Lecture 1: Introduction

Deep Generative Models

Sajjad Amini

Department of Electrical Engineering
Sharif University of Technology

Contents

Section 1

Supervised vs. Unsupervised Learning

Supervised Learning

- **Observations:** $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$

Supervised Learning

- **Observations:** $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$
- **Applications:**
  - Classification
  - Regression
  - ...

Supervised Learning

- **Observations:** $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$
- **Applications:**
  - Classification
  - Regression
  - ...
- **Model:** $p_\theta(y|\boldsymbol{x})$

Supervised Learning

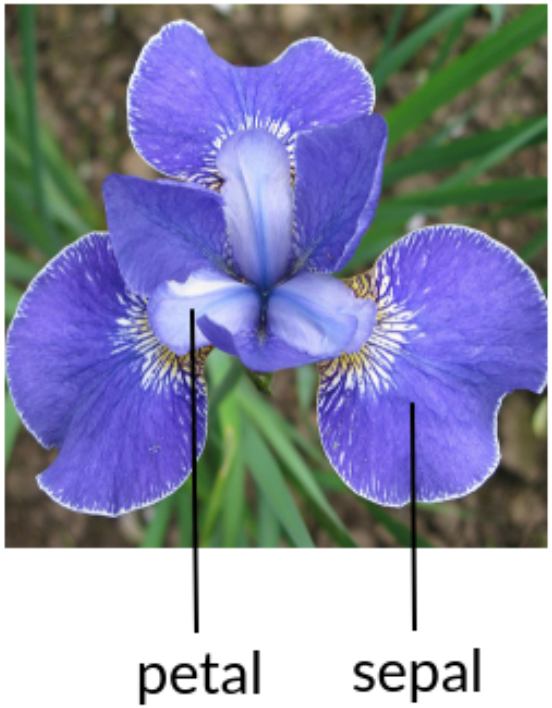
- **Observations:** $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$
- **Applications:**
  - Classification
  - Regression
  - ...
- **Model:** $p_\theta(y|\boldsymbol{x})$
- **Phases:**
  - *Training Phase:*

$$\boldsymbol{\theta}^\star = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{n=1}^{N} \text{Loss}\Big(y_n, p_\theta(y|\boldsymbol{x}_n)\Big)$$

  - *Application Phase:*

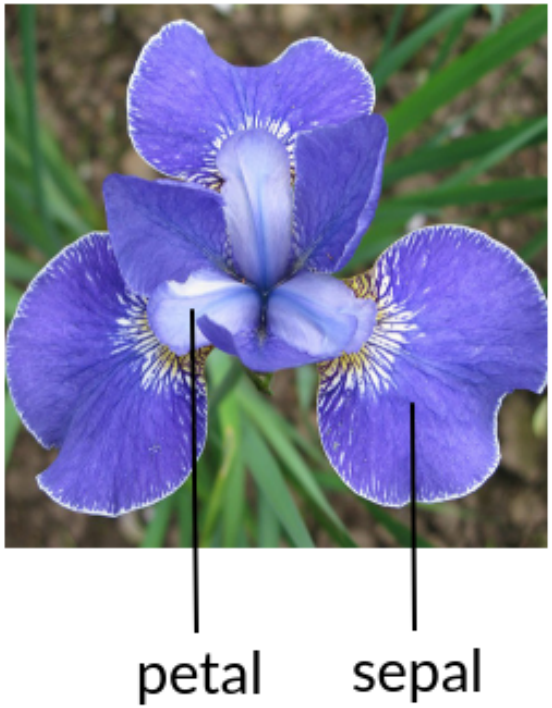
$$p_{\boldsymbol{\theta}^\star}(y|\boldsymbol{x}_{\text{new}})$$

Classification



(a) Setosa ($y = 1$)

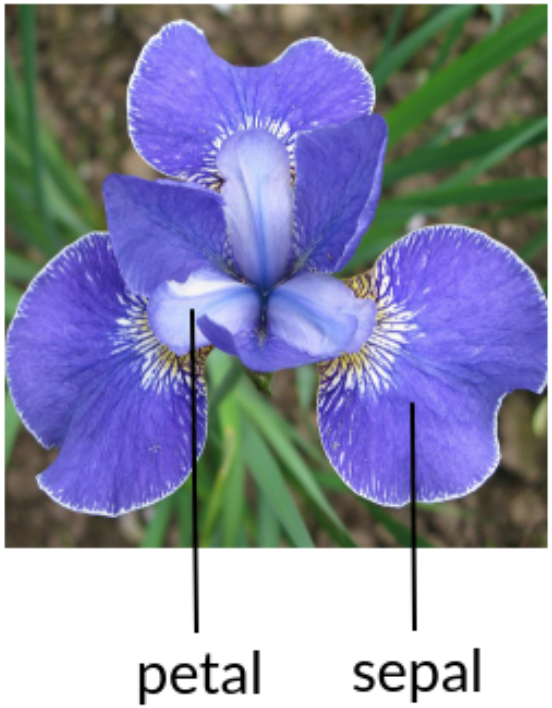
Classification



(a) Setosa ($y = 1$)

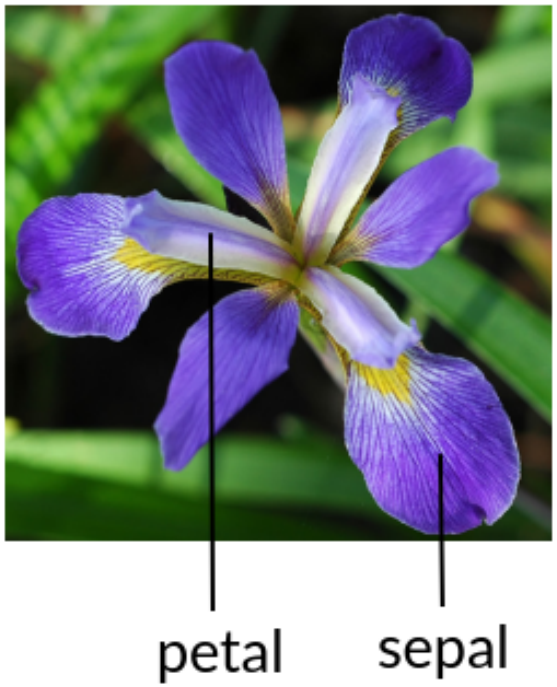
(b) Versicolor ($y = 2$)

Classification



(a) Setosa ($y = 1$)

(b) Versicolor ($y = 2$)



(c) Virginica ($y = 3$)

Figure: Iris flower dataset ($\boldsymbol{x} \in \mathbb{R}^{256 \times 256 \times 3}$ and $y \in \{1, 2, 3\}$)

Classification

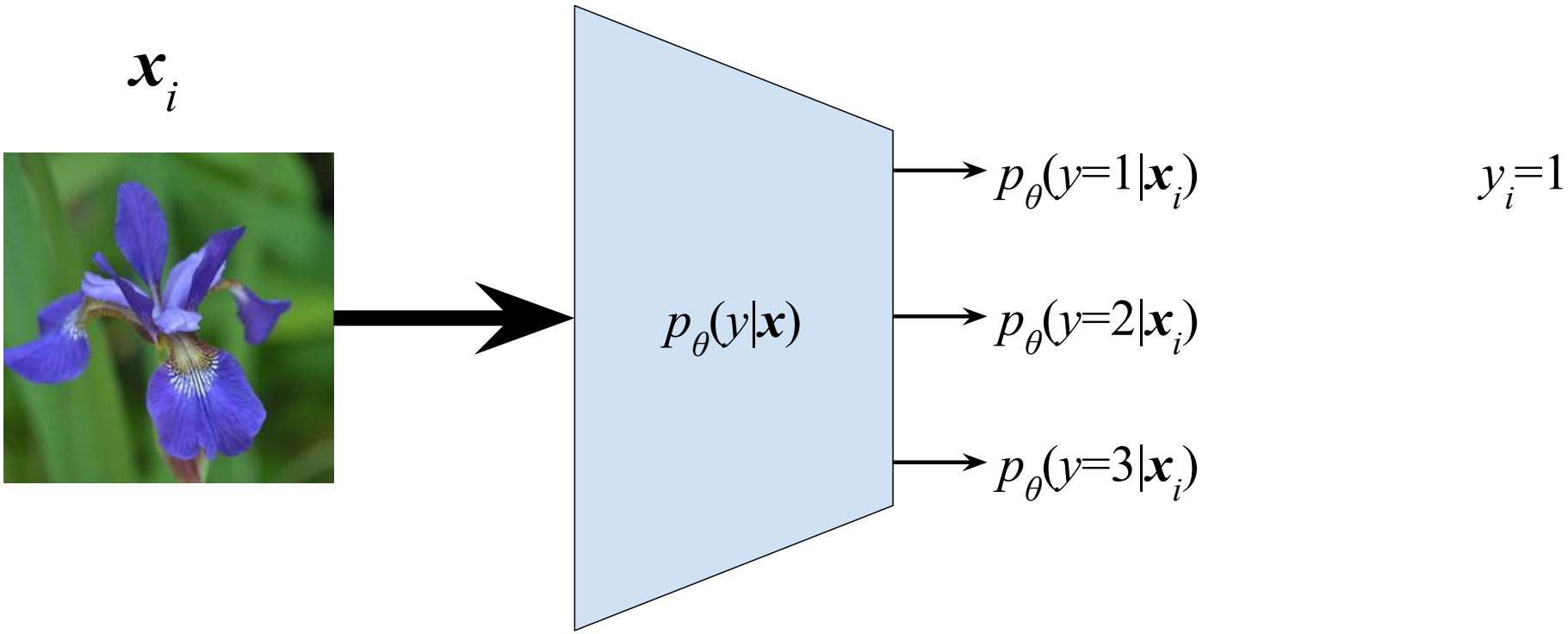


Figure: Training step 1: Evaluate $p_\theta(y|\boldsymbol{x}_i)$

Classification

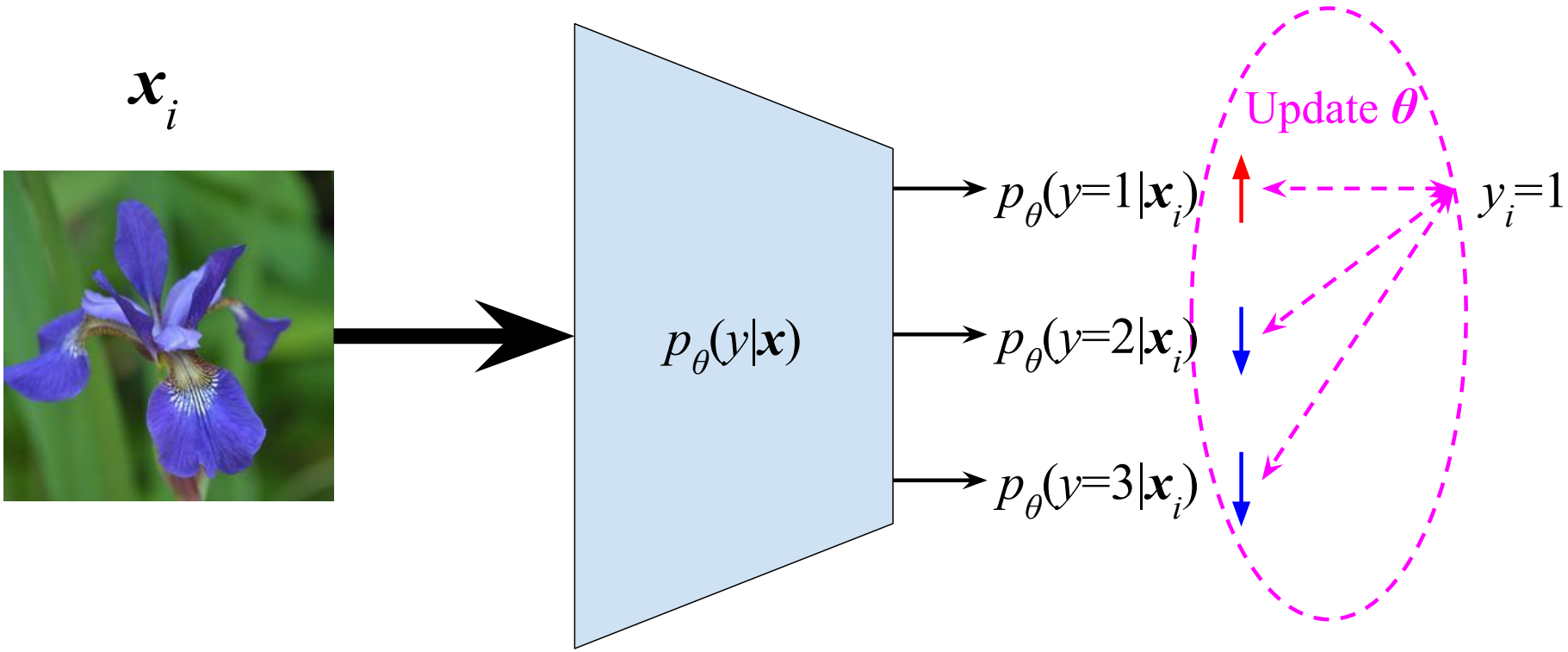


Figure: Training step 2: Update $\boldsymbol{\theta}$ to minimize cross entropy

Classification

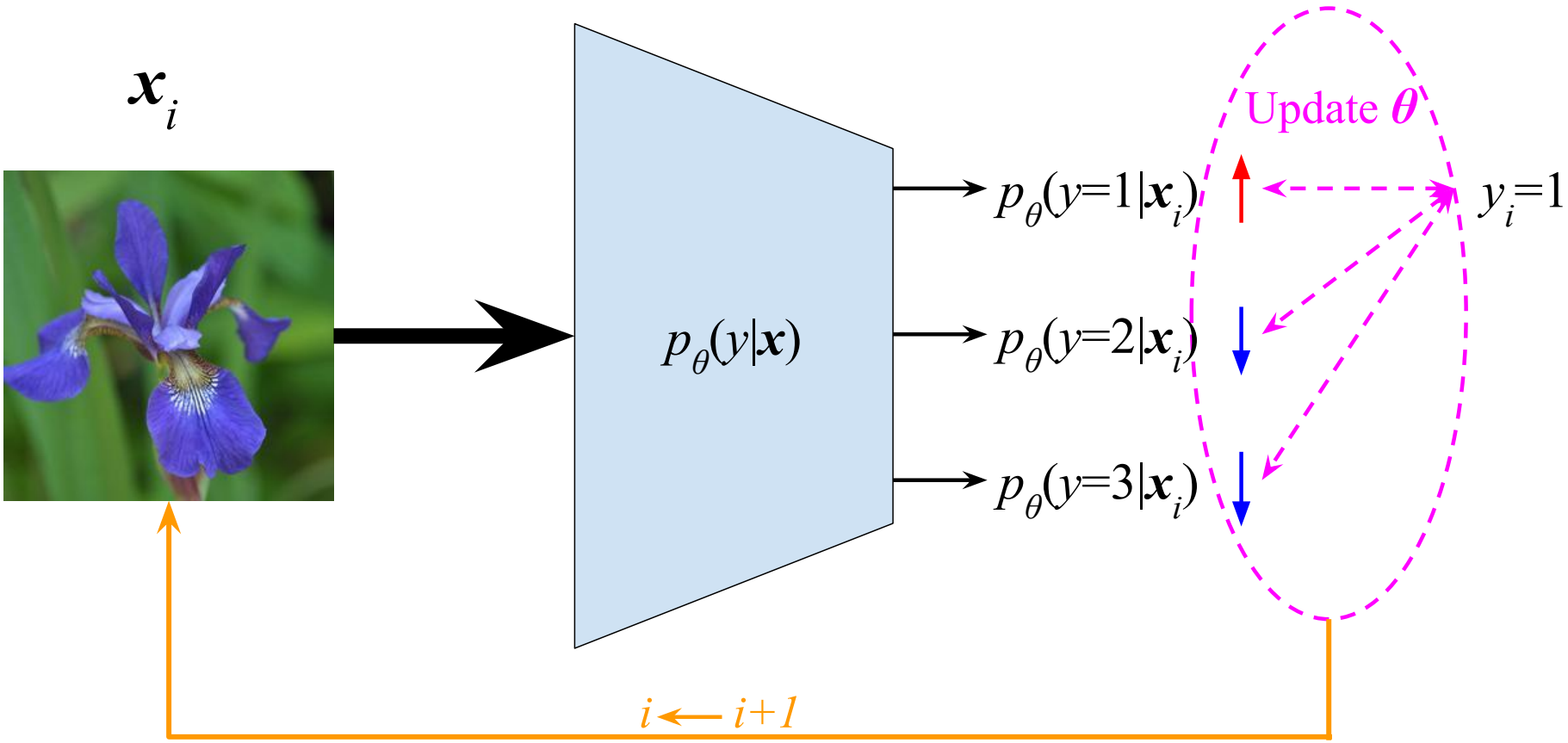


Figure: Training step 3: Go to the next training data

Classification

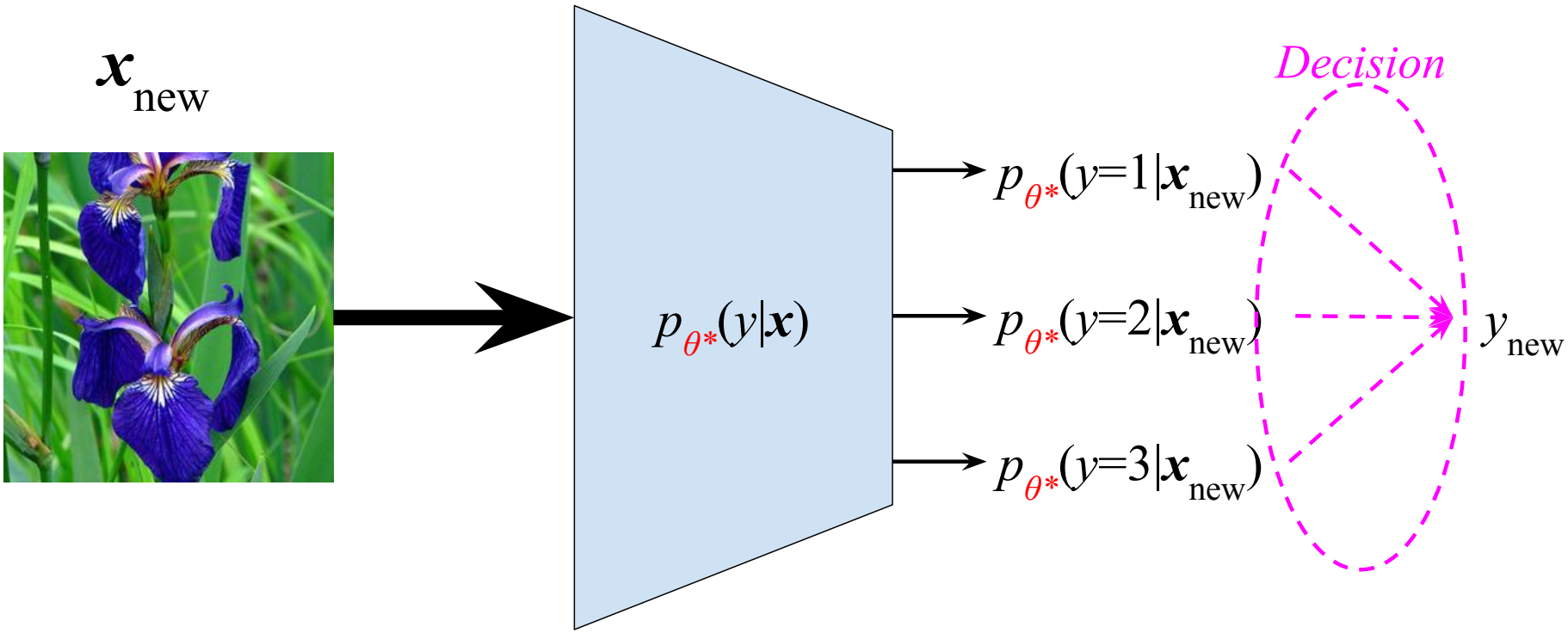


Figure: Inference: Evaluating the trained model $\boldsymbol{\theta}^\star$

Multinomial Logistic Regression Model

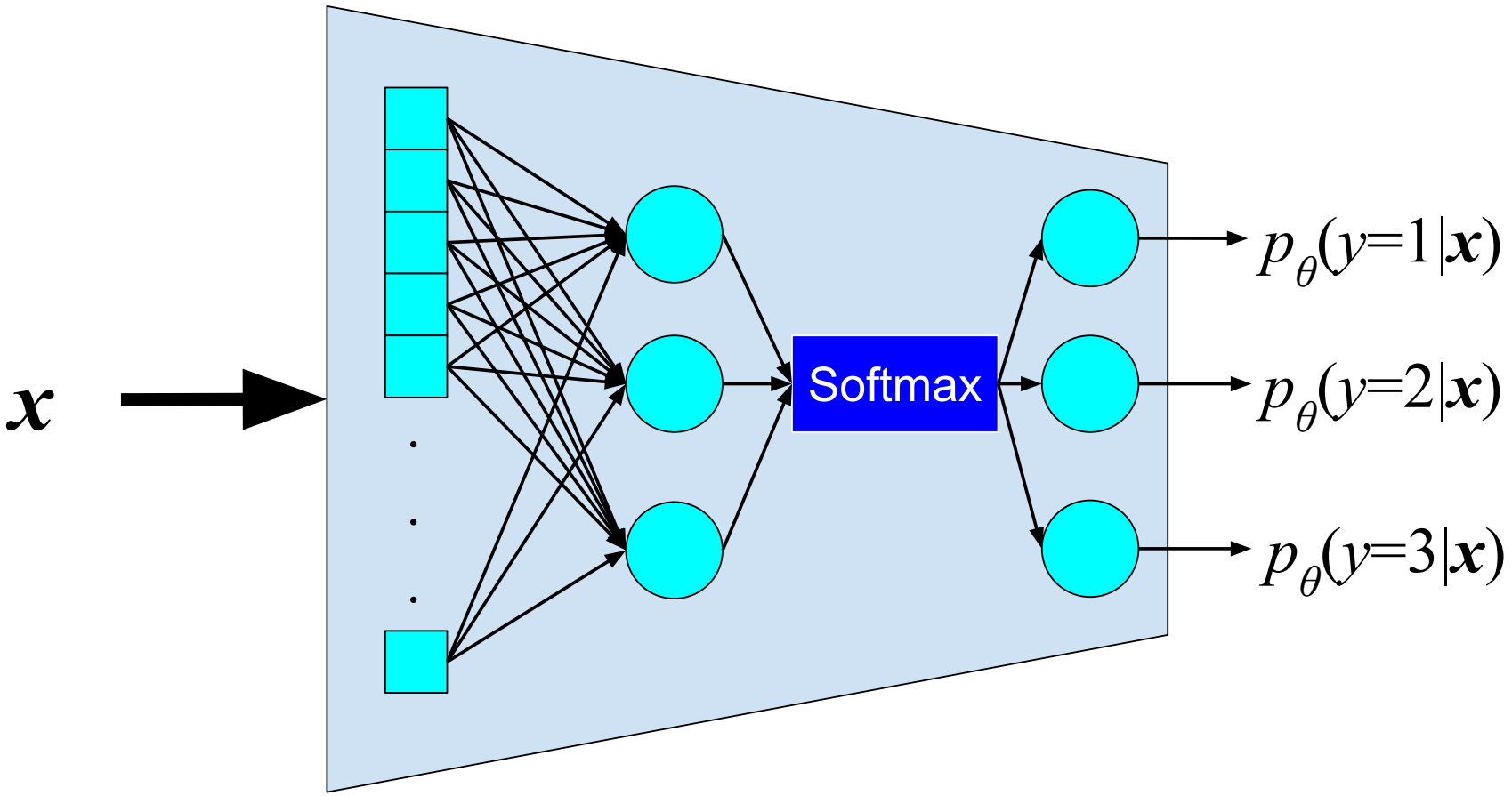


Figure: Multinomial logistic regression (MLR) model to implement $p_\theta(y|\boldsymbol{x})$ for Iris flower classification

Unsupervised Learning

- **Observations:** $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^{N}$

Unsupervised Learning

- **Observations:** $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^{N}$
- **Applications:**
  - Dimensionality reduction
  - Clustering
  - ...

Unsupervised Learning

- **Observations:** $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^{N}$
- **Applications:**
  - Dimensionality reduction
  - Clustering
  - ...
- **Model:** *Dimensionality Reduction* $\Rightarrow \boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}, \begin{cases} \boldsymbol{x} \in \mathbb{R}^D \\ \boldsymbol{W} \in \mathbb{R}^{L \times D} \\ \boldsymbol{y} \in \mathbb{R}^L \end{cases}$

Unsupervised Learning

- **Observations:** $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^{N}$
- **Applications:**
  - Dimensionality reduction
  - Clustering
  - ...
- **Model:** *Dimensionality Reduction* $\Rightarrow \boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}, \begin{cases} \boldsymbol{x} \in \mathbb{R}^D \\ \boldsymbol{W} \in \mathbb{R}^{L \times D} \\ \boldsymbol{y} \in \mathbb{R}^L \end{cases}$
- **Phases:**
  - *Training Phase:*

$$\boldsymbol{W}^\star = \underset{\boldsymbol{W}}{\operatorname{argmin}} \sum_{n=1}^{N} \operatorname{Loss}\Big(\boldsymbol{x}_n, \boldsymbol{W}^T \boldsymbol{W} \boldsymbol{x}\Big)$$

  - *Application Phase:*

$$\boldsymbol{y}_{\text{new}} = \boldsymbol{W}^\star \boldsymbol{x}_{\text{new}}$$

Dimensionality Reduction

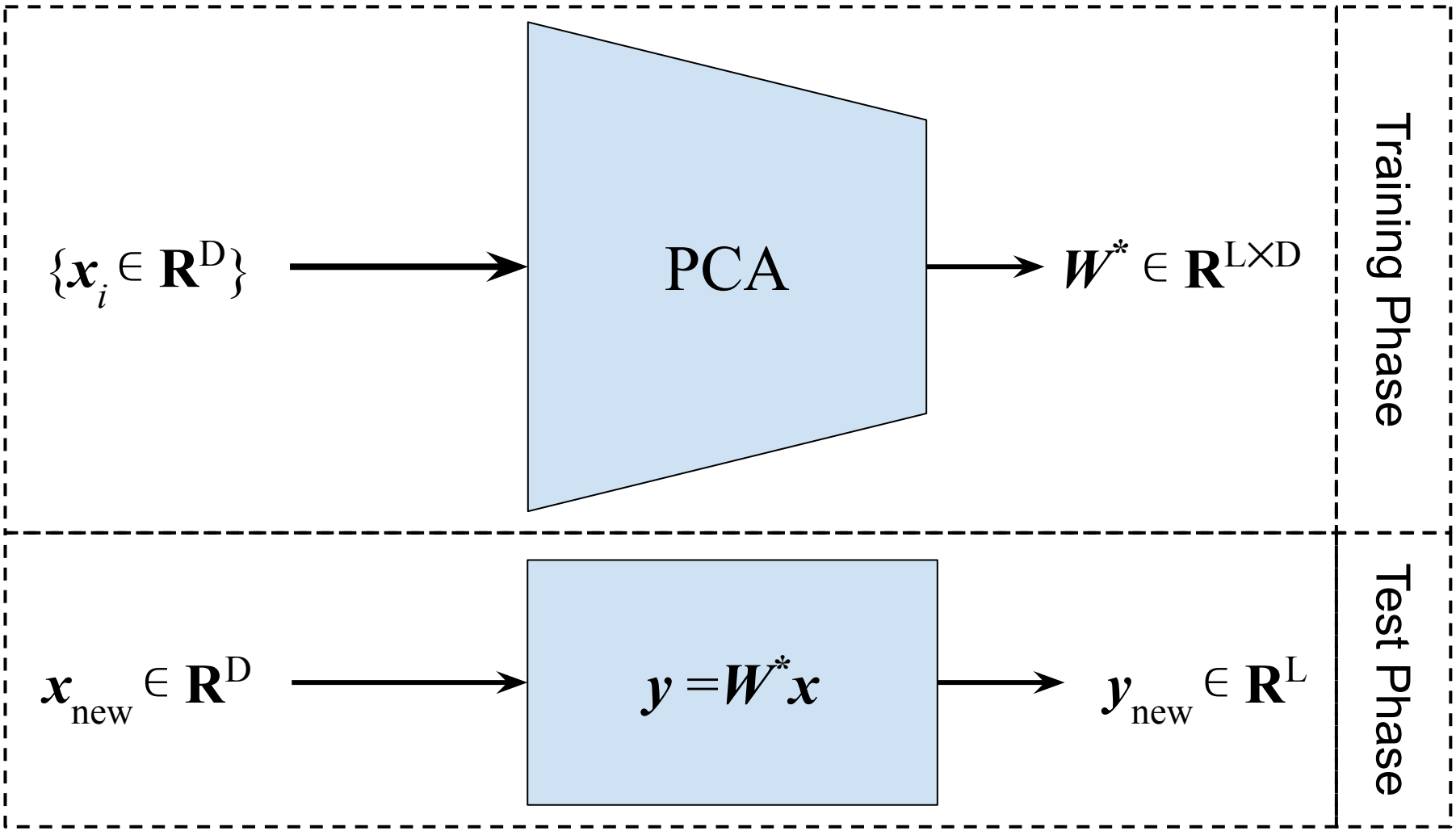


Figure: Training and Test for Dimensionality Reduction

Dimensionality Reduction

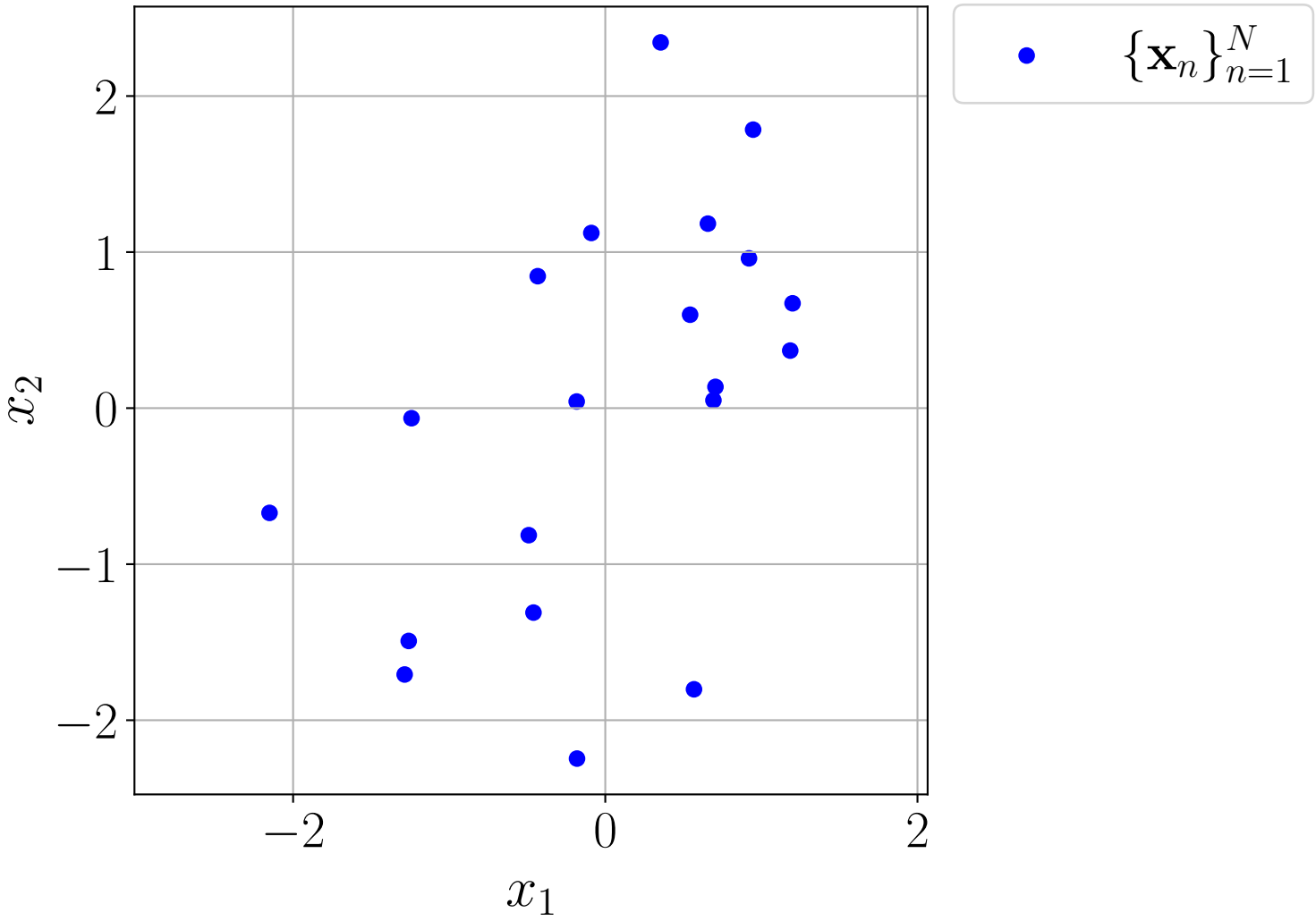


Figure: Training data for dimensionality reduction

Dimensionality Reduction

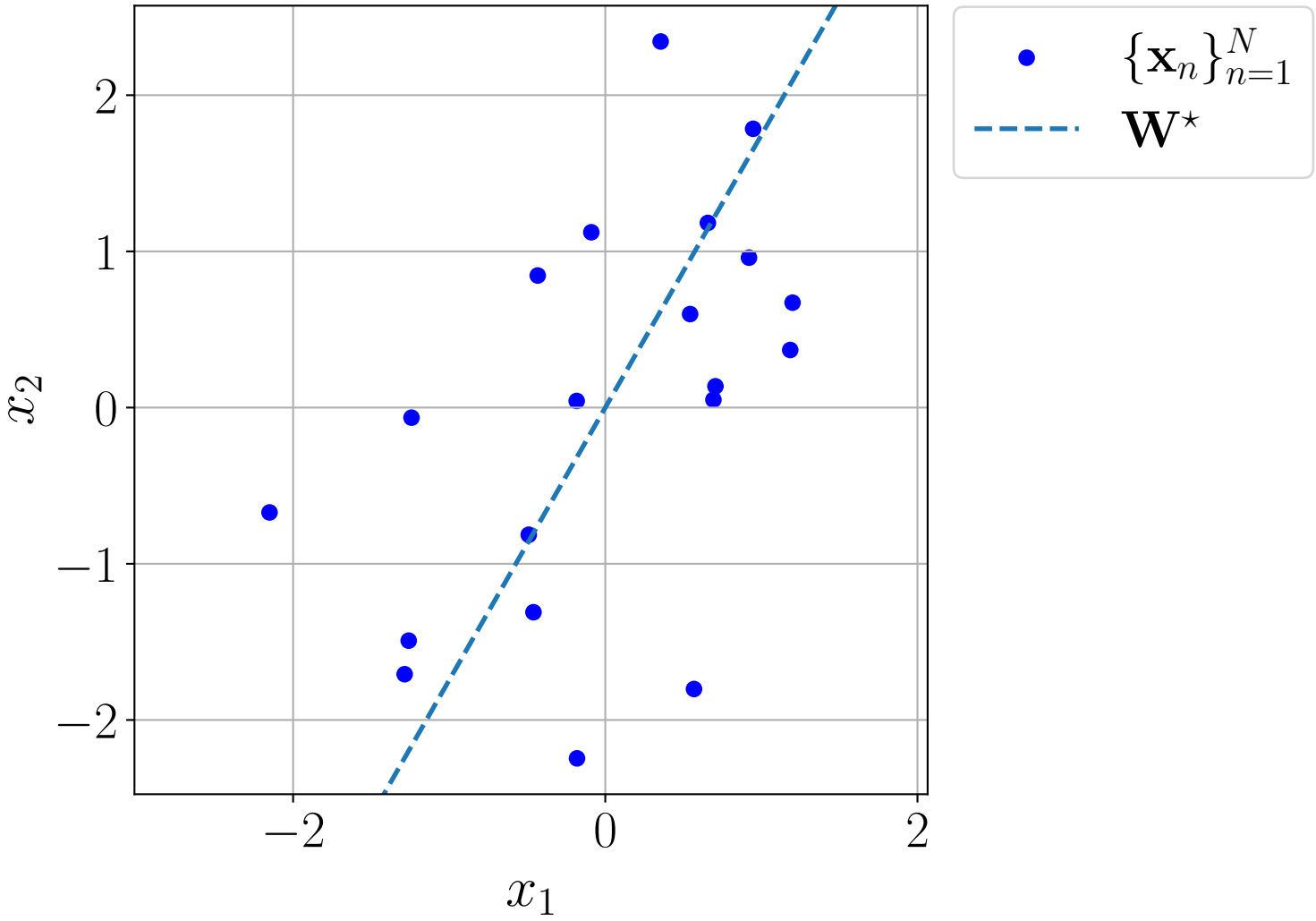


Figure: Learning principle component or projection matrix $\boldsymbol{W}^\star$

Dimensionality Reduction

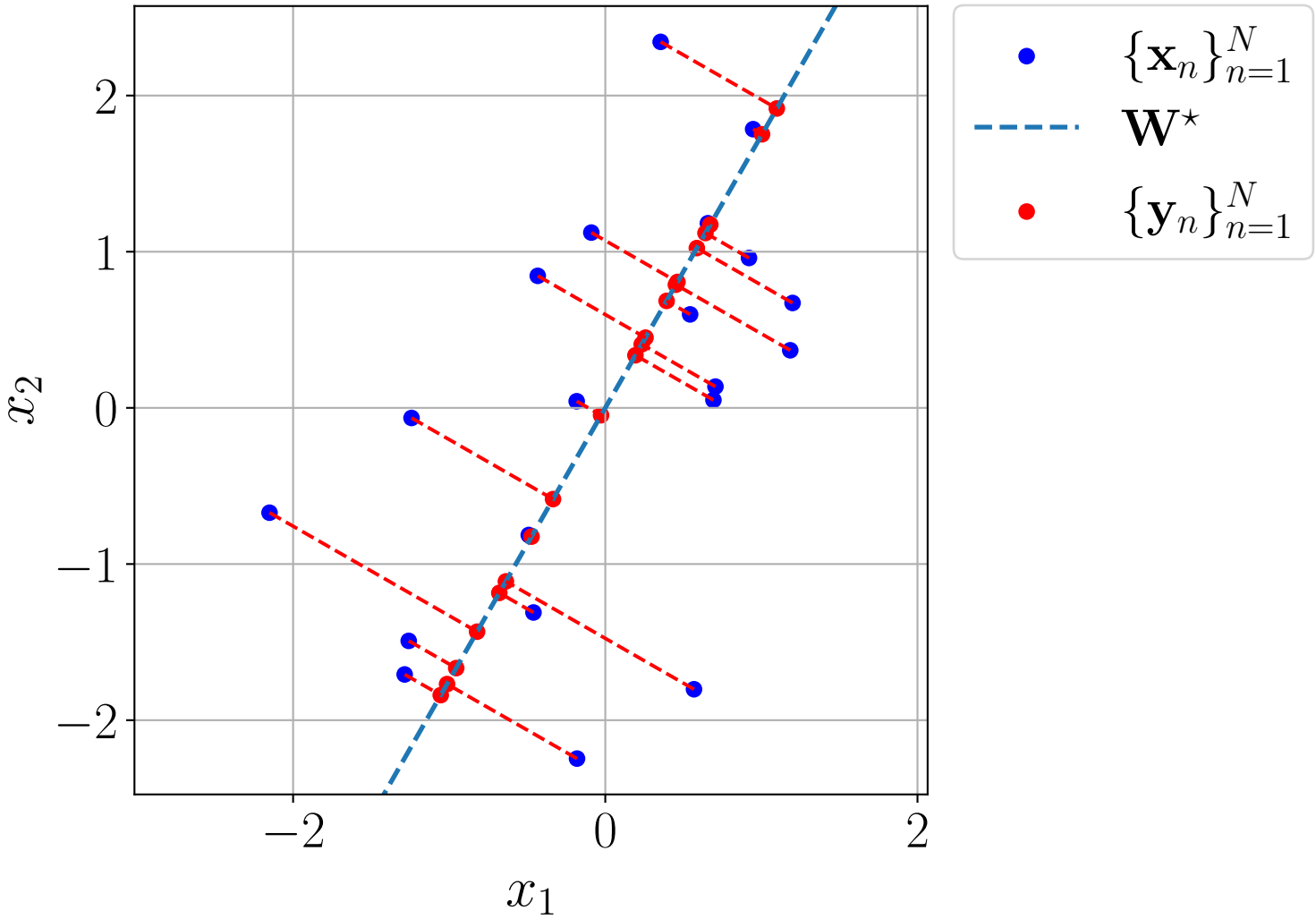


Figure: Reducing the dimensionality of training data

Altogether

Supervised Learning

- **Experience:**

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

- **Applications:**
  - Classification
  - Regression
  - Semantic segmentation
  - ...
- **Objective:**
  - Estimating $p_{\text{data}}(y|\boldsymbol{x})$

Altogether

Supervised Learning

- **Experience:**

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

- **Applications:**
  - Classification
  - Regression
  - Semantic segmentation
  - ...
- **Objective:**
  - Estimating $p_{\text{data}}(y|\boldsymbol{x})$

Unsupervised Learning

- **Experience:**

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

- **Applications:**
  - Clustering
  - Dimensionality Reduction
  - Density estimation
  - ...
- **Objective:**
  - Estimating $p_{\text{data}}(\boldsymbol{x})$

Section 2

Unconditional Generative Models as Unsupervised Learning Models

Unconditional Generative Model

Generative Model Learning

- **Observation:**

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

Unconditional Generative Model

Generative Model Learning

- **Observation:**

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

- **Model:** $p_\theta(\boldsymbol{x})$

Unconditional Generative Model

Generative Model Learning

- **Observation:**

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

- **Model:** $p_\theta(\boldsymbol{x})$
- **Tasks:**
  - Density estimation: $p_\theta(\boldsymbol{x}_{\text{new}})$

Unconditional Generative Model

Generative Model Learning

- **Observation:**

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

- **Model:** $p_\theta(\boldsymbol{x})$
- **Tasks:**
  - Density estimation: $p_\theta(\boldsymbol{x}_{\text{new}})$
  - Generation: $\boldsymbol{x}_{\text{new}} \sim p_\theta(\boldsymbol{x})$

Unconditional Generative Model

Generative Model Learning

- **Observation:**

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

- **Model:** $p_\theta(\boldsymbol{x})$
- **Tasks:**
  - Density estimation: $p_\theta(\boldsymbol{x}_{\text{new}})$
  - Generation: $\boldsymbol{x}_{\text{new}} \sim p_\theta(\boldsymbol{x})$
  - High level representation $\boldsymbol{z}$ (unsupervised representation learning):
    - Assume, $\boldsymbol{x}$ is IRIS flower, then ideally $z \in \{1, 2, 3\}$ corresponding to each type.

Unconditional Generative Model

Generative Model Learning

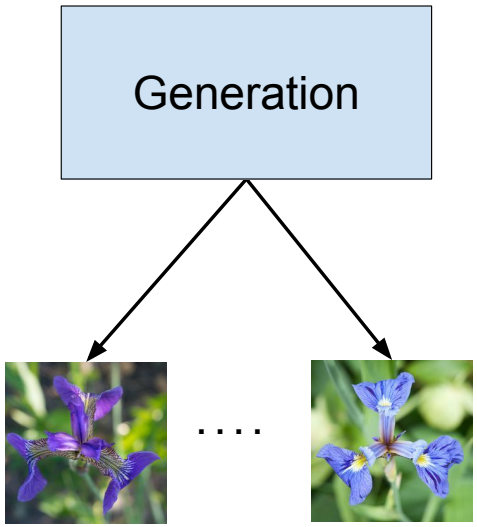
- **Observation:**

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

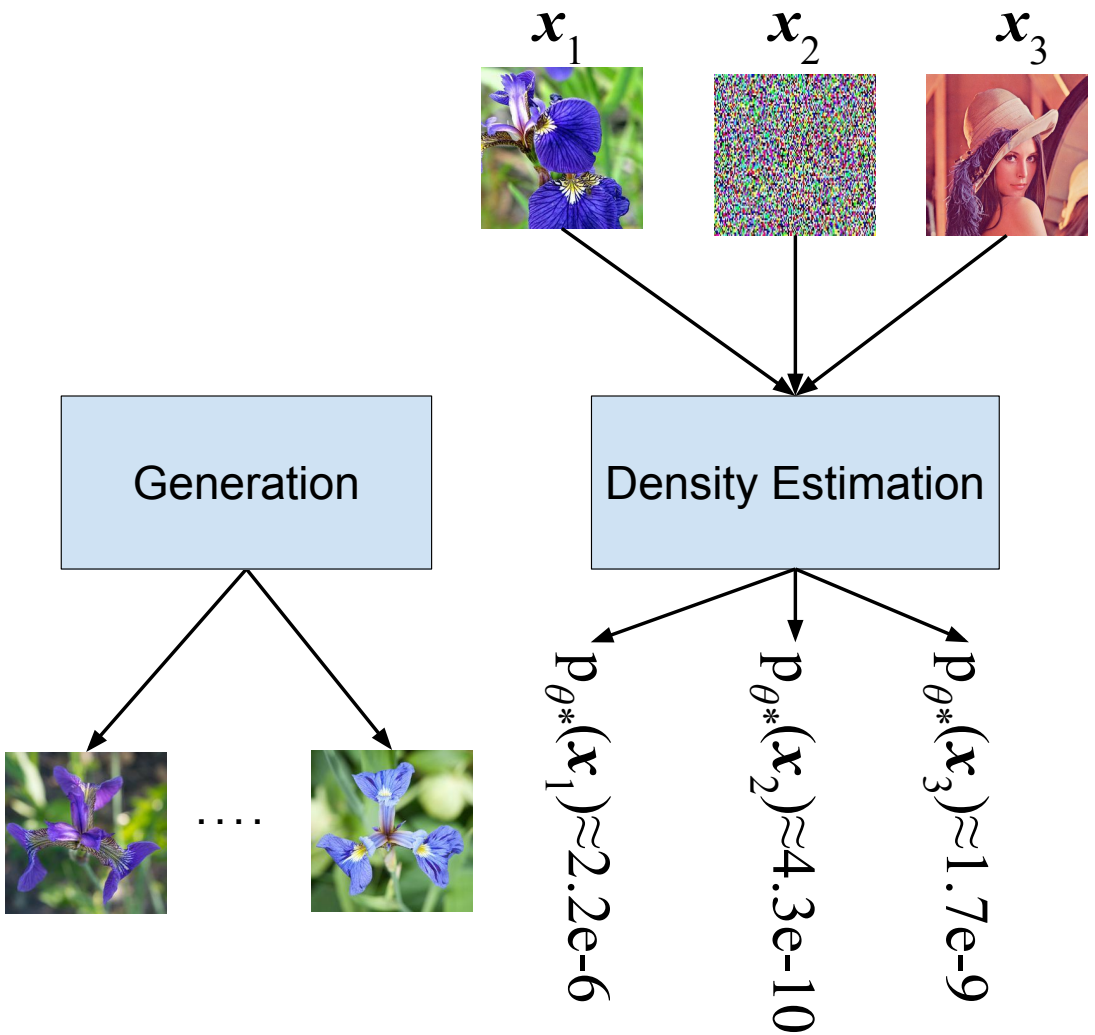
- **Model:** $p_\theta(\boldsymbol{x})$
- **Tasks:**
  - Density estimation: $p_\theta(\boldsymbol{x}_{\text{new}})$
  - Generation: $\boldsymbol{x}_{\text{new}} \sim p_\theta(\boldsymbol{x})$
  - High level representation $\boldsymbol{z}$ (unsupervised representation learning):
    - Assume, $\boldsymbol{x}$ is IRIS flower, then ideally $z \in \{1, 2, 3\}$ corresponding to each type.
- **Objective:**
  - Estimating distribution $p_{\text{data}}(\boldsymbol{x})$

$$\boldsymbol{\theta}^\star = \underset{\boldsymbol{\theta}}{\text{argmin}}\, \text{dist}(p_{\text{data}}, p_\theta)$$

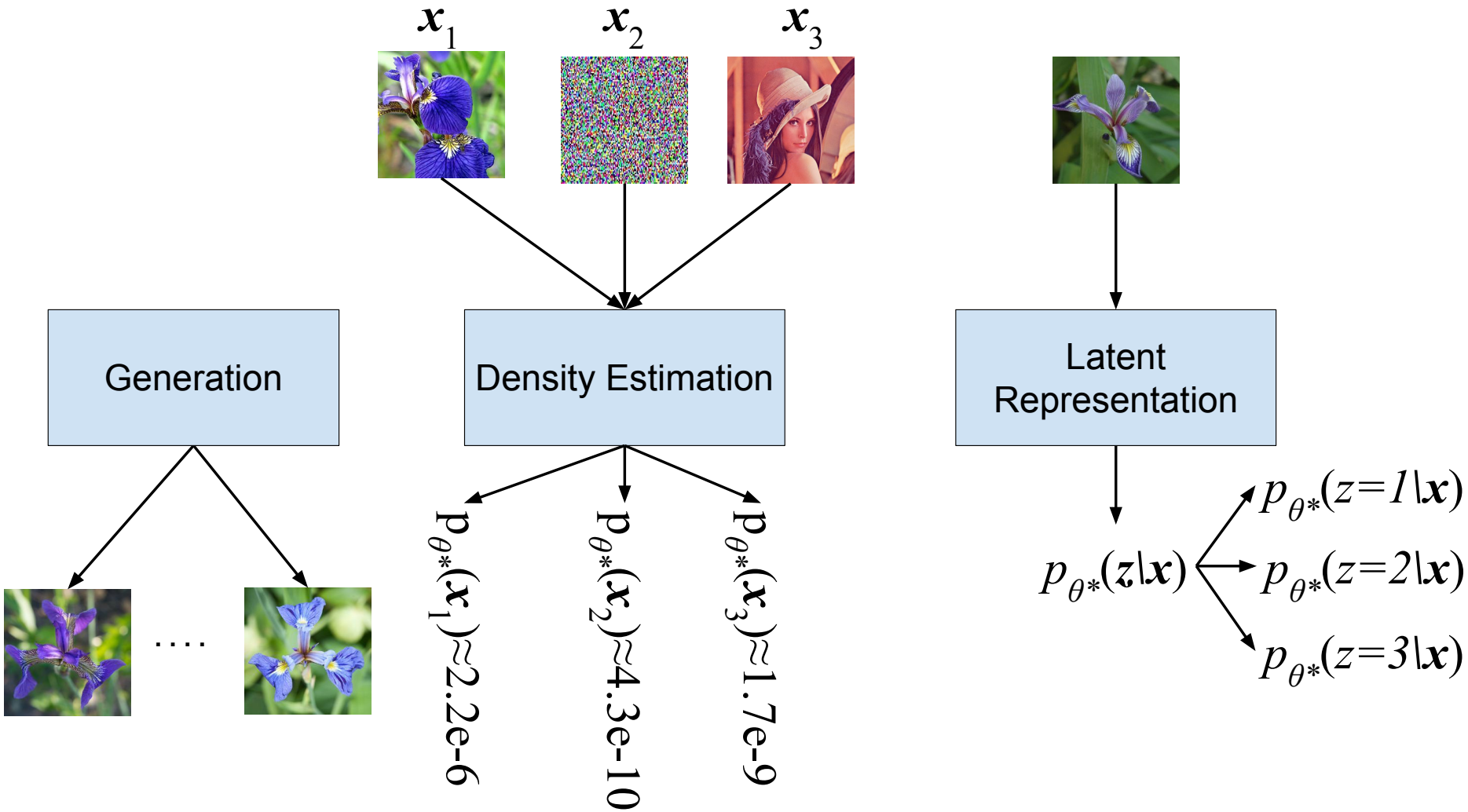
Unconditional Generative Model



Unconditional Generative Model



Unconditional Generative Model



Section 3

Conditional Generative Models as Extention of Supervised Learning Models

Conditional Generative Model

Conditional Generative Model Learning

- **Observation:**

$$\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$$

Conditional Generative Model

Conditional Generative Model Learning

- **Observation:**

$$\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$$

- **Model:** $p_\theta(\boldsymbol{y}|\boldsymbol{x})$

Conditional Generative Model

Conditional Generative Model Learning

- **Observation:**

$$\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$$

- **Model:** $p_\theta(\boldsymbol{y}|\boldsymbol{x})$
- **Tasks:**
  - Conditional density estimation: $p_\theta(\boldsymbol{y}|\boldsymbol{x}_{\text{new}})$

Conditional Generative Model

Conditional Generative Model Learning

- **Observation:**

$$\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$$

- **Model:** $p_\theta(\boldsymbol{y}|\boldsymbol{x})$
- **Tasks:**
  - Conditional density estimation: $p_\theta(\boldsymbol{y}|\boldsymbol{x}_{\text{new}})$
  - Generation: $\boldsymbol{y}_{\text{new}} \sim p_\theta(\boldsymbol{y}|\boldsymbol{x}_{\text{new}})$

Conditional Generative Model

Conditional Generative Model Learning

- **Observation:**

$$\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$$

- **Model:** $p_\theta(\boldsymbol{y}|\boldsymbol{x})$
- **Tasks:**
  - Conditional density estimation: $p_\theta(\boldsymbol{y}|\boldsymbol{x}_{\text{new}})$
  - Generation: $\boldsymbol{y}_{\text{new}} \sim p_\theta(\boldsymbol{y}|\boldsymbol{x}_{\text{new}})$
  - High level representation $\boldsymbol{z}$ (unsupervised representation learning)

Conditional Generative Model

Conditional Generative Model Learning

- **Observation:**

$$\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$$

- **Model:** $p_\theta(\boldsymbol{y}|\boldsymbol{x})$
- **Tasks:**
  - Conditional density estimation: $p_\theta(\boldsymbol{y}|\boldsymbol{x}_{\text{new}})$
  - Generation: $\boldsymbol{y}_{\text{new}} \sim p_\theta(\boldsymbol{y}|\boldsymbol{x}_{\text{new}})$
  - High level representation $\boldsymbol{z}$ (unsupervised representation learning)
- **Objective:**
  - Estimating distribution $p_{\text{data}}(\boldsymbol{y}|\boldsymbol{x})$

$$\boldsymbol{\theta}^\star = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \text{dist}(p_{\text{data}}, p_\theta)$$

Conditional Generative Model - Density Estimation

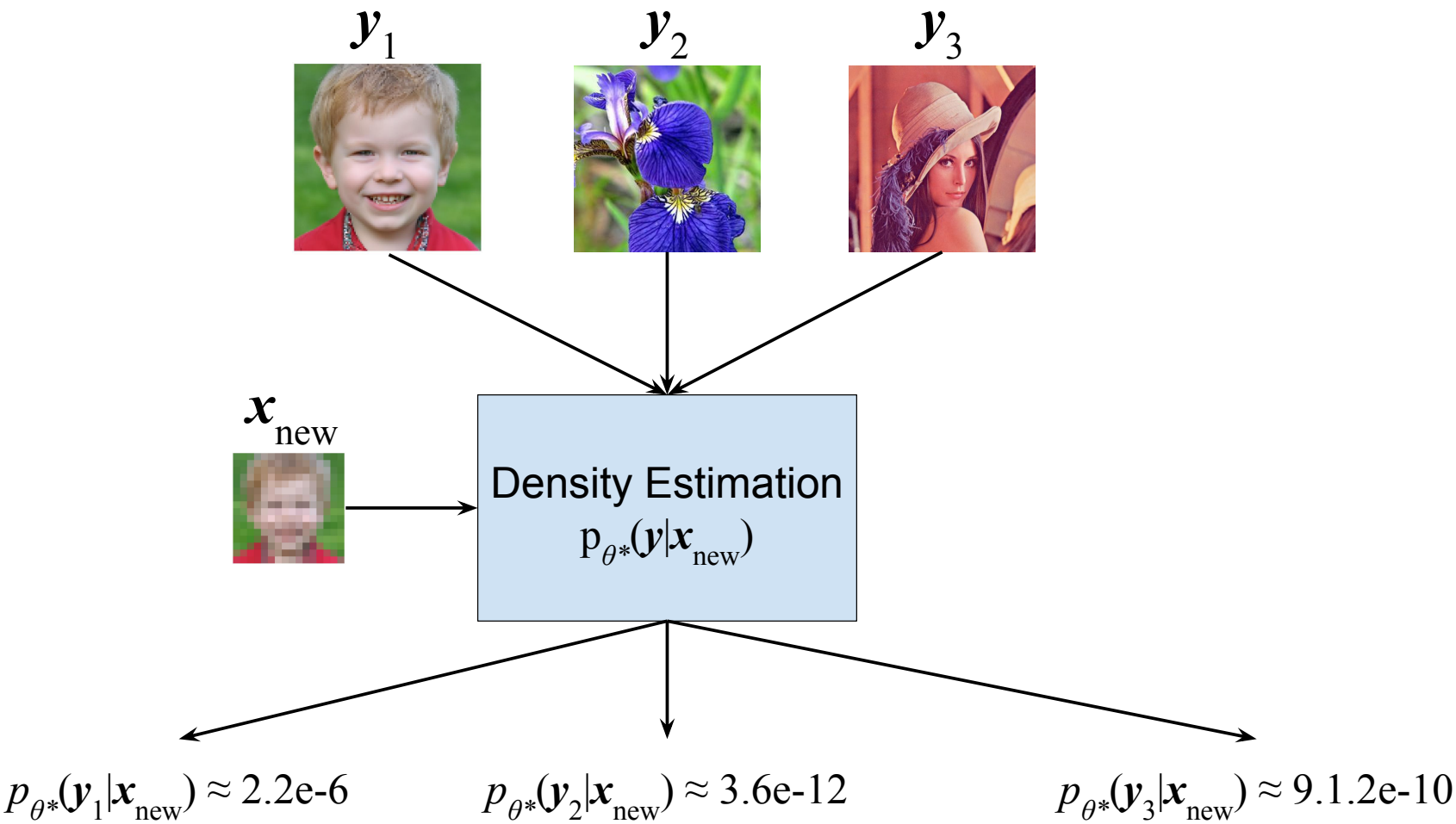


Figure: Assume we trained a conditional generative model and estimate $\boldsymbol{\theta}^\star$. Then we can use it for density estimation of different $\boldsymbol{y}$ vector given $\boldsymbol{x}_{\text{new}}$ (image source: [1])

Conditional Generative Model - Generation

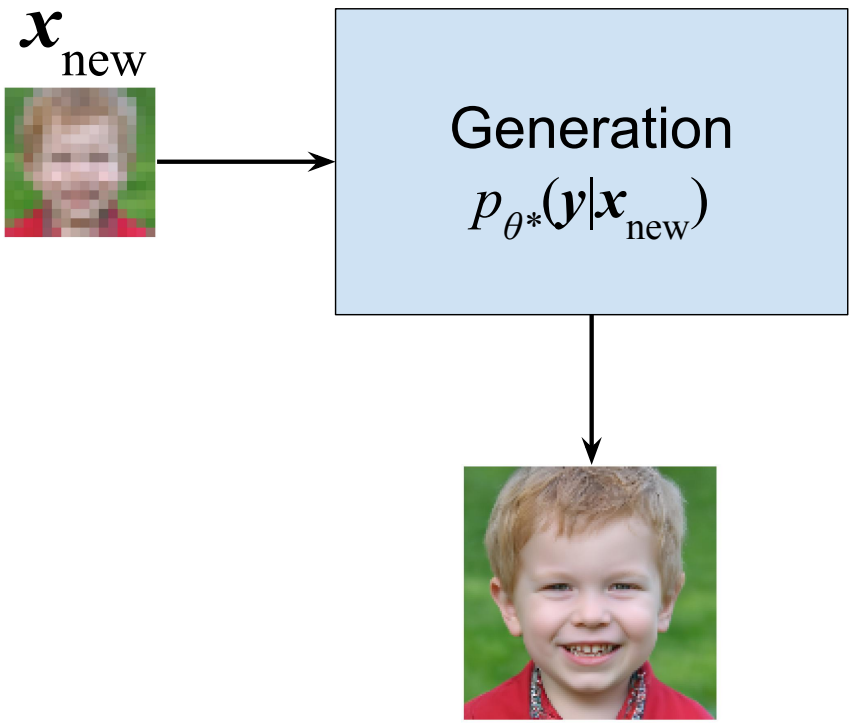


Figure: Assume we trained a conditional generative model and estimate $\boldsymbol{\theta}^\star$. Then we can use it for sampling $\boldsymbol{y}$ given $\boldsymbol{x}_{\text{new}}$ (image source: [1])

Conditional Generative Model - Generation

MLR for Conditional Generative Model

Assume a simple super-resolution problem:

- $\boldsymbol{x} \in \{0,1\}^{r \times c}$
- $\boldsymbol{y} \in \{0,1\}^{h \times w}$

Conditional Generative Model - Generation

MLR for Conditional Generative Model

Assume a simple super-resolution problem:

- $\boldsymbol{x} \in \{0,1\}^{r \times c}$
- $\boldsymbol{y} \in \{0,1\}^{h \times w}$

Then MLR must assign a probability to each possible output which is:

$$\text{Number of possible } \boldsymbol{y} \text{ vectors } \Rightarrow 2^{h \times w}$$

$$\text{Number of MLR parameters} \Rightarrow r \times c \times 2^{h \times w}$$

Conditional Generative Model - Generation

MLR for Conditional Generative Model

Assume a simple super-resolution problem:

- $\boldsymbol{x} \in \{0,1\}^{r \times c}$
- $\boldsymbol{y} \in \{0,1\}^{h \times w}$

Then MLR must assign a probability to each possible output which is:

$$\text{Number of possible } \boldsymbol{y} \text{ vectors } \Rightarrow 2^{h \times w}$$

$$\text{Number of MLR parameters} \Rightarrow r \times c \times 2^{h \times w}$$

So using MLR and more complex models such as supervised deep models is not practical.

Conditional vs Unconditional Deep Generative Models

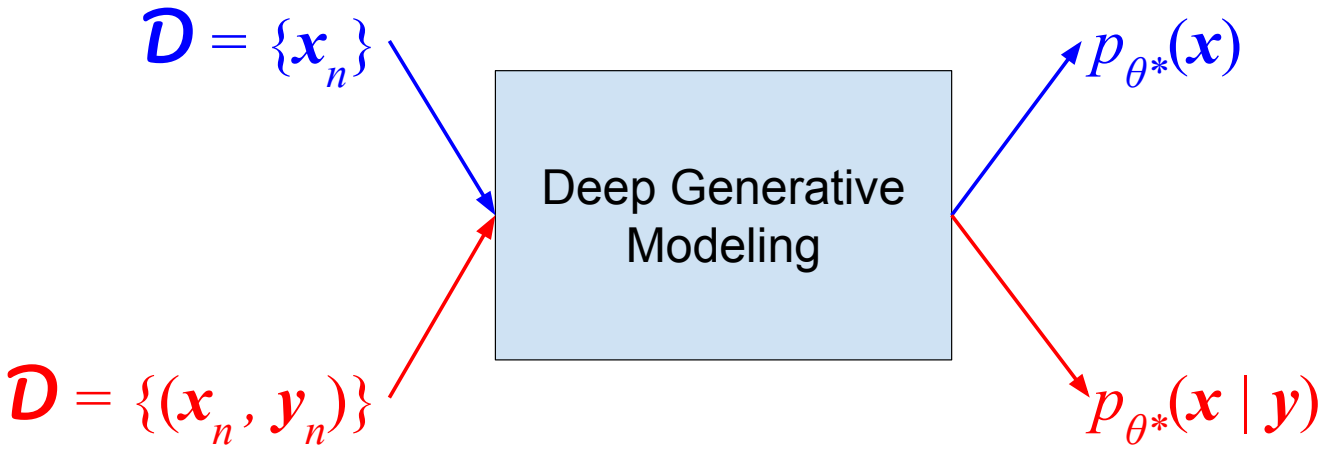


Figure: The methods to train conditional and unconditional generative modeling are almost shared in different approaches

Conditional vs Unconditional Deep Generative Models

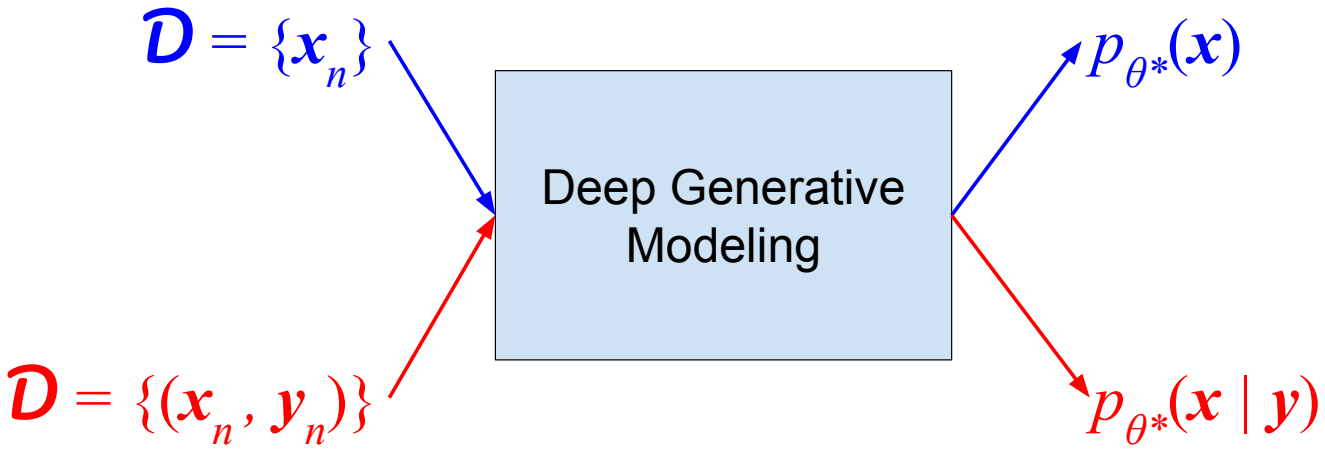


Figure: The methods to train conditional and unconditional generative modeling are almost shared in different approaches

Notation Update

To better share the notation across conditional and unconditional DGM, we use $p_\theta(\boldsymbol{x}|\boldsymbol{y})$ instead of $p_\theta(\boldsymbol{y}|\boldsymbol{x})$ for the conditional case. So we are always working on a probabilistic model over $\boldsymbol{x}$.

Text-to-Speech Models

Text-to-Speech Models

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

Text-to-Speech Models

Text-to-Speech Models

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

Real-World Sample

Listen to the following speech synthesis (source: [2])

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

Text-to-Image Models

Text-to-Image Models

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

Text-to-Image Models

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

Figure: $\boldsymbol{x}$ for $\boldsymbol{y}$ = "Teddy bears swimming at the Olympics 400m Butterfly event." (source: [3])

Image-to-Image Translation

Image Colorization

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

Image-to-Image Translation

Image Colorization

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

(a) $\boldsymbol{y}$

(b) $\boldsymbol{x}$

(c) Ground truth

Figure: Image colorization (source: [4])

Image-to-Image Translation

Image Inpainting

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

Image-to-Image Translation

Image Inpainting

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

(a) $\boldsymbol{y}$

(b) $\boldsymbol{x}$

(c) Ground truth

Figure: Image inpainting (source: [4])

Image-to-Image Translation

Image Uncropping

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

Image-to-Image Translation

Image Uncropping

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

(a) $\boldsymbol{y}$

(b) $\boldsymbol{x}$

(c) Ground truth

Figure: Image uncropping (source: [4])

Image-to-Image Translation

Image Restoration

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

Image-to-Image Translation

Image Restoration

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

(a) $\boldsymbol{y}$

(b) $\boldsymbol{x}$

(c) Ground truth

Figure: Image restoration (source: [4])

Section 4

Discriminative Vs. Generative Classification

Discriminative Approach for Classification

Multinomial Logistic Regression

- Model

$$p_\theta(y|\boldsymbol{x}) = \text{Cat}(y|\mathcal{S}(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}))$$

Discriminative Approach for Classification

Multinomial Logistic Regression

- Model

$$p_\theta(y|\boldsymbol{x}) = \text{Cat}(y|\mathcal{S}(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}))$$

where:

$y \in \{1, 2, 3, \ldots, C\}$	Label
$\boldsymbol{x} \in \mathbb{R}^D$	Input pattern
$\mathcal{S}(\cdot)$	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

- Number of parameters:

$$C \times D + C$$

Discriminative Approach for Classification

Multinomial Logistic Regression

Assume:

$$\boldsymbol{x} = \text{vec}\left(\text{  } \in \mathbb{R}^{600 \times 800} \right)$$

$$y \in \{1 \text{ (LivingRoom)}, 2 \text{ (BedRoom)}, 3 \text{ (DiningRoom)}\}$$

Discriminative Approach for Classification

Multinomial Logistic Regression

Assume:

$$\boldsymbol{x} = \text{vec}\left(\text{  } \in \mathbb{R}^{600 \times 800} \right)$$

$$y \in \{1 \text{ (LivingRoom)}, 2 \text{ (BedRoom)}, 3 \text{ (DiningRoom)}\}$$

then:

$$p_\theta\left(y \middle| \text{  } \right) = \text{Cat}(y|[0.1, 0.7, 0.2]^T) \xrightarrow{\text{Sampling}} \begin{cases} 2 \\ 2 \\ 2 \\ 3 \\ 2 \end{cases}$$

Generative Approach for Classification

Discriminant Analysis

- Model

$$p_\theta(y = c|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x}|y = c)p_\theta(y = c)}{\sum_{c'} p_\theta(\boldsymbol{x}|y = c')p_\theta(y = c')}$$

Generative Approach for Classification

Discriminant Analysis

- Model

$$p_\theta(y = c|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x}|y = c)p_\theta(y = c)}{\sum_{c'} p_\theta(\boldsymbol{x}|y = c')p_\theta(y = c')}$$

where:

$y \in \{1, 2, 3, \ldots, C\}$	Label
$\boldsymbol{x} \in \mathbb{R}^D$	Input pattern
$p_\theta(y = c)$	Prior probability over labels
$p_\theta(\boldsymbol{x}|y = c)$	Class conditional density
$\boldsymbol{\theta}$	Model parameters

Generative Approach for Classification

Discriminant Analysis

- Model

$$p_\theta(y = c|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x}|y = c)p_\theta(y = c)}{\sum_{c'} p_\theta(\boldsymbol{x}|y = c')p_\theta(y = c')}$$

where:

$y \in \{1, 2, 3, \ldots, C\}$	Label
$\boldsymbol{x} \in \mathbb{R}^D$	Input pattern
$p_\theta(y = c)$	Prior probability over labels
$p_\theta(\boldsymbol{x}|y = c)$	Class conditional density
$\boldsymbol{\theta}$	Model parameters

- Number of Parameters (Assuming $p_\theta(\boldsymbol{x}|y = c)$ to be Multivariate Normal):

$$(C - 1) + \overbrace{C \times D}^{\text{Mean Vector}} + \overbrace{C \times \frac{D(D + 1)}{2}}^{\text{Covariance matrix}}$$

Generative Approach for Classification

Discriminant Analysis

Assume:

$$\boldsymbol{x} = \text{vec}\left(\text{  } \in \mathbb{R}^{600 \times 800} \right)$$

$$y \in \{1 \text{ (LivingRoom)}, 2 \text{ (BedRoom)}, 3 \text{ (DiningRoom)}\}$$

then:

$$p_\theta(\boldsymbol{x}|y=2) \xrightarrow{\text{Sampling}}$$




Comparing Discriminative and Generative Approaches

Training

- Number of parameters (Assuming $p_\theta(\boldsymbol{x}|y=c)$ to be Multivariate Normal):

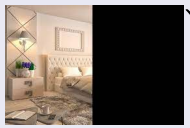
$$\overbrace{C \times D + C}^{\text{Discriminative}} \ll \overbrace{(C-1) + C \times D + C \times \frac{D(D+1)}{2}}^{\text{Generative}}$$

- Training a Discriminative model is much simpler

Comparing Discriminative and Generative Approaches

Challenging Situations

- Discriminative Model:

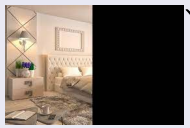
$$p_\theta\left(y \middle| \text{  }\right) = \text{Cat}(y|[0.7, 0.1, 0.2]^T)$$

Comparing Discriminative and Generative Approaches

Challenging Situations

- Discriminative Model:

$$p_\theta\left(y \middle| \right) = \text{Cat}(y|[0.7, 0.1, 0.2]^T)$$



- Generative Model: Assume $\boldsymbol{x}^v$ and $\boldsymbol{x}^m$ to be the visible and masked parts, respectively, we can calculate $p(y|\boldsymbol{x}^v)$ and thus:

$$\max_y p(y|\boldsymbol{x}^v)$$

and decide based on the visible part which is much more *robust*.

Section 5

Big Picture

Course Big Picture

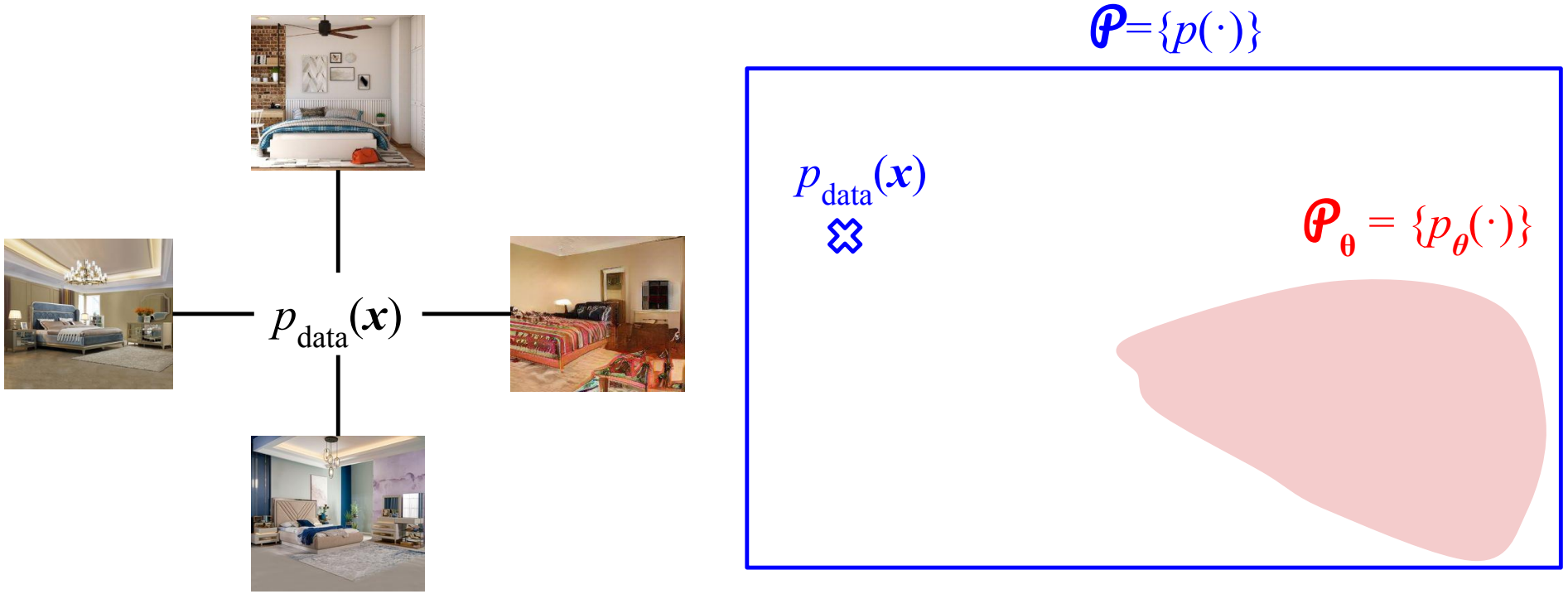
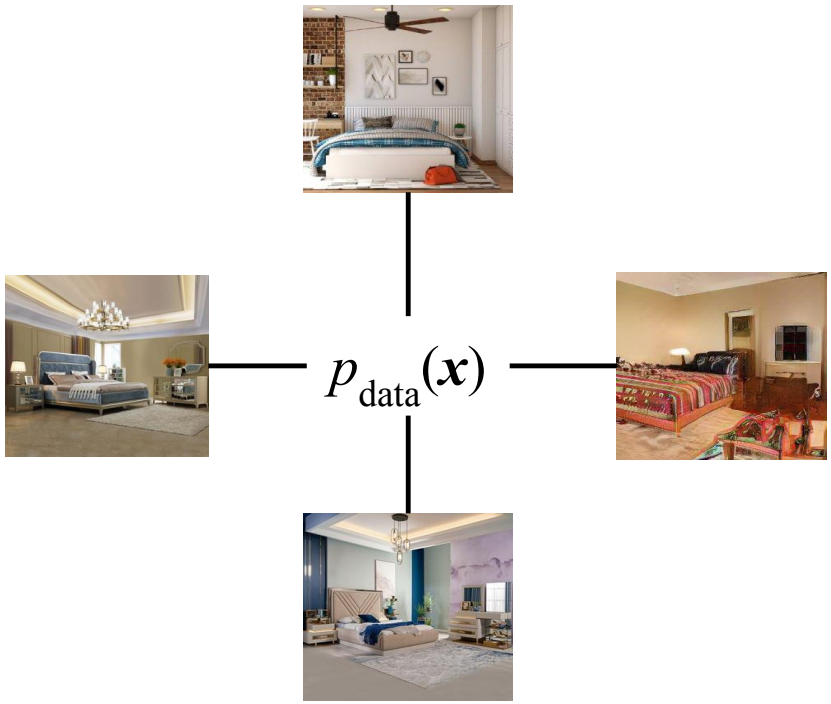


Figure: Selecting a family [5]

Course Big Picture



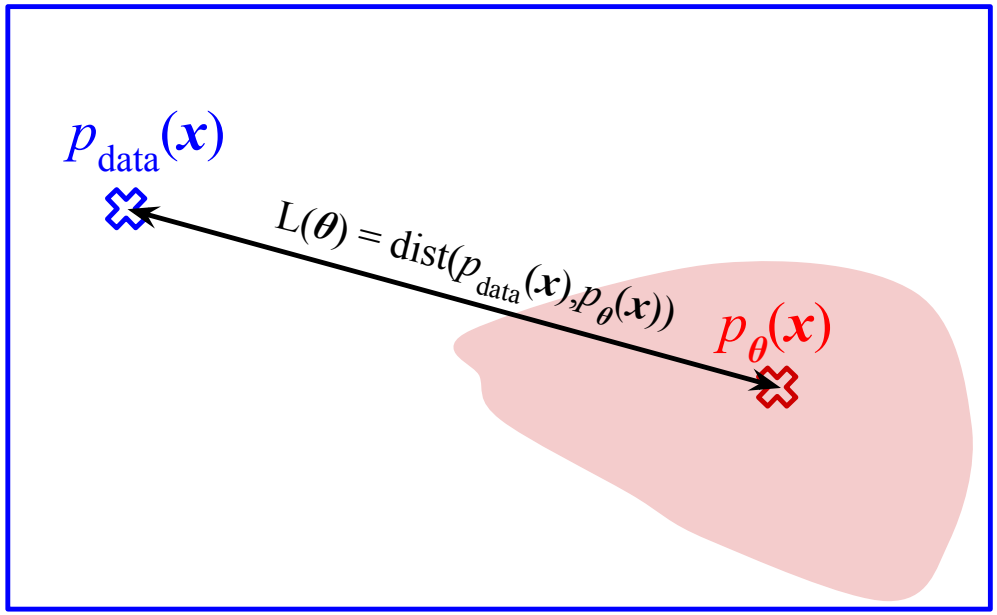
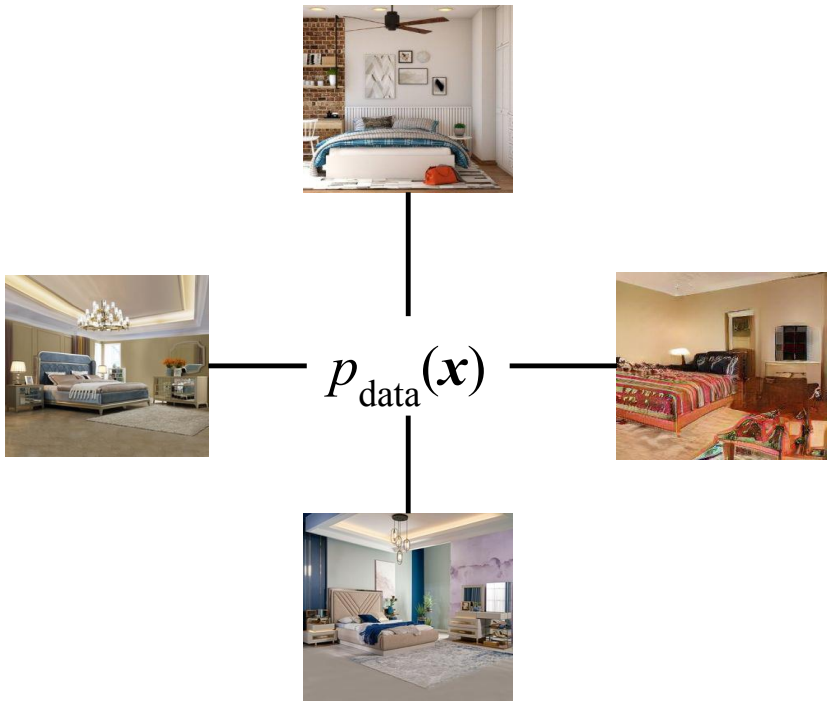


Figure: Developing a distance metric [5]

Course Big Picture



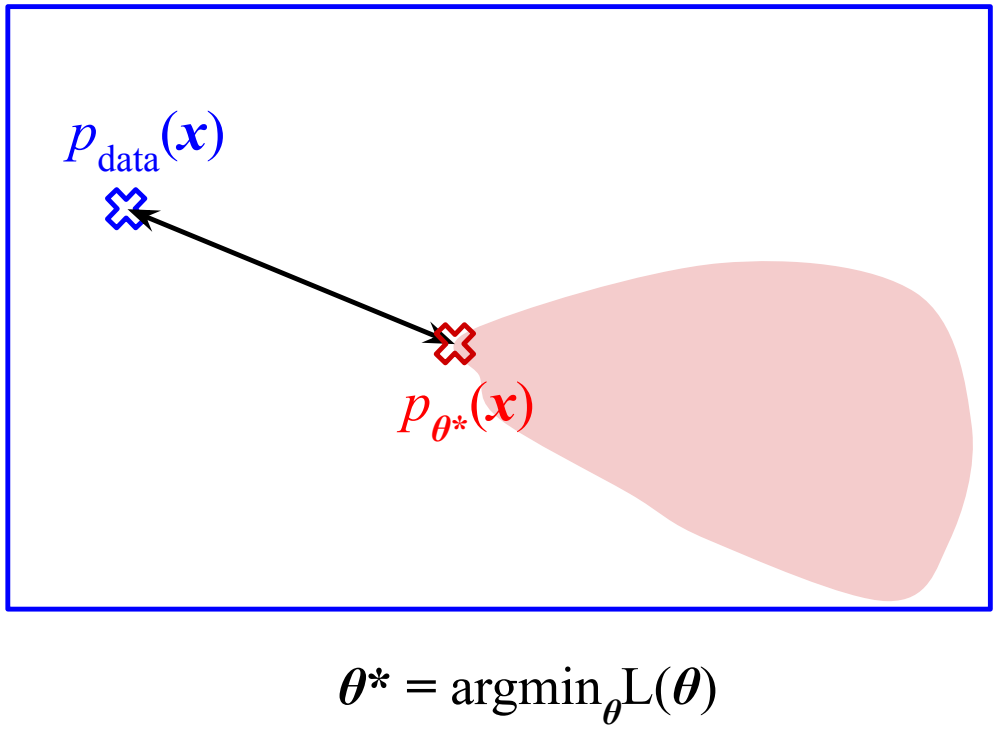
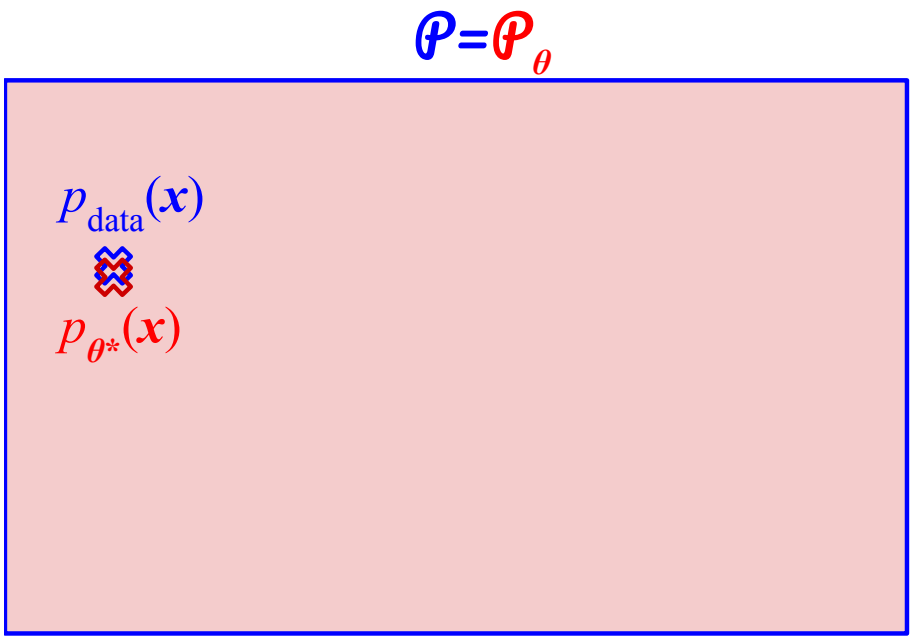
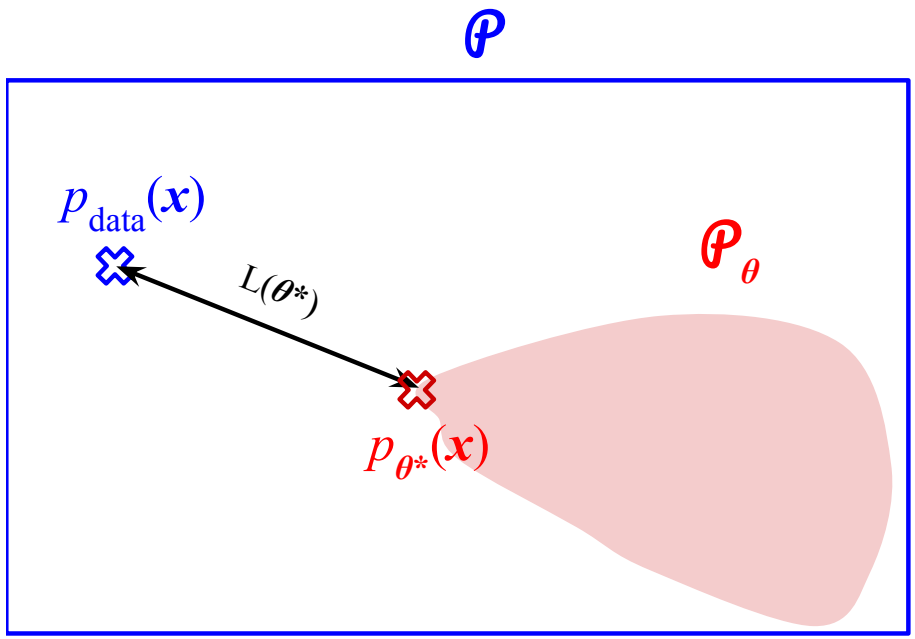


Figure: Minimizing the distance metric [5]

Limited Vs. Unlimited Selection



(a) Unlimited distribution selection



(b) Limited distribution selection

Generative Model for Simple Random Variables

Bernoulli Random Variable (Tossing a coin)

Assume we have a Bernoulli random variable:

$$X \sim \text{Ber}(\beta) \Rightarrow \begin{cases} p(X=1) = \beta \\ p(X=0) = 1-\beta \end{cases}$$

Generative Model for Simple Random Variables

Bernoulli Random Variable (Tossing a coin)

Assume we have a Bernoulli random variable:

$$X \sim \text{Ber}(\beta) \Rightarrow \begin{cases} p(X=1) = \beta \\ p(X=0) = 1-\beta \end{cases}$$

Then:

- The description for all possible distributions is:

$$\mathcal{P} = \{\text{Ber}(\alpha);\ 0 \leq \alpha \leq 1\} \Rightarrow \#\text{Parameters} = 1$$

Generative Model for Simple Random Variables

Bernoulli Random Variable (Tossing a coin)

Assume we have a Bernoulli random variable:

$$X \sim \text{Ber}(\beta) \Rightarrow \begin{cases} p(X=1) = \beta \\ p(X=0) = 1-\beta \end{cases}$$

Then:

- The description for all possible distributions is:

$$\mathcal{P} = \{\text{Ber}(\alpha);\ 0 \leq \alpha \leq 1\} \Rightarrow \#\text{Parameters} = 1$$

- The description for a subset of the whole family is:

$$\mathcal{P}_\theta = \{\text{Ber}(\theta);\ 0.2 \leq \theta \leq 0.3\} \Rightarrow \#\text{Parameters} = 1$$

Generation and Density Estimation from a Trained Model

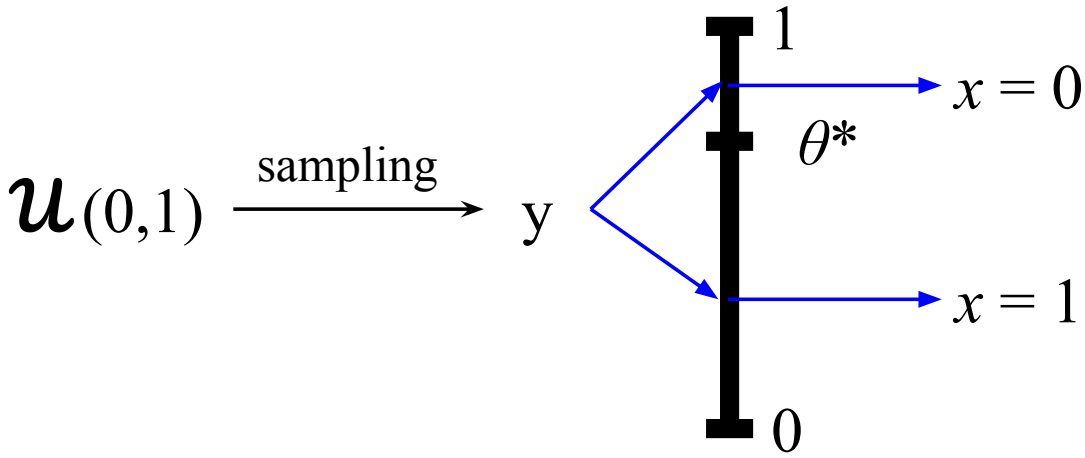


Figure: Sample generation using a trained model for Bernoulli distribution

Generation and Density Estimation from a Trained Model

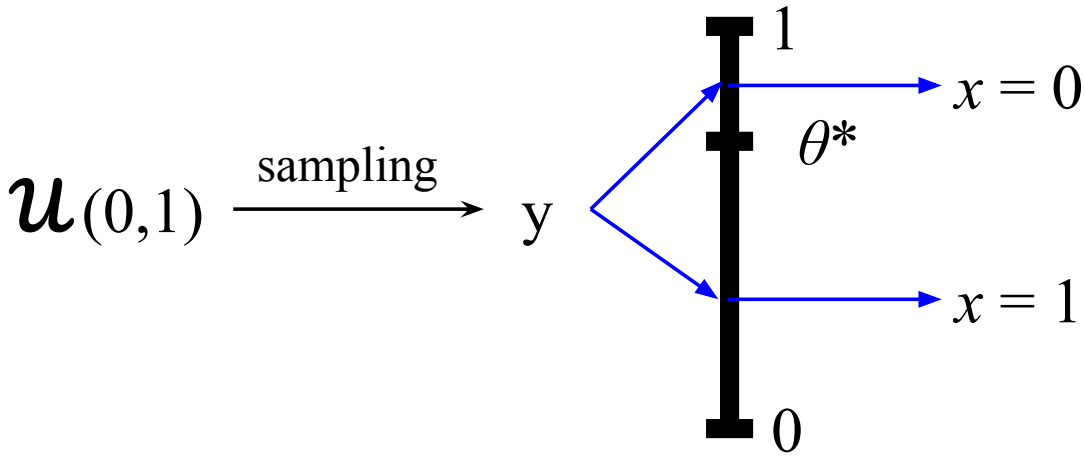


Figure: Sample generation using a trained model for Bernoulli distribution

Density Estimation

You have two states in your probabilistic world $X \in \{0, 1\}$, thus:

$$\begin{cases} p(X = 1) = \theta^\star \\ p(X = 0) = 1 - \theta^\star \end{cases}$$

Generative Model for Simple Random Variables

Categorical Random Variable (Rolling a die)

Assume we have a Categorical random variable with six states:

$$X \sim \text{Cat}([\beta_1, \ldots, \beta_6]) \Rightarrow p(X = i) = \beta_i$$

Generative Model for Simple Random Variables

Categorical Random Variable (Rolling a die)

Assume we have a Categorical random variable with six states:

$$X \sim \text{Cat}([\beta_1, \ldots, \beta_6]) \Rightarrow p(X = i) = \beta_i$$

Then:

- The description for all possible distributions is:

$$\mathcal{P} = \{\text{Cat}(\boldsymbol{\alpha}); 0 \leq \alpha_i \leq 1, \sum_i \alpha_i = 1\} \Rightarrow \#\text{Parameters} = 6 - 1$$

Generative Model for Simple Random Variables

Categorical Random Variable (Rolling a die)

Assume we have a Categorical random variable with six states:

$$X \sim \text{Cat}([\beta_1, \ldots, \beta_6]) \Rightarrow p(X = i) = \beta_i$$

Then:

- The description for all possible distributions is:

$$\mathcal{P} = \{\text{Cat}(\boldsymbol{\alpha}); 0 \leq \alpha_i \leq 1, \sum_i \alpha_i = 1\} \Rightarrow \#\text{Parameters} = 6 - 1$$

- The description for a subset of the whole family is:

$$\mathcal{P}_{\boldsymbol{\theta}} = \left\{ \text{Cat}(\boldsymbol{\theta}); 0 \leq \theta_i \leq 1, \sum_i \theta_i = 1, \begin{cases} \theta_1 = \theta_2 = \theta_3 \\ \theta_4 = \theta_5 = \theta_6 \end{cases} \right\}$$

$$\Rightarrow \#\text{Parameters} = 2 - 1$$

Generation and Density Estimation from a Trained Model

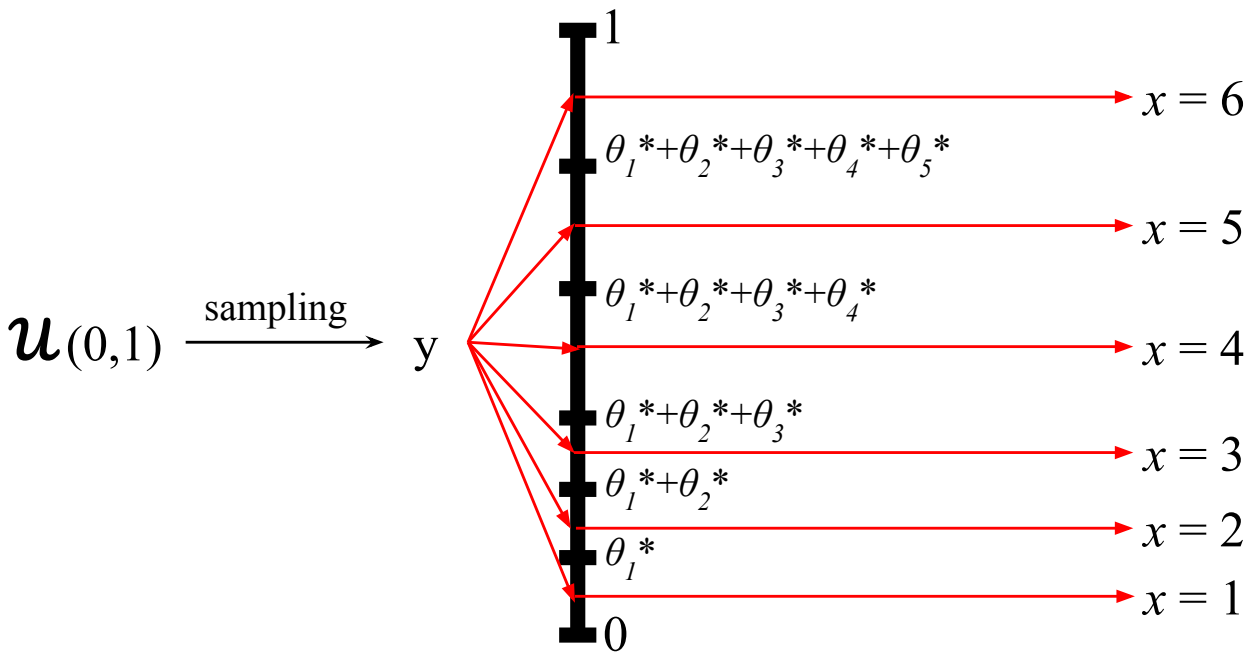


Figure: Sample generation using a trained model for Bernoulli distribution

Generation and Density Estimation from a Trained Model

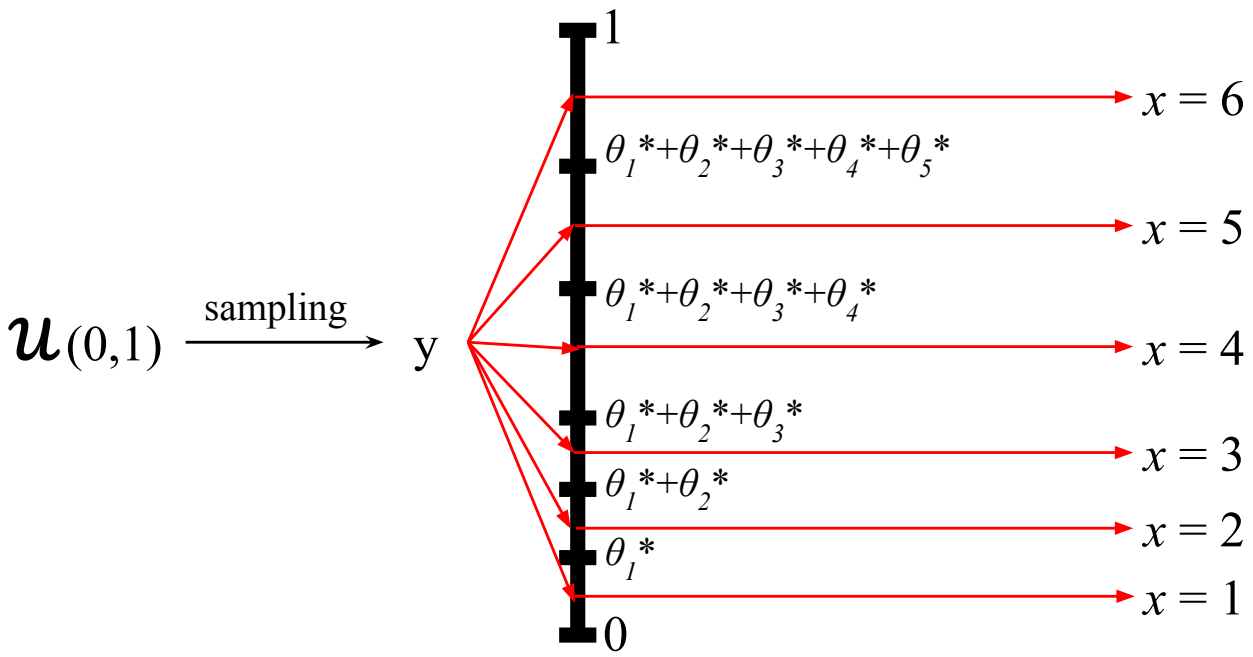


Figure: Sample generation using a trained model for Bernoulli distribution

Density Estimation

You have six states in your probabilistic world $X \in \{1, 2, 3, 4, 5, 6\}$, thus:

$$p(X = i) = \theta_i^\star$$

Real-World Categorical Distribution

RGB pixel

Assume $(r, g, b) \sim p(R, G, B)$, $\begin{cases} \text{SampleSpace}(R) = \{0, \dots, 255\} \\ \text{SampleSpace}(G) = \{0, \dots, 255\} \\ \text{SampleSpace}(B) = \{0, \dots, 255\} \end{cases}$ to be an outcome of distributions over RGB pixel, then:

- Number of Categories: 256^3

Real-World Categorical Distribution

RGB pixel

Assume $(r, g, b) \sim p(R, G, B)$, $\begin{cases} \text{SampleSpace}(R) = \{0, \dots, 255\} \\ \text{SampleSpace}(G) = \{0, \dots, 255\} \\ \text{SampleSpace}(B) = \{0, \dots, 255\} \end{cases}$ to be an outcome of distributions over RGB pixel, then:

- Number of Categories: 256^3
- #Parameters for $\mathcal{P}$: $256^3 - 1$

Real-World Binary Image Distribution

Binary image with n pixels

Assume $\boldsymbol{x} \sim p(\mathbb{X})$ to be a binary image of digits where $\mathbb{X} = [X_1, \ldots, X_n]$, Then:

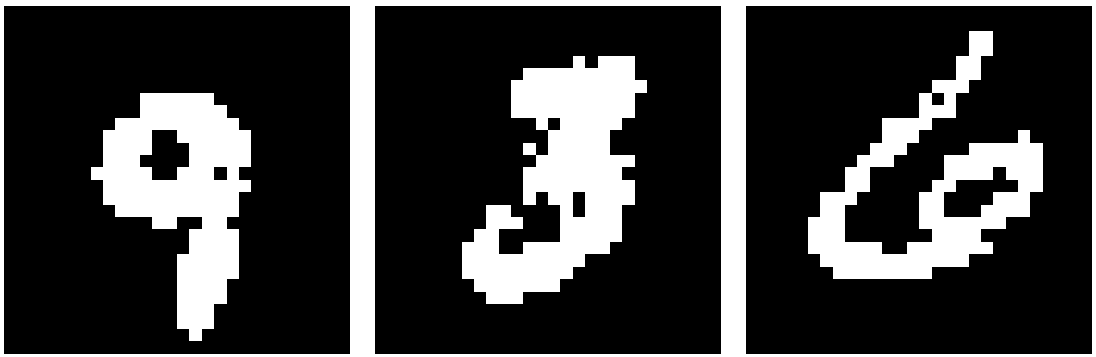
- Number of Categories: 2^n
- #Parameters for $\mathcal{P}$: $2^n - 1$

Real-World Binary Image Distribution

Binary image with n pixels

Assume $\boldsymbol{x} \sim p(\mathbb{X})$ to be a binary image of digits where $\mathbb{X} = [X_1, \ldots, X_n]$, Then:

- Number of Categories: 2^n
- #Parameters for $\mathcal{P}$: $2^n - 1$



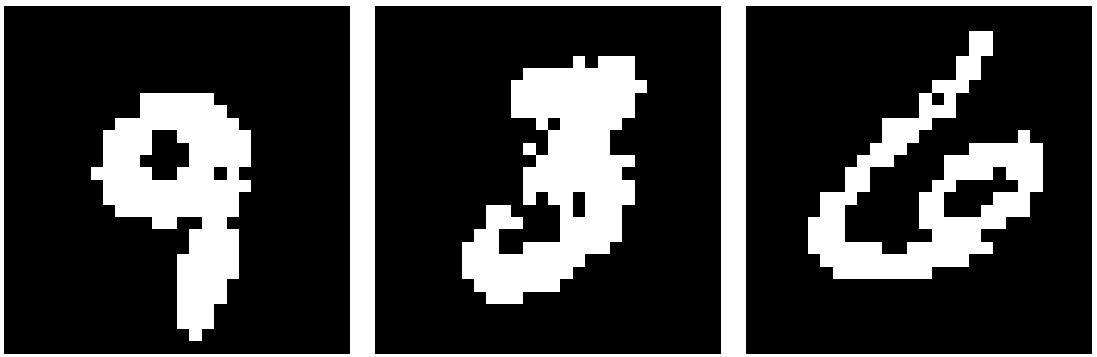
(a) Generated samples from $p_{\theta^\star}$ assuming $\mathcal{P} = \mathcal{P}_\theta$ we can find $p_{\theta^\star}$ efficiently (source: [5])

Real-World Binary Image Distribution

Binary image with n pixels

Assume $\boldsymbol{x} \sim p(\mathbb{X})$ to be a binary image of digits where $\mathbb{X} = [X_1, \ldots, X_n]$, Then:

- Number of Categories: 2^n
- #Parameters for $\mathcal{P}$: $2^n - 1$



(a) Generated samples from $p_{\theta^\star}$ assuming $\mathcal{P} = \mathcal{P}_\theta$ we can find $p_{\theta^\star}$ efficiently (source: [5])

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

Independence as Simplifying Tool

Binary image with n independent pixels

Assume $\boldsymbol{x} \sim p(\mathbb{X})$ to be a binary image of digits where the pixels are independent random variables, in other words, $p(\boldsymbol{x}) = p(x_1) \times p(x_2) \dots p(x_n)$, Then:

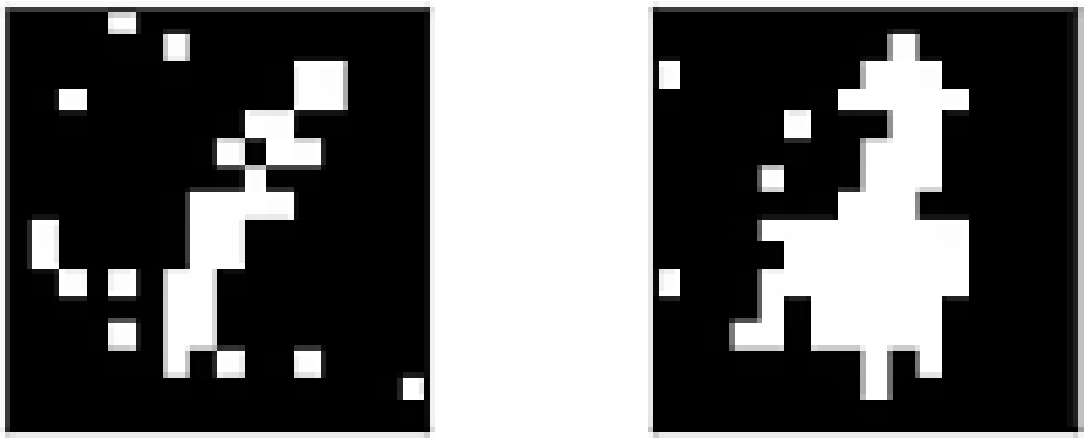
- Number of Categories: 2^n
- #Parameters for $\mathcal{P}$: $2^n - 1$
- #Parameters for $\mathcal{P}_\theta$: n

Independence as Simplifying Tool

Binary image with n independent pixels

Assume $\boldsymbol{x} \sim p(\mathbb{X})$ to be a binary image of digits where the pixels are independent random variables, in other words, $p(\boldsymbol{x}) = p(x_1) \times p(x_2) \dots p(x_n)$, Then:

- Number of Categories: 2^n
- #Parameters for $\mathcal{P}$: $2^n - 1$
- #Parameters for $\mathcal{P}_\theta$: n



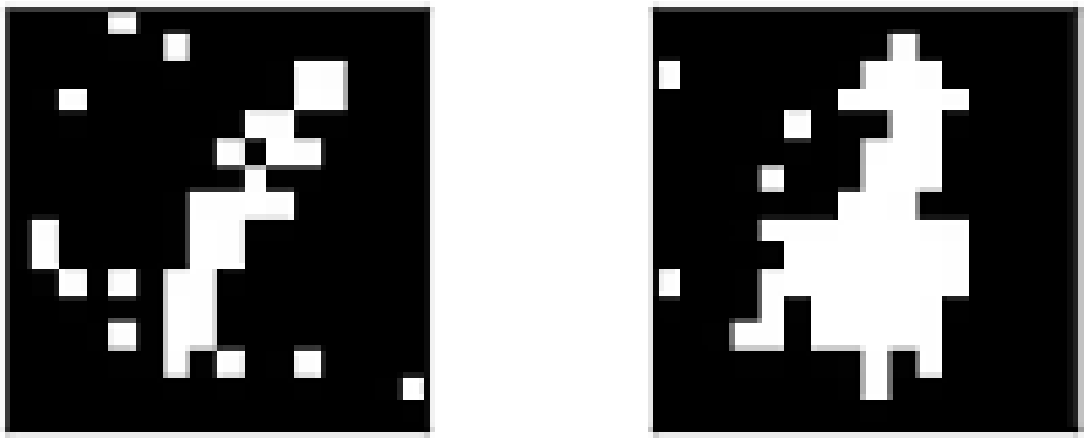
(a) Generated samples from $p_{\theta^\star}$ assuming we can find $p_{\theta^\star}$ efficiently (source: [5])

Independence as Simplifying Tool

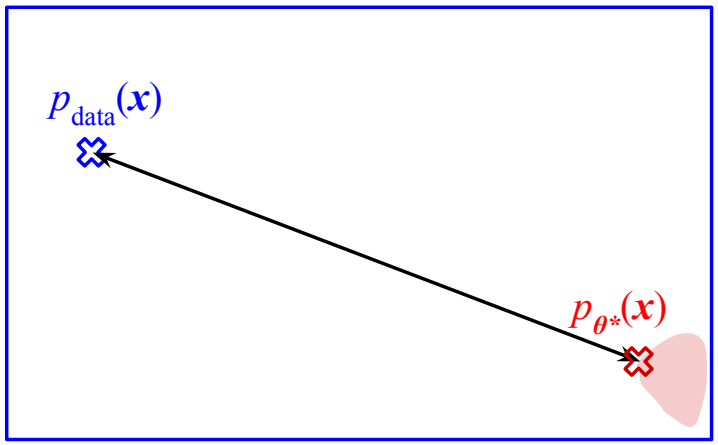
Binary image with n independent pixels

Assume $\boldsymbol{x} \sim p(\mathbb{X})$ to be a binary image of digits where the pixels are independent random variables, in other words, $p(\boldsymbol{x}) = p(x_1) \times p(x_2) \dots p(x_n)$, Then:

- Number of Categories: 2^n
- #Parameters for $\mathcal{P}$: $2^n - 1$
- #Parameters for $\mathcal{P}_\theta$: n



(a) Generated samples from $p_{\theta^\star}$ assuming we can find $p_{\theta^\star}$ efficiently (source: [5])



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

Chain Rule

Binary image with n pixels

Using chain rule, we can write:

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

Chain Rule

Binary image with n pixels

Using chain rule, we can write:

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

Then:

- Number of Categories: 2^n
- #Parameters for $\mathcal{P}$: $2^0 + 2^1 + 2^2 + \ldots + 2^{n-1} = 2^n - 1$

Chain Rule

Binary image with n pixels

Using chain rule, we can write:

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

Then:

- Number of Categories: 2^n
- #Parameters for $\mathcal{P}$: $2^0 + 2^1 + 2^2 + \ldots + 2^{n-1} = 2^n - 1$

Chair rule

Note that the chain rule does not impose any restriction on the distribution. Thus again we are parameterizing the $\mathcal{P}$ and thus the number of needed parameters is unchanged.

Chain Rule + Conditional Indepence

Binary Image with Conditional Independence Property

Now assum $X_{i+1} \perp X_{i-1}, \ldots, X_1 | X_i$, then we can simplify the distribution as:

$$\begin{aligned} p(\boldsymbol{x}) &= p(x_1)p(x_2|x_1)p(x_3|x_2,x_1)\ldots p(x_n|x_{n-1},\ldots,x_1) \\ &= \underbrace{p(x_1)}_{\#1}\underbrace{p(x_2|x_1)}_{\#2}\underbrace{p(x_3|x_2))}_{\#2}\ldots\underbrace{p(x_n|x_{n-1})}_{\#2} \end{aligned}$$

Chain Rule + Conditional Indepence

Binary Image with Conditional Independence Property

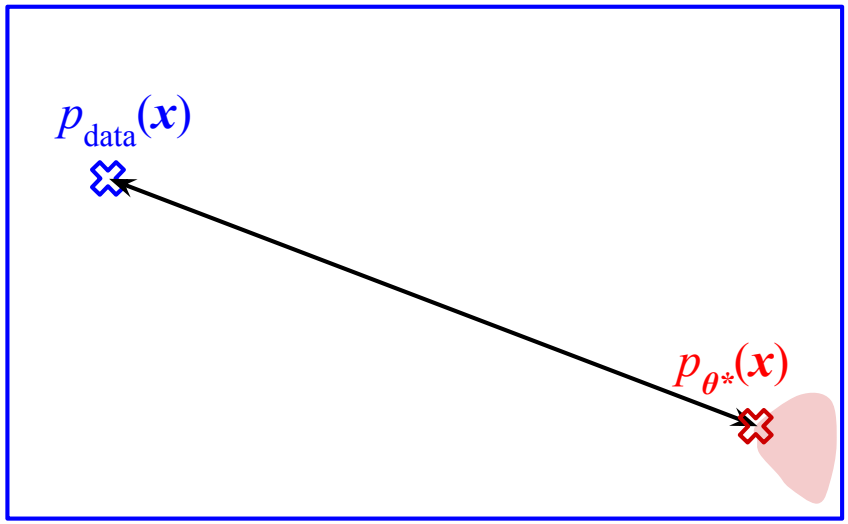
Now assum $X_{i+1} \perp X_{i-1}, \ldots, X_1 | X_i$, then we can simplify the distribution as:

$$\begin{aligned} p(\boldsymbol{x}) &= p(x_1)p(x_2|x_1)p(x_3|x_2, x_1)\ldots p(x_n|x_{n-1}, \ldots, x_1) \\ &= \underbrace{p(x_1)}_{\#1}\underbrace{p(x_2|x_1)}_{\#2}\underbrace{p(x_3|x_2))}_{\#2}\ldots\underbrace{p(x_n|x_{n-1})}_{\#2} \end{aligned}$$

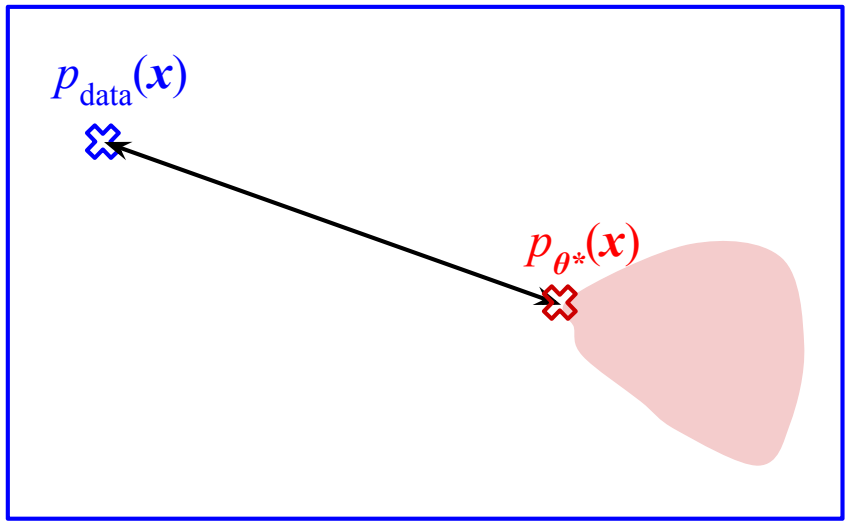
Then:

- Number of Categories: 2^n
- #Parameters for $\mathcal{P}$: $2n - 1$

Chain Rule + Conditional Indepence



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



(b) Typical $\mathcal{P}$ and $\mathcal{P}_\theta$ for first-order Markov chain

Figure: With first-order Markov chain, while controlling the tractability for the number of parameters, the chance to find a better fit to p_{data} increases

Section 6

Conclusions

Conclusions

Our Objective in This Course

- Introduce suitable subset of probability distribution functions $\mathcal{P}_\theta$:
  - Not too large (proposing $\mathcal{P}$ as $\mathcal{P}_\theta$)

Conclusions

Our Objective in This Course

- Introduce suitable subset of probability distribution functions $\mathcal{P}_\theta$:
  - Not too large (proposing $\mathcal{P}$ as $\mathcal{P}_\theta$)
  - Not too small (image with independent pixels)

Conclusions

Our Objective in This Course

- Introduce suitable subset of probability distribution functions $\mathcal{P}_\theta$:
  - Not too large (proposing $\mathcal{P}$ as $\mathcal{P}_\theta$)
  - Not too small (image with independent pixels)
  - Aware of input type (such as image, text, audio, video, etc.)

Conclusions

Our Objective in This Course

- Introduce suitable subset of probability distribution functions $\mathcal{P}_\theta$:
  - Not too large (proposing $\mathcal{P}$ as $\mathcal{P}_\theta$)
  - Not too small (image with independent pixels)
  - Aware of input type (such as image, text, audio, video, etc.)
- Introduce suitable distance metrics between two distributions

Conclusions

Our Objective in This Course

- Introduce suitable subset of probability distribution functions $\mathcal{P}_\theta$:
  - Not too large (proposing $\mathcal{P}$ as $\mathcal{P}_\theta$)
  - Not too small (image with independent pixels)
  - Aware of input type (such as image, text, audio, video, etc.)
- Introduce suitable distance metrics between two distributions
- Introduce the optimization methods to reduce the distance metric between model distribution and true distribution

Conclusions

Our Objective in This Course

- Introduce suitable subset of probability distribution functions $\mathcal{P}_\theta$:
  - Not too large (proposing $\mathcal{P}$ as $\mathcal{P}_\theta$)
  - Not too small (image with independent pixels)
  - Aware of input type (such as image, text, audio, video, etc.)
- Introduce suitable distance metrics between two distributions
- Introduce the optimization methods to reduce the distance metric between model distribution and true distribution
- Evaluate your modeling performance

List of Abbreviations

Complete	Abbreviation
Deep Generative Model	DGM
Multinomial Logistic Regression	MLR

References I


Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi,
"Image super-resolution via iterative refinement,"
IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 4, pp. 4713–4726, 2022.


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.