number of filters, *D*: DCGAN

partial credit by CS311n

Image Super-Resolution



bicubic (21.59dB/0.6423)

SRResNet (23.53dB/0.7832)

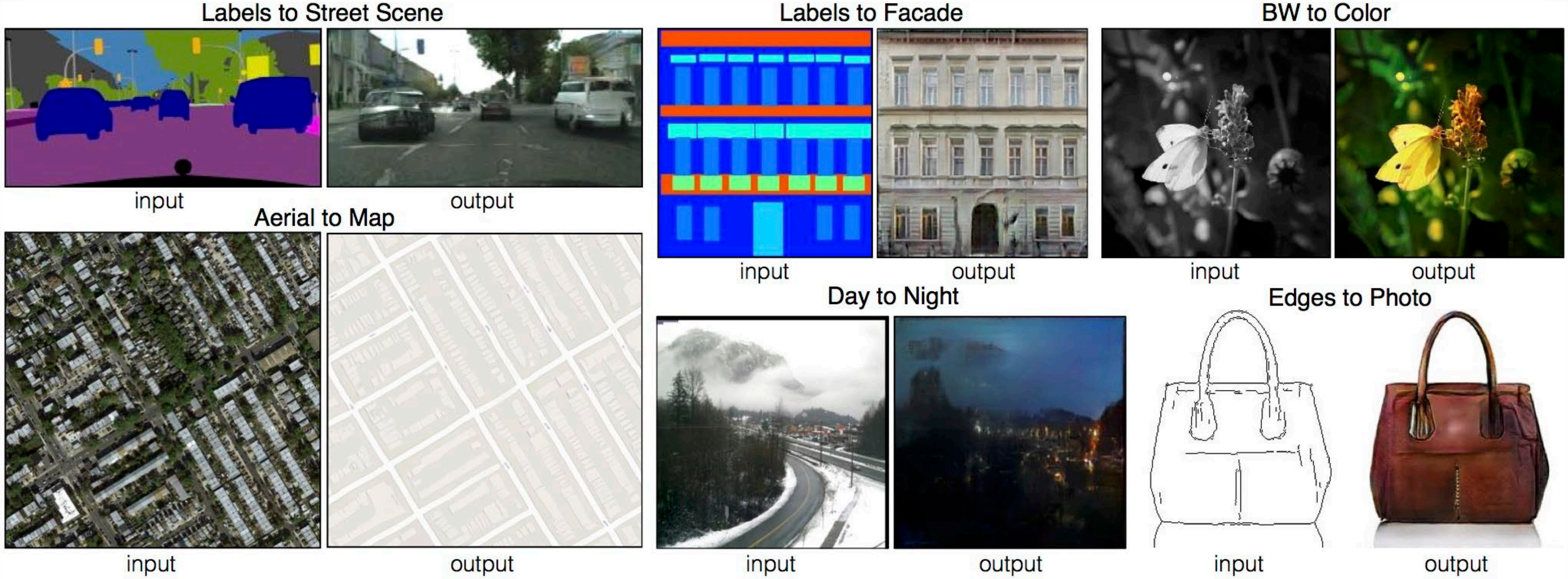
SRGAN (21.15dB/0.6868)

original

Ledig et al. CVPR 2017

partial credit by CS311n

Image-to-Image Translation

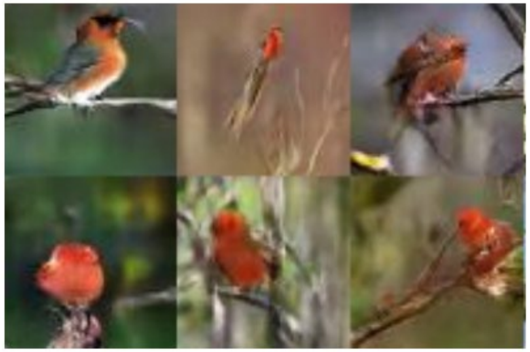


Isola et al. CVPR 2017

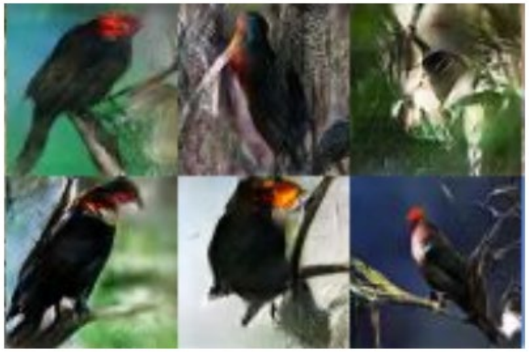
partial credit by CS311n

Text2Image

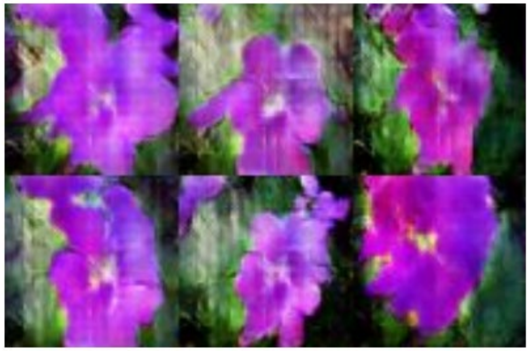
this small bird has a pink breast and crown, and black primaries and secondaries.



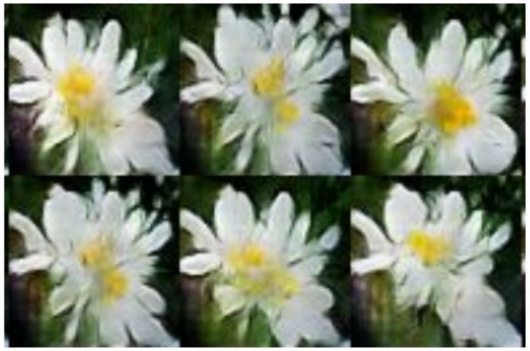
this magnificent fellow is almost all black with a red crest, and white cheek patch.



the flower has petals that are bright pinkish purple with white stigma



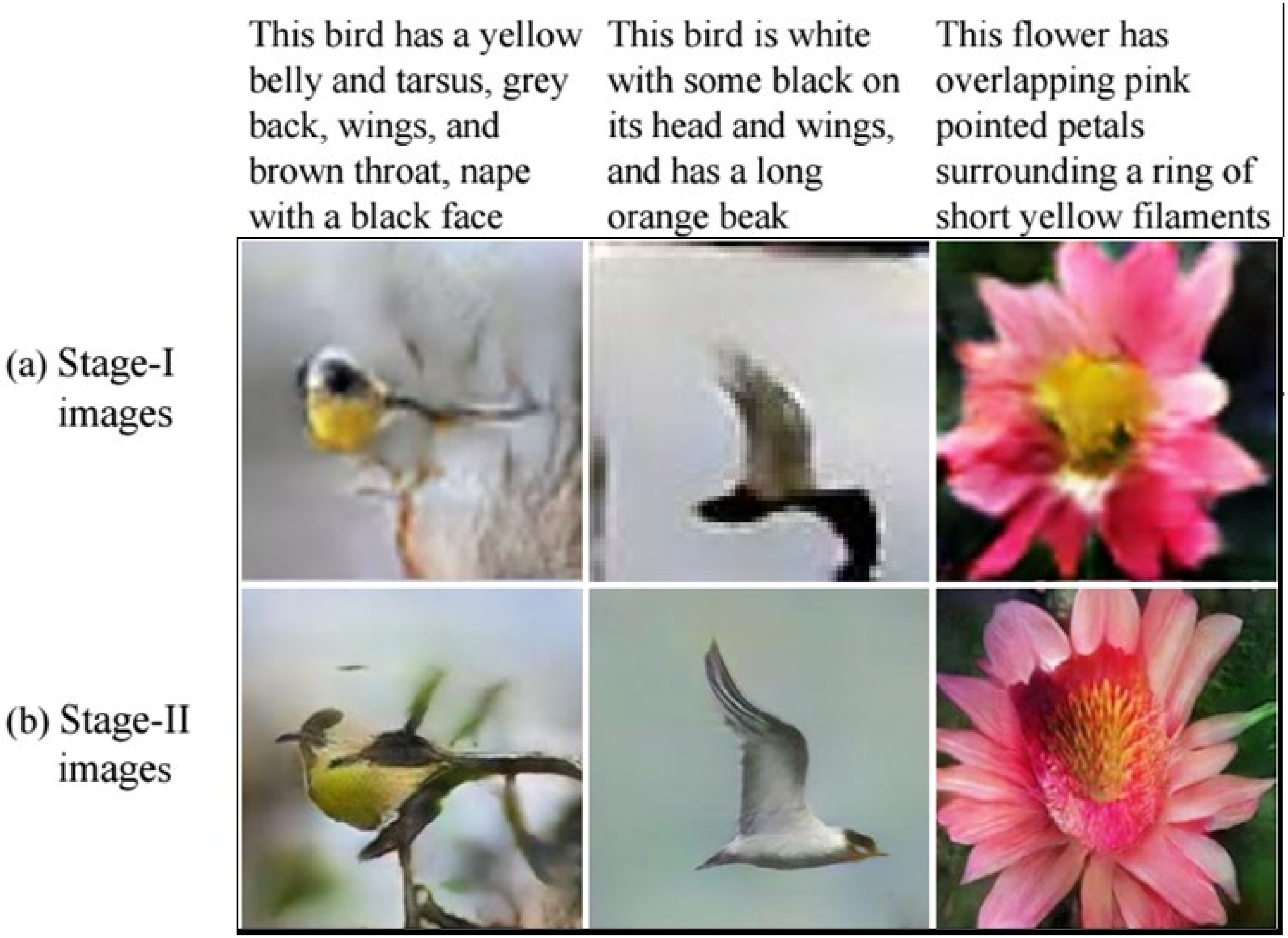
this white and yellow flower have thin white petals and a round yellow stamen



Reed et al. ICML 2016

partial credit by CS311n

StackGAN



This bird has a yellow belly and tarsus, grey back, wings, and brown throat, nape with a black face

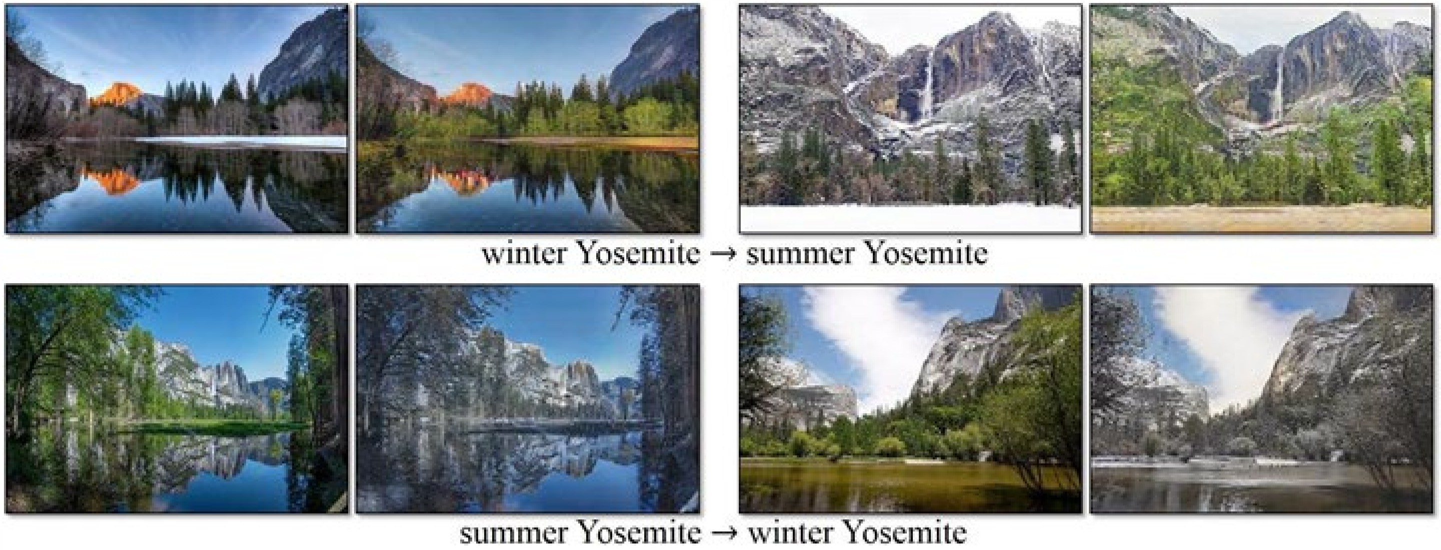
This bird is white with some black on its head and wings, and has a long orange beak

This flower has overlapping pink pointed petals surrounding a ring of short yellow filaments

Zhang et al. 2016

partial credit by CS311n

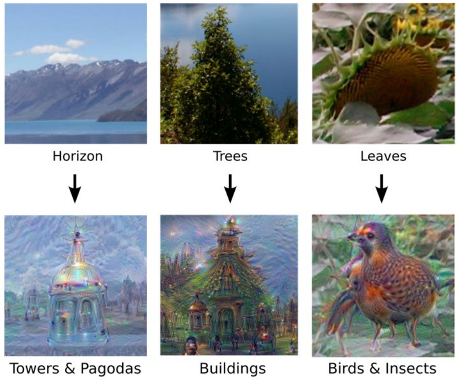
Style Transfer


winter Yosemite → summer Yosemite

summer Yosemite → winter Yosemite

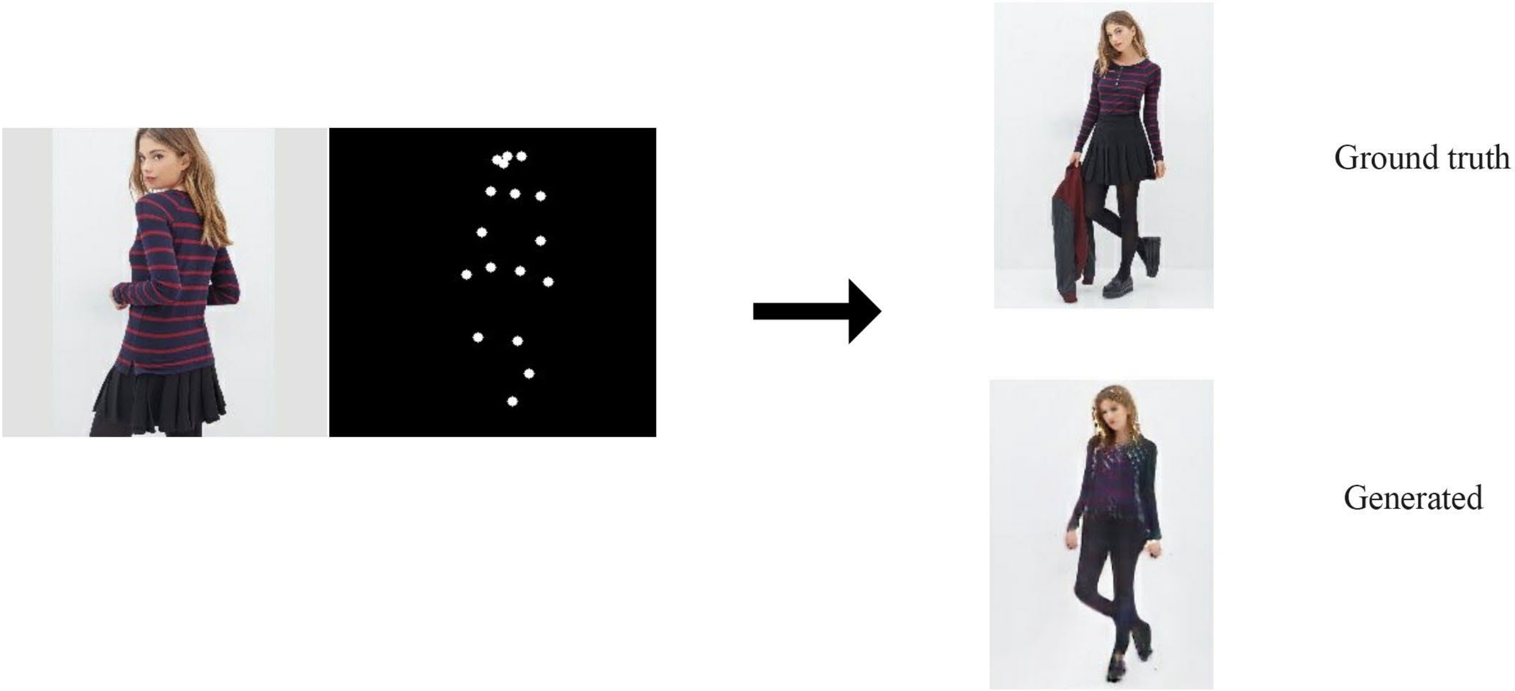
partial credit by CS311n

Style Transfer



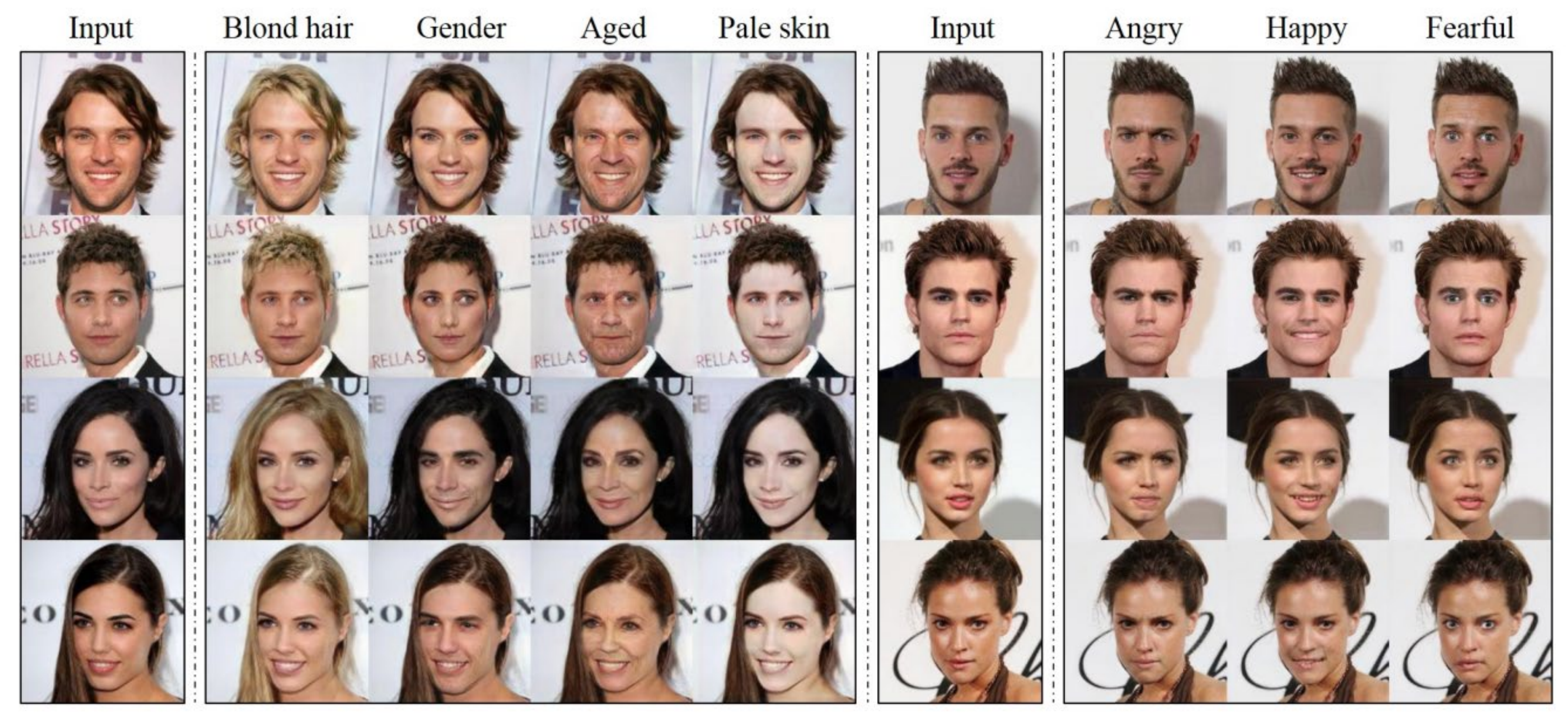
partial credit by CS311n

Pose Generation



Source: https://papers.nips.cc/paper/6644-pose-guided-person-image-generation.pdf

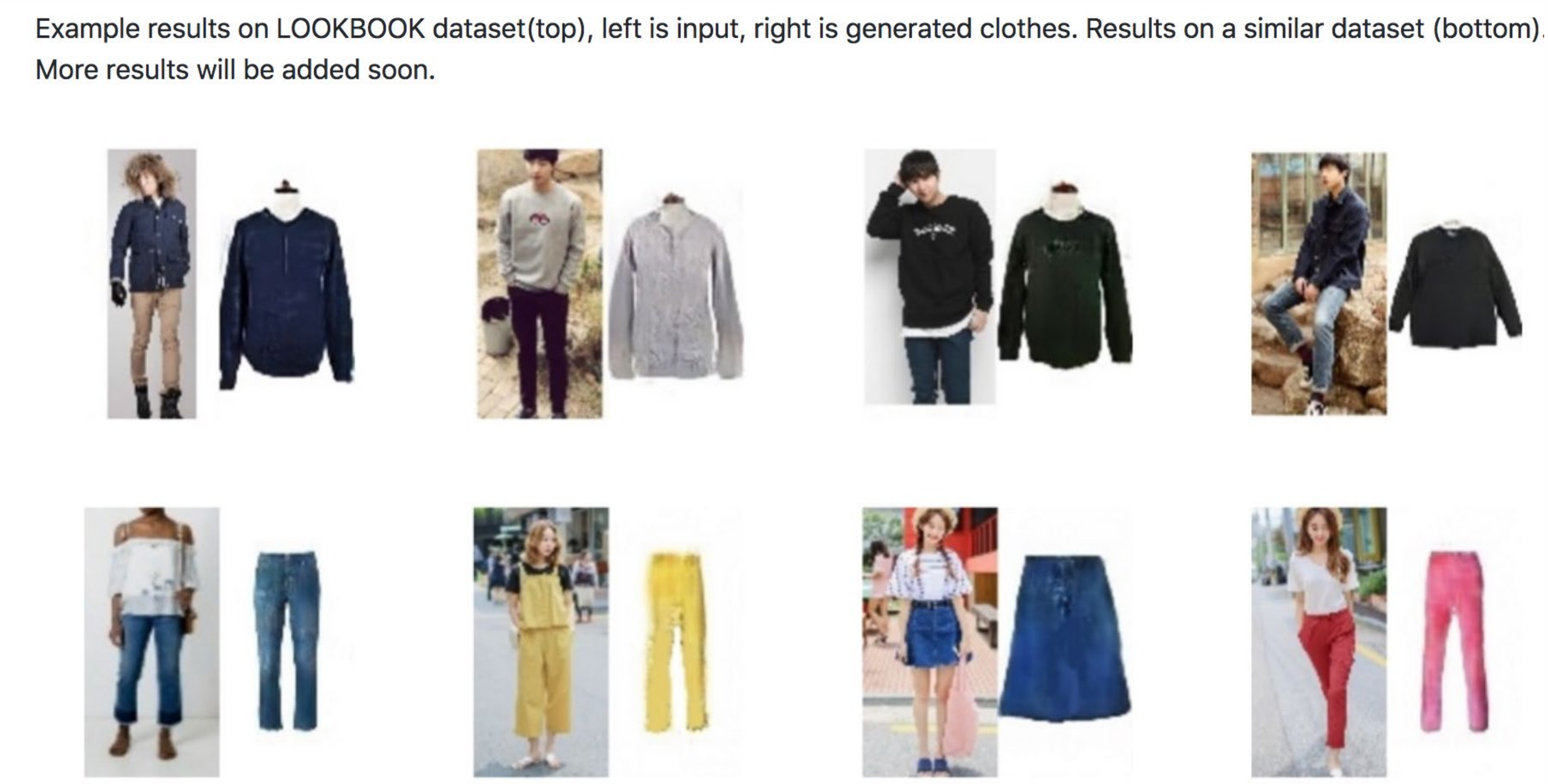
StarGAN



Source: https://arxiv.org/pdf/1711.09020.pdf

partial credit by CS311n

Change the Cloth

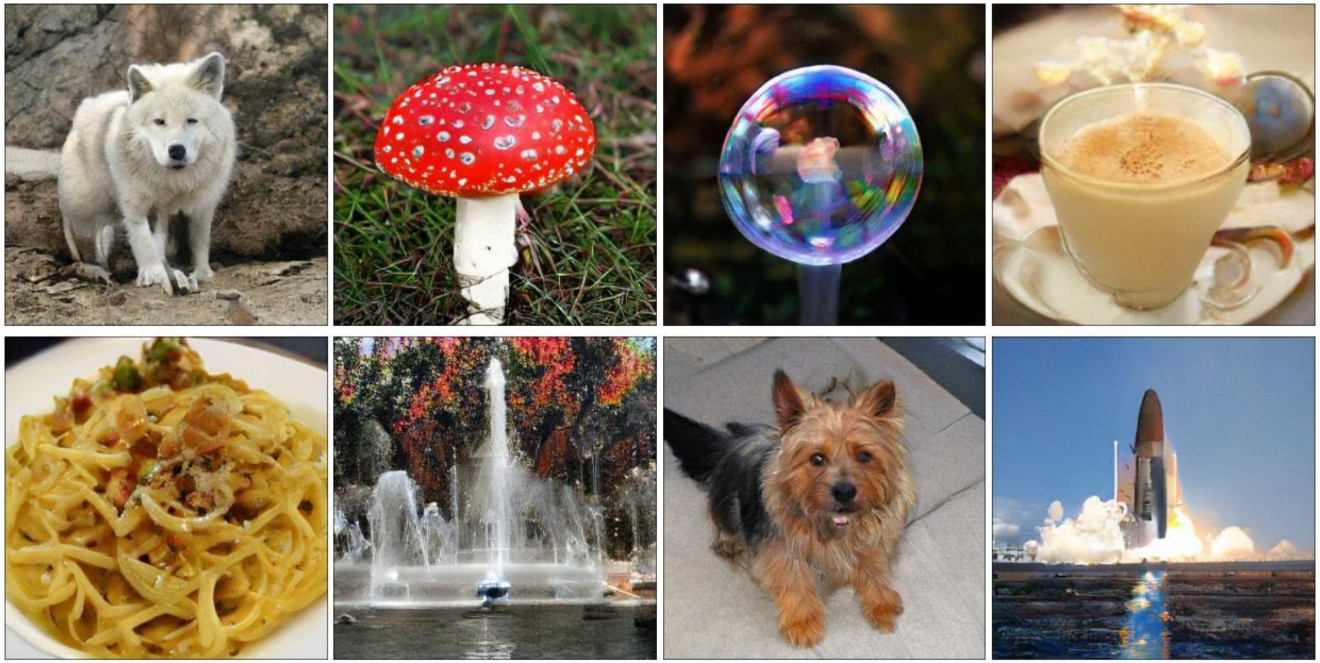


Example results on LOOKBOOK dataset(top), left is input, right is generated clothes. Results on a similar dataset (bottom). More results will be added soon.

Source: https://github.com/fxia22/PixelDTGAN

partial credit by CS311n

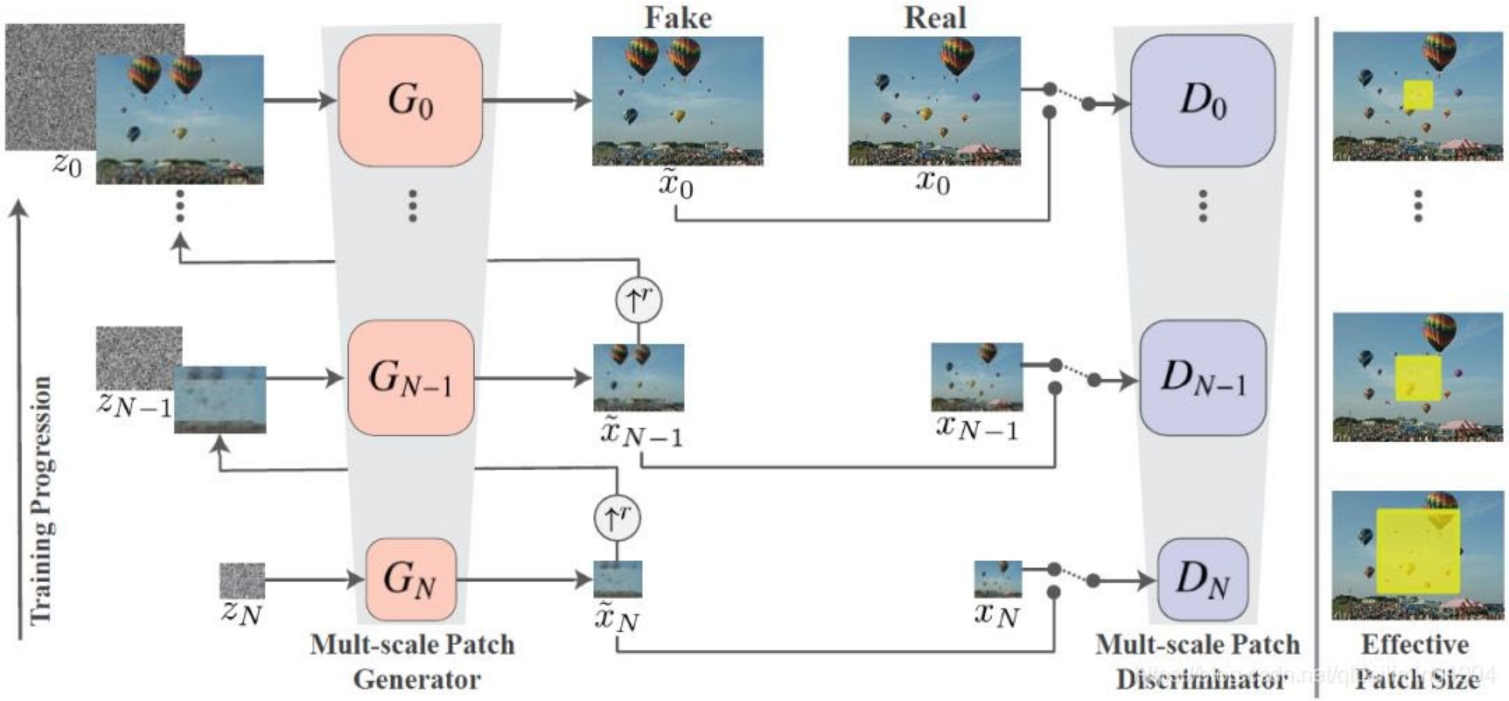
2019: BigGAN



Brock et al., 2019

partial credit by CS311n

SinGAN, ICCV 2019.



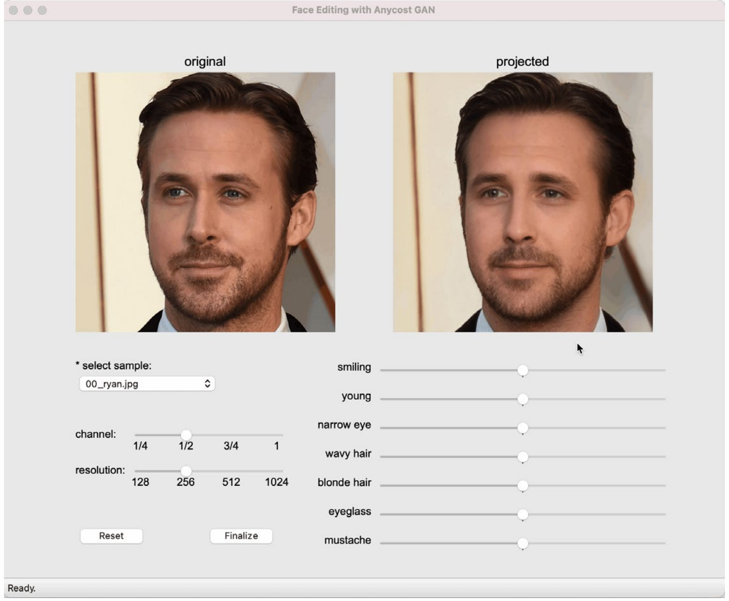
Shaham, Tamar Rott, Tali Dekel, and Tomer Michaeli. "Singan: Learning a generative model from a single natural image." *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

partial credit by CS311n

DeepFake for Faces

- StyleGANv2 [CVPR2020]

- **Anycost GAN [CVPR2021]**



* select sample: 00_ryan.jpg

Ready.

partial credit by CS311n

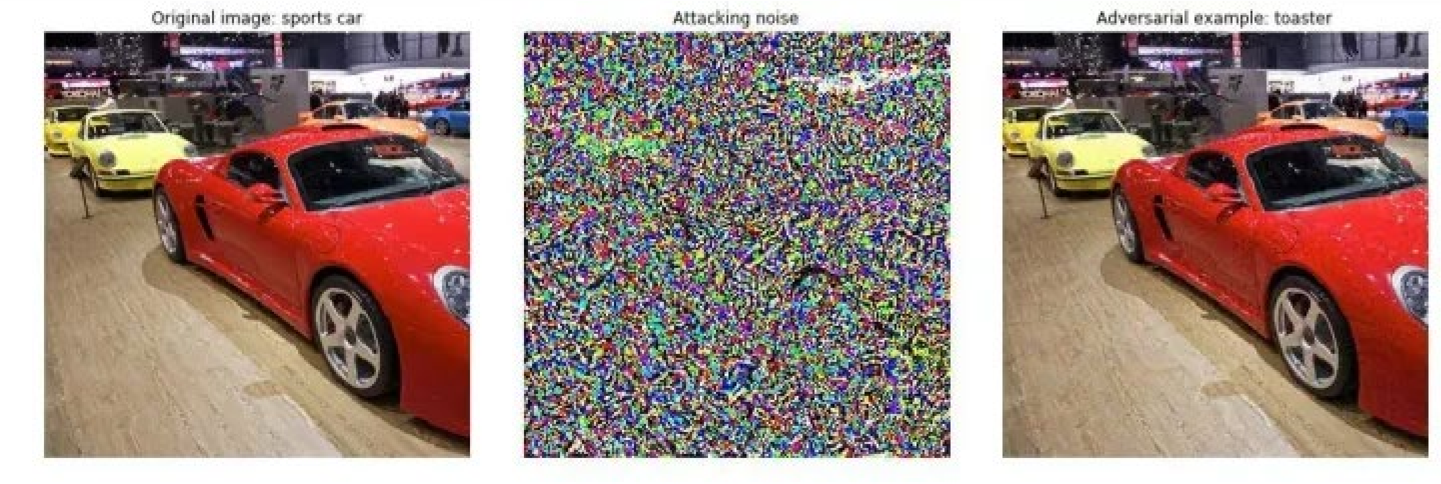
GAN-based Applications

- What application should GAN be used
  - Any task related to image synthesis (合成影像任務包含)
    - Image super-resolution
      - Discriminator can be used to judge its fidelity and resolution
    - Image translation
      - Discriminator can be used to identify its quality
    - Image segmentation
      - Discriminator can be used to tune generated segmentation map
    - Data argumentation
      - Discriminator can be used to check the fidelity of the simulated image
    - etc...

partial credit by CS311n

Rethinking GANs

- Is possible to fool a DNN by adding specified noises?
  - Adversarial attack



(a) Car (b) Noise map (c) Toaster

partial credit by CS311n

Adversarial Attack

- Cause model uninterpretable
  - Goodfellow given some reasons
    - Assume that a linear operation in NN is $\mathbf{z} = \mathbf{w}(\mathbf{x} + \mathbf{b})$
    - Consider that
      - $\mathbf{z} = \mathbf{w}(\mathbf{x} + \mathbf{b}) * \mathbf{d}$, where d is the attach signal
      - When $\mathbf{wd}$ is large
        - Attacking will be very effective
  - How to find $\mathbf{d}$?
    - Yes, we can train it by supervised way
    - Similar to “Discriminator” does

partial credit by CS311n

GANs

- Don’t work with an explicit density function
  - Take game-theoretic approach: learn to generate from training distribution through 2-player game
- Pros:
  - Beautiful, state-of-the-art samples!
- Cons:
  - Trickier / more unstable to train
  - Can’t solve inference queries such as p(x), p(z|x)
- Active areas of research:
  - Better loss functions, more stable training (Wasserstein GAN, LSGAN, many others)
  - Conditional GANs, GANs for all kinds of applications

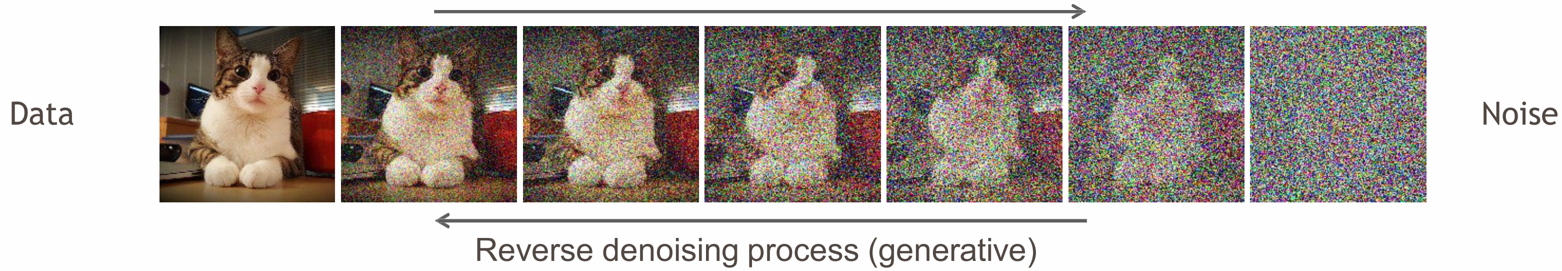
partial credit by CS311n

DENOISED DIFFUSION PROBABILISTIC MODELS

NPIS 2021

Denoising Diffusion Models

- Learning to generate by denoising
  - Denoising diffusion models consist of two processes:
  - Forward diffusion process that gradually adds noise to input
  - Reverse denoising process that learns to generate data by denoising
  - Forward diffusion process (fixed)

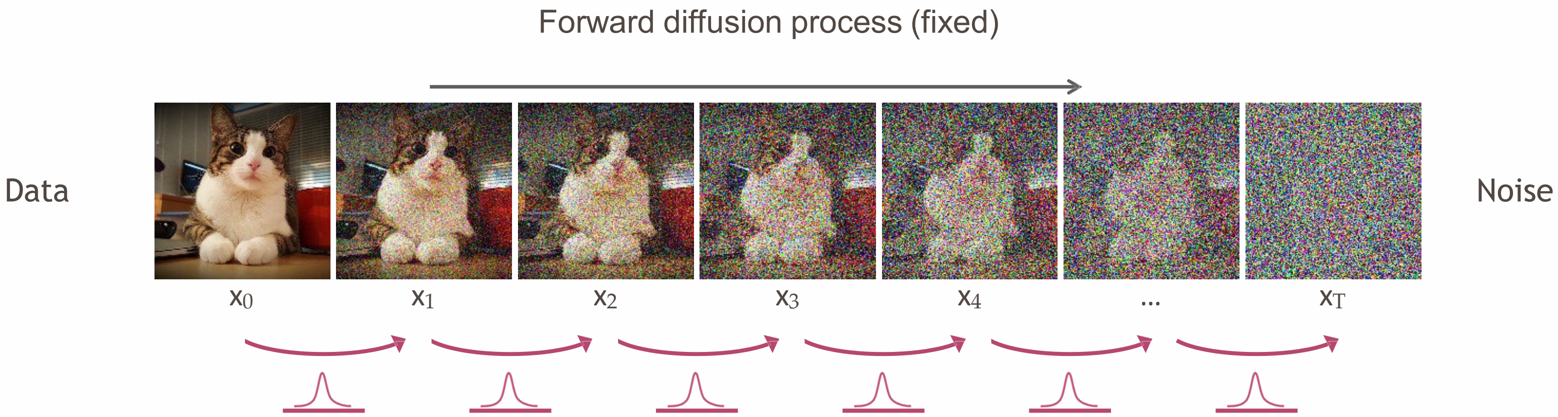


[Sohl-Dickstein et al., Deep Unsupervised Learning using Nonequilibrium Thermodynamics, ICML 2015](#) [Ho et al., Denoising Diffusion Probabilistic Models, NeurIPS 2020](#)
[Song et al., Score-Based Generative Modeling through Stochastic Differential Equations, ICLR 2021](#)

partial credit by CS311n

Forward Diffusion Process

- The formal definition of the forward process in T steps:



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

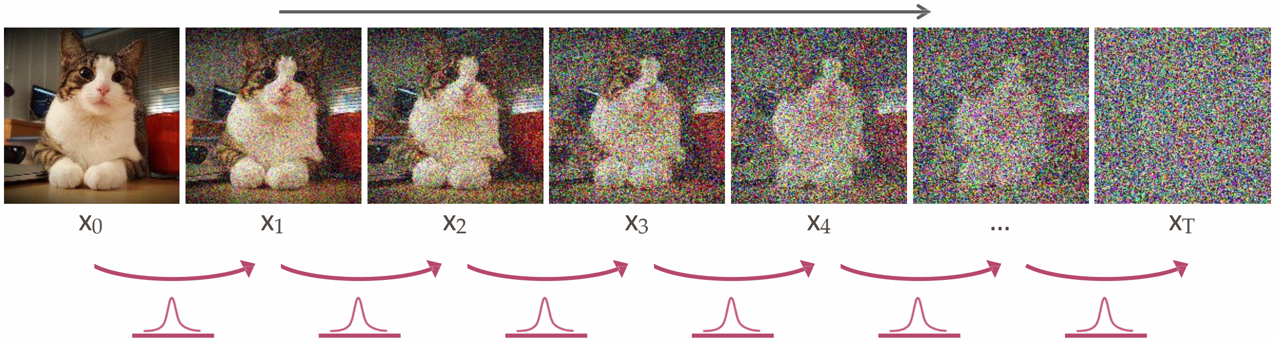
mean

variance

$$\epsilon_{t-1} \sim \mathcal{N}(0, \mathbf{I})$$

partial credit by CS311n

Diffusion Kernel



Data

Noise

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x_t}; \underset{\text{mean}}{\sqrt{1-\beta_t}\,\mathbf{x_{t-1}}}, \underset{\text{variance}}{\beta_t\mathbf{I}})$$

➡ Sample:

$$\epsilon_{t-1} \sim \mathcal{N}(0, \mathbf{I})$$

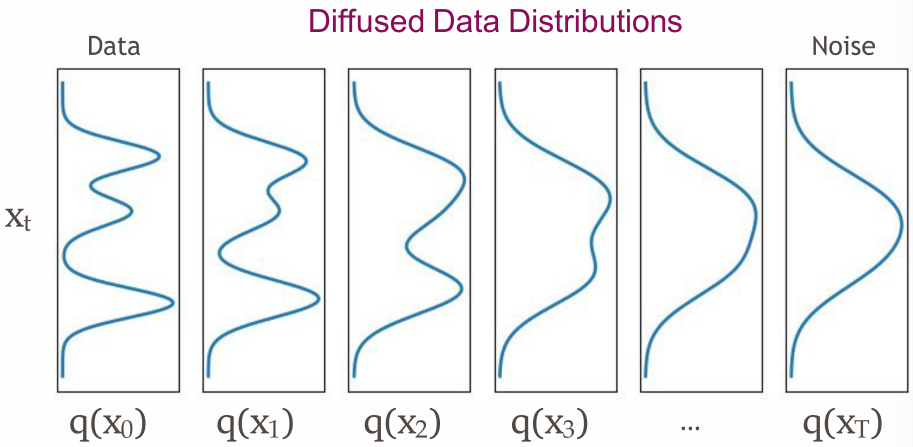
partial credit by CS311n

What happens to a distribution in the forward diffusion?

- So far, we discussed the diffusion kernel $q(\mathbf{x}_t|\mathbf{x}_0)$ but what about $q(\mathbf{x}_t)$

$$\underbrace{q(\mathbf{x}_t)}_{\text{Diffused data dist.}} = \int \underbrace{q(\mathbf{x}_0, \mathbf{x}_t)}_{\text{Joint dist.}} \, d\mathbf{x}_0 = \int \underbrace{q(\mathbf{x}_0)}_{\text{Input data dist.}} \, \underbrace{q(\mathbf{x}_t|\mathbf{x}_0)}_{\text{Diffusion kernel}} \, d\mathbf{x}_0$$

The diffusion kernel is Gaussian convolution.



- We can sample $\mathbf{x}_t \sim q(\mathbf{x}_t)$ by first sampling $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ and then sampling $\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0)$ (i.e., ancestral sampling).

partial credit by CS311n

Generative Learning by Denoising

- Recall, that the diffusion parameters are designed such that

$$q(\mathbf{x}_T) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$$

- Generation:
  - Sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$
- Iteratively sample

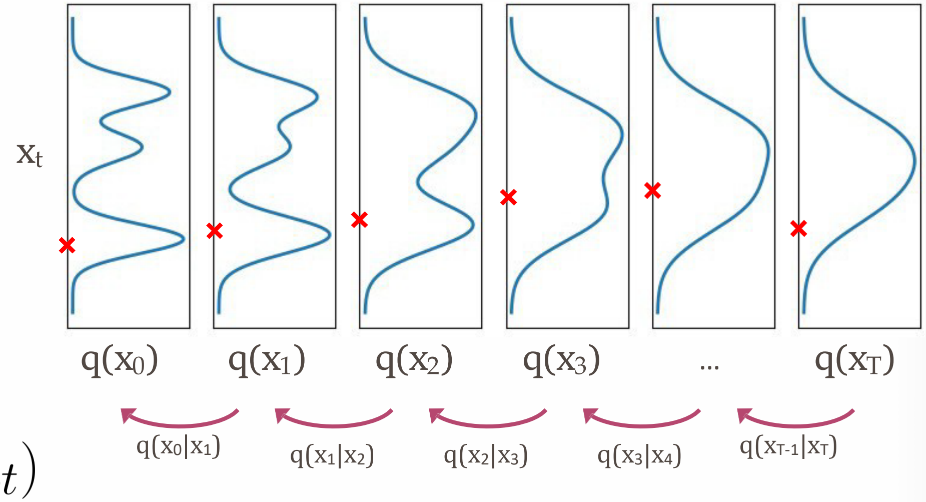
$$\mathbf{x}_{t-1} \sim \underbrace{q(\mathbf{x}_{t-1}|\mathbf{x}_t)}_{\text{True Denoising Dist.}}$$

- In general,

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) \propto q(\mathbf{x}_{t-1})q(\mathbf{x}_t|\mathbf{x}_{t-1})$$

  - is intractable.
- Can we approximate $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$
  - Yes, we can use a Normal distribution if β_t is small in each forward diffusion step.

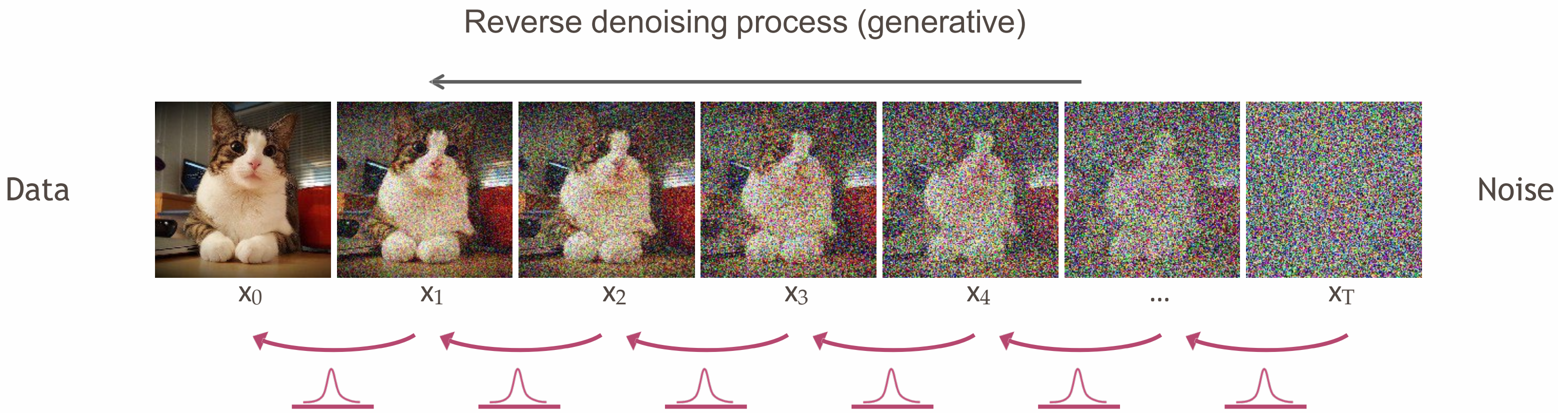
Diffused Data Distributions



partial credit by CS311n

Reverse Denoising Process

- Formal definition of forward and reverse processes in T steps:



$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \underbrace{\mu_\theta(\mathbf{x}_t, t)}_{}, \sigma_t^2\mathbf{I})$$

Trainable network
(U-net, Denoising Autoencoder)

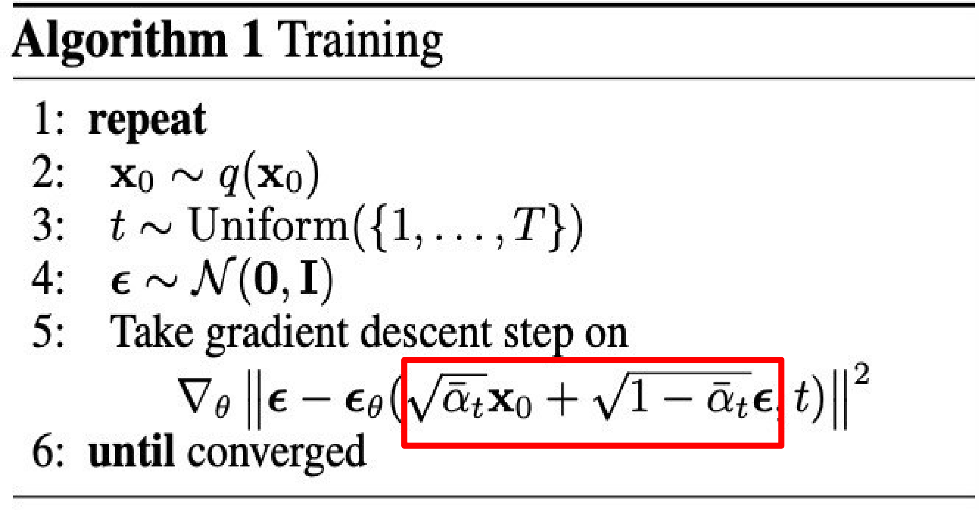
Autoencoder predicts the mean of the denoised image x(t-1) given x(t).

partial credit by CS311n

How do we train? (summary version)

What is the loss function? (Ho et al. NeurIPS 2020)

$$L_{\text{simple}} = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), \epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I}), t \sim \mathcal{U}(1,T)} \left[||\epsilon - \epsilon_\theta(\underbrace{\sqrt{\bar{\alpha}_t}\, \mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon}_{\mathbf{x}_t}, t)||^2 \right]$$



Algorithm 1 Training

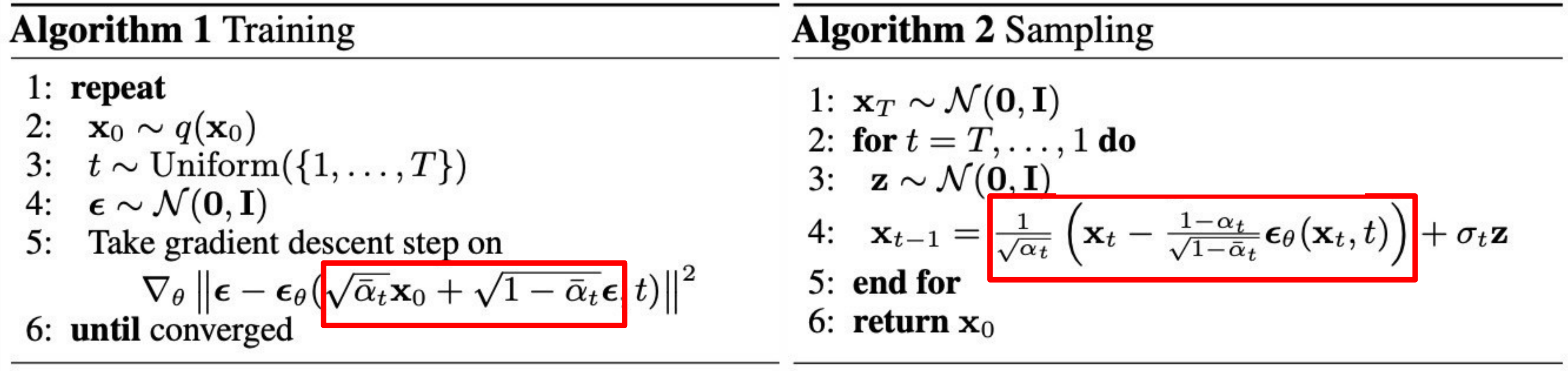
```
repeat
    $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
    $t \sim \text{Uniform}(\{1, \ldots, T\})$
    $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    Take gradient descent step on
        $\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boxed{\sqrt{\bar{\alpha}_t}\mathbf{x}_0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

U-Net autoencoder takes x(t) as input and simply predict a noise. The goal of the training is to generate a noise pattern that is unit normal. Very similar to VAE, right?

partial credit by CS311n

Summary

- Training and Sample Generation



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
        $\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boxed{\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}}, t) \right\|^2$
until converged
```

Algorithm 2 Sampling

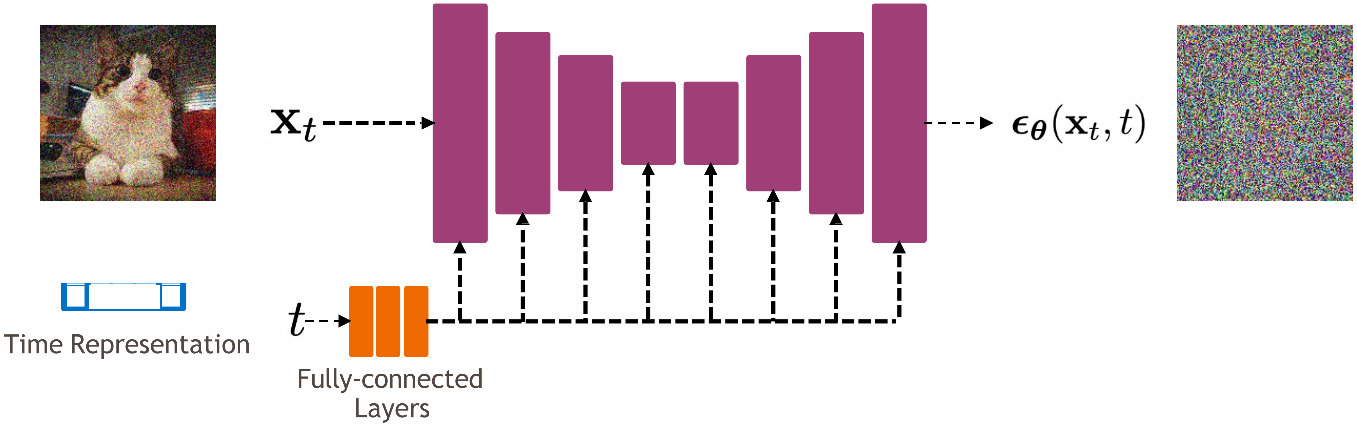
```
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ do
    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    $\mathbf{x}_{t-1} = \boxed{\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,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

Intuitively: During forward process we add noise to image. During reverse process we try to predict that noise with a U-Net and then subtract it from the image to denoise it.

partial credit by CS311n

Implementation Considerations

- Network Architectures
  - Diffusion models often use U-Net architectures with ResNet blocks and self-attention layers to represent $\epsilon_\theta(\mathbf{x}_t, t)$



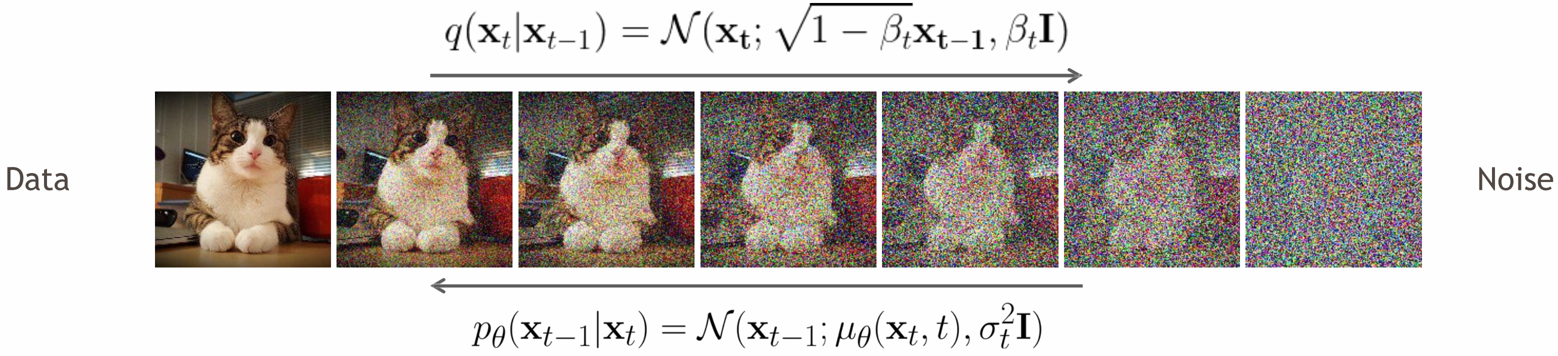
Time representation: sinusoidal positional embeddings or random Fourier features.

Time features are fed to the residual blocks using either simple spatial addition or using adaptive group normalization layers. (see [Dharivwal and Nichol NeurIPS 2021](#))

partial credit by CS311n

Diffusion Parameters

Noise Schedule


$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t\mathbf{I})$
Data
Noise
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2\mathbf{I})$

Above, β_t and σ_t^2 control the variance of the forward diffusion and reverse denoising processes respectively.

Often a linear schedule is used for β_t, and σ_t^2 is set equal to β_t. Slowly increase the amount of added noise.

Kingma et al. NeurIPS 2022 introduce a new parameterization of diffusion models using signal-to-noise ratio (SNR), and show how to learn the noise schedule by minimizing the variance of the training objective.

We can also train σ_t^2 while training the diffusion model by minimizing the variational bound (Improved DPM by Nichol and Dhariwal ICML 2021) or after training the diffusion model (Analytic-DPM by Bao et al. ICLR 2022).

partial credit by CS311n

What happens to an image in the forward diffusion process?

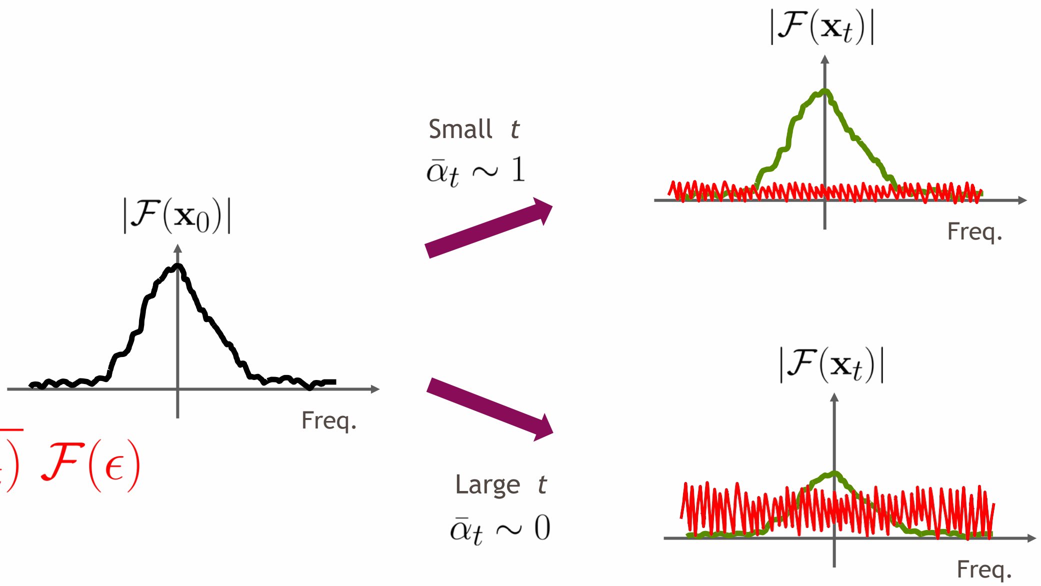
- Recall that sampling from $q(\mathbf{x}_t|\mathbf{x}_0)$ is done using

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\ \mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\ \epsilon \qquad \text{where } \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\ \mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\ \epsilon$$

Fourier Transform

$$\mathcal{F}(\mathbf{x}_t) = \sqrt{\bar{\alpha}_t}\ \mathcal{F}(\mathbf{x}_0) + \sqrt{(1-\bar{\alpha}_t)}\ \mathcal{F}(\epsilon)$$


$|\mathcal{F}(\mathbf{x}_0)|$
Freq.
Small t
$\bar{\alpha}_t \sim 1$
$|\mathcal{F}(\mathbf{x}_t)|$
Freq.
Large t
$\bar{\alpha}_t \sim 0$
$|\mathcal{F}(\mathbf{x}_t)|$
Freq.

- **In the forward diffusion, the high frequency content is perturbed faster.**

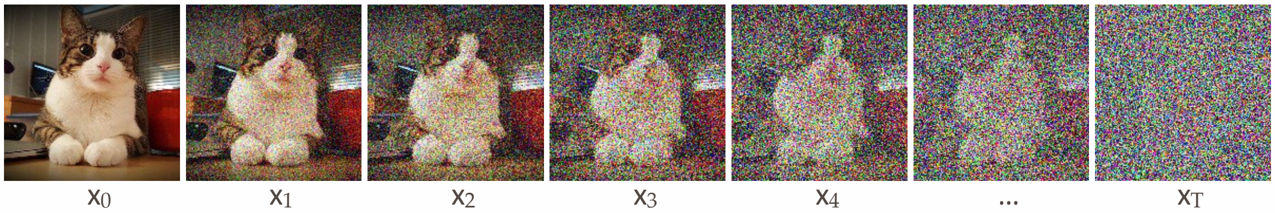
partial credit by CS311n

Content-Detail Tradeoff

$$L_{\text{simple}} = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), \epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I}), t \sim \mathcal{U}(1,T)} \left[||\epsilon - \epsilon_\theta(\underbrace{\sqrt{\bar{\alpha}_t}\ \mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\ \epsilon}_{\mathbf{x}_t}, t)||^2 \right]$$

Reverse denoising process (generative)

Data



Noise

The denoising model is specialized for generating the high-frequency content (i.e., low-level details)

The denoising model is specialized for generating the low-frequency content (i.e., coarse content)

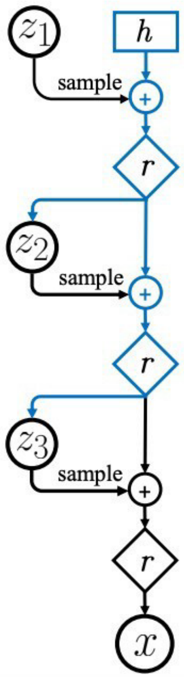
The weighting of the training objective for different timesteps is important!

partial credit by CS311n

Connection to VAEs

- Diffusion models can be considered as a special form of hierarchical VAEs.

- However, in diffusion models:
  - The encoder is fixed
  - The latent variables have the same dimension as the data
  - The denoising model is shared across different timestep
  - The model is trained with some reweighting of the variationa bound.



Vahdat and Kautz, NVAE: A Deep Hierarchical Variational Autoencoder, NeurIPS 2020 Sønderby, et al.. Ladder variational autoencoders, NeurIPS 2016.

partial credit by CS311n

Summary: Denoising Diffusion Probabilistic Models

- Diffusion process can be reversed if the variance of the Gaussian noise added at each step of the diffusion is small enough.

- To reverse the process we train a U-Net that takes input: current noisy image and timestamp, and predicts the noise map..

- Training goal is to make sure that the predicted noise map at each step is unit gaussian (Note that in VAE we also required the latent space to be unit gaussian).

- During sampling/generation, subtract the predicted noise from the noisy image at time t to generate the image at time t-1 (with some weighting).

- The devil is in the details:
  - Network architectures
  - Objective weighting
  - Diffusion parameters (i.e., noise schedule)

partial credit by CS311n

UNSUPERVISED LEARNING?

DIP

Background and Motivation

- State-of-the-art ConvNets for image restoration and generation are almost invariably trained on large datasets of images. One may thus assume that their excellent performance is due to their ability to learn realistic image priors from a large number of example images.

- However, learning alone is insufficient to explain the good performance of deep networks.
- Recent research has shown that generalization requires the structure of the network to "resonate" with the structure of the data.

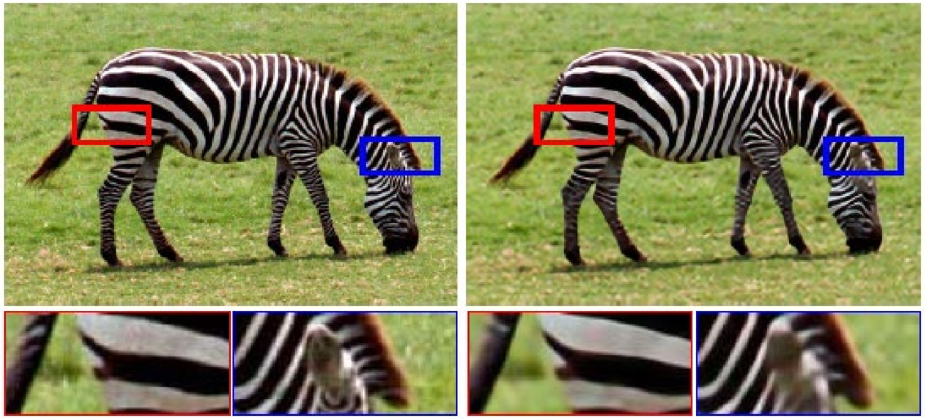
partial credit by CS311n

What did they do?

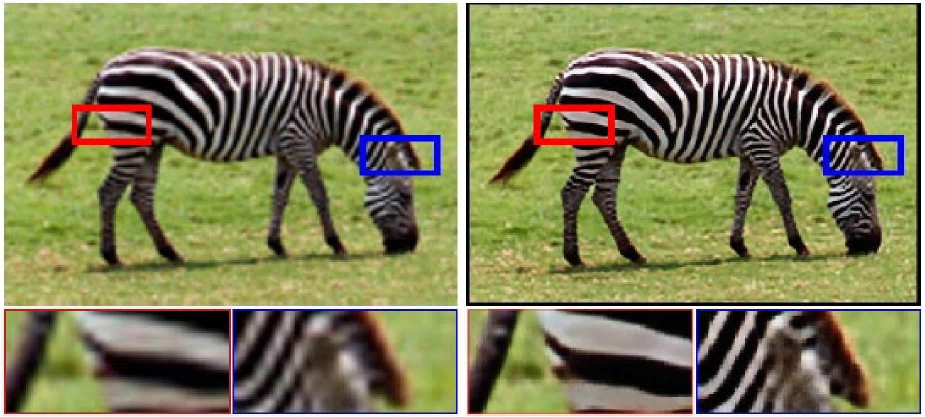
- In this paper, they show that, contrary to the belief that learning is necessary for building good image priors, a great deal of image statistics are captured by the structure of a convolutional image generator independent of learning.
- They cast reconstruction as a conditional image generation problem and show that the only information required to solve it is contained in the single degraded input image and the handcrafted structure of the network used for reconstruction.
- Instead of trying to beat the state-of-art neural networks, they try to show the structure of the network imposes strong prior.

partial credit by CS311n

Result



(a) Ground truth (b) SRResNet [19], **Trained**



(c) Bicubic, **Not trained** (d) Deep prior, **Not trained**

partial credit by CS311n

Image restoration - Method

- x – Clean image
- $\hat{x}$ – Corrupted/degraded image (observed)
- x^*- Restored image

- Degradation for denoising:
- $\hat{x} = x + \epsilon, \ \epsilon \in N(0, \sigma^2)$
- Restoration Model:
- $x^* = \arg\max_x p(x|\hat{x}) = \arg\max_x \underbrace{p(\hat{x}|x)}_{\text{likelihood}} \underbrace{p(x)}_{\text{prior}}$

- If there is no preference for prior, the prior will be a constant. Then,
  - $x^* = \arg\max_x p(\hat{x}|x) = \arg\max_x N(\hat{x}; x, \sigma^2) = \hat{x}$
  - => the best estimation of the clean image is the corrupted image

partial credit by CS311n

- x – Clean image
- $\hat{x}$ – Corrupted image (observed)
- x^*- Restored image

  - $x^* = \arg\max_x p(x|\hat{x}) = \arg\max_x p(\hat{x}|x)p(x)$
  - $= \arg\min_x -logp(\hat{x}|x) - logp(x)$
  - Expressed as energy minimization problem:

- $x^* = \arg\min_x E(x, \hat{x}) + R(x)$
- where $E(x, \hat{x})$ is a task-dependent data term, $R(x)$ is a regularizer

- For example:
- $x^* = \arg\max_x p(\hat{x}|x) = \arg\max_x N(\hat{x}; x, \sigma^2) = \arg\min_x ||x - \hat{x}||^2$

partial credit by CS311n

Deep Image Prior

- $\hat{x}$ – Corrupted image (observed)

- Parametrization:
- Interpreting the neural network as a parametrization
-
-

Fixed input z

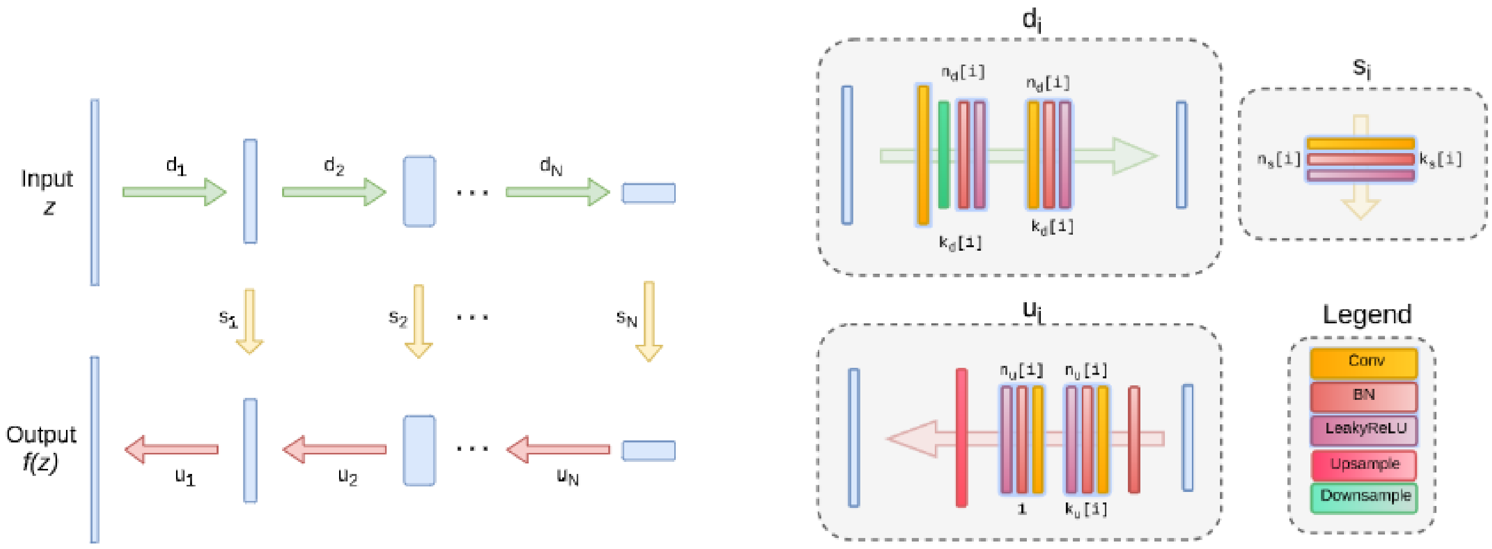
$$x \equiv f_{\theta}(z)$$

Convolutional network with parameter θ

- In particular, most of their experiments are performed using a U-Net type “hourglass” architecture(also known as “decoder-encoder”) with skip-connections, where z and x have the same spatial size.

partial credit by CS311n

Architecture used in the experiments



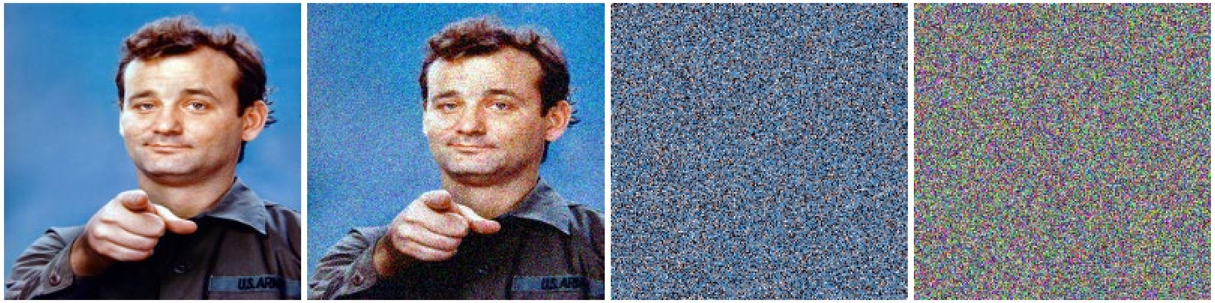
partial credit by CS311n

Deep Image Prior step by step

- $\hat{x}$ – Corrupted image (observed)
- x^*- Restored image

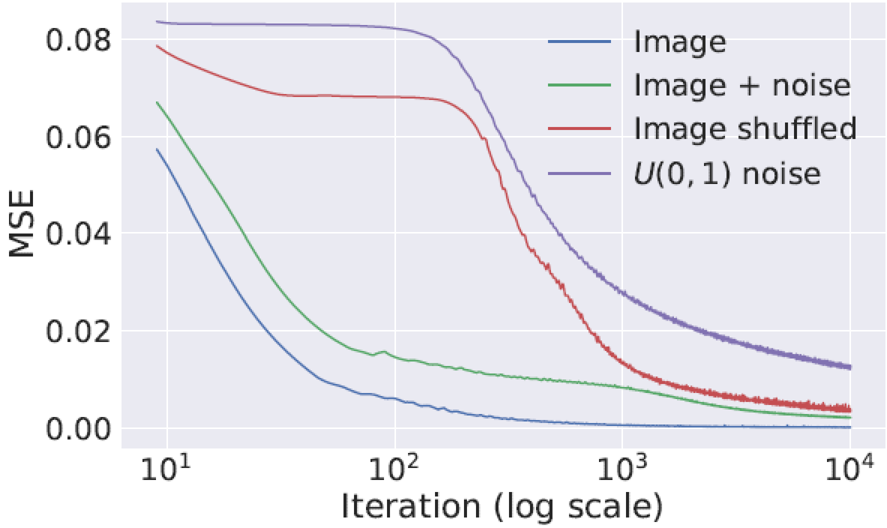
- 1. initialize z
  - For example fill it with uniform noise U(-1, 1)
- 2. Solve $\theta^* = \arg\min_{\theta} E(f_\theta(z); \hat{x}))$
  - With any favorite gradient-based method
  - $\theta^{k+1} = \theta^k - \alpha \frac{\partial E(f_\theta(z); \hat{x}))}{\partial \theta}$;
- 3.Get the solution
- $x^* = f_{\theta^*}(z)$

partial credit by CS311n


(a) Image (b) Image + noise (c) Image shuffled (d) $U(0,1)$ noise

- pedance to

- They restrict the number of iterations in the optimization process to a certain number of iterations.



Data term

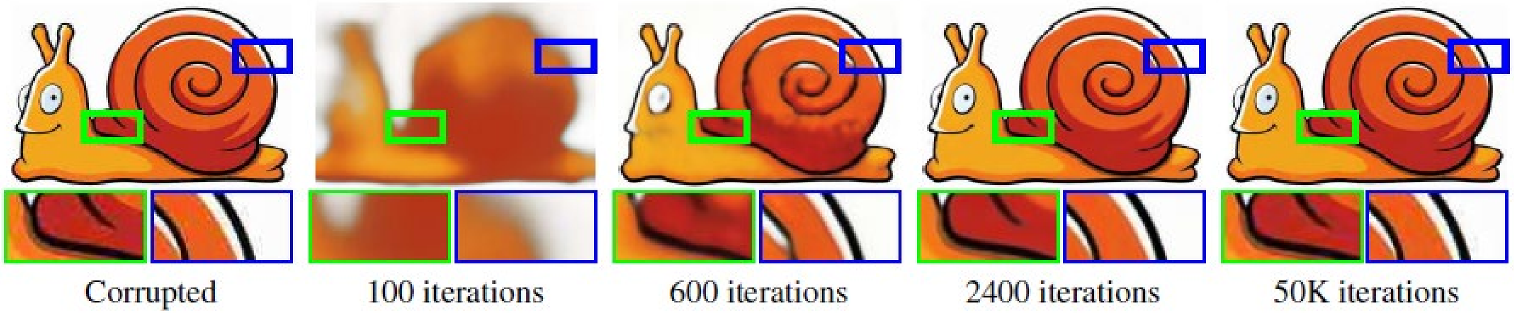
- x – Clean image
- $\hat{x}$ – Corrupted image (observed)
- m- Binary mask

- Objective: $\theta^* = \arg\min_\theta E(f_\theta(z); \hat{x}))$
- Denoising: $E(x, \hat{x}) = ||x - \hat{x}||^2$ Needs early stopping!
- Inpainting: $E(x, \hat{x}) = ||(x - \hat{x}) \odot m||^2$, where $\odot$ is Hadamard's product, m is binary mask
- Super-resolution: $E(x, \hat{x}) = ||d(x) - \hat{x}||^2$, where $d(\cdot)$ is a downsampling operator to resize the image
- Feature-inv: $E(x, \hat{x}) = ||\phi(x) - \phi(\hat{x})||^2$, where ϕ is the first several layers of a neural network trained to perform

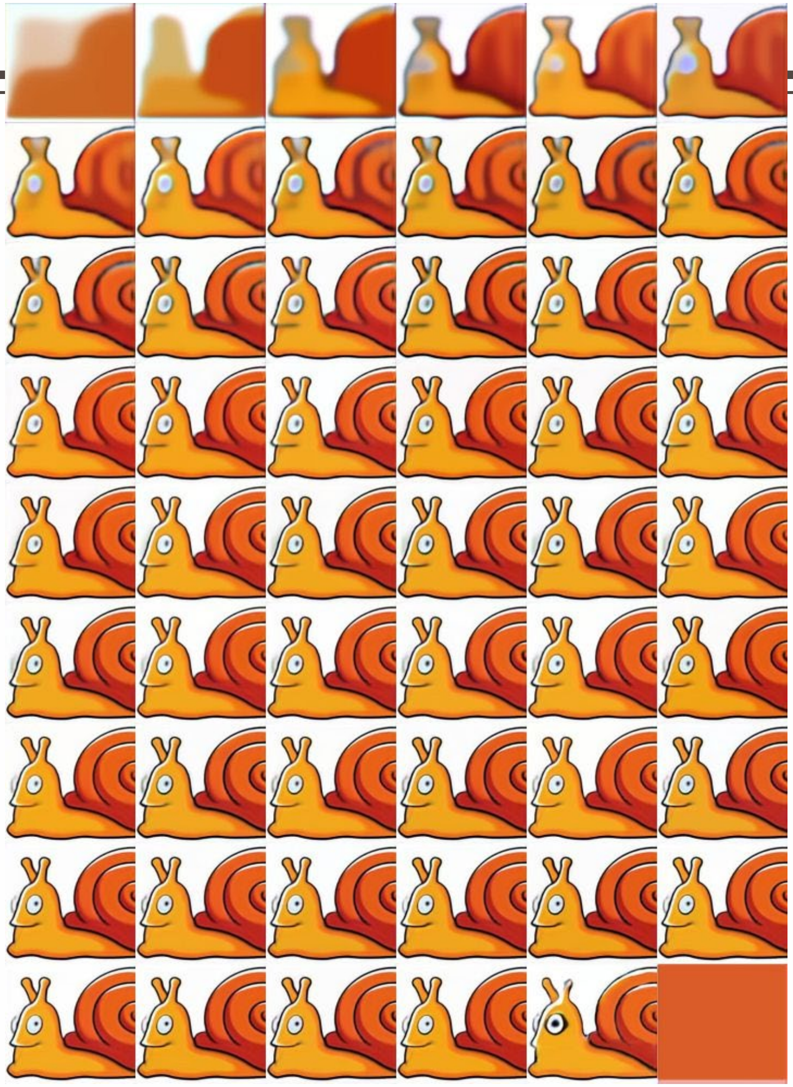
partial credit by CS311n

Experiments

- Denoising and generic reconstruction
- Deep Image Prior approach can restore an image with a complex degradation (JPEG compression in this case). As the optimization process progresses, the deep image prior allows to recover most of the signal while getting rid of halos and blockiness (after 2400 iterations) before eventually overfitting to the input (at 50K iterations).

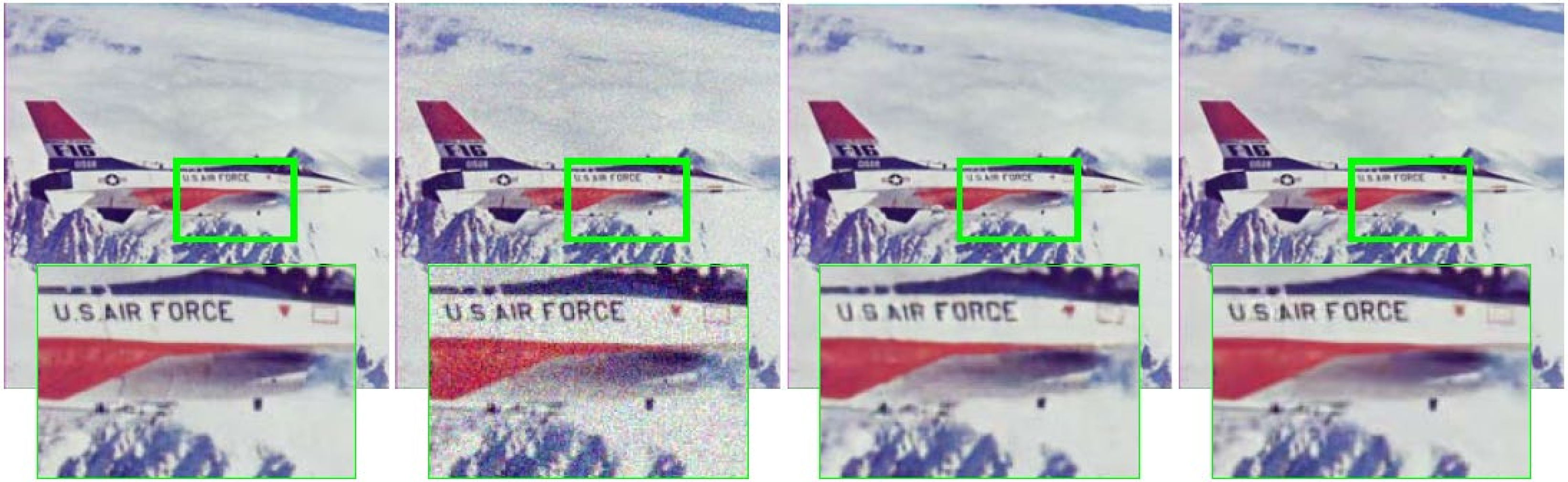


partial credit by CS311n



partial credit by CS311n

- The deep image prior is successful at recovering both man-made and natural patterns.

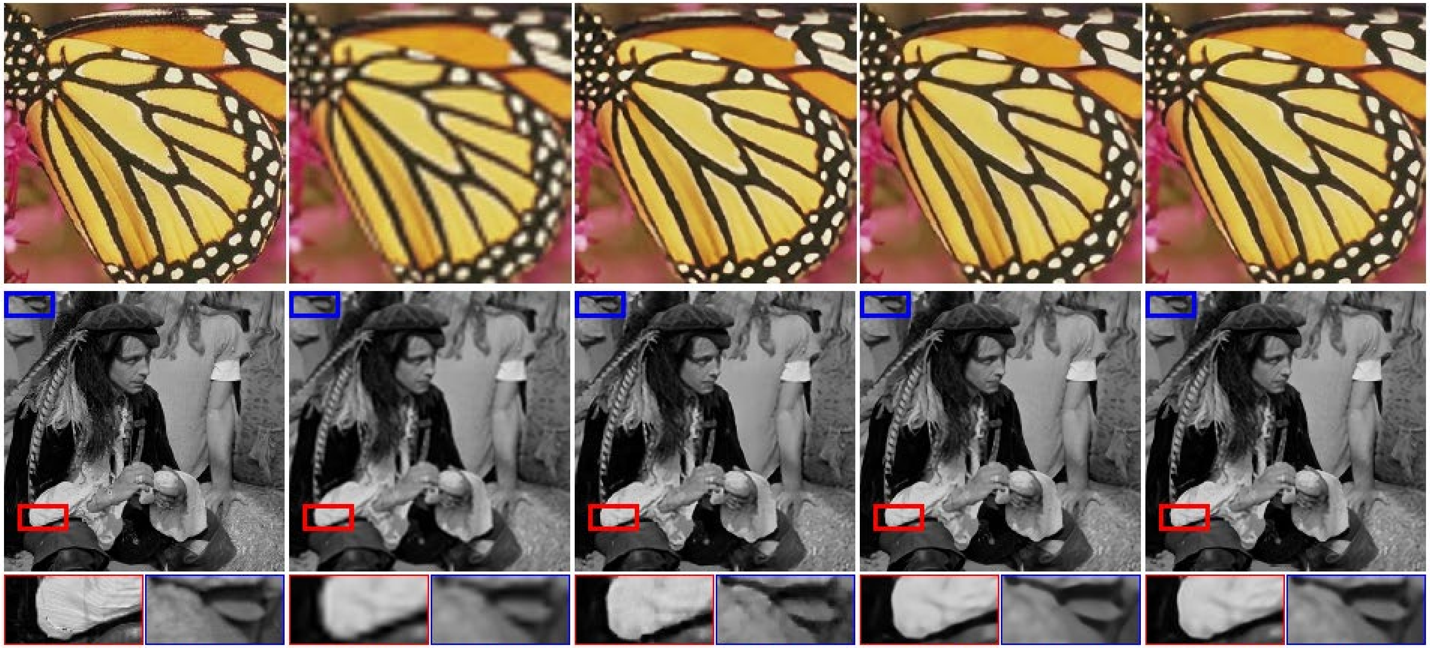


(a) GT (b) Input (c) Ours (d) CBM3D

partial credit by CS311n

Super-resolution

- use a scaling factor of 4 to compare to other works
- fix the number of optimization steps to be 2000 for every image

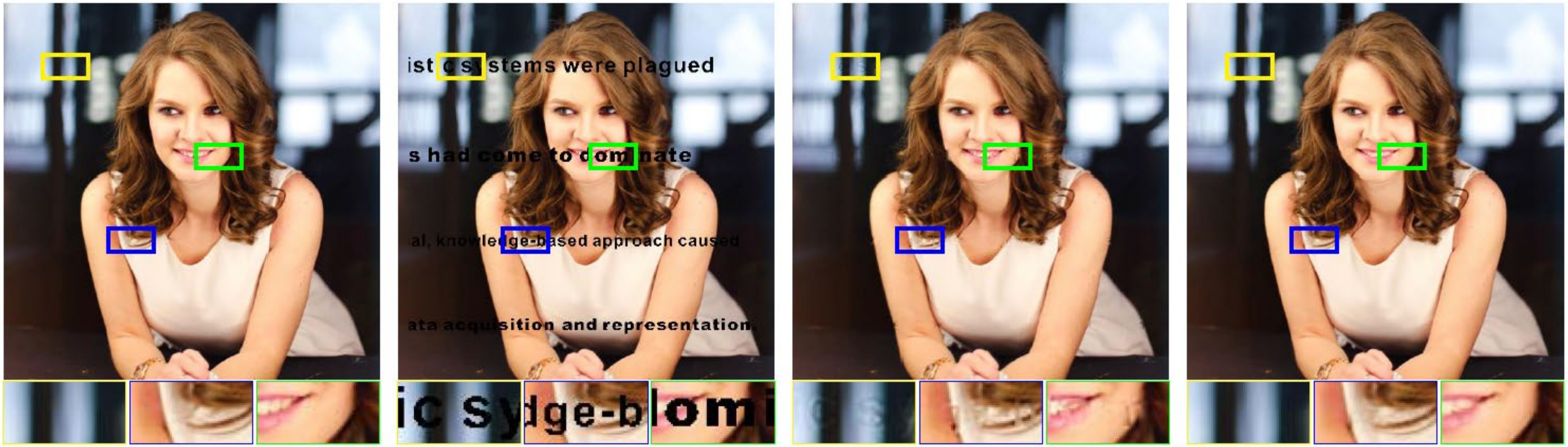


(a) Original image (b) Bicubic, **Not trained** (c) Ours, **Not trained** (d) LapSRN, **Trained** (e) SRResNet, **Trained**

partial credit by CS311n

Inpainting

- Text inpainting

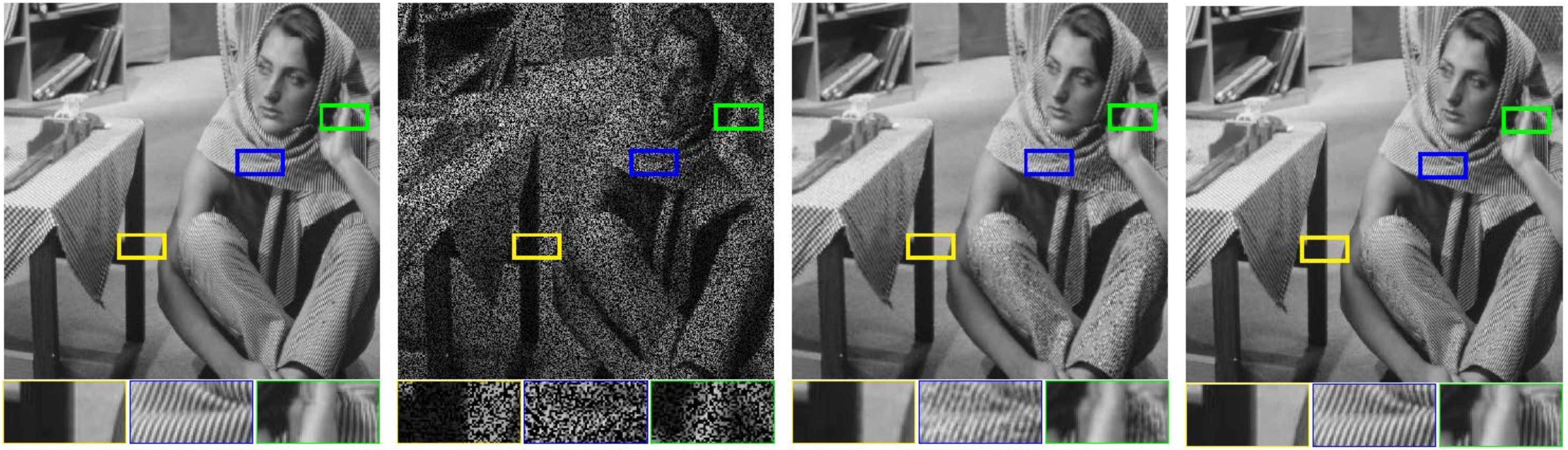

ist c systems were plagued
s had come to dominate
al, knowledge-based approach caused
ata acquisition and representation.
ic Sy
ige-b
omi

(a) Original image (b) Corrupted image (c) Shepard networks [27] (d) Deep Image Prior

partial credit by CS311n

Image restoration

- sampled to drop 50% of pixels at random
- g is the result from comparison with Shepard networks

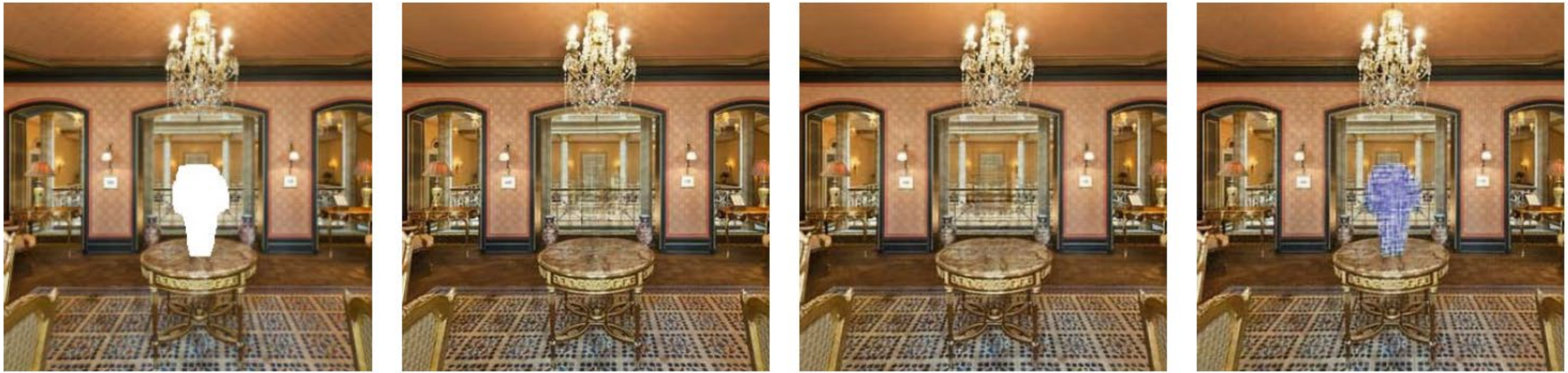


(e) Original image (f) Corrupted image (g) [25], PSNR = 28.1 (h) Deep Img. Prior, PSNR = 32.22

partial credit by CS311n

Inpainting of large holes

- The deep image prior utilizes context of the image and interpolates the unknown region with textures from the known part. Such behaviour highlights the relation between the deep image prior and traditional self-similarity priors

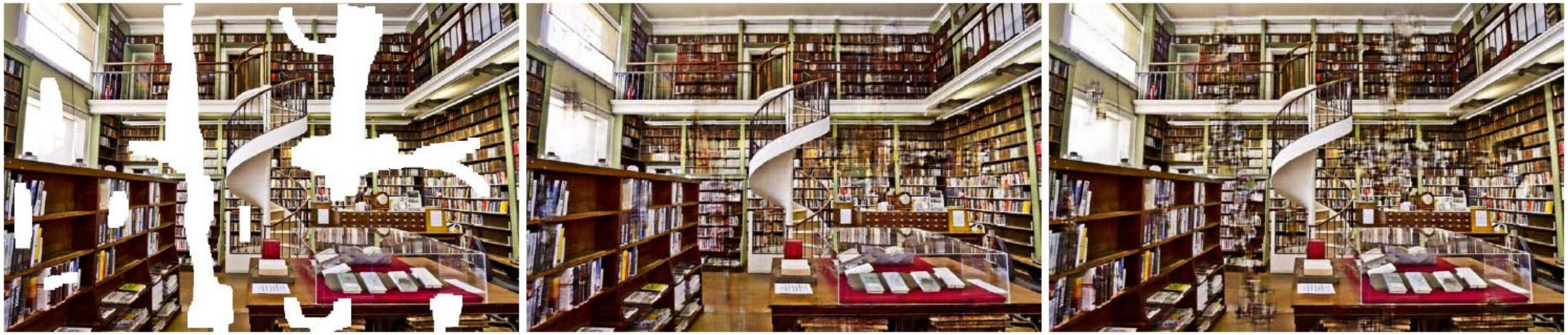


(a) Corrupted image (b) Global-Local GAN [15] (c) Ours, LR $= 0.01$ (d) Ours, LR $= 10^{-4}$

partial credit by CS311n

Inpainting of large holes

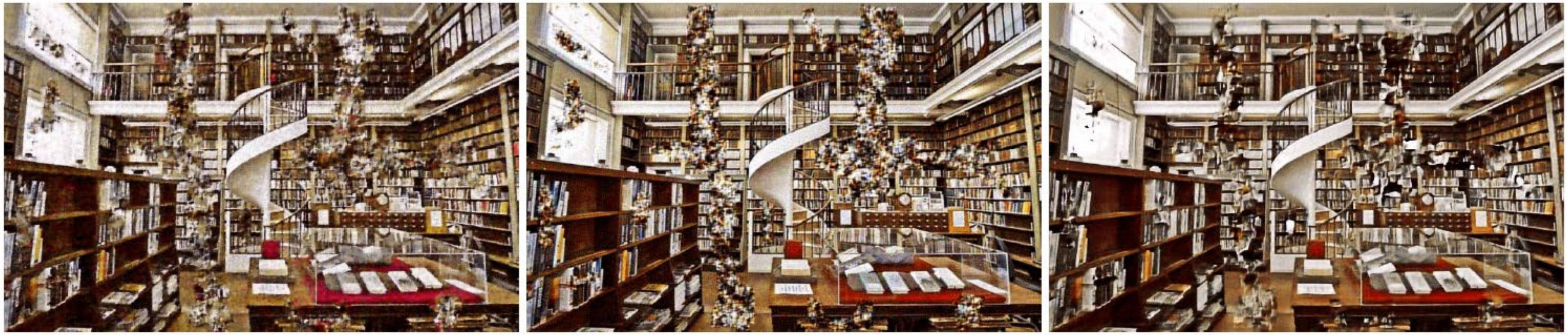
- Inpainting using different depths and architectures.
- The figure shows that much better inpainting results can be obtained by using deeper random networks. However, adding skip connections to ResNet in U-Net is highly detrimental.



(a) Input (white=masked)

(b) Encoder-decoder, depth=6

(c) Encoder-decoder, depth=4



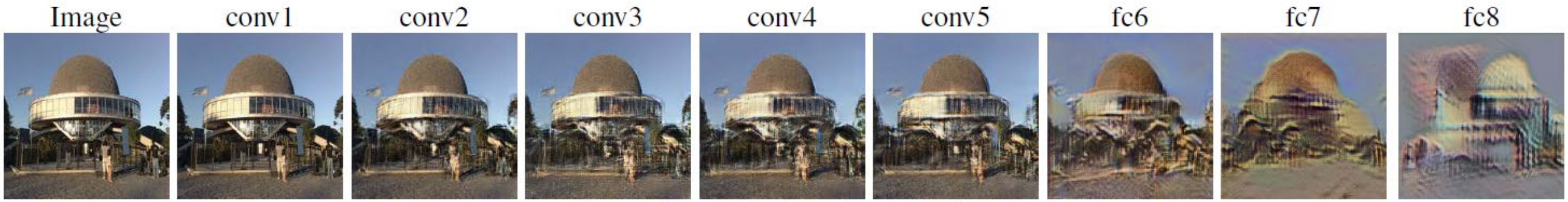
(d) Encoder-decoder, depth=2

(e) ResNet, depth=8

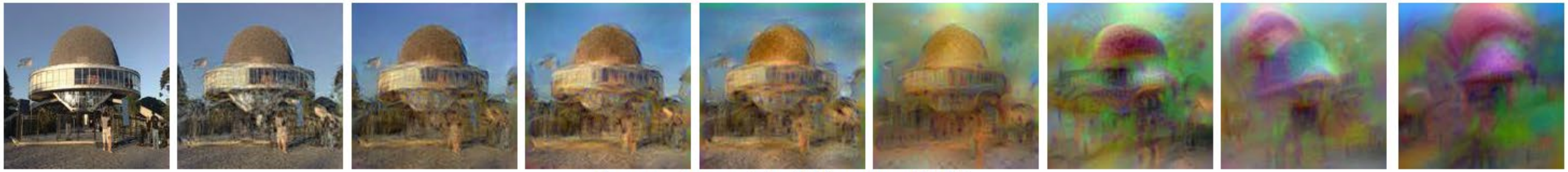
(f) U-net, depth=5

Feature-inversion (AlexNet Inversion)

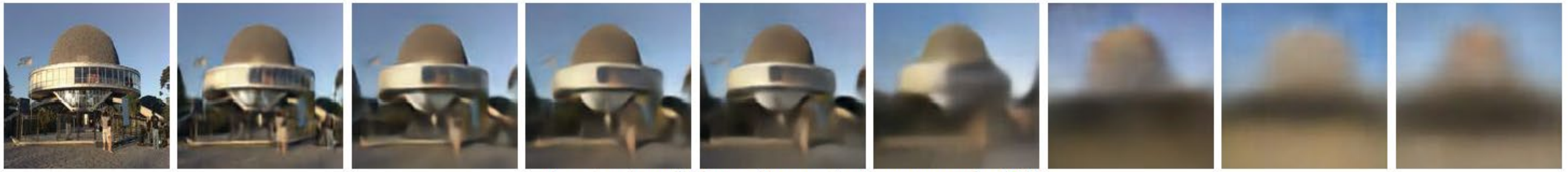
- Given the image on the left, it shows the natural pre-image obtained by inverting different layers of AlexNet using three different regularizers.
- The deep image prior results in inversions at least as interpretable as the ones of [8].



Inversion with deep image prior



Inversion with TV prior [21]

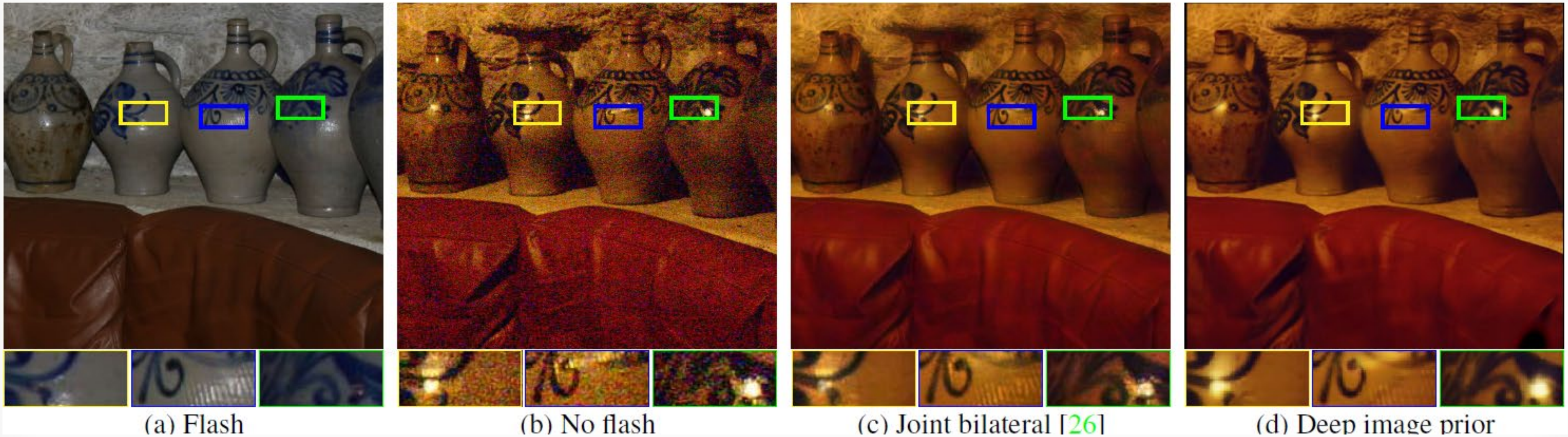


Pre-trained deep inverting network [8]

partial credit

Flash/No Flash

- The proposed approach can be extended to the tasks of the restoration of multiple images
- The flash-no flash image pair-based restoration is to obtain an image of a scene with the lighting similar to a no-flash image, while using the flash image as a guide to reduce the noise level.
- The deep image prior allows to obtain low-noise reconstruction with the lighting very close to the no-flash image.



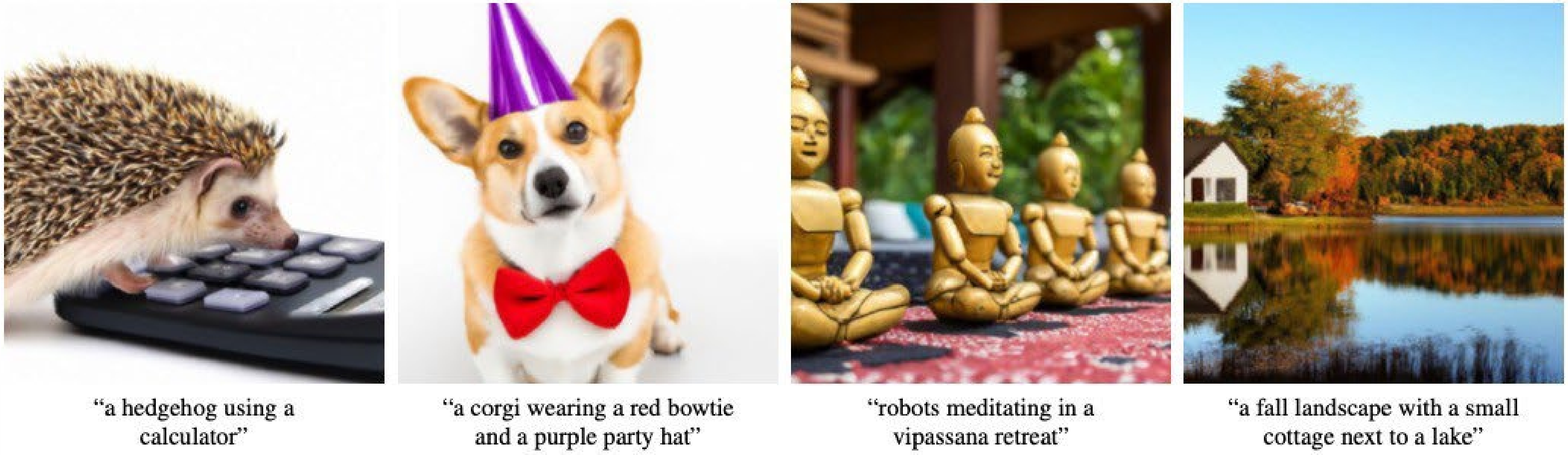
(a) Flash (b) No flash (c) Joint bilateral [26] (d) Deep image prior

partial credit by CS311n

APPLICATIONS
IMAGE SYNTHESIS, CONTROLLABLE GENERATION, TEXT-TO-IMAGE

GLIDE
OpenAI

- A 64x64 base model + a 64x64 → 256x256 super-resolution model.
- Tried classifier-free and CLIP guidance. Classifier-free guidance



“a hedgehog using a calculator”

“a corgi wearing a red bowtie and a purple party hat”

“robots meditating in a vipassana retreat”

“a fall landscape with a small cottage next to a lake”

Samples generated with classifier-free guidance (256x256)

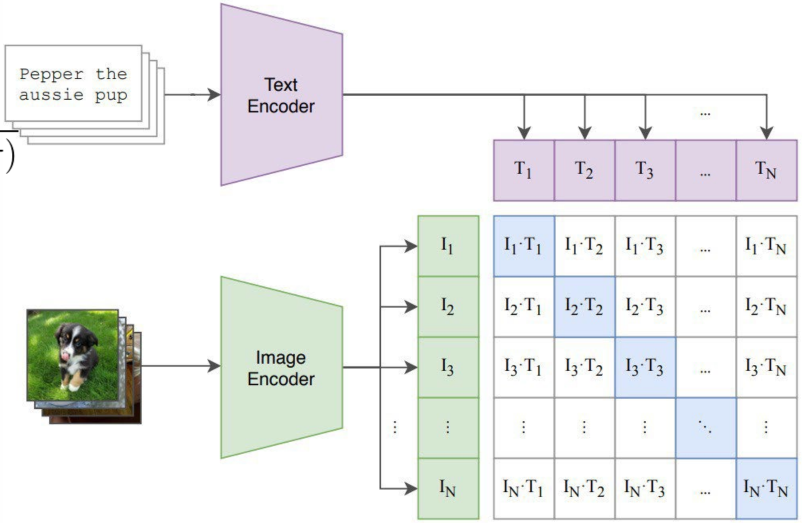
Nichol et al., “GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models”, 2021.

partial credit by CS311n

CLIP guidance

- What is a CLIP model?
- Trained by contrastive cross-entropy loss:

$$-\log \frac{\exp(f(\mathbf{x}_i)\cdot g(\mathbf{c}_j)/\tau)}{\sum_k \exp(f(\mathbf{x}_i)\cdot g(\mathbf{c}_k)/\tau)} - \log \frac{\exp(f(\mathbf{x}_i)\cdot g(\mathbf{c}_j)/\tau)}{\sum_k \exp(f(\mathbf{x}_k)\cdot g(\mathbf{c}_j)/\tau)}$$



	T_1	T_2	T_3	...	T_N
I_1	$I_1 \cdot T_1$	$I_1 \cdot T_2$	$I_1 \cdot T_3$	...	$I_1 \cdot T_N$
I_2	$I_2 \cdot T_1$	$I_2 \cdot T_2$	$I_2 \cdot T_3$	...	$I_2 \cdot T_N$
I_3	$I_3 \cdot T_1$	$I_3 \cdot T_2$	$I_3 \cdot T_3$	...	$I_3 \cdot T_N$
⋮	⋮	⋮	⋮	⋱	⋮
I_N	$I_N \cdot T_1$	$I_N \cdot T_2$	$I_N \cdot T_3$	...	$I_N \cdot T_N$

- The optimal value of $f(\mathbf{x})\cdot g(\mathbf{c})$ is

$$\log \frac{p(\mathbf{x},\mathbf{c})}{p(\mathbf{x})p(\mathbf{c})} = \log p(\mathbf{c}|\mathbf{x}) - \log p(\mathbf{c})$$

Radford et al., "Learning Transferable Visual Models From Natural Language Supervision", 2021.
Nichol et al., "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", 2021.

partial credit by CS311n

CLIP guidance

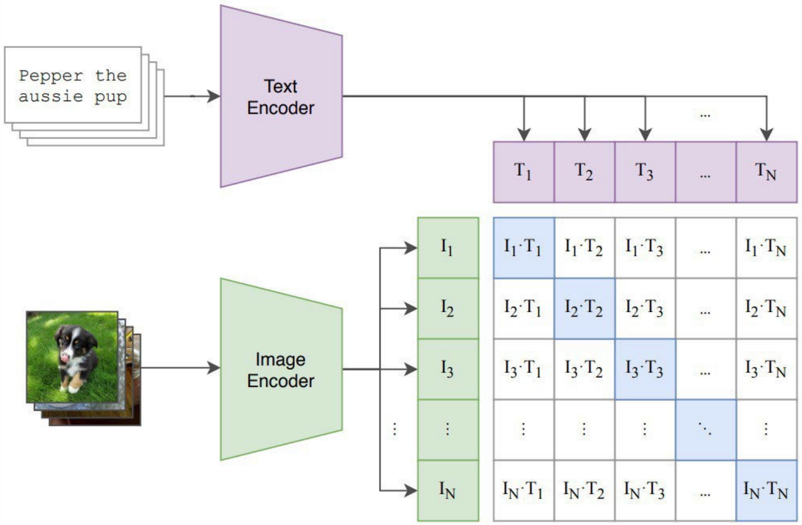
- Replace the classifier in classifier guidance with a CLIP model

- Sample with a modified score:

$$\nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega \log p(\mathbf{c}|\mathbf{x}_t)]$$

$$= \nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega(\underbrace{\log p(\mathbf{c}|\mathbf{x}_t) - \log p(\mathbf{c})}_{\text{CLIP model}})]$$

$$= \nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega(f(\mathbf{x}_t) \cdot g(\mathbf{c}))]$$



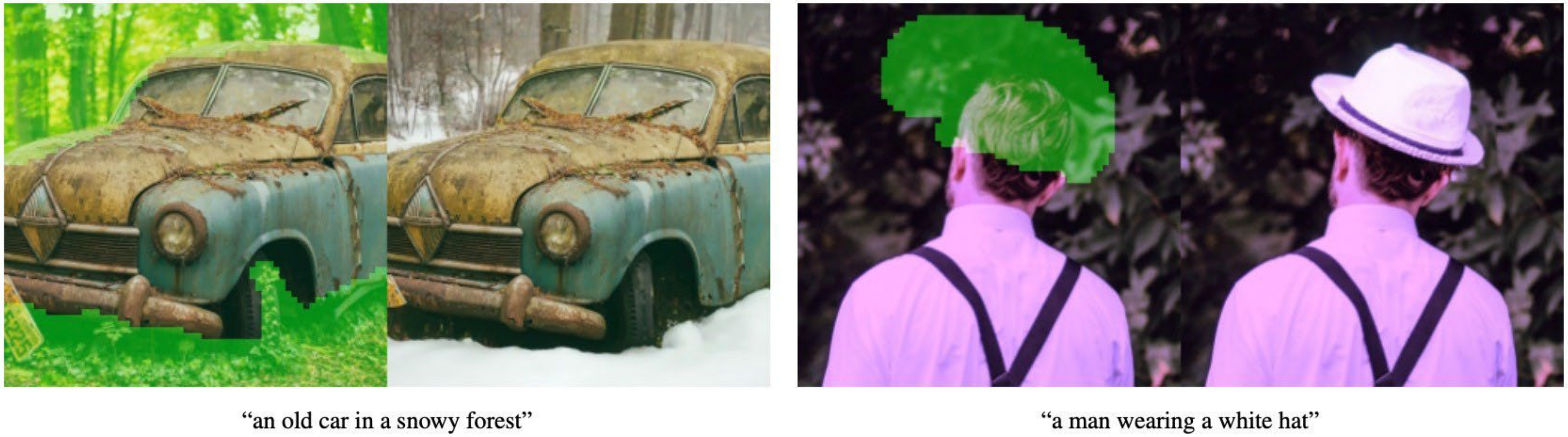
	T_1	T_2	T_3	...	T_N
I_1	$I_1 \cdot T_1$	$I_1 \cdot T_2$	$I_1 \cdot T_3$	...	$I_1 \cdot T_N$
I_2	$I_2 \cdot T_1$	$I_2 \cdot T_2$	$I_2 \cdot T_3$	...	$I_2 \cdot T_N$
I_3	$I_3 \cdot T_1$	$I_3 \cdot T_2$	$I_3 \cdot T_3$	...	$I_3 \cdot T_N$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
I_N	$I_N \cdot T_1$	$I_N \cdot T_2$	$I_N \cdot T_3$	...	$I_N \cdot T_N$

Radford et al., "Learning Transferable Visual Models From Natural Language Supervision", 2021.
Nichol et al., "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", 2021.

partial credit by CS311n

GLIDE
OpenAI

- Fine-tune the model especially for inpainting: feed randomly occluded images with an additional mask channel as the input.



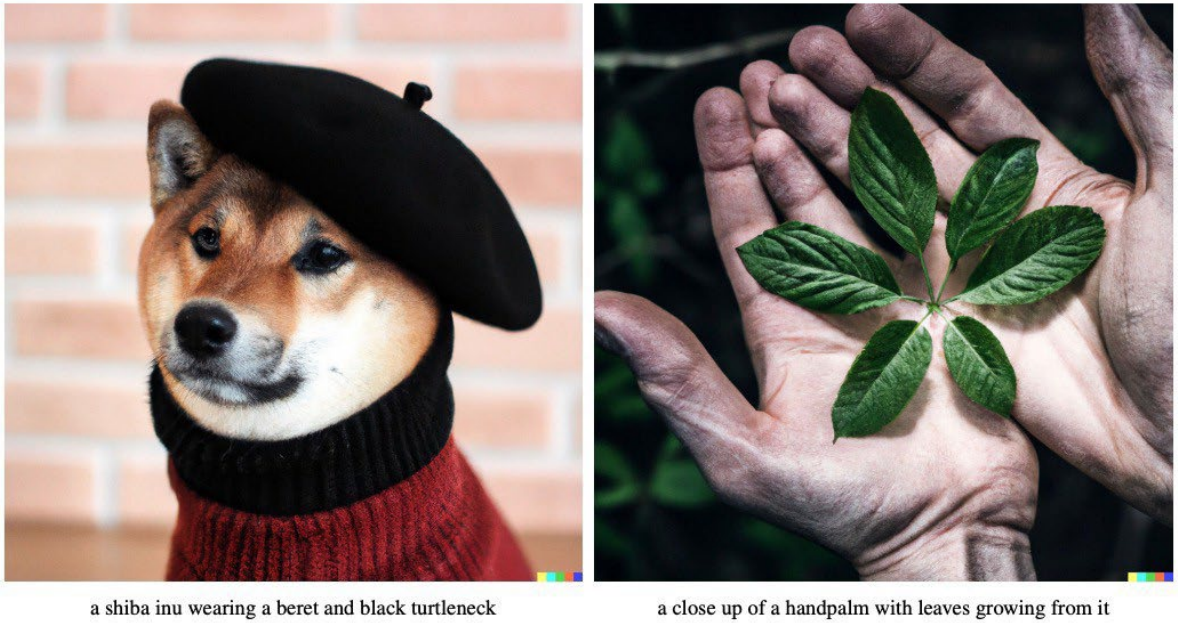
Text-conditional image inpainting examples

Nichol et al., "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", 2021.

partial credit by CS311n

DALL·E 2

- OpenAI


a shiba inu wearing a beret and black turtleneck a close up of a handpalm with leaves growing from it

1kx1k Text-to-image generation.

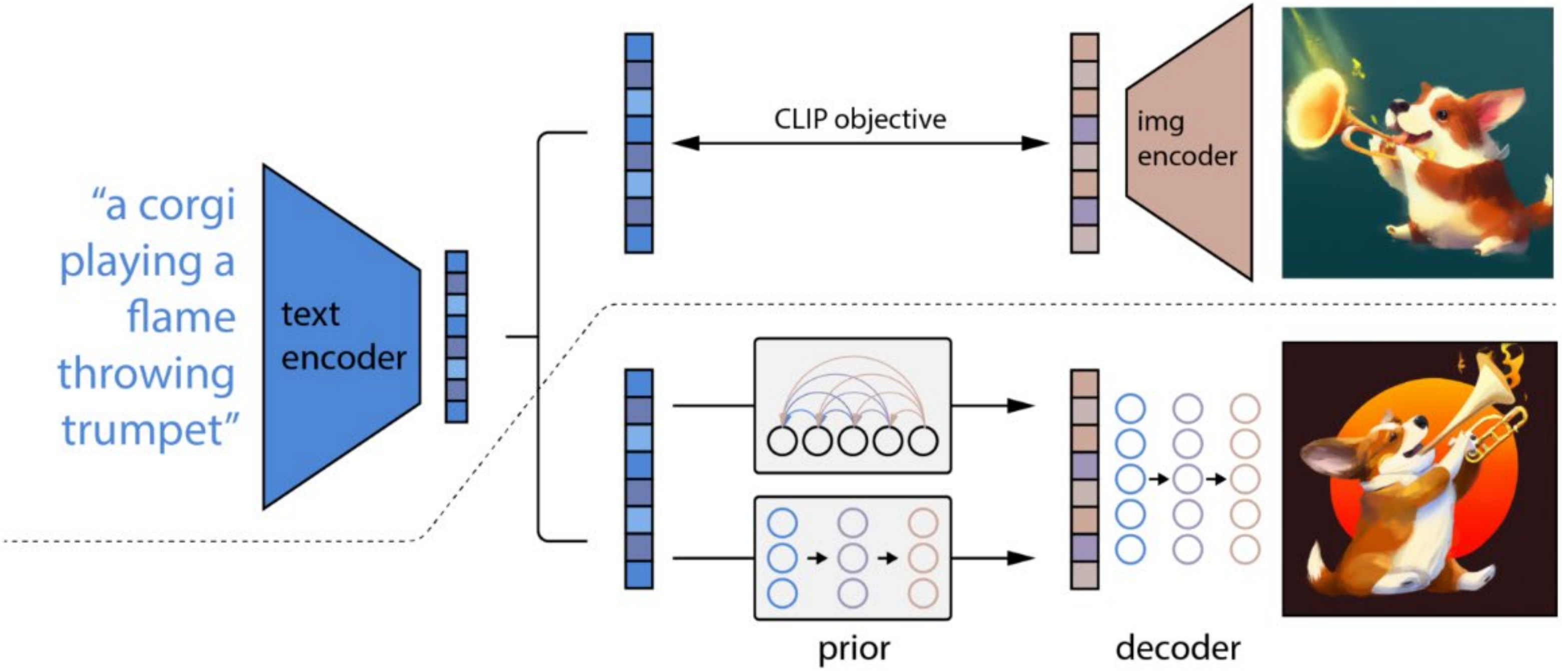
Outperform DALL-E (autoregressive transformer).

[Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.]

partial credit by CS311n

DALL·E 2

- Model components
  - Prior: produces CLIP image embeddings conditioned on the caption.
  - Decoder: produces images conditioned on CLIP image embeddings and text.

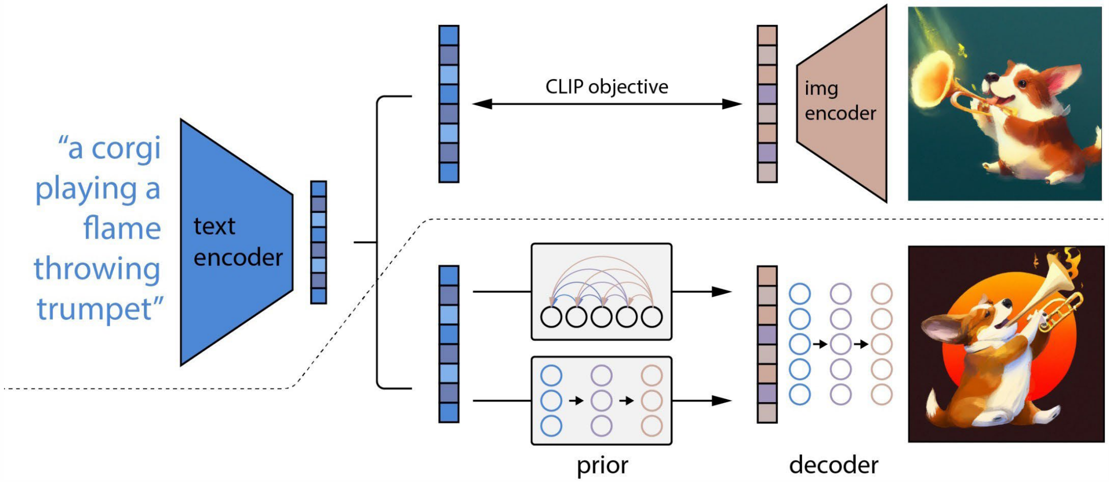


Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

partial credit by CS311n

DALL·E 2
Model components

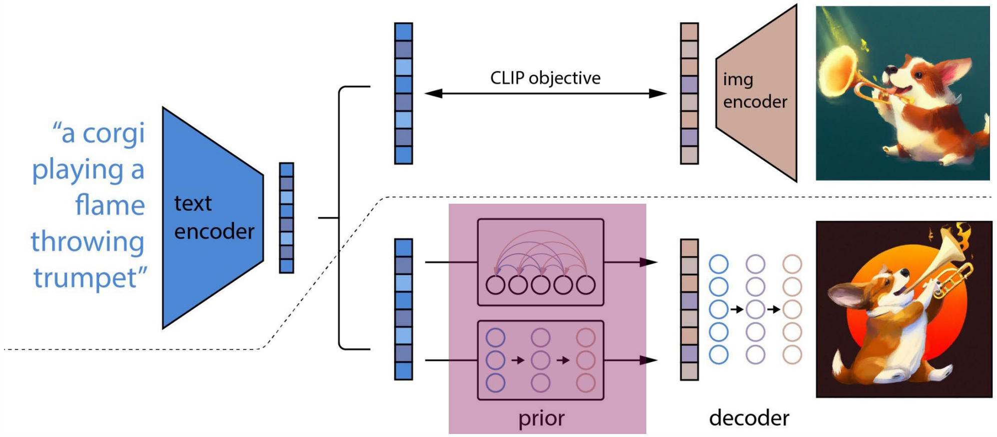
- Why conditional on CLIP image embeddings?
  - CLIP image embeddings capture high-level semantic meaning.
  - Latents in the decoder model take care of the rest.
  - The bipartite latent representation enables several text-guided image manipulation tasks.



partial credit by CS311n

DALL·E 2
Model components (1/2): prior model

- Prior: produces CLIP image embeddings conditioned on the caption.
  - Option 1. autoregressive prior: quantize image embedding to a seq. of discrete codes and predict them autoregressively.
  - Option 2. diffusion prior: model the continuous image embedding by diffusion models conditioned on caption.

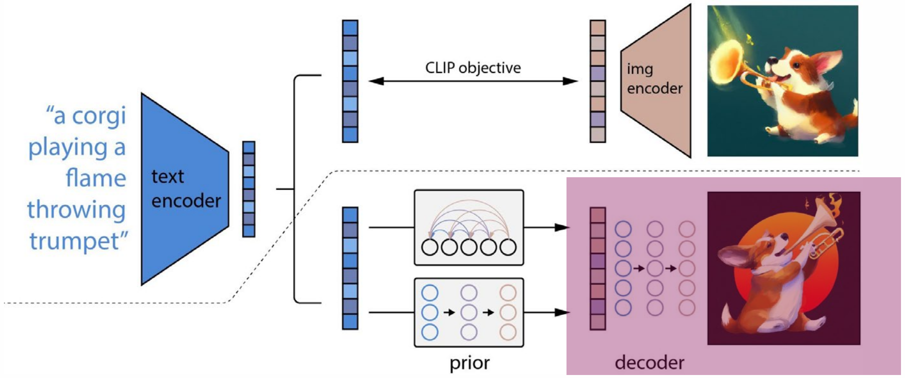


[Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.]()

partial credit by CS311n

DALL·E 2
Model components (2/2): decoder model

- Decoder: produces images conditioned on CLIP image embeddings (and text).
  - Cascaded diffusion models: 1 base model (64x64), 2 super-resolution models (64x64 → 256x256, 256x256 → 1024x1024).
  - Largest super-resolution model is trained on patches and takes full-res inputs at inference time.
  - Classifier-free guidance & noise conditioning augmentation are important.



Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

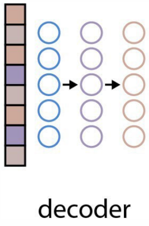
partial credit by CS311n

DALL·E 2
Bipartite latent representations

Bipartite latent representations $(\mathbf{z}, \mathbf{x}_T)$

$\mathbf{z}$: CLIP image embeddings

$\mathbf{x}_T$: inversion of DDIM sampler
(latents in the decoder model)



Near exact reconstruction

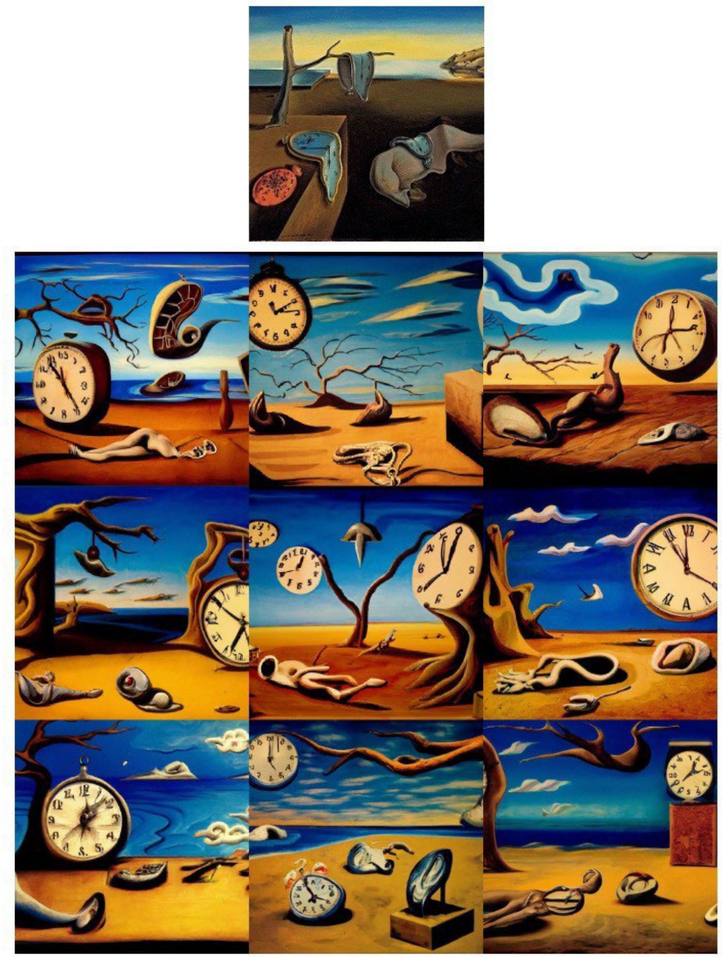
Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

partial credit by CS311n

DALL·E 2
Image variations

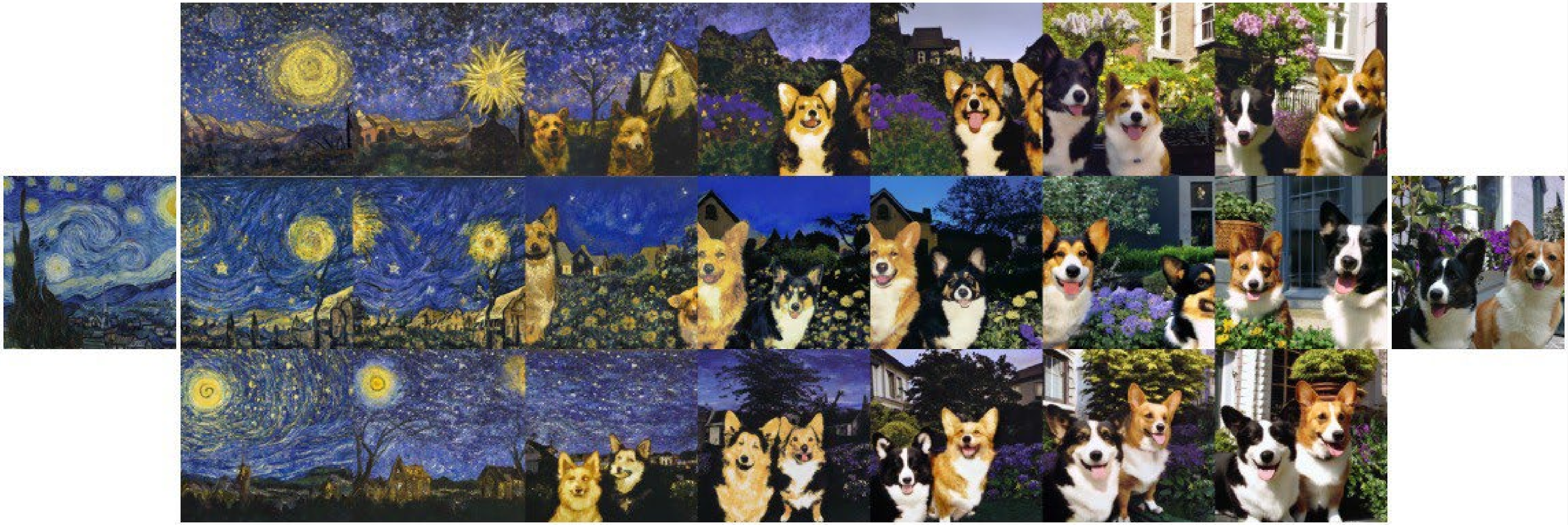
Fix the CLIP embedding $\mathbf{z}$.

Decode using different decoder latents $\mathbf{x}_T$



partial credit by CS311n

DALL·E 2
Image interpolation



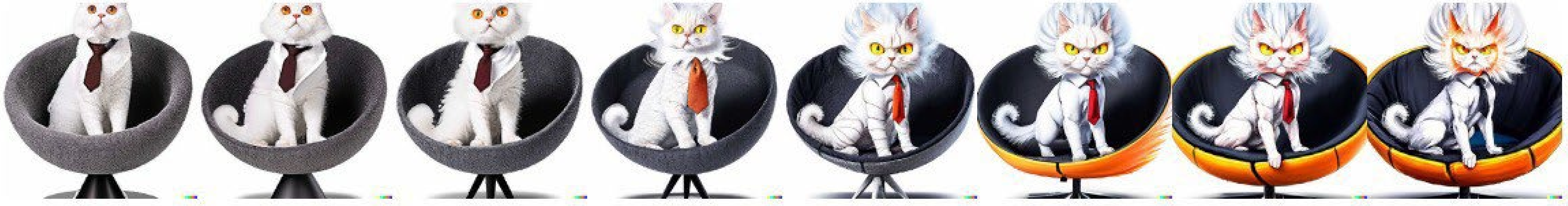
Interpolate image CLIP embeddings $\mathbf{z}$.

Use different $\mathbf{x}_T$ to get different interpolation trajectories.

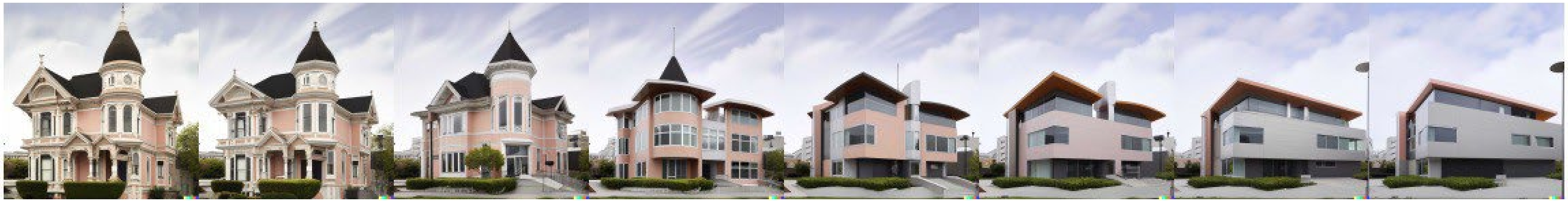
Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

partial credit by CS311n

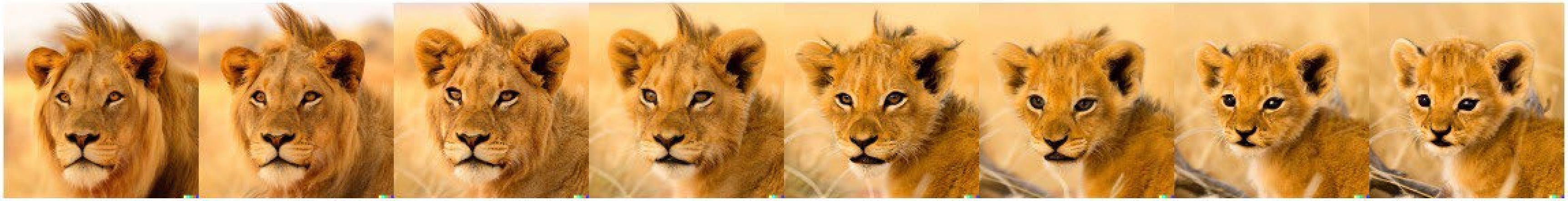
DALL·E 2
Text Diffs



a photo of a cat → an anime drawing of a super saiyan cat, artstation



a photo of a victorian house → a photo of a modern house



a photo of an adult lion → a photo of lion cub

Change the image CLIP embedding towards the difference of the text CLIP embeddings of two prompts.

Decoder latent is kept as a constant.

partial credit by CS311n

Imagen
Google Research, Brain team

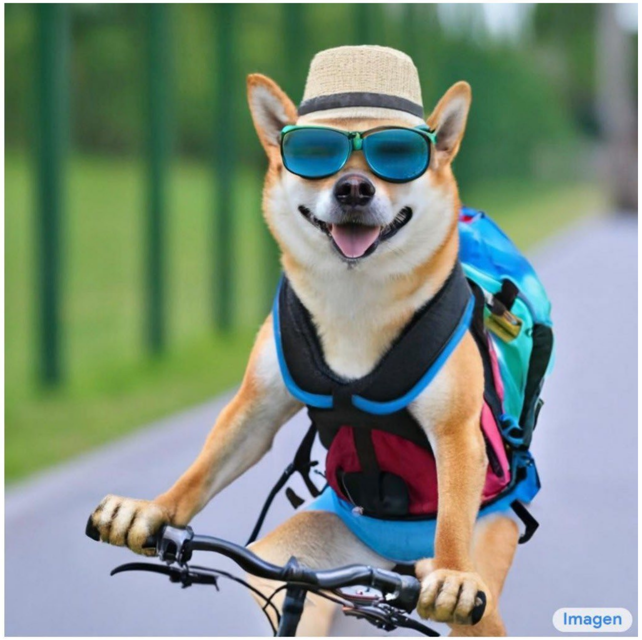
- Input: text; Output: 1kx1k images
  - An unprecedented degree of photorealism
  - SOTA automatic scores & human ratings
  - A deep level of language understanding
  - Extremely simple
  - no latent space, no quantization

A brain riding a rocketship heading towards the moon.

Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.

partial credit by CS311n

Imagen (Google Research, Brain team…被解散了…)

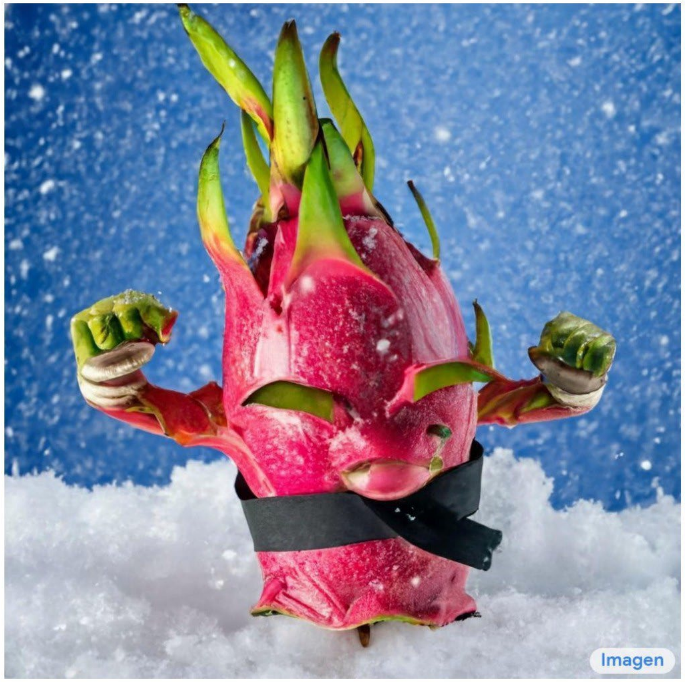


A photo of a Shiba Inu dog with a backpack riding a bike. It is wearing sunglasses and a beach hat.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

partial credit by CS311n

Imagen
Google Research, Brain team

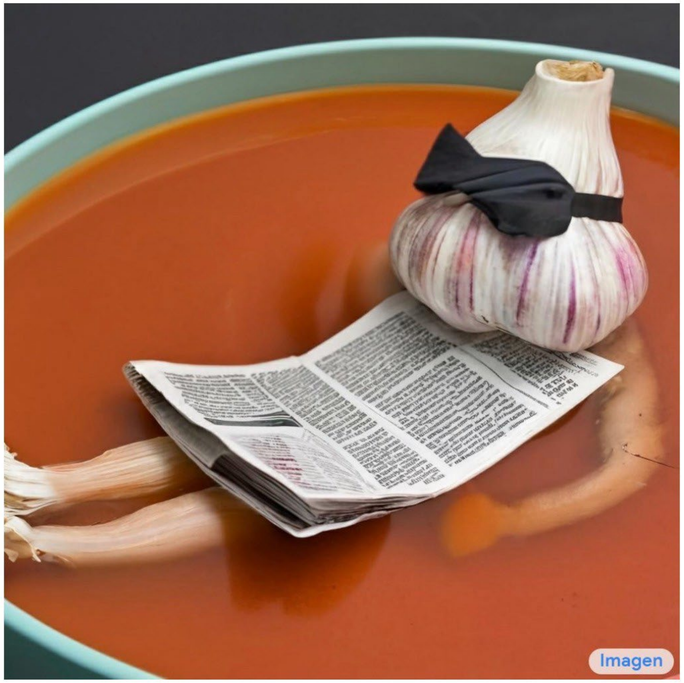


A dragon fruit wearing karate belt in the snow.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

partial credit by CS311n

Imagen
Google Research, Brain team

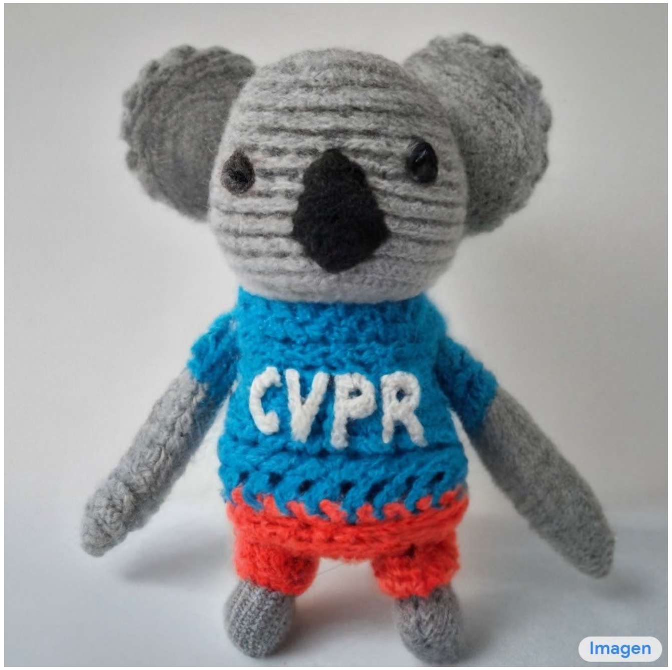


A relaxed garlic with a blindfold reading a newspaper while floating in a pool of tomato soup.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

partial credit by CS311n

Imagen
Google Research, Brain team



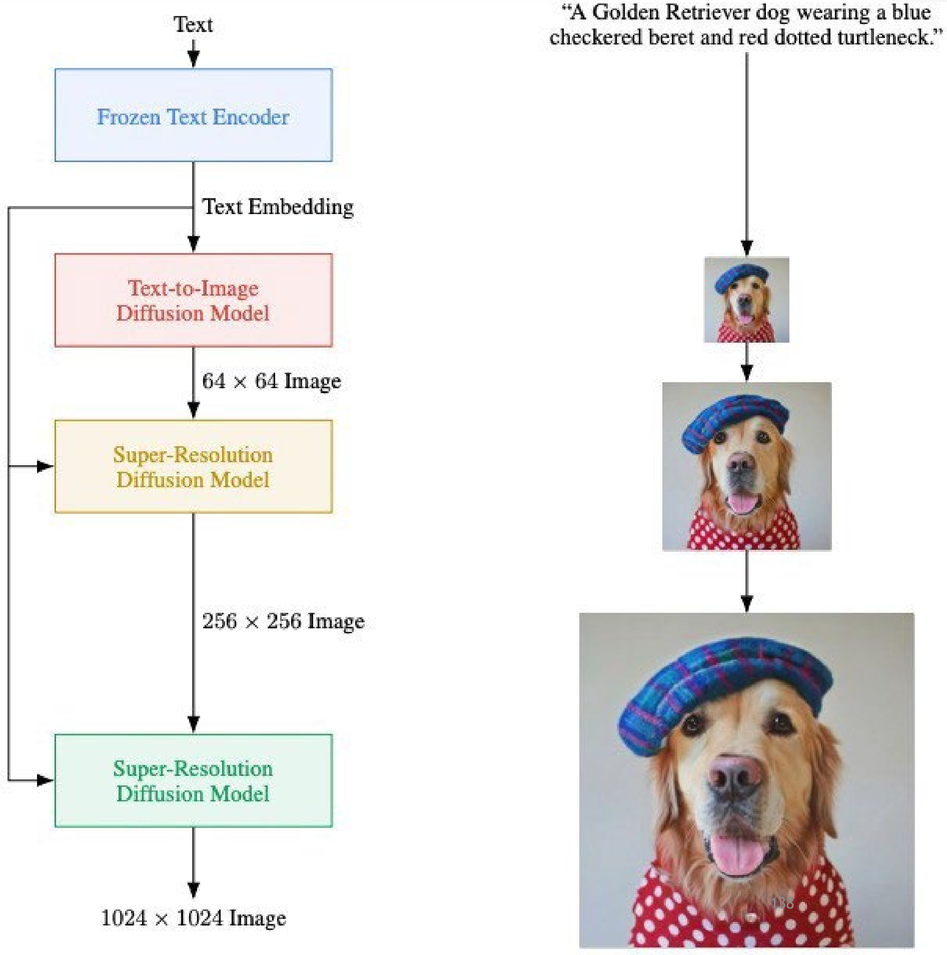
A cute hand-knitted koala wearing a sweater with 'CVPR' written on it.

Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.

partial credit by CS311n

Imagen

- Key modeling components:
  - Cascaded diffusion models
  - Classifier-free guidance and dynamic thresholding.
  - Frozen large pretrained language models as text encoders. (T5-XXL)

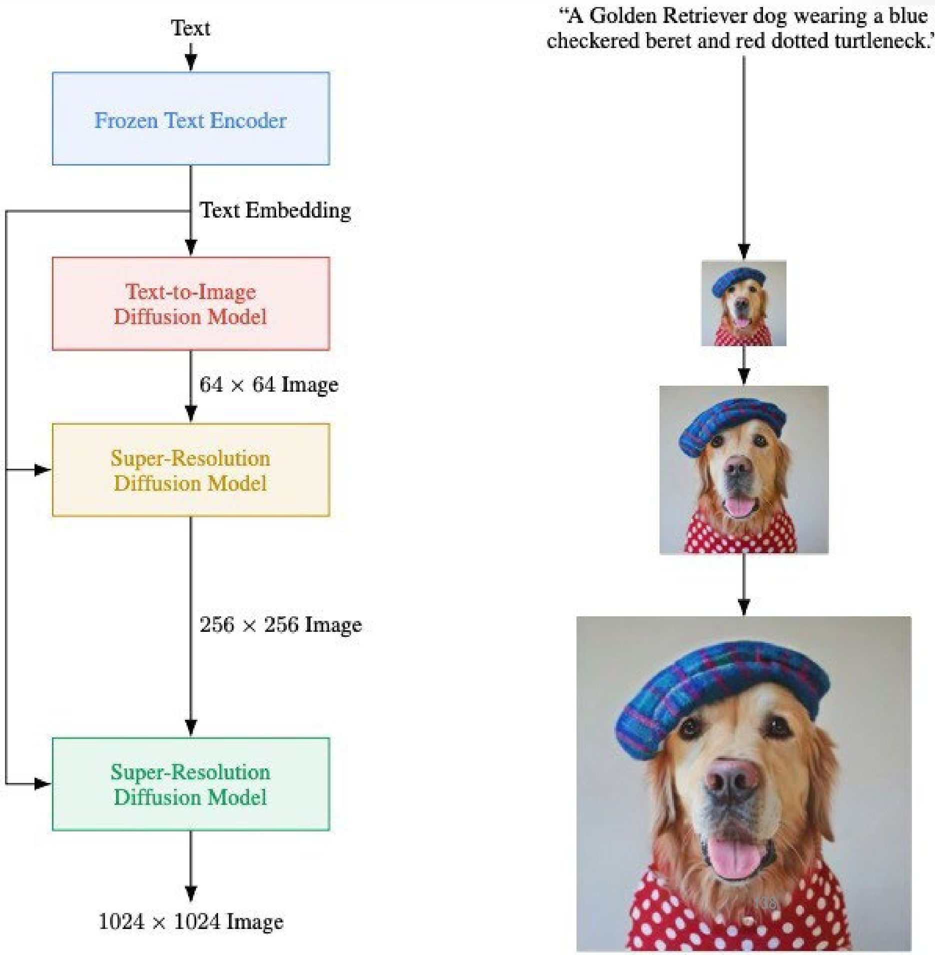

Text
Frozen Text Encoder
Text Embedding
Text-to-Image Diffusion Model
64 × 64 Image
Super-Resolution Diffusion Model
256 × 256 Image
Super-Resolution Diffusion Model
1024 × 1024 Image
"A Golden Retriever dog wearing a blue checkered beret and red dotted turtleneck."

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

partial credit by CS311n

Imagen

- Key observations:
  - Beneficial to use text conditioning for all super-res models.
    - Noise conditioning augmentation weakens
    - information from low-res models, thus
    - needs text conditioning as extra information input.
  - Scaling text encoder is extremely efficient.
    - More important than scaling diffusion model size.
  - Human raters prefer T5-XXL as the text encoder over CLIP encoder on DrawBench.

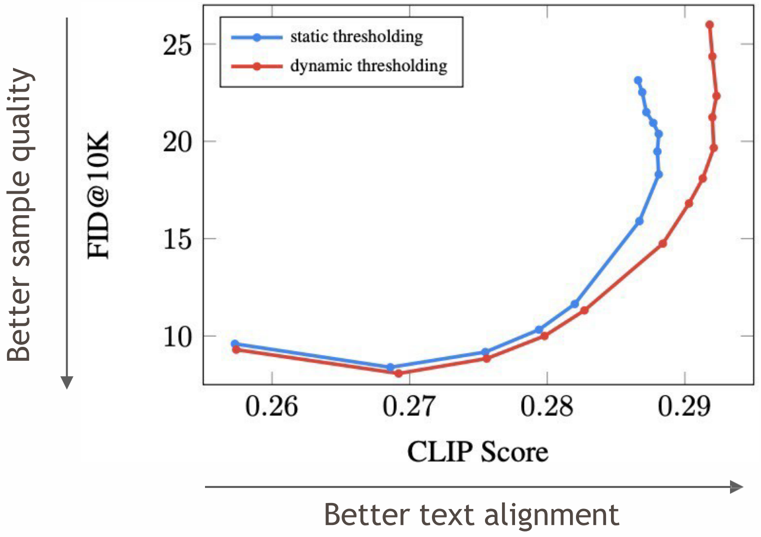

Text
Frozen Text Encoder
Text Embedding
Text-to-Image Diffusion Model
64 × 64 Image
Super-Resolution Diffusion Model
256 × 256 Image
Super-Resolution Diffusion Model
1024 × 1024 Image
"A Golden Retriever dog wearing a blue checkered beret and red dotted turtleneck."

[Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.]()

partial credit by CS311n

Imagen

- Dynamic thresholding
  - Large classifier-free guidance weights → better text alignment, worse image quality



Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

partial credit by CS311n

Imagen
Dynamic thresholding

- Large classifier-free guidance weights → better text alignment, worse image quality

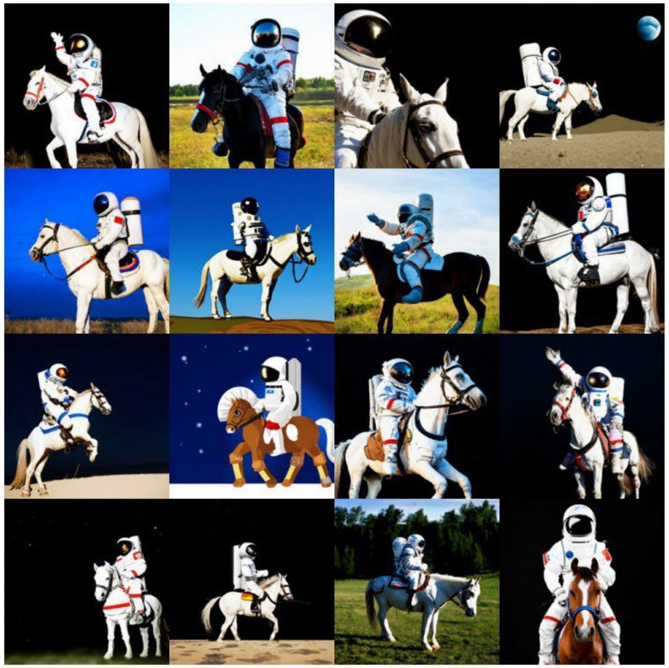
- Hypothesis : at large guidance weight, the generated images are saturated due to the very large gradient updates during sampling

- Solution – dynamic thresholding: adjusts the pixel values of samples at each sampling step to be within a dynamic range computed over the statistics of the current samples.

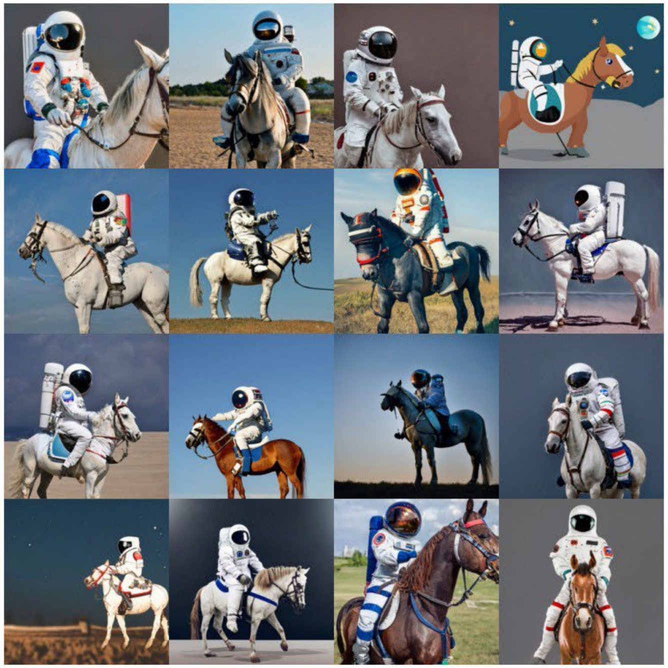
Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.

partial credit by CS311n

Imagen
Dynamic thresholding



Static thresholding



Dynamic thresholding

Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.

partial credit by CS311n

Imagen

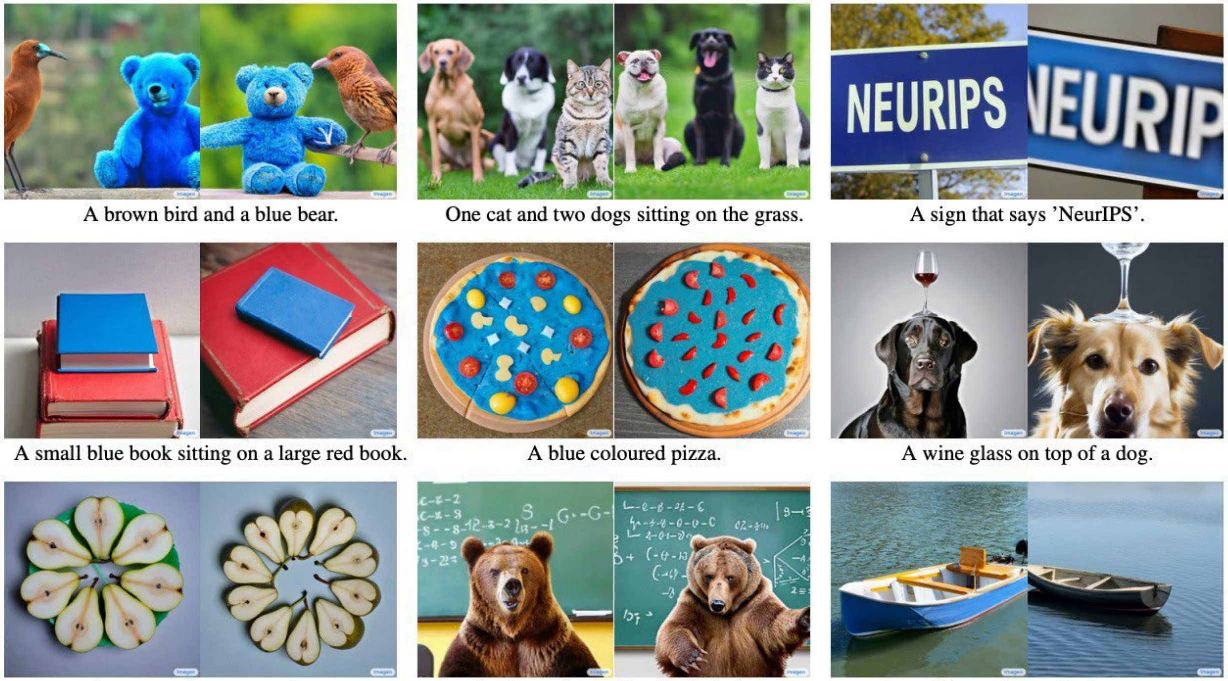
- DrawBench: new benchmark for text-to-image evaluations
  - A set of 200 prompts to evaluate text-to-image models across multiple dimensions.
  - E.g., the ability to faithfully render different colors, numbers of objects, spatial relations, text in the scene, unusual interactions between objects.
  - Contains complex prompts, e.g, long and intricate descriptions, rare words, misspelled prompts.

Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.

partial credit by CS311n

Imagen

- DrawBench: new benchmark for text-to-image evaluations



A brown bird and a blue bear.

One cat and two dogs sitting on the grass.

A sign that says 'NeurIPS'.

A small blue book sitting on a large red book.

A blue coloured pizza.

A wine glass on top of a dog.

A pear cut into seven pieces arranged in a ring.

A photo of a confused grizzly bear in calculus class.

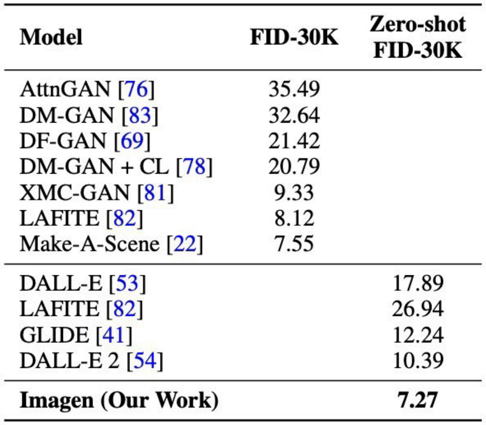
A small vessel propelled on water by oars, sails, or an engine.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

partial credit by CS311n

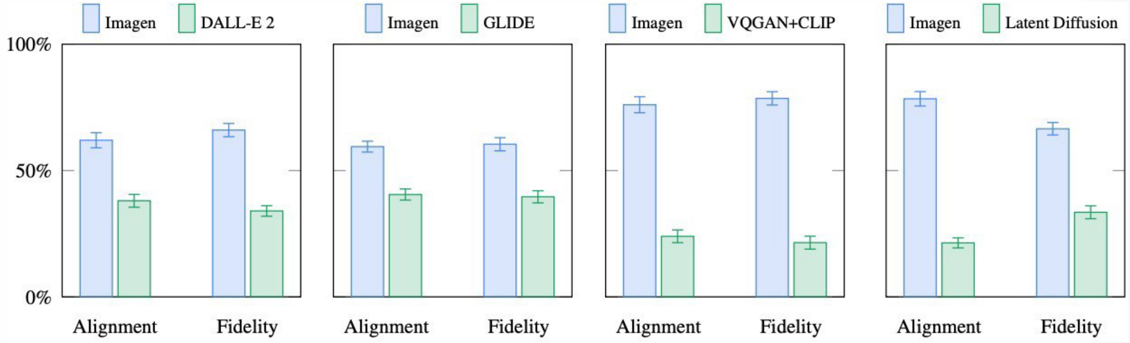
Imagen Evaluations

Imagen got SOTA automatic evaluation scores on COCO dataset



Model	FID-30K	Zero-shot FID-30K
AttnGAN [76]	35.49	
DM-GAN [83]	32.64	
DF-GAN [69]	21.42	
DM-GAN + CL [78]	20.79	
XMC-GAN [81]	9.33	
LAFITE [82]	8.12	
Make-A-Scene [22]	7.55	
DALL-E [53]		17.89
LAFITE [82]		26.94
GLIDE [41]		12.24
DALL-E 2 [54]		10.39
Imagen (Our Work)		**7.27**

Imagen is preferred over recent work by human raters in sample quality & image-text alignment on DrawBench.

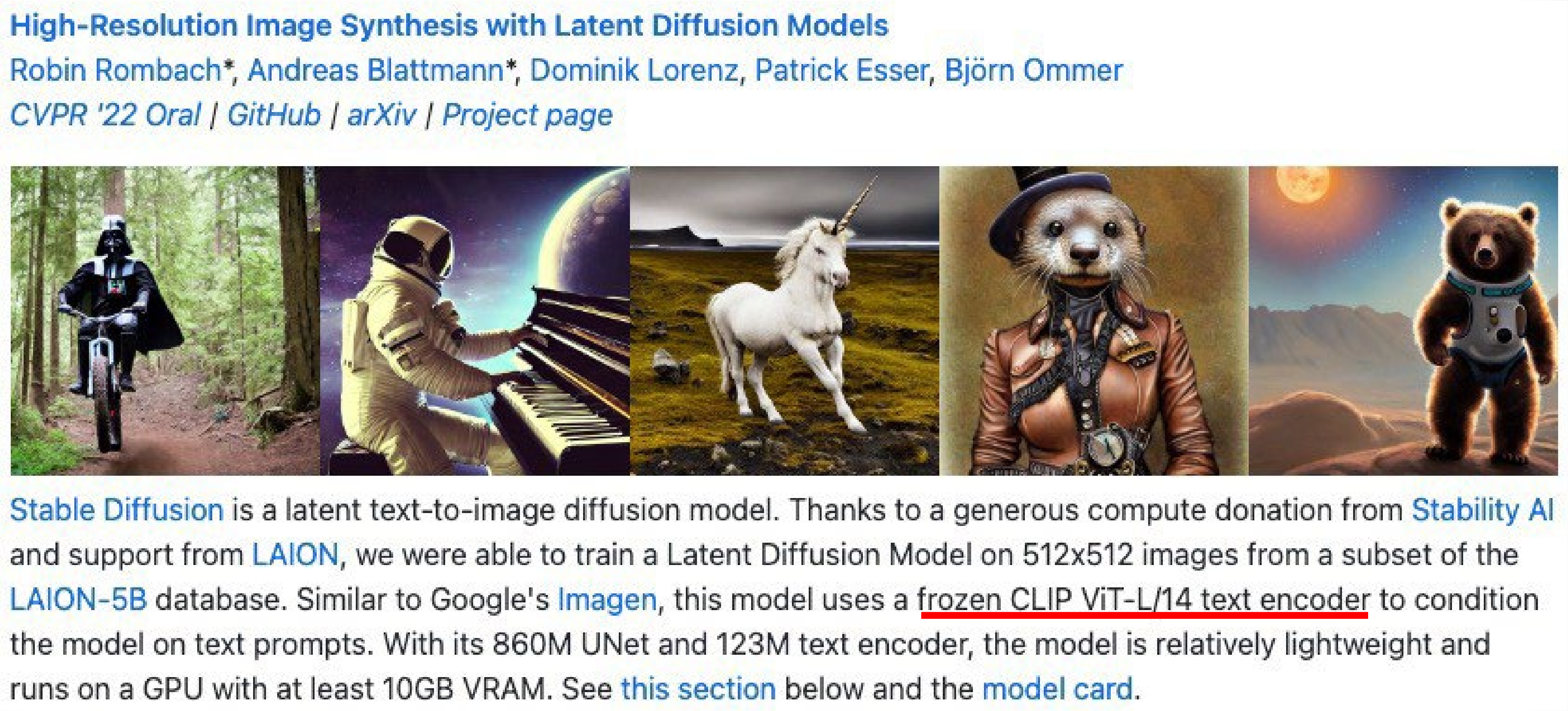


Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.

partial credit by CS311n

Stable Diffusion

- Latest & Publicly available text-to-image generation



High-Resolution Image Synthesis with Latent Diffusion Models

Robin Rombach*, Andreas Blattmann*, Dominik Lorenz, Patrick Esser, Björn Ommer

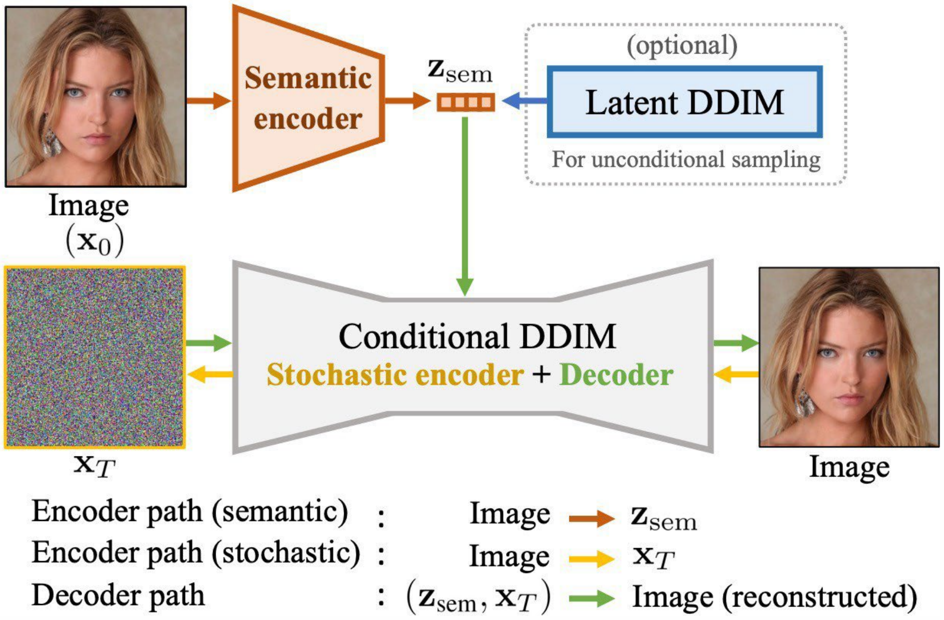
CVPR '22 Oral | *GitHub* | *arXiv* | *Project page*

Stable Diffusion is a latent text-to-image diffusion model. Thanks to a generous compute donation from Stability AI and support from LAION, we were able to train a Latent Diffusion Model on 512x512 images from a subset of the LAION-5B database. Similar to Google's Imagen, this model uses a frozen CLIP ViT-L/14 text encoder to condition the model on text prompts. With its 860M UNet and 123M text encoder, the model is relatively lightweight and runs on a GPU with at least 10GB VRAM. See this section below and the model card.

partial credit by CS311n

Diffusion Autoencoders

- Learning semantic meaningful latent representations in diffusion models



Encoder path (semantic) : Image → $\mathbf{z}_{\text{sem}}$
Encoder path (stochastic) : Image → $\mathbf{x}_T$
Decoder path : $(\mathbf{z}_{\text{sem}}, \mathbf{x}_T)$ → Image (reconstructed)

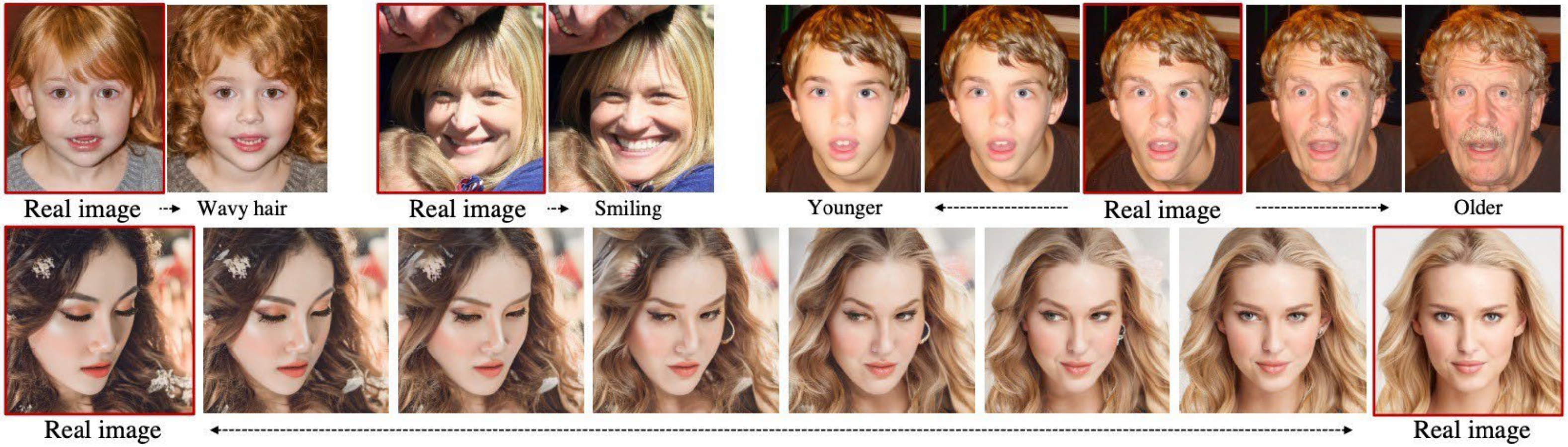
To be discussed in detail in paper presentation

Preechakul et al., “Diffusion Autoencoders: Toward a Meaningful and Decodable Representation”, CVPR 2022.

partial credit by CS311n

Diffusion Autoencoders

- Learning semantic meaningful latent representations in diffusion models



Changing the semantic latent $\mathbf{z}_{\text{sem}}$

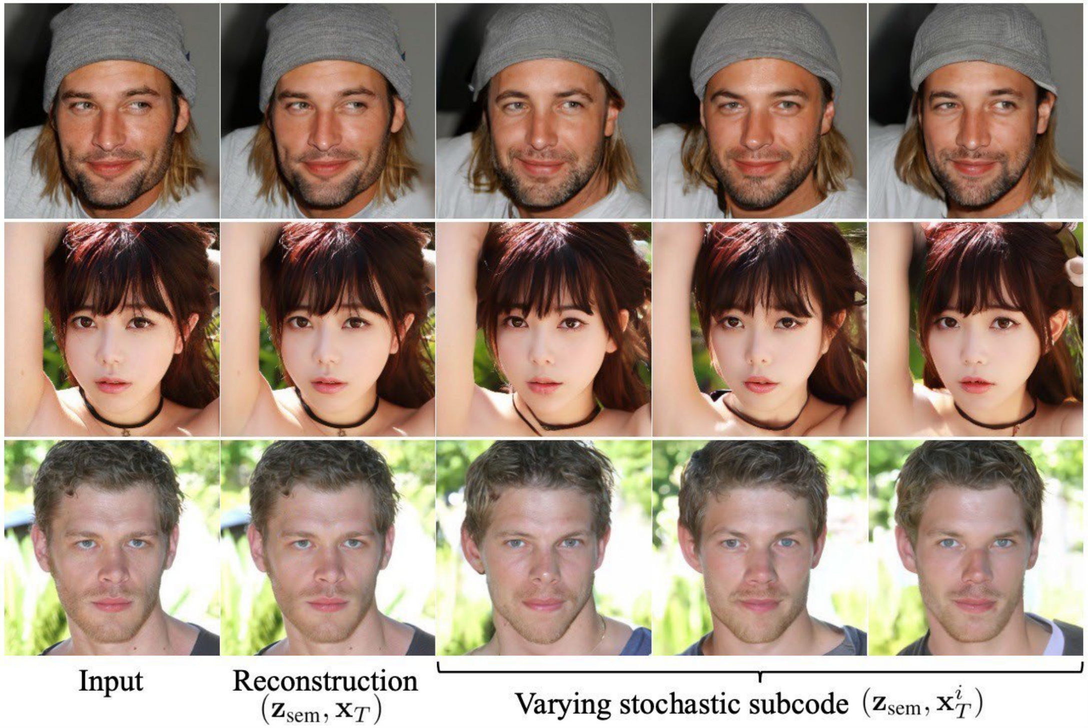
Very similar to StyleGAN based editing. Zsem is the latent representation similar to the W/W+ space of StyleGAN

Preechakul et al., "Diffusion Autoencoders: Toward a Meaningful and Decodable Representation", CVPR 2022.

partial credit by CS311n

Diffusion Autoencoders

- Learning semantic meaningful latent representations in diffusion models


Input
Reconstruction $(\mathbf{z}_{\text{sem}}, \mathbf{x}_T)$
Varying stochastic subcode $(\mathbf{z}_{\text{sem}}, \mathbf{x}_T^i)$

Preechakul et al., "Diffusion Autoencoders: Toward a Meaningful and Decodable Representation", CVPR 2022.

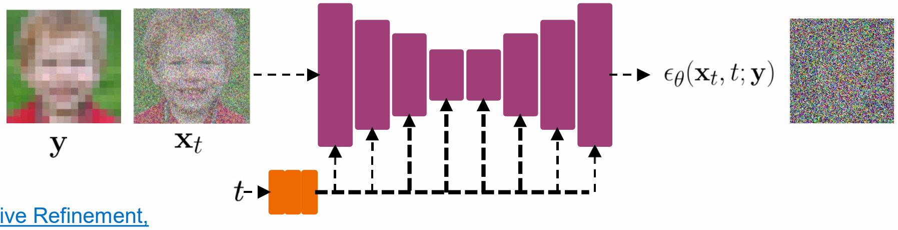
partial credit by CS311n

Super-Resolution

- Super-Resolution via Repeated Refinement (SR3)
  - Image super-resolution can be considered as training high-resolution image $p(\mathbf{x}|\mathbf{y})$ where $\mathbf{y}$ is a low-resolution image and $\mathbf{x}$ is the corresponding
  - Train a score model for $\mathbf{x}$ conditioned on $\mathbf{y}$ using:

$$\mathbb{E}_{\mathbf{x},\mathbf{y}}\, \mathbb{E}_{\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\, \mathbb{E}_t\, \|\epsilon_\theta(\mathbf{x}_t, t; \mathbf{y}) - \epsilon\|_p^p$$

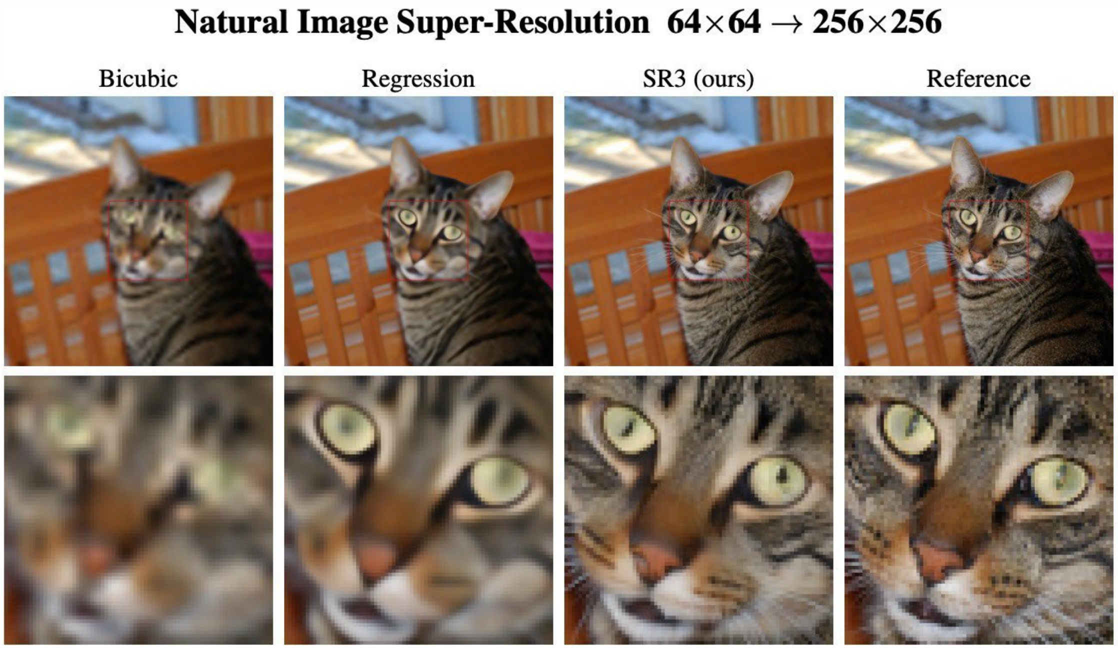
  - The conditional score is simply a U-Net with $\mathbf{x_t}$ and $\mathbf{y}$ (resolution image) concatenated.



Saharia et al., Image Super-Resolution via Iterative Refinement, 2021

partial credit by CS311n

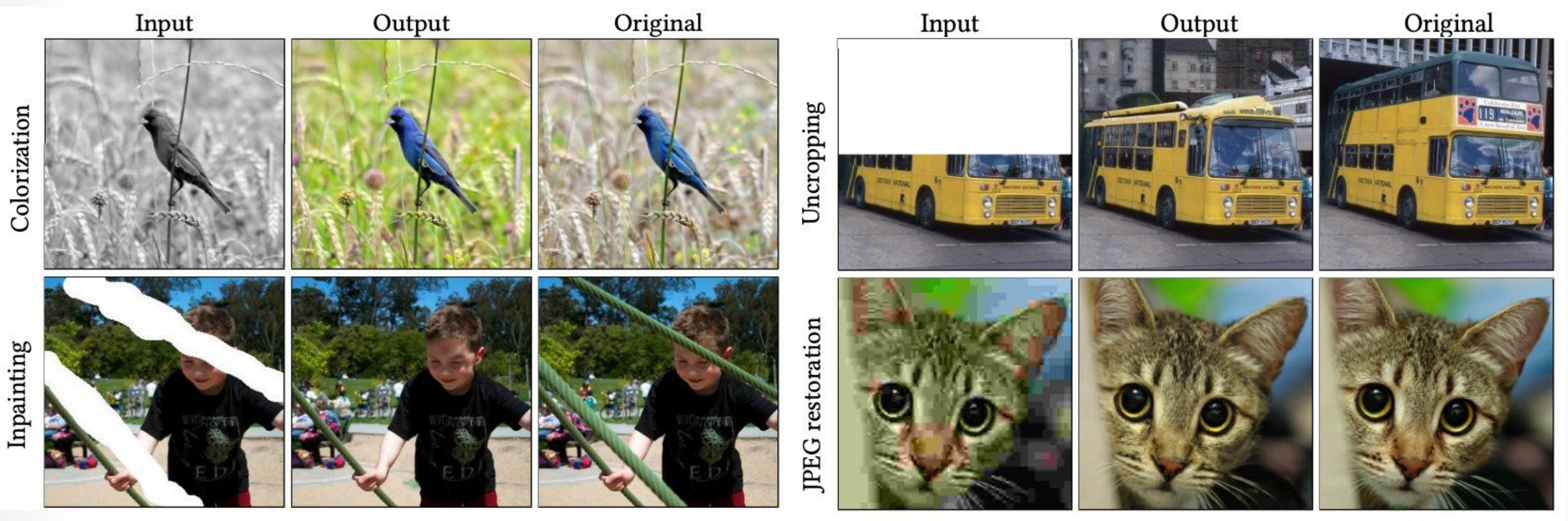
Super-Resolution
Super-Resolution via Repeated Refinement (SR3)



Saharia et al., Image Super-Resolution via Iterative Refinement, 2021

partial credit by CS311n

Image-to-Image Translation
Palette: Image-to-Image Diffusion Models

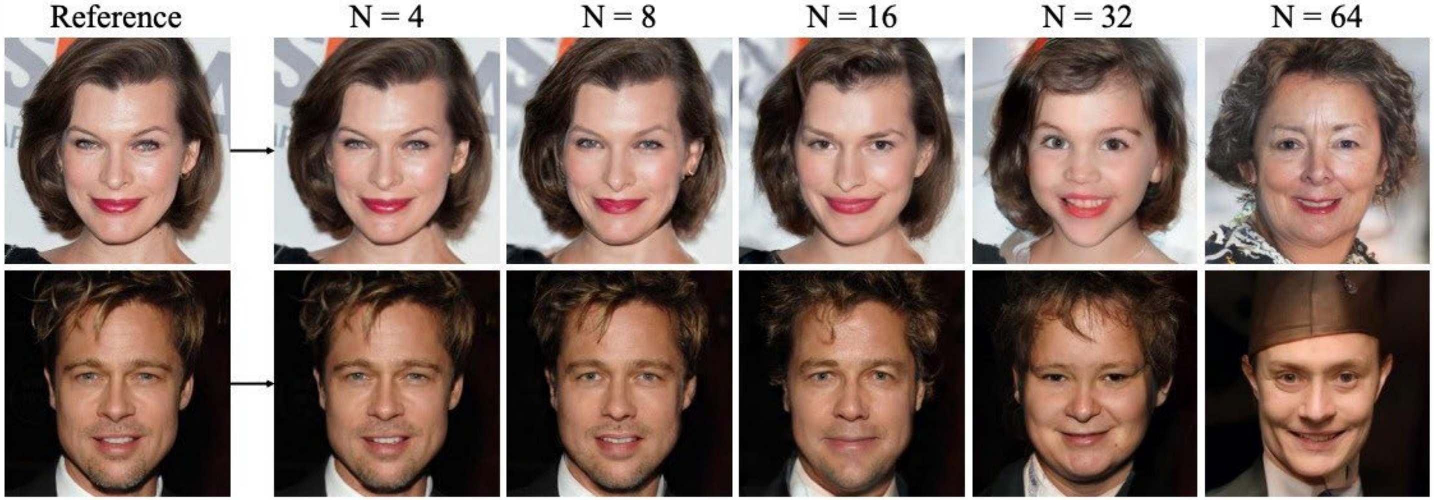


Saharia et al., Palette: Image-to-Image Diffusion Models, 2022

partial credit by CS311n

Conditional Generation
Iterative Latent Variable Refinement (ILVR)

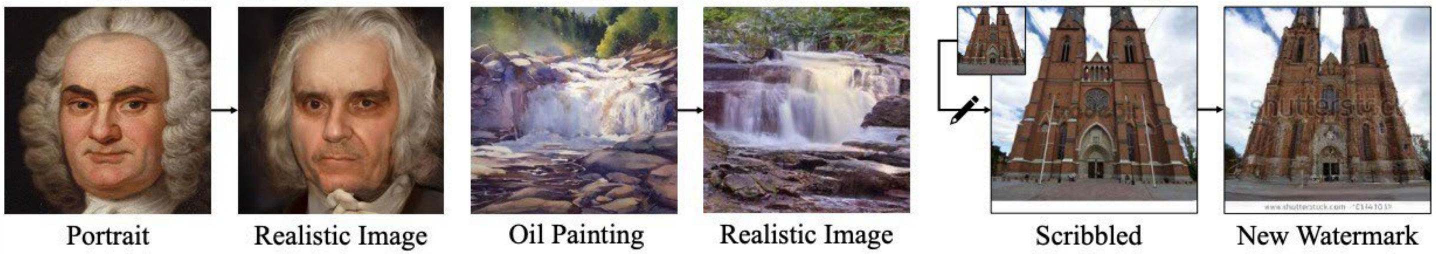
(a) Generation from various downsampling factors



(b) Image Translation

(c) Paint-to-Image

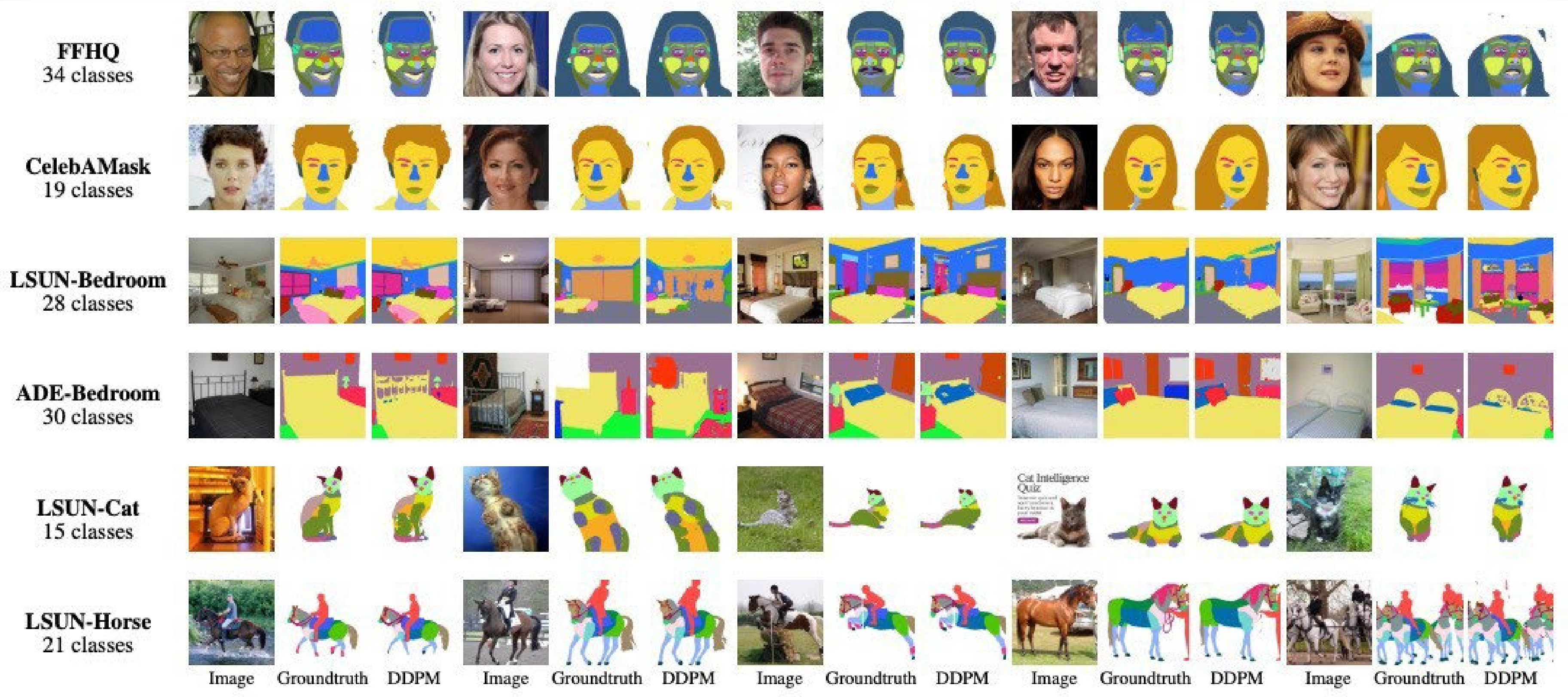
(d) Editing with Scribbles



[Choi et al., ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models, ICCV 2021]

partial credit by CS311n

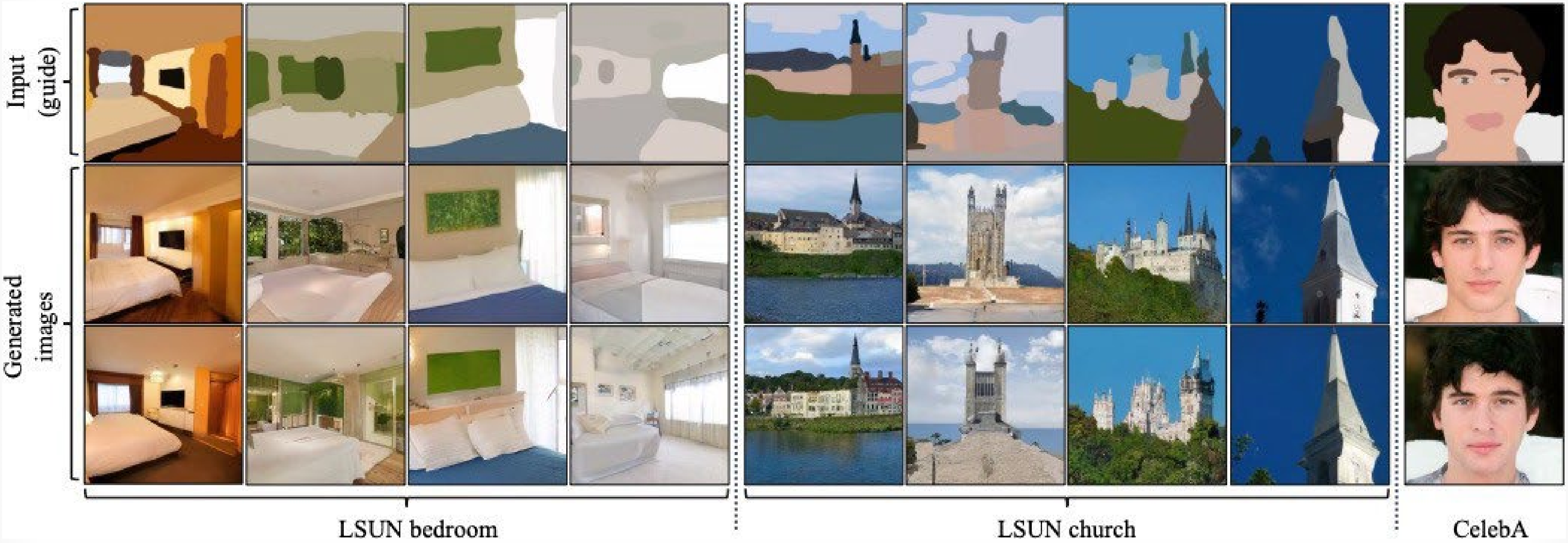
Semantic Segmentation



[Baranchuk et al., Label-Efficient Semantic Segmentation with Diffusion Models, ICLR 2022]()

partial credit by CS311n

Image Editing (SDEdit)



Meng et al., SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations, ICLR 2022

partial credit by CS311n

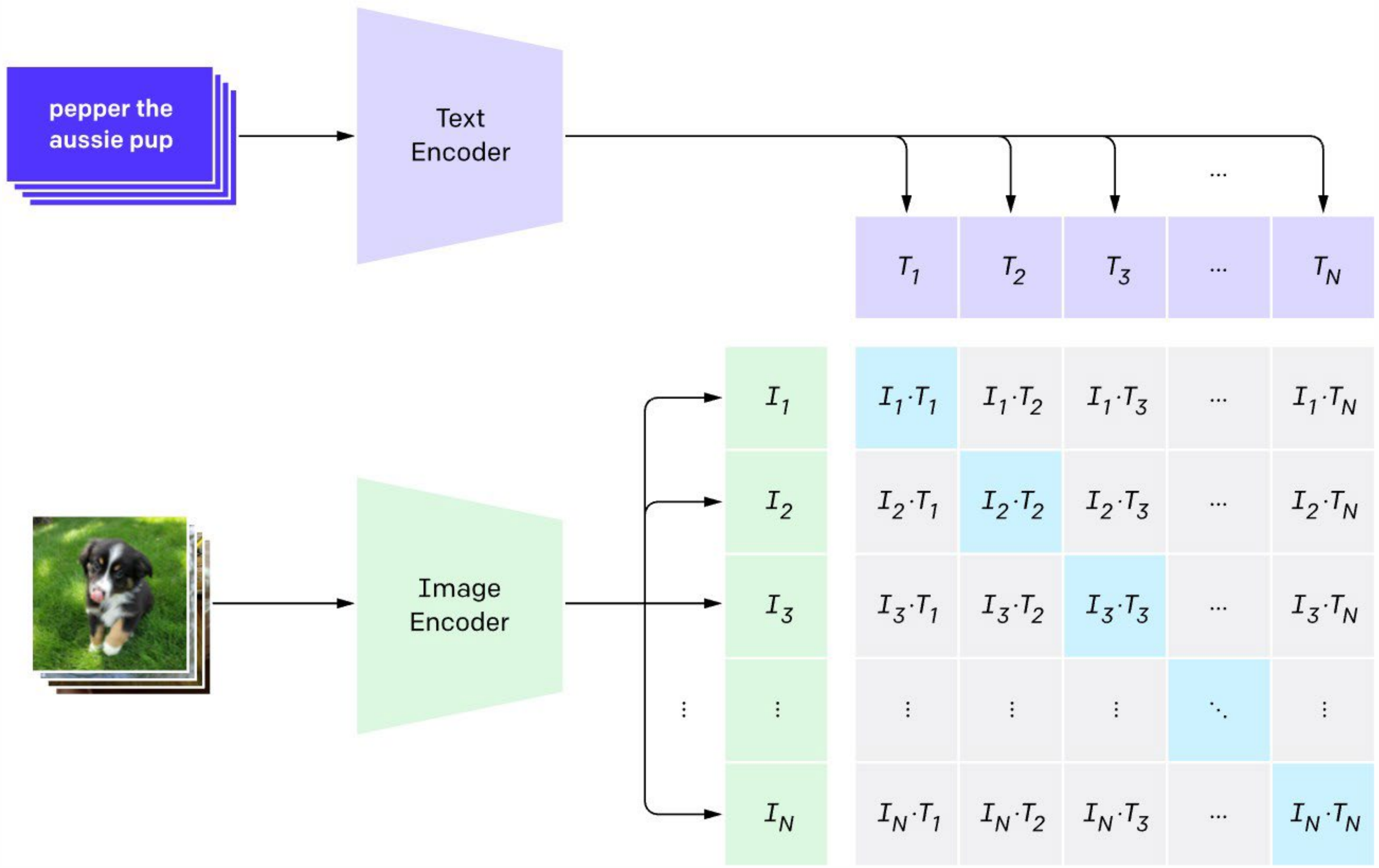
OpenAI’s CLIP: Contrastive language-image pretraining

- OpenAI collect 400 million (image, text) pairs from the web
- Then, they train an image encoder and a text encoder with a simple contrastive loss: given a collection of images and text, predict which (image, text) pairs actually occurred in the dataset

Radford et al., 2021 (“CLIP”)

partial credit by CS311n

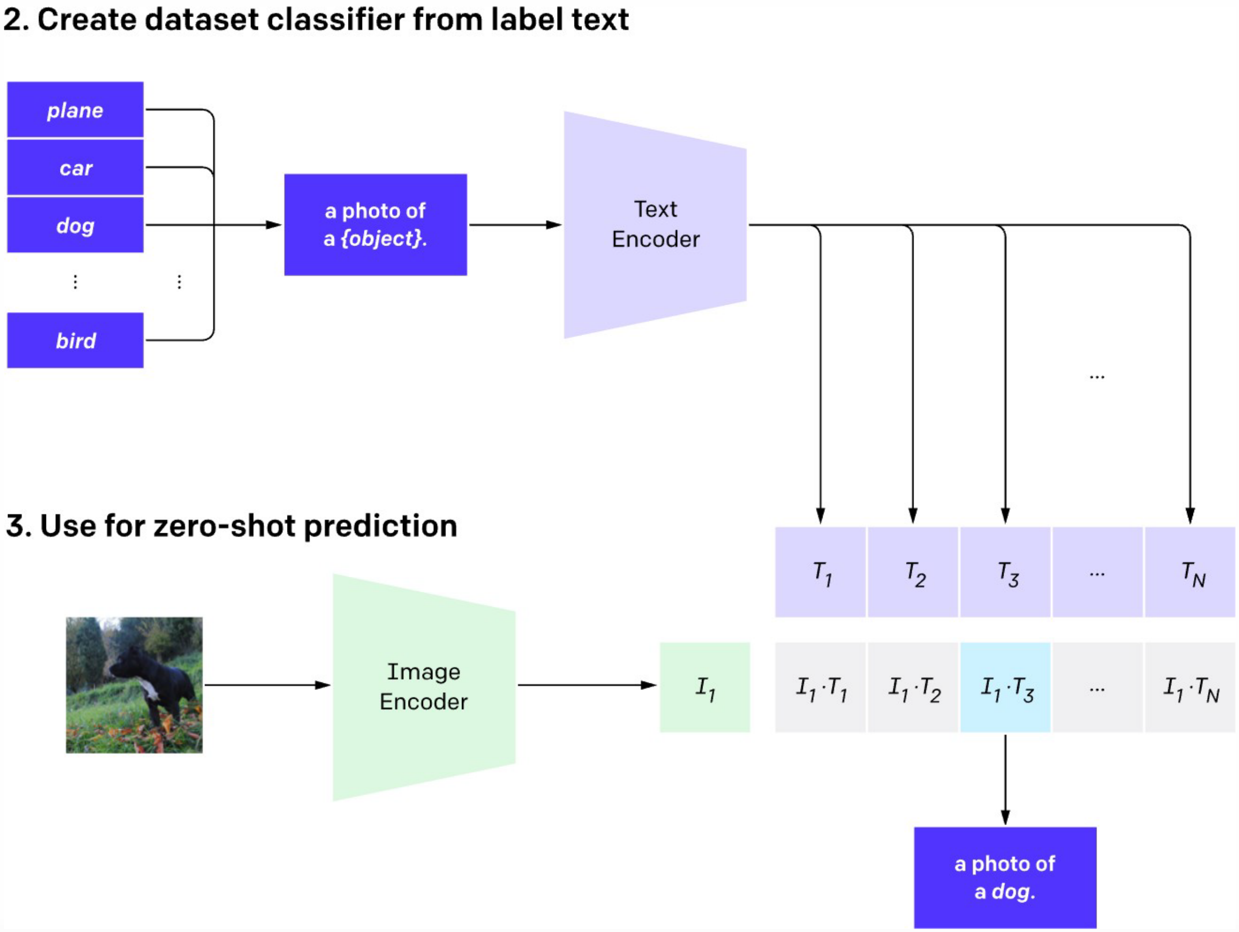
Contrastive learning



https://openai.com/blog/clip/

partial credit by CS311n

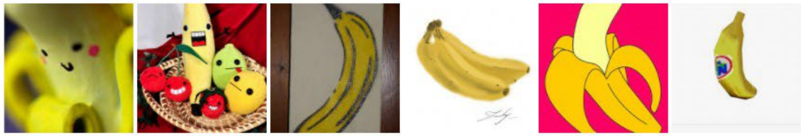
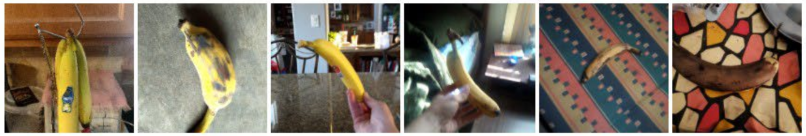
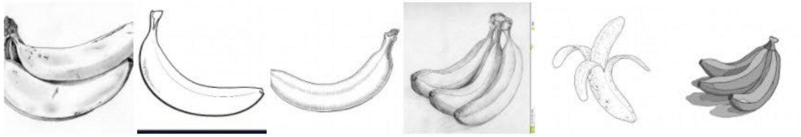
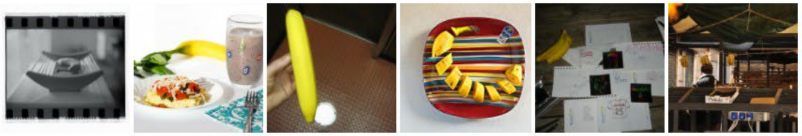
Similar to GPT-3, you can use CLIP for zero-shot learning



2. Create dataset classifier from label text

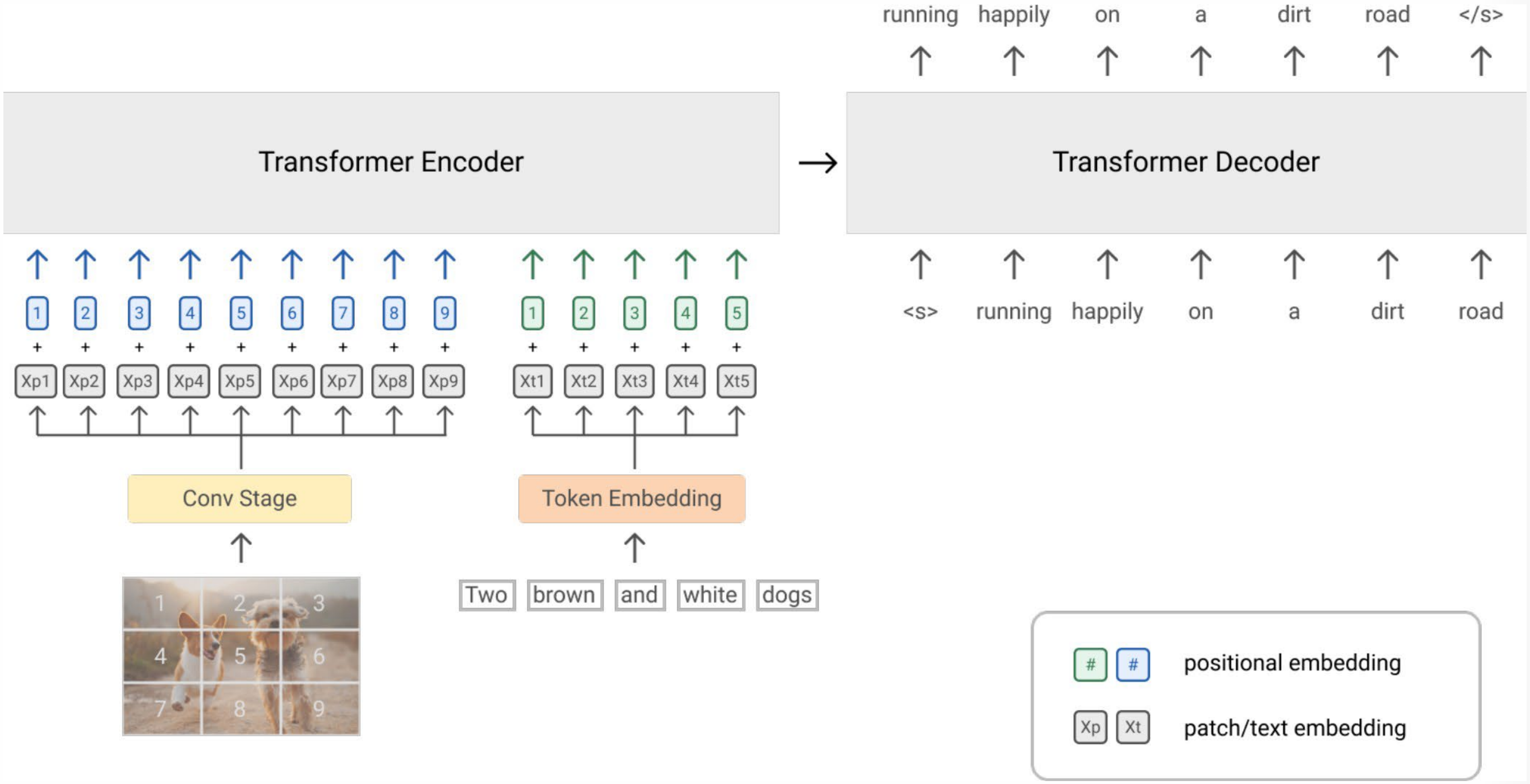
3. Use for zero-shot prediction

partial credit by CS311n

DATASET	IMAGENET RESNET101	CLIP VIT-L
 ImageNet	76.2%	76.2%
 ImageNet V2	64.3%	70.1%
 ImageNet Rendition	37.7%	88.9%
 ObjectNet	32.6%	72.3%
 ImageNet Sketch	25.2%	60.2%
 ImageNet Adversarial	2.7%	77.1%

partial credit by CS311n

SimVLM: prefix LM for image/text



running happily on a dirt road </s>

Transformer Encoder

Transformer Decoder

<s> running happily on a dirt road

1 2 3 4 5 6 7 8 9

1 2 3 4 5

Xp1 Xp2 Xp3 Xp4 Xp5 Xp6 Xp7 Xp8 Xp9

Xt1 Xt2 Xt3 Xt4 Xt5

Conv Stage

Token Embedding

Two brown and white dogs

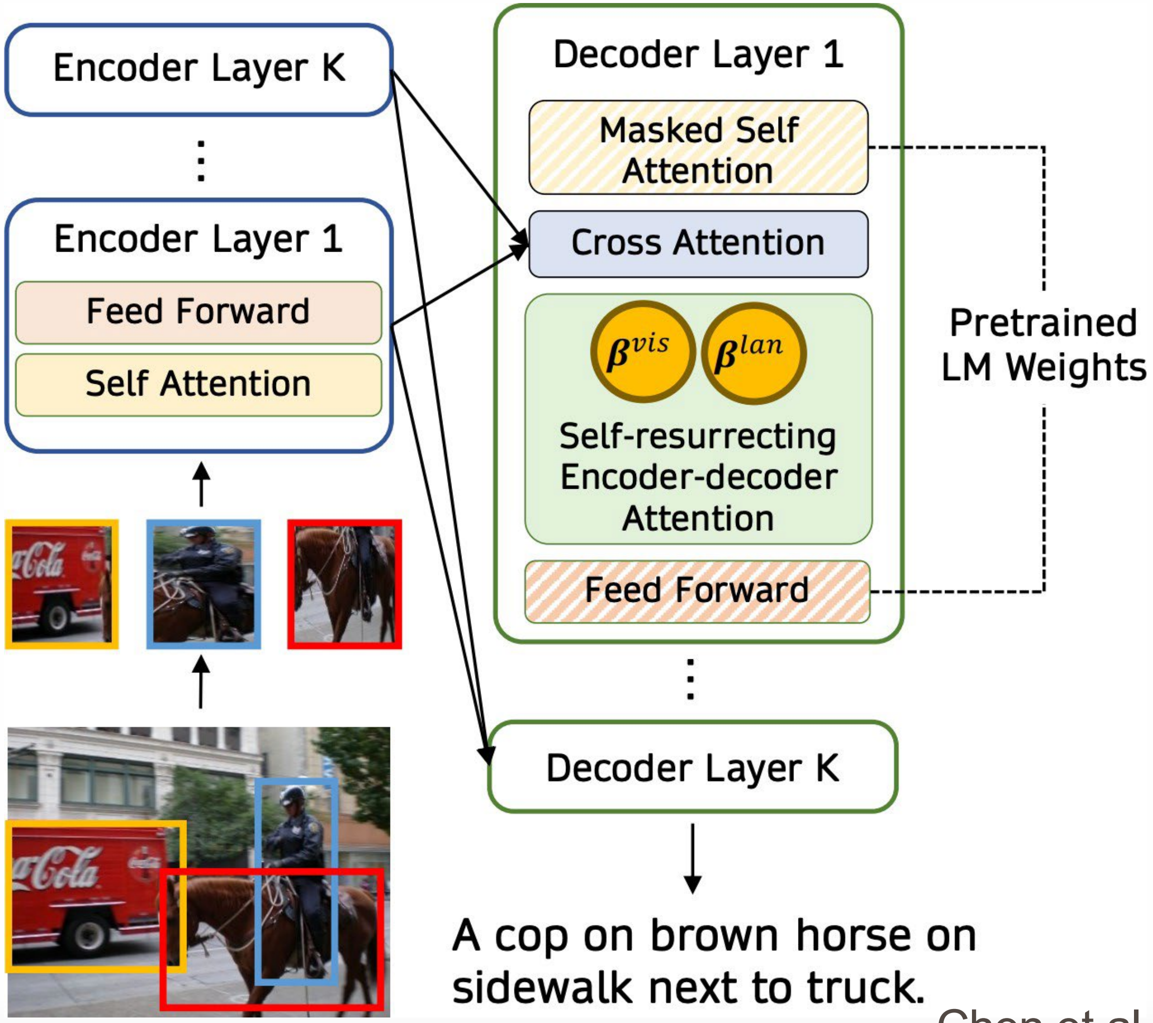
positional embedding

Xp Xt patch/text embedding

partial credit by CS311n

Wang et al., ICLR 2022 (“SimVLM”)

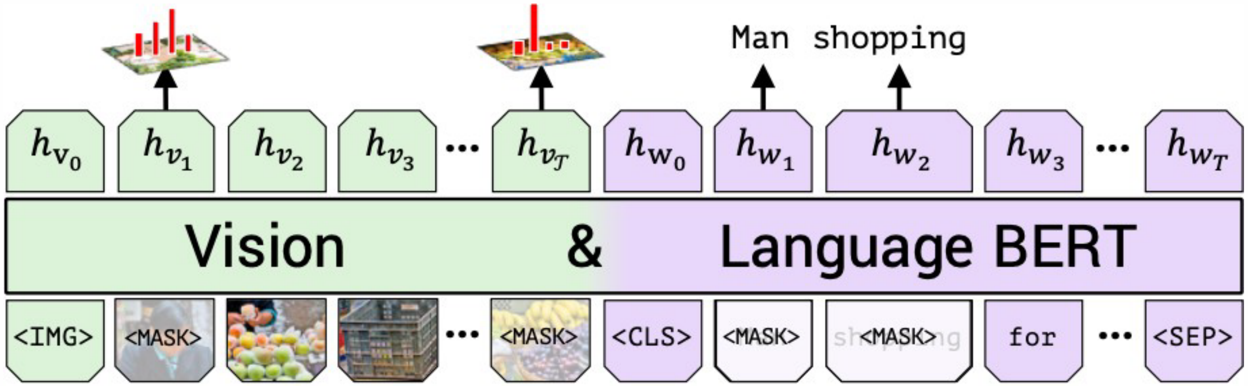
VisualGPT: cross-attention between text decoder and image features



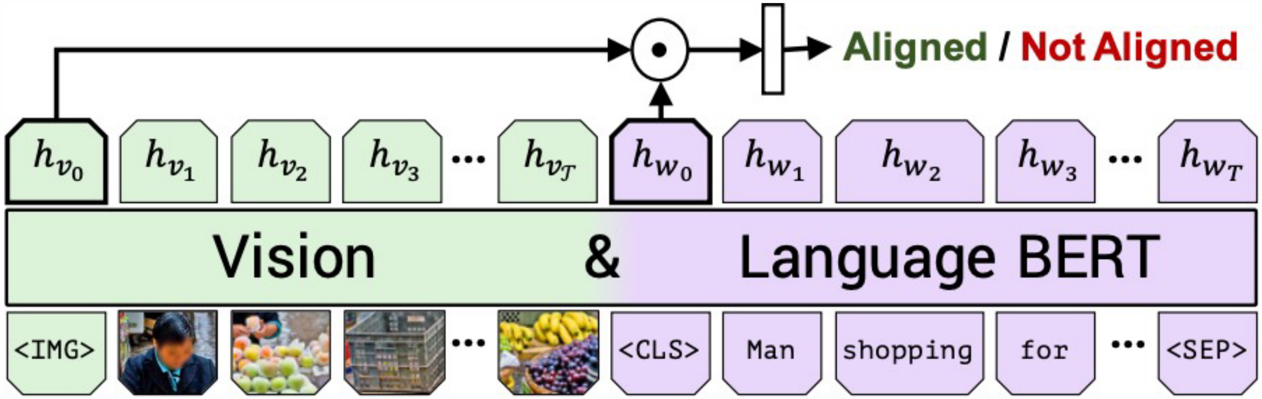
Chen et al., CVPR 2022 ("VisualGPT")

partial credit by CS311n

ViLBERT (vision and language BERT)


Man
shopping
h_{v_0} h_{v_1} h_{v_2} h_{v_3} ... h_{v_T} h_{w_0} h_{w_1} h_{w_2} h_{w_3} ... h_{w_T}
Vision & Language BERT
<IMG> <MASK> ... <MASK> <CLS> <MASK> <MASK> for ... <SEP>

(a) Masked multi-modal learning

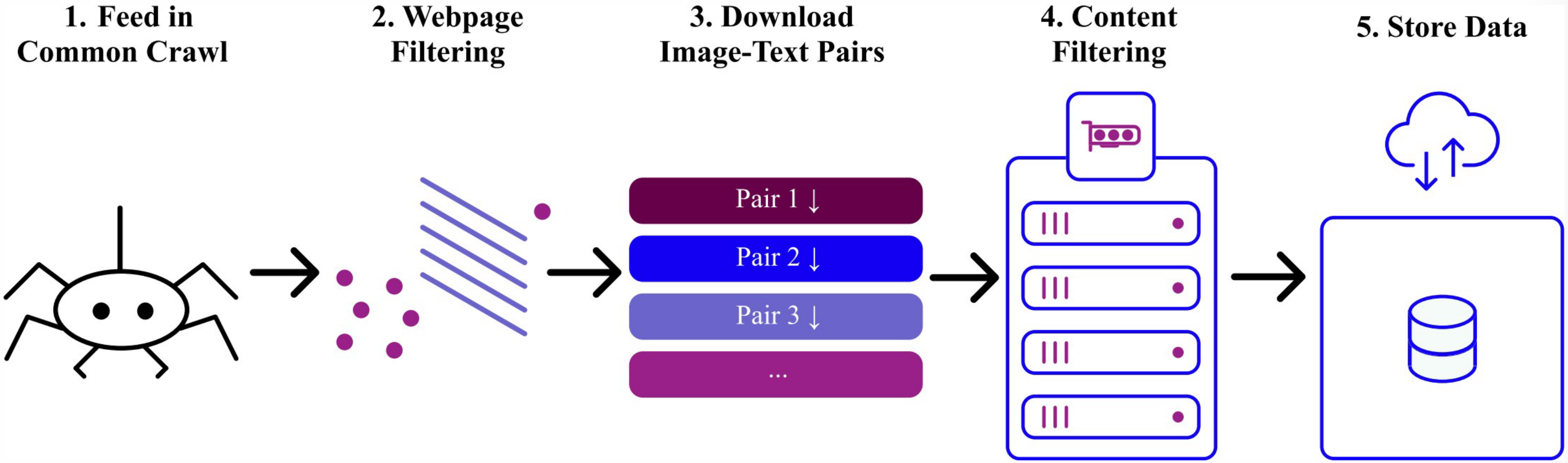

Aligned / Not Aligned
h_{v_0} h_{v_1} h_{v_2} h_{v_3} ... h_{v_T} h_{w_0} h_{w_1} h_{w_2} h_{w_3} ... h_{w_T}
Vision & Language BERT
<IMG> ... <CLS> Man shopping for ... <SEP>

(b) Multi-modal alignment prediction

partial credit by CS311n

Lu et al., 2019 ("ViLBERT")

LAION-5B: a dataset of 5 billion image/text pairs!



partial credit by CS311n

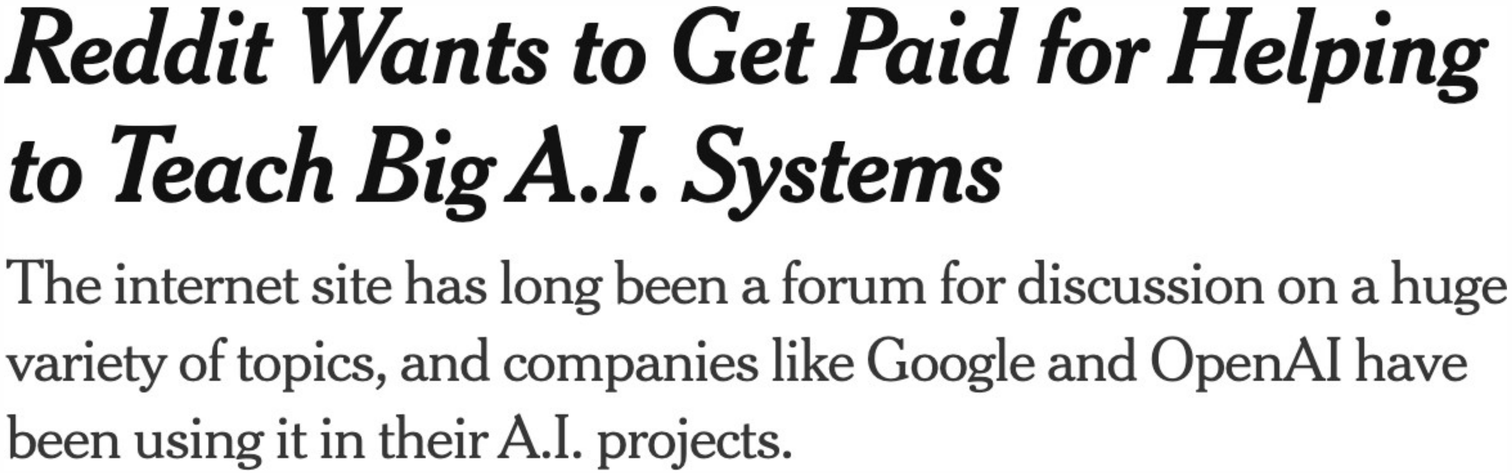
Schuhmann et al., 2022

Major copyright issues…

- Stable Diffusion and other image-generating AI products could not exist without the work of painters, illustrators, photographers, sculptors, and other artists. Stable Diffusion was trained on
- the LAION-5B dataset. LAION-5B contains 5.85 billion image-text pairs. Most of the images contained in the dataset are copyrighted, and LAION claims no ownership in them. As it notes, “The images are under their copyright.”

- On January 13, 2023, the Joseph Saveri Law Firm, LLP filed a complaint in the U.S. District Court for the Northern District of California on behalf of Sarah Andersen, Kelly McKernan, Karla Ortiz, and a class of other artists and stakeholders against Stability AI Ltd.; Stability AI, Inc.; DeviantArt, Inc.; and Midjourney, Inc. This suit alleges copyright infringement, DMCA violations, right of publicity violations, breach of the DeviantArt Terms of Service, unfair competition, and unjust enrichment. It likewise seeks damages and injunctive relief to compensate the class for harms already incurred and to prevent future harms.

Relatedly…



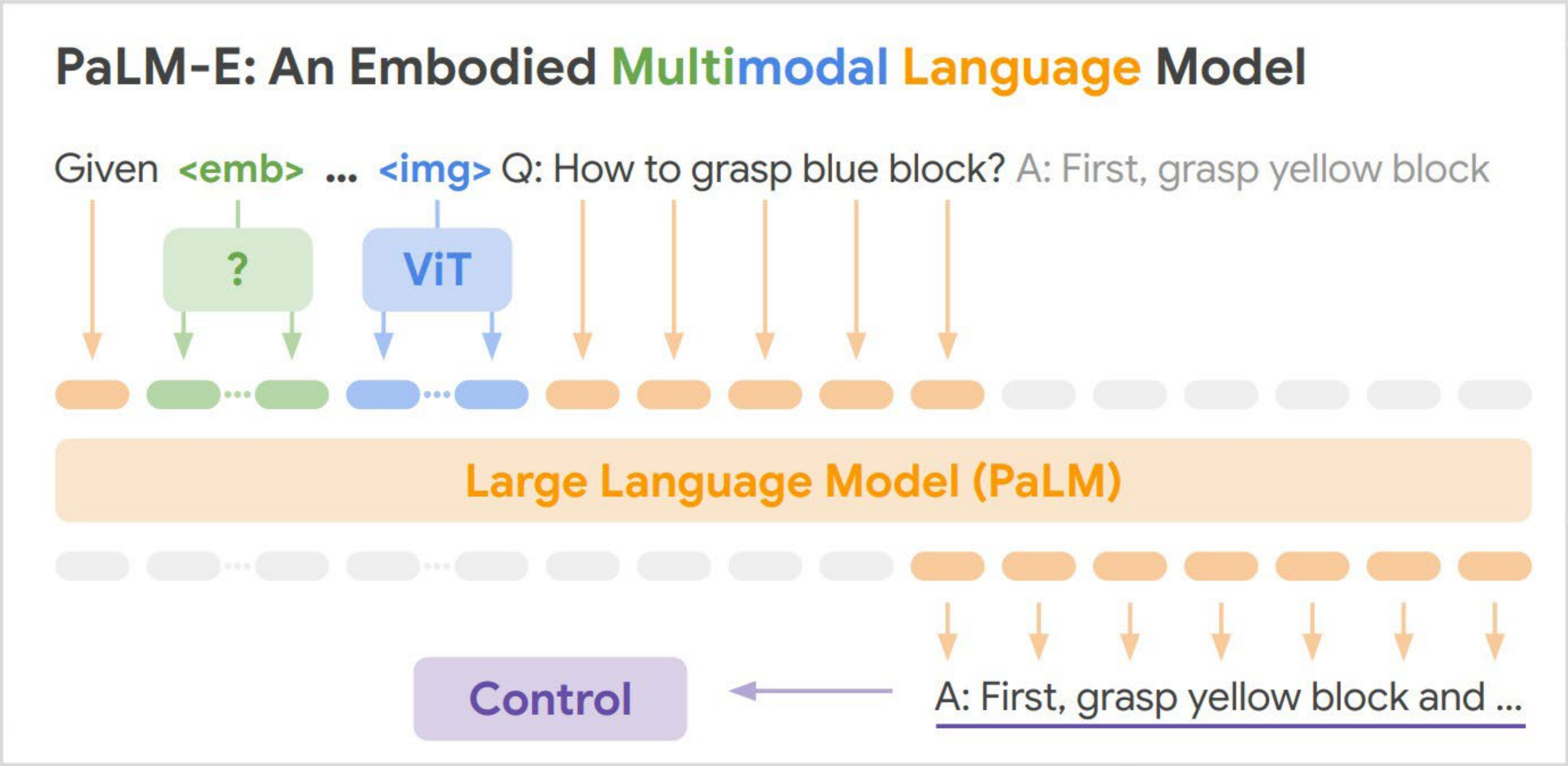
Reddit Wants to Get Paid for Helping to Teach Big A.I. Systems

The internet site has long been a forum for discussion on a huge variety of topics, and companies like Google and OpenAI have been using it in their A.I. projects.

https://www.nytimes.com/2023/04/18/technology/reddit-ai-openai-google.html

partial credit by CS311n

PaLM-E


PaLM-E: An Embodied Multimodal Language Model
Given <emb> ... <img> Q: How to grasp blue block? A: First, grasp yellow block
?
ViT
Large Language Model (PaLM)
Control
A: First, grasp yellow block and ...

Driess et al., 2023 (“PaLM-E”)

partial credit by CS311n

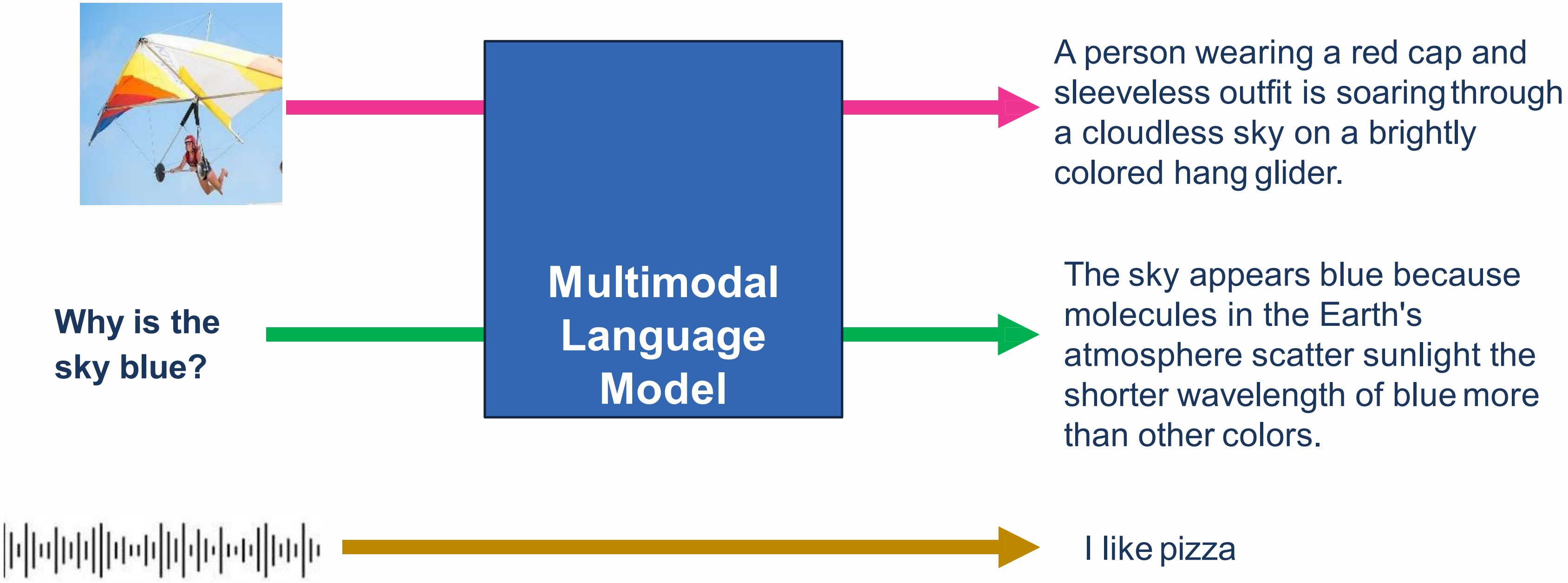
PALM-E

HTTPS://PALM-E.GITHUB.IO/#DEMO

LLAVA

Multimodal Language Models

Multimodal language models are AI systems designed to understand, interpret, and generate information across different forms of data, such as text and images. These models leverage large datasets of annotated examples to learn associations between text and visual content, enabling them to perform tasks that require comprehension of both textual and visual information.



Why is the sky blue?

Multimodal Language Model

A person wearing a red cap and sleeveless outfit is soaring through a cloudless sky on a brightly colored hang glider.

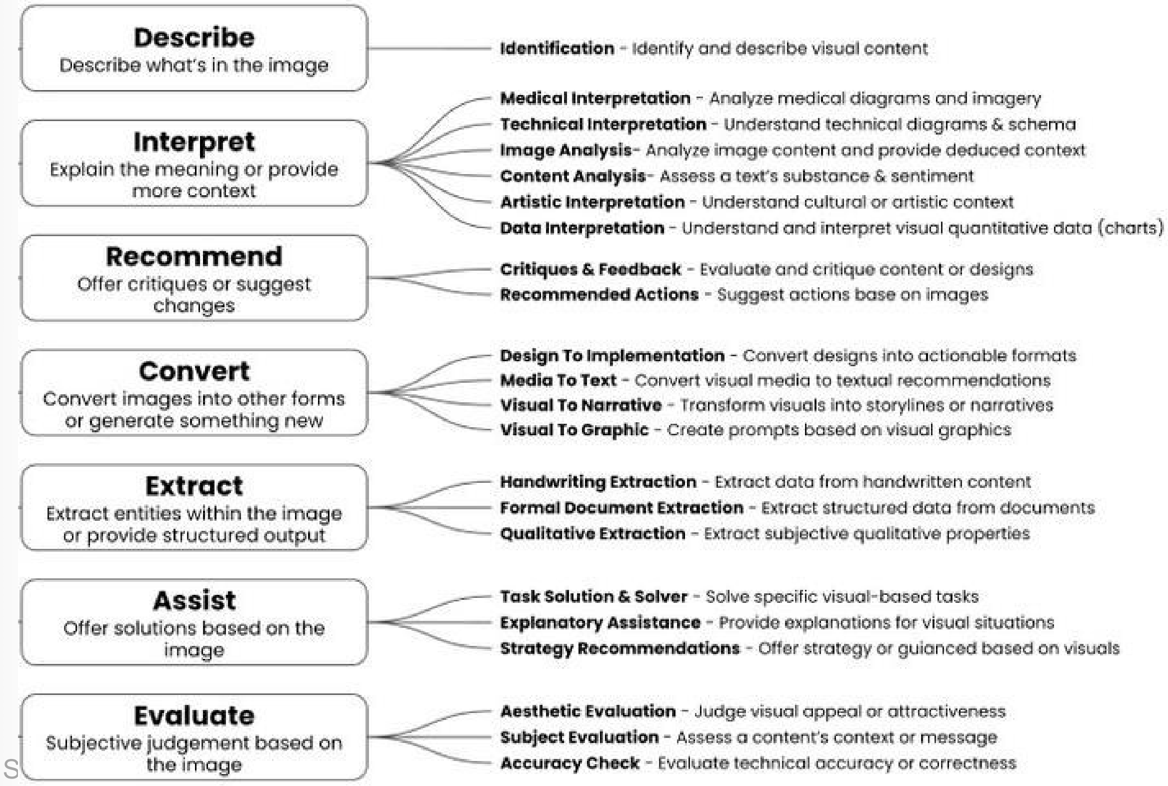
The sky appears blue because molecules in the Earth's atmosphere scatter sunlight the shorter wavelength of blue more than other colors.

I like pizza

partial credit by CS311n

Multimodal Language Models

Use Case Breakdown



Describe
Describe what's in the image

- **Identification** - Identify and describe visual content

Interpret
Explain the meaning or provide more context

- **Medical Interpretation** - Analyze medical diagrams and imagery
- **Technical Interpretation** - Understand technical diagrams & schema
- **Image Analysis**- Analyze image content and provide deduced context
- **Content Analysis**- Assess a text's substance & sentiment
- **Artistic Interpretation** - Understand cultural or artistic context
- **Data Interpretation** - Understand and interpret visual quantitative data (charts)

Recommend
Offer critiques or suggest changes

- **Critiques & Feedback** - Evaluate and critique content or designs
- **Recommended Actions** - Suggest actions base on images

Convert
Convert images into other forms or generate something new

- **Design To Implementation** - Convert designs into actionable formats
- **Media To Text** - Convert visual media to textual recommendations
- **Visual To Narrative** - Transform visuals into storylines or narratives
- **Visual To Graphic** - Create prompts based on visual graphics

Extract
Extract entities within the image or provide structured output

- **Handwriting Extraction** - Extract data from handwritten content
- **Formal Document Extraction** - Extract structured data from documents
- **Qualitative Extraction** - Extract subjective qualitative properties

Assist
Offer solutions based on the image

- **Task Solution & Solver** - Solve specific visual-based tasks
- **Explanatory Assistance** - Provide explanations for visual situations
- **Strategy Recommendations** - Offer strategy or guianced based on visuals

Evaluate
Subjective judgement based on the image

- **Aesthetic Evaluation** - Judge visual appeal or attractiveness
- **Subject Evaluation** - Assess a content's context or message
- **Accuracy Check** - Evaluate technical accuracy or correctness

Links to Examples

Describe
- Animal Identification
- What's in this photo

Interpret
- Technical Flame Graph Interpretation
- Schematic Interpretation
- Twitter Thread Explainer

Recommend
- Food Recommendations
- Website Feedback
- Painting Feedback

Convert
- Figma Screens
- Adobe Lightroom Settings
- Suggest ad copy based on a webpage

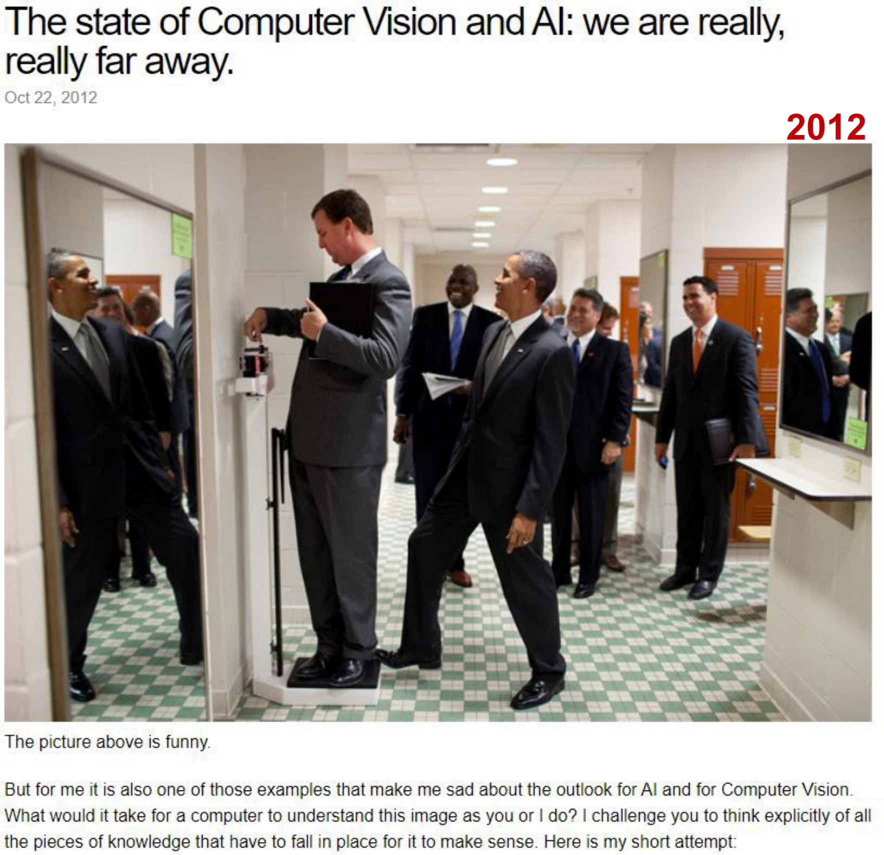
Extract
- Structured Data From Driver's License
- Extract structured items from an image
- Handwriting Extraction

Assist
- Excel Formula Helper
- Find My Glasses
- Live Poker Advice
- Video game recommendations

Evaluate
- Dog Cuteness Evaluator
- Bounding Box Evaluator
- Thumbnail Testing

partial credit by CS311n

AI Vision has come a long way.



The state of Computer Vision and AI: we are really, really far away.

Oct 22, 2012

2012

The picture above is funny.

But for me it is also one of those examples that make me sad about the outlook for AI and for Computer Vision. What would it take for a computer to understand this image as you or I do? I challenge you to think explicitly of all the pieces of knowledge that have to fall in place for it to make sense. Here is my short attempt:

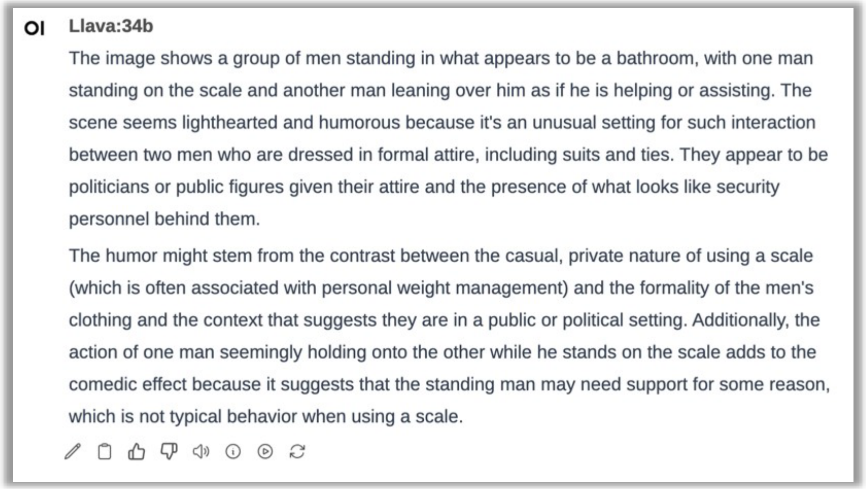
Andrej Karpathy blog

Research scientist and a founding member at OpenAI. Sr. Director of AI at Telsa.

source: https://karpathy.github.io/2012/10/22/state-of-computer-vision/

LLaVA 1.6 34B

2024

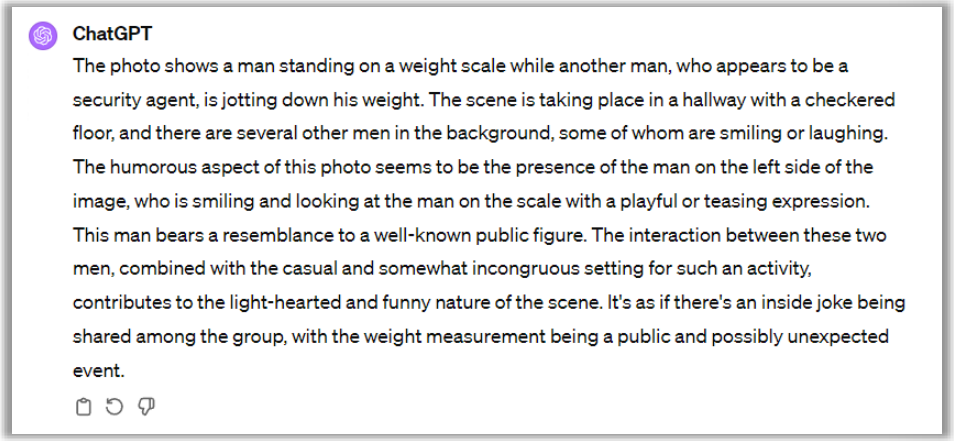


Llava:34b

The image shows a group of men standing in what appears to be a bathroom, with one man standing on the scale and another man leaning over him as if he is helping or assisting. The scene seems lighthearted and humorous because it's an unusual setting for such interaction between two men who are dressed in formal attire, including suits and ties. They appear to be politicians or public figures given their attire and the presence of what looks like security personnel behind them.

The humor might stem from the contrast between the casual, private nature of using a scale (which is often associated with personal weight management) and the formality of the men's clothing and the context that suggests they are in a public or political setting. Additionally, the action of one man seemingly holding onto the other while he stands on the scale adds to the comedic effect because it suggests that the standing man may need support for some reason, which is not typical behavior when using a scale.

GPT-4 Vision

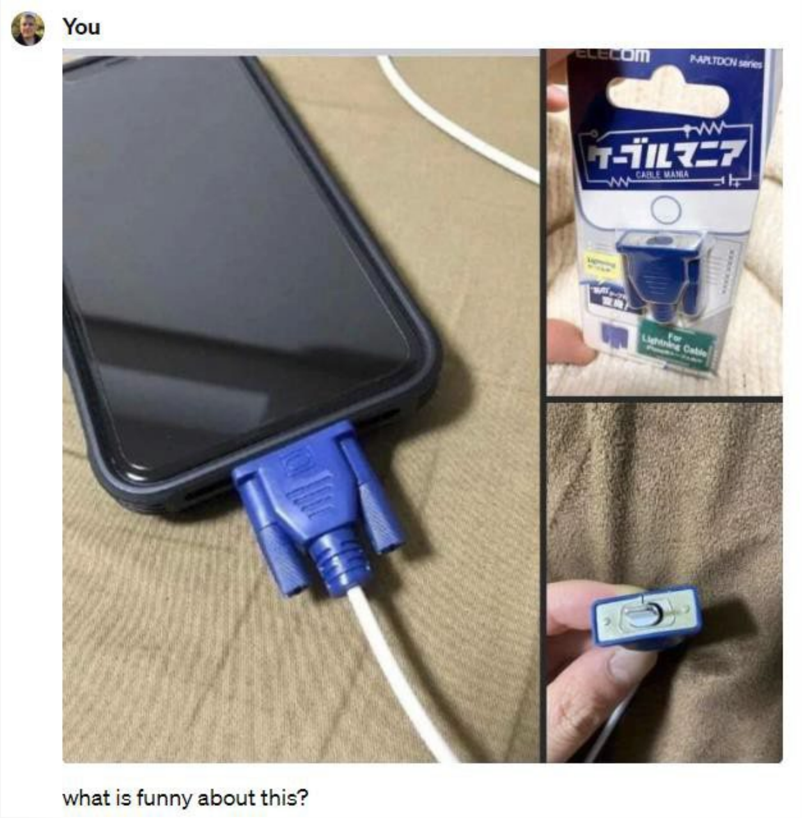


ChatGPT

The photo shows a man standing on a weight scale while another man, who appears to be a security agent, is jotting down his weight. The scene is taking place in a hallway with a checkered floor, and there are several other men in the background, some of whom are smiling or laughing. The humorous aspect of this photo seems to be the presence of the man on the left side of the image, who is smiling and looking at the man on the scale with a playful or teasing expression. This man bears a resemblance to a well-known public figure. The interaction between these two men, combined with the casual and somewhat incongruous setting for such an activity, contributes to the light-hearted and funny nature of the scene. It's as if there's an inside joke being shared among the group, with the weight measurement being a public and possibly unexpected event.

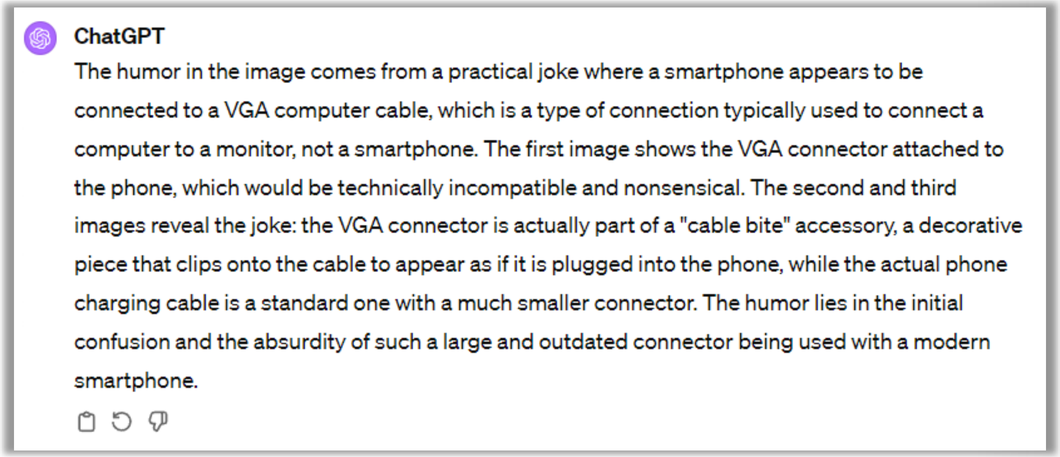
partial credit by CS311n

What’s funny about this? GPT-4 Vision



You

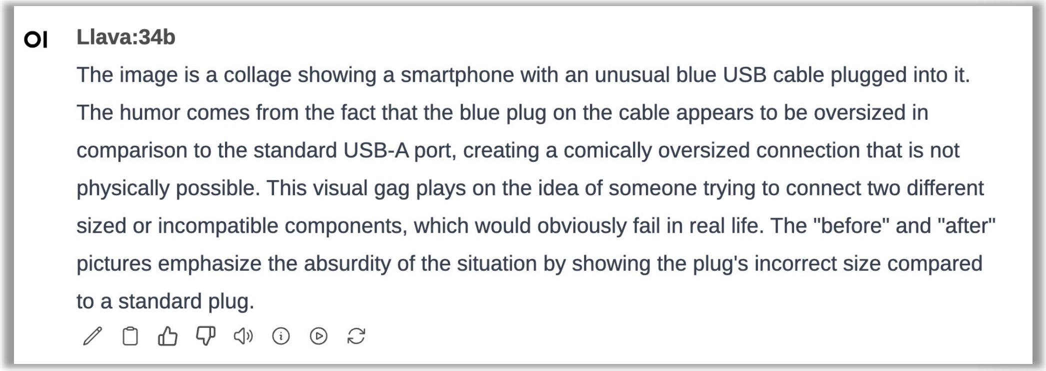
what is funny about this?



ChatGPT

The humor in the image comes from a practical joke where a smartphone appears to be connected to a VGA computer cable, which is a type of connection typically used to connect a computer to a monitor, not a smartphone. The first image shows the VGA connector attached to the phone, which would be technically incompatible and nonsensical. The second and third images reveal the joke: the VGA connector is actually part of a "cable bite" accessory, a decorative piece that clips onto the cable to appear as if it is plugged into the phone, while the actual phone charging cable is a standard one with a much smaller connector. The humor lies in the initial confusion and the absurdity of such a large and outdated connector being used with a modern smartphone.

LLaVA 1.6 34B



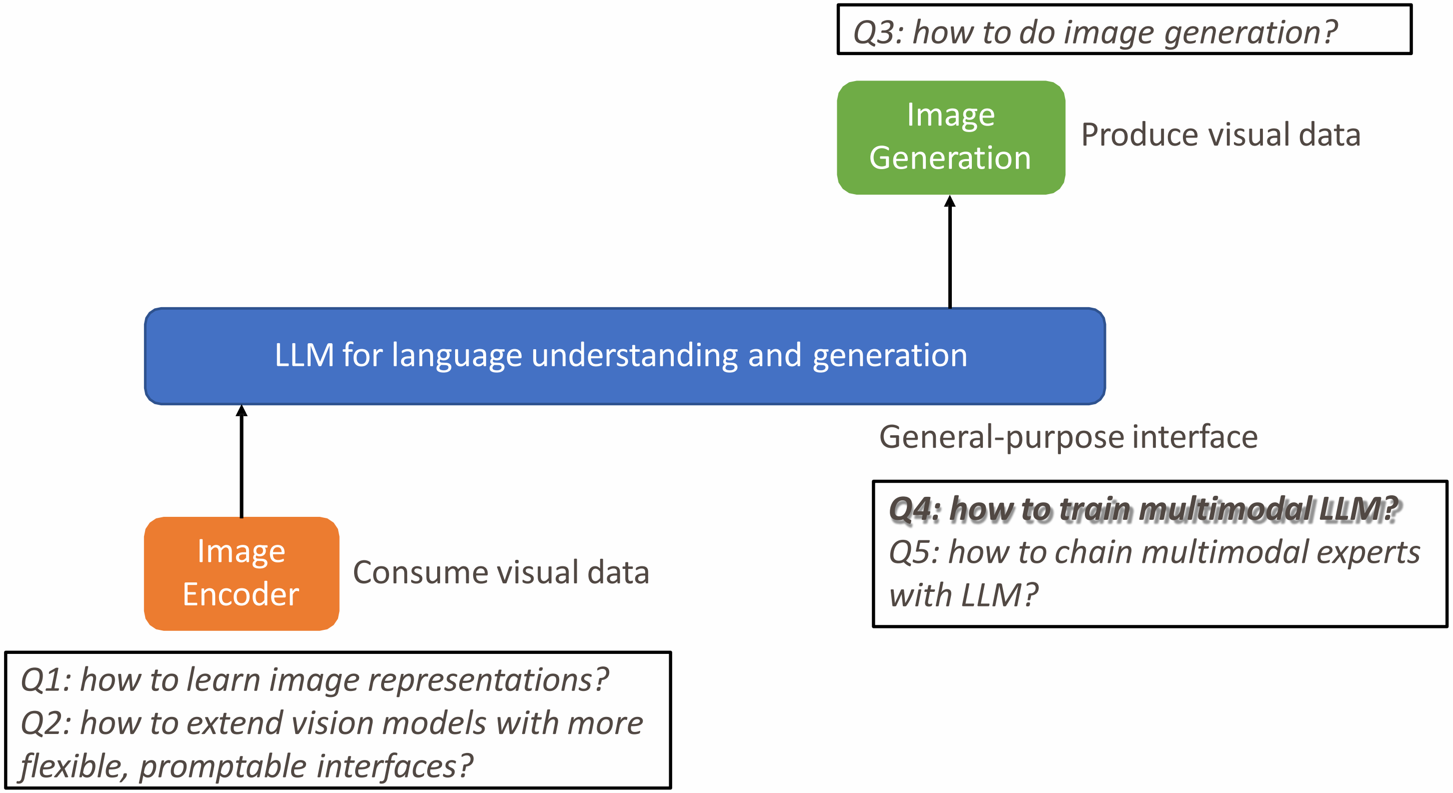
Llava:34b

The image is a collage showing a smartphone with an unusual blue USB cable plugged into it. The humor comes from the fact that the blue plug on the cable appears to be oversized in comparison to the standard USB-A port, creating a comically oversized connection that is not physically possible. This visual gag plays on the idea of someone trying to connect two different sized or incompatible components, which would obviously fail in real life. The "before" and "after" pictures emphasize the absurdity of the situation by showing the plug's incorrect size compared to a standard plug.

Image source: https://www.reddit.com/r/hmmm/comments/ubab5v/hmmm/

partial credit by CS311n

QUICK INTRODUCTION TO TOKENS AND EMBEDDINGS REQUIRED TO UNDERSTAND HOW LLMS PROCESS TEXT AND IMAGES.



Q3: how to do image generation?

Image Generation

Produce visual data

LLM for language understanding and generation

General-purpose interface

Q4: how to train multimodal LLM?
Q5: how to chain multimodal experts with LLM?

Image Encoder

Consume visual data

Q1: how to learn image representations?
Q2: how to extend vision models with more flexible, promptable interfaces?

partial credit by CS311n

Outline

- Background
  - Traditional Large Multimodal Models (LMM)
  - Multimodal GPT4
- Pre-requisite: Instruction Tuning in Large Language Models
- Instructed Tuned Large Multimodal Models
  - **Open-Source Prototypes: LLaVA / MiniGPT4**
  - **Emerging Extensions**
  - **Benchmarking, Properties and Applications**

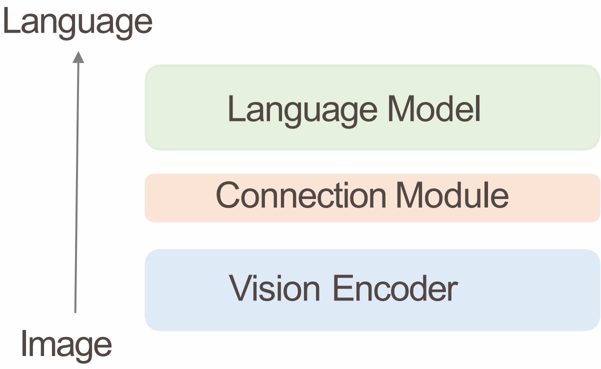
partial credit by CS311n

LARGE MULTIMODAL MODELS (LMM) & GPT4

Large Multimodal Models: Image-to-Text Generative Models

- Model Architectures
  - (Pre-trained) Image Encoder and Language Models
  - Trainable modules to connect to two modalities

A dog lying on the grass next to a frisbee



Language

Language Model

Connection Module

Vision Encoder

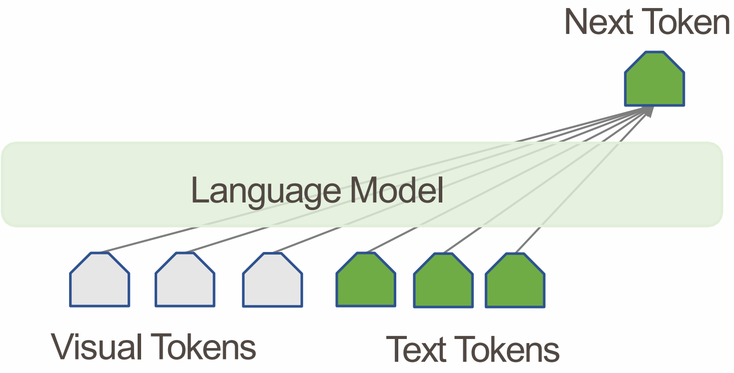
Image

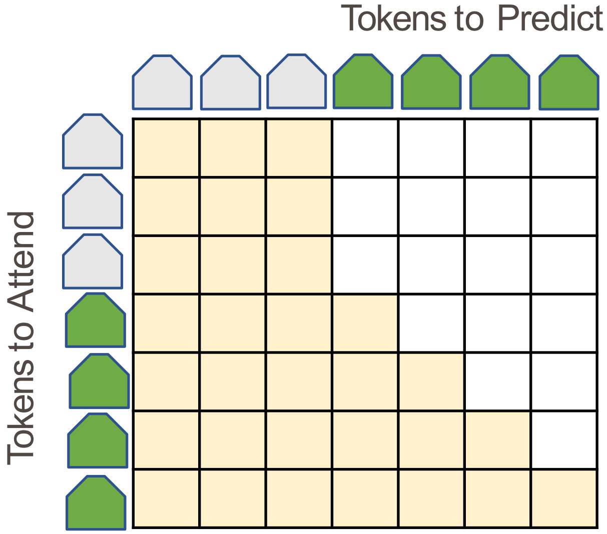
partial credit by CS311n

Large Multimodal Models: Image-to-Text Generative Models

❑ Training Objective

- Cross-Attended Image-to-Text Generation
- Autoregressive loss language output

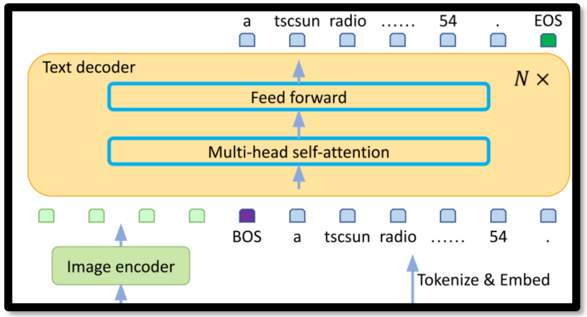




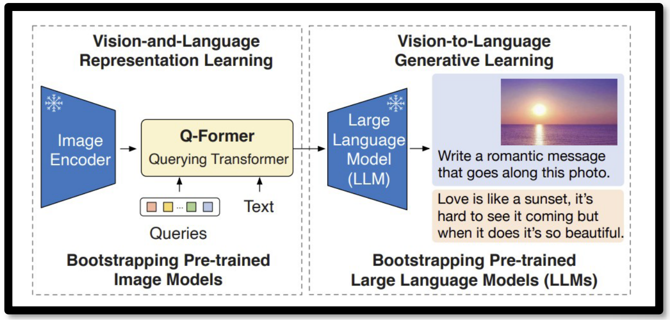
partial credit by CS311n

Example 1: LMM with Image-Text Pairs

- GIT



- BLIP2



Write a romantic message that goes along this photo.

Love is like a sunset, it's hard to see it coming but when it does it's so beautiful.

Language Model	From Scratch	Pre-trained: FLAN-T5/OPT
Connection Module		Q-Former: Lightweight Querying Transformer
Vision Encoder	Contrastive pre-trained: Florence/CLIP	Contrastive pre-trained: EVA/CLIP

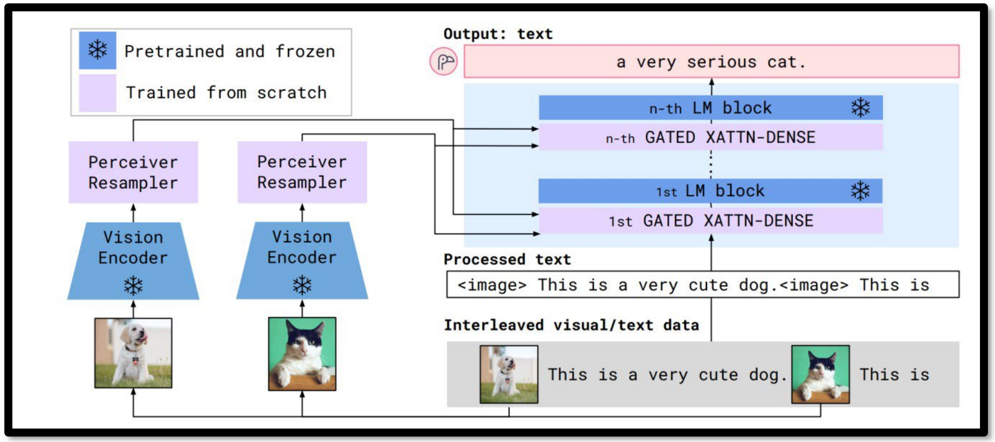
GIT: A Generative Image-to-text Transformer for Vision and Language

Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models

partial credit by CS311n

Example 2: LMM with Interleaved Image-Text Data

- Flamingo:



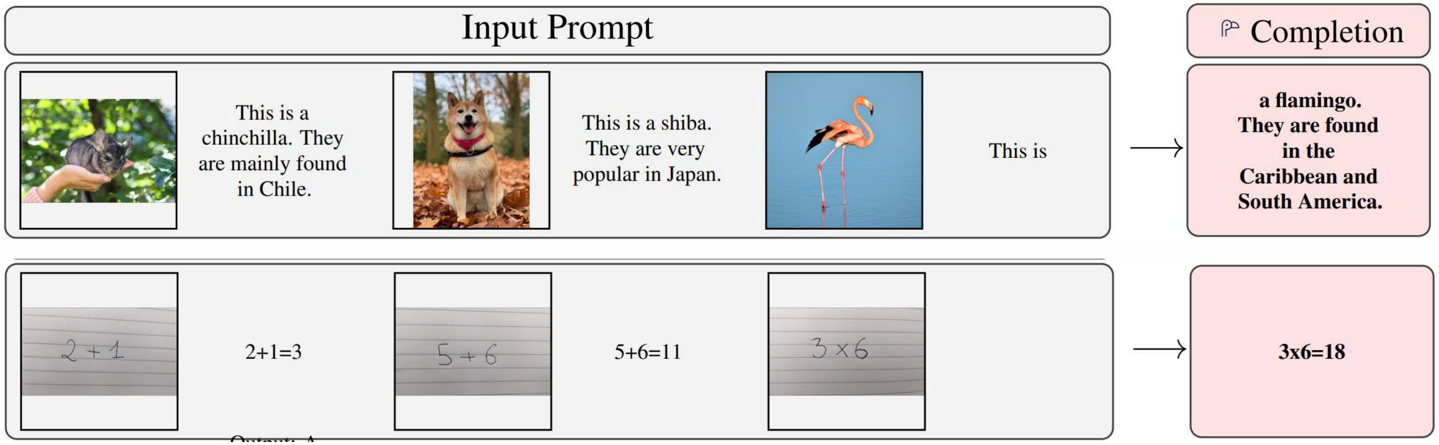
Language Model	Pre-trained: 70B Chinchilla
Connection Module	Perceiver Resampler Gated Cross-attention + Dense
Vision Encoder	Pre-trained: Nonrmalizer-Free ResNet (NFNet)

partial credit by CS311n

Example 2: LMM with Interleaved Image-Text Data

- Flamingo: Multimodal In-Context-Learning

Emerging Property



Input Prompt

Completion

This is a chinchilla. They are mainly found in Chile.

This is a shiba. They are very popular in Japan.

This is

a flamingo. They are found in the Caribbean and South America.

2+1=3

5+6=11

3x6=18

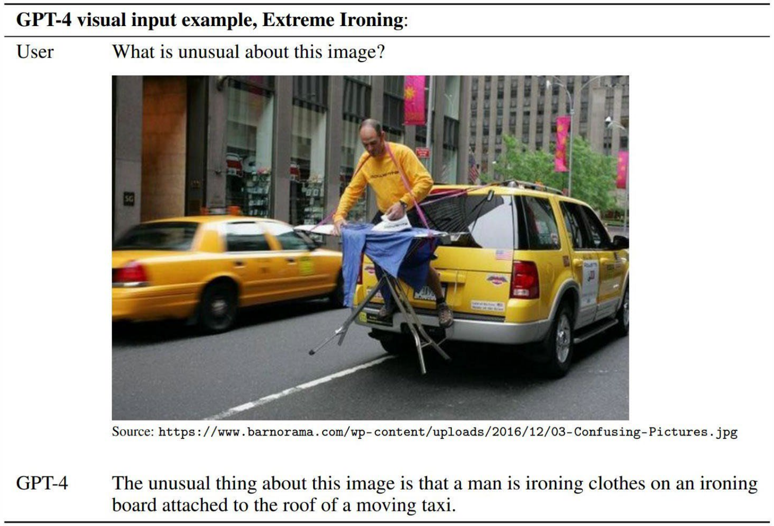
partial credit by CS311n

MultiModal GPT-4

OpenAI

- Model Details: Unknown
- Capability: Strong zero-shot visual understanding & reasoning on many user-oriented tasks in the wild

- How can we build Multimodal GPT-4 like models?

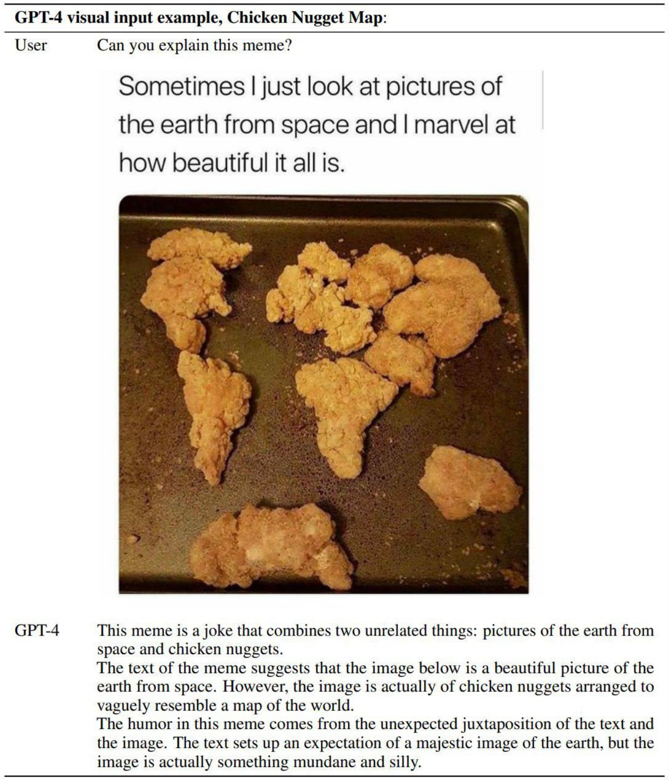


GPT-4 visual input example, Extreme Ironing:

User What is unusual about this image?

Source: `https://www.barnorama.com/wp-content/uploads/2016/12/03-Confusing-Pictures.jpg`

GPT-4 The unusual thing about this image is that a man is ironing clothes on an ironing board attached to the roof of a moving taxi.



GPT-4 visual input example, Chicken Nugget Map:

User Can you explain this meme?

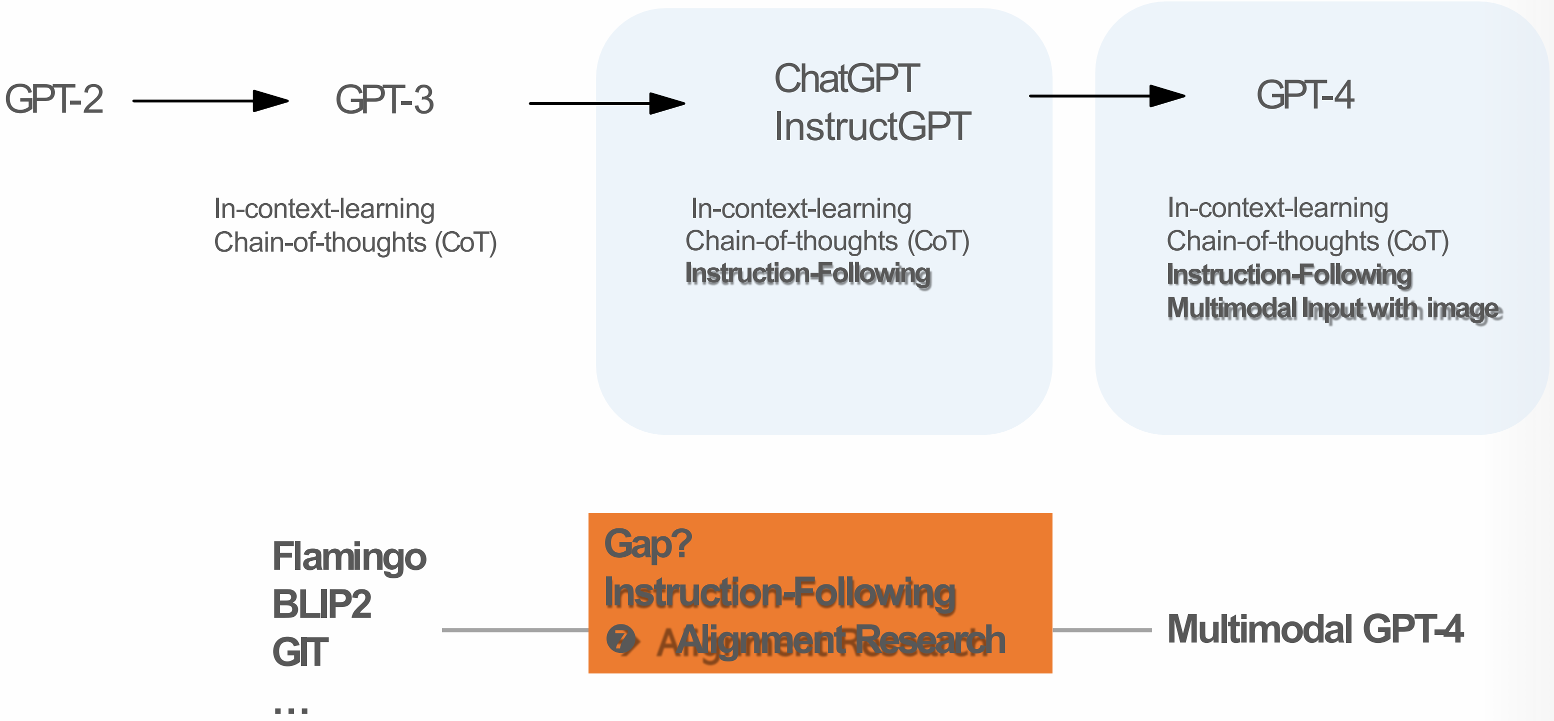
Sometimes I just look at pictures of the earth from space and I marvel at how beautiful it all is.

GPT-4 This meme is a joke that combines two unrelated things: pictures of the earth from space and chicken nuggets.
The text of the meme suggests that the image below is a beautiful picture of the earth from space. However, the image is actually of chicken nuggets arranged to vaguely resemble a map of the world.
The humor in this meme comes from the unexpected juxtaposition of the text and the image. The text sets up an expectation of a majestic image of the earth, but the image is actually something mundane and silly.

GPT-4 Technical Report, OpenAI

partial credit by CS311n

Recap on Language Modeling: Large Language Models (LLM)



OpenAI

GPT-2 → GPT-3 → ChatGPT InstructGPT → GPT-4

What's new?

GPT-3:
In-context-learning
Chain-of-thoughts (CoT)

ChatGPT / InstructGPT:
In-context-learning
Chain-of-thoughts (CoT)
Instruction-Following

GPT-4:
In-context-learning
Chain-of-thoughts (CoT)
Instruction-Following
Multimodal Input with image

Multimodal Space

Flamingo
BLIP2
GIT
…

Gap?
Instruction-Following
? Alignment Research

Multimodal GPT-4

partial credit by CS311n

INSTRUCTION TUNING IN LARGE LANGUAGE MODELS (LLM)

Instruction Tuning

	Input	Output
Translation	*Hello, Vancouver*	你好，溫哥華
Summarization	*CVPR is the premier annual computer vision event comprising the main conference and several co-located workshops and short courses. This year, CVPR will be single track such that everyone (with full passport registration) can attend everything.*	*CVPR: top computer vision event, single-track, accessible to all.*

- Task instructions are implicit.
- Individual models are trained, or multi-tasking without specifying the instructions
- Hard to generalize to new tasks in zero-shot

partial credit by CS311n

Instruction Tuning

Instruction	Input →	Output
Translate English into Simplified Chinese	*Hello, Vancouver*	你好，溫哥華
Summarize in just 10 words to make the message even more brief and easier to remember.	*CVPR is the premier annual computer vision event comprising the main conference and several co-located workshops and short courses. This year, CVPR will be single track such that everyone (with full passport registration) can attend everything.*	*CVPR: top computer vision event, single-track, accessible to all.*

- Task instructions are explicit, expressed in natural language
- One single model is trained, multi-tasking with specified instructions
- Natural and easy to generalize to new tasks in zero-shot

partial credit by CS311n

Instruction Tuning

Instruction

Input → Output

Summarize in Chinese to make it easier to remember.

CVPR 2023 Visas

The CVPR'23 organizing committee deeply regrets that many members of our community could not receive visas to attend CVPR 2023. For several months, the organizers have actively raised concerns with Canadian immigration authorities (IRCC), government agencies, and politicians. In some cases, we have been successful in helping people obtain visas, but in many cases, our efforts were unsuccessful. The organizers acknowledge that the international representation of members from all over the world is what has made CVPR successful. We share in the frustration of those who were unable to attend. We continue to allow virtual to in-person registration switches for attendees who receive their visas before the conference.

*CVPR'23*签证问题: 组委会努力解决，提供虚拟和现场注册转换服务

"CVPR'23 visa issue: organizing committee works to solve and provide virtual and in-person registration switch services."

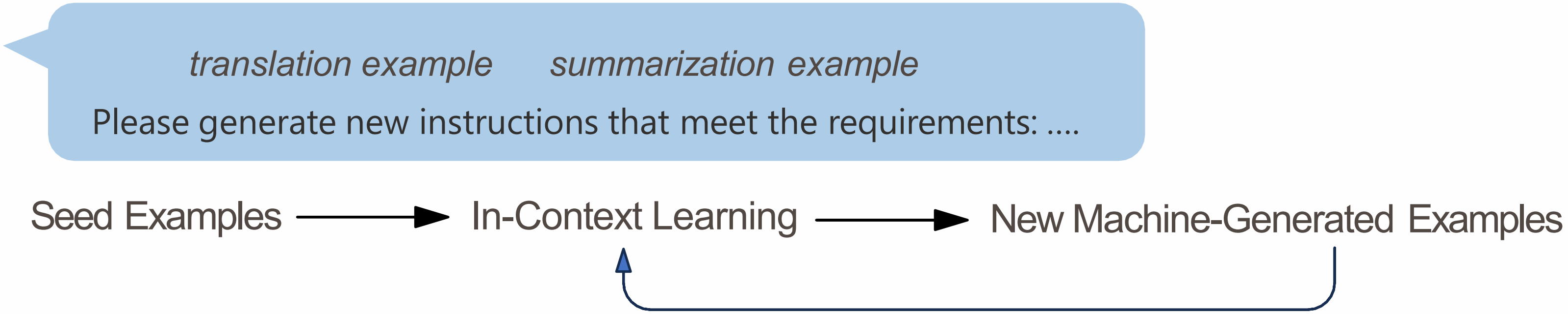
partial credit by CS311n

Self-Instruct Tuning

How to collect a diverse set of high-quality instructions and their responses?

❑ Human-Human: Collected from humans with high cost

❑ Human-Machine: A Strong LLM Teacher such as GPT3 and GPT4



translation example *summarization example*

Please generate new instructions that meet the requirements: ....

Seed Examples → In-Context Learning → New Machine-Generated Examples

partial credit by CS311n

Language Modeling: Large Language Models (LLM)

OpenAI

GPT-2 → GPT-3 → ChatGPT InstructGPT → GPT-4

What's new?

GPT-3:
In-context-learning
Chain-of-thoughts (CoT)

ChatGPT / InstructGPT:
In-context-learning
Chain-of-thoughts (CoT)
Instruction-Following

GPT-4:
In-context-learning
Chain-of-thoughts (CoT)
Instruction-Following
Multimodal Input with image

Open Source Community

LLaMA

Alpaca

Vicuna

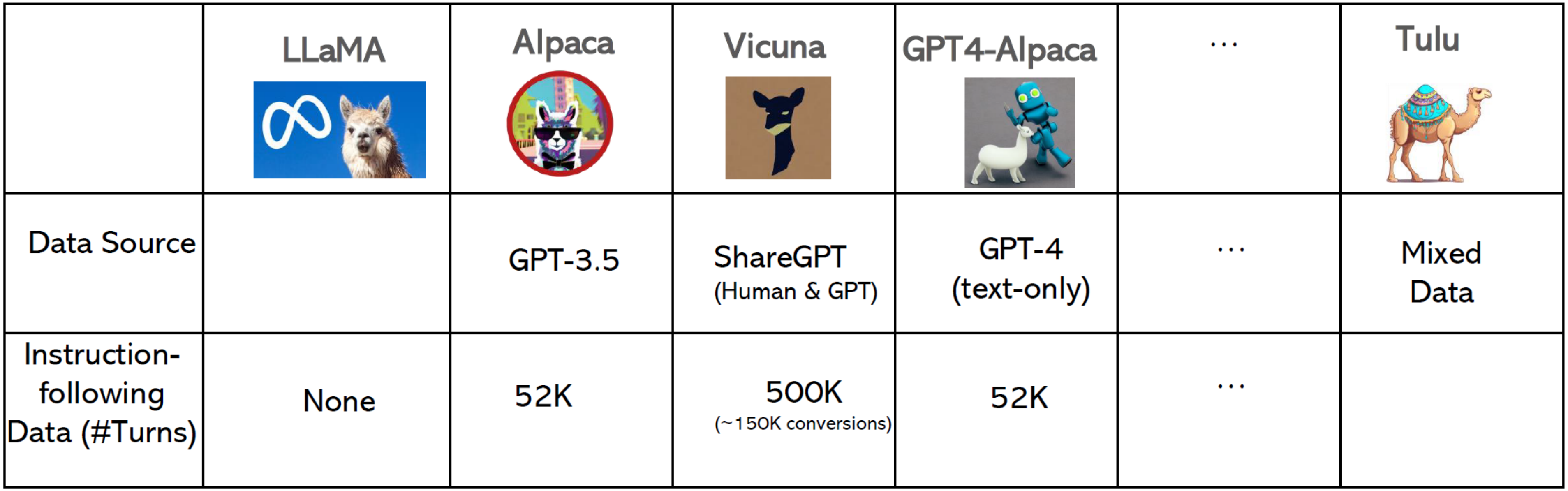
GPT4-Alpaca

Tulu

partial credit by CS311n

Instruction Tuning with Open-Source LLMs

Self-Instruct with Strong Teacher LLMs & Mixed Human Data



	LLaMA	Alpaca	Vicuna	GPT4-Alpaca	…	Tulu
Data Source		GPT-3.5	ShareGPT (Human & GPT)	GPT-4 (text-only)	…	Mixed Data
Instruction-following Data (#Turns)	None	52K	500K (~150K conversions)	52K	…	

partial credit by CS311n

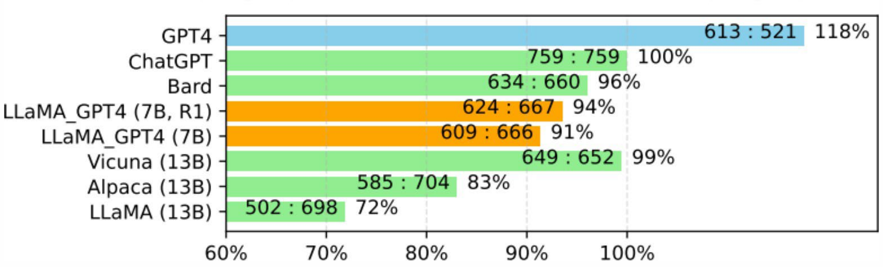
Results on Chatbot

Evaluation Metric: Ask GPT-4 to rate the two model responses (1-10), then compute the ratio, i.e. relative score

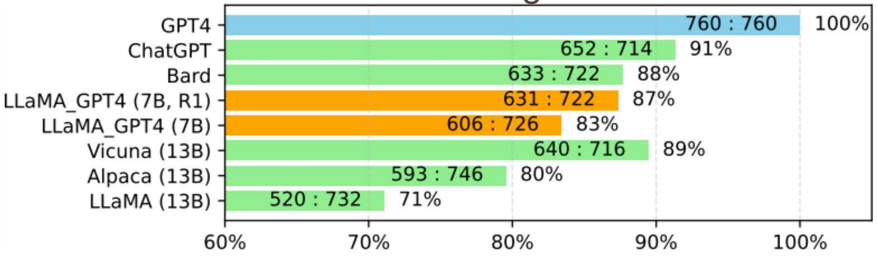
Findings:

- A VERY CONSISTENT Evaluation Metric !
- Opensourced Chatbots mimicked commercial ones

All chatbots against ChatGPT



All chatbots against GPT-4



partial credit by CS311n

Large Multimodal Models

-- Building multimodal gpt4 with open-source resources

LLaVA as a running example in this lecture

- Data
- Model
- Performance

partial credit by CS311n

Visual Instruction Tuning with GPT-4

https://llava-vl.github.io/

Haotian Liu*, Chunyuan Li*, Qingyang Wu, Yong Jae Lee (* Equal contribution)

Self-Instruct with Strong Teacher LLMs

	LLaMA	Alpaca	Vicuna
Teacher		GPT-3.5	ShareGPT (Human & GPT)
Instruction-following Data	None	52K	700K (70 conversions)

But No Teacher is available on multiGPT4?

GPT-4-LLM	LLaVA
GPT-4 (text-only)	GPT-4 (text-only)

- 158K multimodal instruction following data (First & High Quality)

→ Multimodal Chatbot

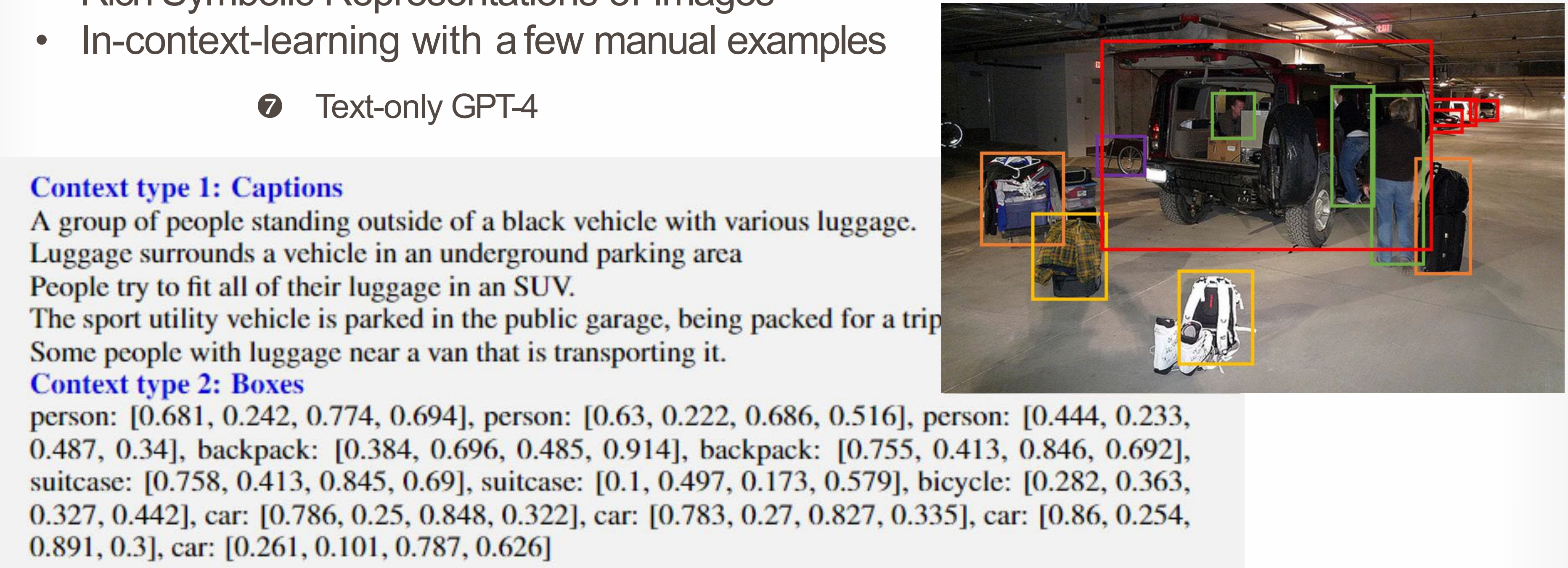
Large Language and Vision Assistant

partial credit by CS311n

GPT-assisted Visual Instruction Data Generation

- Rich Symbolic Representations of Images
- In-context-learning with a few manual examples

❼ Text-only GPT-4



Context type 1: Captions

A group of people standing outside of a black vehicle with various luggage.
Luggage surrounds a vehicle in an underground parking area
People try to fit all of their luggage in an SUV.
The sport utility vehicle is parked in the public garage, being packed for a trip
Some people with luggage near a van that is transporting it.

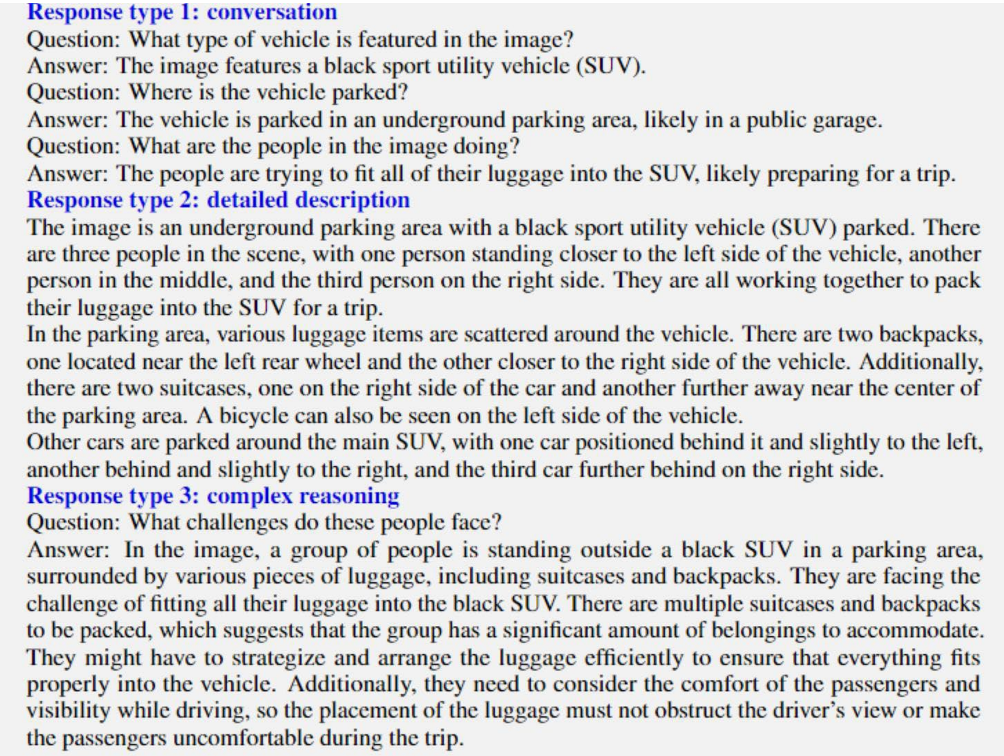
Context type 2: Boxes

person: [0.681, 0.242, 0.774, 0.694], person: [0.63, 0.222, 0.686, 0.516], person: [0.444, 0.233, 0.487, 0.34], backpack: [0.384, 0.696, 0.485, 0.914], backpack: [0.755, 0.413, 0.846, 0.692], suitcase: [0.758, 0.413, 0.845, 0.69], suitcase: [0.1, 0.497, 0.173, 0.579], bicycle: [0.282, 0.363, 0.327, 0.442], car: [0.786, 0.25, 0.848, 0.322], car: [0.783, 0.27, 0.827, 0.335], car: [0.86, 0.254, 0.891, 0.3], car: [0.261, 0.101, 0.787, 0.626]

partial credit by CS311n

GPT-assisted Visual Instruction Data Generation

Three type of instruction-following responses



Response type 1: conversation

Question: What type of vehicle is featured in the image?

Answer: The image features a black sport utility vehicle (SUV).

Question: Where is the vehicle parked?

Answer: The vehicle is parked in an underground parking area, likely in a public garage.

Question: What are the people in the image doing?

Answer: The people are trying to fit all of their luggage into the SUV, likely preparing for a trip.

Response type 2: detailed description

The image is an underground parking area with a black sport utility vehicle (SUV) parked. There are three people in the scene, with one person standing closer to the left side of the vehicle, another person in the middle, and the third person on the right side. They are all working together to pack their luggage into the SUV for a trip.

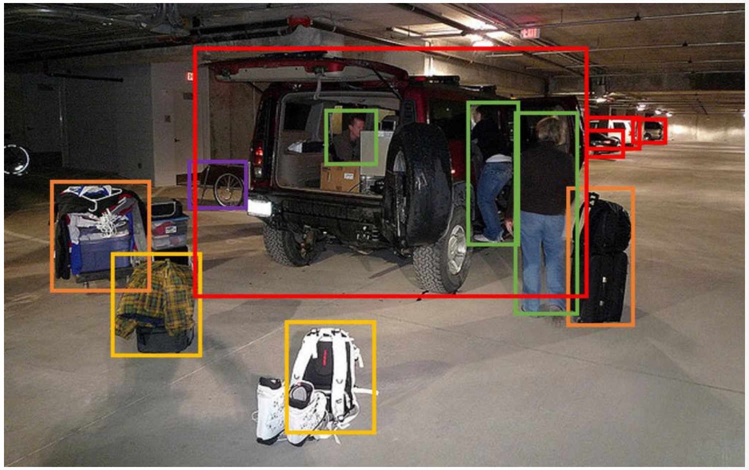
In the parking area, various luggage items are scattered around the vehicle. There are two backpacks, one located near the left rear wheel and the other closer to the right side of the vehicle. Additionally, there are two suitcases, one on the right side of the car and another further away near the center of the parking area. A bicycle can also be seen on the left side of the vehicle.

Other cars are parked around the main SUV, with one car positioned behind it and slightly to the left, another behind and slightly to the right, and the third car further behind on the right side.

Response type 3: complex reasoning

Question: What challenges do these people face?

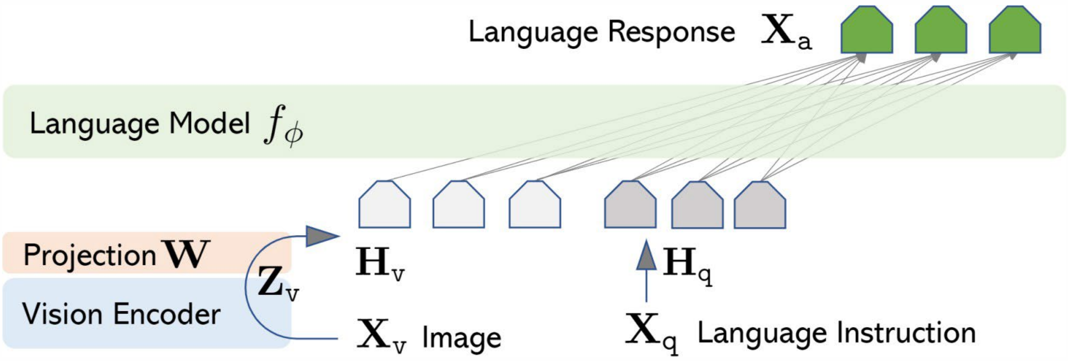
Answer: In the image, a group of people is standing outside a black SUV in a parking area, surrounded by various pieces of luggage, including suitcases and backpacks. They are facing the challenge of fitting all their luggage into the black SUV. There are multiple suitcases and backpacks to be packed, which suggests that the group has a significant amount of belongings to accommodate. They might have to strategize and arrange the luggage efficiently to ensure that everything fits properly into the vehicle. Additionally, they need to consider the comfort of the passengers and visibility while driving, so the placement of the luggage must not obstruct the driver's view or make the passengers uncomfortable during the trip.



partial credit by CS311n

LLaVA: Large Language-and-Vision Assistant

❑ Architecture



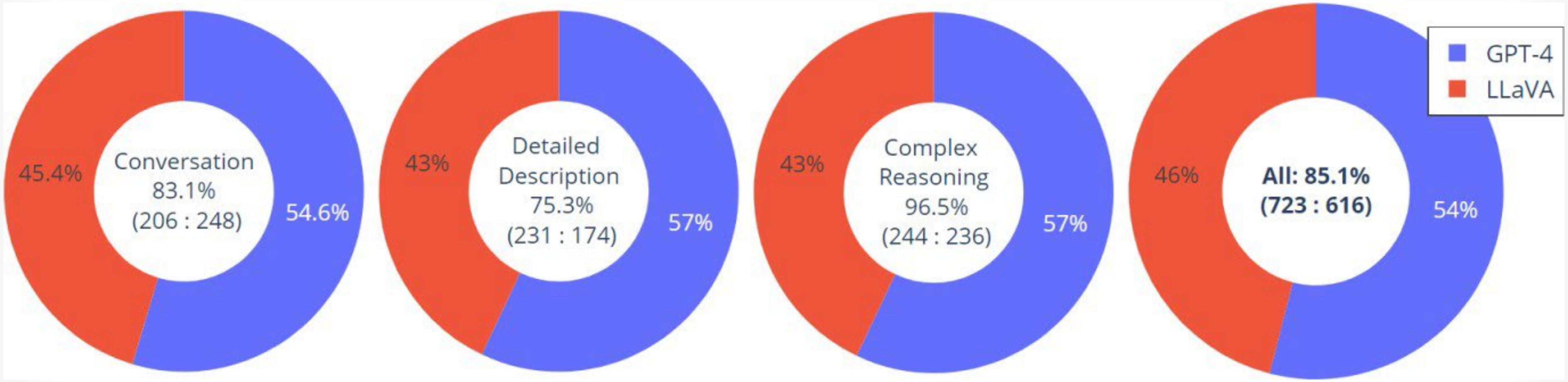
❑ Two-stage Training

- **Stage 1: Pre-training for Feature Alignment**.

  Only the projection matrix is updated, based on a subset of CC3M.
- **Stage 2: Fine-tuning End-to-End.** Both the projection matrix and LLM are updated
  - **Visual Chat**: Our generated multimodal instruction data for daily user-oriented applications.
  - **Science QA**: Multimodal reasoning dataset for the science domain.

partial credit by CS311n

Visual Chat: Towards building multimodal GPT-4 level chatbot



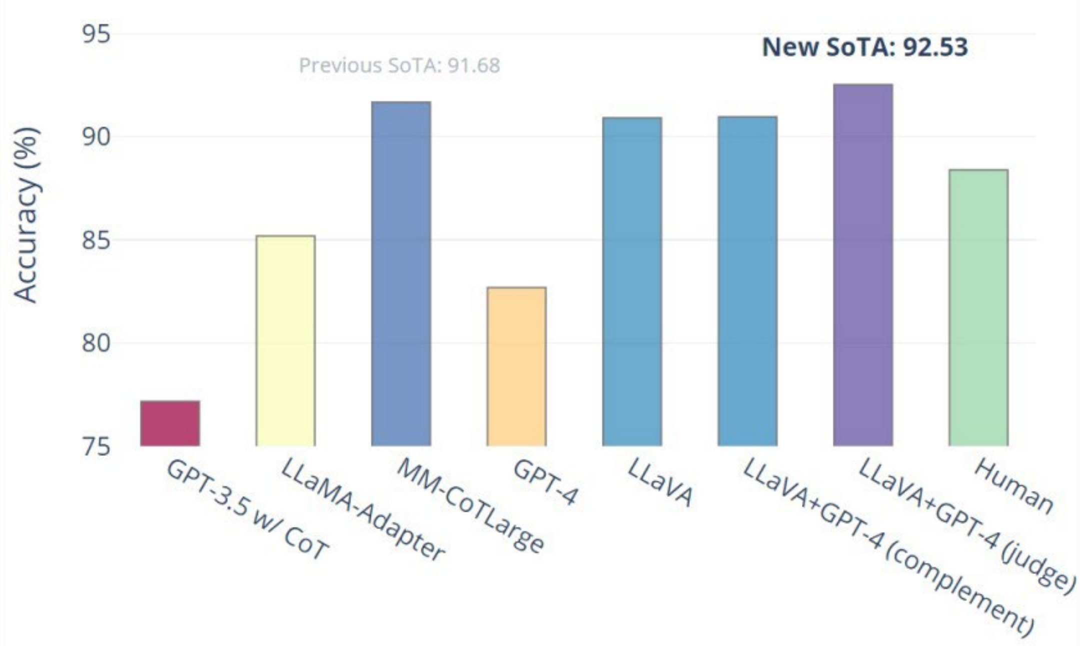
An evaluation dataset with 30 unseen images, 90 new language-image instructions

Overall, LLaVA achieves 85.1% relative score compared with GPT-4

partial credit by CS311n

Science QA: New SoTA with the synergy of LLaVA with GPT-4

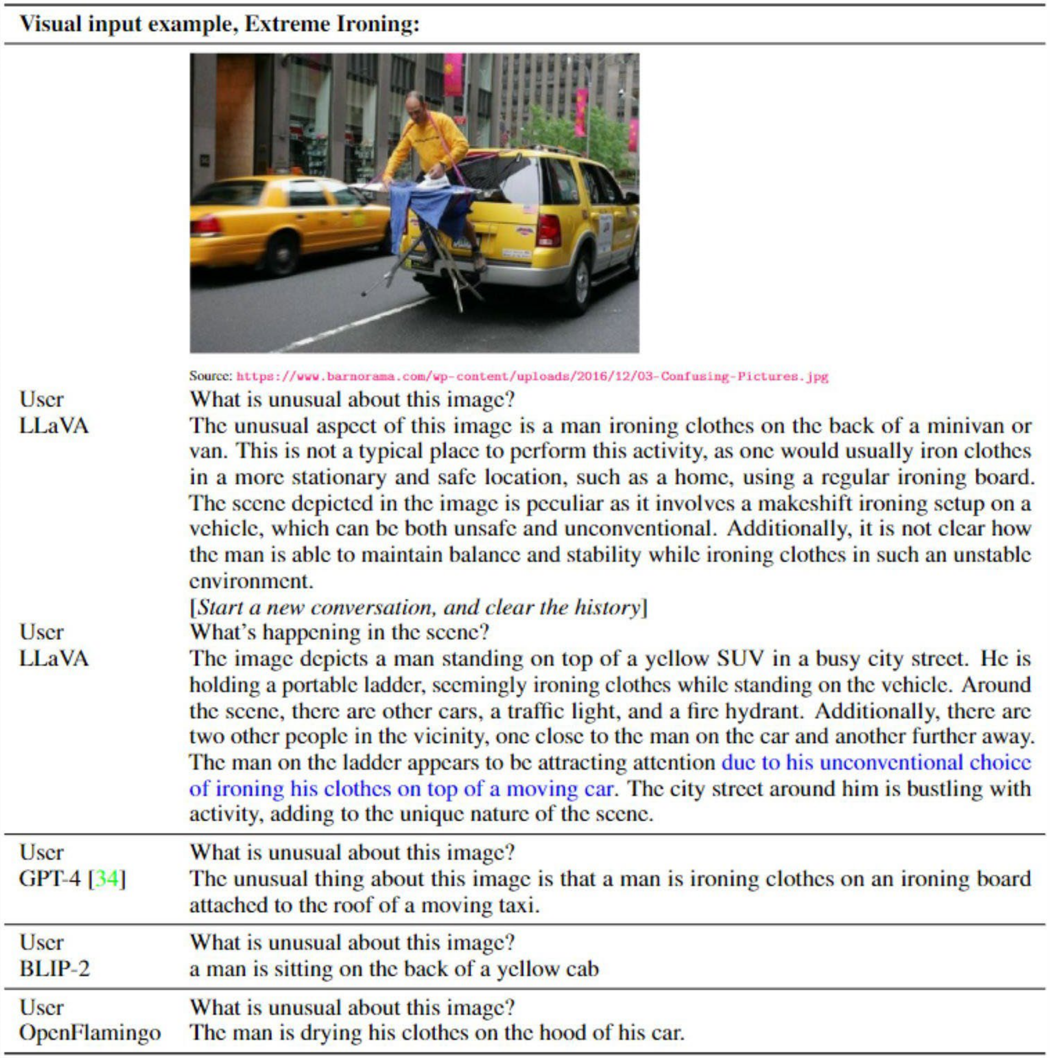
- LLaVA alones achieve 90.92%
- We use the text-only GPT-4 as the juedge, to predict the final answer based on its own previous answers and the LLaVA answers.
- This ``GPT-4 as juedge" scheme yields a new SOTA 92.53%
- GPT-4 is an effective model ensemble method



partial credit by CS311n

Strong Visual Reasoning Ability

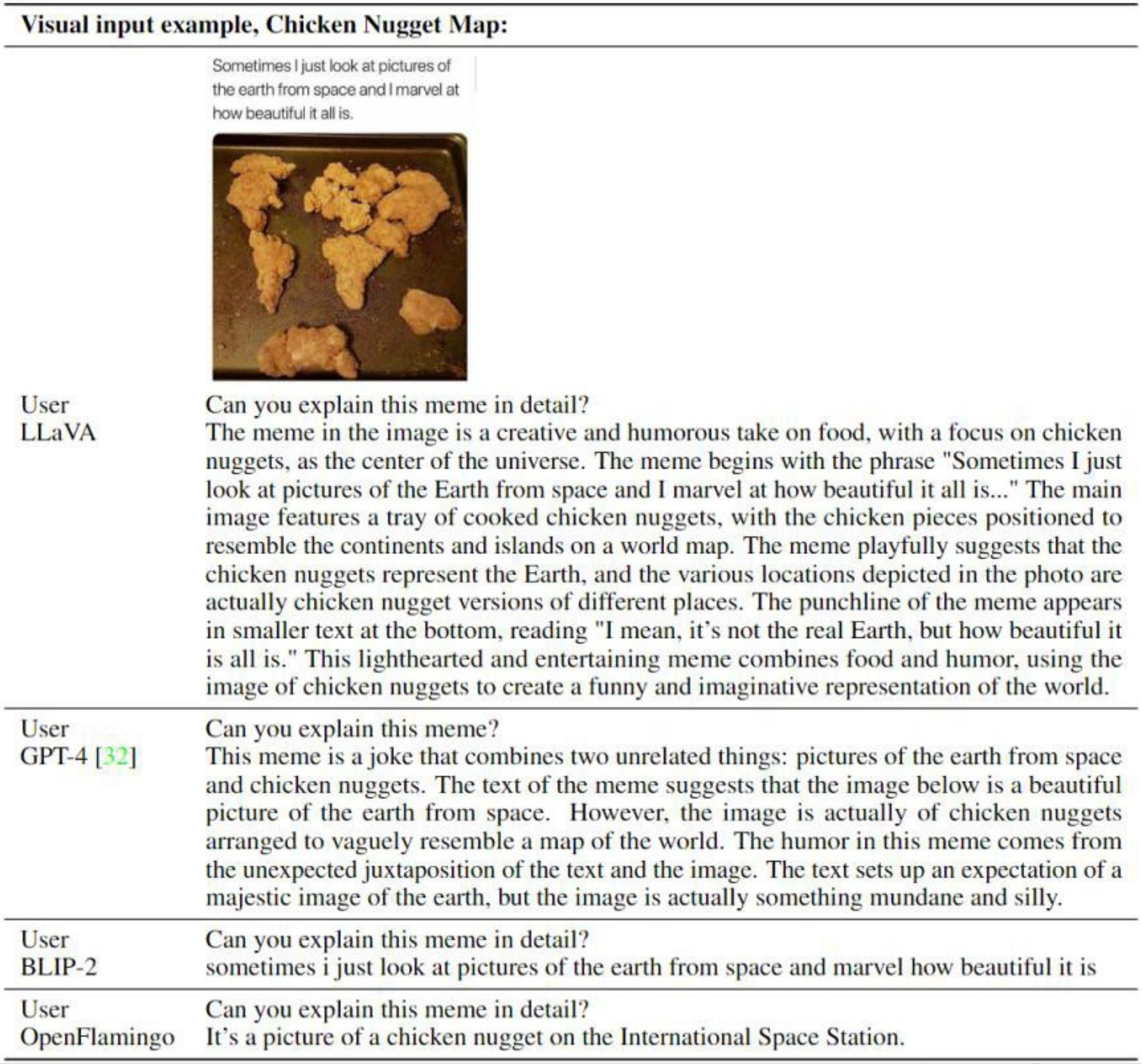
Example 1: Extreme Ironing



Visual input example, Extreme Ironing:	
	Source: https://www.barnorama.com/wp-content/uploads/2016/12/03-Confusing-Pictures.jpg
User	What is unusual about this image?
LLaVA	The unusual aspect of this image is a man ironing clothes on the back of a minivan or van. This is not a typical place to perform this activity, as one would usually iron clothes in a more stationary and safe location, such as a home, using a regular ironing board. The scene depicted in the image is peculiar as it involves a makeshift ironing setup on a vehicle, which can be both unsafe and unconventional. Additionally, it is not clear how the man is able to maintain balance and stability while ironing clothes in such an unstable environment.
	[*Start a new conversation, and clear the history*]
User	What's happening in the scene?
LLaVA	The image depicts a man standing on top of a yellow SUV in a busy city street. He is holding a portable ladder, seemingly ironing clothes while standing on the vehicle. Around the scene, there are other cars, a traffic light, and a fire hydrant. Additionally, there are two other people in the vicinity, one close to the man on the car and another further away. The man on the ladder appears to be attracting attention due to his unconventional choice of ironing his clothes on top of a moving car. The city street around him is bustling with activity, adding to the unique nature of the scene.
User	What is unusual about this image?
GPT-4 [34]	The unusual thing about this image is that a man is ironing clothes on an ironing board attached to the roof of a moving taxi.
User	What is unusual about this image?
BLIP-2	a man is sitting on the back of a yellow cab
User	What is unusual about this image?
OpenFlamingo	The man is drying his clothes on the hood of his car.

Strong Visual Reasoning Ability

Example 2: Chicken Nugget Map

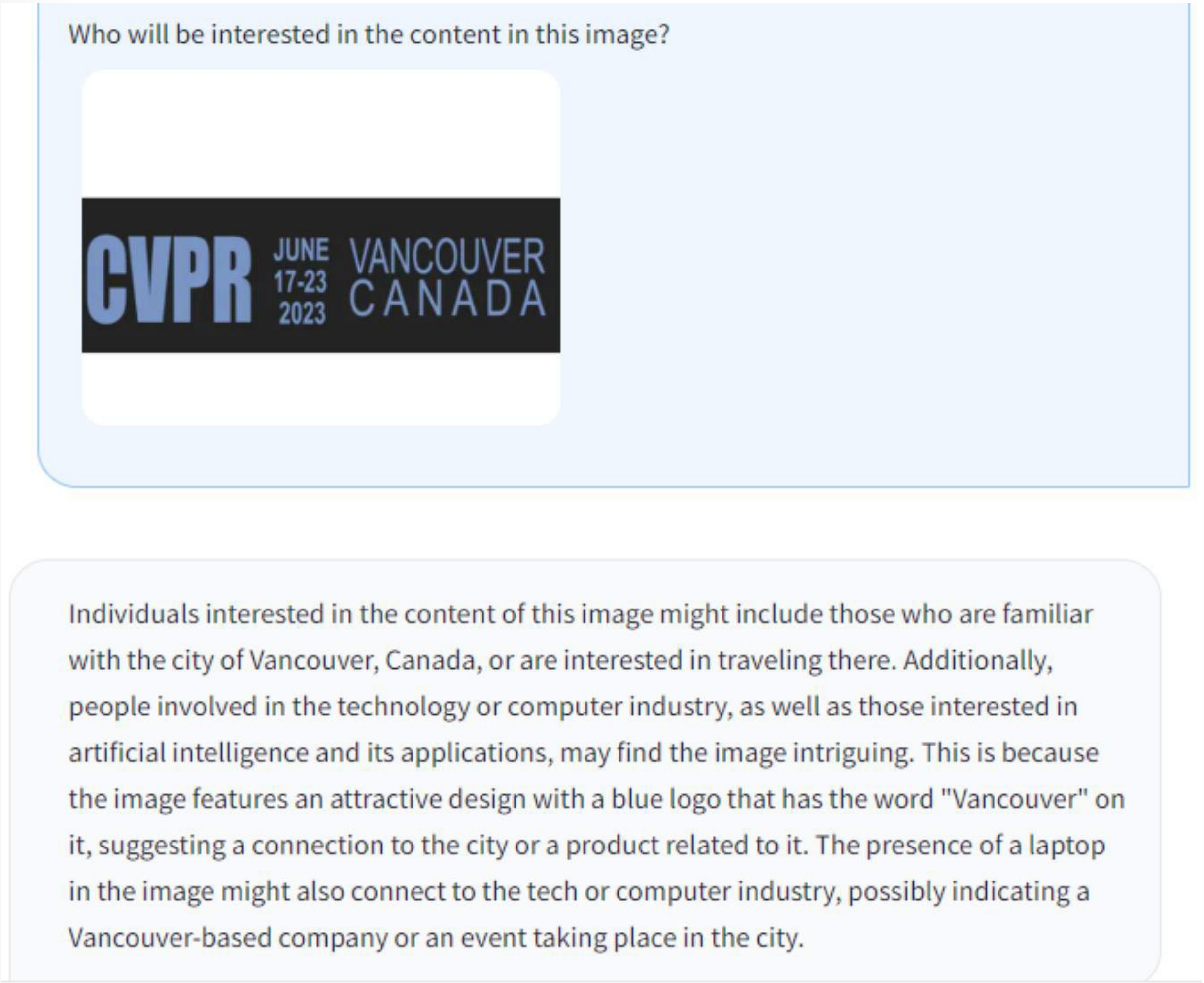


Visual input example, Chicken Nugget Map:

Sometimes I just look at pictures of the earth from space and I marvel at how beautiful it all is.

User	Can you explain this meme in detail?
LLaVA	The meme in the image is a creative and humorous take on food, with a focus on chicken nuggets, as the center of the universe. The meme begins with the phrase "Sometimes I just look at pictures of the Earth from space and I marvel at how beautiful it all is..." The main image features a tray of cooked chicken nuggets, with the chicken pieces positioned to resemble the continents and islands on a world map. The meme playfully suggests that the chicken nuggets represent the Earth, and the various locations depicted in the photo are actually chicken nugget versions of different places. The punchline of the meme appears in smaller text at the bottom, reading "I mean, it's not the real Earth, but how beautiful it is all is." This lighthearted and entertaining meme combines food and humor, using the image of chicken nuggets to create a funny and imaginative representation of the world.
User	Can you explain this meme?
GPT-4 [32]	This meme is a joke that combines two unrelated things: pictures of the earth from space and chicken nuggets. The text of the meme suggests that the image below is a beautiful picture of the earth from space. However, the image is actually of chicken nuggets arranged to vaguely resemble a map of the world. The humor in this meme comes from the unexpected juxtaposition of the text and the image. The text sets up an expectation of a majestic image of the earth, but the image is actually something mundane and silly.
User	Can you explain this meme in detail?
BLIP-2	sometimes i just look at pictures of the earth from space and marvel how beautiful it is
User	Can you explain this meme in detail?
OpenFlamingo	It's a picture of a chicken nugget on the International Space Station.

Strong OCR Ability



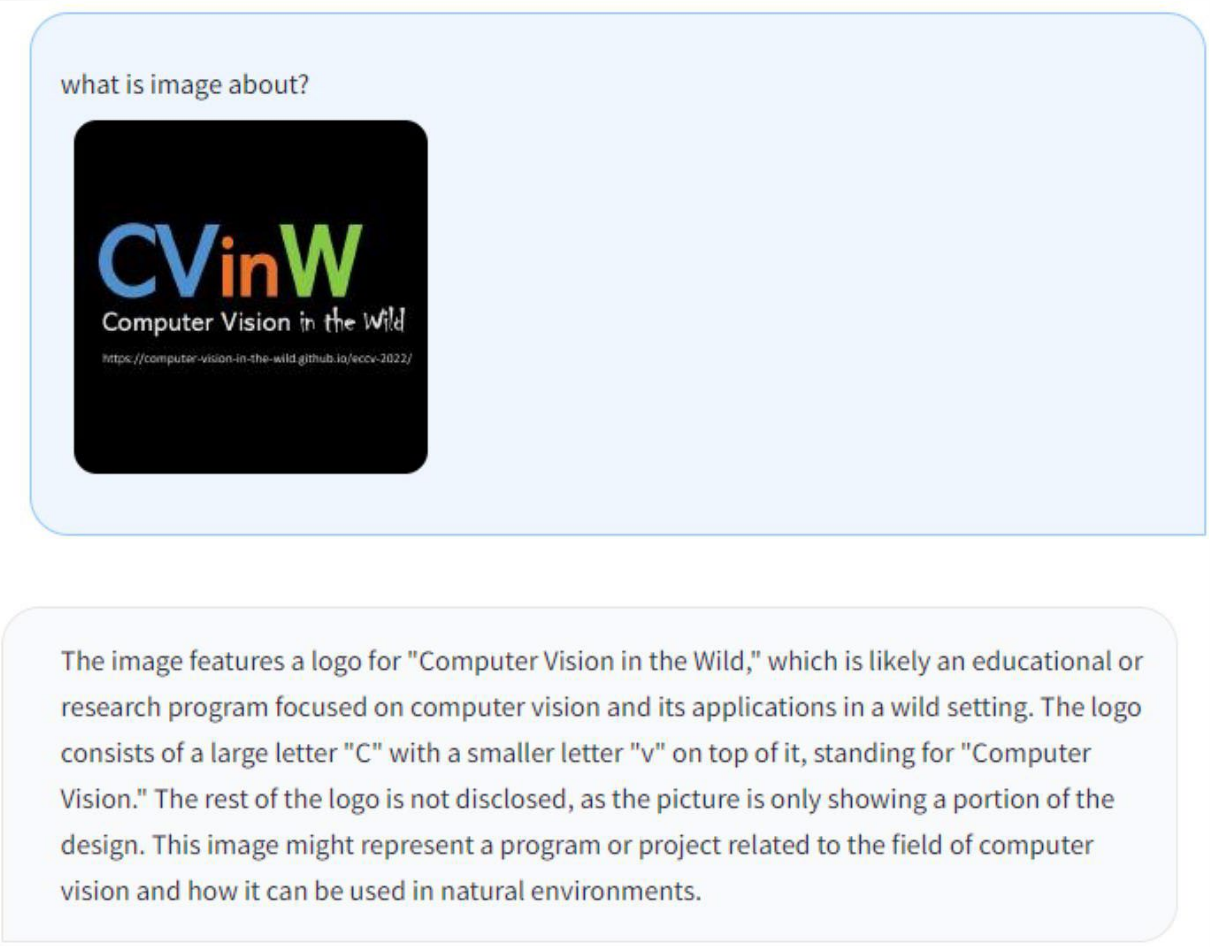
Who will be interested in the content in this image?

Individuals interested in the content of this image might include those who are familiar with the city of Vancouver, Canada, or are interested in traveling there. Additionally, people involved in the technology or computer industry, as well as those interested in artificial intelligence and its applications, may find the image intriguing. This is because the image features an attractive design with a blue logo that has the word "Vancouver" on it, suggesting a connection to the city or a product related to it. The presence of a laptop in the image might also connect to the tech or computer industry, possibly indicating a Vancouver-based company or an event taking place in the city.

Example 1: CVPR& Vancouver

partial credit by CS311n

Strong OCR Ability


CVinW
Computer Vision in the Wild
https://computer-vision-in-the-wild.github.io/eccv-2022/

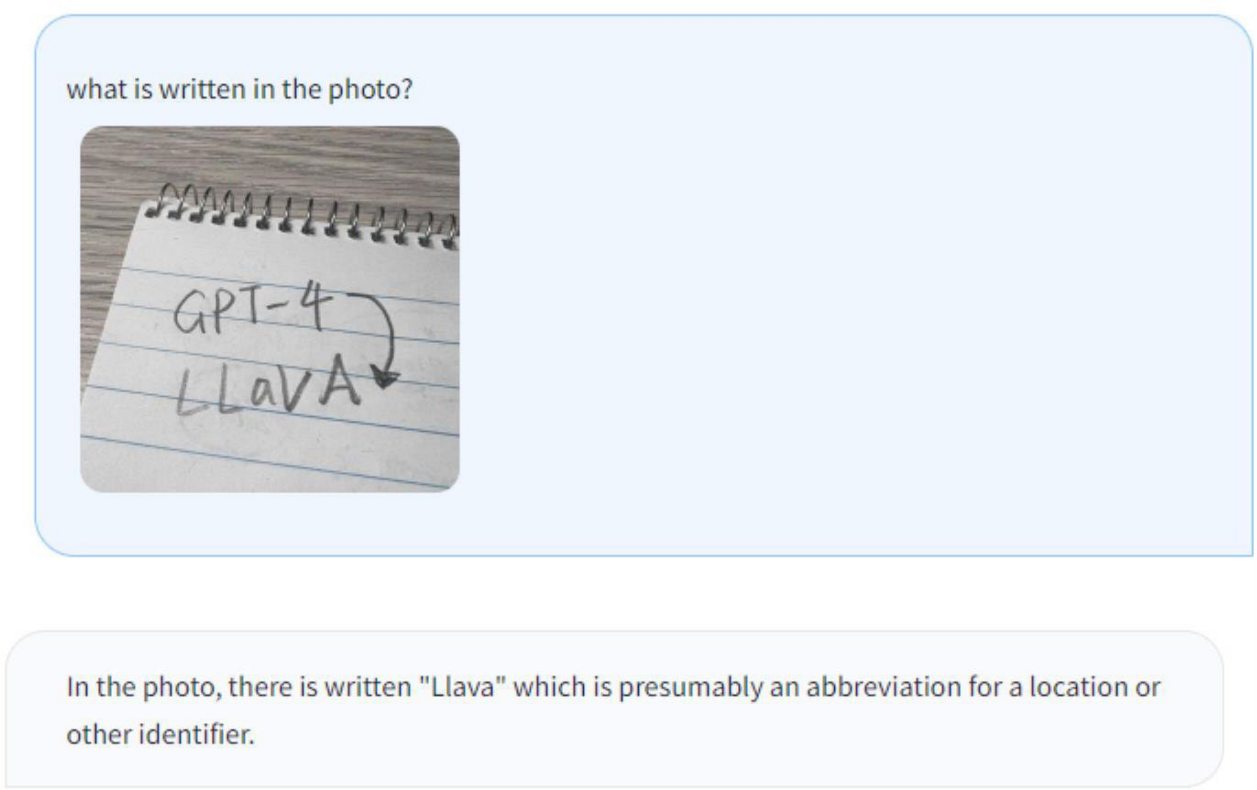
what is image about?

The image features a logo for "Computer Vision in the Wild," which is likely an educational or research program focused on computer vision and its applications in a wild setting. The logo consists of a large letter "C" with a smaller letter "v" on top of it, standing for "Computer Vision." The rest of the logo is not disclosed, as the picture is only showing a portion of the design. This image might represent a program or project related to the field of computer vision and how it can be used in natural environments.

Example 2: CVinW workshop

partial credit by CS311n

Strong OCR Ability



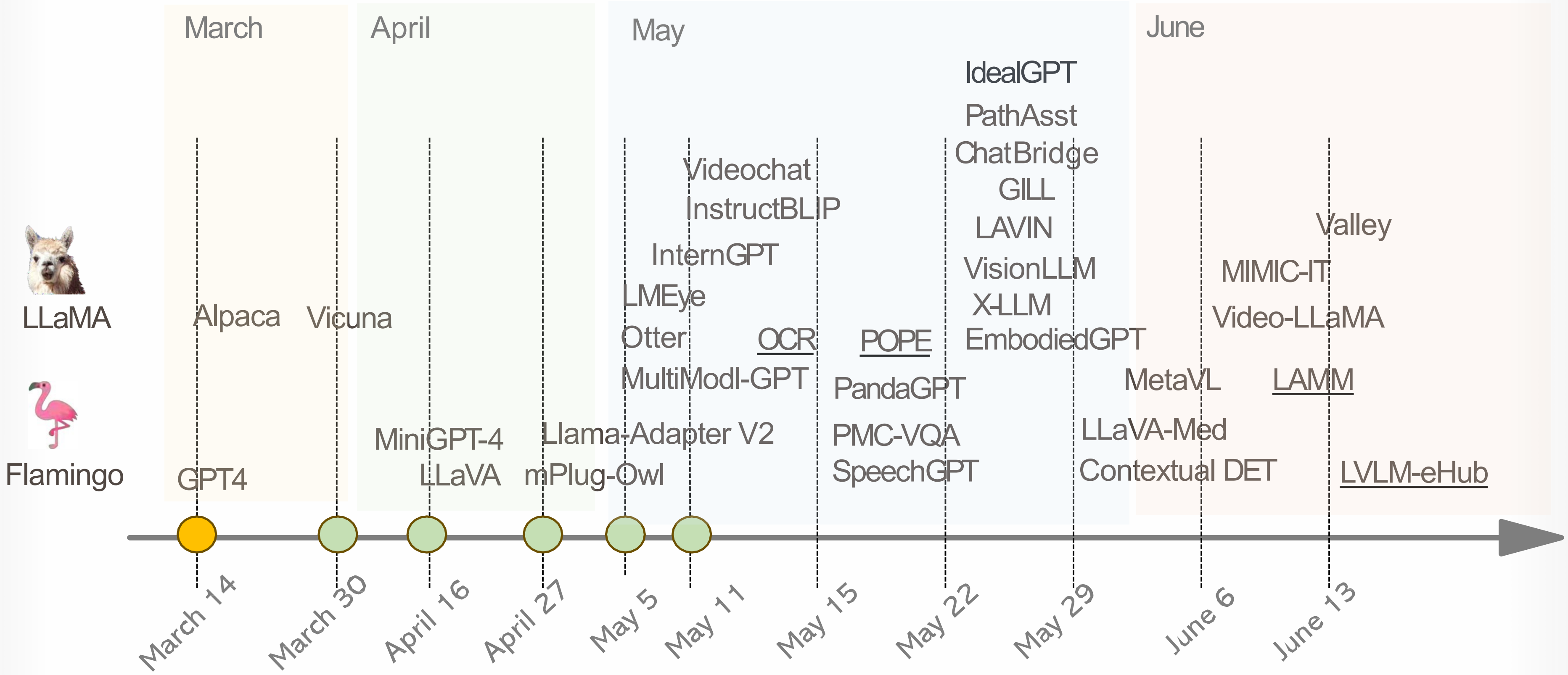
what is written in the photo?

In the photo, there is written "Llava" which is presumably an abbreviation for a location or other identifier.

Example 3: LLaVA

partial credit by CS311n

Emerging Topics



LLaMA

Flamingo

March

GPT4

Alpaca

Vicuna

April

MiniGPT-4

LLaVA

Llama-Adapter V2

mPlug-Owl

May

Videochat

InstructBLIP

InternGPT

LMEye

Otter

MultiModl-GPT

OCR

POPE

PandaGPT

PMC-VQA

SpeechGPT

IdealGPT

PathAsst

ChatBridge

GILL

LAVIN

VisionLLM

X-LLM

EmbodiedGPT

MetaVL

LLaVA-Med

Contextual DET

June

Valley

MIMIC-IT

Video-LLaMA

LAMM

LVLM-eHub

March 14

March 30

April 16

April 27

May 5

May 11

May 15

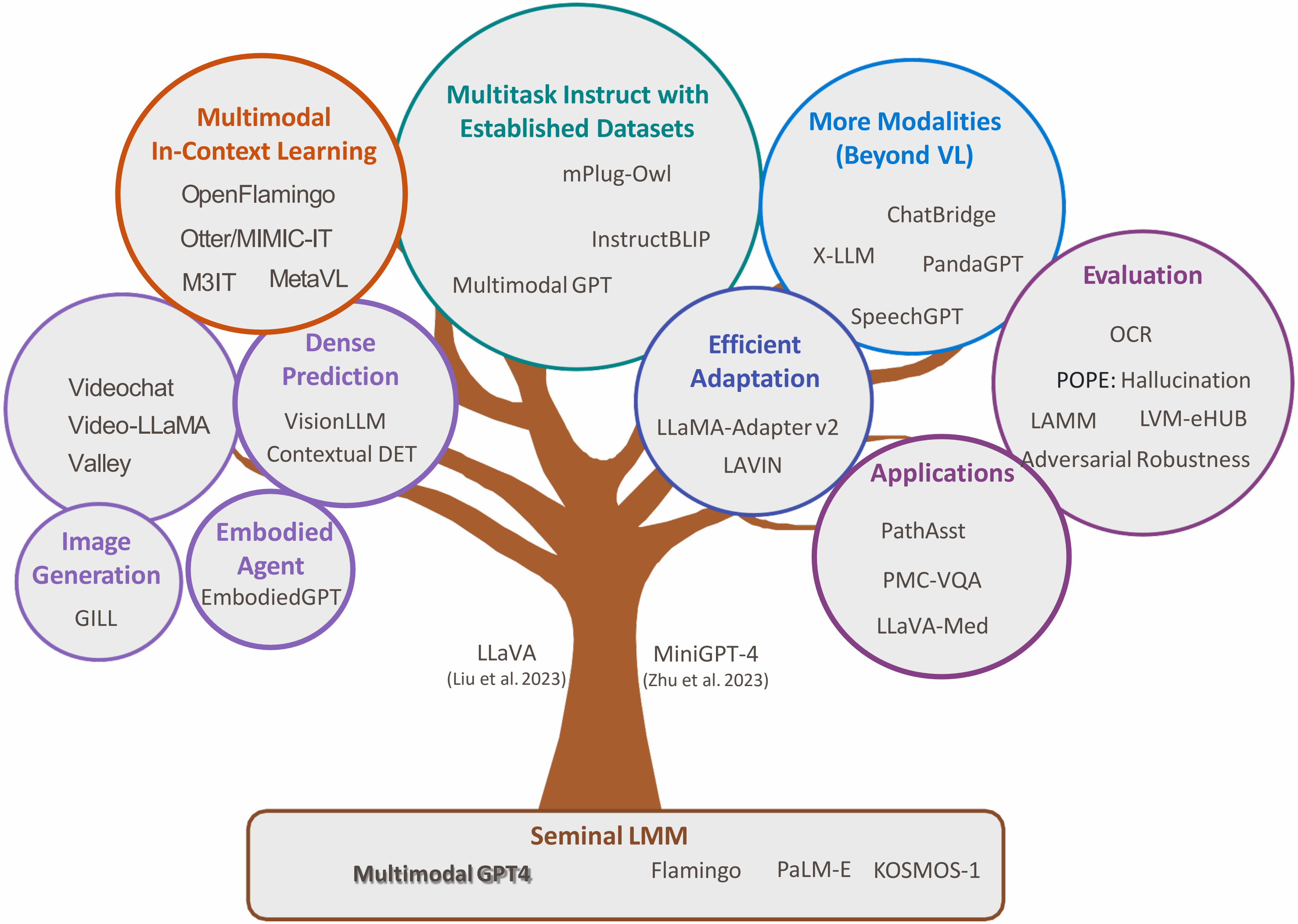
May 22

May 29

June 6

June 13

partial credit by CS311n


Multimodal In-Context Learning
OpenFlamingo
Otter/MIMIC-IT
M3IT
MetaVL
Multitask Instruct with Established Datasets
mPlug-Owl
InstructBLIP
Multimodal GPT
More Modalities (Beyond VL)
ChatBridge
X-LLM
PandaGPT
SpeechGPT
Evaluation
OCR
POPE: Hallucination
LAMM
LVM-eHUB
Adversarial Robustness
Videochat
Video-LLaMA
Valley
Dense Prediction
VisionLLM
Contextual DET
Efficient Adaptation
LLaMA-Adapter v2
LAVIN
Applications
PathAsst
PMC-VQA
LLaVA-Med
Image Generation
GILL
Embodied Agent
EmbodiedGPT
LLaVA
(Liu et al. 2023)
MiniGPT-4
(Zhu et al. 2023)
Seminal LMM
Multimodal GPT4
Flamingo
PaLM-E
KOSMOS-1

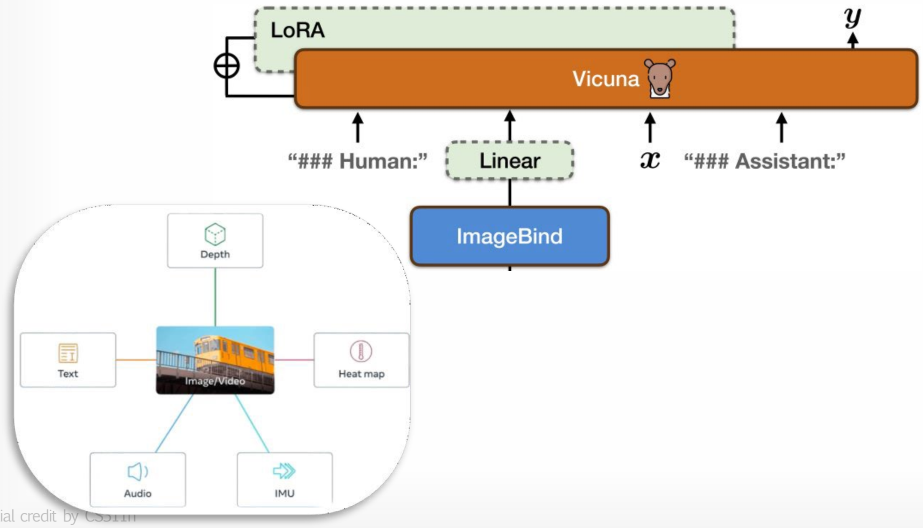
More Modalities (Beyond VL)

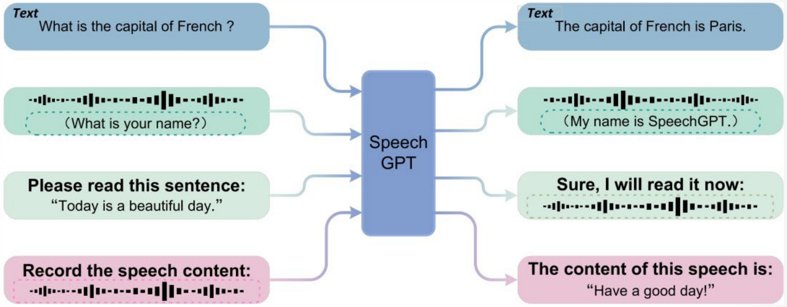
More Modalities

ChatBridge
X-LLM
PandaGPT
SpeechGPT

More Modalities (Beyond VL)

- ChatBridge: Bridging Modalities with Large Language Model as a Language Catalyst
- PandaGPT: One Model To Instruction-Follow Them All
- SpeechGPT: Empowering large language models with intrinsic cross-modal conversational abilities
- X-LLM: Bootstrapping Advanced Large Language Models by Treating Multi-Modalities as Foreign Languages




Text
What is the capital of French ?
(What is your name?)
Please read this sentence:
"Today is a beautiful day."
Record the speech content:
Speech GPT
Text
The capital of French is Paris.
(My name is SpeechGPT.)
Sure, I will read it now:
The content of this speech is:
"Have a good day!"

partial credit by CS511m

Multitask Instruct with Established Academic Datasets/Tasks

Multitask Instruct with Established Datasets

mPlug-Owl
MultiInstruct
InstructBLIP
Multimodal GPT
InstructViT

- MultiInstruct: Improving Multi-Modal Zero-Shot Learning via Instruction Tuning
- mPlug-Owl: Modularization empowers large language models with multimodality
- InstructBLIP: Towards general-purpose vision-language models with instruction tuning
- Multimodal-gpt: A vision and language model for dialogue with humans
- Instruction-ViT: Multi-Modal Prompts for Instruction Learning in ViT
  - Two existing purposes for Instruct Tuning:
    - User-oriented tasks: Daily conversation
    - Academic tasks: Existing datasets

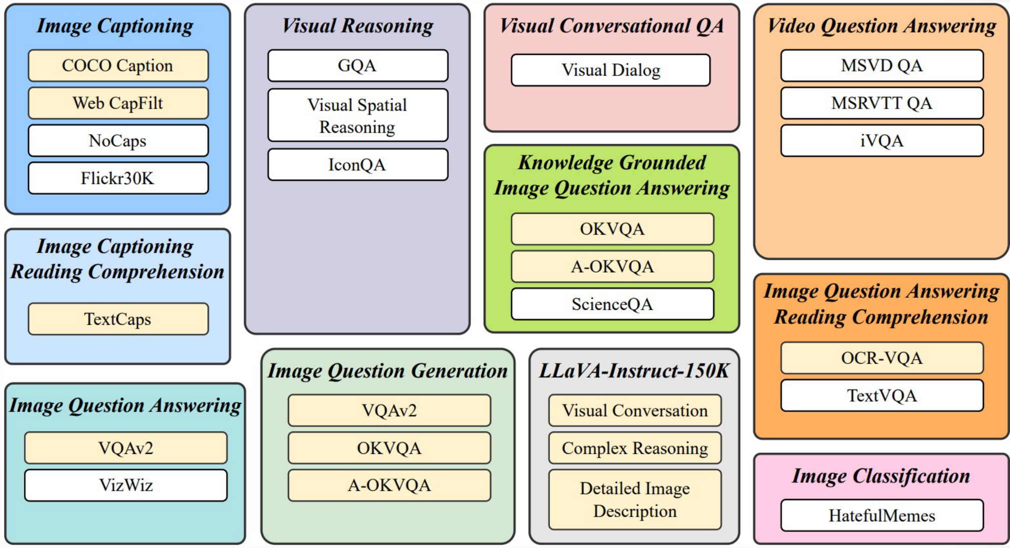


Image Captioning
- COCO Caption
- Web CapFilt
- NoCaps
- Flickr30K

Image Captioning Reading Comprehension
- TextCaps

Image Question Answering
- VQAv2
- VizWiz

Visual Reasoning
- GQA
- Visual Spatial Reasoning
- IconQA

Image Question Generation
- VQAv2
- OKVQA
- A-OKVQA

Visual Conversational QA
- Visual Dialog

Knowledge Grounded Image Question Answering
- OKVQA
- A-OKVQA
- ScienceQA

LLaVA-Instruct-150K
- Visual Conversation
- Complex Reasoning
- Detailed Image Description

Video Question Answering
- MSVD QA
- MSRVTT QA
- iVQA

Image Question Answering Reading Comprehension
- OCR-VQA
- TextVQA

Image Classification
- HatefulMemes

partial credit by CS311n

MultiModal In-Context Learning

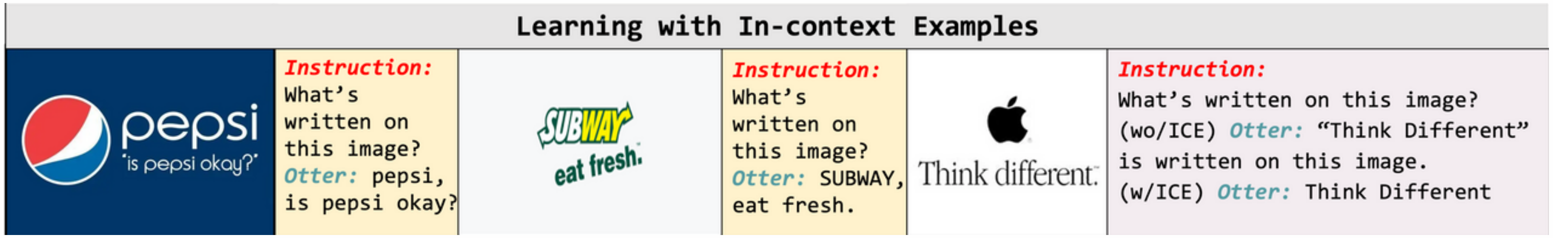
- Otter: A Multi-Modal Model with In-Context Instruction Tuning
- M3IT: A Large-Scale Dataset towards Multi-Modal Multilingual Instruction Tuning
- MetaVL: Transferring In-Context Learning Ability From Language Models to Vision-Language Models

In-Context Learning

OpenFlamingo

Otter/MIMIC-IT

M3IT MetaVL

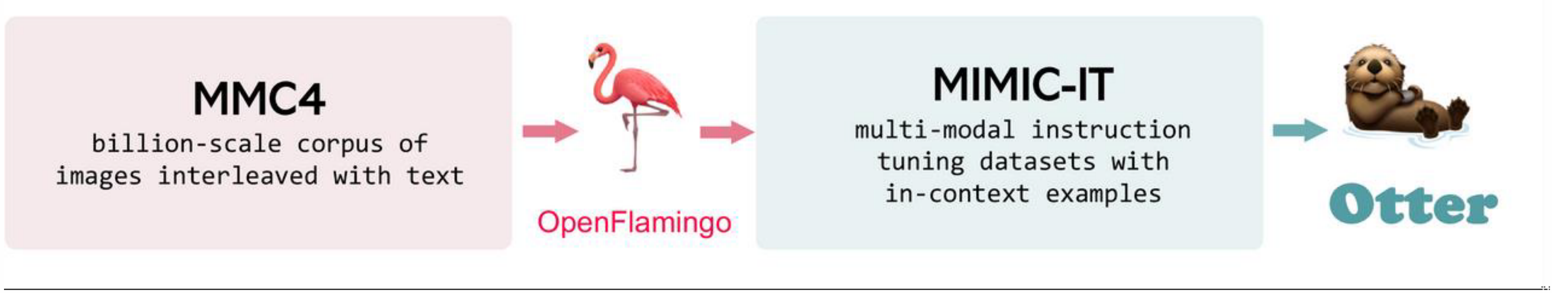


Learning with In-context Examples

Instruction: What's written on this image? *Otter:* pepsi, is pepsi okay?

Instruction: What's written on this image? *Otter:* SUBWAY, eat fresh.

Instruction: What's written on this image? (wo/ICE) *Otter:* "Think Different" is written on this image. (w/ICE) *Otter:* Think Different



MMC4 billion-scale corpus of images interleaved with text → OpenFlamingo → **MIMIC-IT** multi-modal instruction tuning datasets with in-context examples → **Otter**

partial credit by CS311n

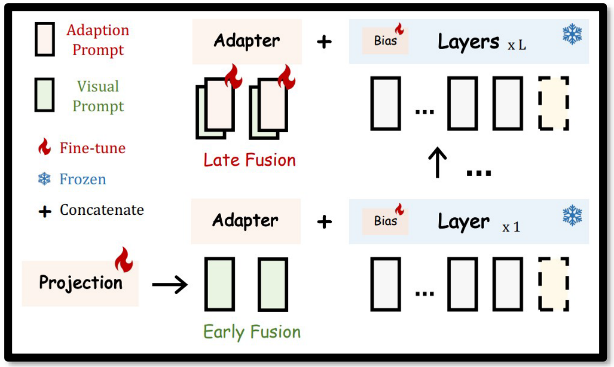
Parameter-Efficient Training

Efficient Adaptation

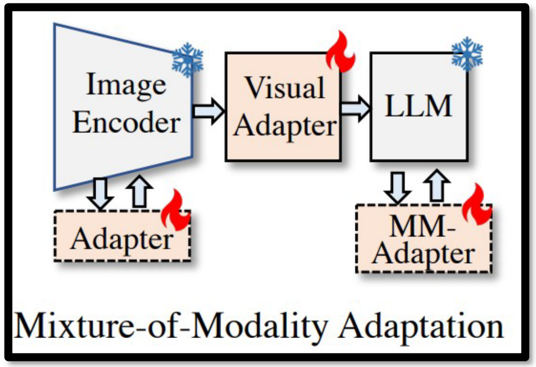
LLaMA-Adapter v2

LAVIN

- LLaMA-Adapter V2: Parameter-Efficient Visual Instruction Model
- Cheap and Quick: Efficient Vision-Language Instruction Tuning for Large Language Models



LLaMA-Adapter V2: 14M parameters



LAVIN: 3.8M parameters

- QLoRA: Efficient Finetuning of Quantized LLMs

Finetuning 65B LLaMA for 24 hours on a single GPU, reaching 99.3% of the performance level of ChatGPT

partial credit by CS311n

Evaluations

Evaluation

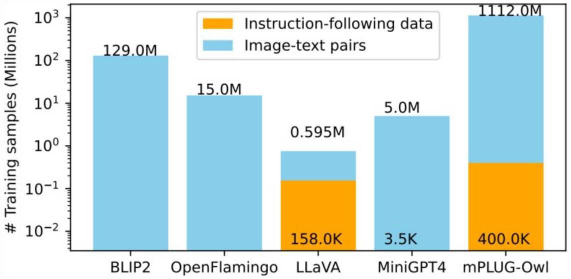
OCR

POPE: Hallucination

LAMM LVM-eHUB

Adversarial Robustness

- Evaluating **Object Hallucination** in Large Vision-Language I
- On Evaluating **Adversarial Robustness** of Large Vision-Langu Models
- On the Hidden Mystery of **OCR** in Large Multimodal Models
- LAMM: Language-Assisted Multi-Modal Instruction-Tuning Dataset, Framework, and **Benchmark**



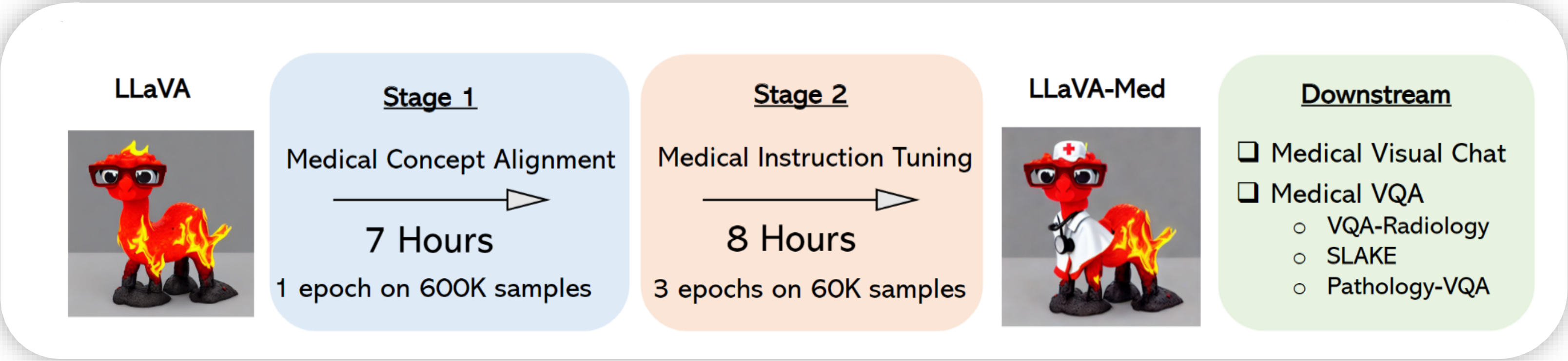
	14 tasks: OCR	10 tasks: OCR + Reasoning
BLIP-2 OPT$_{6.7b}$	62.12	7.31
BLIP-2 FlanT5$_{XXL}$	63.74	13.59
OpenFlamingo	56.68	11.28
LLaVA	53.30	10.37
MiniGPT4	40.12	7.82
mPLUG-Owl	61.79	16.03
Supervised-SoTA	**84.83**	**72.75**

Q: What is the cheapest fruit?

partial credit by CS311n

Applications

- PathAsst: Redefining Pathology through Generative Foundation AI Assistant for Pathology
- PMC-VQA: Visual Instruction Tuning for Medical Visual Question Answering
- LLaVA-Med: Training a Large Language-and-Vision Assistant for Biomedicine in One Day



LLaVA

Stage 1

Medical Concept Alignment

7 Hours

1 epoch on 600K samples

Stage 2

Medical Instruction Tuning

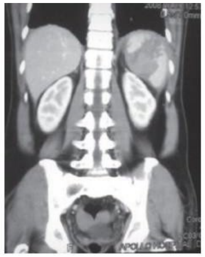
8 Hours

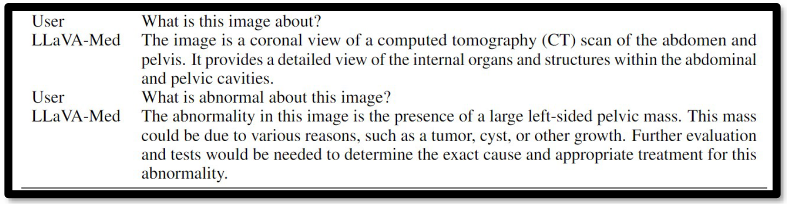
3 epochs on 60K samples

LLaVA-Med

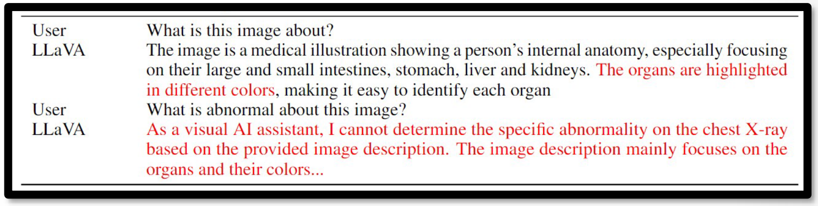
Downstream

- ☐ Medical Visual Chat
- ☐ Medical VQA
  - ○ VQA-Radiology
  - ○ SLAKE
  - ○ Pathology-VQA





User	What is this image about?
LLaVA-Med	The image is a coronal view of a computed tomography (CT) scan of the abdomen and pelvis. It provides a detailed view of the internal organs and structures within the abdominal and pelvic cavities.
User	What is abnormal about this image?
LLaVA-Med	The abnormality in this image is the presence of a large left-sided pelvic mass. This mass could be due to various reasons, such as a tumor, cyst, or other growth. Further evaluation and tests would be needed to determine the exact cause and appropriate treatment for this abnormality.



User	What is this image about?
LLaVA	The image is a medical illustration showing a person's internal anatomy, especially focusing on their large and small intestines, stomach, liver and kidneys. The organs are highlighted in different colors, making it easy to identify each organ
User	What is abnormal about this image?
LLaVA	As a visual AI assistant, I cannot determine the specific abnormality on the chest X-ray based on the provided image description. The image description mainly focuses on the organs and their colors...

partial credit by CS311n

Are we close or surpassing GPT-4?

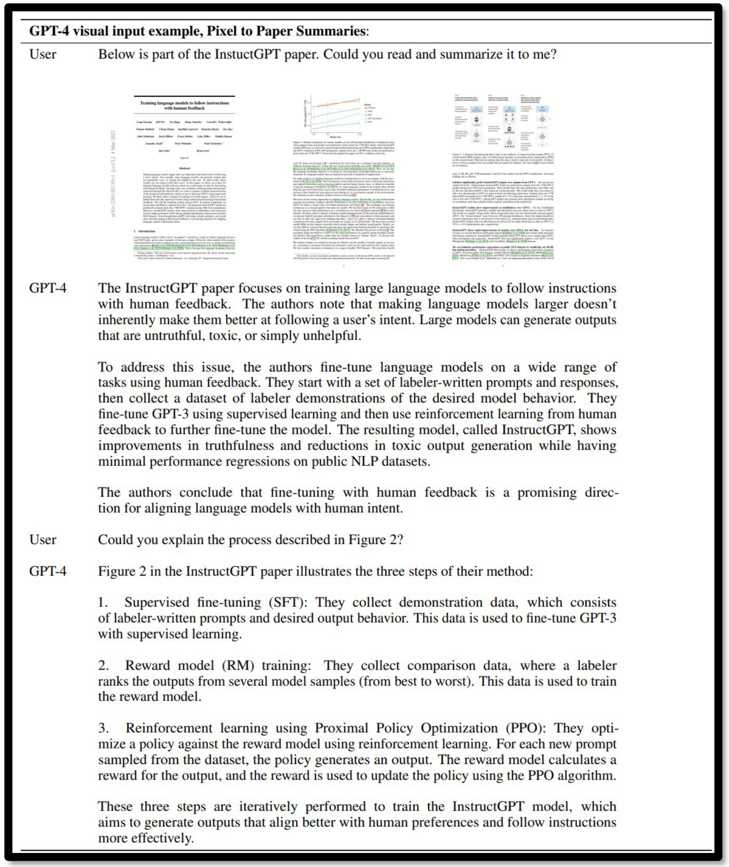
In terms of prototyping new capabilities

A large gap remains…

In terms of scaling the capability

partial credit by CS311n

Strong abilities in
Reading multiple high-resolution images, long sequence
Responding with domain knowledge



GPT-4 visual input example, Pixel to Paper Summaries:

User Below is part of the InstuctGPT paper. Could you read and summarize it to me?

GPT-4 The InstructGPT paper focuses on training large language models to follow instructions with human feedback. The authors note that making language models larger doesn't inherently make them better at following a user's intent. Large models can generate outputs that are untruthful, toxic, or simply unhelpful.

To address this issue, the authors fine-tune language models on a wide range of tasks using human feedback. They start with a set of labeler-written prompts and responses, then collect a dataset of labeler demonstrations of the desired model behavior. They fine-tune GPT-3 using supervised learning and then use reinforcement learning from human feedback to further fine-tune the model. The resulting model, called InstructGPT, shows improvements in truthfulness and reductions in toxic output generation while having minimal performance regressions on public NLP datasets.

The authors conclude that fine-tuning with human feedback is a promising direction for aligning language models with human intent.

User Could you explain the process described in Figure 2?

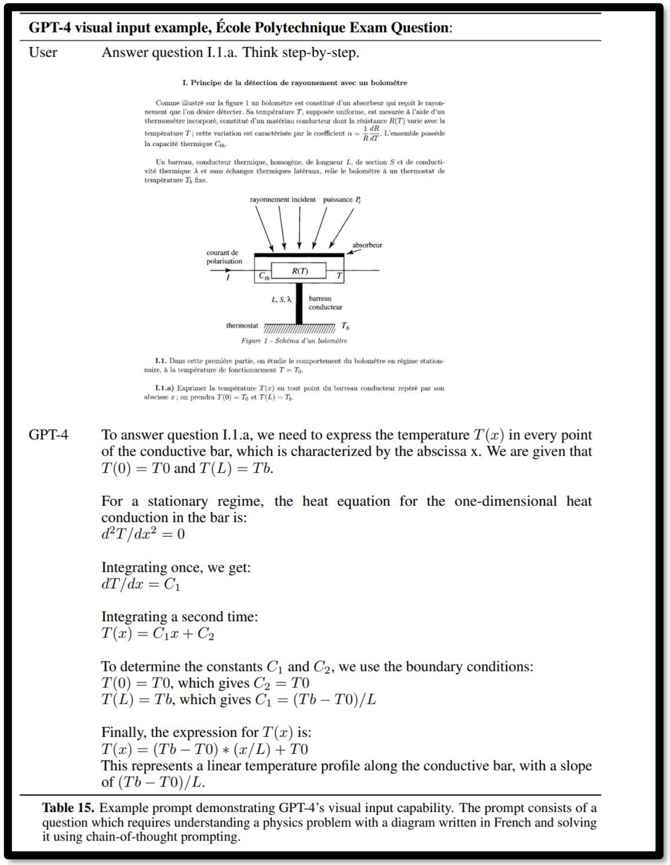
GPT-4 Figure 2 in the InstructGPT paper illustrates the three steps of their method:

1. Supervised fine-tuning (SFT): They collect demonstration data, which consists of labeler-written prompts and desired output behavior. This data is used to fine-tune GPT-3 with supervised learning.

2. Reward model (RM) training: They collect comparison data, where a labeler ranks the outputs from several model samples (from best to worst). This data is used to train the reward model.

3. Reinforcement learning using Proximal Policy Optimization (PPO): They optimize a policy against the reward model using reinforcement learning. For each new prompt sampled from the dataset, the policy generates an output. The reward model calculates a reward for the output, and the reward is used to update the policy using the PPO algorithm.

These three steps are iteratively performed to train the InstructGPT model, which aims to generate outputs that align better with human preferences and follow instructions more effectively.



GPT-4 visual input example, École Polytechnique Exam Question:

User Answer question I.1.a. Think step-by-step.

I. Principe de la détection de rayonnement avec un bolomètre

Comme illustré sur la figure 1 un bolomètre est constitué d'un absorbeur qui reçoit le rayonnement que l'on désire détecter. Sa température T, supposée uniforme, est mesurée à l'aide d'un thermomètre incorporé, constitué d'un matériau conducteur dont la résistance $R(T)$ varie avec la température T ; cette variation est caractérisée par le coefficient $\alpha = \frac{1}{R}\frac{dR}{dT}$. L'ensemble possède la capacité thermique C_{th}.

Un barreau, conducteur thermique, homogène, de longueur L, de section S et de conductivité thermique λ et sans échanges thermiques latéraux, relie le bolomètre à un thermostat de température T_b fixe.

Figure 1 - Schéma d'un bolomètre

I.1. Dans cette première partie, on étudie le comportement du bolomètre en régime stationnaire, à la température de fonctionnement $T = T_0$.

I.1.a) Exprimer la température $T(x)$ en tout point du barreau conducteur repéré par son abscisse x ; on prendra $T(0) = T_0$ et $T(L) = T_b$.

GPT-4 To answer question I.1.a, we need to express the temperature $T(x)$ in every point of the conductive bar, which is characterized by the abscissa x. We are given that $T(0) = T0$ and $T(L) = Tb$.

For a stationary regime, the heat equation for the one-dimensional heat conduction in the bar is:
$d^2T/dx^2 = 0$

Integrating once, we get:
$dT/dx = C_1$

Integrating a second time:
$T(x) = C_1 x + C_2$

To determine the constants C_1 and C_2, we use the boundary conditions:
$T(0) = T0$, which gives $C_2 = T0$
$T(L) = Tb$, which gives $C_1 = (Tb - T0)/L$

Finally, the expression for $T(x)$ is:
$T(x) = (Tb - T0) * (x/L) + T0$
This represents a linear temperature profile along the conductive bar, with a slope of $(Tb - T0)/L$.

Table 15. Example prompt demonstrating GPT-4's visual input capability. The prompt consists of a question which requires understanding a physics problem with a diagram written in French and solving it using chain-of-thought prompting.

partial credit by CS311n

What can we do next as a sustainable research community?

- Industry: Scaling of data/model, New emerging properties etc
- University Labs: Prototypes for new functionalities, Evaluation

- Summary:
  - Strong capabilities of LMM
  - Instruction Tuning from Language to Multimodal
  - Open-Source Prototypes: LLaVA / MiniGPT4
  - Emerging Extensions
  - Benchmarking, Properties and Applications

partial credit by CS311n

APPENDIX

Variational Autoencoders

Now equipped with our encoder and decoder networks, let's work out the (log) data likelihood:

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

Variational Autoencoders

Now equipped with our encoder and decoder networks, let’s work out the (log) data likelihood:

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z \sim q_\phi(z|x^{(i)})} \left[\log p_\theta(x^{(i)}) \right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

Taking expectation wrt. z (using encoder network) will come in handy later

Variational Autoencoders

Now equipped with our encoder and decoder networks, let’s work out the (log) data likelihood:

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \qquad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \qquad \text{(Bayes' Rule)}$$

Variational Autoencoders

Now equipped with our encoder and decoder networks, let’s work out the (log) data likelihood:

$$\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] && (p_\theta(x^{(i)}) \text{ Does not depend on } z) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] && (\text{Bayes' Rule}) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] && (\text{Multiply by constant})
\end{aligned}$$

Variational Autoencoders

Now equipped with our encoder and decoder networks, let's work out the (log) data likelihood:

$$\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad (\text{Bayes' Rule}) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})} \frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad (\text{Multiply by constant}) \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad (\text{Logarithms})
\end{aligned}$$

Variational Autoencoders

Now equipped with our encoder and decoder networks, let’s work out the (log) data likelihood:

$$\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \qquad (p_\theta(x^{(i)}) \text{ Does not depend on } z) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \qquad \text{(Bayes' Rule)} \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \qquad \text{(Multiply by constant)} \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \qquad \text{(Logarithms)} \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))
\end{aligned}$$

Variational Autoencoders

Now equipped with our encoder and decoder networks, let's work out the (log) data likelihood:

$$\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \qquad (p_\theta(x^{(i)}) \text{ Does not depend on } z) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \qquad \text{(Bayes' Rule)} \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \qquad \text{(Multiply by constant)} \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \qquad \text{(Logarithms)} \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))
\end{aligned}$$

The expectation wrt. z (using encoder network) let us write nice KL terms

Variational Autoencoders

Now equipped with our encoder and decoder networks, let’s work out the (log) data likelihood:

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z \sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \qquad (p_\theta(x^{(i)}) \text{ Does not depend on } z) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \qquad (\text{Bayes' Rule}) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})} \frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \qquad (\text{Multiply by constant}) \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \qquad (\text{Logarithms}) \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))
\end{aligned}
$$

Decoder network gives $p_\theta(x|z)$, can compute estimate of this term through sampling. (Sampling differentiable through reparam. trick, see paper.)

This KL term (between Gaussians for encoder and z prior) has nice closed-form solution!

$p_\theta(z|x)$ intractable (saw earlier), can’t compute this KL term :(But we know KL divergence always >= 0.

Variational Autoencoders

Now equipped with our encoder and decoder networks, let's work out the (log) data likelihood:

$$\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad (\text{Bayes' Rule})\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad (\text{Multiply by constant})\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad (\text{Logarithms})\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))
\end{aligned}$$

We want to maximize the data likelihood

Decoder network gives $p_\theta(x|z)$, can compute estimate of this term through sampling. (Sampling differentiable through reparam. trick, see paper.)

This KL term (between Gaussians for encoder and z prior) has nice closed-form solution!

$p_\theta(z|x)$ intractable (saw earlier), can't compute this KL term :(But we know KL divergence always >= 0.

Variational Autoencoders

Now equipped with our encoder and decoder networks, let's work out the (log) data likelihood:

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \qquad (p_\theta(x^{(i)}) \text{ Does not depend on } z) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \qquad \text{(Bayes' Rule)} \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \qquad \text{(Multiply by constant)} \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \qquad \text{(Logarithms)} \\
&= \boxed{\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)}} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{\boxed{\geq 0}}
\end{aligned}
$$

We want to maximize the data likelihood

Tractable lower bound which we can take gradient of and optimize! ($p_\theta(x|z)$ differentiable, KL term differentiable)

Fei-Fei Li & Justin Johnson & Serena Yeung

Variational Autoencoders

Now equipped with our encoder and decoder networks, let’s work out the (log) data likelihood:

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \qquad (p_\theta(x^{(i)}) \text{ Does not depend on } z) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \qquad (\text{Bayes' Rule}) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \qquad (\text{Multiply by constant}) \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \qquad (\text{Logarithms}) \\
&= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{\geq 0}
\end{aligned}
$$

We want to maximize the data likelihood

$$\log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)},\theta,\phi)$$

Variational lower bound (“ELBO”)

$$\theta^*,\phi^* = \arg\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)},\theta,\phi)$$

Training: Maximize lower bound

Variational Autoencoders

Now equipped with our encoder and decoder networks, let’s work out the (log) data likelihood:

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \qquad (p_\theta(x^{(i)}) \text{ Does not depend on } z) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \qquad (\text{Bayes' Rule}) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \qquad (\text{Multiply by constant}) \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \qquad (\text{Logarithms}) \\
&= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{\geq 0}
\end{aligned}
$$

Reconstruct the input data

Make approximate posterior distribution close to prior

$$\log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)},\theta,\phi)$$

Variational lower bound (“ELBO”)

$$\theta^*, \phi^* = \arg\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)},\theta,\phi)$$

Training: Maximize lower bound

Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Let’s look at computing the bound (forward pass) for a given minibatch of input data

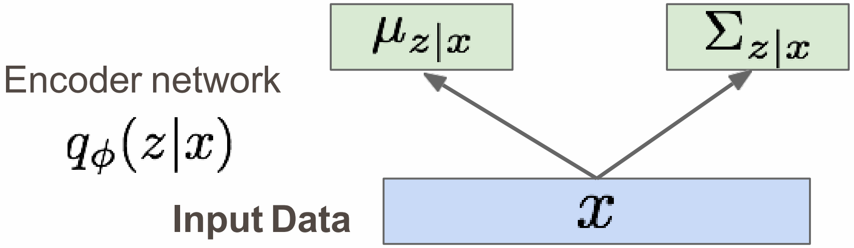
Input Data

x

Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)}$$



Encoder network

$q_\phi(z|x)$

Input Data

x

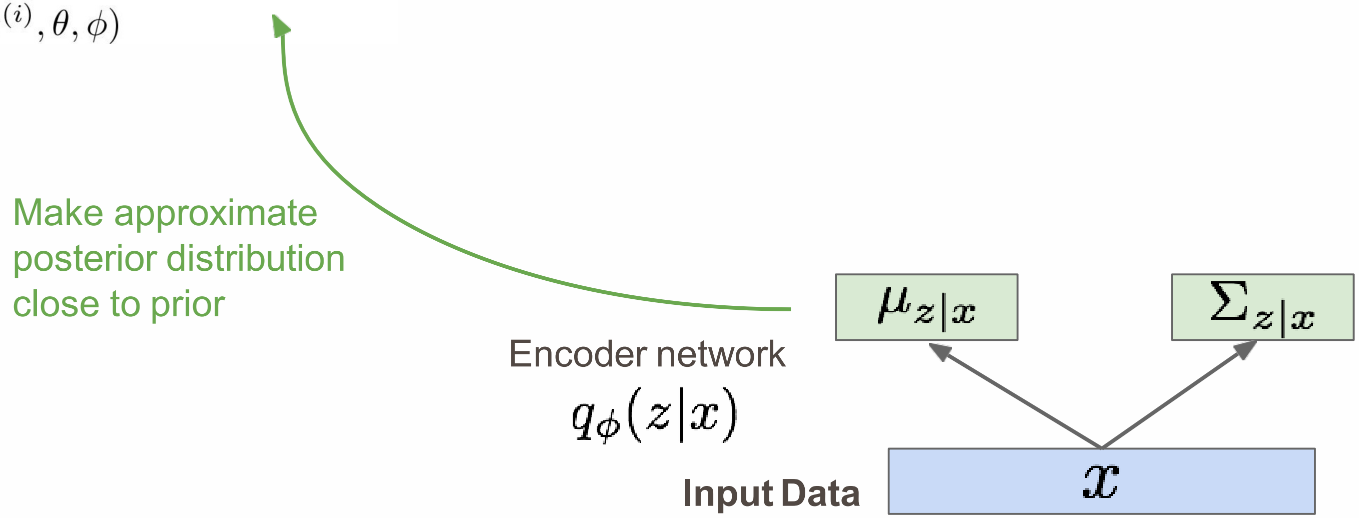
$\mu_{z|x}$

$\Sigma_{z|x}$

Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)}$$



Make approximate posterior distribution close to prior

Encoder network

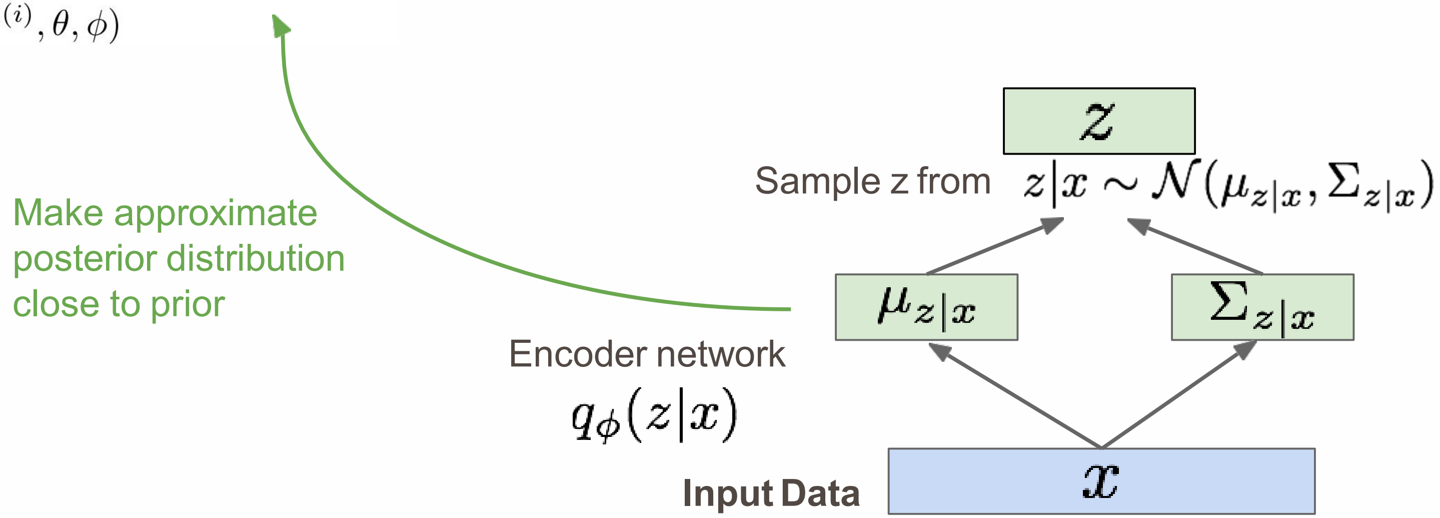
$q_\phi(z|x)$

Input Data

Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)}$$



Make approximate posterior distribution close to prior

Sample z from $z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$

Encoder network $q_\phi(z|x)$

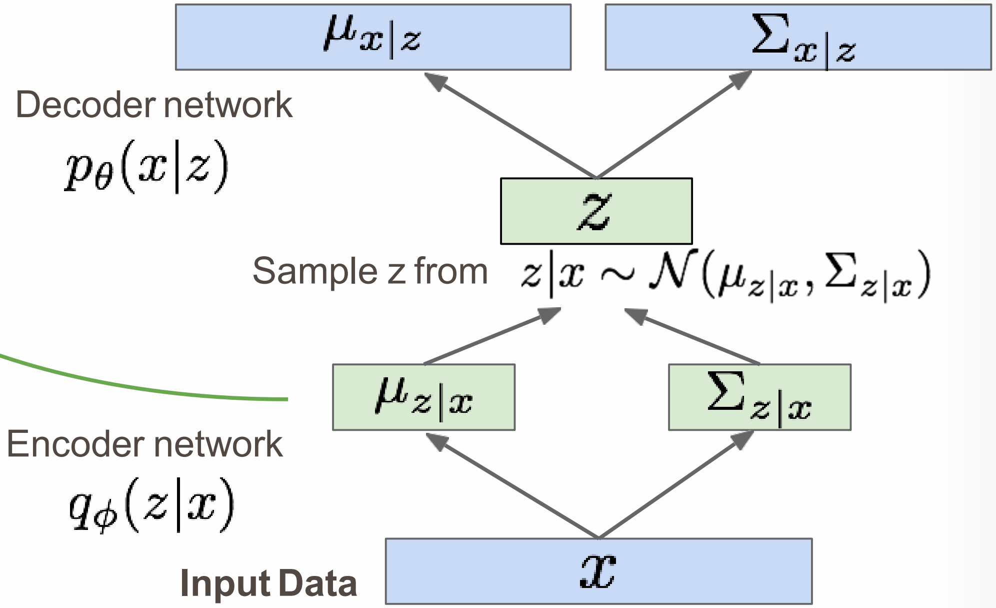
Input Data

Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)}$$

Make approximate posterior distribution close to prior



$\mu_{x|z}$

$\Sigma_{x|z}$

Decoder network

$p_\theta(x|z)$

z

Sample z from $z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$

$\mu_{z|x}$

$\Sigma_{z|x}$

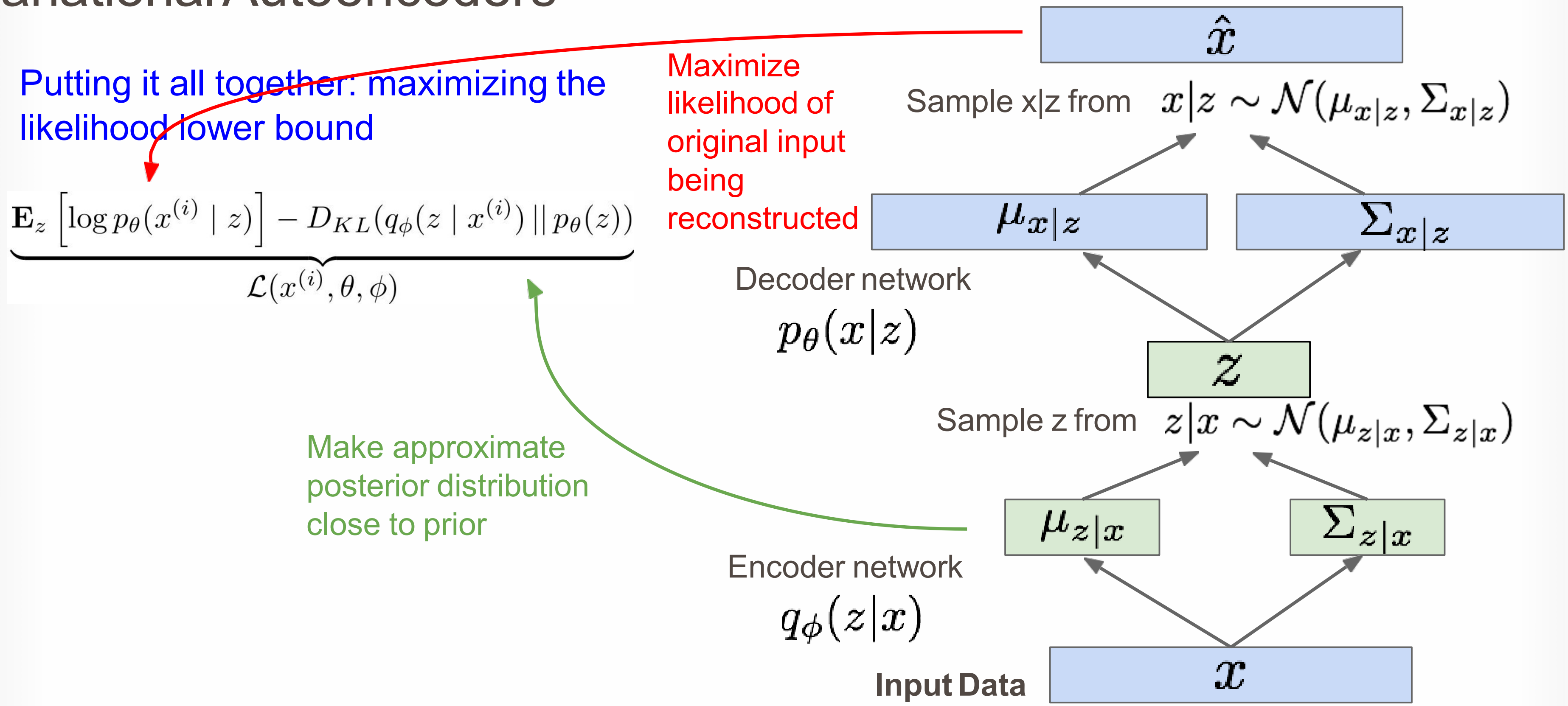
Encoder network

$q_\phi(z|x)$

Input Data

x

Variational Autoencoders



Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)}$$

Maximize likelihood of original input being reconstructed

Make approximate posterior distribution close to prior

Variational Autoencoders

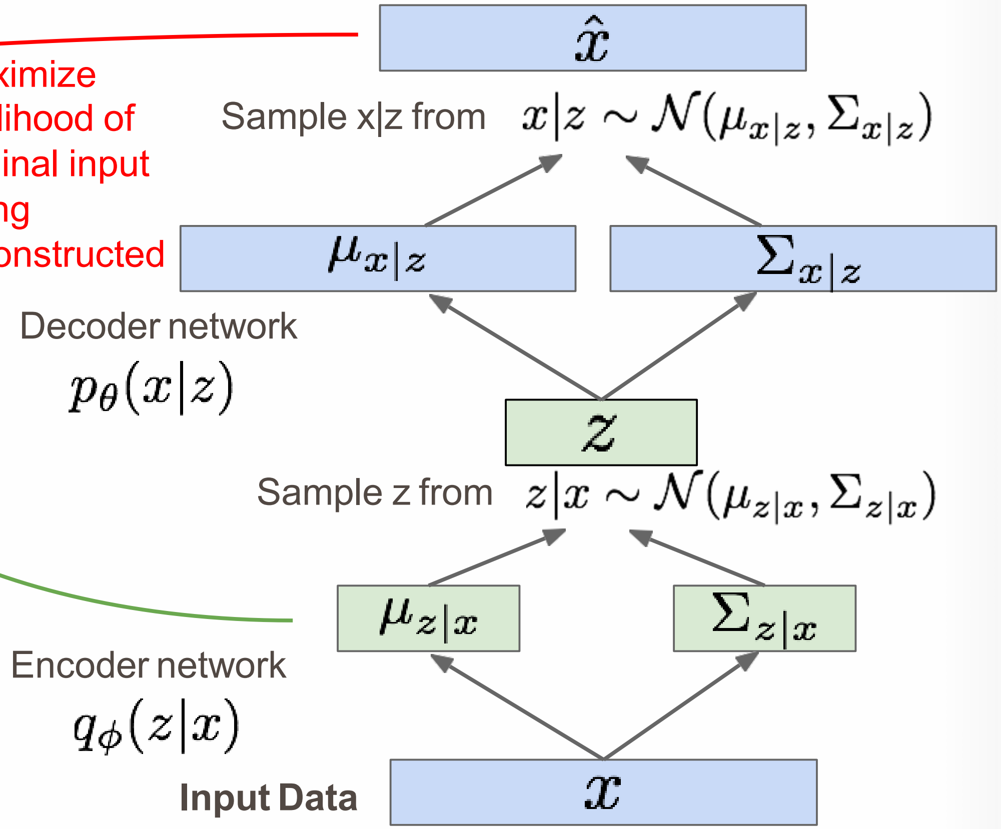
Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Maximize likelihood of original input being reconstructed

Make approximate posterior distribution close to prior

For every minibatch of input data: compute this forward pass, and then backprop!



$\hat{x}$

Sample x|z from $x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$

$\mu_{x|z}$

$\Sigma_{x|z}$

Decoder network

$p_\theta(x|z)$

z

Sample z from $z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$

$\mu_{z|x}$

$\Sigma_{z|x}$

Encoder network

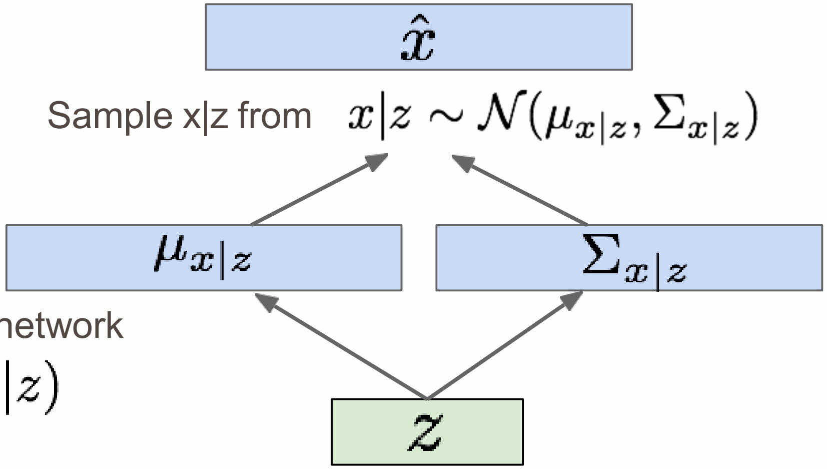
$q_\phi(z|x)$

Input Data

x

Variational Autoencoders: Generating Data!

Use decoder network. Now sample z from prior!



Sample x|z from $x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$

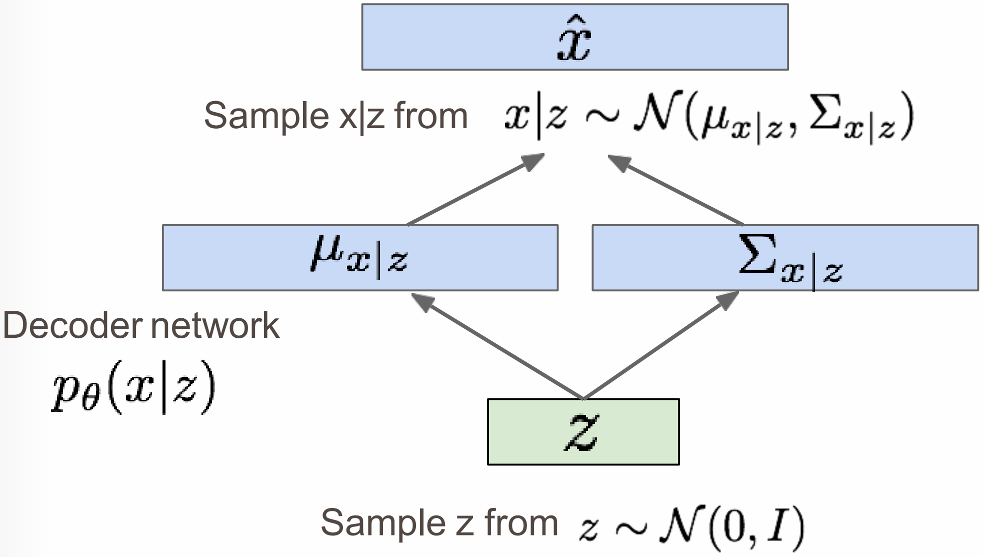
Decoder network $p_\theta(x|z)$

Sample z from $z \sim \mathcal{N}(0, I)$

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

Variational Autoencoders: Generating Data!

Use decoder network. Now sample z from prior!



$\hat{x}$

Sample x|z from $x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$

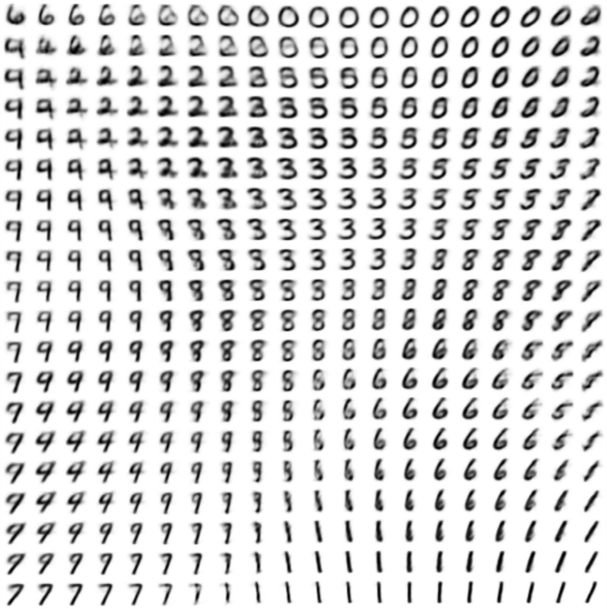
$\mu_{x|z}$ $\Sigma_{x|z}$

Decoder network

$p_\theta(x|z)$

z

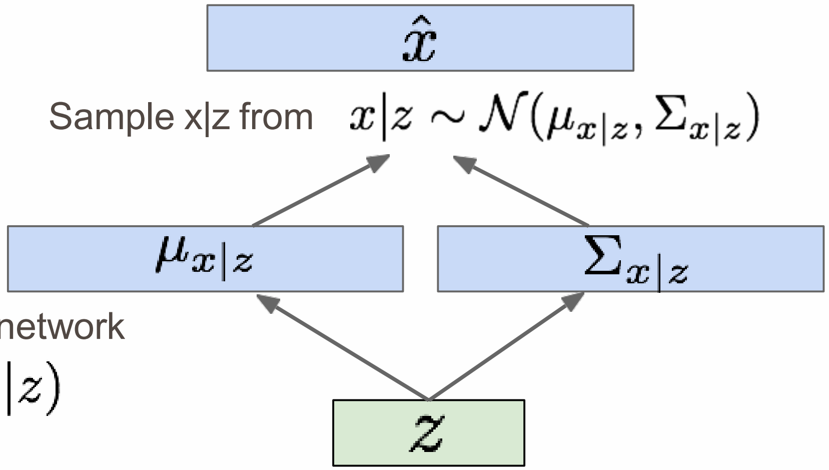
Sample z from $z \sim \mathcal{N}(0, I)$



Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

Variational Autoencoders: Generating Data!

Use decoder network. Now sample z from prior!



Sample x|z from $x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$

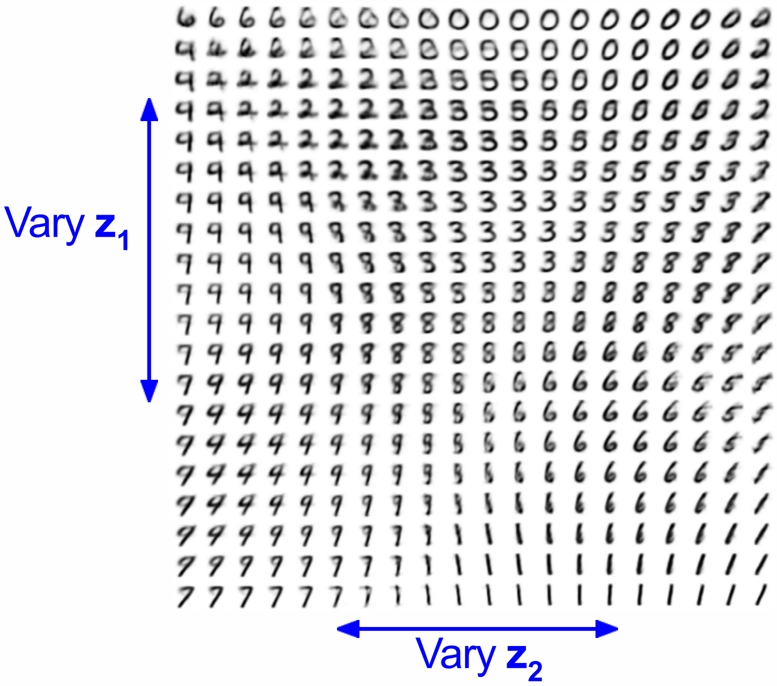
Decoder network

$p_\theta(x|z)$

Sample z from $z \sim \mathcal{N}(0, I)$

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

Data manifold for 2-d **z**



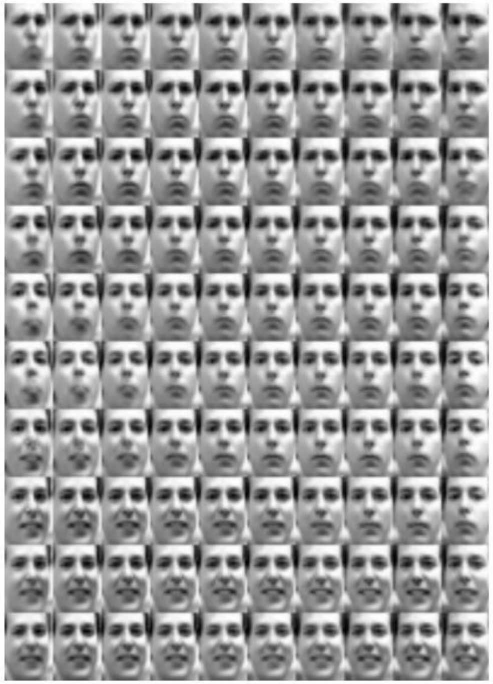
Variational Autoencoders: Generating Data!

Diagonal prior on **z** => independent latent variables

Different dimensions of **z** encode interpretable factors of variation

Degree of smile

Vary $\mathbf{z}_1$



Vary $\mathbf{z}_2$

Head pose

Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

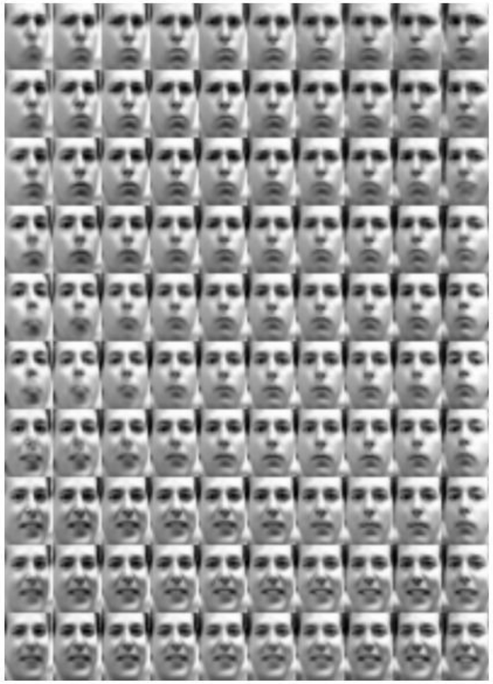
Variational Autoencoders: Generating Data!

Diagonal prior on **z** => independent latent variables

Different dimensions of **z** encode interpretable factors of variation

Degree of smile

Vary $\mathbf{z}_1$



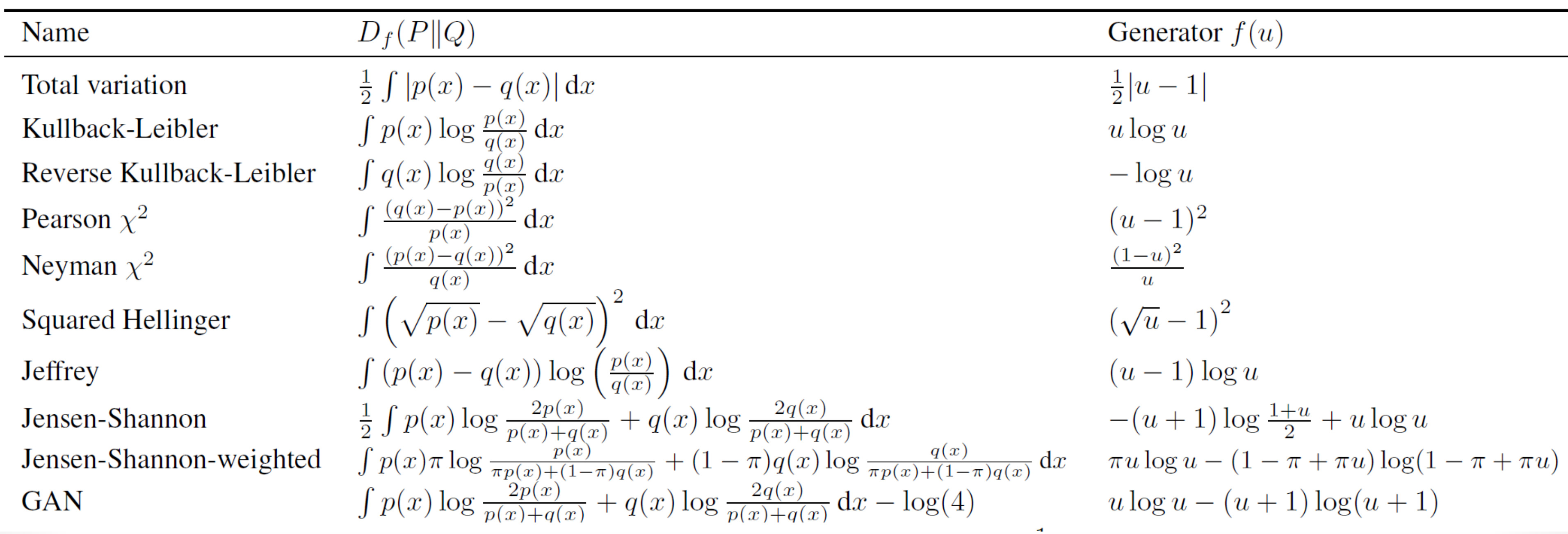
Vary $\mathbf{z}_2$

Head pose

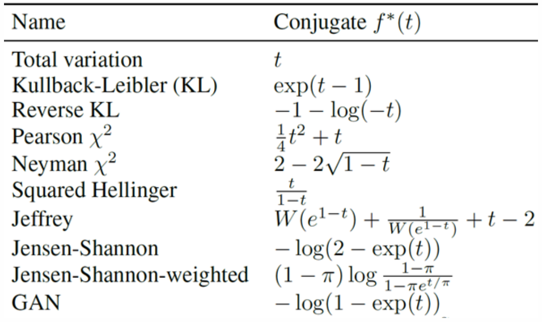
Also good feature representation that can be computed using $q_\phi(z|x)$!

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

Can we use other divergence?



Name	$D_f(P\|Q)$	Generator $f(u)$
Total variation	$\frac{1}{2}\int \lvert p(x)-q(x)\rvert \,\mathrm{d}x$	$\frac{1}{2}\lvert u-1\rvert$
Kullback-Leibler	$\int p(x)\log\frac{p(x)}{q(x)}\,\mathrm{d}x$	$u\log u$
Reverse Kullback-Leibler	$\int q(x)\log\frac{q(x)}{p(x)}\,\mathrm{d}x$	$-\log u$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)}\,\mathrm{d}x$	$(u-1)^2$
Neyman χ^2	$\int \frac{(p(x)-q(x))^2}{q(x)}\,\mathrm{d}x$	$\frac{(1-u)^2}{u}$
Squared Hellinger	$\int \left(\sqrt{p(x)}-\sqrt{q(x)}\right)^2\,\mathrm{d}x$	$\left(\sqrt{u}-1\right)^2$
Jeffrey	$\int (p(x)-q(x))\log\left(\frac{p(x)}{q(x)}\right)\,\mathrm{d}x$	$(u-1)\log u$
Jensen-Shannon	$\frac{1}{2}\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x$	$-(u+1)\log\frac{1+u}{2}+u\log u$
Jensen-Shannon-weighted	$\int p(x)\pi\log\frac{p(x)}{\pi p(x)+(1-\pi)q(x)}+(1-\pi)q(x)\log\frac{q(x)}{\pi p(x)+(1-\pi)q(x)}\,\mathrm{d}x$	$\pi u\log u-(1-\pi+\pi u)\log(1-\pi+\pi u)$
GAN	$\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x-\log(4)$	$u\log u-(u+1)\log(u+1)$



Name	Conjugate $f^*(t)$
Total variation	t
Kullback-Leibler (KL)	$\exp(t-1)$
Reverse KL	$-1-\log(-t)$
Pearson χ^2	$\frac{1}{4}t^2+t$
Neyman χ^2	$2-2\sqrt{1-t}$
Squared Hellinger	$\frac{t}{1-t}$
Jeffrey	$W(e^{1-t})+\frac{1}{W(e^{1-t})}+t-2$
Jensen-Shannon	$-\log(2-\exp(t))$
Jensen-Shannon-weighted	$(1-\pi)\log\frac{1-\pi}{1-\pi e^{t/\pi}}$
GAN	$-\log(1-\exp(t))$

Using the divergence
you want...and take care of it...

partial credit by CS311n