**Example 1.py - Training Loop พื้นฐาน**

**🎯 หลักการหลัก:**

Training Loop คือ "หัวใจ" ของการเรียนรู้ของ AI - เหมือนการสอนเด็กอ่านหนังสือซ้ำๆ จนเก่ง

**🔍 อธิบายแต่ละส่วน:**

**1. Simple Neural Network**

python

class SimpleClassifier(nn.Module):

def \_\_init\_\_(self, input\_size, hidden\_size, num\_classes):

self.layer1 = nn.Linear(input\_size, hidden\_size)

self.layer2 = nn.Linear(hidden\_size, hidden\_size // 2)

self.layer3 = nn.Linear(hidden\_size // 2, num\_classes)

**อุปมา:** เหมือนสมองคนที่มี 3 ชั้น

* **Layer 1:** รับข้อมูลจากภายนอก (เหมือนตา หู)
* **Layer 2:** ประมวลผลข้อมูล (เหมือนสมองส่วนกลาง)
* **Layer 3:** ตัดสินใจ/จำแนก (เหมือนการตอบ)

**2. Training Loop หลัก**

python

def basic\_training\_loop(model, train\_loader, optimizer, criterion, num\_epochs=10):

for epoch in range(num\_epochs): *# ← เรียนซ้ำหลายรอบ*

for data, target in train\_loader: *# ← ดูข้อมูลทีละชุด*

optimizer.zero\_grad() *# ← ลืมการเรียนรู้เก่า*

output = model(data) *# ← ทำนาย*

loss = criterion(output, target) *# ← ดูว่าผิดมากแค่ไหน*

loss.backward() *# ← คิดว่าจะแก้ยังไง*

optimizer.step() *# ← แก้ไข/ปรับปรุง*

**อุปมาการเรียนรู้:**

1. **Zero\_grad()** = ลืมคำตอบเก่า (ไม่งั้นจะสับสน)
2. **Forward pass** = ลองตอบดู
3. **Loss calculation** = ดูว่าผิดมากแค่ไหน
4. **Backward pass** = คิดว่าผิดตรงไหน
5. **Optimizer step** = แก้ไขให้ดีขึ้น

**3. Optimizer คืออะไร?**

* **SGD** = เรียนรู้แบบช้าๆ แต่มั่นคง (เหมือนคนระมัดระวัง)
* **Adam** = เรียนรู้แบบฉลาด ปรับตัวได้ (เหมือนคนเก่ง)
* **RMSprop** = เรียนรู้แบบสมดุล

**🔧 Example 2.py - Backward Propagation**

**🎯 หลักการหลัก:**

Backpropagation คือการ "ย้อนกลับไปแก้ไขความผิด" - เหมือนเวลาทำข้อสอบผิด แล้วย้อนดูว่าผิดตั้งแต่ขั้นไหน

**🔍 อธิบายแต่ละส่วน:**

**1. Manual Backpropagation**

python

def manual\_backward(self, x, y\_true, y\_pred):

*# คำนวณว่าผิดมากแค่ไหน*

dL\_dz2 = 2 \* (y\_pred - y\_true) / batch\_size

*# ย้อนกลับไปหา error ของแต่ละ layer*

dL\_dW2 = torch.matmul(self.a1.t(), dL\_dz2)

dL\_da1 = torch.matmul(dL\_dz2, self.W2.t())

**อุปมา:** เหมือนการทำข้อสอบคณิตศาสตร์

1. **ผลลัพธ์ผิด** → ต้องย้อนดูว่าผิดจากขั้นไหน
2. **ย้อนจากขั้นสุดท้าย** → กลับไปขั้นต้น
3. **แก้ไขแต่ละขั้น** → ให้ได้คำตอบที่ถูก

**2. Gradient Clipping**

python

grad\_norm\_before = torch.nn.utils.clip\_grad\_norm\_(model.parameters(), max\_norm)

**อุปมา:** เหมือนการจำกัดความเร็วรถ

* ถ้า gradient ใหญ่เกินไป = รถวิ่งเร็วเกินไป = อันตราย
* Clipping = จำกัดความเร็ว = ปลอดภัย

**3. Learning Rate Scheduling**

python

scheduler = optim.lr\_scheduler.StepLR(optimizer, step\_size=5, gamma=0.5)

**อุปมา:** เหมือนการเรียนขับรถ

* **ตอนแรก:** เรียนรู้เร็วๆ (learning rate สูง)
* **ทีหลัง:** เรียนรู้ช้าๆ ละเอียด (learning rate ต่ำ)

**📊 Example 3.py - Monitor Loss และ Metrics**

**🎯 หลักการหลัก:**

การติดตามประสิทธิภาพ - เหมือนการตรวจสุขภาพประจำ เพื่อดูว่า AI เรียนรู้ดีแค่ไหน

**🔍 อธิบายแต่ละส่วน:**

**1. MetricsTracker**

python

class MetricsTracker:

def \_\_init\_\_(self):

self.train\_losses = [] *# ← เก็บความผิดพลาดในการเรียน*

self.train\_accuracies = [] *# ← เก็บความแม่นยำในการเรียน*

self.val\_losses = [] *# ← เก็บความผิดพลาดในการสอบ*

self.val\_accuracies = [] *# ← เก็บความแม่นยำในการสอบ*

**อุปมา:** เหมือนสมุดรายงานผลการเรียน

* **Train metrics** = คะแนนการบ้าน (ที่บ้าน)
* **Validation metrics** = คะแนนสอบกลางภาค (ที่โรงเรียน)

**2. Early Stopping**

python

if val\_metrics['loss'] < best\_val\_loss:

best\_val\_loss = val\_metrics['loss']

patience\_counter = 0

best\_model\_state = model.state\_dict().copy()

else:

patience\_counter += 1

if patience\_counter >= early\_stopping\_patience:

print("Early stopping triggered")

**อุปมา:** เหมือนการออกกำลังกาย

* ถ้า**ไม่มีพัฒนาการ**นาน → หยุดเพื่อป้องกันเหนื่อยเปล่า
* เก็บ**สภาพร่างกายที่ดีที่สุด**ไว้

**3. Comprehensive Evaluation**

python

accuracy = accuracy\_score(all\_targets, all\_predictions) \* 100

precision = precision\_score(all\_targets, all\_predictions, average='weighted') \* 100

recall = recall\_score(all\_targets, all\_predictions, average='weighted') \* 100

f1 = f1\_score(all\_targets, all\_predictions, average='weighted') \* 100

**อุปมา:** เหมือนการประเมินนักเรียน

* **Accuracy** = คะแนนรวมทั้งหมด
* **Precision** = ถ้าตอบว่าถูก จริงๆ ถูกไหม?
* **Recall** = คำตอบที่ถูกทั้งหมด เราหาได้กี่ข้อ?
* **F1** = ค่าเฉลี่ยของ Precision และ Recall

**✅ Example 4.py - Validation Loop**

**🎯 หลักการหลัก:**

Validation คือการ "สอบจริง" เพื่อดูว่า AI เรียนรู้ได้ดีจริงหรือแค่จำ

**🔍 อธิบายแต่ละส่วน:**

**1. Cross Validation (K-fold)**

python

def cross\_validation(self, dataset, k\_folds=5):

for fold in range(k\_folds):

*# แบ่งข้อมูลเป็น train/val*

*# เรียนและทดสอบ*

*# เก็บผลลัพธ์*

**อุปมา:** เหมือนการสอบหลายรอบ

* แบ่งข้อมูลเป็น **5 ส่วน**
* ใช้ **4 ส่วนเรียน**, **1 ส่วนสอบ**
* **สลับกัน 5 รอบ** → ได้ผลลัพธ์ที่เชื่อถือได้

**2. Holdout Validation**

python

def split\_dataset(self, dataset):

train\_size = int(self.train\_ratio \* total\_size) *# 70%*

val\_size = int(self.val\_ratio \* total\_size) *# 15%*

test\_size = total\_size - train\_size - val\_size *# 15%*

**อุปมา:** เหมือนการแบ่งหนังสือ

* **70% เรียน** (สำหรับท่องจำ)
* **15% ทดสอบระหว่างเรียน** (สำหรับปรับปรุง)
* **15% สอบจริง** (สำหรับประเมินผลจริง)

**3. Bootstrap Validation**

python

def bootstrap\_validate(self, model, dataset):

for bootstrap\_iter in range(self.n\_bootstrap):

*# สุ่มเลือกข้อมูล (มีซ้ำได้)*

indices = torch.randint(0, dataset\_size, (sample\_size,))

*# เรียนและทดสอบ*

**อุปมา:** เหมือนการจับฉลากทดสอบ

* **จับฉลากหลายครั้ง** (สุ่มข้อมูล)
* **แต่ละครั้งอาจซ้ำได้**
* **เฉลี่ยผลลัพธ์** → ได้ความเชื่อมั่น

**4. Model Selection**

python

class ModelSelector:

def compare\_models(self, dataset):

for model\_name, model\_config in self.models\_config.items():

*# ทดสอบแต่ละ model*

*# เปรียบเทียบผลลัพธ์*

*# เลือก model ที่ดีที่สุด*

**อุปมา:** เหมือนการแข่งขันหาคนเก่ง

* **หลายคนแข่ง** (หลาย model)
* **ใช้เกณฑ์เดียวกัน** (same validation)
* **เลือกคนที่ดีที่สุด** (best model)

**🎯 สรุปภาพใหญ่:**

**เปรียบเทียบกับการเรียนของมนุษย์:**

1. **Example 1** = **การเรียนพื้นฐาน**
   * เรียนรู้วิธีการเรียน
   * ลองวิธีเรียนต่างๆ
2. **Example 2** = **การเข้าใจลึก**
   * เข้าใจว่าทำไมถึงผิด
   * แก้ไขอย่างชาญฉลาด
3. **Example 3** = **การติดตามความก้าวหน้า**
   * วัดผลความก้าวหน้า
   * ป้องกันการเรียนผิดทาง
4. **Example 4** = **การสอบจริง**
   * ทดสอบความรู้จริง
   * เลือกวิธีเรียนที่ดีที่สุด

**ทำไมต้องมีทั้งหมดนี้?**

* **ไม่มี Training Loop** = ไม่รู้จักเรียน
* **ไม่มี Backprop** = เรียนแต่ไม่รู้จักแก้ไข
* **ไม่มี Monitoring** = เรียนตาบอด ไม่รู้ว่าดีขึ้นหรือเปล่า
* **ไม่มี Validation** = อาจจำแต่ไม่เข้าใจ (overfitting)

**ทั้งหมดนี้ทำงานร่วมกันเพื่อสร้าง AI ที่เรียนรู้ได้จริงและใช้งานได้จริงครับ!**