

# CS2109s - Tutorial 7

Eric Han

Oct 25, 2023

## Annoucements

### Important admin

- PS5 Comments:
  - Question 9: Task 3.5: Logistic regression using stochastic gradient descent
    - For iteration, update a randomly selected datapoint.
  - Question 11: Task 3.6.2: Stochastic gradient descent vs batch gradient descent
    - SGD faster BGD slower for every iteration
    - SGD *usually* reaches the same loss faster
    - SGD varies more due to the random updates (depending on point)
  - Question 16: Task 5.3 Linear SVM vs Gaussian Kernel SVM
    - Linear Separability: More features than data points lead to sparsity.
    - Model Complexity
- PS7 will be released on Fri, 3 Nov

### Question 1 [G]

ID	$x_1$	$x_2$	AND	OR	XOR
0	0	0	0	0	0
1	0	1	0	1	1
2	1	0	0	1	1
3	1	1	1	1	0

$x$	NOT
0	1
1	0

- Determine a function that can be used to model the decision boundaries of the logical NOT, OR, and AND gates. What are the weights and bias?
- Is it possible to model the XOR function using a single Perceptron? [G] Proof.
- Model XOR using a number of NOT, OR, and AND perceptrons.
- If data samples are reordered, will the model converges to a different model?
- Does your proposed models (AND, OR, NOT) have high bias and high variance?

### Recap

What is a Perceptron and what is the Perceptron Update Rule?

---

### Answer 1a

$$y = X \times w^T + w_0$$

### AND Gate - 4 iters, 11 updates

```
iter=0 idx=0 w=[-0.1  0.  0. ]
iter=0 idx=3 w=[0.  0.1 0.1]
iter=1 idx=0 w=[-0.1  0.1 0.1]
iter=1 idx=1 w=[-0.2  0.1 0. ]
iter=1 idx=3 w=[-0.1  0.2 0.1]
iter=2 idx=1 w=[-0.2  0.2 0. ]
```

```

iter=2 idx=2 w=[-0.3  0.1  0. ]
iter=2 idx=3 w=[-0.2  0.2  0.1]
iter=3 idx=2 w=[-0.3  0.1  0.1]
iter=3 idx=3 w=[-0.2  0.2  0.2]
iter=4 idx=1 w=[-0.3  0.2  0.1]

```

---

### OR Gate - 2 iters, 5 updates

```

iter=0 idx=0 w=[-0.1  0.  0. ]
iter=0 idx=1 w=[0.  0.  0.1]
iter=1 idx=0 w=[-0.1  0.  0.1]
iter=1 idx=2 w=[0.  0.1  0.1]
iter=2 idx=0 w=[-0.1  0.1  0.1]

```

### NOT Gate - 1 iters, 2 updates

```

iter=0 idx=1 w=[-0.1 -0.1]
iter=1 idx=0 w=[ 0.  -0.1]

```

---

### Answer 1b

XOR gate is not linearly separable

### Answer 1c

$$XOR(x_1, x_2) = AND(NOT(AND(x_1, x_2)), OR(x_1, x_2))$$

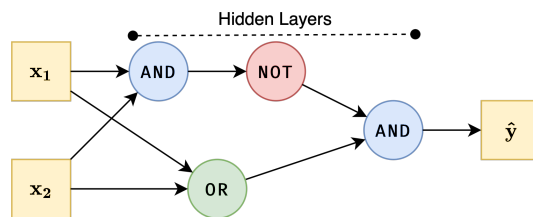


Figure 1: Layers are important to generalize better complex data.

---

### Answer 1d

Ordering	Iterations	No. of Updates	Weight	No. of Correct
[0, 1, 2, 3]	4	11	[-0.3 0.2 0.1]	4
[0, 2, 3, 1]	4	13	[-0.3 0.2 0.1]	4
[0, 2, 1, 3]	4	11	[-0.3 0.1 0.2]	4

- Reordering can help model converge faster
- Reordering can change the optimum point found - potentially many local optimas.

### Answer 1e

The proposed model has low bias and low variance; They all classify correctly.

### Question 2 [G]

Perceptron	MSE Train	MSE Validation
Single	1000	2000
Multi	800	1200

- Why the difference in performance?
- How to improve Single's performance? What are the advantages / disadvantages?
- How to improve the performance of the multi-layer perceptron?

## Recap

What does adding layers do?

---

## Answer 2

- Complexity needed to classify dataset is likely non-linear boundary
  - Single-layer: Less Complex, linear classifier
  - Multi-layer: More Complex, non-linear classifier
- Feature Engineering, to 'transform' the space
  - Add polynomial terms
  - Add interaction terms
- Improve...?
  - Performance: Increase complexity, add hidden layer
  - Reduce overfitting: Regularization

## Question 3

Neural Network with 2D input, one hidden layer, with bias, using ReLU, MSE.

$$W^{[1]} = \begin{bmatrix} 0.1 & 0.1 \\ -0.1 & 0.2 \\ 0.3 & -0.4 \end{bmatrix}, W^{[2]} = \begin{bmatrix} 0.1 & 0.1 \\ 0.5 & -0.6 \\ 0.7 & -0.8 \end{bmatrix}, b = 1, X = [2, 3], y = [.1, .9]$$

Calculate the following values after the forward propagation:  $\mathbf{a}^{[1]}$ ,  $\mathbf{y}^{[2]}$  and  $L(\mathbf{y}^{[2]}, \mathbf{y})$ .

## Recap

- How to do forward pass?
  - What is Loss/MSE?
  - What is ReLU?
- 

## Answer 3

Layer 1:

$$\mathbf{a}^{[1]T} = \text{ReLU}(W^{[1]T} \times X^T) = \text{ReLU}\left(\begin{bmatrix} 0.1 & -0.1 & 0.3 \\ 0.1 & 0.2 & -0.4 \end{bmatrix} \times \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}\right) = \begin{bmatrix} 0.8 \\ 0 \end{bmatrix}$$

Layer 2:

$$\mathbf{y}^{[2]T} = \text{ReLU}(W^{[2]T} \times \mathbf{a}^{[1]T}) = \text{ReLU}\left(\begin{bmatrix} 0.1 & 0.5 & 0.7 \\ 0.1 & -0.6 & -0.8 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0.8 \\ 0 \end{bmatrix}\right) = \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}$$

Loss:

$$L(\mathbf{y}^{[2]}, \mathbf{y}) = 0.5 \|\mathbf{y}^{[2]} - \mathbf{y}\|_2 = 0.5 \times ((0.5 - 0.1)^2 + (0 - 0.9)^2) = 0.5 \times (0.16 + 0.81) = 0.485$$

## Question 4 [G]

$$\hat{y} = g(\mathbf{W}^{[L]\mathbf{T}} \dots g(\mathbf{W}^{[2]\mathbf{T}} \cdot g(\mathbf{W}^{[1]\mathbf{T}} x))$$

where  $\mathbf{W}^{[l] \in \{1, \dots, L\}}$  is a weight matrix. You're given the following weight matrices:

$$\mathbf{W}^{[3]} = \begin{bmatrix} 1.2 & -2.2 \\ 1.2 & 1.3 \end{bmatrix}, \mathbf{W}^{[2]} = \begin{bmatrix} 2.1 & -0.5 \\ 0.7 & 1.9 \end{bmatrix}, \mathbf{W}^{[1]} = \begin{bmatrix} 1.4 & 0.6 \\ 0.8 & 0.6 \end{bmatrix}$$

You are given  $g(z) = \text{SiLU}(z) = \frac{z}{1+e^{-z}}$  between all layers *except the last layer*.

- Is it possible to replace the whole neural network with just one matrix in both cases **with** and **without** non-linear activations  $g(z)$ ?
  - What does this signify about the importance of the non-linear activation?
- 

### Answer 4a

**without** non-linear activations:

$$\begin{aligned} M^T &= \begin{bmatrix} 1.2 & -2.2 \\ 1.2 & 1.3 \end{bmatrix}^T \begin{bmatrix} 2.1 & -0.5 \\ 0.7 & 1.9 \end{bmatrix}^T \begin{bmatrix} 1.4 & 0.6 \\ 0.8 & 0.6 \end{bmatrix}^T \\ &= \begin{bmatrix} 4.56 & 3.408 \\ -6.82 & -3.658 \end{bmatrix} \end{aligned}$$

---

**with** non-linear activations: suppose  $x_1 = [1, 0]$  and  $x_2 = [2, 0]$ :

$$\begin{aligned} [\hat{y}_1, \hat{y}_2] &= \begin{bmatrix} 1.2 & -2.2 \\ 1.2 & 1.3 \end{bmatrix}^T g \left( \begin{bmatrix} 2.1 & -0.5 \\ 0.7 & 1.9 \end{bmatrix}^T g \left( \begin{bmatrix} 1.4 & 0.6 \\ 0.8 & 0.6 \end{bmatrix}^T \begin{bmatrix} 1 \\ 2 \end{bmatrix} \right) \right) \\ &= \begin{bmatrix} 3.0571, 7.7257 \\ -5.2727, -13.2458 \end{bmatrix} \end{aligned}$$

Assume  $\mathbf{M}^T$  exist:

- $x_2 = 2x_1$
  - $\mathbf{M}^T x_2 = 2\mathbf{M}^T x_1 \implies \hat{y}_2 = 2\hat{y}_1$  by linearity of  $\mathbf{M}^T$ .
  - But,  $\hat{y}_2 \neq 2\hat{y}_1$ , thus there exist no such  $\mathbf{M}^T$ .
- 

### Answer 4b

$$\begin{aligned} \hat{y} &= \mathbf{W}^{[L]\mathbf{T}} \dots \mathbf{W}^{[2]\mathbf{T}} \mathbf{W}^{[1]\mathbf{T}} x \\ &= \mathbf{A}x, \quad \text{where } \mathbf{A} = \mathbf{W}^{[L]\mathbf{T}} \dots \mathbf{W}^{[2]\mathbf{T}} \mathbf{W}^{[1]\mathbf{T}} \text{ by matrix multiplication} \end{aligned}$$

- Without non-linear activations, the entire network **collapses to a simple linear model**; adding layers is futile.
- With non-linear activation functions let the network model non-linear relationships.

The non-linear activation gives the Neural Network its representation power - without which the parameters in the network behave the same way.

## Question 5 [G]

Takes in grayscale images of size  $32 \times 32$  and outputs 4 classes, with 3 layers, assuming batch size is 1.

- What are the dimensions of the input vector, the weight matrix, and the output vector of the three linear layers?
- [@] How would this look like for a CNN? Compare with the setup here.

## Recap

- How does one layer interact with the next?

## Answer

layer	Input dim	Weight Matrix dim	Output dim
Linear layer 1	$1024 \times 1$	$1024 \times 512$	$512 \times 1$
Linear layer 2	$512 \times 1$	$512 \times 128$	$128 \times 1$
Linear layer 3	$128 \times 1$	$128 \times 4$	$4 \times 1$

## Q1 Visualization

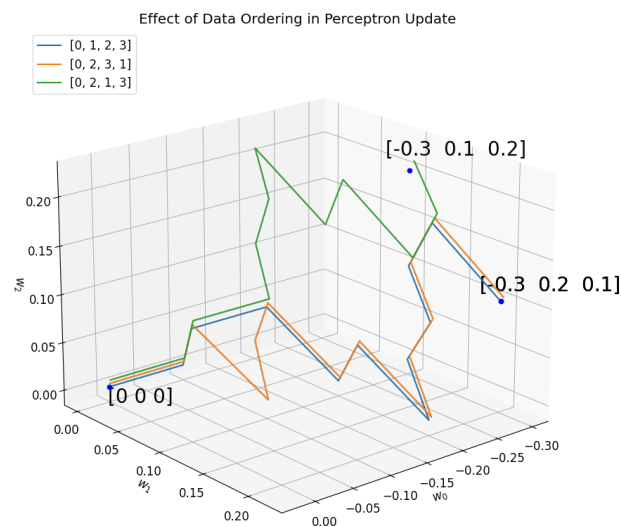


Figure 2: Q1 Viz

## Bonus Qn

To help you further your understanding, not compulsory; Work for Snack/EXP!

### Tasks

1. Implement the missing code for `FconLayer`, `NNetwork` and `M` in the boilerplate code given.
2. You must use Matrix operations where possible.
3. You must use `reduce` where possible. (Prompts in the code)
4. `FconLayer` should work properly with/without bias.

## Buddy Attendance Taking

Take Attendance for your buddy: <https://forms.gle/Ckkq639TNwWEx3NT6>

1. Random checks will be conducted - `python ../checks.py TGO`



Figure 3: Buddy Attendance