An Introduction to Deep Generative Models

Sajjad Amini

Manning College of Information and Computer Sciences
University of Massachusetts Amherst

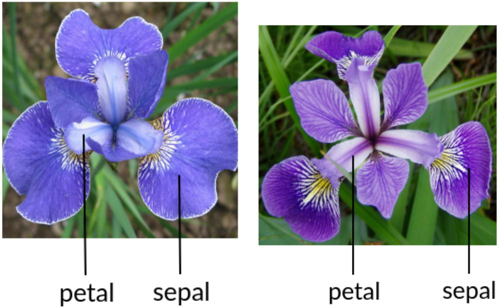
November 7, 2023

Contents

Section 1

The Quest for Deep Generative Models

IRIS dataset



(a) Setosa (b) Versicolor

Figure: Two types of Iris flower

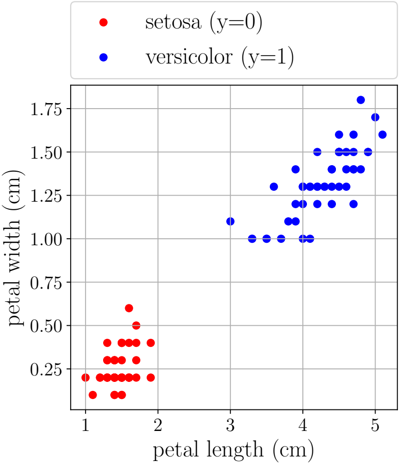


Figure: Dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$

Simple Generative Model: Binary Logistic Regression

Model

Model:

$$p_\theta(y|\boldsymbol{x}) = \mathrm{Ber}(y|\overbrace{\sigma(\boldsymbol{w}^T\boldsymbol{x} + b)}^{p})$$

$$\Rightarrow \begin{cases} p_\theta(y=1|\boldsymbol{x}) = p \\ p_\theta(y=0|\boldsymbol{x}) = 1-p \end{cases}$$

where:

$\sigma(\cdot)$	Sigmoid function
$\boldsymbol{w}$	Weight vector
b	Bias value
$\boldsymbol{\theta} = [b; \boldsymbol{w}]$	Model parameters

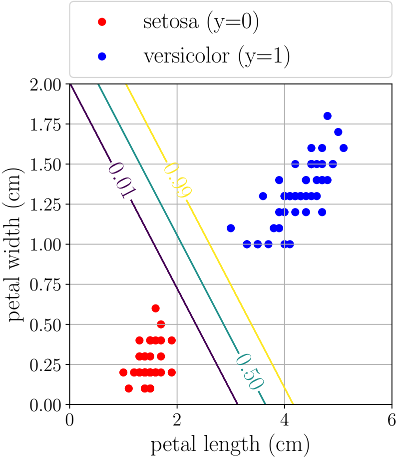


Figure: BLR contours for p

Sampling from Binary Logistic Regression

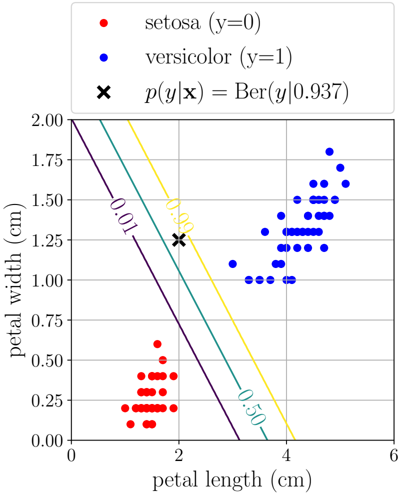


Figure: Query inference

Sampling

Sampling from $p_\theta(y|\boldsymbol{x}_q) = \mathrm{Ber}(y|0.937)$

- $u \sim \mathcal{U}(0, 1)$
- $y = \begin{cases} 0 & \text{if } u \geq 0.937) \\ 1 & \text{if } u < 0.937) \end{cases}$

BLR Limitation

Limitation

- ☞ $y \in \{0,1\}$: y is a binary random variable.

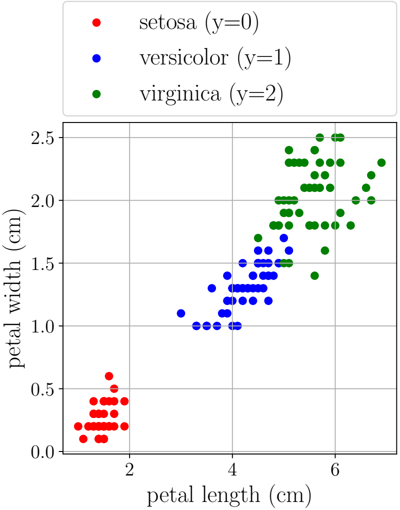


Figure: Dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$

Generative Model: Multinomial Logistic Regression

Model

Model:

$$p_\theta(y|\boldsymbol{x}) = \mathrm{Cat}(y|\ \overbrace{\mathcal{S}(\boldsymbol{W}\boldsymbol{x}+\boldsymbol{b})}^{\boldsymbol{p}=[p_0,\dots,p_{L-1}]^T}\ )$$

$$\Rightarrow \begin{cases} p_\theta(y=0|\boldsymbol{x}) = p_0 \\ \vdots \\ p_\theta(y=L-1|\boldsymbol{x}) = p_{L-1} \end{cases}$$

where:

$\mathcal{S}(\cdot)$	Softmax function
$\boldsymbol{W} \in \mathbb{R}^{C\times D}$	Weight matrix
$\boldsymbol{b} \in \mathbb{R}^{C}$	Bias vector
$\boldsymbol{\theta} = \{\boldsymbol{W}, \boldsymbol{b}\}$	Model parameters
$\boldsymbol{a} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$	logits vector

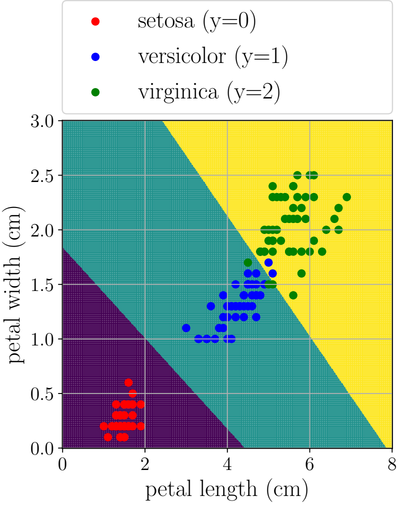


Figure: BLR contours for p

Sampling from Multinomial Logistic Regression

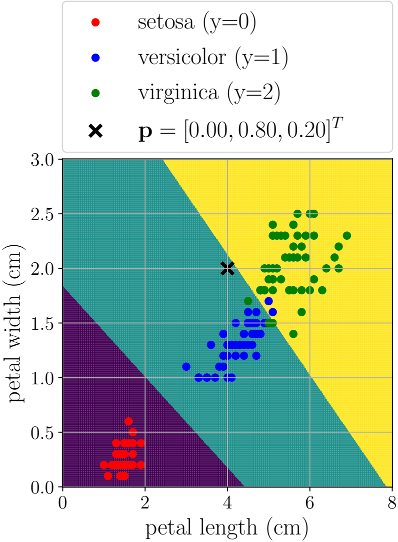


Figure: Query inference

Sampling

Sampling from $p_\theta(y|\boldsymbol{x}_q) = \mathrm{Cat}\left(y \middle| \begin{bmatrix} 0.00 \\ 0.80 \\ 0.20 \end{bmatrix}\right)$

- $\boldsymbol{p} = \begin{bmatrix} 0.0 \\ 0.2 \\ 0.8 \end{bmatrix} \Rightarrow \boldsymbol{c} = \begin{bmatrix} 0.0 \\ 0.2 \\ 1 \end{bmatrix}$
- $u \sim \mathcal{U}(0, 1)$
- $y = \begin{cases} 0 & \text{if } \quad u \leq 0 \\ 1 & \text{if } \quad 0 < u \leq 0.2 \\ 2 & \text{if } \quad u \leq 1 \end{cases}$

MLR Limitation

Limitation

☞ Linear decision boundaries

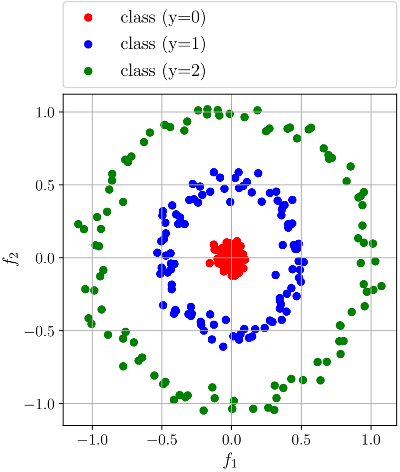
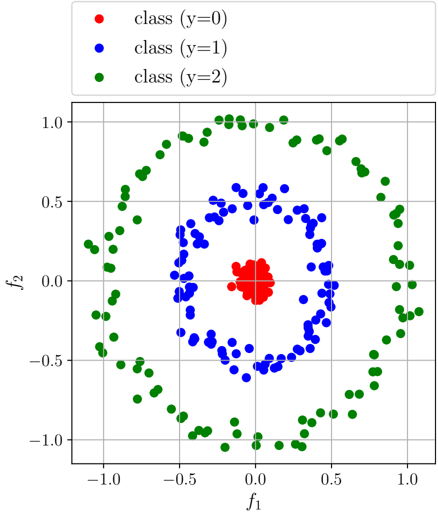
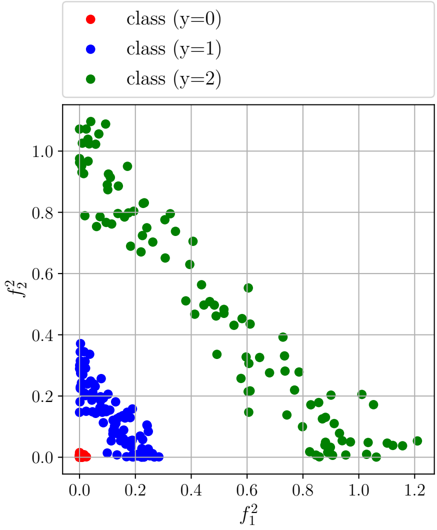


Figure: Not linearly separable dataset

Feature Transformation

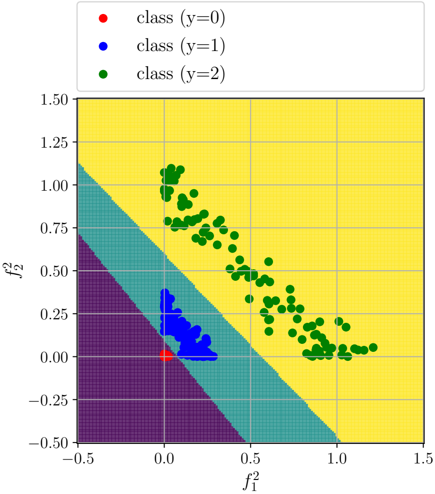


(a) Original features

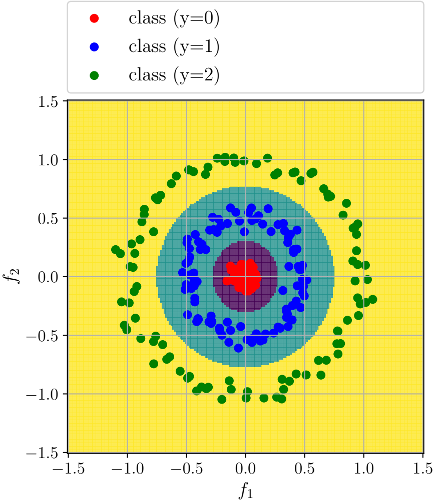


(b) Transformed features

Classification in the Transformed Space



(a) MLR result in the transformed space



(b) Transformed features

The Era of Deep Learning

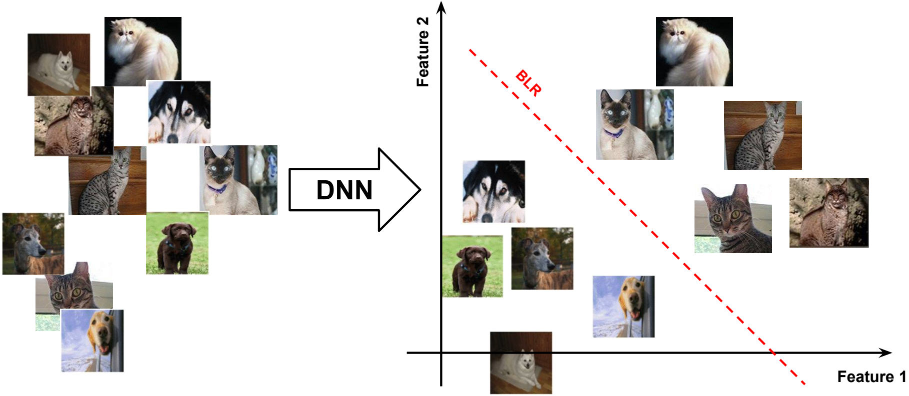
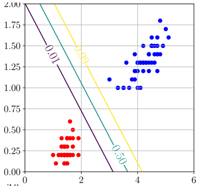
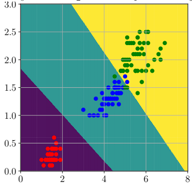
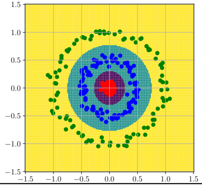


Figure: Learning suitable transformation from data using Deep Neural Networks

What We Can Do So Far

Model	Distribution	Sample
Binary Logistic Regression	$p_\theta(y\|\boldsymbol{x})$ $\begin{cases}\boldsymbol{x} \in \mathbb{R}^D \\ y \in \{0,1\}\end{cases}$	
Multinomial Logistic Regression	$p_\theta(y\|\boldsymbol{x})$ $\begin{cases}\boldsymbol{x} \in \mathbb{R}^D \\ y \in \{1,2,\ldots,L\}\end{cases}$	
DNN Feature Transformation + Multinomial Logistic Regression	$p_\theta(y\|\boldsymbol{x})$ $\begin{cases}\boldsymbol{x} \in \mathbb{R}^D \\ y \in \{1,2,\ldots,L\}\end{cases}$	

Limitation

Limitation

y is assumed to be a one-dimensional random variable and cannot be extended to the case where $\boldsymbol{y}$ is a high-dimensional random vector.

Text-to-Speech Models [1]

Text-to-Speech Models

$$p(\boldsymbol{y}|\boldsymbol{x}) : \begin{cases} \boldsymbol{y} : \text{An audio file} \\ \boldsymbol{x} : \text{A text} \end{cases}$$

$$\boldsymbol{x} = \begin{array}{c} \text{“A single Wavenet can} \\ \text{capture the characteristics of many} \\ \text{different speakers with equal fidelity,} \\ \text{not it's fast.”} \end{array} \xRightarrow{\text{Sampling} p(\boldsymbol{x}|\boldsymbol{y})} \boldsymbol{y} = \boxed{\text{Play}}$$

Text-to-Image Models [2]

Text-to-Image Models

$$p(\boldsymbol{y}|\boldsymbol{x}) : \begin{cases} \boldsymbol{y} : \text{An image} \\ \boldsymbol{x} : \text{A text} \end{cases}$$

Figure: $\boldsymbol{x}$ for $\boldsymbol{y}$ = "Teddy bears swimming at the Olympics 400m Butterfly event."

Image-to-Image Translation [3]

Image Colorization

$$p(\boldsymbol{y}|\boldsymbol{x}) : \begin{cases} \boldsymbol{y} : \text{A Colored image} \\ \boldsymbol{x} : \text{A Gray} - \text{scale image} \end{cases}$$

(a) $\boldsymbol{x}$

(b) $\boldsymbol{y}$

(c) Ground truth

Image-to-Image Translation [3]

Image Inpainting

$$p(\boldsymbol{y}|\boldsymbol{x}) : \begin{cases} \boldsymbol{y} : \text{A clean image} \\ \boldsymbol{x} : \text{A corrupted image} \end{cases}$$

(a) $\boldsymbol{x}$

(b) $\boldsymbol{y}$

(c) Ground truth

Image-to-Image Translation [3]

Image Uncropping

$$p(\boldsymbol{y}|\boldsymbol{x}) : \begin{cases} \boldsymbol{y} : \text{A clean image} \\ \boldsymbol{x} : \text{A cropped image} \end{cases}$$

(a) $\boldsymbol{x}$

(b) $\boldsymbol{y}$

(c) Ground truth

Image-to-Image Translation [3]

Image Restoration

$$p(\boldsymbol{y}|\boldsymbol{x}) : \begin{cases} \boldsymbol{y} : \text{A clean image} \\ \boldsymbol{x} : \text{A degraded image} \end{cases}$$

(a) $\boldsymbol{x}$

(b) $\boldsymbol{y}$

(c) Ground truth

The Era of Deep Learning

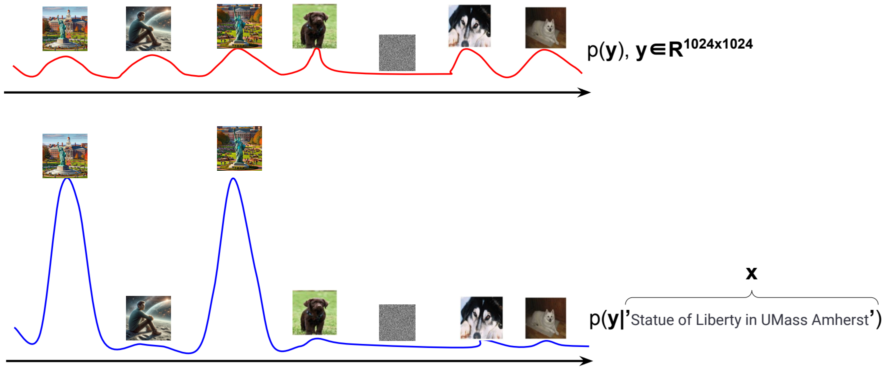
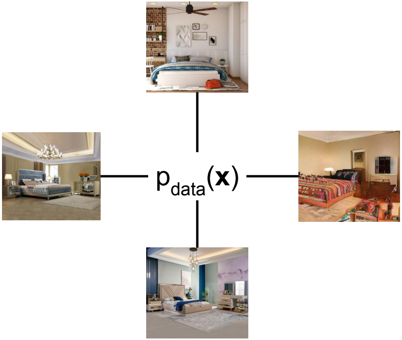


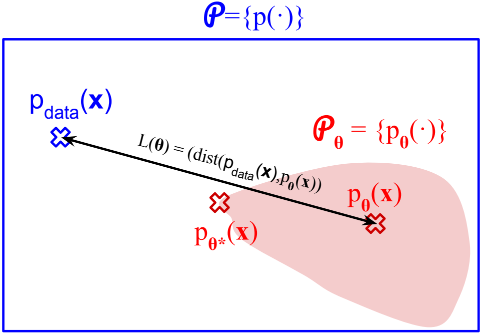
Figure: Probabilistic intuition to deep generative models

Section 2

Big Picture

Deep Generative Modeling Big Picture [4]





$$\boldsymbol{\theta}^* = \mathrm{argmin}_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$$

Generative Model

Generative Model Learning

- **Experience:**

$$\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^{N}$$

- **Objective:**
  - Estimating distribution $p_\theta(\boldsymbol{x})$
- **Tasks:**
  - Density estimation: $p_\theta(\boldsymbol{x}_{\text{new}})$
  - Generation: $\boldsymbol{x}_{\text{new}} \sim p_\theta(\boldsymbol{x})$
  - High-level representation $\boldsymbol{z}$

Section 3

Generative Adversarial Networks (Latent Variable Model)

Subsection 1

Family Selection

Family Overview

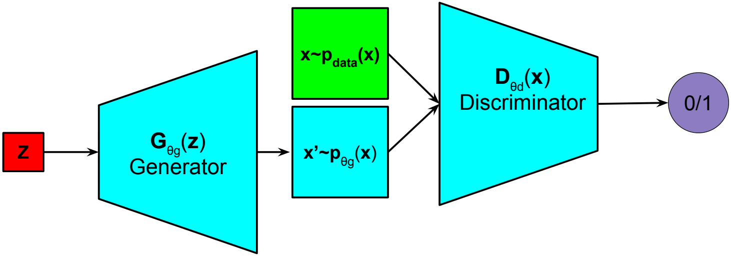


Figure: GAN Architecture

Specifications

- $\boldsymbol{z} \in \mathbb{R}^{100} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
- $\boldsymbol{x}' \in \mathbb{R}^{64\times64\times3}$: Generated Images
- $\boldsymbol{x} \in \mathbb{R}^{64\times64\times3}$: Real Images
- $\boldsymbol{\theta}_g, \boldsymbol{\theta}_d$: Generator and discriminator parameters

Generator in Deep Convolutional GAN [5]

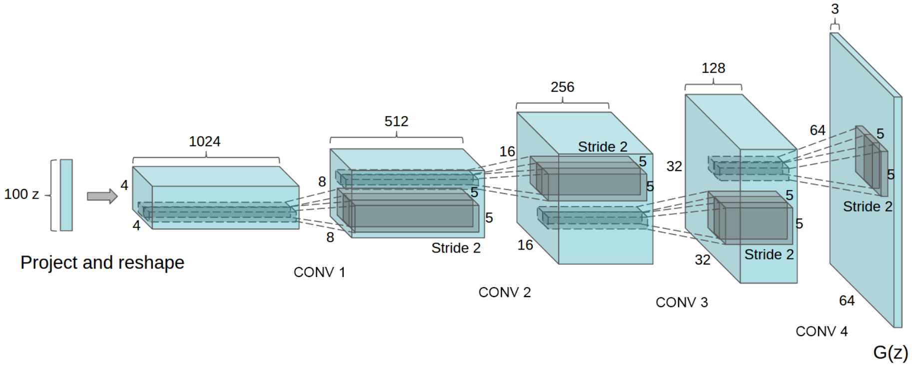


Figure: Generator Architecture in DCGAN (input: $\boldsymbol{z} \in \mathbb{R}^{100}$, output: $\boldsymbol{x}' \in \mathbb{R}^{64\times 64\times 3}$)

Discriminator in Deep Convolutional GAN [5]

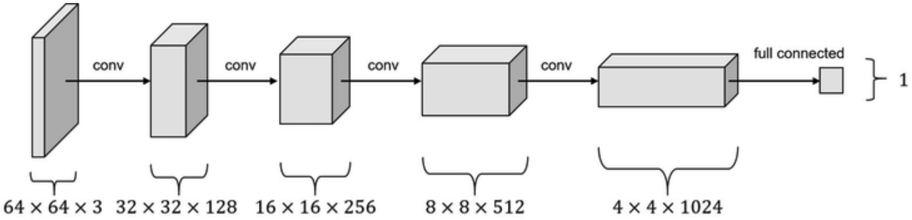


Figure: Discriminator Architecture in DCGAN (input: $\boldsymbol{x} \in \mathbb{R}^{64\times64\times3}$, output: $p \in [0, 1]$) [6]

Subsection 2

Distance Metric

Distance Metric

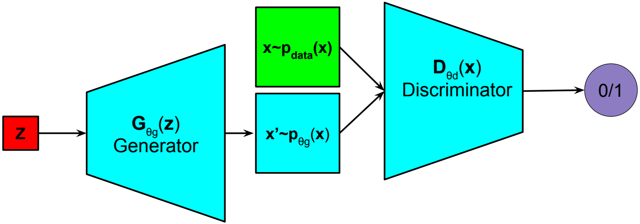


Figure: GAN Architecture

Training Generator

Assume we have trained a powerful discriminator such that: $\begin{cases} \mathcal{D}_{\theta_d}(\boldsymbol{x}) \simeq 1 \\ \mathcal{D}_{\theta_d}(\boldsymbol{x}') \simeq 0 \end{cases}$

Then a good generator can be trained as:

$$\min_{\boldsymbol{\theta}_g} \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})} \log(1 - D_{\theta_d}(\overbrace{G_{\theta_g}(\boldsymbol{z})}^{\boldsymbol{x}'}))$$

Distance Metric

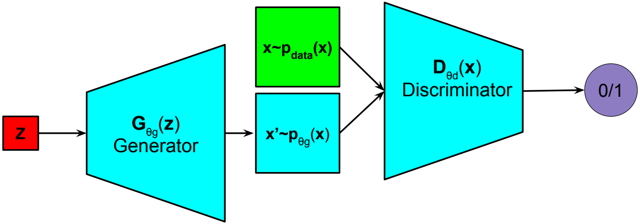


Figure: GAN Architecture

Training Discriminator

Assume we have trained a powerful generator that can generate samples quite similar to real ones. Then a good discriminator can be trained as:

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

Distance Metric

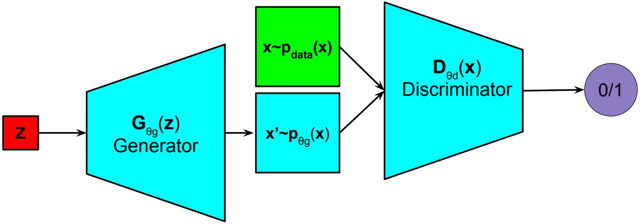


Figure: GAN Architecture

Training GAN

In practice, there is neither a powerful generator nor a powerful discriminator available at the start. Thus both should be trained in a min-max optimization as:

$$\underset{\boldsymbol{\theta}_g}{\operatorname{argmin}}\, \underset{\boldsymbol{\theta}_d}{\operatorname{argmax}} \left[\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D_{\theta_d}(\boldsymbol{x}) + \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})} \log \left(1 - D_{\theta_d}(G_{\theta_g}(\boldsymbol{z}))\right)\right]$$

Distance Metric

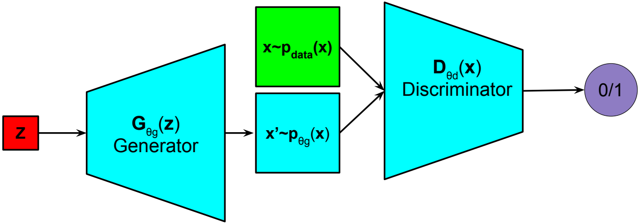


Figure: GAN Architecture

Desired Convergence situation

We have a successful learning if:

- A powerful discriminator is trained.
- The generator can fool the discriminator with high probability.

GANs

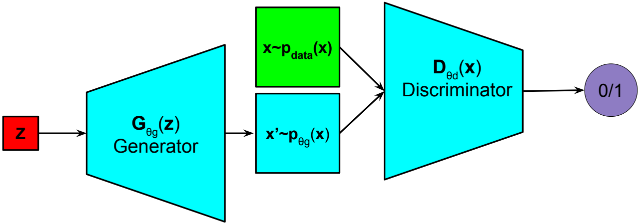


Figure: GAN Architecture

Monte Carlo Estimation

$$\underset{\boldsymbol{\theta}_g}{\operatorname{argmin}}\ \underset{\boldsymbol{\theta}_d}{\operatorname{argmax}} \left[\underbrace{\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D_{\theta_d}(\boldsymbol{x})}_{\frac{1}{K}\sum_{k=1}^{K} \log D_{\theta_d}(\boldsymbol{x}_k)} + \underbrace{\mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})} \log\left(1 - D_{\theta_d}(G_{\theta_g}(\boldsymbol{z}))\right)}_{\frac{1}{K}\sum_{k=1}^{K} \log\left(1-D_{\theta_d}(G_{\theta_g}(\boldsymbol{z}_k))\right)} \right]$$

Subsection 3

Applications

Samples from Deep Convolutional GAN

Figure: Generated sample images from DCGAN trained on bedroom images [5]

Application of Latent Representation [5]

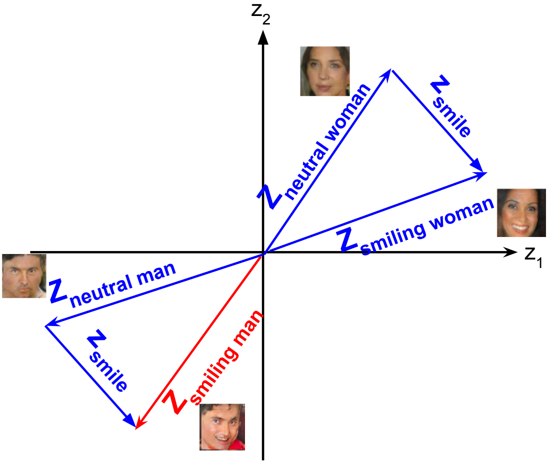


Figure: Finding the direction of *smile* in latent space $\boldsymbol{z}_{\text{smile}}$

Application of Latent Representation

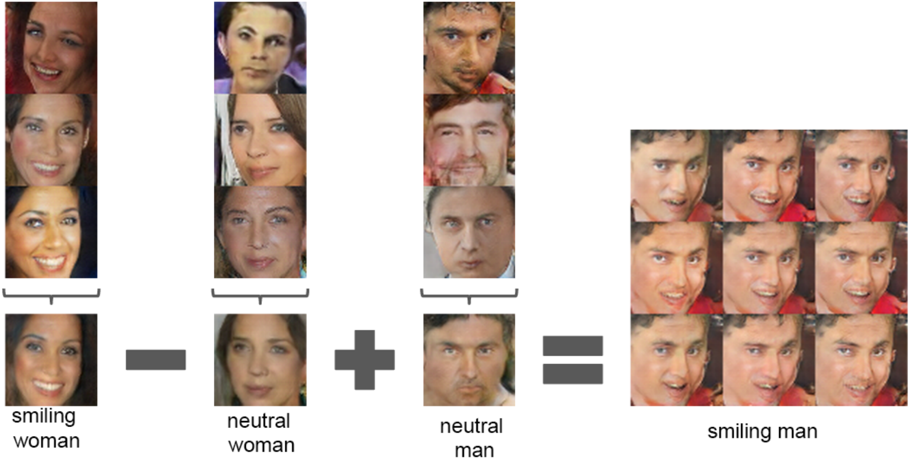


Figure: Finding the direction of *smile* in latent space $\boldsymbol{z}$ [5]

Application of Latent Representation

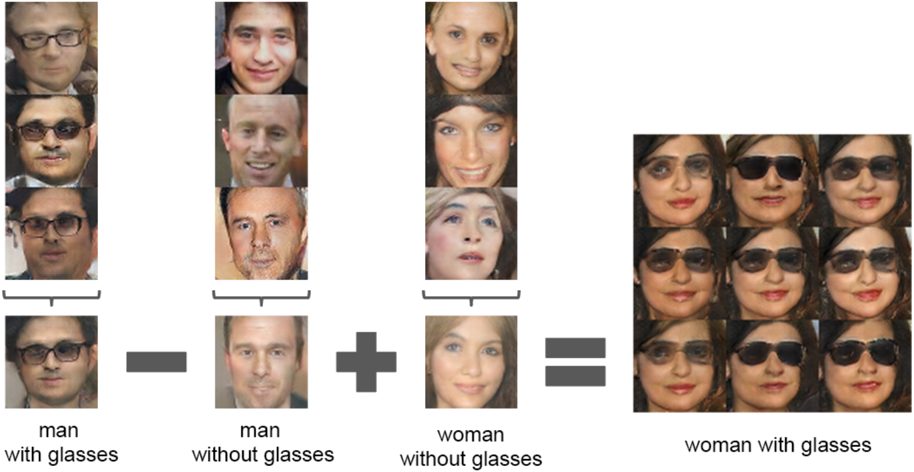


Figure: Finding the direction of *wearing glasses* in latent space $\boldsymbol{z}$ [5]

Section 4

Autoregressive Models (Likelihood Based Model)

Subsection 1

Family Selection

Sample Dataset

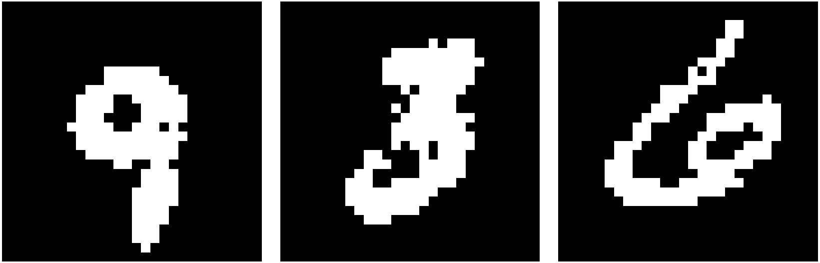
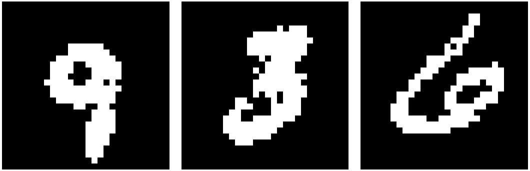


Figure: Samples from binary MNIST dataset ($\boldsymbol{x} \in \mathbb{R}^{784}$ and $x_i \in \{0, 1\}$)

Full Joint Using Chain Rule

$$p(\boldsymbol{x}) = p(x_1)p(x_2|x_1)p(x_3|x_2, x_1)\ldots p(x_{784}|x_{783}, \ldots, x_1)$$

(a) Typical $\mathcal{P}$ and $\mathcal{P}_\theta$



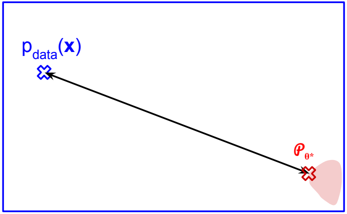
(b) Generated samples from $p_{\theta^\star}$ assuming $\mathcal{P} = \mathcal{P}_\theta$ we can find $p_{\theta^\star}$ efficiently

Challenges

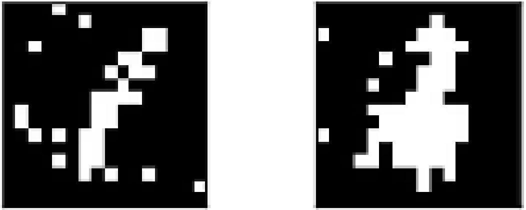
$$p(\boldsymbol{x}) = \overbrace{p(x_1)}^{\#p=1} \overbrace{p(x_2|x_1)}^{\#p=2} \dots \overbrace{p(x_{784}|x_{783}, \dots, x_1)}^{\#p=2^{n-1}}$$

- Intractable number of parameters: $2^{784} - 1$
- Hard to optimize

Small Subset Search



(a) Typical $\mathcal{P}$ and $\mathcal{P}_\theta$



(b) Generated samples from $p_{\theta^\star}$ assuming we can find $p_{\theta^\star}$ efficiently

Challenges

$$p(\boldsymbol{x}) = \overbrace{p(x_1)}^{\#p=1}\overbrace{p(x_2)}^{\#p=1}\dots\overbrace{p(x_{784})}^{\#p=1}$$

- The resulting $p_\theta(\boldsymbol{x})$ cannot capture the data distribution.

Chain Rule

Chain Rule

Using the chain rule, we have:

$$p(\boldsymbol{x}) = p(x_1)p(x_2|\boldsymbol{x}_{<2})\dots p(x_i|\boldsymbol{x}_{<i})\dots p(x_{784}|\boldsymbol{x}_{<784}),\ \boldsymbol{x}_{<i} \triangleq [x_1,\dots,x_{i-1}]^T$$

Resembelence to BLR

Consider a factor in the right-hand side of the chain rule as:

$$p(x_i|\boldsymbol{x}_{<i}), x_i \in \{0,1\}$$

This problem can be easily solved via BLR (or BLR over DNN features) as:

$$p(x_i|\boldsymbol{x}_{<i};\boldsymbol{w}_i,b_i) = \mathrm{Ber}(x_i|\sigma(b_i + \boldsymbol{w}_i^T\boldsymbol{x}_{<i})), \begin{cases} b_i \in \mathbb{R} \\ \boldsymbol{w}_i \in \mathbb{R}^{i-1} \end{cases}$$

Our New Parametric Familty

Name

The new parametric distribution family is called **Deep Autoregressive Models**, as they calculate the current state x_i given its past $\boldsymbol{x}_{<i}$.

Number of Parameters

We have:

$$p(\boldsymbol{x}) = p(x_1)p(x_2|\boldsymbol{x}_{<2})\dots p(x_n|\boldsymbol{x}_{<n})$$

$$p(\boldsymbol{x}) = \overbrace{\mathrm{Ber}(x_1|\sigma(b_1))}^{\#p=1}\overbrace{\mathrm{Ber}(x_2|\sigma(b_2 + w_2^T\boldsymbol{x}_{<2}))}^{\#p=2}\dots\overbrace{\mathrm{Ber}(x_n|\sigma(b_n + \boldsymbol{w}_n^T\boldsymbol{x}_{<n}))}^{\#p=n}$$

Thus

- $\#$Parameters for $\mathcal{P}_\theta$: $1 + 2 + 3 + \ldots + n = \frac{n(n+1)}{2}$.
- $\boldsymbol{\theta} = \{b_1, b_2, w_2, \ldots, b_n, \boldsymbol{w}_n\}$

Subsection 2

Distance Metric

Model Likelihood Estimation

Model Likelihood Estimation

We are interested in solving the following problem:

$$\boldsymbol{\theta}^{\star} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}} \left[\log p_{\theta}(\boldsymbol{x}) \right]$$

Solution via Monte Carlo Estimate

Using Monte Carlo estimate we have:

$$\mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}} \left[\log p_{\theta}(\boldsymbol{x}) \right] \simeq \frac{1}{|\mathcal{D}|} \sum_{i=1}^{N} \log p_{\theta}(\boldsymbol{x}_i)$$

Thus:

$$\boldsymbol{\theta}^{\star} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \frac{1}{|\mathcal{D}|} \sum_{i=1}^{N} \log p_{\theta}(\boldsymbol{x}_i)$$

Last Piece of the Puzzle

Calculating $p_\theta(\boldsymbol{x}_i)$

Assume $\boldsymbol{x} \in \mathbb{R}^2$ and our model is:

$$p(\boldsymbol{x}) = \mathrm{Ber}(x_1|\sigma(b_1))\ \ \mathrm{Ber}(x_2|\sigma(b_2 + w_2 \times x_1))$$

To compute the probability $\boldsymbol{x}_i = [1, 0]^T$ w have:

$$\begin{aligned} p(\boldsymbol{x}_i = \begin{bmatrix} 1 \\ 0 \end{bmatrix}) &= \mathrm{Ber}(x_1 = 1|\sigma(b_1))\ \ \mathrm{Ber}(x_2 = 0|\sigma(b_2 + w_2 \times 1)) \\ &= \sigma(b_1) \times [1 - \sigma(b_2 + w_2 \times 1)] \end{aligned}$$

Subsection 3

Sampling

Sampling

Assume $\boldsymbol{x} \in \mathbb{R}^2$ and our trained model is (trained parameters are: $\theta^\star = \{b_1^\star, b_2^\star, w_2^\star\}$):

$$p(\boldsymbol{x}) = \mathrm{Ber}(x_1 | \sigma(b_1^\star)) \;\; \mathrm{Ber}(x_2 | \sigma(b_2^\star + w_2^\star \times x_1))$$

To sample we do:

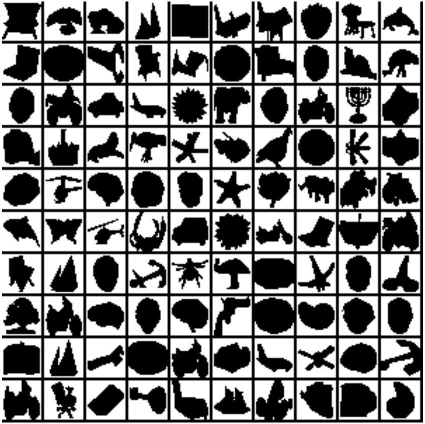
- $\widehat{x}_1 \leftarrow \mathrm{Sample}\Big(\mathrm{Ber}(x_1 | \sigma(b_1^\star)) \Big)$
- $\widehat{x}_2 \leftarrow \mathrm{Sample}\Big(\mathrm{Ber}(x_2 | \sigma(b_2^\star + w_2^\star \times \widehat{x}_1)) \Big)$

The generated sample is: $\widehat{\boldsymbol{x}} = [\widehat{x}_1, \widehat{x}_2]^T$

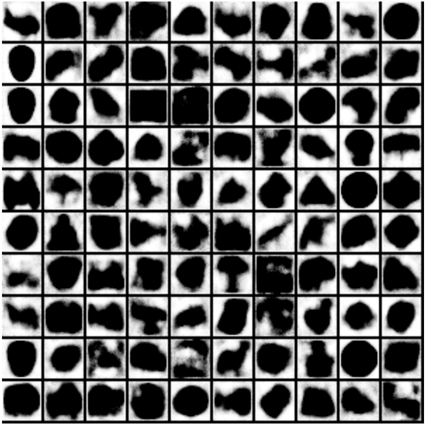
Subsection 4

Application

FVSBN



(a) Dataset samples



(b) Generated samples

Figure: FVSBN performance over Caltech 101 dataset [7]

Section 5

Latent Variable Models with Likelihood Training

Evidence Lower Bound (ELBO)

ELBO

ELBO is a tractable lower bound to data log-likelihood as:

$$\mathrm{ELBO}(\boldsymbol{x}_k) \leq \log p_\theta(\boldsymbol{x}_k)$$

Thus the following problem can be replaced for generative modeling:

$$\boldsymbol{\theta}^\star = \underset{\theta}{\operatorname{argmax}} \frac{1}{K} \sum_{k=1}^{K} \mathrm{ELBO}(\boldsymbol{x}_k)$$

Models Using ELBO

- Variational AutoEncoder
- Diffusion probabilistic models

Change-of-Variable Technique

Change-of-Variable Technique

Consider the following two assumptions:

- $\boldsymbol{z} \sim \mathcal{N}(0, \boldsymbol{I}), \boldsymbol{z} \in \mathbb{R}^D$
- $\boldsymbol{x} = \boldsymbol{f}_\theta(\boldsymbol{z})$ where $\boldsymbol{f}(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ is a nonlinear invertible function implemented by a DNN

Then $p_\theta(\boldsymbol{x}_i)$ can be calculated analytically using change-of-variable technique.

Models Using Change-of-Variable Technique

- Normalizing Flow

Section 6

Summary

Summary

Model	Density	Sampling	Training	Latent	Architecture
Autoregressive	Exact	Slow	MLE*	-	RNN/Transformer
GAN	-	Fast	JSD**	$\mathbb{R}^L$	Generator/Discriminator
VAE	Lower Bound	Fast	ELBO	$\mathbb{R}^L$	Encoder/Decoder
Normalizing Flow	Exact	Slow/Fast	MLE	$\mathbb{R}^D$	Invertible
Diffusion	Lower Bound	Slow	ELBO	$\mathbb{R}^D$	Encoder/Decoder

*Maximum Likelihood Estimation

**Jenson-Shanon Divergence

Table: Summary of Deep Generative Models

References I

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu,
"Wavenet: A generative model for raw audio,"
arXiv preprint arXiv:1609.03499, 2016.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al.,
"Photorealistic text-to-image diffusion models with deep language understanding,"
Advances in Neural Information Processing Systems, vol. 35, pp. 36479–36494, 2022.

Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi,
"Palette: Image-to-image diffusion models,"
in *ACM SIGGRAPH 2022 Conference Proceedings*, 2022, pp. 1–10.

Stefano Ermon and Yang Song,
"Cs236: Deep generative models," Fall 2021.

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

References II

Guangyuan Zhang, Xiaoping Rui, Stefan Poslad, Xianfeng Song, Yonglei Fan, and Bang Wu,
"A method for the estimation of finely-grained temporal spatial human population density distributions based on cell phone call detail records,"
Remote Sensing, vol. 12, no. 16, pp. 2572, 2020.

Zhe Gan, Ricardo Henao, David Carlson, and Lawrence Carin,
"Learning deep sigmoid belief networks with data augmentation,"
in *Artificial Intelligence and Statistics*. PMLR, 2015, pp. 268–276.