Lecture 22: Generative Modeling

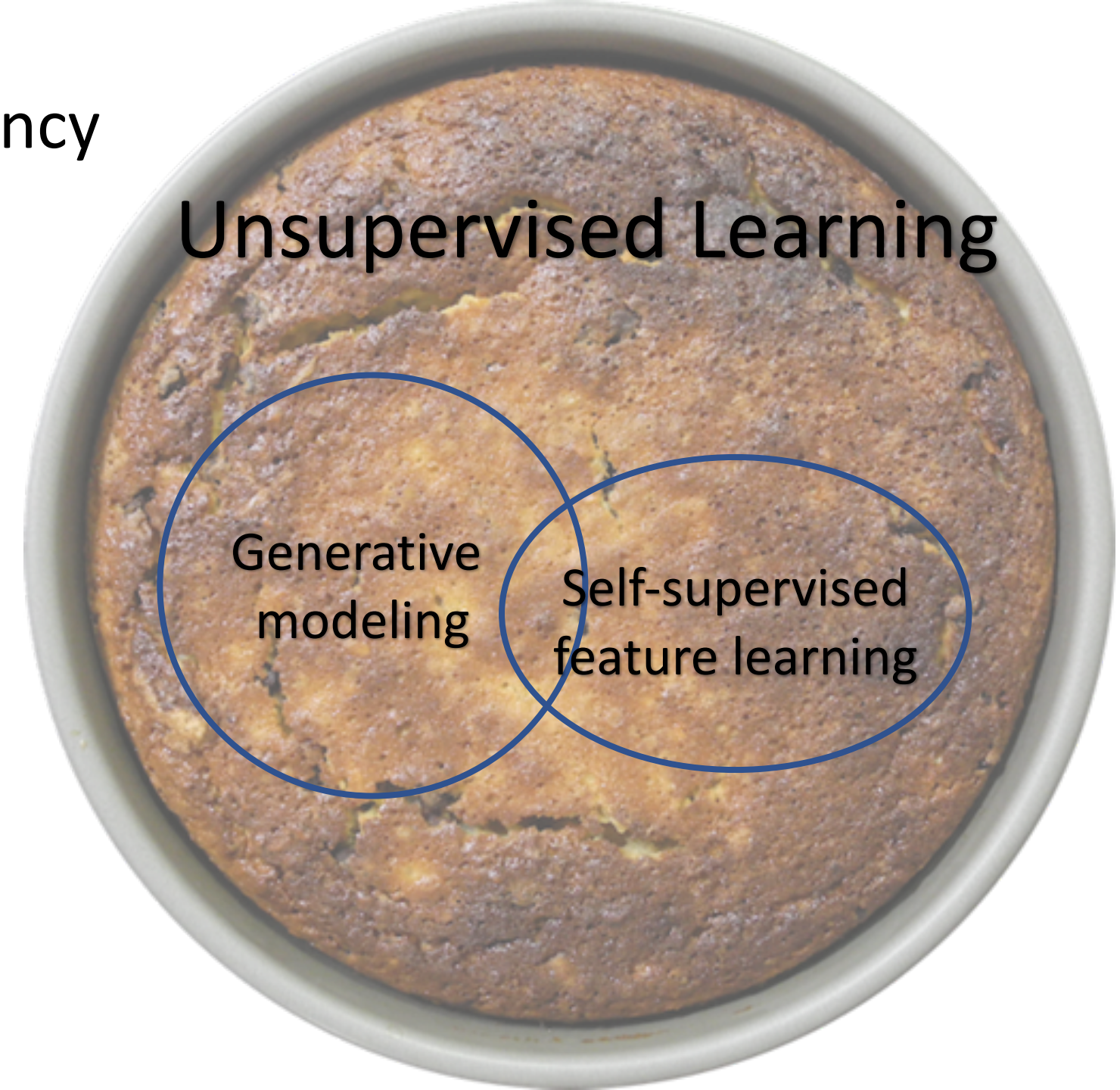
Bolei Zhou
The Chinese University of Hong Kong

Several slides are from Stefan Ermon's cs236 course

Yann LeCun's Cake

- **"Pure" Reinforcement Learning** (cherry)
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning** (icing)
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Unsupervised/Predictive Learning** (cake)
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

Let’s remove the fancy cherry and icings



Richard Feynman: “What I cannot create, I do not understand”

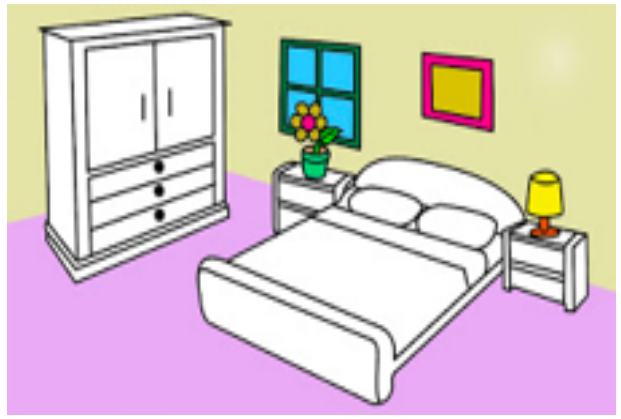
Inference and Generation

Inference

Predictions:
bedroom
wood
foliage

layout

Generation



Statistical Generative Models

- Data: samples such as images of bedrooms
- Prior knowledge: distribution functions (Gaussian, or other non-linear?), loss function (e.g., maximum likelihood, adversarial loss?),

learning

learned probability distribution p(x)

sampling

Progress for Image Generation

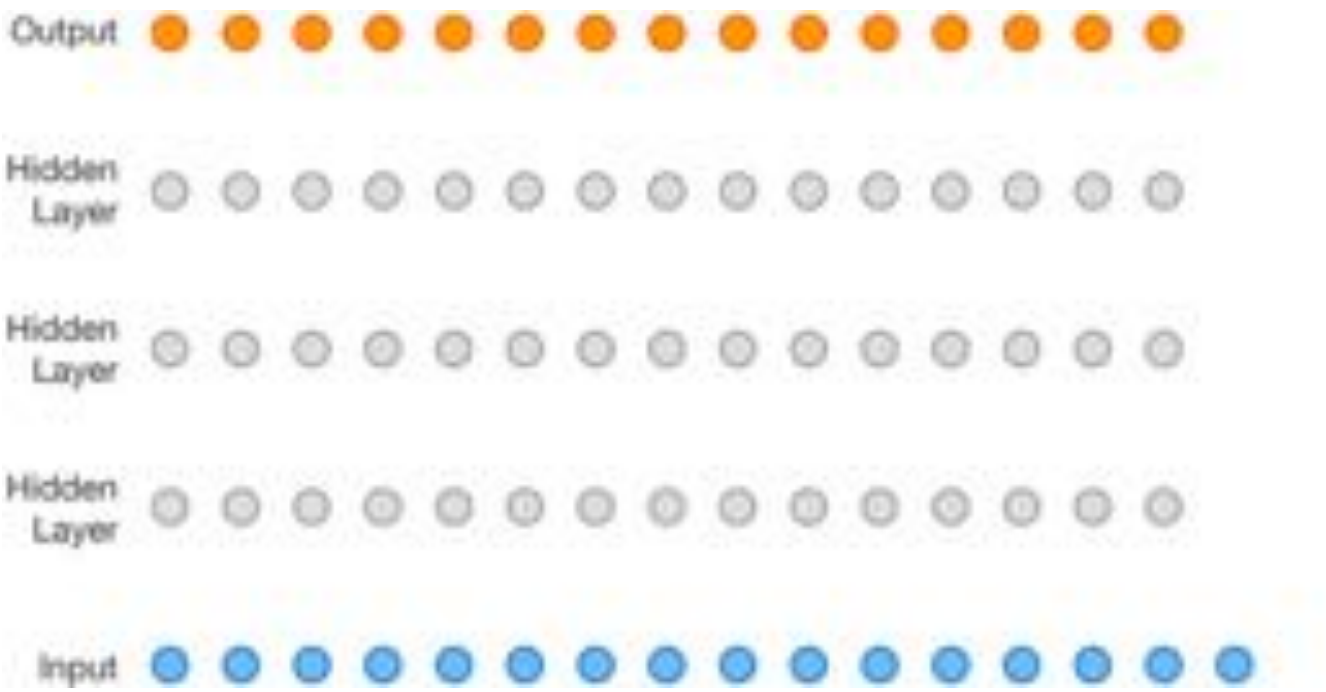
PG-GAN

StyleGAN

BigGAN

Progress for Speech Synthesis

WaveNet



Text to Speech

- Parametric
- Concatenative
- WaveNet

- Unconditional
- Music

van den Oord et al, 2016c

Image Super Resolution

Conditional generative model P(high res image | low res image)

Ledig et al., 2017

Machine Translation

Conditional generative model P(English text| Chinese text)

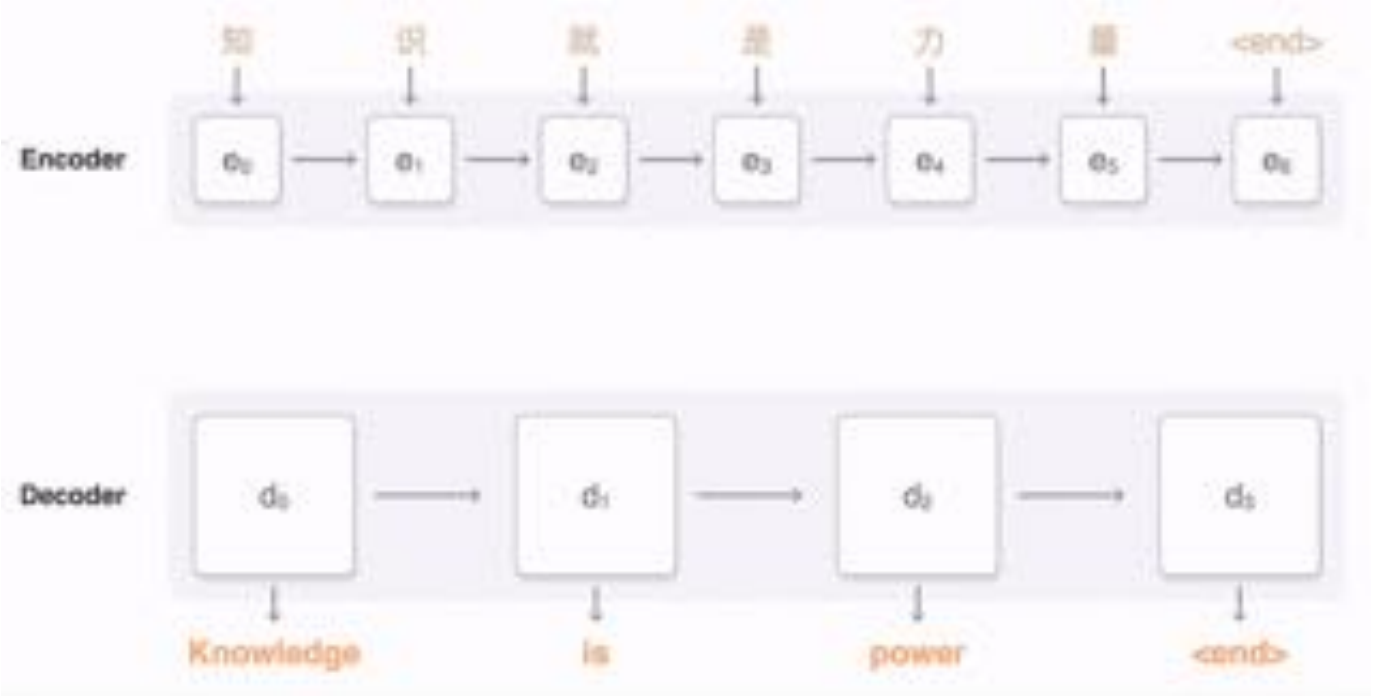
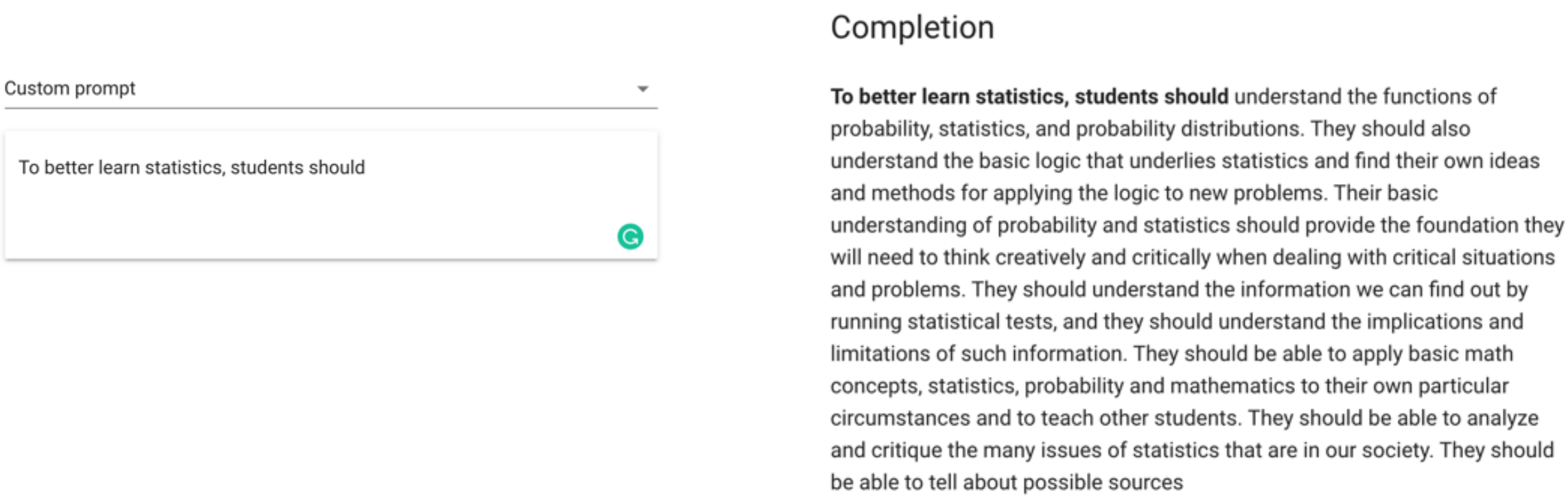


Figure from Google AI research blog.

Paragraph Generation



Radford et al., 2019. Demo from talktotransformer.com

DeepFakes (video+audio)

Image Translation

Conditional generative model P(zebra images| horse images)

Zhu et al., 2017

(Very) Brief look on generative models

- Likehood-based models: Autoregressive models
- Latent variable models: Variational Autoencoder
- Implicit generative models: Generative Adversarial Networks (GANs)
- Latent semantics in GANs

Learning a generative model

- We want to learn a probability distribution p(x) over images x such that we can sample it to generate x

learning

learned probability distribution p(x)

sampling

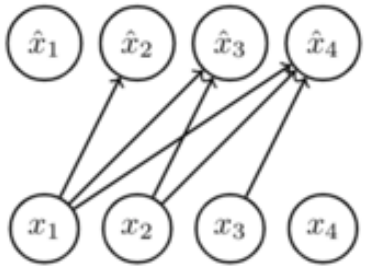
Autoregressive generative model

- Given a dataset of handwritten digits (binarized MNIST)

- Each image has n = 28x28 =784 pixels, with each pixel either being black (0) or white (1)
- Goal: Learn a probability distribution $p(x) = p(x_1, \ldots, x_{784})$

Autoregressive generative model

- We can use chain rule for factorization:
  - $p(x_1, \ldots, x_{784}) = p(x_1)p(x_2|x_1)p(x_3|x_1,x_2)\ldots p(x_n|x_1,\ldots,x_{n-1})$
- Then we can have the following simple autoregressive model
  - $P(x_1=1;a_1) = a_1$, $p(x_1=0;a_1) = 1-a_1$
  - $P(x_2=1|x_1;a_2)= \text{logit}(a_{02}+a_{12}x_1)$
  - $P(x_3=1|x_1,x_2;a_3)=\text{logit}(a_{03}+a_{13}x_1+a_{23}x_2)$



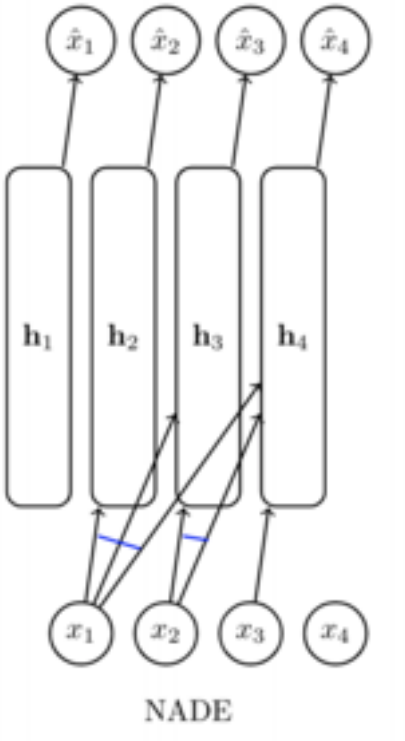
FVSBN

$$\hat{x}_i = p(X_i = 1|x_1, \cdots, x_{i-1}; \boldsymbol{\alpha}^i) = p(X_i = 1|x_{<i}; \boldsymbol{\alpha}^i) = \sigma(\alpha_0^i + \sum_{j=1}^{i-1} \alpha_j^i x_j)$$

Fully visible sigmoid belief network (FVSBN)

NADE: Neural Autoregressive Density Estimation

- To improve the factorization model: use one-layer neural network instead of logistic regression



$$\mathbf{h}_i = \sigma(A_i \mathbf{x}_{<i} + \mathbf{c}_i)$$

$$\hat{x}_i = p(x_i | x_1, \cdots, x_{i-1}; \underbrace{A_i, \mathbf{c}_i, \boldsymbol{\alpha}_i, b_i}_{\text{parameters}}) = \sigma(\boldsymbol{\alpha}_i \mathbf{h}_i + b_i)$$

$$\mathbf{h}_2 = \sigma\left(\underbrace{\begin{pmatrix} \vdots \\ w_1 \\ \vdots \end{pmatrix}}_{A_2} x_1\right) \quad \mathbf{h}_3 = \sigma\left(\underbrace{\begin{pmatrix} \vdots & \vdots \\ w_1 & w_2 \\ \vdots & \vdots \end{pmatrix}}_{A_3} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\right) \quad \mathbf{h}_4 = \sigma\left(\underbrace{\begin{pmatrix} \vdots & \vdots & \vdots \\ w_1 & w_2 & w_3 \\ \vdots & \vdots & \vdots \end{pmatrix}}_{A_3} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}\right)$$

Training data

Samples from the probability

Figure from The Neural Autoregressive Distribution Estimator, 2011.

RNN: Recurrent Neural Networks

Suppose $x_i \in \{h, e, l, o\}$. Use one-hot encoding:

- h encoded as $[1, 0, 0, 0]$, e encoded as $[0, 1, 0, 0]$, etc.

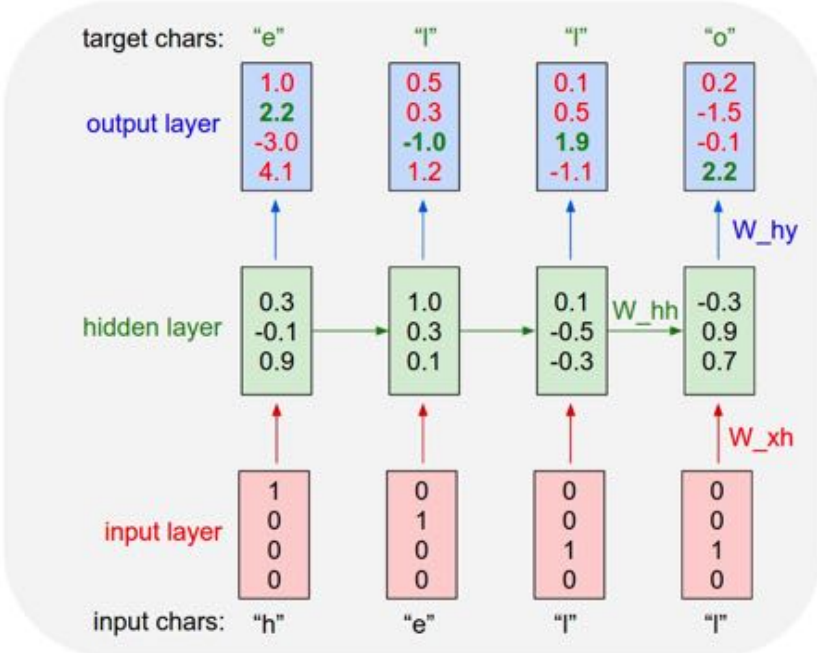
Autoregressive: $p(x = hello) = p(x_1 = h)p(x_2 = e|x_1 = h)p(x_3 = l|x_1 = h, x_2 = e) \cdots p(x_5 = o|x_1 = h, x_2 = e, x_3 = l, x_4 = l)$

For example,

$$p(x_2 = e|x_1 = h) = softmax(o_1) = \frac{\exp(2.2)}{\exp(1.0) + \cdots + \exp(4.1)}$$

$$o_1 = W_{hy} h_1$$

$$h_1 = tanh(W_{hh} h_0 + W_{xh} x_1)$$



http://karpathy.github.io/2015/05/21/rnn-effectiveness/

RNN: Recurrent Neural Networks

RNN model on Wikipedia

```
Naturalism and decision for the majority of Arab countries' capitalide was grounded
by the Irish language by [[John Clair]], [[An Imperial Japanese Revolt]], associated
with Guangzham's sovereignty. His generals were the powerful ruler of the Portugal
in the [[Protestant Immineners]], which could be said to be directly in Cantonese
Communication, which followed a ceremony and set inspired prison, training. The
emperor travelled back to [[Antioch, Perth, October 25|21]] to note, the Kingdom
of Costa Rica, unsuccessful fashioned the [[Thrales]], [[Cynth's Dajoard]], known
in western [[Scotland]], near Italy to the conquest of India with the conflict.
Copyright was the succession of independence in the slop of Syrian influence that
was a famous German movement based on a more popular servicious, non-doctrinal
and sexual power post. Many governments recognize the military housing of the
[[Civil Liberalization and Infantry Resolution 265 National Party in Hungary]],
that is sympathetic to be to the [[Punjab Resolution]]
(PJS)[http://www.humah.yahoo.com/guardian.
```

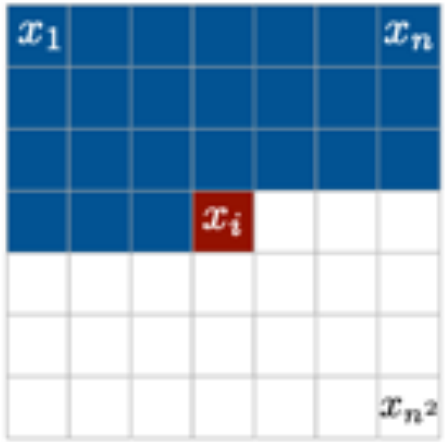
RNN model on Linux source code

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    /*
     * The kernel blank will coeld it to userspace.
     */
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
  segaddr = in_SB(in.addr);
  selector = seg / 16;
  setup_works = true;
  for (i = 0; i < blocks; i++) {
    seq = buf[i++];
    bpf = bd->bd.next + i * search;
    if (fd) {
      current = blocked;
    }
  }
```

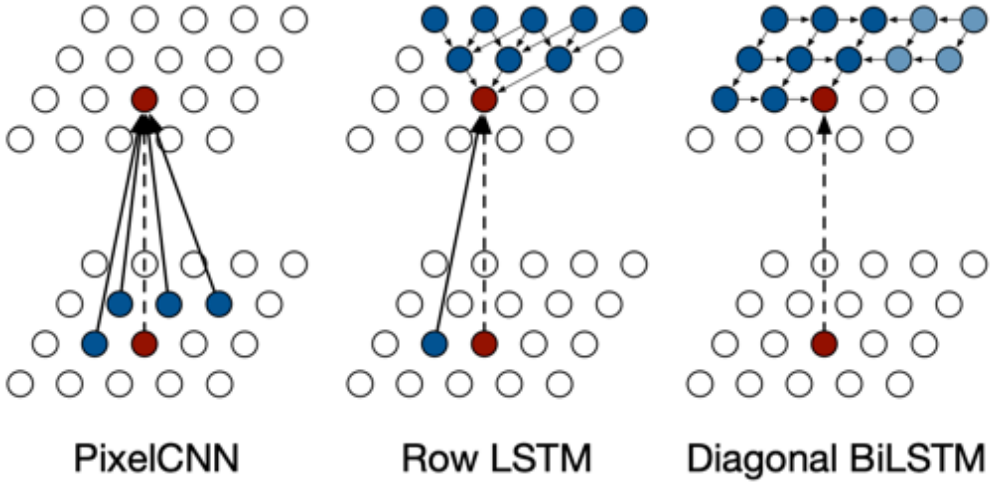
http://karpathy.github.io/2015/05/21/rnn-effectiveness/

PixelRNN and PixelCNN (Oord et al. 2016)

- Model image pixel by pixel using raster scan order



$$p(x_t \mid x_{1:t-1}) = p(x_t^{red} \mid x_{1:t-1})p(x_t^{green} \mid x_{1:t-1}, x_t^{red})p(x_t^{blue} \mid x_{1:t-1}, x_t^{red}, x_t^{green})$$

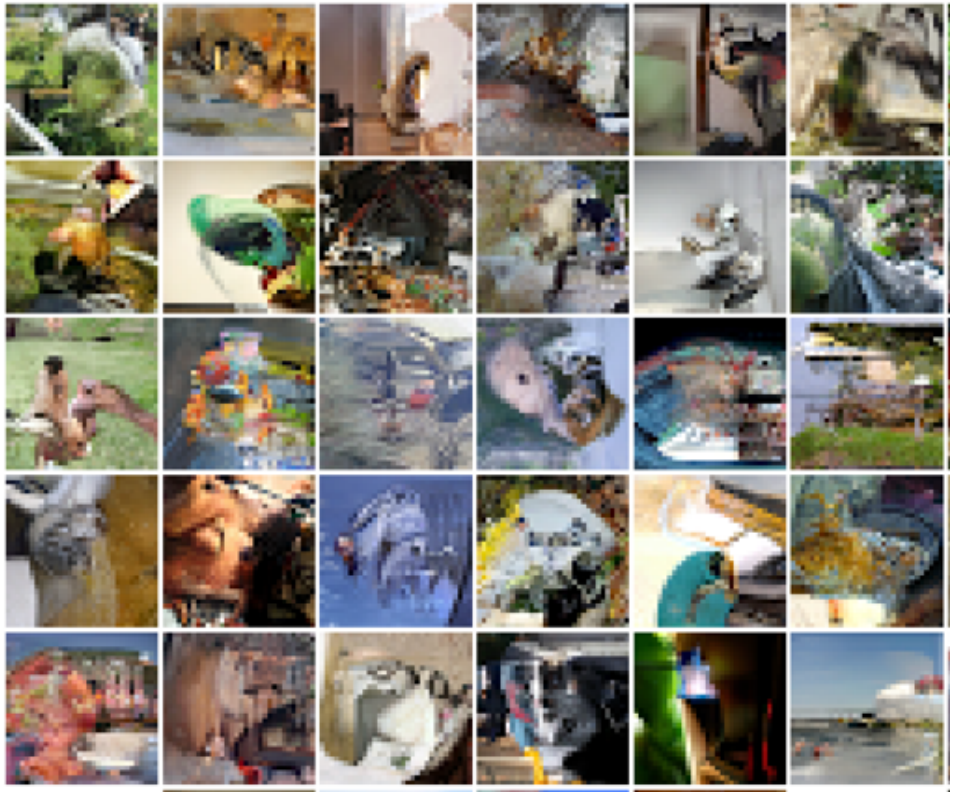


https://arxiv.org/pdf/1601.06759.pdf

PixelCNN and PixelRNN (Oord et al. 2016)

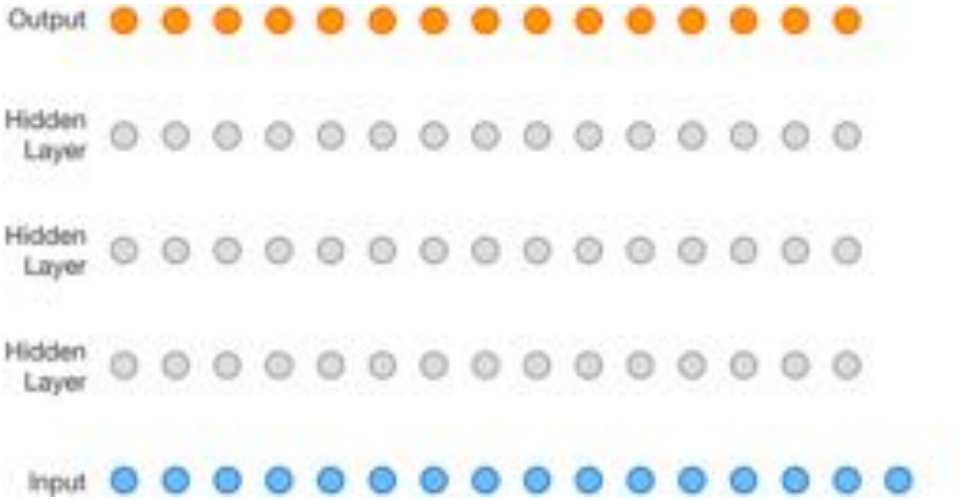
Figure 1. Image completions sampled from a PixelRNN.

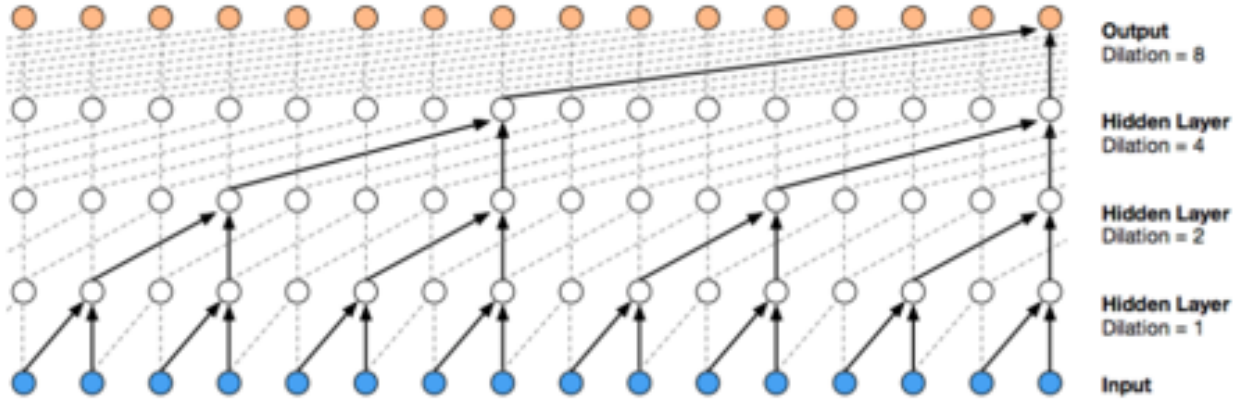
Samples from PixelRNN trained on ImageNet



WaveNet (Oord et al. 2016)

- CNN with 1D dilated convolutions





https://deepmind.com/blog/article/wavenet-generative-model-raw-audio

Summary of Autoregressive Models

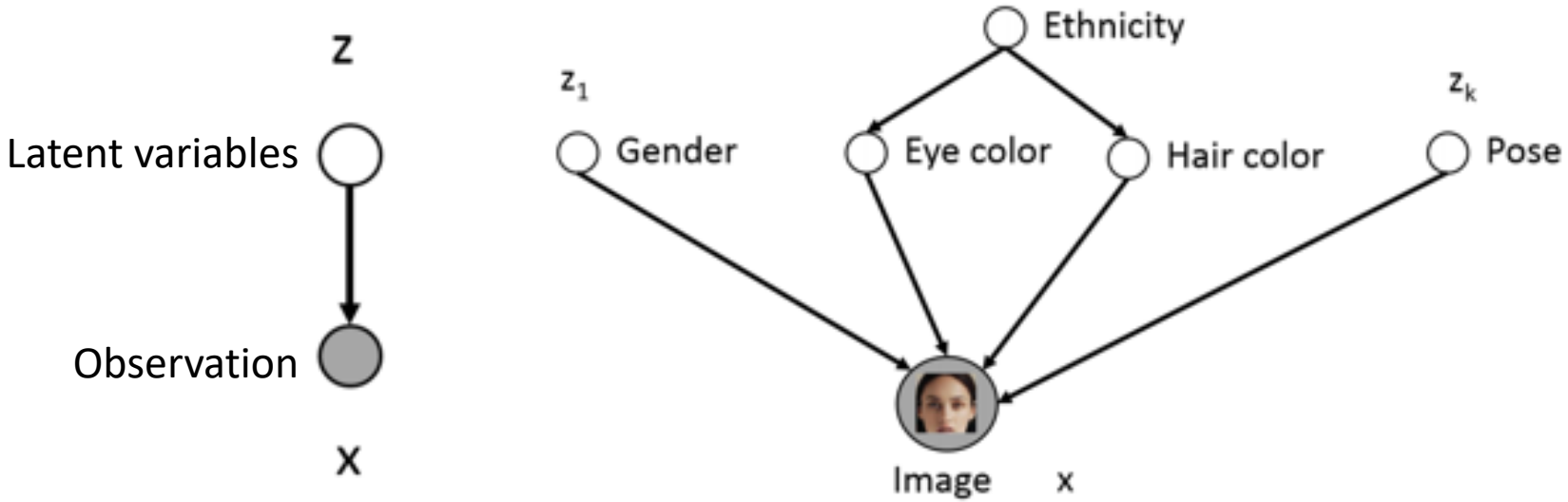
- Autoregressive models:
  - Chain-rule factorization is fully general
  - Compact representation via conditional independence and/or neural parameterizations
- Pros:
  - Easy to evaluate likelihoods
  - Easy to train
- Cons:
  - Requires an ordering
  - Generation is sequential
  - Cannot learn features in an unsupervised way

Latent Variable Models

- Gender, eye color, race, pose, etc are the variation factors in images, unless annotated, they are not explicitly available
- Then we can model them as (low-dimensional) **latent** variables

Latent Variable Models

Data generation process could be driven by the latent variables



Latent Variable Models

- We can define complex models p(x) in terms of simple building blocks p(x|z)
- It can be applied for unsupervised learning tasks easily
- Challenge: much more difficult to learn due to the latent variables.

A mixture of an infinite number of Gaussians:

1. $\mathbf{z} \sim \mathcal{N}(0, I)$
2. $p(\mathbf{x} \mid \mathbf{z}) = \mathcal{N}\left(\mu_\theta(\mathbf{z}), \Sigma_\theta(\mathbf{z})\right)$ where μ_θ,Σ_θ are neural networks
3. Even though $p(\mathbf{x} \mid \mathbf{z})$ is simple, the marginal $p(\mathbf{x})$ is very complex/flexible

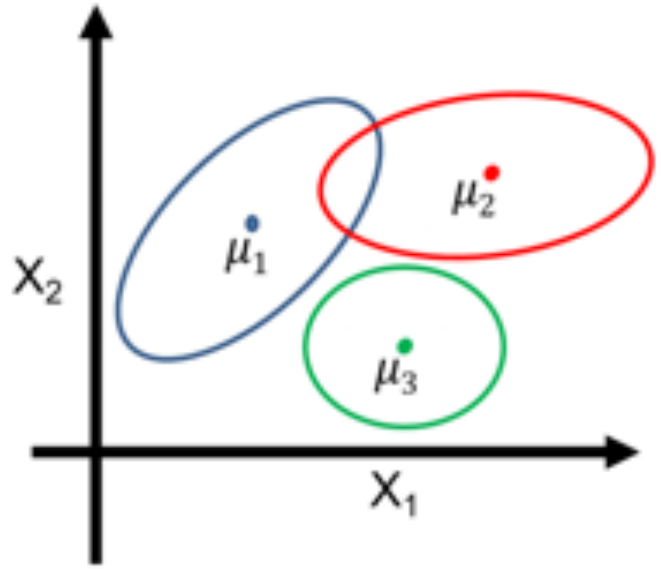
Mixture of Gaussians as a simple latent variable model

Mixture of Gaussians. Bayes net: $\mathbf{z} \rightarrow \mathbf{x}$.

1. $\mathbf{z} \sim \text{Categorical}(1, \cdots, K)$
2. $p(\mathbf{x} \mid \mathbf{z} = k) = \mathcal{N}(\mu_k, \Sigma_k)$

Generative process:

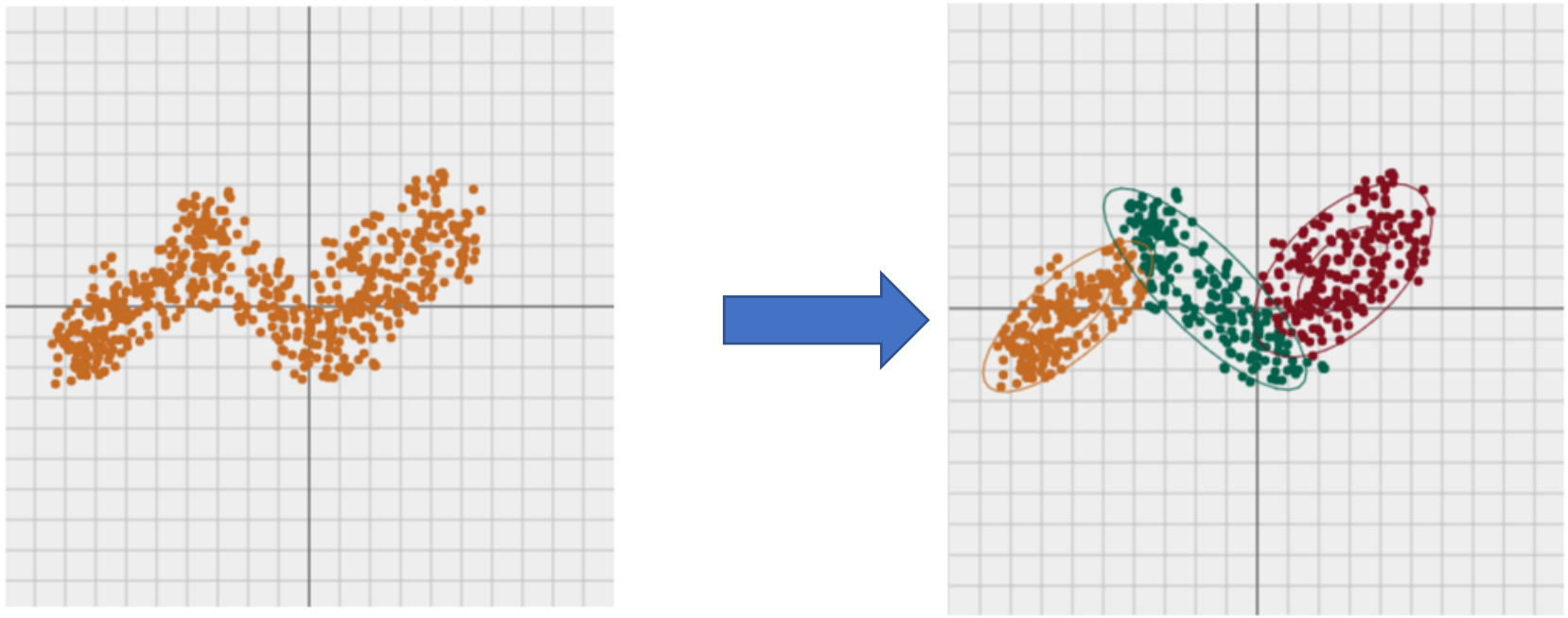
- Pick a mixture component k by sampling z
- Generate a data point by sampling from that Gaussian



Mixture of Gaussians as a simple latent variable model

- Unsupervised learning: the posterior p(z|x) identifies the mixture component

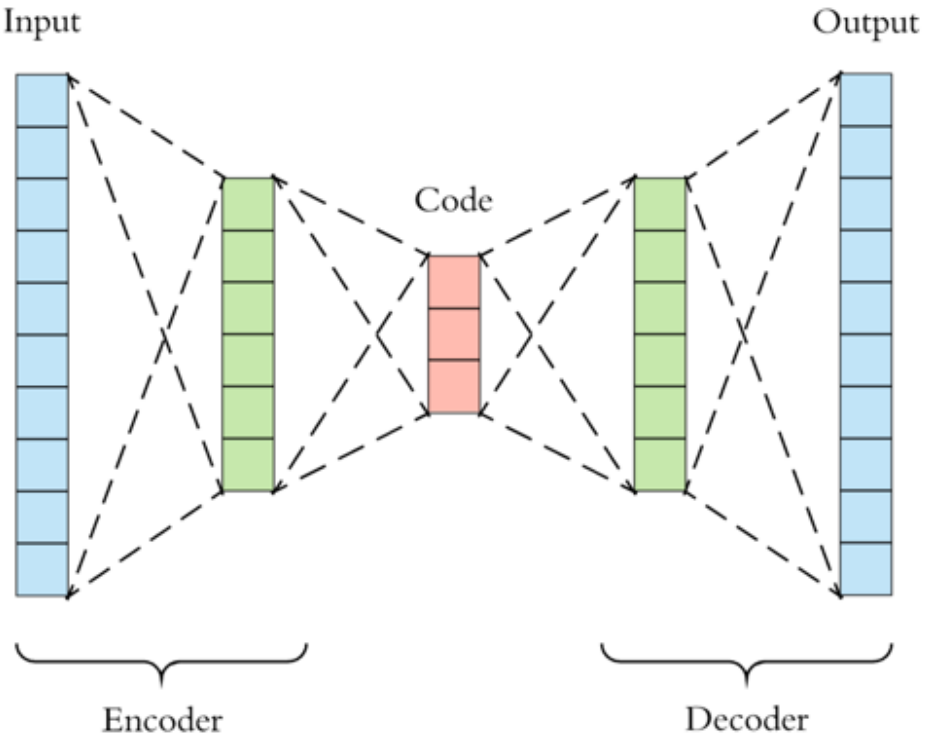
i = argmax p(z=i|x)



$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) = \sum_{\mathbf{z}} p(\mathbf{z}) p(\mathbf{x} \mid \mathbf{z}) = \sum_{k=1}^{K} p(\mathbf{z} = k) \underbrace{\mathcal{N}(\mathbf{x}; \mu_k, \Sigma_k)}_{\text{component}}$$

Autoencoder

- Autoencoder is originally proposed for dimensionality reduction and feature learning



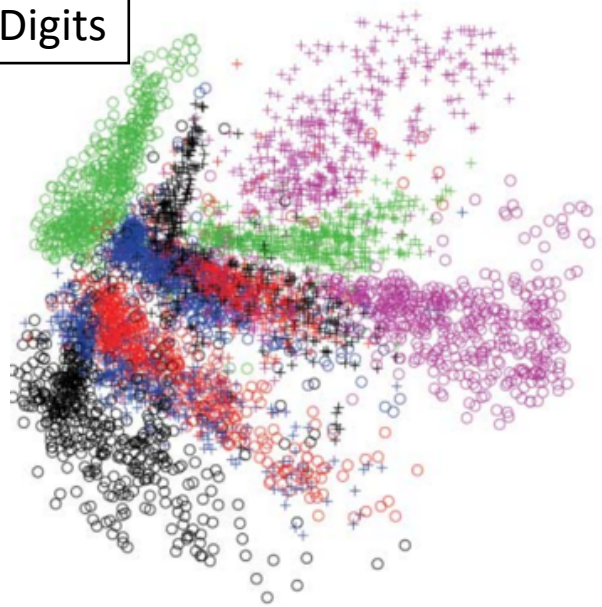
G.E. Hinton et al Science, 2016.

Autoencoder

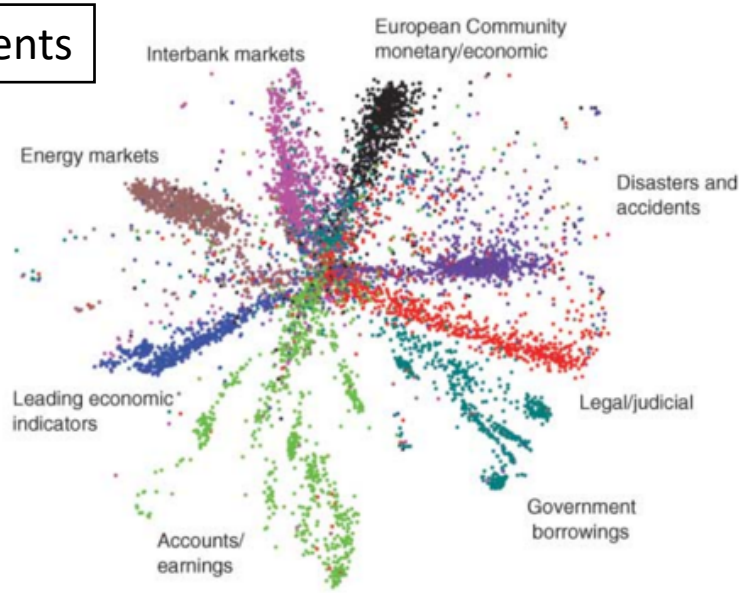
- Autoencoder is originally proposed for dimensionality reduction and feature learning

Clustering using the latent features

MNIST Digits



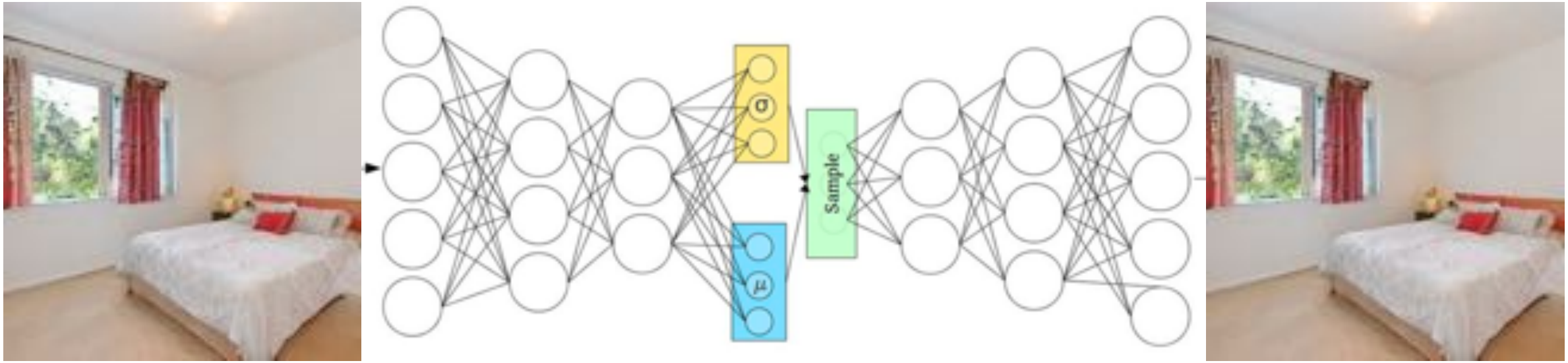
Documents



G.E. Hinton et al Science, 2016.

Variational Autoencoder

- Problem with the vanilla autoencoder is that we cannot sample the latent code to generate new image
- Thus variational autoencoder is introduced



https://github.com/pytorch/examples/blob/master/vae/main.py

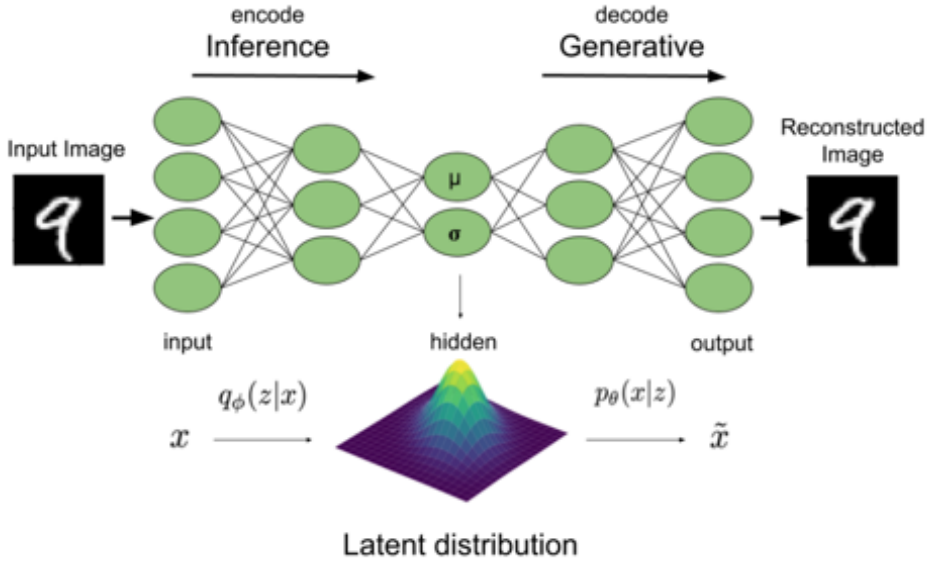
Variational Inference

Evidence Lower Bound (ELBO) holds for any q

$$
\begin{aligned}
\log p(\mathbf{x};\theta) &\geq \sum_{\mathbf{z}} q(\mathbf{z}) \log\left(\frac{p_\theta(\mathbf{x},\mathbf{z})}{q(\mathbf{z})}\right) \\
&= \sum_{\mathbf{z}} q(\mathbf{z}) \log p_\theta(\mathbf{x},\mathbf{z}) \underbrace{- \sum_{\mathbf{z}} q(\mathbf{z}) \log q(\mathbf{z})}_{\text{Entropy } H(q) \text{ of } q} \\
&= \sum q(\mathbf{z}) \log p_\theta(\mathbf{x},\mathbf{z}) + H(q) \\
&= \mathcal{L}(\mathbf{x};\theta,\phi) + D_{KL}(q(\mathbf{z};\phi) \| p(\mathbf{z}|\mathbf{x};\theta))
\end{aligned}
$$

The better $q(\mathbf{z};\phi)$ can approximate the posterior $p(\mathbf{z}|\mathbf{x};\theta)$, the smaller $D_{KL}(q(\mathbf{z};\phi)\|p(\mathbf{z}|\mathbf{x};\theta))$ we can achieve, the closer ELBO will be to $\log p(\mathbf{x};\theta)$. Next: jointly optimize over θ and ϕ to maximize the ELBO over a dataset

Variational Inference



$$\begin{aligned}\mathcal{L}(\mathbf{x};\theta,\phi) &= E_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{z},\mathbf{x};\theta) - \log q_\phi(\mathbf{z}|\mathbf{x}))] \\ &= E_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{z},\mathbf{x};\theta) - \log p(\mathbf{z}) + \log p(\mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x}))] \\ &= E_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z};\theta)] - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))\end{aligned}$$

1. Take a data point $\mathbf{x}^i$
2. Map it to $\hat{\mathbf{z}}$ by sampling from $q_\phi(\mathbf{z}|\mathbf{x}^i)$ (*encoder*)
3. Reconstruct $\hat{\mathbf{x}}$ by sampling from $p(\mathbf{x}|\hat{\mathbf{z}};\theta)$ (*decoder*)

What does the training objective $\mathcal{L}(\mathbf{x};\theta,\phi)$ do?

- First term encourages $\hat{\mathbf{x}} \approx \mathbf{x}^i$ ($\mathbf{x}^i$ likely under $p(\mathbf{x}|\hat{\mathbf{z}};\theta)$)
- Second term encourages $\hat{\mathbf{z}}$ to be likely under the prior $p(\mathbf{z})$

https://deepgenerativemodels.github.io/assets/slides/cs236_lecture5.pdf
https://deepgenerativemodels.github.io/assets/slides/cs236_lecture6.pdf

Variational Autoencoder

Synthesis quality looked good, but not good for recent two-year standard

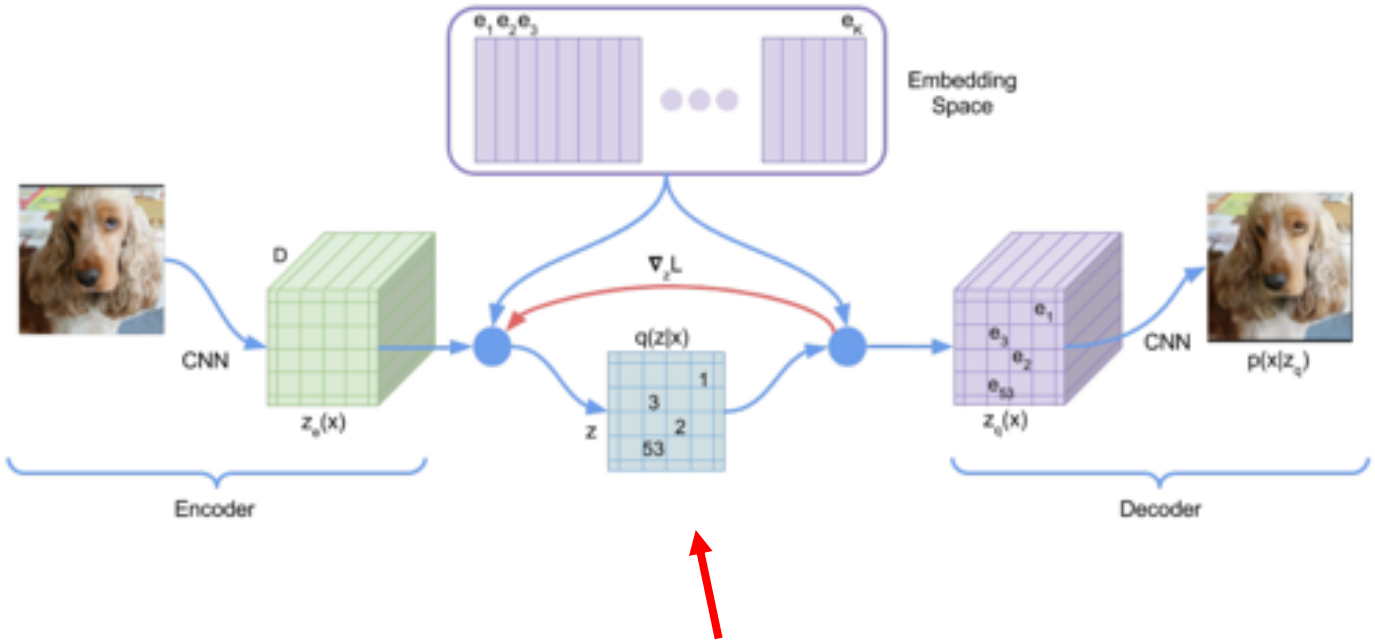
VQ-VAE and VQ-VAE-2

- Vector Quantized Variational AutoEncoder (VQ-VAE)
- Discrete latent representation + PixelCNN for handling posterior collapse

Figure 1: Class-conditional 256x256 image samples from a two-level model trained on ImageNet.

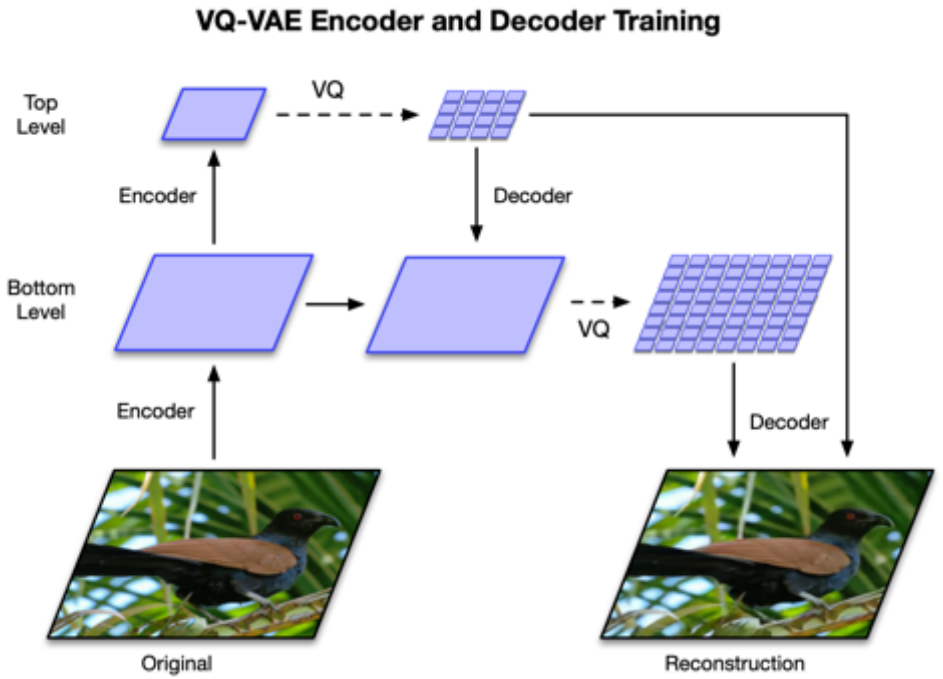
https://arxiv.org/pdf/1711.00937.pdf https://arxiv.org/pdf/1906.00446.pdf

VQ-VAE-1



PixeCNN is learned on the discrete latent space

VQ-VAE-2



https://arxiv.org/pdf/1711.00937.pdf https://arxiv.org/pdf/1906.00446.pdf

VQ-VAE and VQ-VAE-2

Very well handle mode-collapse issue in generative models

VQ-VAE on ImageNet's Tinca class

BigGAN

https://arxiv.org/pdf/1711.00937.pdf https://arxiv.org/pdf/1906.00446.pdf

Summary of Latent Variable Models

Pros:

- Easy to build flexible models
- Suitable for unsupervised learning

Cons:

- Hard to evaluate likelihood
- Hard to train via maximum-likelihood, as posterior inference $p(z|x)$ is hard which requires variational approximations

Autoregressive models and Variational Autoencoders

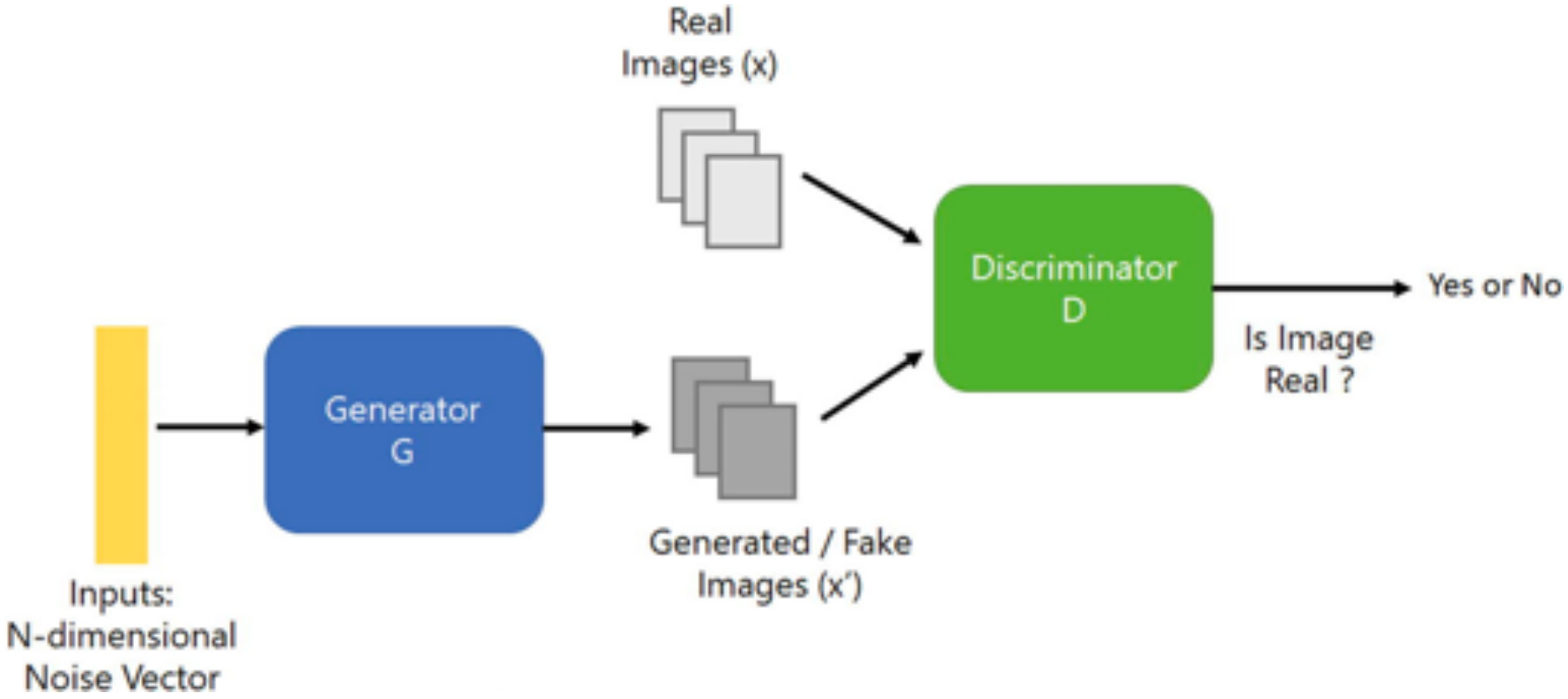
- Both model families are based on maximizing likelihoods (or approximation)

$$\text{Autoregressive Models: } p_\theta(\mathbf{x}) = \prod_{i=1}^{n} p_\theta(x_i|\mathbf{x}_{<i})$$
$$\text{Variational Autoencoders: } p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z}$$

- Is that likelihood a good indicator? Any other likelihood-free generative modeling?
- Can we throw away KL-divergence and likelihood???

Generative Adversarial Networks

A two-player minimax game between a generator and a discriminator



Goodfellow et al. NIPS'14

Training objective for discriminator:

$$\max_D V(G, D) = E_{\mathbf{x} \sim p_{\text{data}}}[\log D(\mathbf{x})] + E_{\mathbf{x} \sim p_G}[\log(1 - D(\mathbf{x}))]$$

For a fixed generator G, the discriminator is performing binary classification with the cross entropy objective

- Assign probability 1 to true data points $\mathbf{x} \sim p_{\text{data}}$
- Assing probability 0 to fake samples $\mathbf{x} \sim p_G$

Optimal discriminator

$$D_G^*(\mathbf{x}) = \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})}$$

Training objective for generator:

$$\min_G V(G, D) = E_{\mathbf{x}\sim p_{\text{data}}}[\log D(\mathbf{x})] + E_{\mathbf{x}\sim p_G}[\log(1 - D(\mathbf{x}))]$$

For the optimal discriminator $D_G^*(\cdot)$, we have

$$\begin{aligned}
& V(G, D_G^*(\mathbf{x})) \\
&= E_{\mathbf{x}\sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x})+p_G(\mathbf{x})}\right] + E_{\mathbf{x}\sim p_G}\left[\log \frac{p_G(\mathbf{x})}{p_{\text{data}}(\mathbf{x})+p_G(\mathbf{x})}\right] \\
&= E_{\mathbf{x}\sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\mathbf{x})}{\frac{p_{\text{data}}(\mathbf{x})+p_G(\mathbf{x})}{2}}\right] + E_{\mathbf{x}\sim p_G}\left[\log \frac{p_G(\mathbf{x})}{\frac{p_{\text{data}}(\mathbf{x})+p_G(\mathbf{x})}{2}}\right] - \log 4 \\
&= \underbrace{D_{KL}\left[p_{\text{data}}, \frac{p_{\text{data}} + p_G}{2}\right] + D_{KL}\left[p_G, \frac{p_{\text{data}} + p_G}{2}\right]}_{2\times\text{Jenson-Shannon Divergence (JSD)}} - \log 4 \\
&= 2D_{JSD}[p_{\text{data}}, p_G] - \log 4
\end{aligned}$$

Symmetric KL divergence

Optimal generator

$$p_G = p_{\text{data}}$$

Joint loss:

$$\min_{\theta} \max_{\phi} V(G_\theta, D_\phi) = E_{\mathbf{x}\sim p_{\text{data}}}[\log D_\phi(\mathbf{x})] + E_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D_\phi(G_\theta(\mathbf{z})))]$$

GAN training algorithm

Repeat:

- Sample minibatch of m training points $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)}$ from $\mathcal{D}$
- Sample minibatch of m noise vectors $\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \ldots, \mathbf{z}^{(m)}$ from p_z
- Update the generator parameters θ by stochastic gradient **descent**

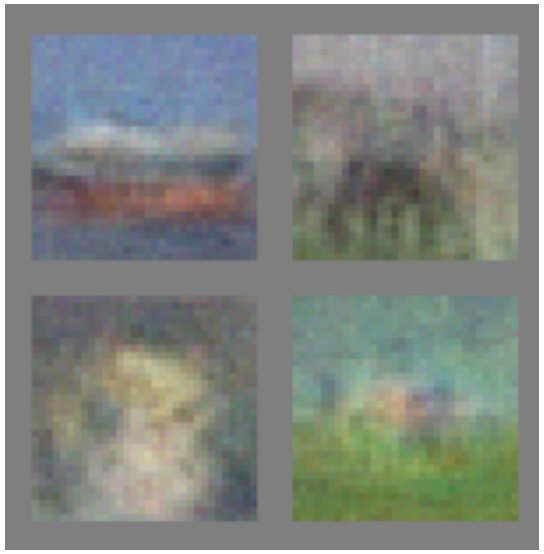
$$\nabla_\theta V(G_\theta, D_\phi) = \frac{1}{m} \nabla_\theta \sum_{i=1}^{m} \log(1 - D_\phi(G_\theta(\mathbf{z}^{(i)})))$$

- Update the discriminator parameters ϕ by stochastic gradient **ascent**

$$\nabla_\phi V(G_\theta, D_\phi) = \frac{1}{m} \nabla_\phi \sum_{i=1}^{m} [\log D_\phi(\mathbf{x}^{(i)}) + \log(1 - D_\phi(G_\theta(\mathbf{z}^{(i)})))]$$

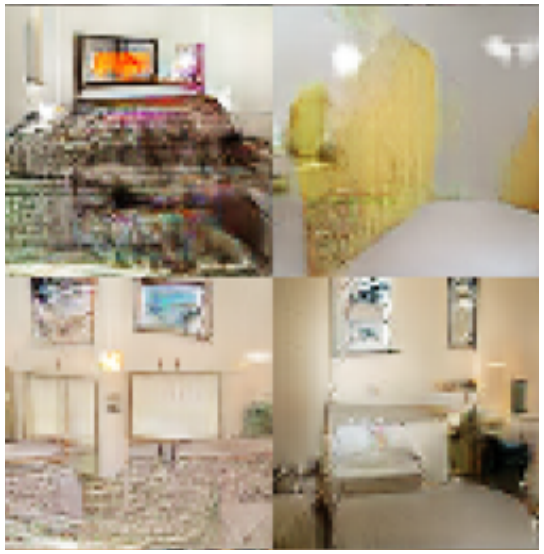
GAN Progress for Image Synthesis

2014



GANs [Goodfellow et al.]

2015



DCGAN [Radford et al.]

2017

PG-GAN [Karras et al.]

2018

BigGAN [Brock et al.]

• • •

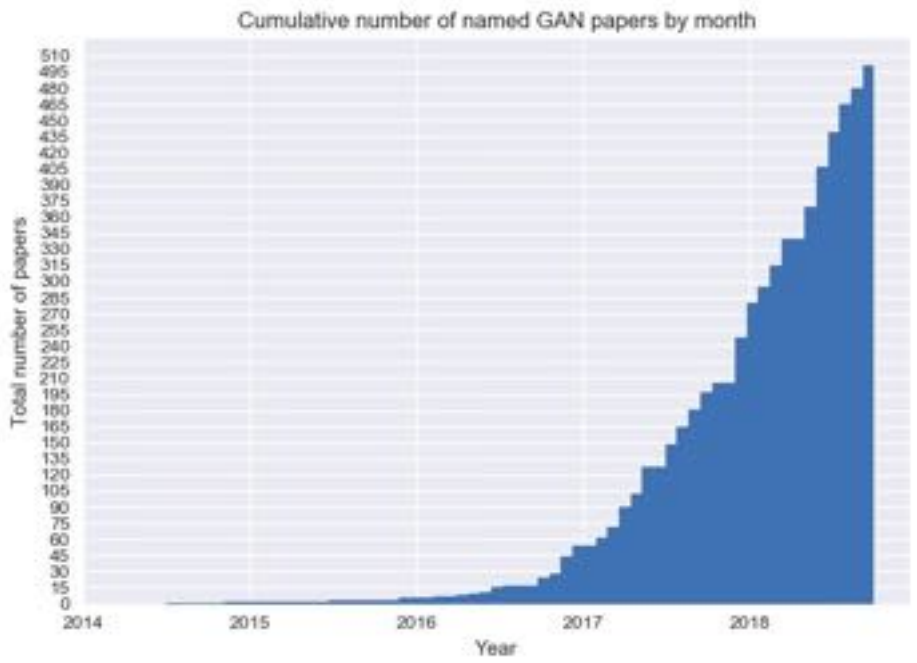
GAN Progress for Image Synthesis

PG-GAN

StyleGAN

BigGAN

GANs explode!



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

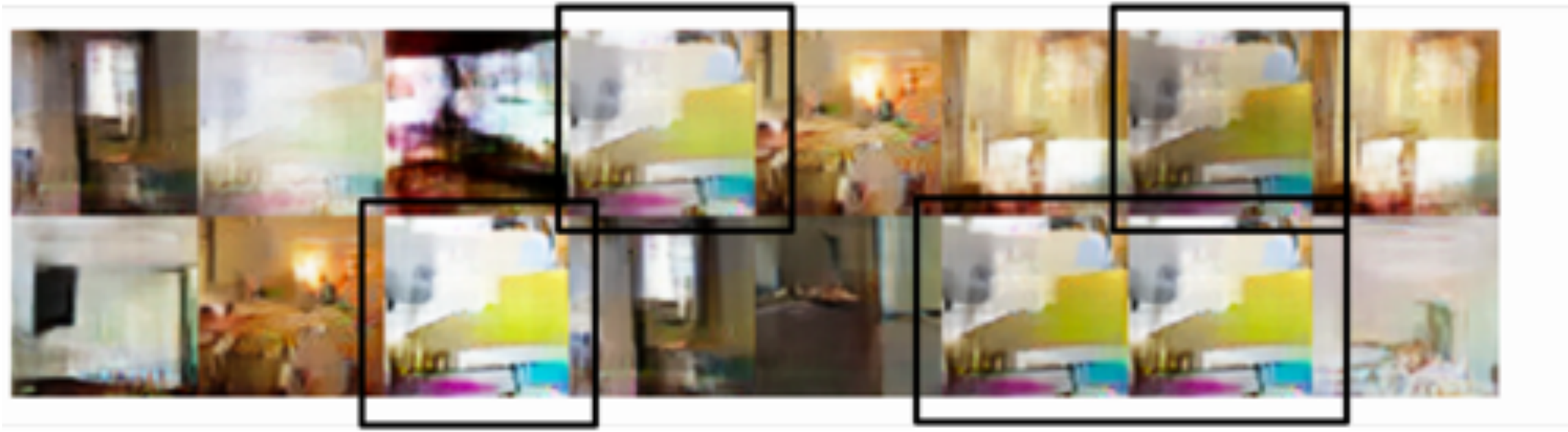
- 3D-ED-GAN
- 3D-GAN - L
- 3D-IWGAN
- 3D-PhysNet
- 3D-RecGAN
- ABC-GAN -
- ABC-GAN -
- AC-GAN - C
- acGAN - Fac
- ACGAN - Co
- acGAN - On
- ACtuAL - AC
- AdaGAN - A
- Adaptive GA
- AdvEntuRe -
- AdvGAN - G
- AE-GAN - A
- AE-OT - Lat
- AEGAN - Le
- AF-DCGAN System
- AffGAN - Ar
- AIM - Gener
- AL-CGAN -
- ALI - Advers
- AlignGAN -
- AlphaGAN -
- AM-GAN - A
- AmbientGAN
- AMC-GAN -
- AnoGAN - L
- APD - Adver
- APE-GAN -
- ARIGAN - ARIG
- ArtGAN - ArtGA
- ASDL-GAN - Au
- ATA-GAN - Atte
- Attention-GAN -
- AttGAN - Arbitra
- AttnGAN - Attn
- AVID - AVID: Ad
- B-DCGAN - B-D
- b-GAN - Genera
- BAGAN - BAGA
- Bayesian GAN -
- Bayesian GAN -
- BCGAN - Bayes
- BCGAN - Bidire
- BEAM - Boltzma
- BEGAN - BEGA
- BEGAN-CS - Es
- Bellman GAN -
- BGAN - Binary
- Bi-GAN - Auton Aided Genetic A
- BicycleGAN - To
- BiGAN - Advers
- BinGAN - BinGA
- BourGAN - Bou
- BranchGAN - Br
- BRE - Improving
- BridgeGAN - Ge
- BS-GAN - Boun
- BubGAN - BubG
- BWGAN - Bana
- C-GAN - Face A
- CatGAN - CatG
- CausalGAN - C
- CC-GAN - Sem
- cd-GAN - Con
- CDcGAN - Sim
- CE-GAN - Dee
- CFG-GAN - Co
- CGAN - Condi
- CGAN - Contro
- Chekhov GAN
- ciGAN - Condi
- CinCGAN - Un
- CipherGAN - U
- ClusterGAN - C
- CM-GAN - CM
- CoAtt-GAN - A
- CoGAN - Coup
- ComboGAN - C
- ConceptGAN -
- Conditional cyc
- constrast-GAN
- Context-RNN-
- CorrGAN - Cor
- Coulomb GAN
- Cover-GAN - C
- cowboy - Defe
- CR-GAN - CR-
- Cramèr GAN -
- Cross-GAN - C
- crVAE-GAN - C
- CS-GAN - Imp
- CSG - Speech
- CT-GAN - CT-
- EBGAN - En
- ecGAN - eC
- ED//GAN - S
- Editable GAN
- EGAN - Enha
- EL-GAN - EL
- ELEGANT -
- EnergyWGA
- ESRGAN - E
- ExGAN - Eye
- ExposureGA
- ExprGAN - E
- f-CLSWGAN
- f-GAN - f-GA
- FairGAN - Fa
- Fairness GAN
- FakeGAN - D
- FBGAN - Fe
- FBGAN - Fea
- FC-GAN - Fa
- FF-GAN - To
- FGGAN - Ad
- Fictitious GA
- FIGAN - Fra
- Fila-GAN - S
- First Order G
- Fisher GAN -
- Flow-GAN -
- FrankenGAN
- FSEGAN - E
- FTGAN - Hie
- FusedGAN -
- FusionGAN -
- GANosaic - (
- GANVO - GA Networks
- GAP - Conte
- GAP - Gener
- GATS - Samp
- GAWWN - L
- GC-GAN - G
- GcGAN - Ge
- GeneGAN - (
- GeoGAN - G
- Geometric G
- GIN - Genera
- GLCA-GAN -
- GM-GAN - G
- GMAN - Ger
- GMM-GAN -
- GoGAN - Ga
- GONet - GO
- GP-GAN - G
- GP-GAN - G
- GPU - A gen
- GRAN - Gen
- Graphical-GA
- GraphSGAN
- GraspGAN -
- GT-GAN - D
- HAN - Chine
- HAN - Bidire
- HiGAN - Exp
- HP-GAN - H
- HR-DCGAN
- hredGAN - M

Challenges for GANs

- Many tricks have been applied to train GANs
- There are still many challenges:
  - Unstable optimization
  - Mode collapse
  - Evaluation: how to evaluate the synthesis result

Mode Collapse

GAN often collapses to one or few samples (modes)



Arjovsky et al., 2017

Mode Collapse

Target

- True distribution is a mixture of Gaussians

Step 0 Step 5k Step 10k Step 15k Step 20k

Source: Metz et al., 2017

Comparison of VQ-VAE and BigGAN

Likelihood-based

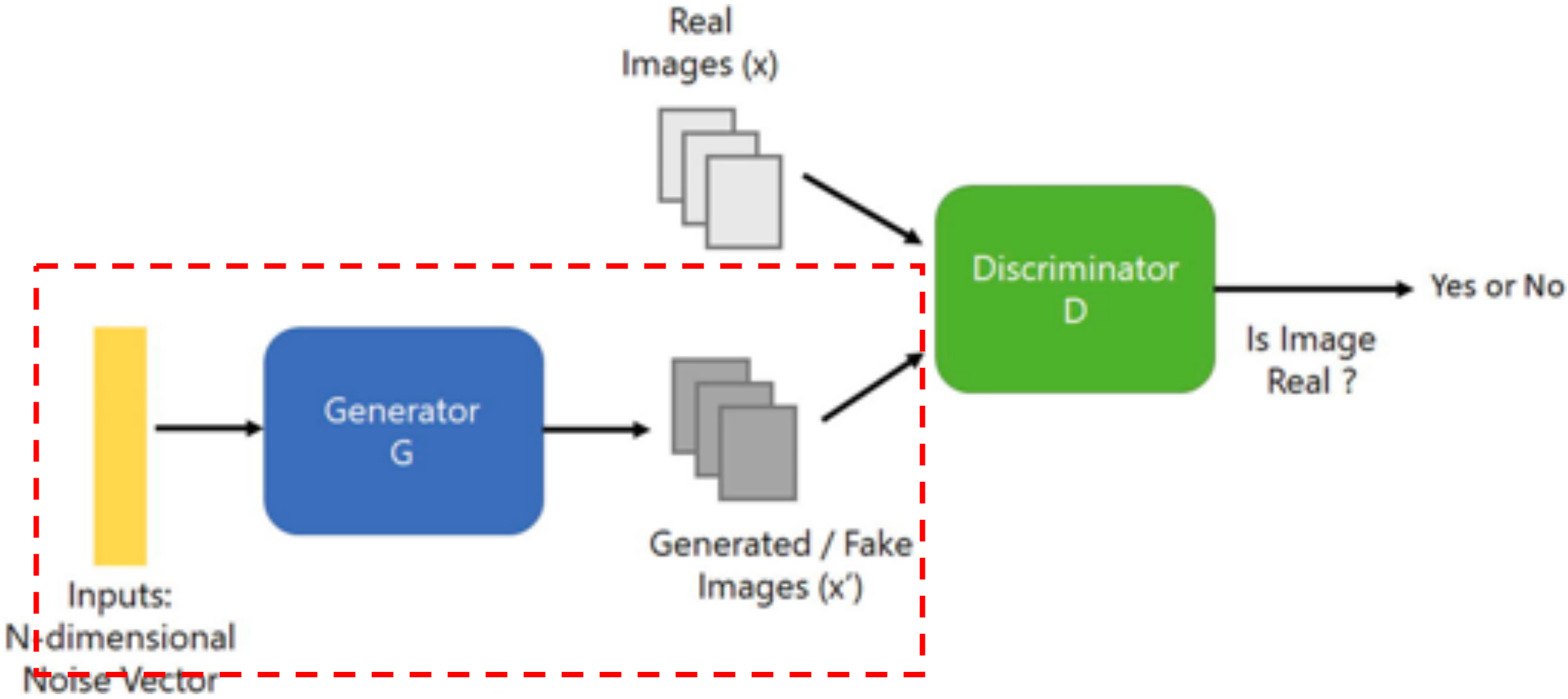
Adversarial learning-based

Comparison of VQ-VAE and BigGAN

Likelihood-based

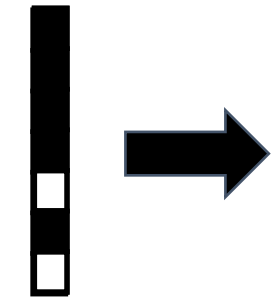
Adversarial learning-based

What have been learned inside the GANs?



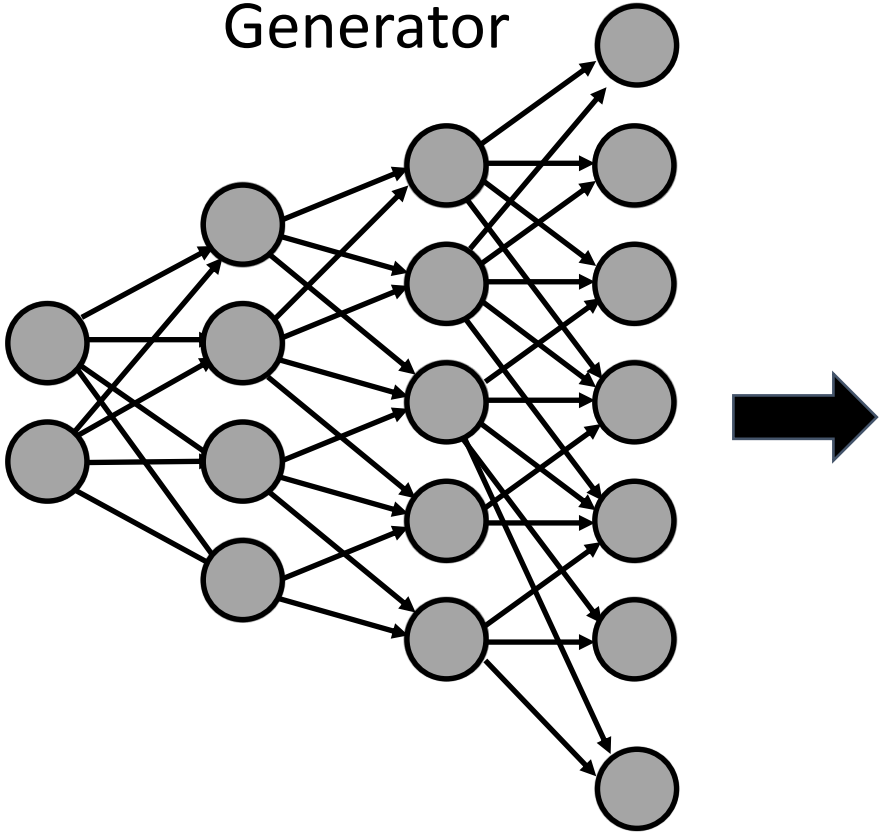
Deep Generative Representation

Random vector



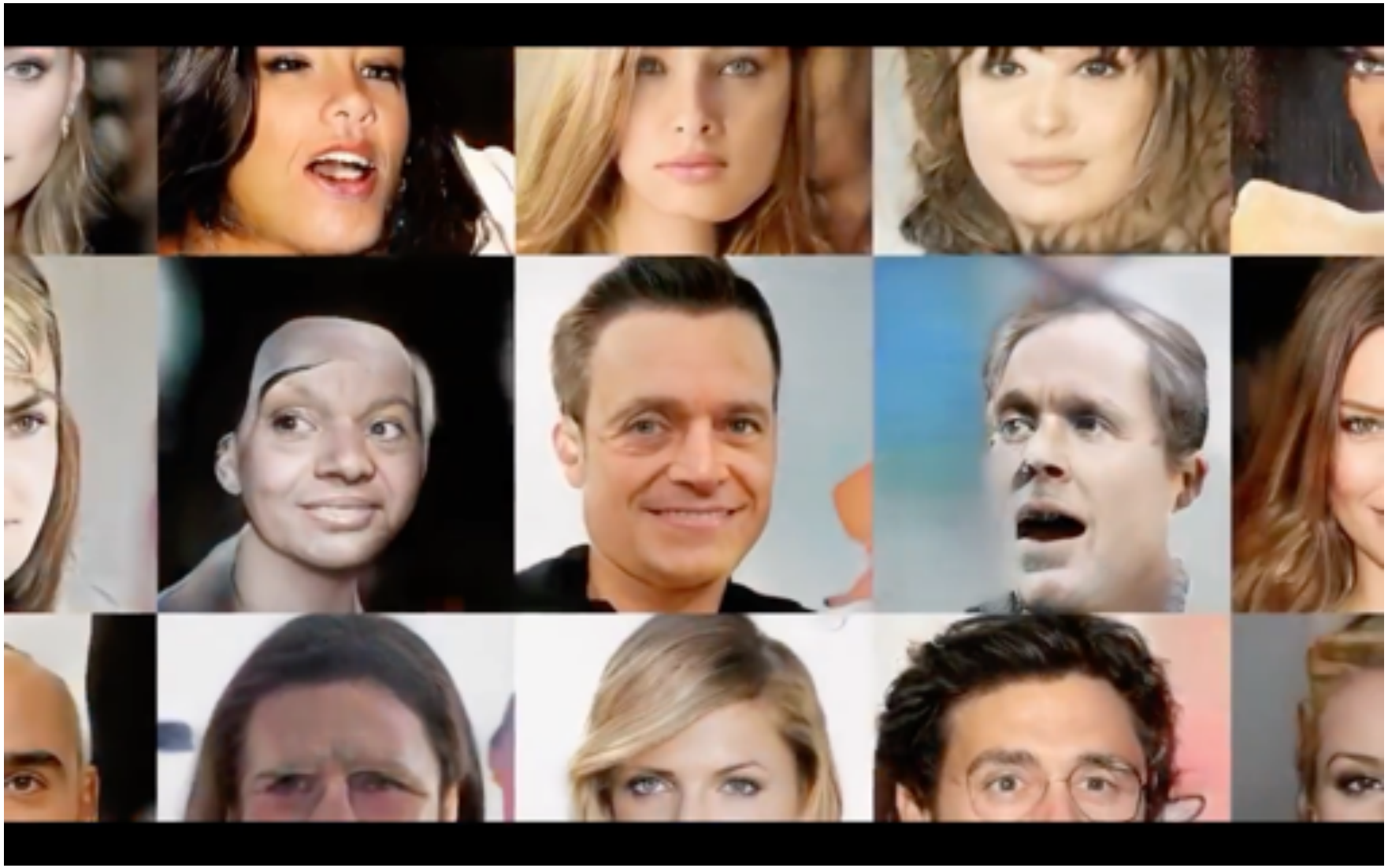
512 dimensions

Generator



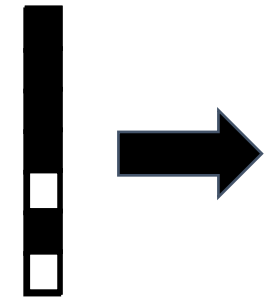
Randomly generated image

Random Walk in the Latent Space



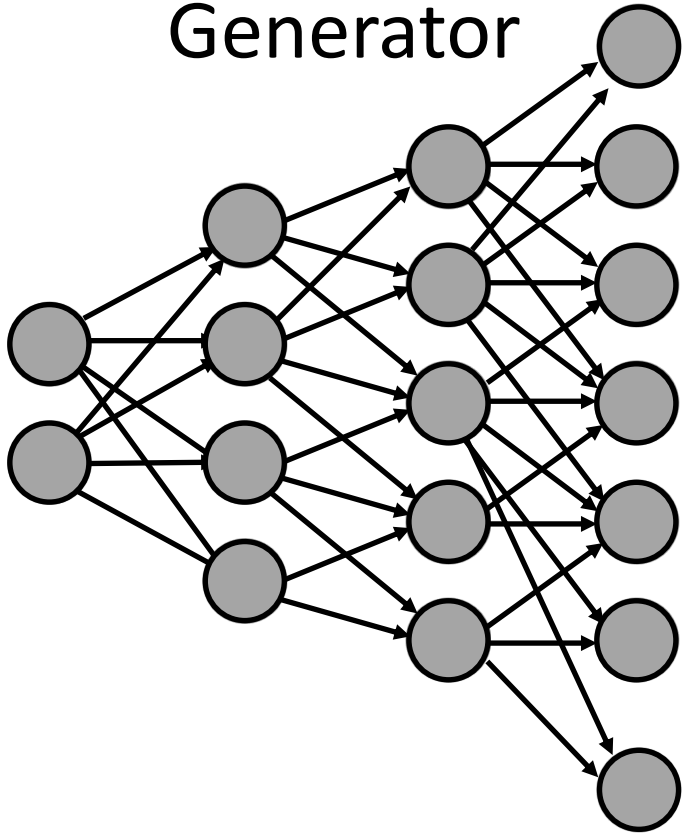
How to Customize the Synthesis Output?

Latent space



512 dimensions

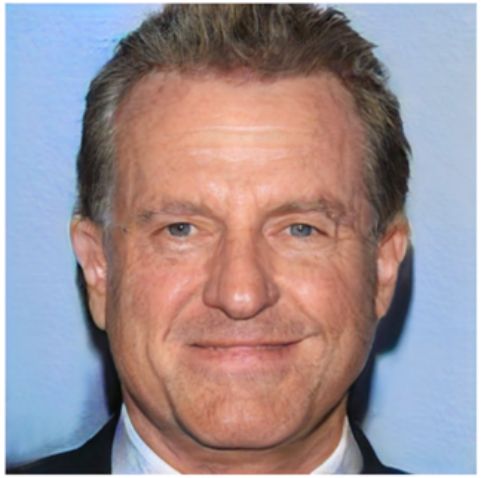
Generator



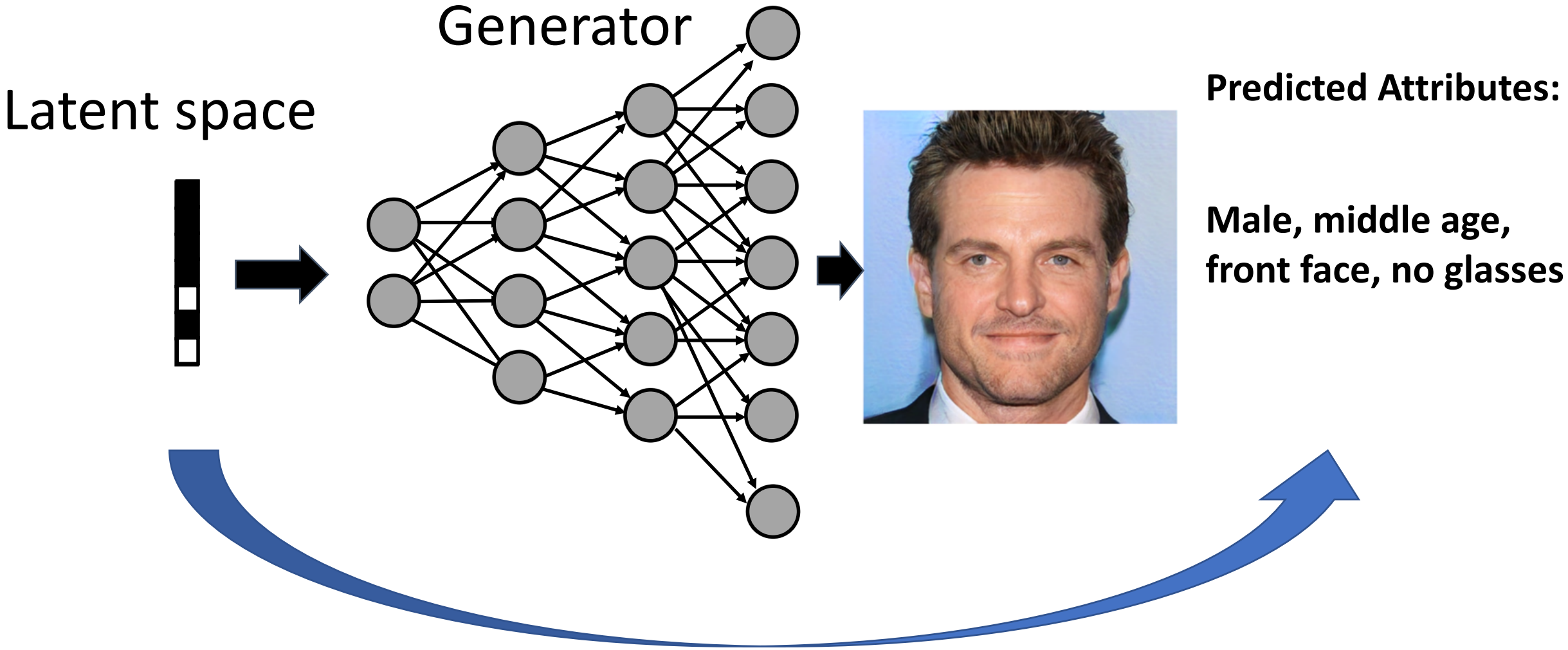
Output

Different angle

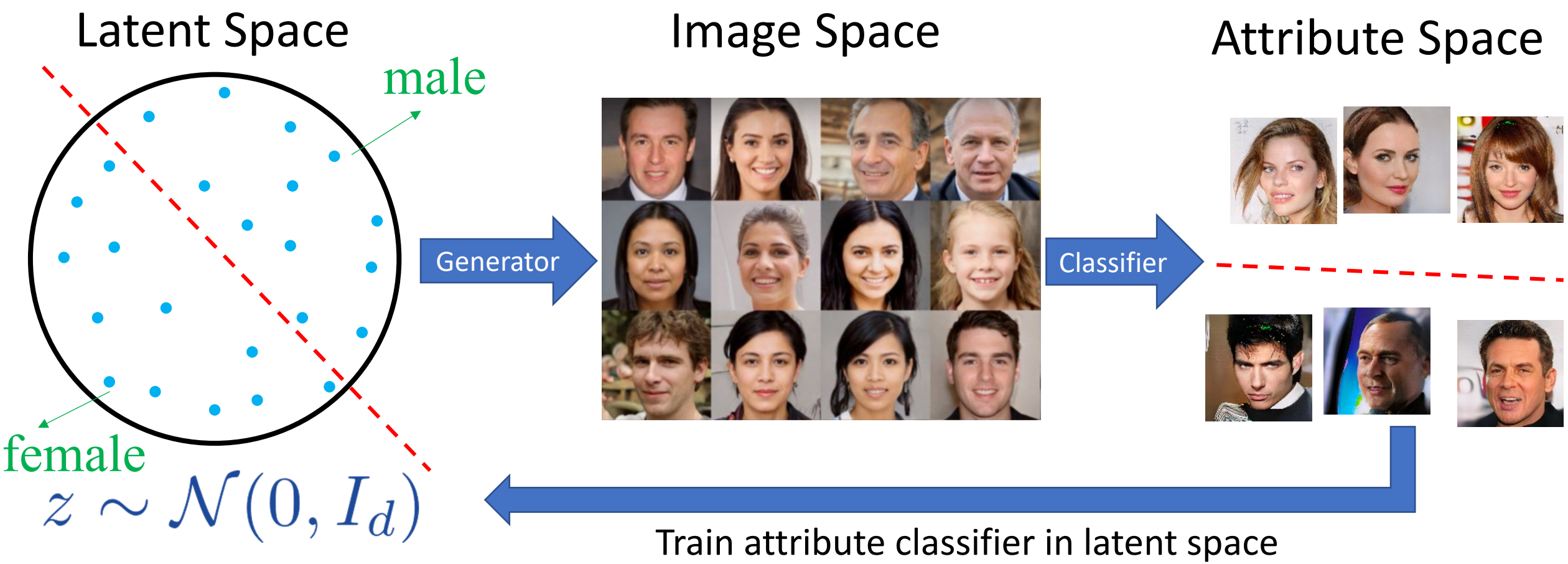
Different age



Causal Relations in Latent Space

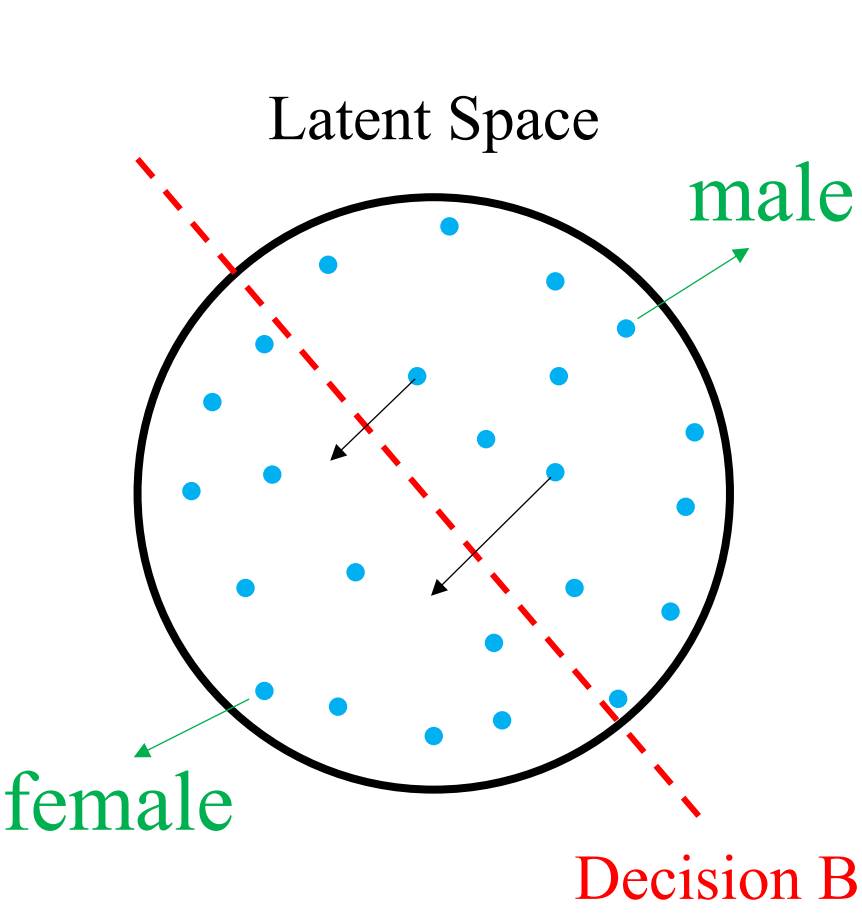


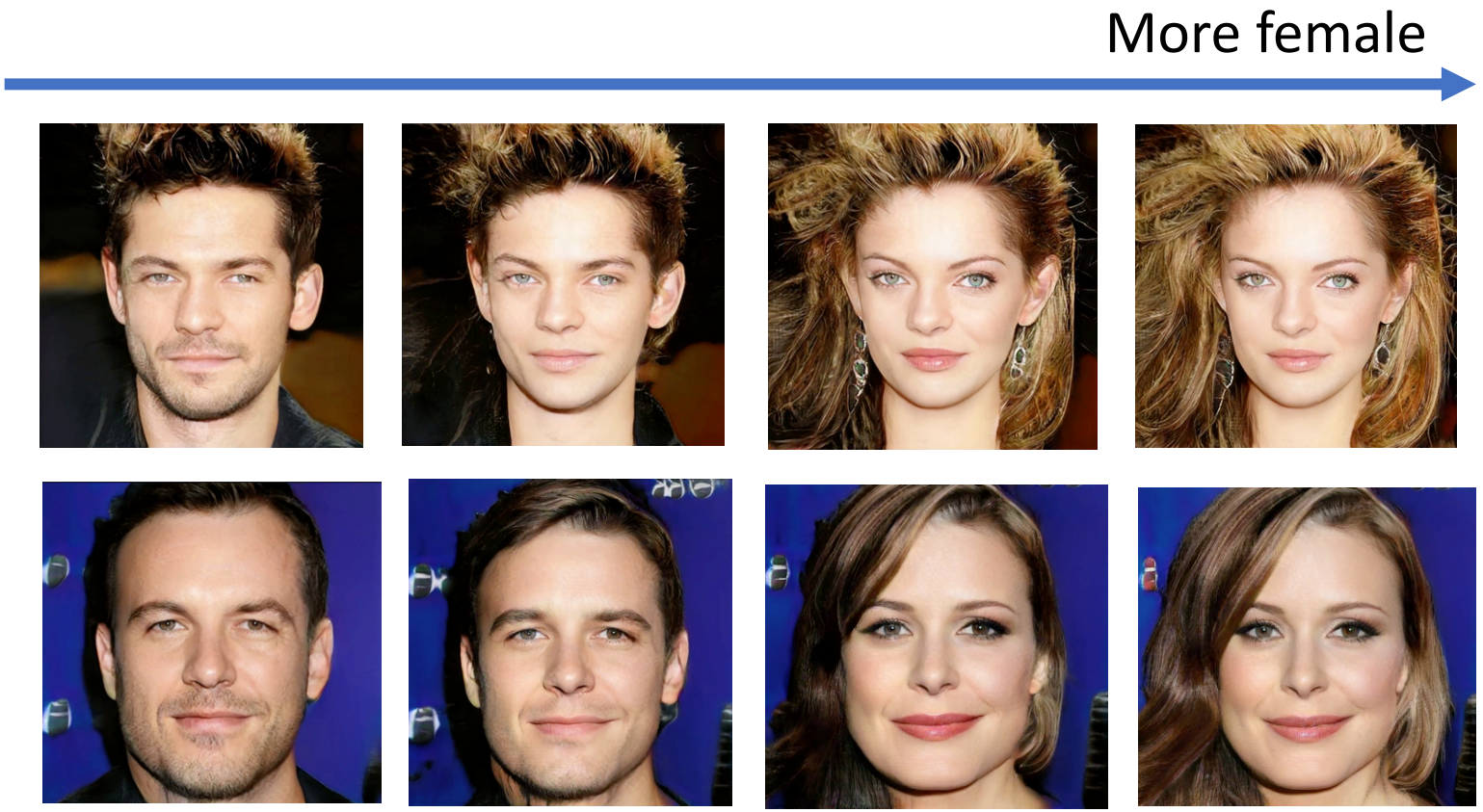
InterFaceGAN: Interpreting Semantics in Face GANs



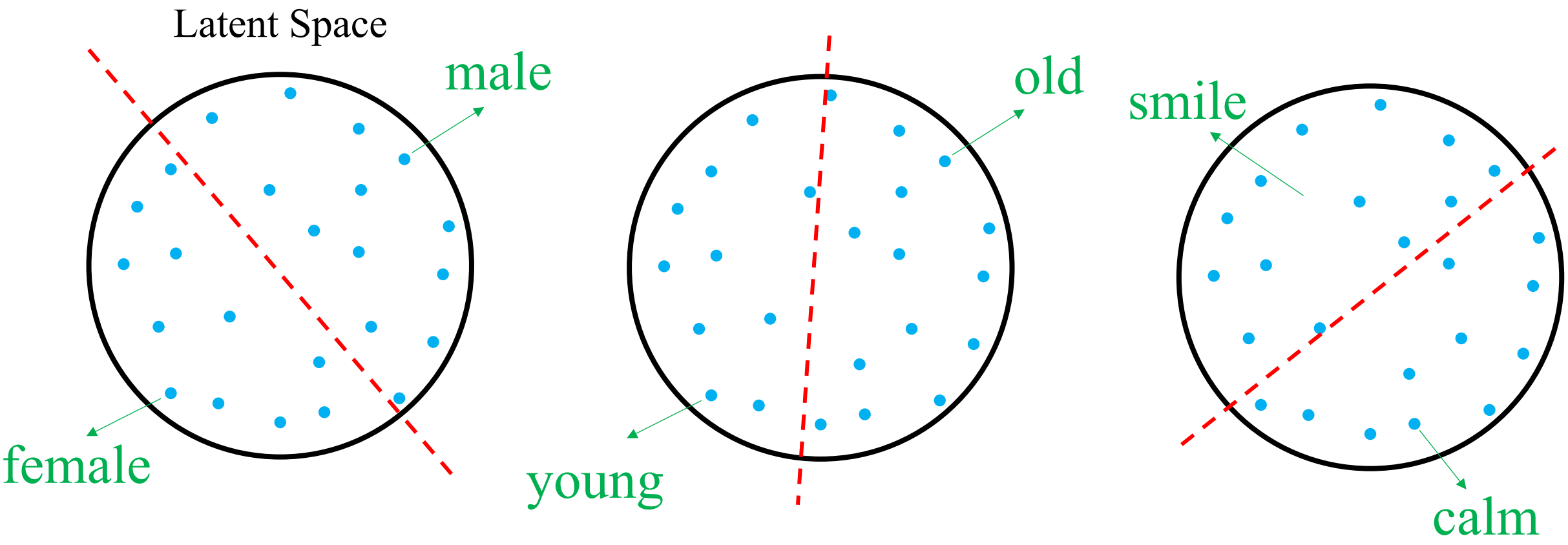
Yujun Shen, Jinjin Gu, Xiaoou Tang, Bolei Zhou. CVPR2020

Varying the Latent Code through Boundary





Various Attribute Boundaries to Divide the Latent Space



Make me cooler

Make me younger

Make me front faced

Make me more man

Correcting the Mistakes Made by GAN

Demo Video

Latent Semantics in GANs for Scene Synthesis

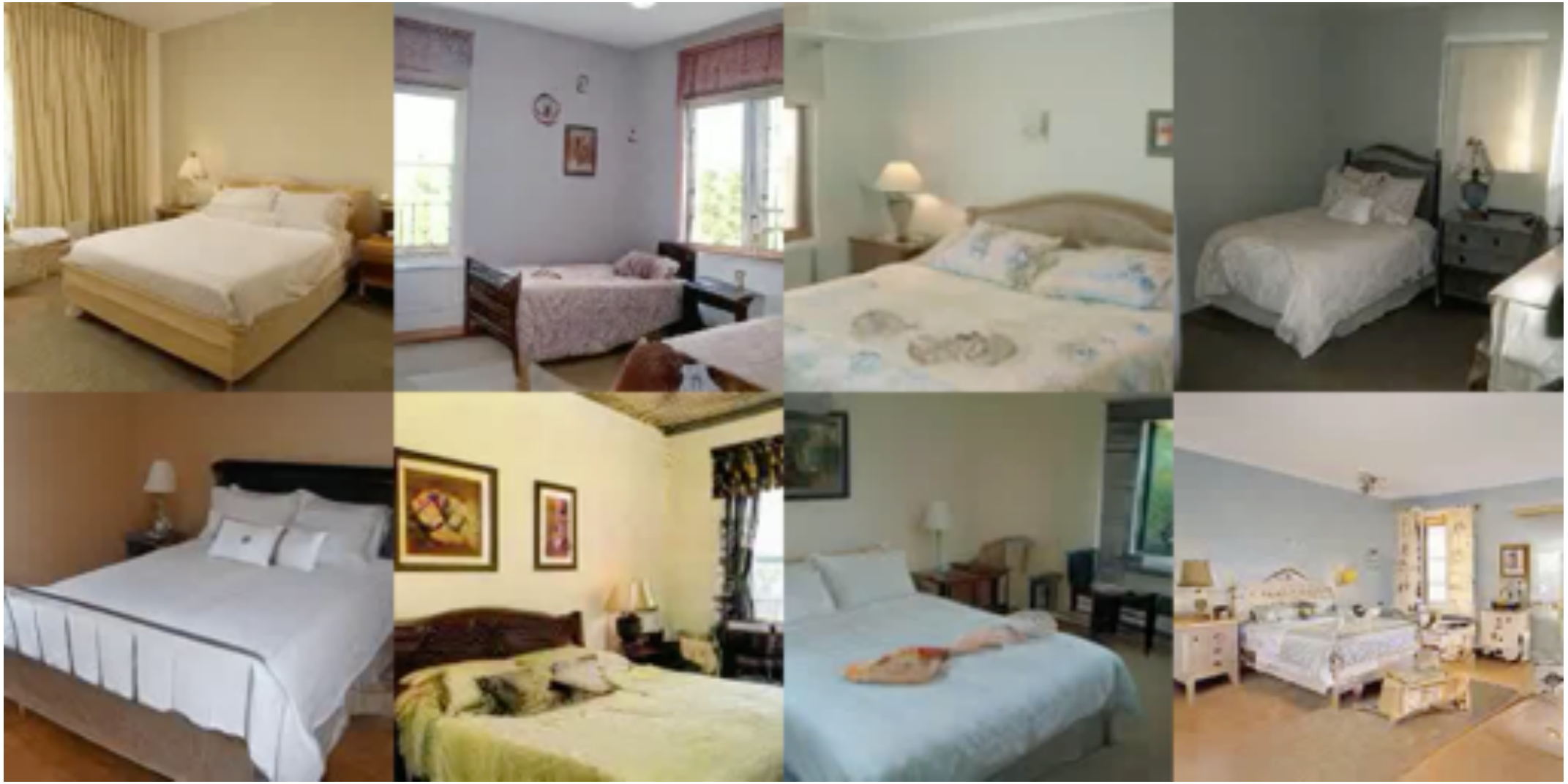
Random vector

512 dimensions

Generator

Randomly generated image

Random Walk in Latent Space of Bedroom



Multiple Levels of Abstractions for Scenes

Scene category:

bedroom

Scene attributes:

nature lighting

wood

foliage

…

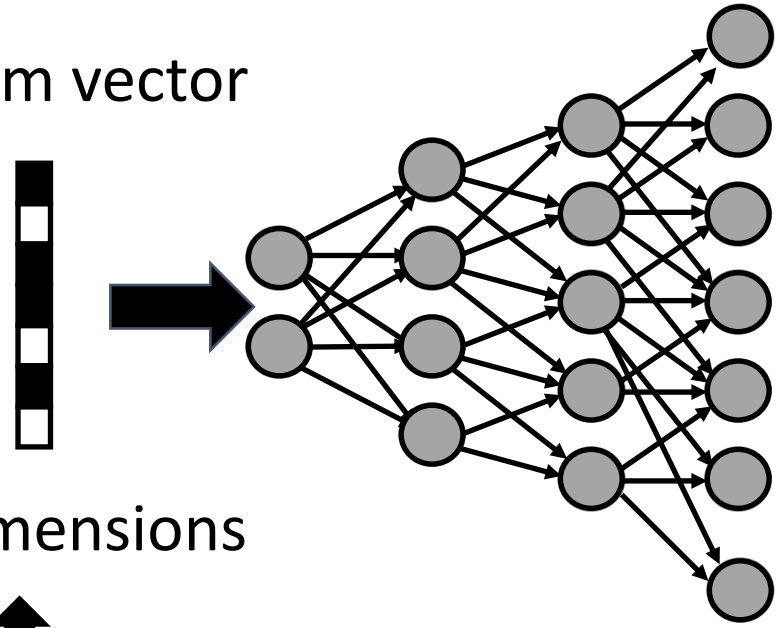
Layout

Segmentation

Identifying the Causality in Latent Space

Predicted semantics

Random vector



512 dimensions

Scene category: bedroom

Scene attributes: nature lighting, wood, foliage

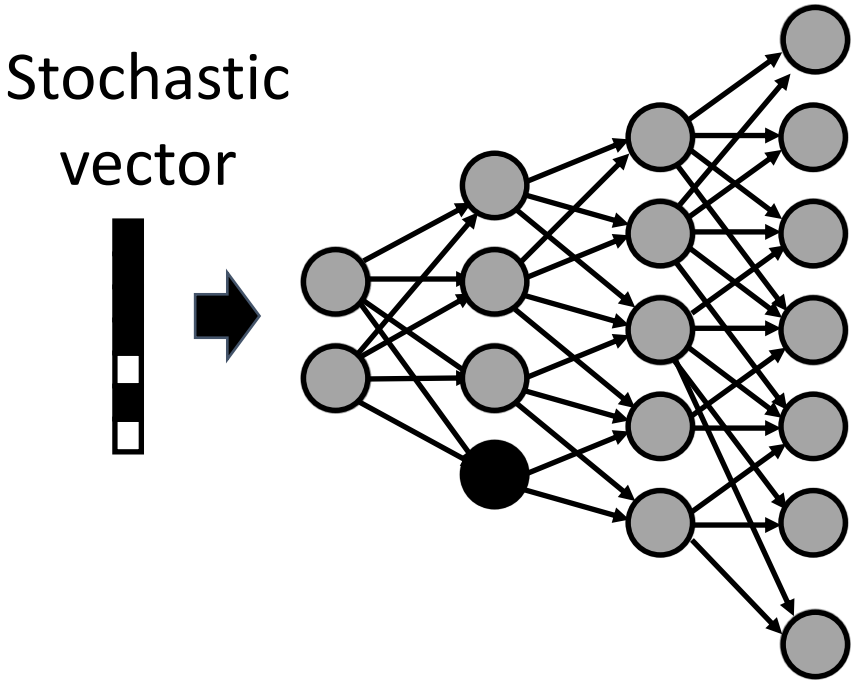
To identify the cause-effect relations

Ceyuan Yang*, Yujun Shen*, Bolei Zhou. Semantic Hierarchy Emerges in Deep Generative Representations for Scene Synthesis.
https://arxiv.org/pdf/1911.09267.pdf

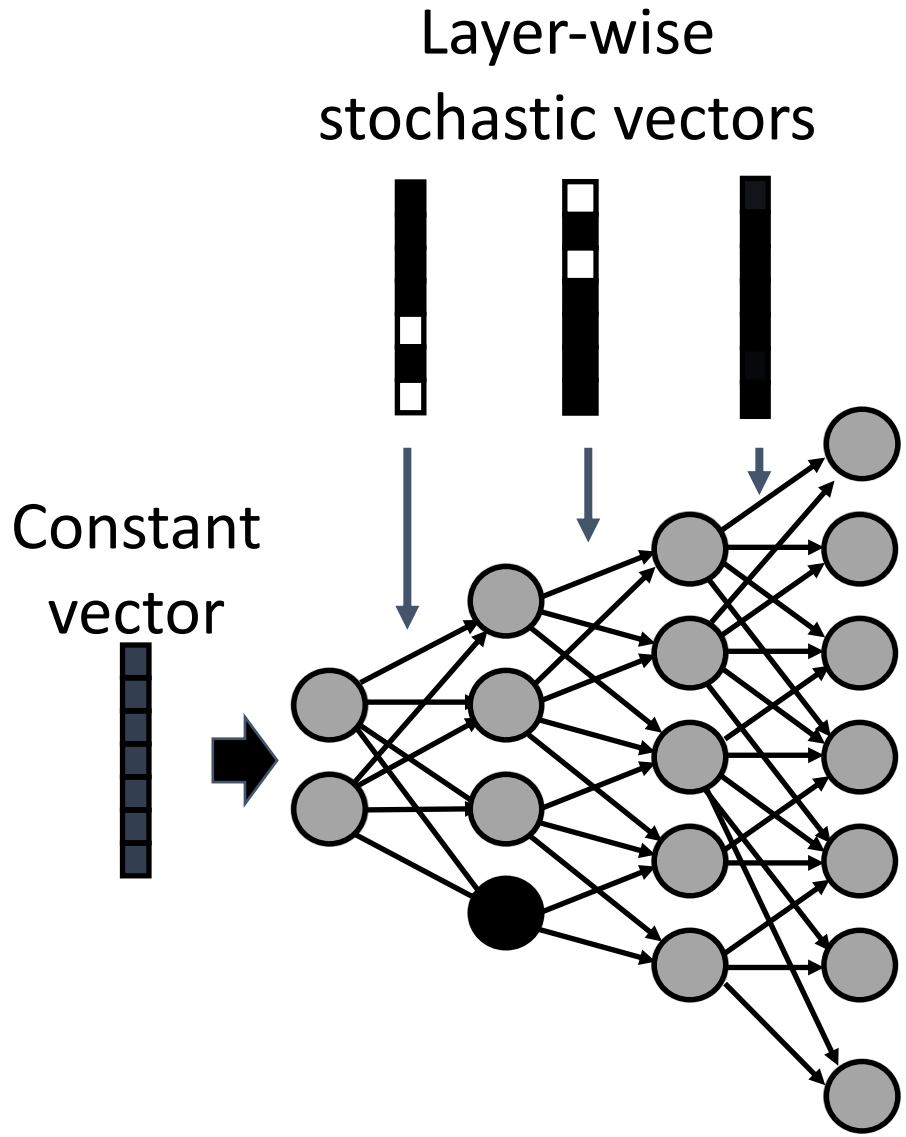
Result on turning up the lights

Result on ageing the scenes

Layer-wise Stochasity

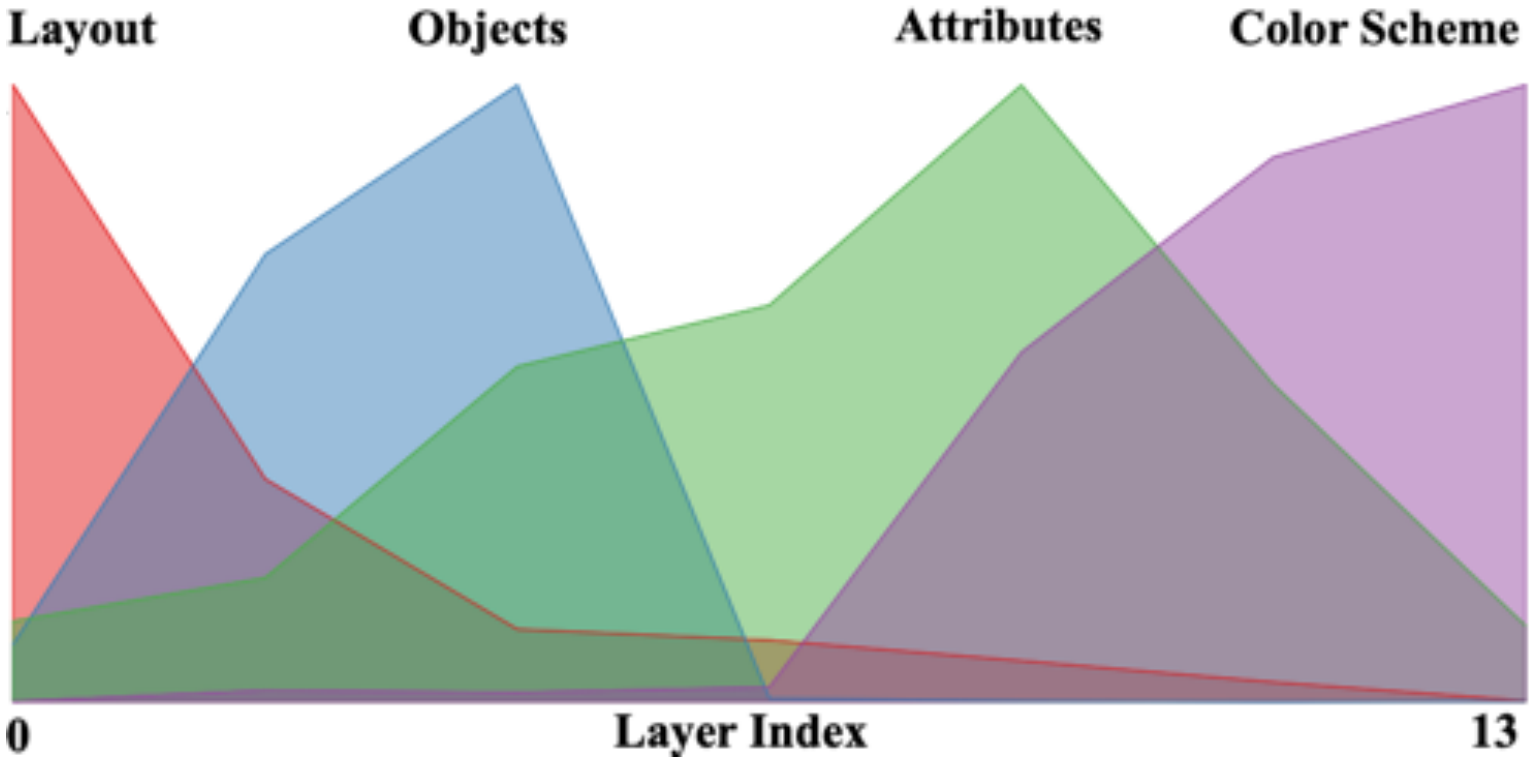


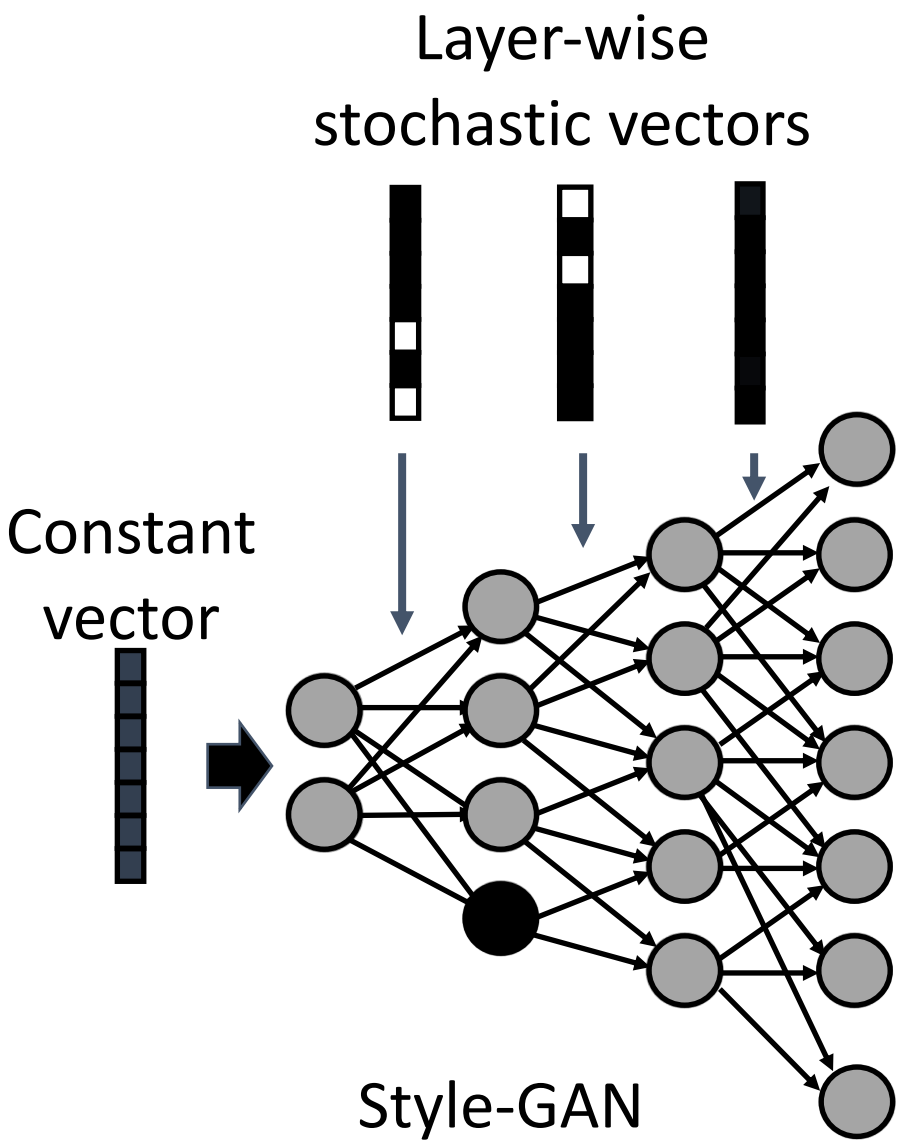
DC-GAN, PG-GAN



Style-GAN [Karras et al]

Semantic hierarchy emerges across the layers of generator





Ceyuan Yang, Yujun Shen, Bolei Zhou. Semantic Hierarchy Emerges in Deep Generative Representations for Scene Synthesis.
https://arxiv.org/pdf/1911.09267.pdf

Varying viewpoints

Bedroom to Dining Room

GenForce

Research Initiative on Generative Modeling

Team

Principal Investigator

Bolei Zhou

Team Members

Yujun Shen

Ceyuan Yang

Jinjin Gu

Jiapeng Zhu

Projects

In-Domain GAN Inversion for Real Image Editing
Jiapeng Zhu*, Yujun Shen*, Deli Zhao, Bolei Zhou
arXiv.2004.00049 preprint
[Paper] [Project Page] [Code] [Demo Video]

Semantic Hierarchy Emerges in Deep Generative Representations for Scene Synthesis
Ceyuan Yang*, Yujun Shen*, Bolei Zhou
arXiv.1911.09267 preprint
[Paper] [Project Page] [Code] [Demo Video]

Image Processing Using Multi-Code GAN Prior
Jinjin Gu, Yujun Shen, Bolei Zhou
Computer Vision and Pattern Recognition (CVPR), 2020
[Paper] [Project Page] [Code]

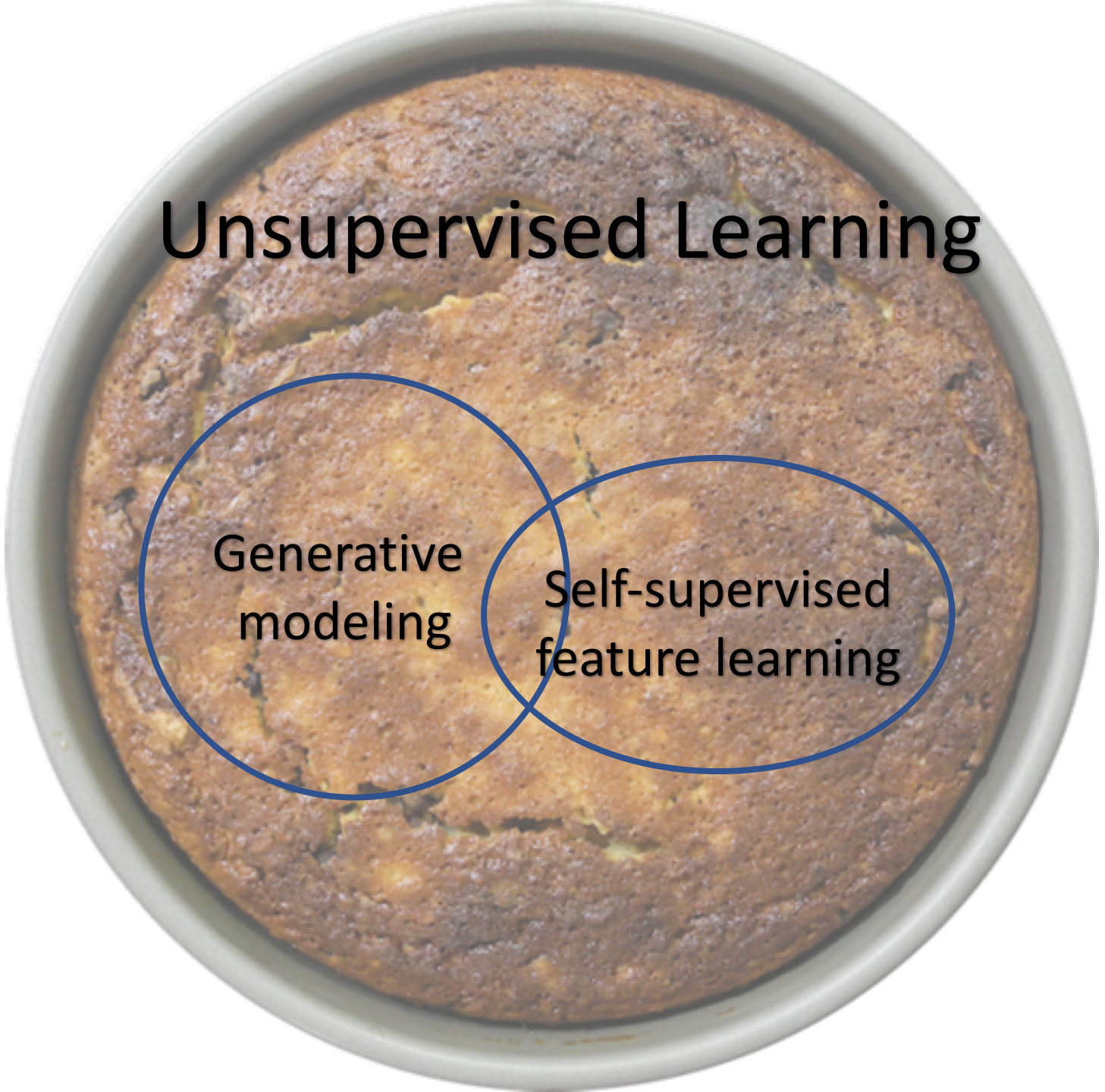
Interpreting the Latent Space of GANs for Semantic Face Editing
Yujun Shen, Jinjin Gu, Xiaoou Tang, Bolei Zhou
Computer Vision and Pattern Recognition (CVPR), 2020
[Paper] [Project Page] [Code] [Demo Video]

https://genforce.github.io/

Conclusion

- Likehood-based models: Autoregressive models
- Latent variable models: Variational Autoencoder
- Implicit generative models: Generative Adversarial Networks (GANs)
- Latent semantics in GANs

Conclusion


Unsupervised Learning
Generative modeling
Self-supervised feature learning