Statistical Learning and Deep Learning, Project 2

Jie Lin 林杰 (B11705048)[1]

[1]Department of Information Management, National Taiwan University

December 26, 2024

1 檔案說明

主要程式檔案:

- `training_swin.ipynb`: 用於訓練 Swin Transformer 模型的程式，包含資料處理、模型建立、訓練及儲存模型等步驟。
- `training_swin.html`: `training_swin.ipynb` 的 Jupyter Notebook 輸出結果，提供訓練過程和結果。
- `training_resnet50_dropout.ipynb`: 用於訓練 ResNet50 模型並加上 Dropout 層的程式，處理資料、模型設計、訓練及儲存模型。
- `training_resnet50_dropout.html`: `training_resnet50_dropout.ipynb` 的 Jupyter Notebook 輸出結果，展示訓練過程和最終模型的效果。
- `predicts.ipynb`: 進行模型推論的程式，將訓練好的模型應用於 `test_data`，並生成預測結果。
- `predicts.html`: `predicts.ipynb` 的 Jupyter Notebook 輸出結果，顯示推論過程和預測結果。
- `datasets.py`: 定義資料集處理的程式碼，例如：批次處理等功能。

其他主程式需要的檔案:

- `class_mapping.json`: 包含類別標籤與數值對應的 JSON 檔案。
- `ground_truth_probabilities.csv`: 訓練所使用的 Ground Truth 檔案。

最終上傳到 Kaggle 的預測檔案:

- `test_predictions_ensemble_mixed2.csv`: 由 Swin Transformer 和 ResNet50 模型加權平均後的預測結果，並提交至 Kaggle 進行評估。

我的結果是使用 `Swin Transformer` 和 `ResNet50 with Dropout` 共同產出的,所以在 `predicts.ipynb` 中，我將這兩個模型的預測結果進行了加權平均，並且將最終的預測結果輸出到 `test_predictions_ensemble_mixed2.csv` 中。

2 訓練與執行方式

1. 訓練(I):在 `training_swin.ipynb` 中執行所有的 cell,會產生 `model_epoch_56_swin_no_freeze.pth`。
2. 訓練(II):在 `training_resnet50_dropout.ipynb` 中執行所有的 cell,會產生 `model_epoch_39.pth`。
3. 推論: 執行 `predicts.ipynb` 中的所有 cell，會產生 `test_predictions_ensemble_mixed2.csv`。

3 實驗設計與數據分割

在我的所有的實驗中，我將所有的訓練資料分割為訓練集和驗證集，比例為 90% 訓練集和 10% 驗證集，透過驗證集的結果來實驗模型的效能。

4 模型嘗試與對比實驗

在這次的實驗中，我應用了多個深度學習技巧與模型：

4.1 模型嘗試

嘗試的模型包括：

1. **EfficientNet (B0)**：EfficientNet 是一系列高效的卷積神經網絡（CNN）模型。B0 是其中的最小版本，透過結合模型大小、深度、解析度和寬度來達到最佳的效能和效率。其主要特點是使用了複合縮放方法，能在保持計算效率的情況下提升模型效能。

2. **ResNet50**：ResNet50 是一種深度卷積神經網絡，具有 50 層，並使用 Residual Network 架構，這讓模型在層數增加時依然能避免梯度消失或爆炸的問題。

3. **Swin Transformer**：Swin Transformer 是一種基於 Transformer 的視覺模型，主要特點是將 Transformer 的 Self Attention 引入圖像識別領域。與傳統的 CNN 模型不同，Swin Transformer 使用分層的窗口自注意力機制來捕捉局部與全局的圖像特徵，並且能夠在不同解析度下進行有效的圖像處理。

4. **ResNet50 + Swin Transformer**：這個模型將 ResNet50 和 Swin Transformer 進行結合，利用 ResNet50 的深度學習特性和 Swin Transformer 在捕捉圖像全局特徵方面的優勢。這樣的結合可以讓模型在處理圖像時同時發揮兩者的優勢，提升預測的準確度和效能。

以下為訓練 ResNet50 的 Training mAP 與 Validation mAP 的圖表：

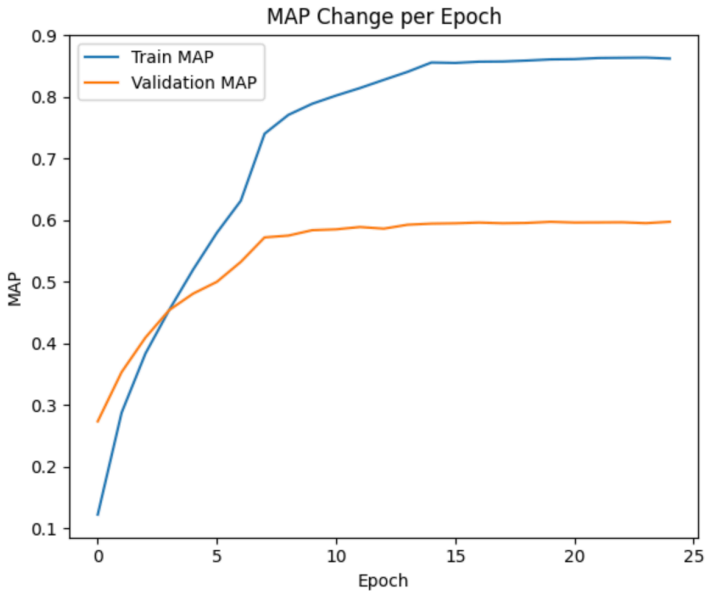

Figure 1: ResNet50 Training mAP 與 Validation mAP

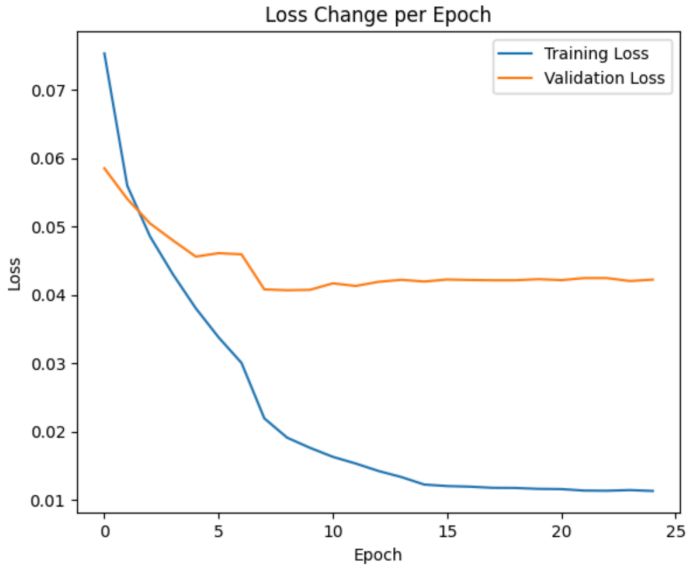

Figure 2: ResNet50 Training Loss 與 Validation Loss

因為在訓練 ResNet50 的過程中，發現其非常容易 Overfitting，所以我在後續的實驗中實驗了多種深度學習技巧，以提高模型的泛化能力。

4.2 深度學習技巧嘗試

我嘗試的技巧包括：

- **I.** 數據增強技術：如隨機裁剪、旋轉、色彩抖動等，來提高模型的泛化能力。
- **II.** Mixup 技術：將不同樣本進行加權混合。
- **III.** Dropout(rate = 0.5)：在 ResNet50 和 Swin Transformer 模型中加入 Dropout 層，以防止過擬合。

我在訓練時並未進行 Early Stopping，而是將每個 epoch 的模型保存，並在訓練結束後選擇最佳的模型。我發現使用以上三種技巧的 ResNet50 + Swin Transformer ，大約都會在 35 到 55 個 epoch 左右達到最佳效果。

4.3 性能比較

以下表格總結了各種前處理方式與模型組合的性能表現（基於驗證集 mAP）：

模型	前處理方法	驗證集 mAP
EfficientNet (B0)	**I**	0.3020
ResNet50	**I**	0.5723
ResNet50	**I**, **III**	0.6422
ResNet50	**I**, **II**, **III**	0.6754
Swin Transformer	**I**, **II**, **III**	0.6653
ResNet50 + Swin Transformer	**I**, **II**, **III**	**0.6989**

Table 1: 各種前處理方式與模型組合的性能表現

Table 1 顯示，前處理方法 **I**, **II**, **III** 搭配 ResNet50 + Swin Transformer 的組合可以帶來最好的結果。最終的預測結果是這兩個模型的預測結果的平均。這也是我提交的程式碼中包含的內容。

4.4 最終模型訓練

在最終的模型訓練中，我使用了 ResNet50 和 Swin Transformer 兩個模型，配合上述防止 Overfitting 的技巧，並使用了所有提供的資料集進行訓練。最後，我將這兩個模型的預測結果進行加權平均，並將最終的預測結果輸出到 `test_predictions_ensemble_mixed2.csv` 中。Kaggle 上的 public score 為 0.70315 ，排名為 16/77。

Figure 3: Kaggle Public Score