2025 Data Mining HW2 報告

名子: 鄭睿宏
學號: 314554025

Q1: Explain your implementation which get the best performance in detail.

Ans:
本次實作採用基於**Convolutional Autoencoder, CAE** 的非監督式異常檢測方法。核心策略是僅使用「正常」樣本的數據來訓練模型，使其學會如何精確地重建「正常」數據的特徵。在評估時，當模型遇到「異常」樣本，其重建的誤差（重建損失）將會顯著高於正常樣本，我們便以此作為判斷異常的依據。

以下為詳細的實作步驟：

- **1. 資料前處理 (Data Preprocessing)：**
  - **FFT 轉換：** 這是本次實作最關鍵的前處理步驟。我們不直接使用原始像素，而是將影像轉換到頻域 (Frequency Domain) 進行分析。
    i. 讀取圖片並轉換為**灰階 (Grayscale)**。
    ii. 將圖片統一 resize 為 **256x256**。
    iii. 使用 `np.fft.fft2` 進行二維傅立葉轉換 (FFT)。
    iv. 使用 `np.fft.fftshift` 將零頻率分量移至頻譜中心。
    v. 計算**對數幅度譜 (Log-Magnitude Spectrum)**：$20 \cdot \log(|F(u,v)| + \epsilon)$，這有助於視覺化並壓縮高動態範圍。
    vi. 最後，對幅度譜進行 **Min-Max 歸一化**至 [0, 1] 區間，使其適合神經網路的輸入（特別是對應到 Decoder 最後的 Sigmoid 激活函數）。
  - **自定義資料集 (Custom Dataset)：** 使用自定義的 `FFTDataset` ，在 `__getitem__` 階段動態讀取圖片並執行上述的 FFT 前處理流程。
- **2. 模型架構 (Model Architecture)：**
  - 我們使用了一個對稱的 `ConvAutoencoder` ，包含編碼器 (Encoder) 和解碼器 (Decoder) 兩部分，專門處理 1x256x256 的頻譜輸入。
  - **Encoder (編碼器)：**
    - `Conv2d(1, 16, k=3, s=2, p=1) + ReLU` $\rightarrow$ (16, 128, 128)

        - `Conv2d(16, 32, k=3, s=2, p=1) + ReLU` → (32, 64, 64)
        - `Conv2d(32, 64, k=3, s=2, p=1) + ReLU` → (64, 32, 32)
    - **Decoder (解碼器)：**
        - `ConvTranspose2d(64, 32, k=3, s=2, p=1, op=1) + ReLU` → (32, 64, 64)
        - `ConvTranspose2d(32, 16, k=3, s=2, p=1, op=1) + ReLU` → (16, 128, 128)
        - `ConvTranspose2d(16, 1, k=3, s=2, p=1, op=1) + Sigmoid` → (1, 256, 256)
    - **模型細節：** Encoder 使用 `stride=2` 的卷積層進行降維，Decoder 使用 `stride=2` 的轉置卷積層 (ConvTranspose2d) 將其還原。最後一層使用 `Sigmoid` 激活函數，以匹配輸入數據的 [0, 1] 範圍。
- **3. 訓練過程 (Training Process)：**
    - **Loss Function：** 我們使用**均方誤差 (Mean Squared Error, `nn.MSELoss`)** 作為損失函數。它計算輸入的 FFT 幅度譜與 Autoencoder 重建的幅度譜之間的像素級差異。
    - **Optimizer：** 使用 `optim.Adam` 優化器，學習率 (Learning Rate) 設為 `1e-3`。
    - **超參數 (Hyperparameters)：** `batch_size` 設為 32， `num_epochs` 設為 50。
    - **訓練優化：** 為了加速訓練，導入了 **自動混合精度 (AMP)** (`torch.cuda.amp.GradScaler` 和 `autocast`)，在 CUDA 設備上能有效減少 VRAM 佔用並提升運算速度。
- **4. 異常分數 (Anomaly Score)：**
    - 在 `evaluate` 函式中，我們將測試集影像（同樣經過 FFT 處理）輸入到訓練好的 Autoencoder。
    - 異常分數被定義為**重建損失 (Reconstruction Loss)**。
    - 具體來說，我們使用 `nn.MSELoss(reduction='none')` 來計算每張影像的重建 MSE，然後取該影像所有像素點 MSE 的**平均值** (`.mean(dim=[1,2,3])`)，這個平均 MSE 值即為該樣本的最終異常分數。
    - 分數越高，代表模型重建該樣本的難度越大，該樣本為異常的可能性也越高。
- **5. 閾值與評估 (Thresholding & Evaluation)：**
    - 為了在本地端驗證準確率，我們採用**中位數 (Median)** 作為分類閾值。
    - 在 `run_evaluation_and_save` 函式中，計算測試集所有樣本異常分數的**中位數** (`np.median(all_scores)`)。
    - 凡是異常分數**高於**此中位數閾值的樣本，即被判斷為異常 (prediction=1)，反之則為正常 (prediction=0)。
    - 最後，將此二元預測結果與 `submission_h.csv` (Ground Truth) 進行比對，計算出「簡易準確率」，以監控模型在訓練過程中的表現（例如每 20 個 epoch 評估一次）。

Q2: Explain the rationale for using auc score instead of F1 score for binary classification in this homework.

Ans:
在這次作業中，使用 AUC (Area Under the Curve) 分數 而非 F1 分數作為評估指標，主要基於以下幾個考量，這些考量都與本次「異常檢測」(Anomaly Detection) 任務的本質有關：

- **1. 處理類別不平衡 (Imbalanced Data)：**
  - 異常檢測任務天生就是高度不平衡的。訓練集中只包含「正常」樣本，而測試集 中「異常」樣本的數量想必也遠少於「正常」樣本。
  - F1 分數（由精確率 Precision 和召回率 Recall 計算而來）在面對嚴重不平衡的數據時，可能會產生誤導。例如，一個模型若將所有樣本都預測為「正常」（Negative Class），它會獲得極高的準確率 (Accuracy) 和高精確率，但 F1 分數會極低（因為召回率為 0）。
  - AUC 衡量的是模型將「正樣本」排在「負樣本」前面的能力，它對類別不平衡的數據集更為穩健 (robust)，能更公允地評估模型在稀有類別（異常）上的辨識能力。
- **2. 閾值獨立性 (Threshold-Independent)：**
  - F1 分數的計算**依賴於一個特定的分類閾值**（Threshold）。例如，Autoencoder 會產生重建損失，我們必須設定一個閾值（例如：損失 > 0.1 才是異常），才能計算 F1 分數。選擇不同的閾值，F1 分數也會跟著劇烈變化。
  - AUC (ROC AUC) 則是「閾值獨立」的。它評估的是模型在**所有可能閾值**下的綜合表現。這在異常檢測中至關重要，因為我們關心的是模型排序（ranking）的能力——即模型是否能穩定地給予「異常」樣本比「正常」樣本更高的異常分數（無論這個分數的絕對值是多少）。
- **3. 任務目標的對焦：**
  - 本作業的目標是**檢測**異常，而不一定是立即做出「是/否」的決策。
  - AUC 衡量了模型區分兩個類別的**整體能力**。一個高 AUC 分數意味著模型很擅長將異常和正常樣本分開。
  - F1 分數則更側重於在某個**特定決策點**（閾值）上的「精確度」和「完整性」的平衡。在現實世界的工業檢測中，最終的閾值可能會根據商業考量（例如：寧可錯殺一百，不可放過一個）來動態調整，因此一個能提供良好排序（高 AUC）的模型比一個僅在某個點上 F1 高的模型更有價值。

總結來說，由於異常檢測的高度不平衡性，以及評估模型「排序能力」而非「特定閾值分類」的需求，AUC 是比 F1 分數更合適、更穩健的評估指標。

Q3: Discuss the difference between semi-supervised

learning and unsupervised learning.

Ans:
半監督學習 (Semi-supervised Learning) 和非監督學習 (Unsupervised Learning) 都是機器學習的分支，它們之間最核心的區別在於**訓練數據中是否包含label (Labels)**。

- **1. 非監督學習 (Unsupervised Learning)**
  - **訓練數據：**
    - 完全**不使用**任何label。
    - 算法的輸入只有特徵數據 (X)，沒有對應的答案 (y)。
  - **學習目標：**
    - 從數據中找出內在的結構、模式或分佈。
    - 目標不是預測一個特定label，而是「理解數據」本身。
  - **常見任務：**
    - **分群 (Clustering)：** 將相似的數據點歸為一類（例如本作業提到的 K-means）。
    - **降維 (Dimensionality Reduction)：** 在保留重要資訊的前提下，減少數據的特徵維度（例如 PCA 或本作業提到的 Autoencoder）。
    - **密度估計 (Density Estimation)：** 學習數據的潛在分佈（例如本作業的 One-Class SVM 或 Robust Covariance，它們試圖學習「正常」數據的分佈）。
  - **在本次作業中的應用：**
    - 本次作業的訓練集只包含正常樣本，沒有「異常」label。因此，我們使用的方法（如 Autoencoder 或 One-Class SVM）都是在學習「正常」數據的模式，這本質上是一種非監督的學習範式。
- **2. 半監督學習 (Semi-supervised Learning)**
  - **訓練數據：**
    - **混合使用**label數據。
    - 訓練集中包含**一小部分**有label的數據 (X_labeled, y_labeled) 和**大（多）部分**沒有label的數據 (X_unlabeled)。
  - **學習目標：**
    - 利用大量的無label數據來**輔助**和**提升**在少量有label數據上學習到的模型效能。
    - 其核心假設是：無label數據的結構和分佈有助於更準確地劃分決策邊界。
  - **常見任務：**
    - **半監督分類：** 主要目標是分類，但label數據很稀少（例如：標記了 100 張「正常」和 100 張「異常」的圖片，但還有 10000 張未標記的圖片）。

- **label傳播 (Label Propagation)：** 將已知label的資訊，「傳播」到數據結構中相近的未標記樣本上。

• 總結：

特徵	非監督學習 (Unsupervised)	半監督學習 (Semi-supervised)
訓練數據	**完全沒有**label	**少量**label + **大量**無label
學習目標	探索數據結構、分群、降維	利用無label數據， 提升label預測效能
是否有預測目標 (y)	否（探索 X）	是（預測 y）
本次作業 (HW2) 範例	One-Class SVM, Autoencoder (學習正常數據分佈)	(不適用，因為我們沒有任何「異常」label)