

雲林科技大學資訊管理學系

機器學習

Department of Information Management National Yunlin University of
Science and Machine Learning

專案作業二

基於兩個資料集使用 CNN 模型進行圖像分類預測之績效比較

Student Name: 彭冠穎

Student ID: M10923025

E-mail: M10923025@gmail.yumtech.edu.tw

Student Name: 趙國翔

Student ID: M10923013

E-mail: M10923013@gmail.yumtech.edu.tw

Student Name: 曾信嘉

Student ID: M10923032

E-mail: M10923032@gmail.yumtech.edu.tw

Student Name: 陳楚柔

Student ID: M10923011

E-mail: M10923011@gmail.yumtech.edu.tw

指導教授:許中川

Advisor : Chung-Chian Hsu, Ph.D.

中華民國 110 年 4 月

April 2021

摘要

CNN 在影像辨識方面的十分強大，而其也衍生出許多傑出的模型。因此，本研究使用 CNN 與其他三種模型於 Mango 與 CIFAR10 資料集中，進行實驗績效比較。最後實驗結果表明在 Mango 與 CIFAR10 中 DenseNet201 模型皆達到最優秀的績效，且使用並搭配 Early stopping 與 ReduceLROnPlateau 等方法與 Dropout 和資料增強技術能夠有效地提升模型。

關鍵字：CNN、ResNet50、DenseNet201、VGG16

第一章、緒論

1.1 動機

近年來影像辨識領域是深度學習中最為發展的一塊領域，影像辨識技術將使用深度學習進行訓練時，從中學習人類視覺辨識能力，並自動判讀影像中的關鍵資訊，而目前應用於無人駕駛、醫療影像等技術。深度學習中 CNN 在影像辨識方面具有強大的威力，許多影像辨識的模型也是以 CNN 為架構基礎去做延伸，因此本研究將透過深度學習 CNN 模型進行影像分類，使用 Aidea 提供之 Mango 資料集與 CIFAR10 資料集進行。

1.2 目的

本研究使用 Mango 與 CIFAR10 資料集使用深度學習中 CNN 及相關延伸模型進行影像分類預測，透過嘗試不同 CNN 層數模型與不同延伸模型進行實驗，使用 Accuracy, Precision, Recall 及 F1 做為績效指標，並比較不同模型績效結果優劣。

第二章、方法

2.1 程式操作方法

Step1：請至 UCI 下載本研究所使用的 adult 資料集與 Abalone 資料集。

Step2：請至本小組 GitHub 專案資料夾下載程式碼

Step3：請將下載的程式碼放在資料夾底下，使用 Jupyter notebook 執行程式碼檔案即可操作程式。




第三章、實驗

3.1 資料集

- **Mango**

本資料集對芒果進行三種等級分類，分為 A、B、C 三個等級，依序為出口用、內銷用、加工用。本研究將透過影像辨識技術透過芒果影像進行辨識技術。

表 1 影像範例圖表

		
A 類	B 類	C 類

- **CIFAR10**

本資料集是一組大小為 32*32 的 RGB 影像，總共包含十個類別：飛機、汽車、鳥、貓、鹿、狗、青蛙、馬、船及卡車。共包含 60000 張圖片，每種類別 6000 張。

表 2 影像範例圖表

圖片					
類別	airplane	automobile	bird	cat	deer
圖片					
類別	dog	frog	horse	ship	truck

3.2 前置處理

- **圖片大小前處理**

全連接層的參數就和輸入圖像大小有關，進行訓練時需把輸入的所有像素點連接起來，並指定輸入層神經元個數和輸出層神經元個數，因此需要規定輸入圖片的大小，本研究透過 opencv-python 將圖片大小調整成一致。

- **資料正規化**

因資料中不同顏色通道的分布狀況不同，在進行訓練時可能會使模型複雜度提升，進而造成 Overfitting，因此為預防此狀況可透過正規化的方式將資

料按照同比例縮放，使資料落在某一特定區間。本研究將透過資料正規化將輸入參數進行特徵縮放至 0~1。

- **資料增強**

資料增強是人工增加訓練集的一種方式，通過修改資料集中的圖片達成，本研究透過 keras 框架中 ImageDataGenerator 提供了資料增強的相關功能，包含位移、旋轉、縮放等方式增加更多的訓練資料。

3.3 實驗設計

表 3 類神經網路參數定義表

變數	定義
activation	1) ReLU 在神經網絡中，線性整流作為神經元的激活函數，定義了該神經元在線性變換之後的非線性輸出結果。 2) Linear 對輸入數據應用線性轉換
kernel_initializer	設置 Keras 各層權重隨機初始值的方法
loss	loss 這個函數是用來最小化甚麼目標函數的計算方法的。 sklearn 提供了多種選擇： 1) 輸入均方誤差「mean_squared_error」，使用最小平方法的目標函數預測值與實際值的差距之平均值。 2) 輸入 Hinge Error「hinge」，是一種單邊誤差，不考慮負值 3) 輸入 Cross Entropy「categorical_crossentropy」，預測值與實際值愈相近，損失函數就愈小，反之差距很大，就會更影響損失函數的值。
Optimizer	RMSprop 使用普通的動量，而不是 Nesterov 動量。保留梯度移動平均值，並使用平均值來估計方差。 Adam 是一種可以替代傳統隨機梯度下降過程的一階優化算法，它能基於訓練數據疊代地更新神經網絡權重。
validation_split	用於在沒有提供驗證集的時候，按一定比例從訓練集中取出一部份作為驗證集
epochs	epochs 被定義為向前和向後傳播中所有批次的單次訓練疊代。
batch_size	每次修正的樣本數
callbacks	1) EarlyStopping: 用於指定每個 epoch 開始和結束時進行特定操作，可以減少在訓練時發生 Overfitting。 2) ModelCheckpoint: 模型在訓練過程中記錄最優模型的權重，避免訓練過程中發生錯誤。 3) ReduceLROnPlateau: 經過一定的 epoch 迭代後，模型效果不在提升，需要適度修改學習率，進而提升模型效果。

表 4 基礎模型架構表

模型	Parameters	Depth
ResNet50	25,636,712	-
DenseNet201	20,242,984	201
VGG16	138,357,544	23
InceptionResNetV2	55,873,736	572

3.3.1 模型設計

- CNN

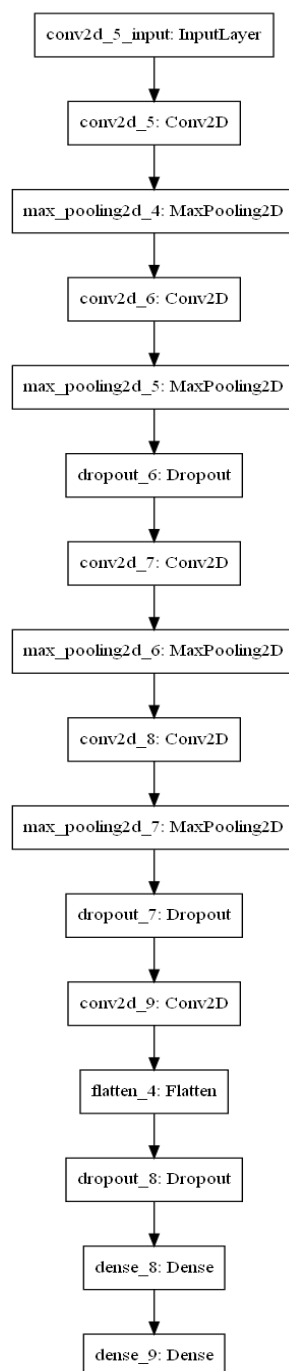


圖 1 CNN 模型架構圖

- DenseNet201

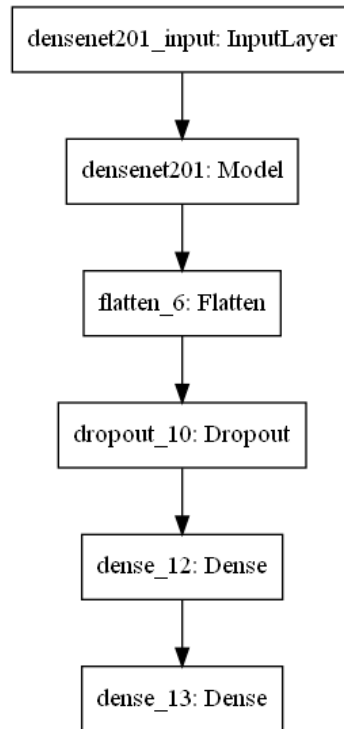


圖 2 DenseNet201 模型架構圖

- Resnet50

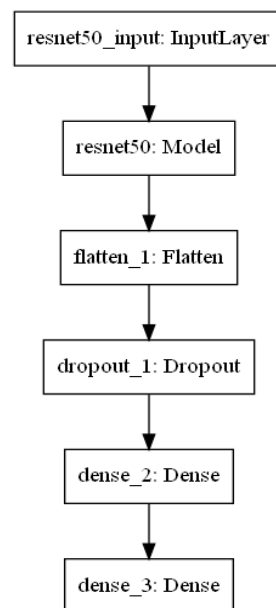


圖 3 Resnet50 模型架構圖

- VGG16

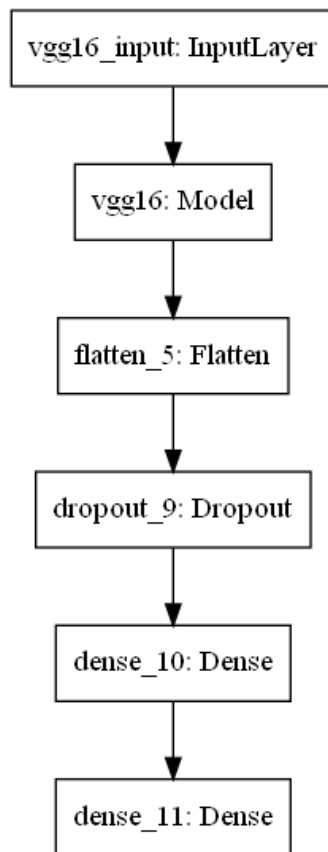


圖 4 VGG16 模型架構圖

3.5 實驗結果

Mango 資料集

- CNN 調整

建立一個基礎 CNN，模型架構如下

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 112, 112, 32)	896
max_pooling2d_1 (MaxPooling2D)	(None, 56, 56, 32)	0
conv2d_2 (Conv2D)	(None, 56, 56, 64)	18496
max_pooling2d_2 (MaxPooling2D)	(None, 28, 28, 64)	0
conv2d_3 (Conv2D)	(None, 28, 28, 64)	36928
max_pooling2d_3 (MaxPooling2D)	(None, 14, 14, 64)	0
conv2d_4 (Conv2D)	(None, 14, 14, 128)	73856
flatten_1 (Flatten)	(None, 25088)	0
dropout_1 (Dropout)	(None, 25088)	0
dense_1 (Dense)	(None, 512)	12845568
dense_2 (Dense)	(None, 3)	1539
Total params: 12,977,283		
Trainable params: 12,977,283		
Non-trainable params: 0		

圖 5 基礎 CNN 模型架構圖

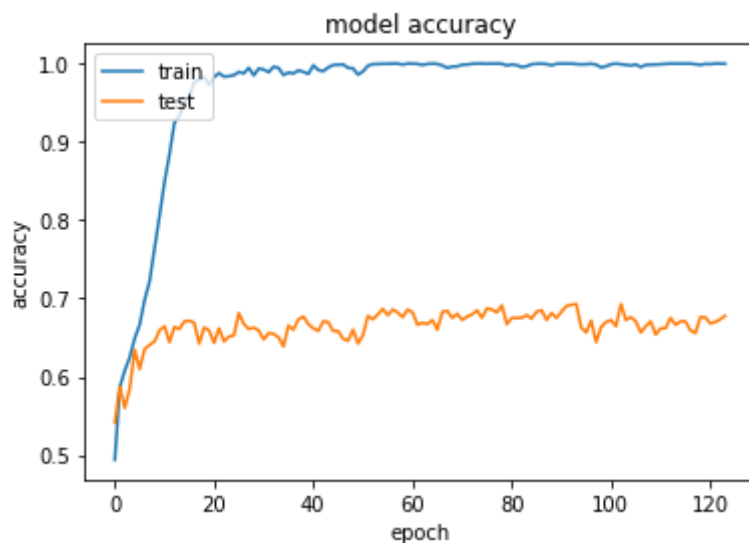


圖 6 基礎 CNN 模型 accuracy 圖

預測結果 accuracy 為 0.692，precision 為 0.700，recall 為 0.698，F1 為 0.699，結果不如預期，嘗試增加層數與使用 Dropout 進行分類，修改基礎 CNN 模型架構如下：

Layer (type)	Output Shape	Param #
conv2d_31 (Conv2D)	(None, 112, 112, 32)	896
max_pooling2d_25 (MaxPooling)	(None, 56, 56, 32)	0
conv2d_32 (Conv2D)	(None, 56, 56, 64)	18496
max_pooling2d_26 (MaxPooling)	(None, 28, 28, 64)	0
dropout_19 (Dropout)	(None, 28, 28, 64)	0
conv2d_33 (Conv2D)	(None, 28, 28, 64)	36928
max_pooling2d_27 (MaxPooling)	(None, 14, 14, 64)	0
conv2d_34 (Conv2D)	(None, 14, 14, 128)	73856
max_pooling2d_28 (MaxPooling)	(None, 7, 7, 128)	0
dropout_20 (Dropout)	(None, 7, 7, 128)	0
conv2d_35 (Conv2D)	(None, 7, 7, 128)	147584
flatten_11 (Flatten)	(None, 6272)	0
dropout_21 (Dropout)	(None, 6272)	0
dense_21 (Dense)	(None, 512)	3211776
dense_22 (Dense)	(None, 3)	1539
Total params: 3,491,075		
Trainable params: 3,491,075		
Non-trainable params: 0		

圖 7 修改後基礎 CNN 模型架構

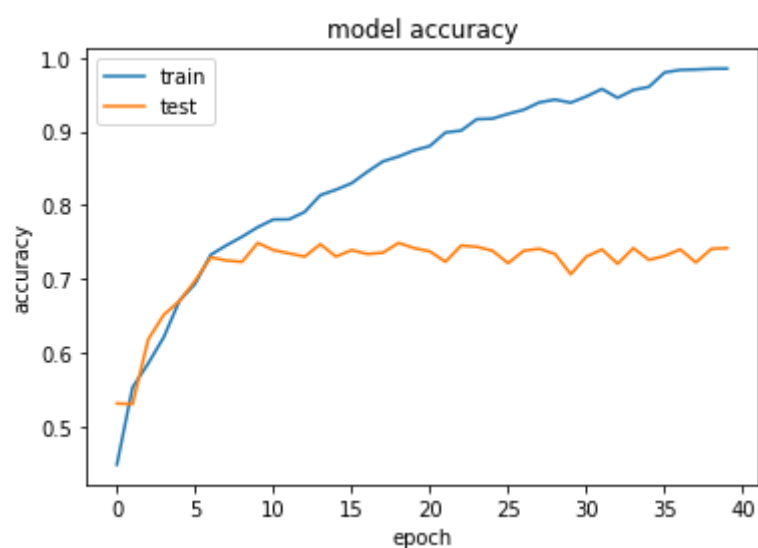


圖 8 修改後基礎 CNN 模型 accuracy 圖

預測結果提升許多，accuracy 為 0.767，precision 為 0.775，recall 為 0.785，F1 為 0.767。

由圖 7 修改後基礎 CNN 模型 accuracy 圖中發現有 overfitting 的問題，訓練精準度隨著時間呈現線性增加，而驗證準確度卻停留在 70~74%，為了解決 overfitting 本研究將使用資料增強法套用至本模型，透過資料增強從現有的訓練樣本中生成更多訓練資料，使用後基礎 CNN 模型有大幅提升，accuracy 為 0.784，precision 為 0.787，recall 為 0.794，F1 為 0.788，使用資料增強方式有助於讓 model 學習更多面向的資料，並得到更佳的普遍性。

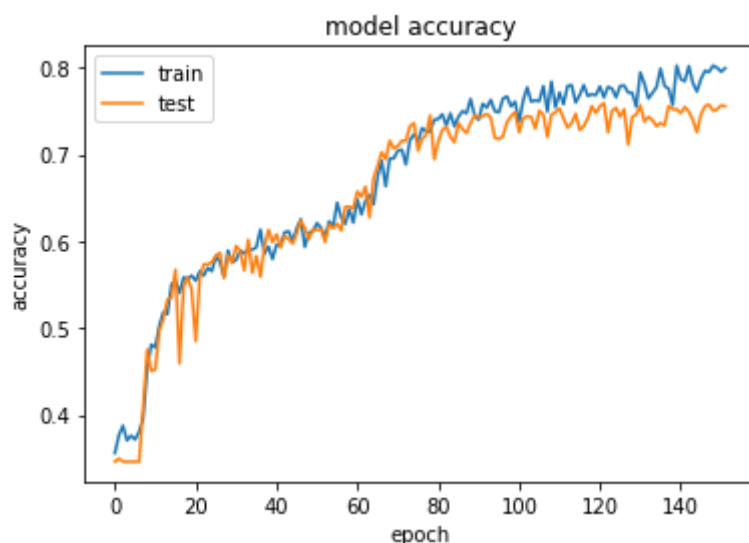


圖 9 使用資料增強後基礎 CNN 模型 accuracy 圖

使用不同 CNN 延伸模型進行比較測試，未使用資料增強時預測結果如下表

表 5 Mango 資料集實驗結果表

	Accuracy	Precision	Recall	F1
CNN	0.943/0.692	0.944/0.700	0.943/0.698	0.944/0.699
DenseNet201	0.955/0.756	0.956/0.757	0.955/0.766	0.955/0.760
Resnet50	0.952/ 0.804	0.952/ 0.804	0.954/ 0.816	0.953/ 0.805
VGG16	0.368/0.384	0.122/0.128	0.333/0.333	0.179/0.184

訓練資料/測試資料

未使用資料增強前預測結果為 Resnet50 在測試集中績效達到 0.8%，是在目前最高的績效 accuracy 為 0.804，precision 為 0.804，recall 為 0.816，F1 為 0.805。

使用不同 CNN 衍生模型進行比較測試，使用資料增強時預測結果如下表：

表 6 Mango 資料集資料增強實驗結果表

	Accuracy	Precision	Recall	F1
CNN	0.805 /0.784	0.809 /0.787	0.808 /0.794	0.808 /0.788
DenseNet201	0.892/ 0.828	0.893/ 0.828	0.895/ 0.837	0.894/ 0.830
Resnet50	0.883/0.82	0.884/0.819	0.886/0.830	0.885/0.822
VGG16	0.368/ 0.384	0.122/ 0.128	0.333/ 0.333	0.179/ 0.184

訓練資料/測試資料

使用資料增強後預測結果為 DenseNet201 在測試集中績效達到 0.82%，增加了 0.02%，accuracy 為 0.828，precision 為 0.828，recall 為 0.837，F1 為 0.830。

CIFAR10 資料集

使用不同 CNN 延伸模型進行比較測試，預測結果如下表：

表 7 CIFAR10 資料集實驗結果比較表

	Accuracy	Precision	Recall	F1
CNN	0.819/0.768	0.819/0.768	0.819/0.768	0.818/0.767
DenseNet201	0.795/ 0.858	0.799/ 0.859	0.795/ 0.858	0.792/ 0.858
Resnet50	0.904/0.830	0.904/0.830	0.904/0.830	0.904/0.829
VGG16	0.795/0.786	0.799/0.790	0.795/0.786	0.792/0.782

訓練資料/測試資料

預測結果最好的為 DenseNet201，accuracy 達到了 0.858，precision 為 0.859，recall 為 0.858，F1 為 0.858。

第四章、結論

本研究使用 Mango 與 CIFAR10 資料集進行深度學習 CNN 圖像分類預測。使用不同 CNN 延伸模型進行比較結果為 DenseNet201 績效達到最優秀，在 CIFAR10 資料集中 accuracy 達到了 0.858，precision 為 0.859，recall 為 0.858、F1 為 0.858。而在 Mango 資料集中使用資料增強後 accuracy 為 0.828、precision 為 0.828、recall 為 0.837、F1 為 0.830。製作基礎 CNN 時發現適當加入 Dropout 使訓練期間隨機丟棄 layer 的一些輸出特徵，在測試層時的輸出值能夠依照丟棄率的比率縮小，以平衡訓練時輸出被歸零的影響，使總數值不會偏差太大。在訓練基礎模型時發現有 overfitting 的問題，為解決此問題本研究使用資料增強方式由現有訓練樣本中生成更多訓練資料，使模型能夠得到更佳的普遍性。

參考資料:

ResNet50.Retrieved.from: <https://keras.io/api/applications/resnet/>

DenseNet201.Retrieved from: <https://keras.io/api/applications/densenet/>

VGG16.Retrieved from: <https://keras.io/api/applications/vgg/>

InceptionResNetV2.Retrieved from:

<https://keras.io/api/applications/inceptionresnetv2/>

OpenCV.Retrieved.from: https://docs.opencv.org/master/d6/d00/tutorial_py_root.html

Convolutional Neural Network (CNN) . Retrieved from:

<https://ithelp.ithome.com.tw/articles/10205389>

EarlyStopping.Retrieved from: https://keras.io/api/callbacks/early_stopping/

ModelCheckpoint.Retrieved from:

https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/ModelCheckpoint

ImageDataGenerator.Retrieved from: <https://keras.io/api/preprocessing/image/>

Sklearn.metrics.precision_score.Retrieved from:

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html#sklearn-metrics-precision-score

Sklearn.metrics.recall_score.Retrieved from:

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html

Sklearn.metrics.f1_score. Retrieved from:

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html