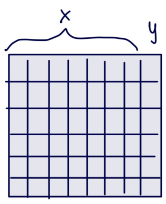
12 ก.พ. 2024

Lecture : Deep Learning

คาบที่9

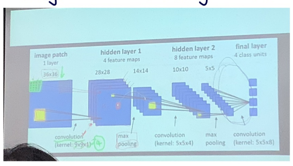
12 กพ (คาบ 9)

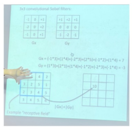
Classical

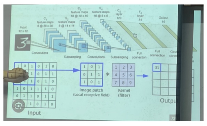


$feture = \{x_1, x_2, x_3, \dots, x_n\} | x \in \mathbb{R}$

Histogram of Oriented gradients



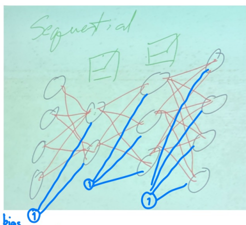




คาบที่10

19 ก.พ. Deep Learning

Sequential



bias
- ไม่ต้องเรียนรู้อะไร
- ไม่รับ input อะไร
- เลข 1 ตลอด

flatten

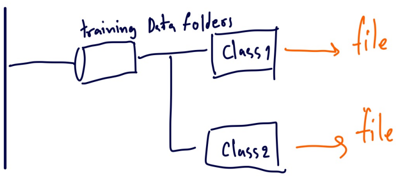
optimizer = วิ่ง

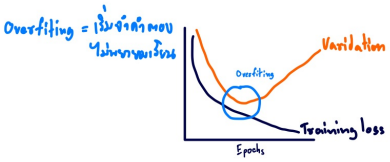
loss = ผิดมาก ผิดน้อย

metrics = ตัววัดความแม่นยำ

Data Pipeline

└ โหลดภาพทีละส่วน ๆ



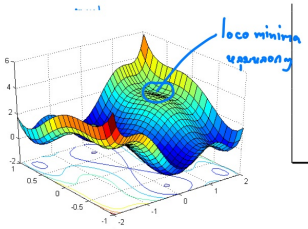


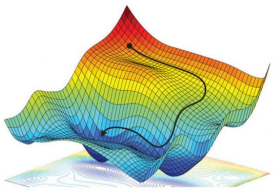
ตารางที่ 6–17 อัลกอริทึมกฎการเรียนรู้เพอร์เซปตรอน

```
Algorithm: Perceptron-Learning-Rule
1. Initialize weights w_i of the perceptron.
2. UNTIL the termination condition is met DO
     2.1 FOR EACH training example DO
           - Input the example and compute the output.
           - Change the weights if the output from the
             perceptron is not equal to the target
             output using the following rule.
                 w_i ← w_i + Δw_i
                 Δw_i ← α(t−o)x_i
             where t, o and α are the target output, the
             output from the perceptron and the learning
             rate, respectively.
```

Loss function

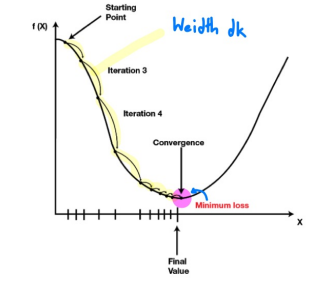
$$\text{Loss} = -\frac{1}{\text{output size}} \sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log (1 - \hat{y}_i)$$

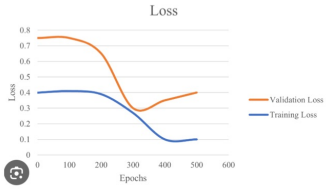


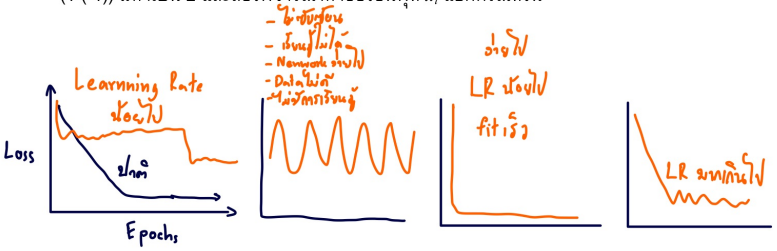


การปรับน้ำหนักตามกฎการเรียนรู้เพอร์เซปตรอนโดยใช้อัตราการเรียนรู้ที่มีค่าน้อยเพียงพอ จะได้ระนาบหลายมิติที่จะลู่เข้าสู่ระนาบหนึ่งที่สามารถแบ่งข้อมูลออกเป็นสองส่วน (ในกรณีที่ข้อมูลสามารถแบ่งได้) เพื่ออธิบายผลที่เกิดจากการปรับค่าน้ำหนัก เราจะลองพิจารณาพฤติกรรมของกฎการเรียนรู้นี้ดูว่าทำไมการปรับน้ำหนักเช่นนี้จึงลู่เข้าสู่ระนาบที่แบ่งข้อมูลได้อย่างถูกต้อง

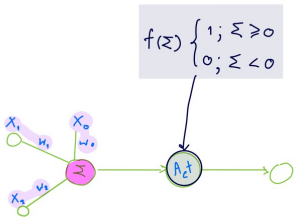
- พิจารณากรณีแรกที่เพอร์เซปตรอนแยกตัวอย่างสอนตัวหนึ่งที่รับเข้ามาได้ถูกต้อง กรณีนี้จะพบว่า (t-o) จะมีค่าเป็น 0 ดังนั้น Δw_i ไม่เปลี่ยนแปลงเพราะ $\Delta w_i = \alpha(t\text{-}o)x_i$
- พิจารณาในกรณีที่เพอร์เซปตรอนให้เอาต์พุตเป็น –1 แต่เอาต์พุตเป้าหมายหรือค่าที่แท้จริงเท่ากับ 1 ในกรณีนี้หมายความว่าค่าที่เราต้องการคือ 1 แต่ค่าน้ำหนักไม่เหมาะสม ดังนั้นเพื่อที่จะทำให้เพอร์เซปตรอนให้เอาต์พุตเป็น 1 น้ำหนักต้องถูกปรับให้สามารถเพิ่มค่าของ $\vec{w} \cdot \vec{x}$ ในกรณีนี้หมายความว่าผลรวมเชิงเส้นน้อยเกินไปและน้อยกว่า 0 จึงได้เอาต์พุตเป็น -1 ดังนั้นสิ่งที่เราต้องการคือการเพิ่มค่าผลรวมเชิงเส้นเพราะถ้าเราเพิ่มค่าได้เรื่อยๆ จนมากกว่า 0 เพอร์เซปตรอนจะให้เอาต์พุตเป็น 1 ซึ่งตรงกับที่เราต้องการ พิจารณาดูดังต่อไปนี้ว่าการปรับค่าโดยกฎเรียนรู้ทำให้ผลรวมเชิงเส้นเพิ่มขึ้นได้อย่างไร กรณีนี้เราจะได้ว่า (t-o) เท่ากับ (1-(-1)) มีค่าเป็น 2 และลองพิจารณาค่าของอินพุต x_i แยกกรณีดังนี้







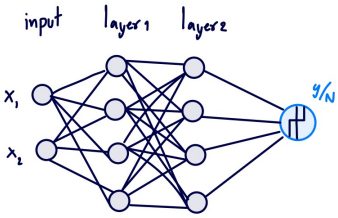
wi ←wi +Δwi
Δwi ← α(t–o)xi



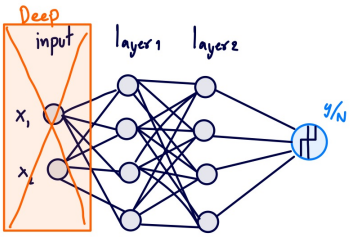
ตารางที่ 6–19 ผลการเรียนรู้ฟังก์ชัน AND โดยกฎการเรียนรู้เพอร์เซปตรอน

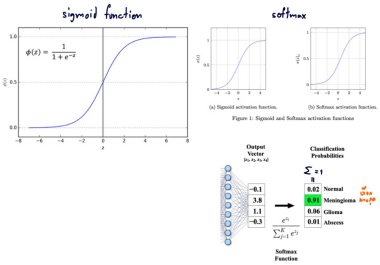
Perceptron Learning Example - Function AND											
		Input node			(Σ)	y					
		Bias Input x0=+1					Alpha =	0.5			
Input	Input				Net Sum	Target	Actual	Alpha*	Weight Values		
x1	x2	1.0*w0	x1*w1	x2*w2	Input	Output	Output	Error	w0	w1	w2
									0.1	0.1	0.1
0	0	0.10	0.00	0.00	0.10	0	1	-0.50	-0.40	0.10	0.10
0	1	-0.40	0.00	0.10	-0.30	0	0	0.00	-0.40	0.10	0.10
1	0	-0.40	0.10	0.00	-0.30	0	0	0.00	-0.40	0.10	0.10
1	1	-0.40	0.10	0.10	-0.20	1	0	0.50	0.10	0.60	0.60
0	0	0.10	0.00	0.00	0.10	0	1	-0.50	-0.40	0.60	0.60
0	1	-0.40	0.00	0.60	0.20	0	1	-0.50	-0.90	0.60	0.10
1	0	-0.90	0.60	0.00	-0.30	0	0	0.00	-0.90	0.60	0.10
1	1	-0.90	0.60	0.10	-0.20	1	0	0.50	-0.40	1.10	0.60
0	0	-0.40	0.00	0.00	-0.40	0	0	0.00	-0.40	1.10	0.60
0	1	-0.40	0.00	0.60	0.20	0	1	-0.50	-0.90	1.10	0.10
1	0	-0.90	1.10	0.00	0.20	0	1	-0.50	-1.40	0.60	0.10
1	1	-1.40	0.60	0.10	-0.70	1	0	0.50	-0.90	1.10	0.60
0	0	-0.90	0.00	0.00	-0.90	0	0	0.00	-0.90	1.10	0.60
0	1	-0.90	0.00	0.60	-0.30	0	0	0.00	-0.90	1.10	0.60
1	0	-0.90	1.10	0.00	0.20	0	1	-0.50	-1.40	0.60	0.60
1	1	-1.40	0.60	0.60	-0.20	1	0	0.50	-0.90	1.10	1.10
0	0	-0.90	0.00	0.00	-0.90	0	0	0.00	-0.90	1.10	1.10
0	1	-0.90	0.00	1.10	0.20	0	1	-0.50	-1.40	1.10	0.60
1	0	-1.40	1.10	0.00	-0.30	0	0	0.00	-1.40	1.10	0.60
1	1	-1.40	1.10	0.60	0.30	1	1	0.00	-1.40	1.10	0.60
0	0	-1.40	0.00	0.00	-1.40	0	0	0.00	-1.40	1.10	0.60
0	1	-1.40	0.00	0.60	-0.80	0	0	0.00	-1.40	1.10	0.60
1	0	-1.40	1.10	0.00	-0.30	0	0	0.00	-1.40	1.10	0.60
1	1	-1.40	1.10	0.60	0.30	1	1	0.00	-1.40	1.10	0.60

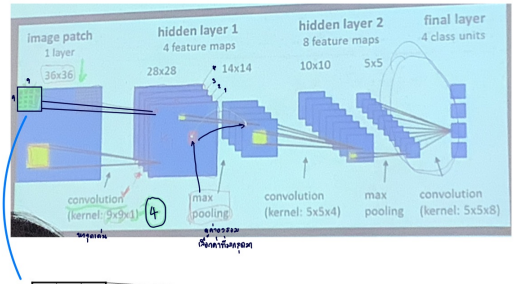
ขั้นตอนแรกเริ่มจากการสุ่มค่า w_0 จนถึง w_2 ในที่นี้กำหนดให้เป็น 0.1 ทั้งสามตัว จากนั้นก็เริ่มป้อนตัวอย่างเข้าไป (ทีละแถว) ตัวอย่างแรกได้ผลรวมเชิงเส้น (Net Sum) เป็น 0.10 ซึ่งมากกว่า 0 ดังนั้นเปอร์เซปตรอนจะให้เอาต์พุตจริง (Actual Output) ออกมาเป็น 1 ซึ่งผิด เพราะเอาต์พุตเป้าหมาย (Target Output) จะต้องได้เป็น 0 ทำให้อัตราการเรียนรู้คูณค่าผิดพลาด (Alpha x Error) ได้ –0.50 หลังจากนี้ก็นำไปปรับน้ำหนักตาม $w_i \leftarrow w_i + \Delta w_i$ และ $\Delta w_i \leftarrow \alpha(t-o)x_i$ ดังนั้นจะได้เป็น $w_0 \leftarrow w_0 + \alpha(t-o)x_0$ = w_0 + 0.50(-1) x 1 = 0.10 + (-0.5) = -0.4 ต่อไปก็ปรับค่า w_1 ในทำนองเดียวกัน $w_1 \leftarrow w_1 + \alpha(t-o)x_1$ = w_1 + 0.50(-1) x 0 ดังนั้น w_1 จะเท่ากับ 0.10 คือไม่เปลี่ยนแปลง เช่นเดียวกับ w_2 ที่ไม่เปลี่ยนแปลง จะเห็นได้ว่าแม้มีค่าผิดพลาดแต่ไม่มีการปรับค่า w_1 และ w_2 เนื่องจากอินพุตที่ใส่เข้าไปเป็น 0 ทำ



Fully Connected
Dense layer







Sobel filters

H11	H12	H13
H21	H22	H23
H31	H32	H33

X

F11	F12	F13	F14	F15	F16
F21	F22	F23	F24	F25	F26
F31	F32	F33	F34	F35	F36
F41	F42	F43	F44	F45	F46
F51	F52	F53	F54	F55	F56
F61	F62	F63	F64	F65	F66

=

G11	G12	G13	G14	G15	G16
G21	G22	G23	G24	G25	G26
G31	G32	G33	G34	G35	G36
G41	G42	G43	G44	G45	G46
G51	G52	G53	G54	G55	G56
G61	G62	G63	G64	G65	G66