1 Course Introduction

Zhonglei Wang

WISE and SOE, XMU, 2025

Contents

1. What we have learnt

2. Problems

3. Learning goal

4. Cross entropy

What we have learnt

1. Types of datasets:
   - Training data of size n : $\{(\boldsymbol{x}_i, y_i) : i = 1, \ldots, n\}$
   - Test data
2. Regressions:
   - Linear regression: $y_i = b_0 + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}_0 + \epsilon_i \quad (i = 1, \ldots, n)$
   - Logistic regression: $\operatorname{logit} P(y_i = 1 \mid \boldsymbol{x}_i) = b_0 + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}_0 \quad (i = 1, \ldots, n)$
     - ▷ $\operatorname{logit} p = \log(p) - \log(1 - p), \quad \forall p \in (0, 1)$
   - Multinomial logistic regression (Softmax regression)

Notations

1. n : number of examples in the training data
2. d : dimension of features
3. $\boldsymbol{X} \in \mathbb{R}^{n \times d}$: design matrix
   - $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^{\mathrm{T}}$
   - $\boldsymbol{x}_i \in \mathbb{R}^{d \times 1}$: feature for the ith example
4. $\boldsymbol{Y} \in \mathbb{R}^{n \times 1}$: vector of labels
   - $\boldsymbol{Y} = (y_1, \ldots, y_n)^{\mathrm{T}}$
   - $y_i \in \mathbb{R}$: label for the ith example

Linear regression -- Model setup

1. Training dataset
    - Feature, label: $\boldsymbol{x}_i \in \mathbb{R}^{d\times 1},\ y_i \in \mathbb{R} \quad (i = 1, \ldots, n)$
2. Goal
    - Learn the relationship between $\boldsymbol{x}$ and y
3. True Model (used to generate data)
    - $y_i = b_0 + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}_0 + \epsilon_i \quad (i = 1, \ldots, n)$
        - ▷ $\boldsymbol{\theta}_0 = (b_0, {\boldsymbol{w}_0}^{\mathrm{T}})^{\mathrm{T}}$: true model parameters
        - ▷ ϵ_i : noise satisfying $E(\epsilon_i \mid \boldsymbol{x}_i) = 0,$ and $\mathrm{var}(\epsilon_i \mid \boldsymbol{x}_i) = \sigma^2 > 0$

Linear regression -- Model setup

1. Proposed Model (used to fit data)

- $E(y_i \mid \boldsymbol{x}_i) = b + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w} \quad (i = 1, \ldots, n)$
    - ▷ $\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$: (Proposed) Model parameters
    - ▷ Have the same form with the true model, but we need to estimate parameters
    - ▷ By fitting a model, we mean to estimate parameters of the proposed model

Linear regression -- Parameter estimation

1. Loss function: For the ith example,
   - l_2-loss: $\mathcal{L}(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$
   - $\hat{y}_i = b + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}$
   - $\hat{y}_i$ is a function of $\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$, but we omit its argument for simplicity
   - $\mathcal{L}(y_i, \hat{y}_i)$ is actually a function of model parameters $\boldsymbol{\theta}$
2. Cost function

$$\mathcal{J}(\boldsymbol{\theta}) = n^{-1} \sum_{i=1}^{n} \mathcal{L}(y_i, \hat{y}_i) = n^{-1} \sum_{i=1}^{n} (y_i - b - \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w})^2$$

3. Sometimes, we do not distinguish loss function and cost function

Linear regression -- Parameter estimation

1. Solve

$$\frac{\mathcal{J}}{\partial b} = 0, \quad \frac{\mathcal{J}}{\partial \boldsymbol{w}} = 0$$

2. Problem: low computational efficiency due to summation

Vectorization

1. Denote $\hat{\boldsymbol{Y}} = b\mathbf{1}_n + \boldsymbol{X}\boldsymbol{w}$
   - $\mathbf{1}_n = (1, \ldots, 1)^{\mathrm{T}} \in \mathbb{R}^{n\times 1}$: vector of 1's with length n
2. Cost function

$$
\begin{aligned}
\mathcal{J}(\boldsymbol{\theta}) &= n^{-1}(\boldsymbol{Y} - \hat{\boldsymbol{Y}})^{\mathrm{T}}(\boldsymbol{Y} - \hat{\boldsymbol{Y}}) \\
&= n^{-1}(\boldsymbol{Y} - b\mathbf{1}_n - \boldsymbol{X}\boldsymbol{w})^{\mathrm{T}}(\boldsymbol{Y} - b\mathbf{1}_n - \boldsymbol{X}\boldsymbol{w}) \\
&= n^{-1}(\boldsymbol{Y} - \tilde{\boldsymbol{X}}\boldsymbol{\theta})^{\mathrm{T}}(\boldsymbol{Y} - \tilde{\boldsymbol{X}}\boldsymbol{\theta})
\end{aligned}
$$

   - $\tilde{\boldsymbol{X}} = (\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_n)^{\mathrm{T}}$ (Integrate b and $\boldsymbol{w}$ together)
   - $\tilde{\boldsymbol{x}}_i = (1, \boldsymbol{x}_i^{\mathrm{T}})^{\mathrm{T}}$

Vectorization

1. Consider

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}} = 0$$

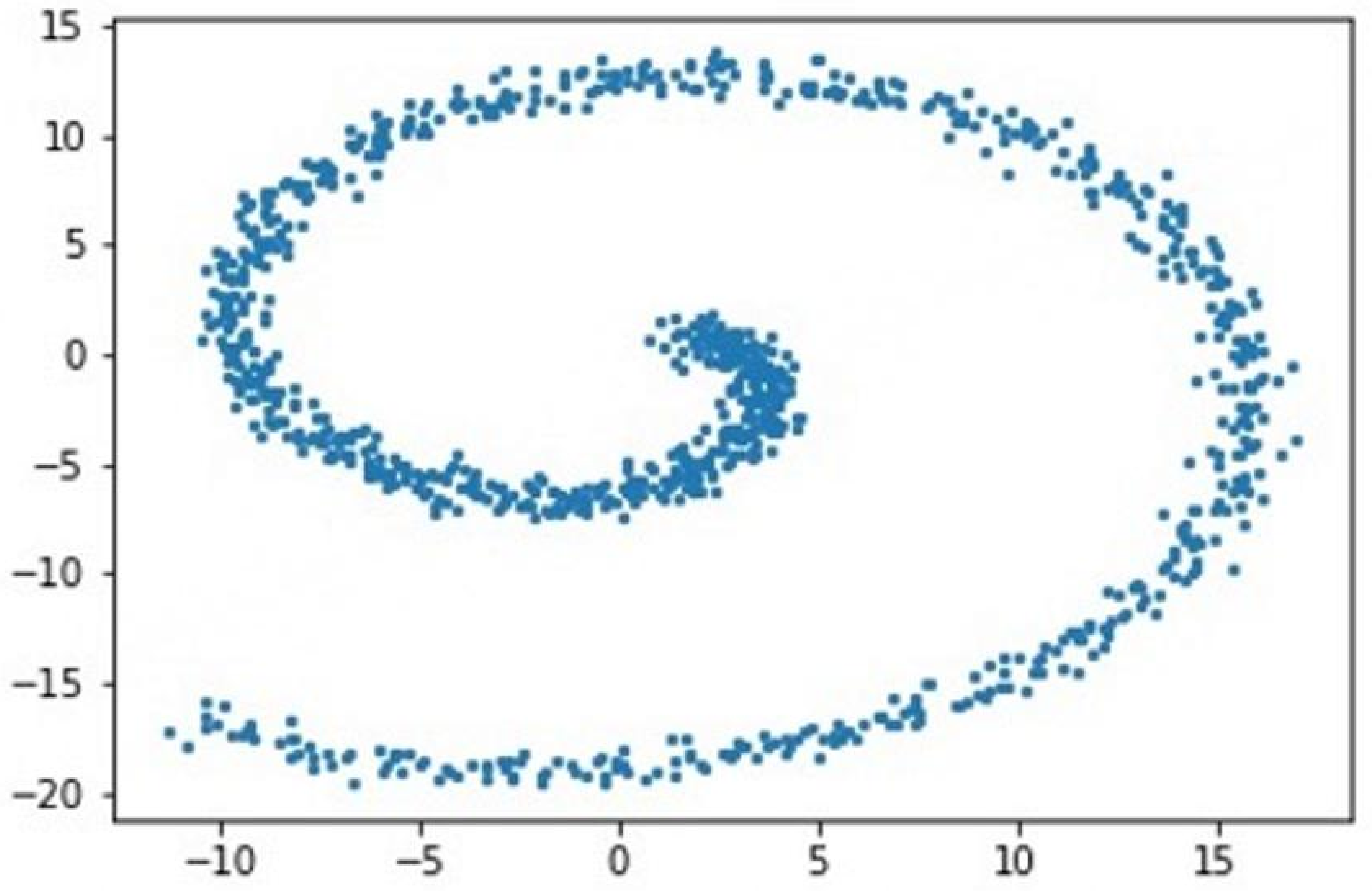
2. Obtain the normal equation

$$\tilde{\boldsymbol{X}}^{\mathrm{T}} \tilde{\boldsymbol{X}} \boldsymbol{\theta} = \tilde{\boldsymbol{X}}^{\mathrm{T}} \boldsymbol{Y}$$

3. Solution (Closed-form or analytical solution)

$$\hat{\boldsymbol{\theta}} = (\tilde{\boldsymbol{X}}^{\mathrm{T}} \tilde{\boldsymbol{X}})^{-1} \tilde{\boldsymbol{X}}^{\mathrm{T}} \boldsymbol{Y}$$

Example



Logistic regression -- Model setup

1. Training data
   - Feature, labels: $\boldsymbol{x}_i \in \mathbb{R}^{d\times 1}$, $y_i \in \{0, 1\} \quad (i = 1, \ldots, n)$
2. Goal
   - Learn the relationship between $\boldsymbol{x}$ and y
3. True model (used to generate data)

$$P(y_i = 1 \mid \boldsymbol{x}_i) = \sigma(b_0 + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}_0) \quad (i = 1, \ldots, n)$$

   - $\sigma(z) = \{1 + \exp(-z)\}^{-1}$ (Why we need this transformation?)

Logistic regression -- Model setup

1. Proposed Model (used to fit data)

- $P(y_i = 1 \mid \boldsymbol{x}_i) = \sigma(b_0 + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}_0) \quad (i = 1, \ldots, n)$
  - ▷ $\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$: (Proposed) Model parameters
  - ▷ Have the same form with the true model, but we need to estimate parameters
  - ▷ By fitting a model, we mean to estimate parameters of the proposed model

Logistic regression -- Parameter estimation

1. Loss function: For the ith example,

- Cross entropy (minus log-likelihood)

$$\mathcal{L}(y_i, a_i) = -\{y_i \log a_i + (1 - y_i) \log(1 - a_i)\}$$

  - $a_i = \sigma(z_i)$ (Estimated probability)
  - $z_i = b + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}$

- a_i is a function of $\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$, but we omit its argument for simplicity
- Cross entropy is actually a function of $\boldsymbol{\theta}$

Logistic regression -- Parameter estimation

1. Cost function

$$\mathcal{J}(\boldsymbol{\theta}) = n^{-1} \sum_{i=1}^{n} \mathcal{L}\{y_i, a_i\}$$
$$= -n^{-1} \sum_{i=1}^{n} \{y_i \log a_i + (1 - y_i) \log\{1 - a_i\}\}$$

- $\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$
- $a_i = \sigma(b + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w})$

Logistic regression -- Parameter estimation

1. Solve

$$\frac{\mathcal{J}}{\partial \boldsymbol{\theta}} = 0$$

2. No analytical solution

Newton-Raphson algorithm

Step 1. Randomly initialize $\boldsymbol{\theta}^{(0)}$

Step 2. Based on $\boldsymbol{\theta}^{(t)}$ obtain

$$\nabla \mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right) = \frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}}\left(\boldsymbol{\theta}^{(t)}\right)$$

$$H(\mathcal{J})\left(\boldsymbol{\theta}^{(t)}\right) = \frac{\partial^2 \mathcal{J}}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^{\mathrm{T}}}\left(\boldsymbol{\theta}^{(t)}\right)$$

Step 3. Update parameter

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \left[H(\mathcal{J})\left(\boldsymbol{\theta}^{(t)}\right)\right]^{-1} \nabla \mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right)$$

Step 4. Go back to Step 2 until convergence

Discussion

1. Fast convergence by incorporating a Hessian matrix
2. Computation efficiency is sacrificed at the same time
3. Feasible when the parameter dimension is low
4. Not applicable for deep learning models
5. What algorithms should we use for deep learning models?

(Batch) gradient descent algorithm

Step 1. Randomly initialize $\boldsymbol{\theta}^{(0)}$

Step 2. Based on $\boldsymbol{\theta}^{(t)}$ obtain

$$\nabla \mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right) = \frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta}^{(t)})$$

Step 3. Update parameter

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \alpha \nabla \mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right)$$

Step 4. Go back to Step 2 until convergence

(Batch) gradient descent algorithm

1. α : learning rate
   - Controls the speed of convergence
   - Affects the performance of the model
   - To be discussed

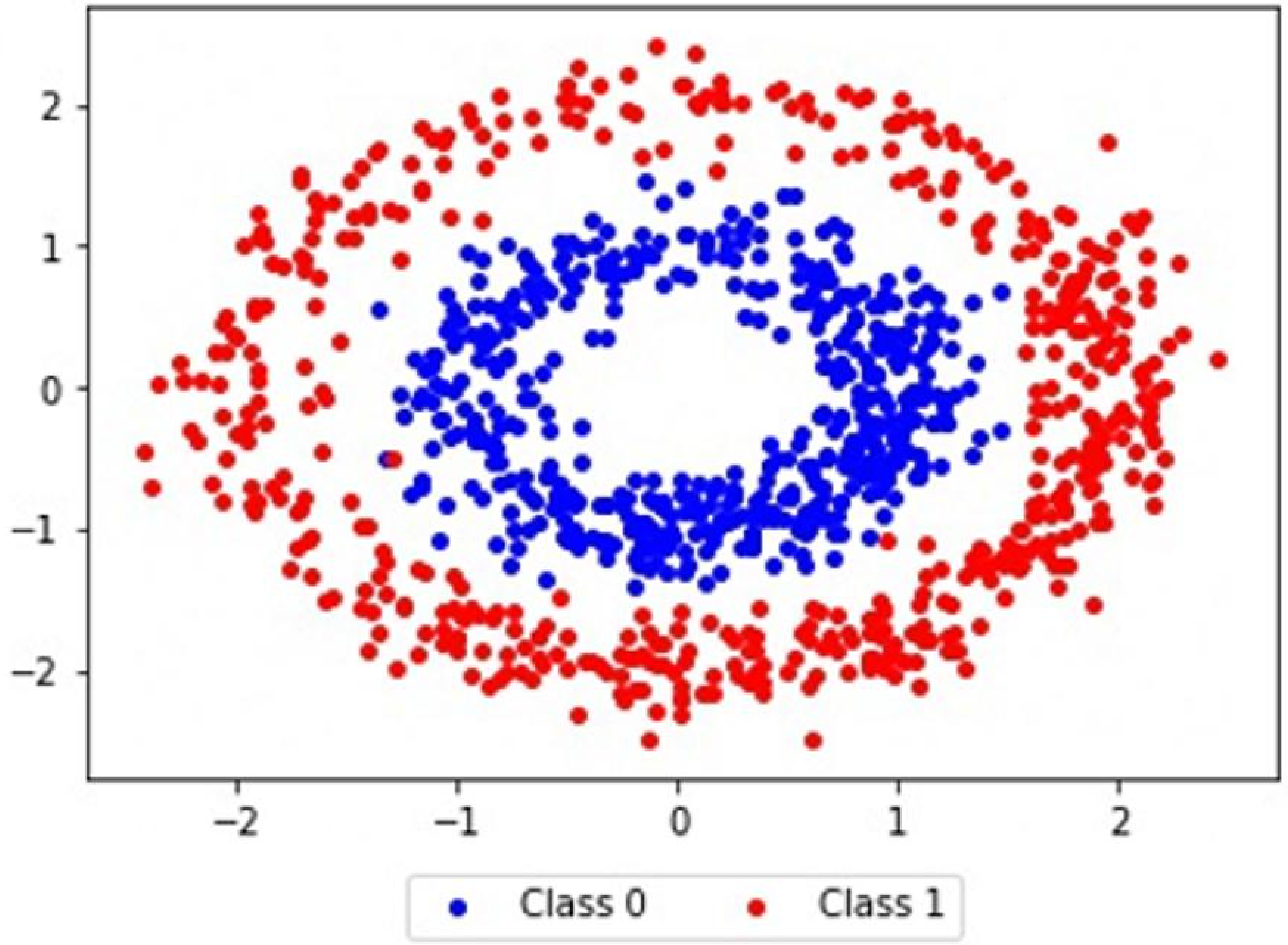
Comparison of algorithms

1. Newton-Raphson algorithm
   - Advantages:
     - ▷ Fast convergence rate
     - ▷ High accuracy
   - Disadvantages:
     - ▷ Low computation efficiency due to the Hessian matrix
     - ▷ Low stability if the Hessian matrix is ill-conditioned

Comparison of algorithms (Cont'd)

1. (Batch) gradient descent algorithm
   - Advantages :
     - ▷ High computation efficiency
     - ▷ Each to implement
   - Disadvantages :
     - ▷ Low convergence rate
     - ▷ Sensitive to learning rate

Example



Problems

1. Up to now, we have assumed the same form for the true model and the proposed
2. However, it is commonly not the case in practice
3. Model misspecification
   - The proposed (statistical) models are too simple
   - True model, however, is COMPLEX

Learning goal

1. Fully connected neural network (multiple layer perceptron)
2. Convolutional Neural Network (CNN, ...)
3. Sequential modeling (RNN, LSTM, transformers, ...)
4. (If we have time,) More topics （GNN, GCN, ...）

Learning goal

1. Fully connected neural network (multiple layer perceptron)
2. Convolutional Neural Network (CNN, ...)
3. Sequential modeling (RNN, LSTM, transformers, ...)
4. (If we have time,) More topics （GNN, GCN, ...）

FNN Example

Test image

Model (FNN with 2 hidden layers)

Estimated result

Input layer $d^{[0]} = 784$

1st hidden layer $d^{[1]} = 128$

2nd hidden layer $d^{[2]} = 64$

Output layer $d^{[3]} = 10$

FNN Example

Test image

Model (FNN with 2 hidden layers)

Input layer $d^{[0]} = 784$

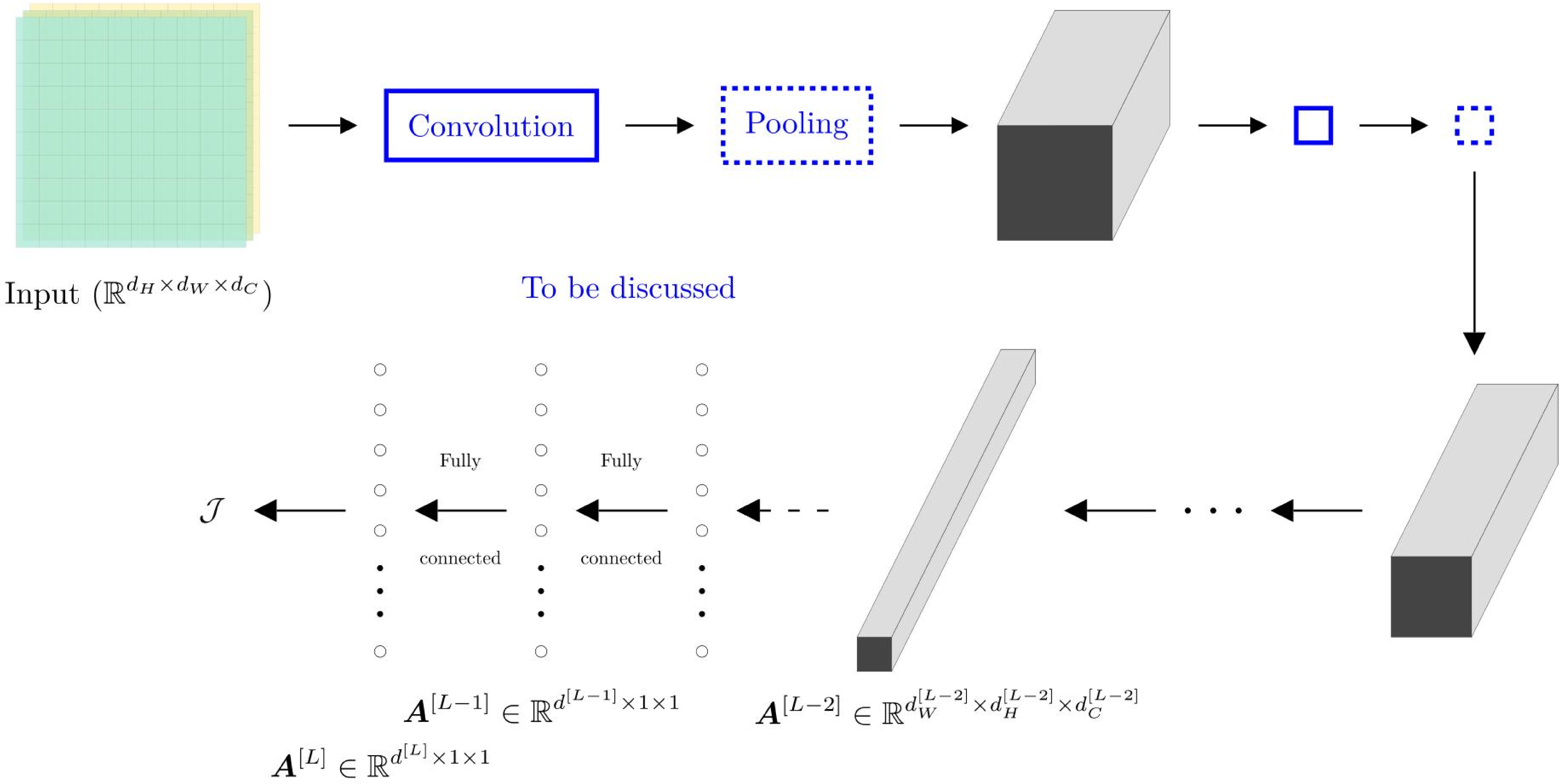
1st hidden layer $d^{[1]} = 128$

2nd hidden layer $d^{[2]} = 64$

Output layer $d^{[3]} = 10$

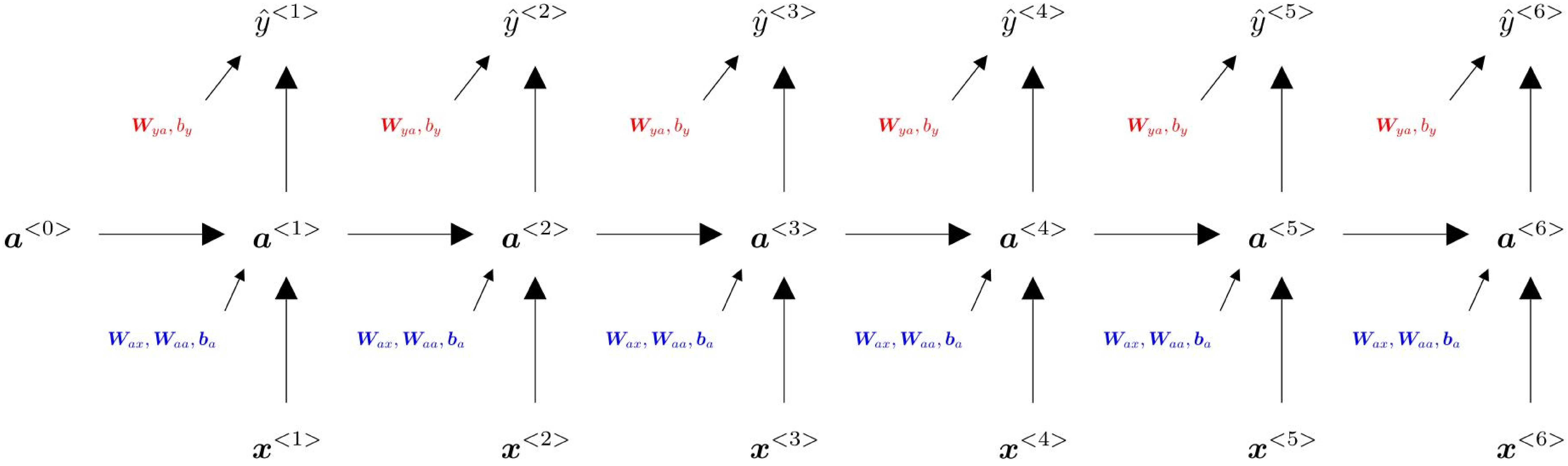
Estimated result

$\longrightarrow$ 2


Convolution
Pooling
Input ($\mathbb{R}^{d_H \times d_W \times d_C}$)
To be discussed
Fully
connected
Fully
connected
$\mathcal{J}$
$\boldsymbol{A}^{[L-1]} \in \mathbb{R}^{d^{[L-1]} \times 1 \times 1}$
$\boldsymbol{A}^{[L-2]} \in \mathbb{R}^{d_W^{[L-2]} \times d_H^{[L-2]} \times d_C^{[L-2]}}$
$\boldsymbol{A}^{[L]} \in \mathbb{R}^{d^{[L]} \times 1 \times 1}$

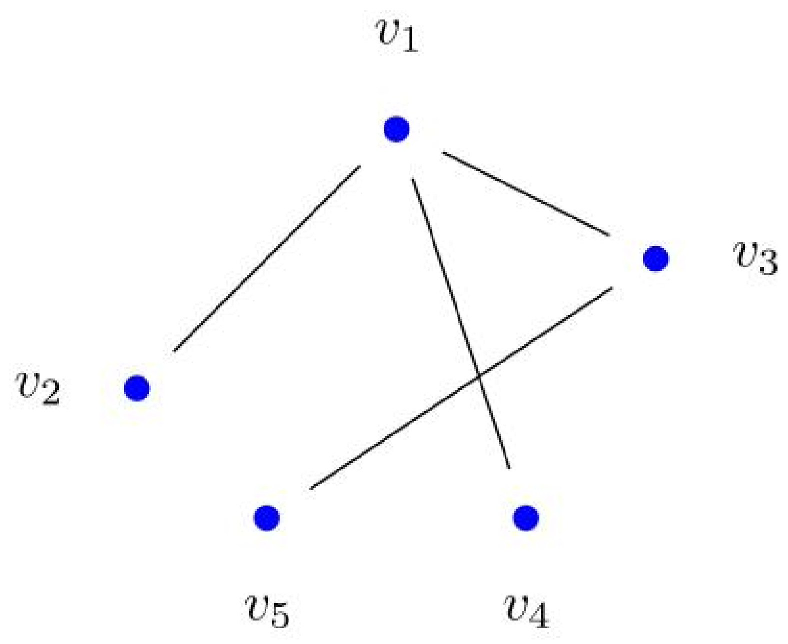
Building block for RNN

$$\hat{y}^{<i>} = \sigma_{ya}(\boldsymbol{W}_{ya}\boldsymbol{a}^{<i>} + b_y) \quad (i = 1, \ldots, 6)$$



$$\boldsymbol{a}^{<i>} = \sigma_{ax}(\boldsymbol{W}_{ax}\boldsymbol{x}^{<i>} + \boldsymbol{W}_{aa}\boldsymbol{a}^{<i-1>} + \boldsymbol{b}_a) \quad (i = 1, \ldots, 6)$$

Undirected graph



Vertex set: $\mathcal{V} = \{v_i : i = 1, \ldots, n\}$

Edge set: $\mathcal{E}$

Undirected graph: $\mathcal{G} = \mathcal{V} \cup \mathcal{E}$

Adjacent matrix (binary or weighted)

$$A = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Degree matrix (diagonal, summation of each row in A)

$$D = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Cross entropy

1. A fact: under regularity conditions,
$$E_{X\sim P}\{\log p(X)\} \geq E_{X\sim P}\{\log q(X)\}$$
   - $X \sim P$: random variable X is generated from a distribution P
   - $p(x)$: density function of the distribution P
   - $q(x)$: another density function of a certain distribution
2. Cross entropy for two density functions p and q
$$H(p,q) = -E_{X\sim P}\{\log q(X)\} = -E\{\log q(X)\}$$
   - Measures how well a density q approximates the distribution of X.

Cross entropy

1. Only observe a random sample $\{x_1, \ldots, x_n\}$
2. Cross entropy $H(p, q)$ can be approximated by

$$\hat{H}(p, q) = -\frac{1}{n} \sum_{i=1}^{n} \log q(x_i)$$

   - It is the cross entropy between the empirical density $P_n(x) = n^{-1} \sum_{i=1}^{n} \delta(x = x_i)$ and q
3. Goal: Find a density function q to minimize $\hat{H}(p, q)$
   - That is, to approximate the distribution of the random sample well

Revisit logistic regression

1. Assume $\{(\boldsymbol{x}_i, y_i) : i = 1, \ldots, n\}$ is i.i.d. with density function

$$p(\boldsymbol{x}, y) = p(\boldsymbol{x})p(y \mid \boldsymbol{x})$$

- $p(\boldsymbol{x})$: unspecified marginal density for $\boldsymbol{x}$
- $p(y \mid \boldsymbol{x}) = \sigma(b_0 + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}_0)$: parametric conditional density for y

2. Goal: Find density$q(\boldsymbol{x}, y)$ to minimize

$$\hat{H}(p, q) = -\frac{1}{n} \sum_{i=1}^{n} \log q(\boldsymbol{x}_i, y_i)$$

Revisit logistic regression

1. A little more math

$$-\frac{1}{n}\sum_{i=1}^{n}\log q(\boldsymbol{x}_i, y_i) = -\frac{1}{n}\sum_{i=1}^{n}\{\log q(\boldsymbol{x}_i) + \log q(y_i \mid \boldsymbol{x}_i)\}$$

$$= -\frac{1}{n}\sum_{i=1}^{n}\log q(\boldsymbol{x}_i) - \frac{1}{n}\sum_{i=1}^{n}[y_i \log a_i(\boldsymbol{\theta}) + (1-y_i)\log\{1-a_i(\boldsymbol{\theta})\}]$$

- $q(\boldsymbol{x}, y) = q(\boldsymbol{x})q(y \mid \boldsymbol{x})$
- $q(\boldsymbol{x})$: unspecified
- $q(y \mid \boldsymbol{x}) = a(\boldsymbol{\theta})^y\{1-a(\boldsymbol{\theta})\}^{1-y}$
- $a(\boldsymbol{\theta}) = \sigma(b + \boldsymbol{x}^{\mathrm{T}}\boldsymbol{w})$: with parameter $\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$

Revisit logistic regression

1. Thus,

$$-\frac{1}{n}\sum_{i=1}^{n}\log q(\boldsymbol{x}_i, y_i) = -\frac{1}{n}\sum_{i=1}^{n}\log q(\boldsymbol{x}_i) - \frac{1}{n}\sum_{i=1}^{n}[y_i \log a_i(\boldsymbol{\theta}) + (1-y_i)\log\{1-a_i(\boldsymbol{\theta})\}]$$

   - The first part is of no interest
   - We focus on estimating parameters in the second part

2. Thus, it is equivalent to minimizing

$$-\frac{1}{n}\sum_{i=1}^{n}[y_i \log a_i(\boldsymbol{\theta}) + (1-y_i)\log\{1-a_i(\boldsymbol{\theta})\}]$$

   - The cost function for logistic regression