

## สารบัญ

|  |          |
|--|----------|
| <b>บทที่ 1 บทนำ</b>  | <b>3</b> |
| 1.1 ความเป็นมาและความสำคัญของปัญหา . . . . .                                   | 3        |
| 1.2 วัตถุประสงค์ของการศึกษา . . . . .  | 3        |
| 1.3 ขอบเขตของการทำโครงการ . . . . .  | 3        |
| 1.4 ระยะเวลาและแผนดำเนินงาน . . . . .  | 4        |
| 1.5 ประโยชน์ที่คาดว่าจะได้รับ . . . . .  | 4        |
| 1.6 คำนิยามศัพท์เฉพาะ . . . . .  | 4        |
| <b>บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง</b>                                   | <b>6</b> |
| 2.1 จักรกลเรียนรู้ . . . . .   | 6        |
| 2.2 การเรียนรู้เชิงลึก . . . . .   | 6        |
| 2.2.1 เพอร์เซปตรอน (Perceptron) . . . . .                                      | 6        |
| 2.2.2 เพอร์เซปตรอนแบบหลายชั้น (Multi Layer Perceptron) . . . . .               | 8        |
| 2.3 ฟังก์ชันกระตุ้นและความฉลาดของโครงข่ายประสาทเทียม . . . . .                 | 10       |
| 2.3.1 ทฤษฎีบทตัวประมาณฟังก์ชันครอบคลุมจักรวาล . . . . .                        | 10       |
| 2.3.2 ข้อสังเกตต่อฟังก์ชันกระตุ้นและความฉลาด . . . . .                         | 10       |
| 2.4 โครงข่ายประสาทเทียมแบบสังวัตนาการ . . . . .                                | 10       |
| 2.5 ค่าสูญเสีย . . . . .   | 11       |
| 2.5.1 ค่าสูญเสียเมื่อมองจากมุมมองของตัวแปรเสริม . . . . .                      | 13       |
| 2.6 ขั้นตอนวิธีเกรเดียนต์ลดหลั่น และการก้าวเคลื่อนถอยหลัง . . . . .            | 13       |
| 2.6.1 ตัวดำเนินการเกรเดียนต์ . . . . .   | 13       |
| 2.6.2 ขั้นตอนวิธีก้าวเคลื่อนถอยหลัง . . . . .                                  | 14       |
| 2.7 สัญญาณรบกวน . . . . .  | 15       |
| 2.8 คำอธิบายต่อการเกิดขึ้นของสัญญาณรบกวน . . . . .                             | 15       |
| 2.8.1 การประพุดติตัวเป็นเส้นตรง . . . . .                                      | 15       |
| 2.8.2 ทฤษฎีชุดคุณสมบัติแบบอ่อนและแบบเข้ม . . . . .                             | 16       |
| 2.9 การคำนวณหาสัญญาณรบกวน . . . . .  | 17       |
| 2.9.1 การหาสัญญาณรบกวนด้วยวิธีการก้าวเคลื่อนถอยหลัง . . . . .                  | 17       |
| 2.9.2 การหาสัญญาณรบกวนด้วยวิธีการเครื่องหมายเกรเดียนต์แบบเร็ว (FGSM) . . . . . | 17       |

|                |  |           |
|----------------|--|-----------|
| 2.9.3          | การฉายเกรเดียนต์ลดหลั่น $k$ ชั้น ( $k$ -PGD) | 17        |
| 2.9.4          | ความต่างของขั้นตอนวิธี                       | 18        |
| 2.10           | การเรียนรู้เชิงโจมตี                         | 18        |
| <b>บทที่ 3</b> | <b>วิธีดำเนินงาน</b>                         | <b>19</b> |
| 3.1            | การวิเคราะห์และออกแบบขั้นตอนวิธี             | 19        |
| 3.1.1          | พฤติกรรมของคลัสเตอร์สัญญาณรบกวน              | 19        |
| 3.1.2          | การเสริมความแข็งแกร่งด้วยวิธีการผสมคลัสเตอร์ | 19        |
| 3.2            | การทดลองวัดประสิทธิภาพ                       | 19        |

## บทที่ 1

### บทนำ

#### 1.1 ความเป็นมาและความสำคัญของปัญหา

แบบจำลองจักรกลเรียนรู้ (Machine Learning models) นั้นถูกใช้อย่างกว้างขวางในปัจจุบัน อย่างไรก็ตาม แบบจำลองใดๆ นั้นอาจมีความผิดพลาดต่อการทำการโจมตีประสงค์ร้าย (Adversarial attacks) เพื่อจงใจให้ผลลัพธ์ที่แบบจำลองนั้นคาดเดามีความผิดพลาดจากผลลัพธ์ที่ควรจะเป็น

ในการเรียนรู้เชิงตัวแปรเสริม (parameter-based learning) นั้น ตัวแปรเสริม (parameters) ค่าน้ำหนัก (weights) บนแบบจำลองการเรียนรู้เชิงลึก (deep Learning models) เป็นตัวกำหนดความฉลาดของแบบจำลอง อาจมีตัวแปรเสริมบางชุดที่ทำให้แบบจำลองมีช่องโหว่ต่อการโจมตีประสงค์ร้าย การโจมตีนั้นอาจเกิดจากการเพิ่มสัญญาณรบกวนซึ่งผ่านการคำนวณ (calculated artefacts) เข้าสู่ข้อมูลรับเข้า (inputs) ซึ่งทำให้ความผิดพลาดของแบบจำลองในการพยากรณ์คำตอบนั้นเปลี่ยนไปอย่างชัดเจน

โครงการวิศวกรรมคอมพิวเตอร์นี้มุ่งหวังจะนำตัวแปรเสริมบนแบบจำลองมาสร้างภาพแสดง (visualise) ถึงจุดโหว่ในการพยากรณ์ใดๆ ของแบบจำลอง เพื่อลดความเสียหายอันอาจเกิดขึ้นได้จากการโจมตีแบบจำลองขณะถูกใช้งานจริง

#### 1.2 วัตถุประสงค์ของการศึกษา

โครงการนี้มีวัตถุประสงค์และเป้าหมายดังนี้

1. สร้างแบบจำลองเชิงลึก (Deep Learning models) ซึ่งสามารถถูกโจมตีประสงค์ร้าย (Adversarial attacks) ได้
2. นำแบบจำลองในข้อ (1) มาสร้างเป็นรูปภาพแสดง (visualisation) เพื่อหาจุดโหว่ต่อการโจมตี รวมถึงคาดเดาแนวโน้มการโจมตีที่เป็นไปได้
3. ใช้ความรู้ในข้อ (2) สร้างแบบจำลองที่ทนทาน (prone) ต่อการโจมตีมากขึ้น

#### 1.3 ขอบเขตของการทำโครงการ

โครงการนี้มีขอบเขตการดำเนินงานดังนี้

1. สร้างแบบจำลองเชิงลึก (Deep Learning models) ซึ่งสามารถถูกโจมตีประสงค์ร้าย (Adversarial attacks) ได้
2. นำแบบจำลองในข้อ (1) มาสร้างเป็นรูปภาพแสดง (visualisation) เพื่อหาจุดโหว่ต่อการโจมตี รวมถึงคาดเดาแนวโน้มการโจมตีที่เป็นไปได้
3. ใช้ความรู้ในข้อ (2) สร้างแบบจำลองที่ทนทาน (prone) ต่อการโจมตีมากขึ้น

#### 1.4 ระยะเวลาและแผนดำเนินงาน

การดำเนินงานของโครงการจะใช้วิธีจัดการงาน (workflow) แบบเอจิล (agile) เพื่อมุ่งเน้นประสิทธิภาพในการทำงานและสร้างระบบการทำงานที่เหมาะสมต่อการดำเนินการในระยะเวลาและปัจจัยการดำเนินงานที่ไม่อาจคาดเดาได้ การทำงานจะใช้วิธีการแบ่งรอบการวนทวน (iteration) โดยมุ่งเน้นให้แต่ละรอบการวนทวนมีความก้าวหน้าของงานในปริมาณที่เหมาะสมกับเวลาและข้อจำกัดต่างๆ หนึ่งรอบการวนทวนนั้นกินระยะเวลาสองสัปดาห์ดังแสดงในตารางที่ 1.1 และจะประกอบด้วยขั้นตอนต่อไปนี้

1. ประชุมสรุป (iteration meeting) หนึ่งถึงสองครั้งต่อสัปดาห์
2. เขียนรอบป้อนย้อนหลัง (backlog) และหยิบมาทำตามจำนวนที่เหมาะสม
3. กิจกรรมมองย้อนรอบการวนทวนด้วยตนเอง (self-retrospective) เพื่อพิจารณาความเหมาะสมในการทำงาน และปรับปรุงการทำงานในรอบการวนทวนต่อไป

อย่างไรก็ดี เพื่อเป็นการตั้งเป้าหมายงานในระยะยาว โครงการนี้จะมีวิสัยทัศน์ผลิตภัณฑ์ (product vision) โดยคร่าวตามขอบเขตการดำเนินงาน และในทุกประมาณ 4-6 รอบการวนทวน จะมีการวางแผนปล่อยผลิตภัณฑ์ (release planning) หนึ่งครั้งเพื่อสรุปงานออกมาเป็นความคืบหน้าที่ต้องได้อันเกิดจากการทำงานในกลุ่มรอบการวนทวนนั้น

#### 1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. เข้าใจถึงพื้นฐาน หลักการทำงาน และระบบจักรกลเรียนรู้แบบต่างๆ
2. เข้าใจถึงจุดอ่อนของระบบจักรกลเรียนรู้ในแต่ละกรณี
3. สามารถโจมตีระบบจักรกลเรียนรู้ เพื่อสร้างระบบจักรกลเรียนรู้ที่ทนทานต่อการโจมตีได้

#### 1.6 คำนิยามศัพท์เฉพาะ

- **จักรกลเรียนรู้ (machine learning)** คือระบบ หรือโค้ด หรือโปรแกรมคอมพิวเตอร์ที่เรียนรู้โครงสร้างของชุดคำถาม และคำตอบโดยมีจำเป็นต้องทำการโปรแกรมลำดับการทำงานอย่างชัดเจน (explicitly)
- **การเรียนรู้เชิงโจมตี (adversarial learning)** หมายถึงการศึกษาถึงการโจมตีแบบจำลอง (model) ของจักรกลเรียนรู้ (machine learning)

|                     | 2562 |      |      |      | 2563 |      |       |
|---------------------|------|------|------|------|------|------|-------|
|                     | ก.ย. | ต.ค. | พ.ย. | ธ.ค. | ม.ค. | ก.พ. | มี.ค. |
| รอบการรณทวนที่ 1    | /    |      |      |      |      |      |       |
| รอบการรณทวนที่ 2    | /    |      |      |      |      |      |       |
| รอบการรณทวนที่ 3    |      | /    |      |      |      |      |       |
| รอบการรณทวนที่ 4    |      | /    |      |      |      |      |       |
| รอบการรณทวนที่ 5    |      |      | /    |      |      |      |       |
| รอบการรณทวนที่ 6    |      |      | /    |      |      |      |       |
| แผนการปล่อยงานที่ 1 |      |      | /    |      |      |      |       |
| รอบการรณทวนที่ 7    |      |      |      | /    |      |      |       |
| รอบการรณทวนที่ 8    |      |      |      | /    |      |      |       |
| รอบการรณทวนที่ 9    |      |      |      |      | /    |      |       |
| รอบการรณทวนที่ 10   |      |      |      |      | /    |      |       |
| แผนการปล่อยงานที่ 2 |      |      |      |      | /    |      |       |
| รอบการรณทวนที่ 11   |      |      |      |      |      | /    |       |
| รอบการรณทวนที่ 12   |      |      |      |      |      | /    |       |
| รอบการรณทวนที่ 13   |      |      |      |      |      |      | /     |
| รอบการรณทวนที่ 14   |      |      |      |      |      |      | /     |
| แผนการปล่อยงานที่ 3 |      |      |      |      |      |      | /     |

ตารางที่ 1.1: ตารางแสดงรอบการรณทวนและช่วงเวลา

## บทที่ 2

### ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

#### 2.1 จักรกลเรียนรู้

ระบบจักรกลเรียนรู้ (machine learning) อาจนิยามได้ว่าเป็นระบบที่ไม่ต้องมีการป้อนข้อมูล หรือวิธีทำงาน เข้าไป ยังโค้ดโปรแกรมอย่างชัดเจน (explicitly) โดยระบบดังกล่าวจะถูกฝึกสอนด้วยชุดของข้อมูลหรือประสบการณ์ (experience) และปรับตัวเองให้ส่งออกคำตอบซึ่งอิงจากประสบการณ์ที่ตนเองเคยได้เรียนรู้มา

หากจะกล่าวให้ละเอียด เราสามารถนิยามโปรแกรมซึ่งสามารถทำการเรียนรู้ได้ดังนี้ [1]

**บทนิยาม 2.1.1.** โปรแกรมใดๆ เรียน (learn) จากประสบการณ์ (experience)  $E$  บนงาน (task)  $T$  และการวัดประสิทธิผล (performance measurement)  $P$  หากประสิทธิผลบน  $T$  ซึ่งถูกวัดโดย  $P$  เพิ่มขึ้นตามประสบการณ์  $E$

#### 2.2 การเรียนรู้เชิงลึก

การเรียนรู้เชิงลึก (Deep Learning) คือความพยายามในการจำลองเซลล์ประสาทของมนุษย์ให้อยู่ในรูปแบบจำลอง คณิตศาสตร์ ด้วยความเชื่อทางหลักประสาทวิทยา (neurosciences) ว่าความฉลาดของสมองมนุษย์เกิดขึ้นได้จากโครงข่าย ประสาทจำนวนมาก ที่เชื่อมเข้าถึงกัน [2]

##### 2.2.1 เพอร์เซปตรอน (Perceptron)

เพอร์เซปตรอน (Perceptron) [3] เป็นแบบจำลองทางคณิตศาสตร์ของเซลล์สมองหนึ่งเซลล์ โดยมีคุณสมบัติดังนี้

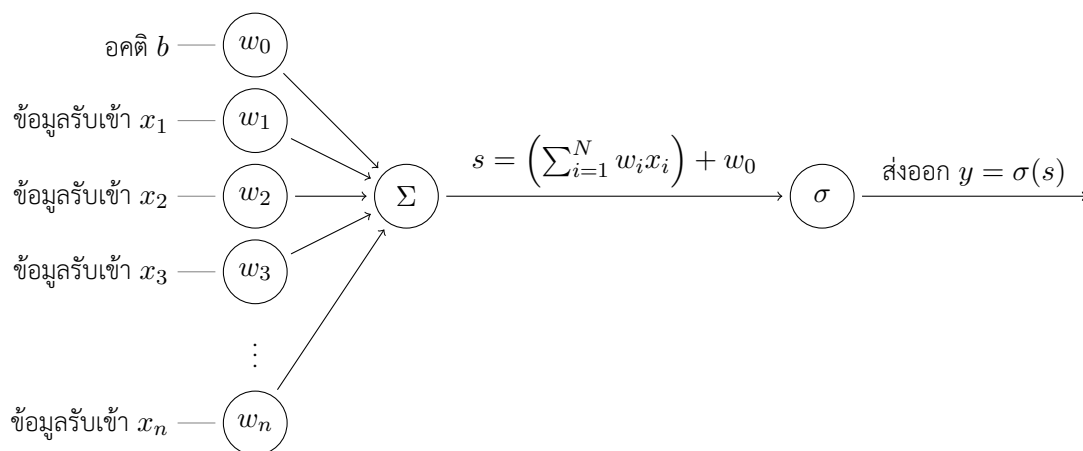
- รับเข้าข้อมูลมาในเซลล์จากหลายแหล่ง และให้น้ำหนักกับข้อมูลนั้นต่างกันไป
- ส่งออกข้อมูลเพียงค่าเดียว

ดังนั้น แบบจำลองทางคณิตศาสตร์สามารถเขียนออกมาจากหลักการสองข้อดังกล่าวได้ด้วยสมการ

$$y = \sigma(W^T X + b) \quad (2.1)$$

เมื่อ  $W$  และ  $X$  เป็นเมทริกซ์ขนาด  $1 \times n$  (โดย  $n$  เป็นจำนวนข้อมูลรับเข้า),  $b$  เป็นค่าสัมประสิทธิ์คงที่ (อคติ: bias) และ  $\sigma$  เป็นฟังก์ชันกระตุ้น (activation function) ซึ่งอาจเขียนรูปร่างของเพอร์เซปตรอนให้มีลักษณะรูปคล้ายเซลล์ สมองได้ในลักษณะรูปที่ 2.1

ยกตัวอย่างการใช้เพอร์เซปตรอนในการแก้ปัญหาอย่างง่ายได้ในที่นี้



รูปที่ 2.1: เพอร์เซปตรอน

### การคาดการณ์ราคาส่งหาอสังหาริมทรัพย์

หากสำรวจราคาอสังหาริมทรัพย์แล้วพบว่า

- ราคาอสังหาริมทรัพย์จะเพิ่มขึ้นตามที่ดิน โดยเพิ่มขึ้นทุก 10,000 บาทต่อตารางวา
- ราคาอสังหาริมทรัพย์จะเพิ่มขึ้นตามจำนวนห้องนอน โดยเพิ่มขึ้นทุก 200,000 บาทต่อห้องนอน
- ราคาอสังหาริมทรัพย์จะลดลงตามจำนวนอายุปี โดยลดลงทุก 7,000 บาทต่ออายุของอสังหาริมทรัพย์
- ราคาที่กำหนดจริง (fixed cost) ของอสังหาริมทรัพย์ อยู่ที่ 100,000 บาท

จะสามารถเขียนเพอร์เซปตรอนเพื่อคาดการณ์ราคาส่งหาอสังหาริมทรัพย์ได้โดย

$$y = \sigma(W^T X)$$

เมื่อ  $W$  ซึ่งเป็นค่าสัมประสิทธิ์แสดงถึงความสัมพันธ์ข้อมูลรับเข้า ซึ่งเขียนได้จากความสัมพันธ์ดังแสดงด้านล่าง

$$W^T = \begin{bmatrix} 100000 & 10000 & 200000 & -7000 \end{bmatrix}$$

หากต้องการคาดการณ์ราคาบ้านที่มี 3 ห้องนอน เนื้อที่ 100 ตารางวา และมีอายุ 7 ปี จะสามารถเขียนเมทริกซ์  $X$  ได้เป็น

$$X = \begin{bmatrix} 1 \\ 3 \\ 100 \\ 7 \end{bmatrix}$$

โปรดสังเกตว่า  $x_0 = 1$  เนื่องจากผลคูณของเทอม  $w_0$  และ  $x_0$  เป็นค่าที่เรียกว่าค่าอคติ (bias) ของแบบจำลอง

เนื่องจากเพอร์เซปตรอนตัวนี้ถูกใช้ในการทำนายราคา ซึ่งกล่าวว่ามีความสัมพันธ์กันกับตัวแปรที่กำหนดข้างต้นในเชิงเส้น ดังนั้นจะกล่าวได้ว่าฟังก์ชันกระตุ้น (activation function) ที่เลือกใช้ จะเลือกใช้ฟังก์ชันเส้นตรง (linear function)  $\sigma(x) = x$

ดังนั้น ผลการทำนายราคาบ้านคำนวณได้จาก

$$\begin{aligned}
 y &= \sigma(W^T X + b) \\
 &= \sigma \left( \begin{bmatrix} 100000 & 10000 & 200000 & -7000 \end{bmatrix} \times \begin{bmatrix} 1 \\ 3 \\ 100 \\ 7 \end{bmatrix} \right) \\
 &= \sigma(100000 + 30000 + 20000000 + (-49000)) = f(20981000) \\
 &= 20981000
 \end{aligned}$$

### การสร้างประตูลัญญานตรกะด้วยเปอร์เซปตรอน

เราสามารถสร้างประตูลัญญานตรกะ (logic gates) บางชนิดได้ด้วยเปอร์เซปตรอน เช่นการสร้าง AND และ OR gate

ยกตัวอย่างโครงสร้างของประตูลัญญานตรกะและซึ่งสามารถสร้างได้ด้วยการกำหนดให้

- $X$  เป็นเมทริกซ์ขนาด  $1 \times 2$  กล่าวคือเมื่อรับค่า  $x_1, x_2$  เป็นค่า 0 หรือ 1 แทนสัญญาณจริงหรือเท็จแล้ว

$$X = \begin{bmatrix} 1 \\ a_1 \\ a_2 \end{bmatrix}$$

- กำหนดค่าของเมทริกซ์  $W$  เป็น

$$W^T = \begin{bmatrix} -2 & 1 & 1 \end{bmatrix}$$

- กำหนดฟังก์ชัน  $\sigma(x)$  เป็น step function กล่าวคือ

$$\sigma(x) = \begin{cases} 1; & x \geq 0 \\ 0; & \text{ในกรณีอื่น} \end{cases}$$

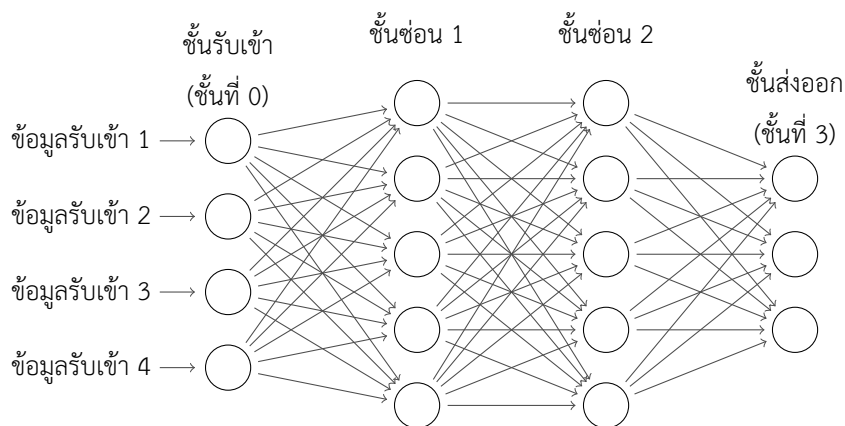
และการสร้างประตูลัญญานตรกะหรือสามารถทำได้ในลักษณะเดียวกันโดยเปลี่ยนชุดน้ำหนัก เป็น

$$W^T = \begin{bmatrix} -1 & 1 & 1 \end{bmatrix}$$

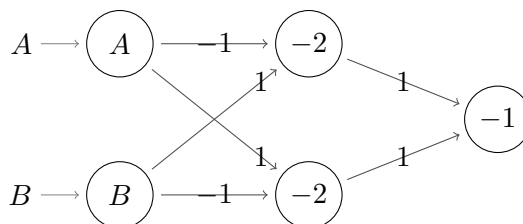
### 2.2.2 เปอร์เซปตรอนแบบหลายชั้น (Multi Layer Perceptron)

เราอาจสังเกตว่าเปอร์เซปตรอนหนึ่งตัวนั้นทำหน้าที่ได้เพียงแยก (classify) หรือถดถอย (regress) ปัญหาที่เป็นปัญหาเชิงเส้น (linear problems) ได้เท่านั้น อย่างไรก็ตามหากเรากำหนดให้ฟังก์ชัน  $f$  เป็นฟังก์ชันที่ไม่ใช่ฟังก์ชันเส้นตรงแล้ว เราอาจสร้างเปอร์เซปตรอนแบบหลายชั้น (Multi Layer Perceptron) ขึ้นมาได้โดยมีลักษณะดังรูปที่ 2.2





รูปที่ 2.2: เพอร์เซปตรอนแบบหลายชั้น



รูปที่ 2.3: เพอร์เซปตรอนแบบหลายชั้นซึ่งทำหน้าที่เป็นประตูสัญญาณเฉพาะหรือ

เราอาจเขียนแทนน้ำหนักของโครงข่ายจากเพอร์เซปตรอนชั้นที่  $i$  ไปยังชั้นที่  $j$  ( $j = i + 1$ ) ได้เป็น

$$\mathbf{W}_{ij} = \begin{bmatrix} w_{10} & w_{20} & \dots & w_{n_i 0} \\ w_{11} & w_{21} & \dots & w_{n_i 1} \\ w_{12} & w_{22} & \dots & w_{n_i 2} \\ \vdots & \vdots & \ddots & \vdots \\ w_{1n_j} & w_{2n_j} & \dots & w_{n_j n_i} \end{bmatrix}$$

เมื่อจำนวนเพอร์เซปตรอนในชั้นที่  $k$  เขียนแทนด้วย  $n_k$

ยกตัวอย่างเช่น เราจะสามารถสร้างประตูสัญญาณเฉพาะหรือ (XOR gate) ได้จากเพอร์เซปตรอนแบบหลายชั้น ดังแสดงในรูปที่ 2.3 โดยเลขในแต่ละเพอร์เซปตรอนแทนค่าอคติ ( $b$ ) และเลขบนเส้นเชื่อมแทนค่าน้ำหนัก ( $w$ ) และกำหนดให้ฟังก์ชันกระตุ้น  $\sigma$  เป็นฟังก์ชันขั้นบันได (step function) กล่าวคือ

$$\sigma(x) = \begin{cases} 1; & x \geq 0 \\ 0; & \text{ในกรณีอื่น} \end{cases}$$

เพอร์เซปตรอนดังกล่าว เมื่อรับค่า  $A$  และ  $B$  เป็น 0 หรือ 1 จะส่งออกมา  $A \oplus B$

## 2.3 ฟังก์ชันกระตุ้นและความฉลาดของโครงข่ายประสาทเทียม

### 2.3.1 ทฤษฎีบทตัวประมาณฟังก์ชันครอบจักรวาล

เหตุผลที่โครงข่ายประสาทเทียมสามารถทำงานได้ดี เนื่องจากมีการพิสูจน์ว่าโครงข่ายประสาทเทียมนั้นสามารถทำหน้าที่เป็นตัวประมาณฟังก์ชันครอบจักรวาล [4] (universal function approximator) กล่าวคือโครงข่ายประสาทเทียม  $N : \mathbb{R}^k \rightarrow \mathbb{R}^n$  ที่มีความซับซ้อนมากเพียงพอ (ซึ่งจะกล่าวถึงความซับซ้อนนี้ในภายหลัง) สามารถที่จะจำลองฟังก์ชัน  $f : \mathbb{R}^k \rightarrow \mathbb{R}^n$  (กล่าวคือฟังก์ชันที่มีโดเมน และเรนจ์ เป็นจำนวนจริงใดๆ ในมิติที่เหมือนกับมิติข้อมูลรับเข้าและข้อมูลส่งออกของโครงข่ายประสาทเทียม  $N$ ) ได้ [5] [6] [7]

บทพิสูจน์ของทฤษฎีบทนี้ในรูปแบบของกรณีไม่ตีกรอบความกว้าง (unbounded width case) และกรณีตีกรอบความกว้าง (bounded width case) สามารถศึกษาได้จากแหล่งอ้างอิง รวมถึงแหล่งอ้างอิงเพิ่มเติมที่ใช้การแสดงทัศนภาพ (visualisation) เพื่อการพิสูจน์ทฤษฎีบทดังกล่าว [8]

### 2.3.2 ข้อสังเกตต่อฟังก์ชันกระตุ้นและความฉลาด

บทพิสูจน์ที่ได้กล่าวถึงไปก่อนหน้านี้สำหรับกรณีไม่ตีกรอบความกว้าง และตีกรอบความกว้าง เป็นบทพิสูจน์ที่ใช้ฟังก์ชันกระตุ้นเป็นฟังก์ชันซิกมอยด์ (sigmoid) และฟังก์ชันรีลู (ReLU) ตามลำดับ

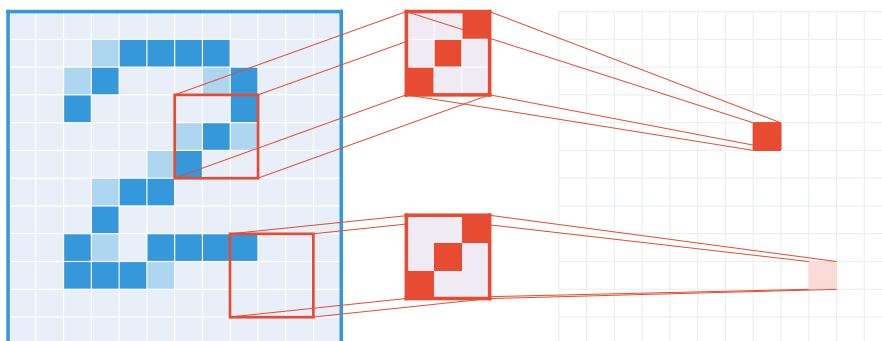
อย่างไรก็ดี หากพิจารณาโครงข่ายประสาทเทียมใดๆ ที่ใช้ฟังก์ชันกระตุ้นเป็นฟังก์ชันเชิงเส้น  $f(x) = x$  เราจะพบว่าโครงข่ายประสาทเทียมใดๆ จะสามารถยุบให้อยู่ในรูปของเปอร์เซปตรอนเพียงตัวเดียว และทำให้ไม่สามารถตัดสินใจปัญหาได้มากกว่าปัญหาที่แบ่งแยกเชิงเส้นได้ (linearly separable problems)

ดังนั้น อาจกล่าวด้วยการพิจารณา (intuition) ในลักษณะดังกล่าวได้ว่า ส่วนหนึ่งของความเป็นไปได้ของการที่โครงข่ายประสาทเทียมใดๆ สามารถทำหน้าที่เป็นตัวประมาณฟังก์ชันครอบจักรวาลได้ ส่วนหนึ่งมาจากการที่ฟังก์ชันกระตุ้นทำหน้าที่เป็นตัวบีบ (squeezer) ช่วงของข้อมูลรับเข้าบนโดเมนจำนวนจริงใดๆ ( $\mathbb{R}$ ) ให้กลายเป็นช่วงจำกัดช่วงอื่น (เช่น ช่วง  $(0, 1)$  ของฟังก์ชันซิกมอยด์ หรือช่วง  $[0, \infty)$  ของฟังก์ชันรีลู)

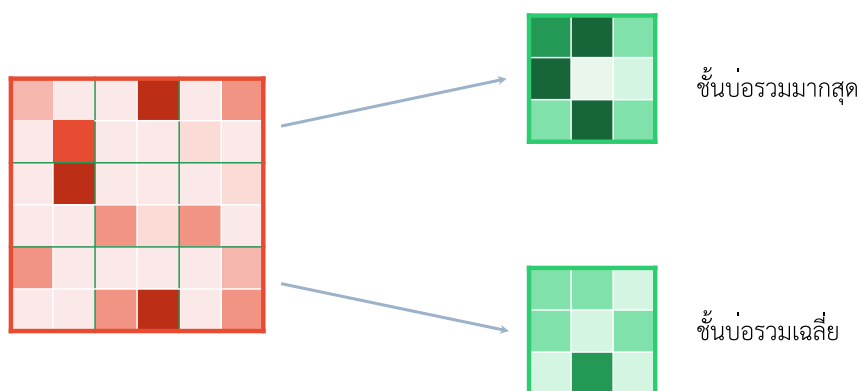
## 2.4 โครงข่ายประสาทเทียมแบบสังวัฒนาการ

โครงข่ายประสาทเทียมแบบสังวัฒนาการ (Convolutional Neural Networks: CNN) [9] เป็นโครงข่ายประสาทเทียมซึ่งมักถูกใช้กับข้อมูลภาพ [10] โดยคร่าวแล้วโครงข่ายประสาทเทียมในลักษณะดังกล่าวมักประกอบด้วยชั้นประสาทเทียมในลักษณะดังนี้

- **ชั้นสังวัฒนาการ** (convolution layer) เป็นชั้นที่กระทำตัวดำเนินการสังวัฒนาการ (convolve) ตัวกรอง (filter)  $F$  บนข้อมูลนำเข้า  $I$  ด้วยระยะเคลื่อน (stride)  $S$  ผลลัพธ์จากการสังวัฒนาการนี้จะเรียกว่าแผนที่ลักษณะ (feature map) ยกตัวอย่างการสังวัฒนาการเพื่อหาเส้นเฉียงในรูปที่ 2.4 สังเกตว่าการสังวัฒนาการด้วยตัวกรองเส้นเฉียงบนเส้นเฉียงบริเวณข้อมูลนำเข้า จะให้ค่าส่งออกที่มากกว่าการสังวัฒนาการตัวกรองเส้นเฉียงบนจุดพื้นที่อื่นของข้อมูลนำเข้า (ในที่นี้เขียนแทนด้วยสีแดงเข้ม และสีแดงอ่อน)
- **ชั้นบ่อรวม** (pooling layer) เป็นชั้นที่ทำการสุ่มตัวอย่างแบบลดขนาด (downsampling) เพื่อลดขนาดของข้อมูล ในขณะที่ยังคงไว้ซึ่งชุดคุณสมบัติที่ข้อมูลรับเข้ามี ชั้นบ่อรวมอาจแบ่งเป็นสองประเภทหลัก
  - **ชั้นบ่อรวมแบบมากที่สุด** (maximum pooling layer) เป็นชั้นบ่อรวมที่พบได้บ่อยที่สุด



รูปที่ 2.4: ชั้นสังวัตนาการ ซึ่งแสดงข้อมูลนำเข้าด้วยสีฟ้า และตัวกรองด้วยสีแดง



รูปที่ 2.5: ชั้นบ่อรวม ทั้งแบบบ่อรวมมากที่สุดและแบบบ่อรวมเฉลี่ย โดยพิจารณาบ่อตามขอบเขตสี่เหลี่ยม

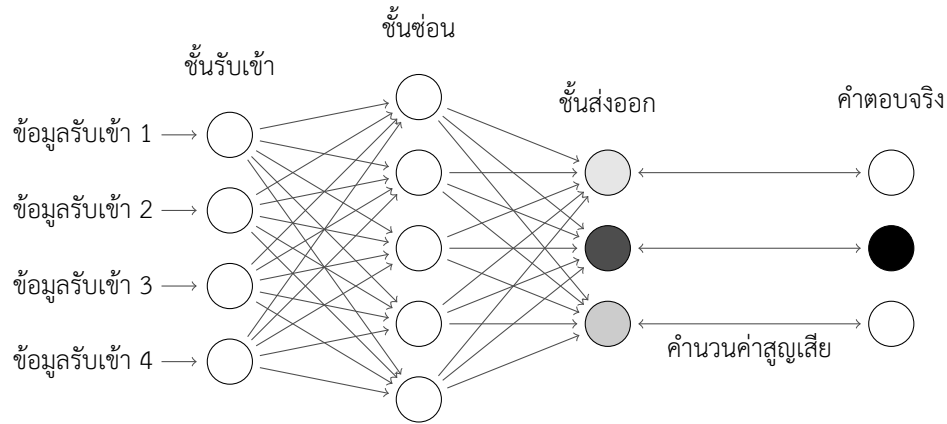
- **ชั้นบ่อรวมแบบเฉลี่ย** (average pooling layer) เป็นชั้นบ่อรวมที่พบในโครงข่ายประสาทเทียมแบบสังวัตนาการบางรูปแบบ เช่น LeNet
- **ชั้นเชื่อมต่อถึงกันหมด** (fully connected layer) ซึ่งมีลักษณะเหมือนเปอร์เซปตรอนแบบหลายชั้นทั่วไป

การสังวัตนาการของชั้นสังวัตนาการในโครงข่ายประสาทเทียม ทำหน้าที่เป็นตัวตรวจจับคุณสมบัติ (feature detector) เช่นการตรวจจับขอบ (edge detection) และชั้นบ่อรวมทำให้ขนาดของผลลัพธ์จากชั้นสังวัตนาการมีขนาดเล็กลง เพื่อให้จำนวนค่าน้ำหนักของโครงข่ายประสาทเทียมที่ต้องคำนวณนั้นน้อยลง

การเรียนรู้ด้วยวิธีก้าวเคลื่อนถอยหลัง (backpropagation learning) เป็นวิธีการเรียนรู้ที่ได้รับความนิยมมากที่สุด ในปัจจุบัน ทั้งนี้ เราอาจพิจารณาการเรียนรู้ถอยหลังได้โดยทำความเข้าใจถึงฟังก์ชันสูญเสีย (loss function) และการปรับค่าตัวแปรเสริม (parameters) ดังนี้

## 2.5 ค่าสูญเสีย

ค่าสูญเสีย (loss) เป็นค่าที่ใช้ในการบอกว่าแบบจำลองใดๆ ตอบผิดมากหรือน้อยเพียงใด โดยค่าสูญเสียนั้นหมายถึงแบบจำลองตอบผิดมากเท่านั้น



รูปที่ 2.6: การคำนวณค่าสูญเสีย

ยกตัวอย่างเช่น หากเราสร้างแบบจำลองที่ต้องการส่งออกค่าเป็นค่าในลักษณะของการเข้ารหัสแบบหนึ่งจุดร้อน (one-hot encoding) ของค่าที่เป็นไปได้ 3 ชั้น (classes) จากข้อมูลตัวที่  $i$  บนชุดฝึกหัด ดังแสดงในรูปที่ 2.6 ซึ่งต้องการคำตอบ  $t_i$  ที่ถูกต้องเป็น

$$t_i = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$$

ทว่า แบบจำลองกลับให้คำตอบ  $o_i$  จากแบบจำลองเป็น

$$o_i = \begin{bmatrix} 0.1 & 0.7 & 0.2 \end{bmatrix}$$

เราอาจนิยามฟังก์ชันสูญเสียอย่างง่าย เพื่อยกตัวอย่างการคำนวณดังกล่าว โดยกำหนดให้ฟังก์ชันสูญเสียเป็นผลรวมของผลต่างกำลังสอง

$$l_i(t_i, o_i) = \sum_{j=1}^n (t_i[j] - o_i[j])^2$$

ดังนั้น ในกรณีนี้ จะได้ค่าสูญเสียของจุดฝึกหัดนี้เป็น

$$\begin{aligned} l_i(t_i, o_i) &= \sum_{j=1}^n (t_i[j] - o_i[j])^2 \\ &= ((0 - 0.1)^2 + (1 - 0.7)^2 + (0 - 0.2)^2) \end{aligned}$$

จะเห็นว่า ยิ่งค่า  $t_i$  ใกล้เคียง  $o_i$  มากขึ้นเท่าใด ค่าสูญเสียก็จะน้อยลงเท่านั้น

นอกจากนี้ เราจะนิยามค่าสูญเสียบนชุดฝึกหัดทั้งชุด เป็น

$$\mathcal{L}(T, O) = \sum_{i=1}^N l_i(t_i, o_i) \quad (2.2)$$

เมื่อ  $T$  และ  $O$  เป็นชุดคำตอบ และค่าส่งออกจากแบบจำลองของทั้งชุดฝึกหัด ซึ่งชุดฝึกหัดมีความยาวเป็น  $N$

อย่างไรก็ตาม ฟังก์ชันสูญเสียในลักษณะดังกล่าว เป็นฟังก์ชันอย่างง่าย ในการฝึกสอนแบบจำลองทั่วไปมักนิยมใช้ฟังก์ชันอื่น เช่น ค่าสูญเสียแบบความวุ่นวายข้ามชั้น (cross entropy loss) สำหรับการฝึกสอนแบบจำลองเพื่อการทำการจำแนกหมวดหมู่ (classification)

$$\ell_i = - \sum_{c=1}^M y_{o,c} \ln(p_{o,c})$$

เมื่อ  $M$  เป็นจำนวนชั้น (class) ที่เป็นไปได้  $y$  เป็นค่าฐานสองที่บ่งบอกว่าชั้นข้อมูล (class)  $c$  เป็นคำตอบที่ถูกต้องสำหรับการคาดเดา (observation)  $o$  และ  $p$  เป็นค่าความน่าจะเป็นที่การคาดเดา  $o$  ตอบว่าเป็นชั้นข้อมูล  $c$

### 2.5.1 ค่าสูญเสียเมื่อมองจากมุมมองของตัวแปรเสริม

สมการที่นำเสนอไปข้างต้น มองค่าสูญเสียเปลี่ยนไป เมื่อใส่ชุดของข้อมูลส่งออกจากแบบจำลอง  $O$  และค่าคำตอบจริง  $T$  ต่างกันออกไป ทว่า หากพิจารณาว่า

- แบบจำลองใดๆ สามารถปรับค่าตัวแปรเสริม (parameters) ได้อย่างอิสระ
- ค่าส่งออก  $O$  เป็นฟังก์ชันของค่ารับเข้า  $I$  โดย  $O = f(I)$  เมื่อ  $f$  เป็นฟังก์ชันของโครงข่ายประสาทเทียม
- ความมุ่งหมายฝึกสอนแบบจำลองใดๆ ให้มีประสิทธิภาพ อยู่บนการฝึกสอนบนชุดของค่าคำตอบจริง  $T$  เดิม

เราจะสามารถมองฟังก์ชันสูญเสีย เป็นฟังก์ชันที่รับค่าตัวแปรเสริม (กล่าวคือค่าน้ำหนักและอคติของแบบจำลอง) และส่งออกค่าสูญเสียของชุดตัวแปรเสริมนั้น

กล่าวอีกนัย หากเรามีชุดของตัวแปรเสริม  $\theta_1, \theta_2, \dots, \theta_i$  บนโครงสร้างของแบบจำลองการเรียนรู้เชิงลึก (deep learning models) ที่มีโครงสร้างเหมือนกัน เราอาจพิจารณาค่าฟังก์ชันสูญเสีย  $\mathcal{L}(\theta_i)$  บนชุดตัวแปรเสริม  $\theta_i$  และกล่าวว่าแบบจำลองที่ใช้ชุดตัวแปรเสริม  $\theta_i$  นั้นทำงานได้ดีกว่า  $\theta_j$  หาก  $\mathcal{L}(\theta_i) < \mathcal{L}(\theta_j)$

## 2.6 ขั้นตอนวิธีเกรเดียนต์ลดหลั่น และการก้าวเคลื่อนถอยหลัง

### 2.6.1 ตัวดำเนินการเกรเดียนต์

พิจารณาตัวดำเนินการเกรเดียนต์ ซึ่งดำเนินการบนเวกเตอร์ใดๆ

**บทนิยาม 2.6.1.** เกรเดียนต์ ของฟังก์ชัน  $f(\vec{x})$  ซึ่งเป็นฟังก์ชันของชุดตัวแปร  $\vec{x} = (x_1, x_2, \dots, x_n)$  ใดๆ จะถูกนิยามเป็นตัวดำเนินการ  $\vec{\nabla} f(\vec{x})$

$$\vec{\nabla} f(\vec{x}) = \sum_{i=1}^n \frac{\partial}{\partial x_i} (f(\vec{x}) \cdot \hat{x}_i)$$

เมื่อ  $\hat{x}_i$  เป็นเวกเตอร์หนึ่งหน่วย (unit vector) ตามแกนของ  $x_i$

กรุณาสังเกตว่า ทิศทางของเกรเดียนต์นั้นจะชี้ไปในทิศทางที่เพิ่มขึ้นของฟังก์ชันเสมอ

เมื่อพิจารณาฟังก์ชันสูญเสีย ซึ่งเป็นฟังก์ชันที่เราต้องการลดค่า เราอาจพิจารณาหาค่าที่น้อยลงของฟังก์ชันได้ ด้วยการคำนวณเกรเดียนต์ของฟังก์ชันสูญเสียใดๆ แล้ว “เดิน” ไปในทิศทางตรงข้ามกับเกรเดียนต์ เปรียบประหนึ่งการเดินทาง

เขา ขั้นตอนวิธีดังกล่าวเรียกว่าขั้นตอนวิธีเกรเดียนต์ดลัน (gradient descent algorithm) โดยพิจารณาการปรับแบบจำลองอยู่บนเกรเดียนต์ของฟังก์ชันสูญเสีย

$$\vec{\theta}' = \vec{\theta} - l \vec{\nabla} \mathcal{L}(\vec{\theta}) \quad (2.3)$$

เมื่อ  $l$  เป็นค่าอัตราการเรียนรู้ (learning rate) โดยปกติมักมีค่าไม่มาก

---

**ขั้นตอนวิธี 1** ขั้นตอนวิธีเกรเดียนต์ดลันเพื่อการฝึกสอนแบบจำลอง

---

ประกาศตัวแปร  $\hat{\theta}$  เป็นค่าสุ่มของเวกเตอร์ความยาวเท่าตัวแปรเสริมที่แบบจำลอง  $M$  ต้องการ

สำหรับ  $i = 0$  ถึง  $N$  ทำ

    คำนวณเกรเดียนต์ของ  $\vec{\theta}$  โดย

$$\vec{\nabla} \mathcal{L}(\vec{\theta})$$

    ปรับค่า  $\vec{\theta}$  โดย

$$\vec{\theta}' = \vec{\theta} - l \vec{\nabla} \mathcal{L}(\vec{\theta})$$

จบการวนสำหรับ

    ส่งคืนค่า  $\vec{\theta}$

---

รหัสเทียมของขั้นตอนวิธีดังกล่าวสามารถศึกษาได้จากขั้นตอนวิธี 1

หากอธิบายโดยคร่าว ขั้นตอนวิธีเกรเดียนต์ดลัน พยายามหาค่าตัวแปรเสริม  $\theta_{\text{opt}}$  โดยการเริ่มจากการสุ่มตัวแปรเสริม  $\theta$  แล้วคำนวณเกรเดียนต์ของฟังก์ชันสูญเสีย และค่อยๆ ปรับค่า  $\theta$  ตามทิศทางข้ามกับเกรเดียนต์เรื่อยๆ จนกระทั่งถึงจุดที่ฟังก์ชันสูญเสียน้อยที่สุด

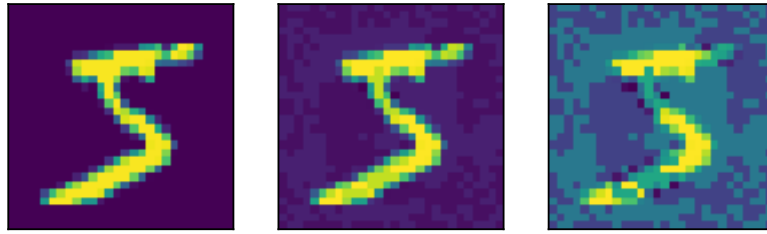
## 2.6.2 ขั้นตอนวิธีกาวเคลื่อนถอยหลัง

หนึ่งในประเด็นสำคัญที่จำเป็นต้องกล่าวถึงต่อมา คือวิธีการในการคำนวณเกรเดียนต์  $\vec{\nabla}(\vec{x})$  ซึ่งจำเป็นต้องคำนวณอนุพันธ์ (derivation) ของตัวแปรเสริมใดๆ เทียบกับฟังก์ชันสูญเสีย พึงพิจารณาว่าตัวแปรเสริมของแบบจำลองการเรียนรู้เชิงลึก คือชุดน้ำหนักและค่าติดต่างๆ

อย่างไรก็ตาม การคำนวณเกรเดียนต์และอนุพันธ์ดังกล่าวสามารถทำได้โดยง่าย ผ่านการใช้กฎลูกโซ่ (chain rule) ซึ่งนิยามได้ว่า

$$\frac{dy}{dx} = \frac{dy}{du} \cdot \frac{du}{dx} \quad (2.4)$$

ในเล่มรายงานนี้จะไม่ลงรายละเอียดถึงขั้นตอนวิธีกาวเคลื่อนถอยหลัง อย่างไรก็ตาม สามารถพิจารณาได้ว่า (1) อนุพันธ์ของฟังก์ชันสูญเสียกับน้ำหนักของชั้นส่งออกสามารถพิจารณาได้โดยตรง และ (2) อนุพันธ์ของฟังก์ชันสูญเสียกับน้ำหนักของชั้นซ่อนใดๆ สามารถพิจารณาได้จากกฎลูกโซ่ดังสมการที่ 2.4 เป็นผลคูณของอนุพันธ์ของฟังก์ชันสูญเสียเทียบกับน้ำหนักของชั้นส่งออก และอนุพันธ์ของน้ำหนักชั้นส่งออกเทียบกับชั้นซ่อน



รูปที่ 2.7: (จากซ้ายไปขวา) รูปเลข 5 ที่ถูกเจือด้วยความเข้มข้นของสัญญาณรบกวนที่ต่างกันที่ระดับ 0.0, 0.01 และ 0.03 ตามลำดับ

## 2.7 สัญญาณรบกวน

การโจมตีแบบจำลองการเรียนรู้เชิงลึกด้วยการหาสัญญาณรบกวน  $\eta$  ที่โจมตีแบบจำลองการเรียนรู้  $M$  นั้นมีจุดประสงค์หลักคือหาค่า  $\eta$  ซึ่งอยู่บนโดเมนเดียวกับข้อมูลรับเข้าโครงข่ายประสาทเทียม ที่ทำการเพิ่มค่าของฟังก์ชันสูญเสียจนถึงขีดสุด

เราจะเรียก  $x' = x + \eta$  ว่าเป็นตัวอย่างประสงค์ร้าย (adversarial example) เนื่องจากเป็นตัวอย่างที่ทำให้ค่าสูญเสียของโครงข่ายประสาทเทียมเพิ่มสูงขึ้นที่สุด

การหาสัญญาณรบกวนอาจมองเป็นปัญหาการเพิ่มประสิทธิภาพสูงสุด (optimisation problem) โดยพิจารณาการหา

$$\eta = \text{argmax}_{\eta} \mathcal{L}(x + \eta, y) \quad (2.5)$$

เมื่อคู่อันดับ  $(x, y)$  แทนที่ตำแหน่งของชุดข้อมูลฝึกหัด (training point) หนึ่งจุด

ทั้งนี้ทั้งนั้น ขนาด (norm) ของ  $\eta$  จะต้องมิต้านน้อยเพียงพอเมื่อเทียบกับ  $x$  เพื่อทำให้ความเข้มของสัญญาณรบกวนไม่เข้มจนเกินไปจนสามารถแยกแยะด้วยตาของมนุษย์ได้ว่าเป็นภาพที่ถูกเจือด้วยสัญญาณรบกวน ดังเช่นแสดงในรูปที่ 2.7

## 2.8 คำอธิบายต่อการเกิดขึ้นของสัญญาณรบกวน

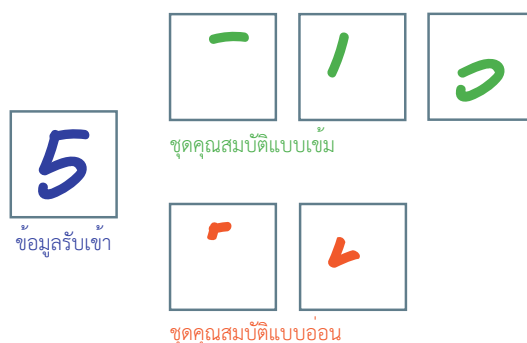
มีหลายทฤษฎีพยายามอธิบายการเกิดขึ้นของการโจมตีแบบจำลอง ซึ่งอาจยกตัวอย่างทฤษฎีและคำอธิบายได้ดังนี้

### 2.8.1 การประพจน์ตัวเป็นเส้นตรง

LeCun และคณะ [11] ศึกษาผลของการโจมตีที่เกิดขึ้นจาก  $\tilde{x}$  โดยอาจพิจารณาได้จากการคุณสมบัติการหาค่าส่งออกจากชุดน้ำหนัก (weights) ของชั้นแบบจำลองการเรียนรู้เชิงลึก (deep learning layers)

$$w^T \tilde{x} = w^T x + w^T \eta \quad (2.6)$$

คณะวิจัยสังเกตพฤติกรรมว่าสัญญาณรบกวน  $\eta$  กระตุ้นส่วนของชุดน้ำหนักและฟังก์ชันกระตุ้น (activation function) ในแบบจำลองให้ประพจน์ตัวเยี่ยงฟังก์ชันเส้นตรง (linear functions) ซึ่งการแสดงพฤติกรรมดังเส้นตรง (linearity) ในกรณีชายขอบ (edge case) ของข้อมูลรับเข้านั้นก่อให้เกิดความเป็นไปได้ที่แบบจำลองจะถูกโจมตี



รูปที่ 2.8: ตัวอย่างชุดคุณสมบัติแบบอ่อน และแบบเข้มที่เป็นไปได้ จากเลข 5

เพื่อพิสูจน์ทฤษฎีดังกล่าว LeCun และคณะ พิจารณาผลความน่าจะเป็นของคำตอบที่ออกจากแบบจำลองเมื่อปรับค่า  $\epsilon$  ดังแสดงในสมการที่ 2.5 และพบว่าความน่าจะเป็นของข้อมูลส่งออก (output) ของแต่ละชั้นข้อมูล (class) มีความสัมพันธ์เชิงเส้นตรงกับค่า  $\epsilon$  ที่เพิ่มขึ้นเรื่อยๆ

### 2.8.2 ทฤษฎีชุดคุณสมบัติแบบอ่อนและแบบเข้ม

Ilyas และคณะ [12] ศึกษาโครงสร้างของแบบจำลองเชิงลึก จนนำมาสู่ข้อสรุปว่า “ช่องโหว่ในการโจมตีแบบจำลองเป็นผลโดยตรงจากความอ่อนไหวของแบบจำลองในการวางหลักการบนชุดคุณสมบัติของข้อมูล” (“Adversarial vulnerability is a direct result of our models’ sensitivity to well-generalizing features in the data”)

หากกล่าวให้ละเอียด พิจารณาว่าโครงสร้างของแบบจำลองเชิงลึกสามารถเรียนรู้ชุดคุณสมบัติ (features) ของข้อมูลรับเข้าได้สองแบบ ซึ่งในงานวิจัยเรียกว่าชุดคุณสมบัติแบบอ่อน (weak features) และชุดคุณสมบัติแบบเข้ม (strong features)

- ชุดคุณสมบัติแบบเข้ม คือชุดคุณสมบัติที่มนุษย์มองเห็นโดยทั่วไป กล่าวคือเป็นชุดคุณสมบัติที่มนุษย์สามารถสังเกตทำความเข้าใจ และวางหลักการในการจำแนกได้
- ชุดคุณสมบัติแบบอ่อน คือชุดคุณสมบัติที่มนุษย์ไม่สามารถมองเห็น หรือมองเห็นแต่ไม่ได้หยิบมาเป็นตัวปัจจัยหลักในการตัดสินใจ และวางหลักการในการจำแนก

จะยกตัวอย่างกรณีการจำแนกเลข 5 เราอาจพิจารณาว่าเลข 5 ดังแสดงในรูปที่ 2.8 ประกอบขึ้นจากขีดหนึ่งขีด แนวขวาง ขีดหนึ่งขีดแนวตั้ง และส่วนโค้งคล้ายวงกลม เป็นชุดคุณสมบัติที่มนุษย์สังเกตและเข้าใจโดยทั่วไป รวมถึงเป็นคุณสมบัติที่มนุษย์ใช้ในการสังเกตเห็นเส้นที่เชื่อมต่อกันจนประกอบเป็นเลข 5 อย่างไรก็ตาม แบบจำลองการเรียนรู้ใดๆ อาจเห็นมุมรอยต่อระหว่างขอบ (ซึ่งอาจสังเกตได้ว่าไม่มีเลขตัวใดเลยนอกจาก 1 ถึง 9 ยกเว้น 5 ที่มีมุมและขอบดังแสดง) เป็นตัวตัดสินใจในการเรียนรู้เลข 5 อย่างไรก็ตาม พึงสังเกตว่าแบบจำลองอาจจะแม้กระทั่งเลือกสังเกตเห็นพื้นที่ว่างบริเวณที่แตกต่างกันไป และใช้พื้นที่ว่างเหล่านั้นเพื่อสร้างข้อสรุปหรือตัดสินใจว่าเลขที่มองเห็นเป็นเลขใด (ซึ่งการนำมาซึ่ง “ข้อสรุป” จากที่ว่างนั้น ขัดกับวิสัยปกติของมนุษย์ในการสังเกตและมองเห็นอย่างชัดเจน)



## 2.9 การคำนวณหาสัญญาณรบกวน

ขั้นตอนวิธีการหาสัญญาณรบกวนนั้นมีรายละเอียดต่างกันไปตามวิธีการคำนวณ และแนวคิดของการคำนวณ ดังแสดงตัวอย่างวิธีต่อไปนี้

### 2.9.1 การหาสัญญาณรบกวนด้วยวิธีการก้าวเคลื่อนถอยหลัง

เมื่อฝึกสอนแบบจำลองการเรียนรู้เชิงลึกโดยได้ชุดตัวแปรเสริม  $\theta$  สำหรับแบบจำลอง  $M$  ซึ่งต่อไปนี้จะเรียกชุดแบบจำลองและตัวแปรเสริมรวมกันว่า  $M_\theta$  แล้ว เมื่อให้คู่จุดข้อมูลรับเข้าและส่งออก  $(x, y)$  ใดๆ เราอาจหาสัญญาณรบกวนได้ว่า

$$\eta' = \eta + l \frac{\partial}{\partial \eta} \mathcal{L}(x) \quad (2.7)$$

เมื่อ  $l$  เป็นค่าอัตราการเรียนรู้ (learning rate) โดยปกติมักมีค่าไม่มาก

จะสังเกตได้ว่าสมการที่ 2.7 มีลักษณะคล้ายกับสมการที่ 2.3 เป็นอย่างมาก แตกต่างกันเพียงแต่เครื่องหมายบวกหรือลบ และตัวแปรเทียบสำหรับการทำอนุพันธ์หลายตัวแปร (multivariable derivation) ขอให้สังเกตว่าในขณะที่สมการ 2.3 พยายามหาค่า  $\theta$  ที่ทำให้ฟังก์ชันสูญเสีย  $\mathcal{L}$  มีค่าต่ำที่สุด สมการที่ 2.7 กลับพยายามหาสัญญาณรบกวน  $\eta$  ที่ทำให้ฟังก์ชันสูญเสีย  $\mathcal{L}$  มีค่ามากที่สุด กล่าวคือตอบผิวดมมากที่สุด

### 2.9.2 การหาสัญญาณรบกวนด้วยวิธีการเครื่องหมายเกรเดียนต์แบบเร็ว (FGSM)

Goodfellow และคณะ [11] ได้เสนอขั้นตอนวิธีสำหรับคำนวณหาสัญญาณรบกวน  $\eta$  สำหรับชุดคู่จุด  $(x, y)$  โดยสังเกตทิศทางของเกรเดียนต์ของฟังก์ชันสูญเสีย  $\mathcal{L}$  เทียบกับ  $x$  ซึ่งสามารถคำนวณได้จาก

$$\eta = \epsilon \text{sign} \left( \vec{\nabla}_x \mathcal{L}(\theta, x, y) \right) \quad (2.8)$$

เรียกขั้นตอนวิธีนี้ว่าวิธีการเครื่องหมายเกรเดียนต์แบบเร็ว (Fast Gradient Sign Method: FGSM)

แนวคิดเบื้องหลังขั้นตอนวิธีนี้คือ เมื่อมีเกรเดียนต์ของฟังก์ชันสูญเสียต่อแบบจำลอง ทิศทางของเกรเดียนต์นั้นย่อมทำให้เกิดการเปลี่ยนแปลงต่อค่าของฟังก์ชันสูญเสียมากขึ้นที่สุด

ค่าจากฟังก์ชัน sign นั้นจะเป็นค่าบวกหรือลบหนึ่ง กล่าวคือเราสนใจพิจารณาเฉพาะทิศทางของเกรเดียนต์ และจะทำการคูณค่านั้นด้วยค่าคงที่  $\epsilon$  เพื่อเจือสัญญาณรบกวนไม่ให้มีความเข้มข้นมากเกินไป

สังเกตว่าความซับซ้อนเชิงเวลา (time complexity) ของขั้นตอนวิธีนี้เป็น  $\mathcal{O}(1)$  เพราะไม่มีการวนรอบซ้ำ

### 2.9.3 การฉายเกรเดียนต์ลดหลั่น $k$ ชั้น ( $k$ -PGD)

ขั้นตอนวิธี FGSM นั้นมีข้อดีในด้านเวลาการทำงาน อย่างไรก็ตาม การประมาณค่าของเกรเดียนต์ที่มากที่สุดโดยไม่ได้พิจารณาถึงพื้นผิวของฟังก์ชัน ย่อมทำให้ค่าของเกรเดียนต์ที่ออกมา นั้นผิดเพี้ยน และไม่ได้ค่าฟังก์ชันสูญเสียที่มากที่สุด ทำให้ไม่ใช่ชุดสัญญาณรบกวนที่ดีที่สุดที่จะโจมตีแบบจำลองได้

Madry และคณะ [13] จึงศึกษาขั้นตอนวิธีที่ใช้การวนซ้ำเพื่อหาค่าของ  $x' = x + \eta$  โดยมุ่งให้การวนซ้ำนั้นช่วยในการปรับค่าของ  $\eta$  เพื่อเพิ่มค่าฟังก์ชันสูญเสียอย่างแม่นยำ โดยปรับค่าของ  $x'$  ในแต่ละรอบ  $t$  ได้จากสมการ

$$x'_t = \text{Proj}_{x'}^\epsilon(x_{t-1} + \alpha \vec{\nabla}_{x_{t-1}} \mathcal{L}(x_{t-1})) \quad (2.9)$$

เมื่อ  $x_0 = x$

ขั้นตอนวิธีดังกล่าวเรียกว่าขั้นตอนวิธีการฉายเกรเดียนต์ลดหลั่น  $k$  ชั้น ( $k$ -Projected Gradient Descent:  $k$ -PGD) โดยจะวนซ้ำขั้นตอนนี้เป็นจำนวน  $k$  ครั้ง จะสังเกตว่าในรอบการวนซ้ำจะทำการฉาย (project) ทิศทางของเกรเดียนต์ของฟังก์ชันสูญเสียออกไปเป็นระยะทาง  $\alpha$  และฉายกลับเข้ามายังกรอบ  $\epsilon$  ที่กำหนดไว้หากขนาดของสัญญาณรบกวนมีค่าเกินตามต้องการ

#### 2.9.4 ความต่างของขั้นตอนวิธี

แม้ว่าขั้นตอนวิธี  $k$ -PGD จะทำงานได้ช้า แต่การทำงานที่เป็นแบบการวนซ้ำ (iteratively) ช่วยให้มั่นใจว่าการเพิ่มขึ้นของสัญญาณรบกวนจากเกรเดียนต์เป็นไปได้อย่างแม่นยำและสร้างความเสียหายใจการโจมตีได้มาก

ในขณะเดียวกัน ขั้นตอนวิธี FGSM แม้จะไม่สามารถสร้างสัญญาณโจมตีที่มีประสิทธิภาพ แต่ก็สามารถสร้างสัญญาณโจมตีที่โจมตีในกรณีทั่วไปได้ และสร้างได้อย่างรวดเร็ว

#### 2.10 การเรียนรู้เชิงโจมตี

การเรียนรู้เชิงโจมตีเป็นขั้นตอนวิธีหนึ่งในการเสริมความทนทานต่อการโจมตีให้แบบจำลองการเรียนรู้เชิงลึก ขั้นตอนวิธีการเรียนรู้เชิงโจมตีที่ง่ายที่สุดคือการสร้างชุดข้อมูลประสงคร้ายและนำไปฝึกสอนร่วมกับชุดข้อมูลฝึกหัดตั้งต้น ดังแสดงใน

---

**ขั้นตอนวิธี 2** การเสริมความแข็งแกร่ง (การเรียนรู้เชิงโจมตี)

---

รับเข้า:  $M$  เป็นแบบจำลองการเรียนรู้

รับเข้า:  $X$  เป็นชุดข้อมูลที่จะทำการโจมตี

รับเข้า:  $e$  เป็นจำนวนรอบการวนซ้ำ (epochs) ในการฝึกสอน

รับเข้า:  $m$  และ  $n$  เป็นขนาดของชุดฝึกสอนเล็กจิ๋ว (minibatch) บนชุดข้อมูลต้นฉบับและชุดข้อมูลประสงคร้าย

สำหรับ  $i = 1$  ถึง  $e$  ทำ

สร้างชุดฝึกสอนจิ๋วบนชุดข้อมูลต้นฉบับ  $B = \{x_1, x_2, \dots, x_m\}$ .

สร้างชุดฝึกสอนจิ๋วบนชุดข้อมูลประสงคร้าย  $B' = \{x'_1, x'_2, \dots, x'_n\}$ .

ฝึกสอนแบบจำลองด้วยชุดฝึกสอนเล็กจิ๋ว  $B$  และ  $B'$

จบการวนสำหรับ

ส่งออก: แบบจำลอง  $M$

---

## บทที่ 3

### วิธีดำเนินงาน

#### 3.1 การวิเคราะห์และออกแบบขั้นตอนวิธี

โครงการวิศวกรรมคอมพิวเตอร์ชั้นนี้มุ่งเสนอแนวคิดสำหรับการวิเคราะห์คลัสเตอร์เพื่อโจมตีแบบจำลองและใช้การโจมตีนั้นในการสอนแบบจำลองให้ทนทานต่อการโจมตี โดยตั้งอยู่บนแนวคิดและสมมติฐานต่อไปนี้

##### 3.1.1 พฤติกรรมของคลัสเตอร์สัญญาณรบกวน

หนึ่งในปัญหาของขั้นตอนวิธีเสริมความแข็งแกร่งแบบจำลอง คือการสร้างชุดสัญญาณรบกวนเพื่อฝึกสอนนั้นกินเวลานาน แม้จะมีขั้นตอนวิธีที่คำนวณได้อย่างรวดเร็วเช่นขั้นตอนวิธี FGSM แต่ขั้นตอนวิธีที่ใช้เวลาคิดคำนวณเร็วไม่สามารถสร้างสัญญาณรบกวนที่มีประสิทธิภาพในการโจมตีแบบจำลอง และสัญญาณรบกวนดังกล่าวไม่สามารถใช้ฝึกสอนแบบจำลองให้ทนทานต่อการโจมตีได้ [13]

เพื่อลดปัญหาดังกล่าว โครงการชั้นนี้มุ่งเสนอแนวคิดว่า

**ข้อเฝ้าระวัง 1:** เมื่อให้ชุดข้อมูลประสงคร้าย  $X'$  ที่โจมตีชุดข้อมูล  $X$  พฤติกรรมของสัญญาณรบกวน  $\eta$  สามารถจัดเป็นกลุ่มย่อยๆ ได้ และการโจมตีในสมาชิกของกลุ่มย่อยสามารถเกิดขึ้นได้ด้วยชุดสัญญาณโจมตีอื่นๆ ในจุดกลุ่มนั้น

**ข้อเฝ้าระวัง 2:** ผลการจัดกลุ่มคลัสเตอร์สัญญาณรบกวน ไม่ว่าจะบนสัญญาณรบกวนที่ได้มาจากวิธีการใด จะมีสมาชิกของแต่ละคลัสเตอร์ใกล้เคียงกัน

ด้วยข้อสมมติฐานที่ 2 เราสามารถสร้างสัญญาณรบกวนด้วยขั้นตอนวิธีที่เร็วเพื่อสร้างคลัสเตอร์ของสัญญาณรบกวนได้ เมื่อประกอบกับข้อสมมติฐานที่ 1 เราสามารถรู้ของคลัสเตอร์มาช่วยร่นเวลาในการโจมตีแบบจำลองได้

ขั้นตอนวิธี 3 เสนอการคลัสเตอร์ข้อมูล และพยายามสร้างสัญญาณโจมตีด้วยขั้นตอนวิธีที่แม่นยำเพียงหนึ่งสัญญาณต่อคลัสเตอร์ กล่าวคือขั้นตอนวิธีดังกล่าวเรียกใช้ฟังก์ชันสำหรับสร้างสัญญาณรบกวนที่รวดเร็วเพียงเพื่อจัดกลุ่มของสัญญาณรบกวน ก่อนจะทำการโจมตีอย่างแม่นยำด้วยขั้นตอนวิธีที่แม่นยำ (แต่ไม่มีประสิทธิภาพทางด้านเวลา) ต่อไป

##### 3.1.2 การเสริมความแข็งแกร่งด้วยวิธีการผสมคลัสเตอร์

จากขั้นตอนวิธี 3 เราสามารถนำชุดสัญญาณโจมตีมาฝึกสอนแบบจำลองเพื่อเสริมความแข็งแกร่งได้ ดังแสดงในขั้นตอนวิธี 4

#### 3.2 การทดลองวัดประสิทธิภาพ

นอกจากการเสนอขั้นตอนวิธีแล้ว ผู้จัดทำมุ่งความสนใจไปยังวิธีการวัดประสิทธิภาพของขั้นตอนวิธีด้วยเช่นกัน ผู้จัดทำโครงการจัดทำชุดคำสั่งสำหรับขั้นตอนวิธีที่เสนอในโครงการวิศวกรรมชั้นนี้ บนไลบรารี (library) การเรียนรู้เชิงลึก PyTorch [14]

---

**ขั้นตอนวิธี 3** ขั้นตอนวิธีสร้างสัญญาณรบกวนจากคลัสเตอร์

---

รับเข้า:  $M$  เป็นแบบจำลองการเรียนรู้

รับเข้า:  $X$  เป็นชุดข้อมูลที่จะทำการโจมตี

รับเข้า:  $k$  เป็นจำนวนคลัสเตอร์

รับเข้า:  $f_s$  เป็นฟังก์ชันสร้างสัญญาณรบกวนที่รวดเร็ว

รับเข้า:  $f_e$  เป็นฟังก์ชันสร้างสัญญาณรบกวนที่มีประสิทธิภาพ

สร้างชุดสัญญาณรบกวน  $P_s$  ที่โจมตีแบบจำลอง  $M$  บนชุดข้อมูล  $X$  ด้วยขั้นตอนวิธี  $f_s$

วิเคราะห์คลัสเตอร์ด้วยขั้นตอนวิธี  $k$ -มีนส์ บน  $P_s$  ด้วยจำนวนคลัสเตอร์  $k$  คลัสเตอร์

สำหรับ ทุกคลัสเตอร์  $c_i \in C$  ทำ

สร้างสัญญาณรบกวน  $p_i$  ที่มีความสามารถโจมตีทุกจุด  $c_i$  บนแบบจำลอง  $M$  ด้วยขั้นตอนวิธี  $f_e$

เก็บสัญญาณรบกวน  $p_i$  และเซต  $C_i$  ซึ่งมีสมาชิก  $c_i$  ทุกตัวในคลัสเตอร์  $C$

จบการวนสำหรับ

ส่งออก: ค่า  $p_i$  และ  $c_i$  สำหรับทุก  $i = 1$  ถึง  $k$

---



---

**ขั้นตอนวิธี 4** การเสริมความแข็งแกร่งด้วยวิธีการผสมคลัสเตอร์

---

รับเข้า:  $M$  เป็นแบบจำลองการเรียนรู้

รับเข้า:  $X$  เป็นชุดข้อมูลที่จะทำการโจมตี

รับเข้า:  $e$  เป็นจำนวนรอบการวนซ้ำ (epochs) ในการฝึกสอน

รับเข้า:  $m$  และ  $n$  เป็นขนาดของชุดฝึกสอนเล็กจิ๋ว (minibatch) บนชุดข้อมูลต้นฉบับและชุดข้อมูลประสงคร้าย

รับเข้า:  $w$  และ  $w'$  เป็นน้ำหนักของชุดฝึกสอนเล็กจิ๋ว (minibatch) บนชุดข้อมูลต้นฉบับและชุดข้อมูลประสงคร้าย

สร้างสัญญาณรบกวนจากคลัสเตอร์ ที่โจมตีข้อมูล  $X$  บนแบบจำลอง  $M$  โดยใช้ขั้นตอนวิธี 3.

สำหรับ  $i = 1$  ถึง  $e$  ทำ

สร้างชุดฝึกสอนจิ๋วบนชุดข้อมูลต้นฉบับ  $B = \{x_1, x_2, \dots, x_m\}$ .

สร้างชุดฝึกสอนจิ๋วบนชุดข้อมูลประสงคร้าย  $B' = \{x'_1, x'_2, \dots, x'_n\}$ .

ฝึกสอนแบบจำลองแบบถ่วงน้ำหนัก โดยให้น้ำหนัก  $w$  บน  $B$  และ  $w'$  บน  $B'$ .

จบการวนสำหรับ

ส่งออก: แบบจำลอง  $M$

---

ชุดคำสั่งทั้งหมดสามารถเข้าถึงออนไลน์ได้ผ่านระบบควบคุมเวอร์ชัน (Software Version Control) GitHub ผ่านทาง <https://github.com/srakrnxKU/adversarial-project/>

## บรรณานุกรม

- [1] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.
- [2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [3] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain,” *Psychological Review*, vol. 65, no. 6, p. 386–408, 1958.
- [4] K. Hornik, “Approximation capabilities of multilayer feedforward networks,” *Neural Networks*, vol. 4, no. 2, p. 251–257, 1991.
- [5] G. Cybenko, “Approximation by superpositions of a sigmoidal function,” *Mathematics of Control, Signals, and Systems*, vol. 2, pp. 303–314, Dec. 1989.
- [6] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, “Multilayer feedforward networks with a nonpolynomial activation function can approximate any function,” *Neural Networks*, vol. 6, pp. 861–867, Jan. 1993.
- [7] A. Kratsios, “Universal approximation theorems,” 2019.
- [8] M. A. Nielsen, “Neural networks and deep learning,” Jan 1970.
- [9] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” in *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, 1998.
- [10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, NIPS’12, (USA)*, pp. 1097–1105, Curran Associates Inc., 2012.
- [11] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 2014.
- [12] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, “Adversarial examples are not bugs, they are features,” 2019.
- [13] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” 2017.
- [14] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32* (H. Wallach, H. Larochelle,

A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), pp. 8024–8035, Curran Associates, Inc., 2019.