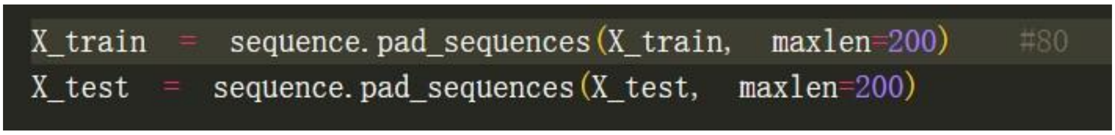
深度學習期末專題　　M1233011 溫紫宸

專題 1

- 專題名稱：Sentiment_Analysis_using_RNN
- 來源網址：https://github.com/navneetkrc/Keras_IMDB/blob/master/Sentiment_Analysis_using_RNN.ipynb
- 任務描述：用 LSTM 神經網路對 IMDB 電影評論資料集進行二元分類。
- 使用資料集：IMDB
- 修改部分解說：

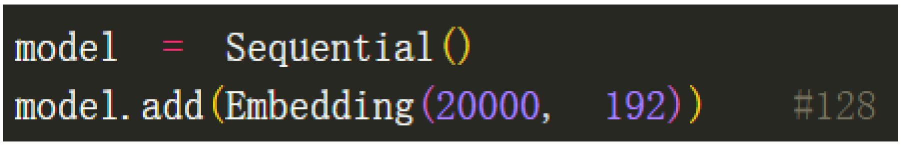
1. 提升 padding 長度



```
X_train = sequence.pad_sequences(X_train, maxlen=200)  #80
X_test = sequence.pad_sequences(X_test, maxlen=200)
```

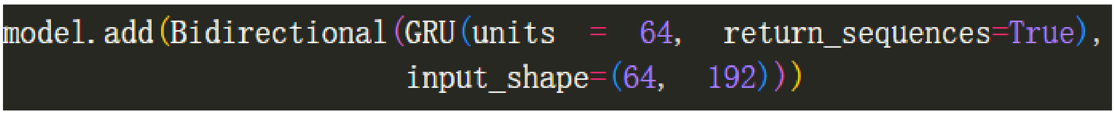
→原先設定 maxlen 為 80，可能不足以包含所有句子資訊，因此將其提升為 200。

2. 模型修改



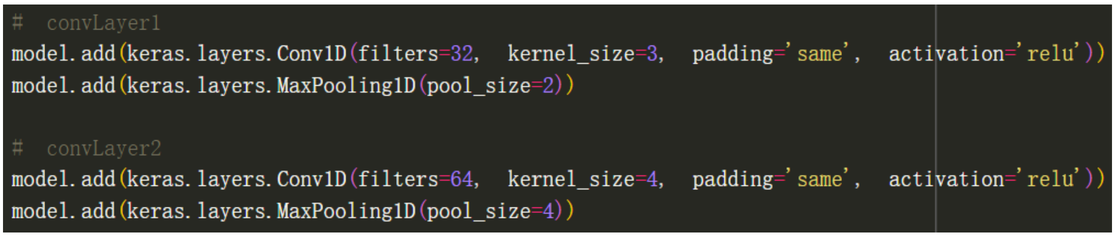
```
model = Sequential()
model.add(Embedding(20000, 192))  #128
```

→將 Embedding 的維度長度由 128 提升為 192，讓詞嵌入的向量能更準確表達句子的意思。



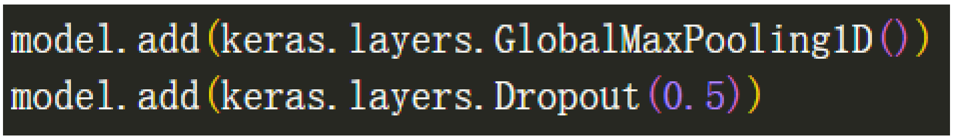
```
model.add(Bidirectional(GRU(units = 64, return_sequences=True),
                        input_shape=(64, 192)))
```

→使用 GRU 替代 LSTM 以減少模型運算時間，並採用雙向方式來避免循環神經網路的梯度消失問題。



```
# convLayer1
model.add(keras.layers.Conv1D(filters=32, kernel_size=3, padding='same', activation='relu'))
model.add(keras.layers.MaxPooling1D(pool_size=2))

# convLayer2
model.add(keras.layers.Conv1D(filters=64, kernel_size=4, padding='same', activation='relu'))
model.add(keras.layers.MaxPooling1D(pool_size=4))
```

→增加兩層卷積層去提取特徵。



```
model.add(keras.layers.GlobalMaxPooling1D())
model.add(keras.layers.Dropout(0.5))
```

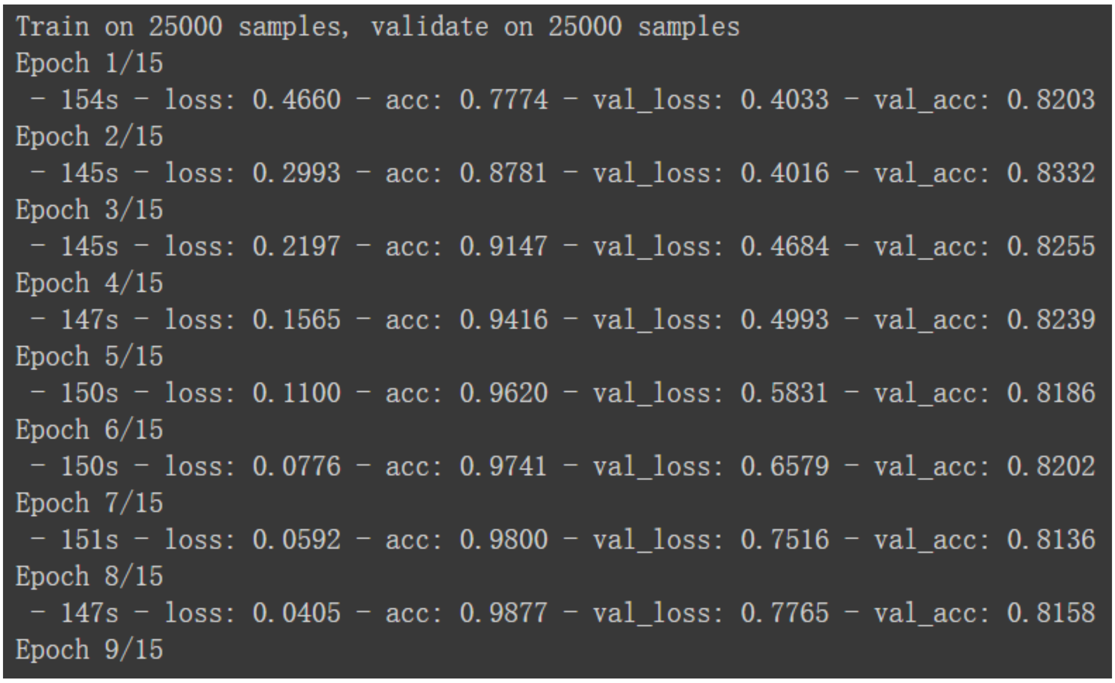
→使用 GlobalMaxPooling 進一步壓縮特徵，再增加一 Dropout 層，避免過度

擬合。

- 訓練結果：

  ➢修改前

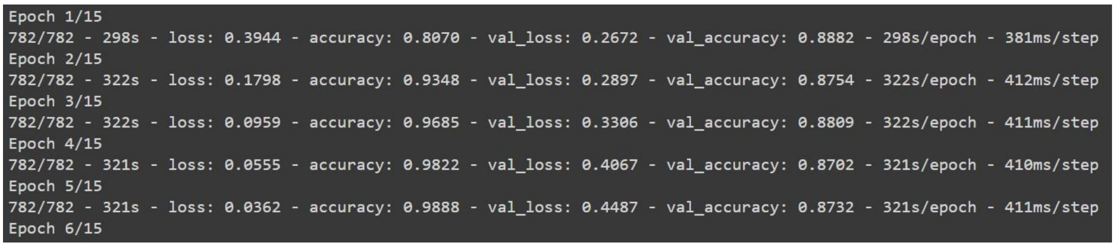
  最高 val_acc 為 0.8332。



```
Train on 25000 samples, validate on 25000 samples
Epoch 1/15
 - 154s - loss: 0.4660 - acc: 0.7774 - val_loss: 0.4033 - val_acc: 0.8203
Epoch 2/15
 - 145s - loss: 0.2993 - acc: 0.8781 - val_loss: 0.4016 - val_acc: 0.8332
Epoch 3/15
 - 145s - loss: 0.2197 - acc: 0.9147 - val_loss: 0.4684 - val_acc: 0.8255
Epoch 4/15
 - 147s - loss: 0.1565 - acc: 0.9416 - val_loss: 0.4993 - val_acc: 0.8239
Epoch 5/15
 - 150s - loss: 0.1100 - acc: 0.9620 - val_loss: 0.5831 - val_acc: 0.8186
Epoch 6/15
 - 150s - loss: 0.0776 - acc: 0.9741 - val_loss: 0.6579 - val_acc: 0.8202
Epoch 7/15
 - 151s - loss: 0.0592 - acc: 0.9800 - val_loss: 0.7516 - val_acc: 0.8136
Epoch 8/15
 - 147s - loss: 0.0405 - acc: 0.9877 - val_loss: 0.7765 - val_acc: 0.8158
Epoch 9/15
```

  ➢修改後

  最高 val_acc 為 0.8882。



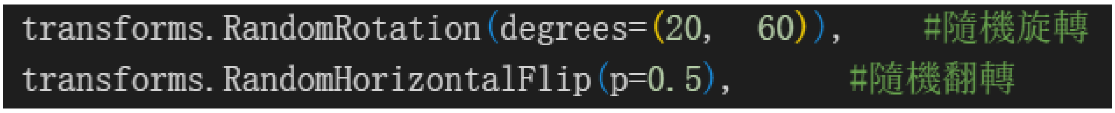
```
Epoch 1/15
782/782 - 298s - loss: 0.3944 - accuracy: 0.8070 - val_loss: 0.2672 - val_accuracy: 0.8882 - 298s/epoch - 381ms/step
Epoch 2/15
782/782 - 322s - loss: 0.1798 - accuracy: 0.9348 - val_loss: 0.2897 - val_accuracy: 0.8754 - 322s/epoch - 412ms/step
Epoch 3/15
782/782 - 322s - loss: 0.0959 - accuracy: 0.9685 - val_loss: 0.3306 - val_accuracy: 0.8809 - 322s/epoch - 411ms/step
Epoch 4/15
782/782 - 321s - loss: 0.0555 - accuracy: 0.9822 - val_loss: 0.4067 - val_accuracy: 0.8702 - 321s/epoch - 410ms/step
Epoch 5/15
782/782 - 321s - loss: 0.0362 - accuracy: 0.9888 - val_loss: 0.4487 - val_accuracy: 0.8732 - 321s/epoch - 411ms/step
Epoch 6/15
```

  →最高準確度提升約 0.055。

專題 2

- 專題名稱：ML2023-HW3-ImageClassification
- 來源網址： https://speech.ee.ntu.edu.tw/~hylee/ml/2023-spring.php
- 任務描述：用 CNN 卷積神經網路對 food11 資料集進行圖像分類。
- 使用資料集：food11
- 修改部分解說：

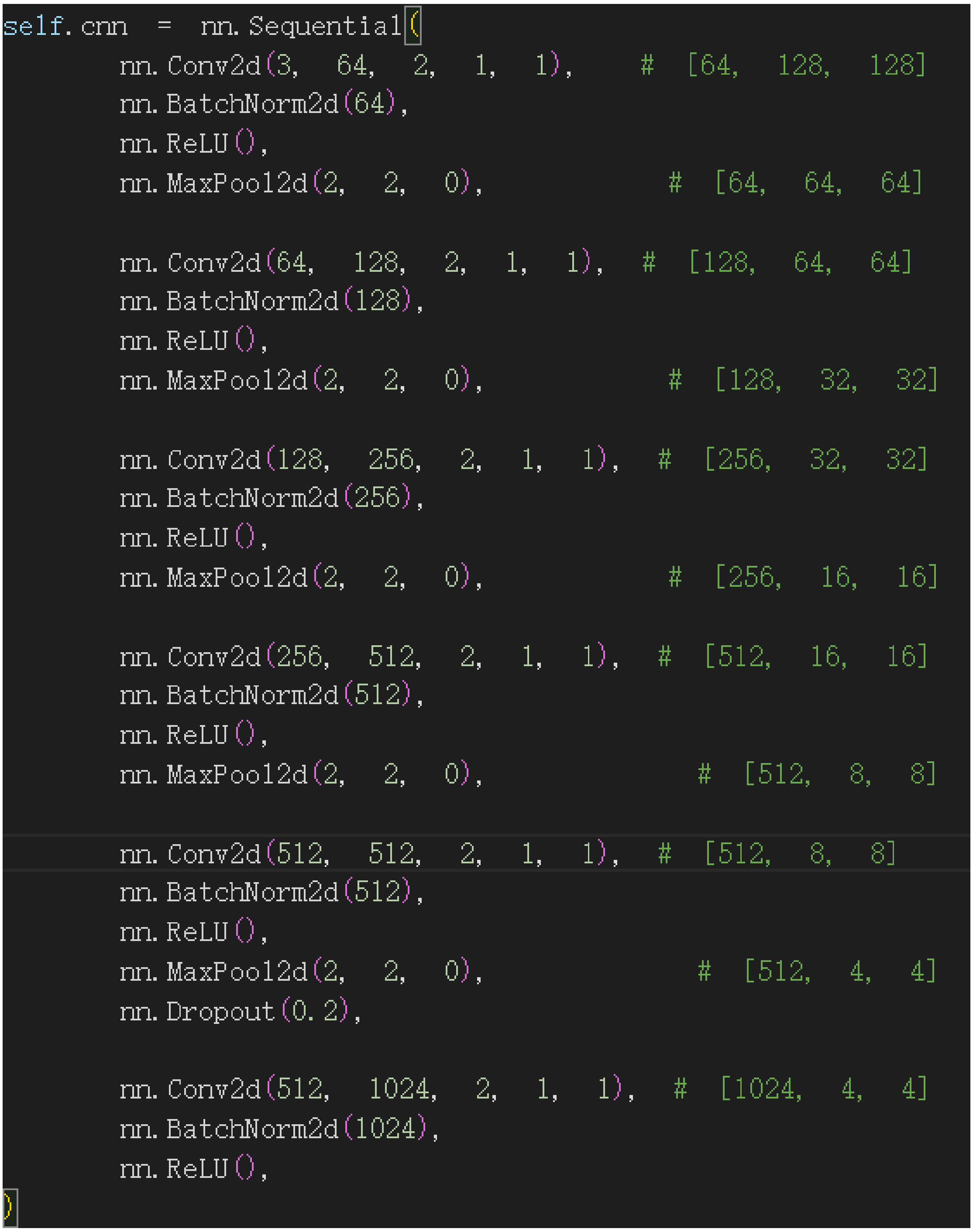
  1. 資料增強



```
transforms.RandomRotation(degrees=(20,  60)),      #隨機旋轉
transforms.RandomHorizontalFlip(p=0.5),          #隨機翻轉
```

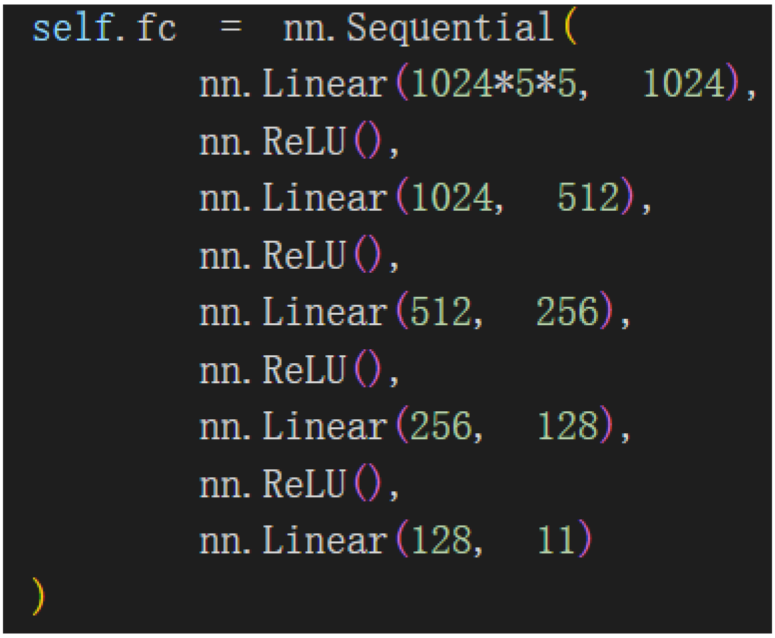
→加入兩種資料增強方法，對訓練資料進行隨機旋轉及隨機翻轉。

2. 模型修改



```
self.cnn = nn.Sequential(
        nn.Conv2d(3, 64, 2, 1, 1),      # [64, 128, 128]
        nn.BatchNorm2d(64),
        nn.ReLU(),
        nn.MaxPool2d(2, 2, 0),           # [64, 64, 64]

        nn.Conv2d(64, 128, 2, 1, 1),    # [128, 64, 64]
        nn.BatchNorm2d(128),
        nn.ReLU(),
        nn.MaxPool2d(2, 2, 0),           # [128, 32, 32]

        nn.Conv2d(128, 256, 2, 1, 1),   # [256, 32, 32]
        nn.BatchNorm2d(256),
        nn.ReLU(),
        nn.MaxPool2d(2, 2, 0),           # [256, 16, 16]

        nn.Conv2d(256, 512, 2, 1, 1),   # [512, 16, 16]
        nn.BatchNorm2d(512),
        nn.ReLU(),
        nn.MaxPool2d(2, 2, 0),            # [512, 8, 8]

        nn.Conv2d(512, 512, 2, 1, 1),   # [512, 8, 8]
        nn.BatchNorm2d(512),
        nn.ReLU(),
        nn.MaxPool2d(2, 2, 0),            # [512, 4, 4]
        nn.Dropout(0.2),

        nn.Conv2d(512, 1024, 2, 1, 1),  # [1024, 4, 4]
        nn.BatchNorm2d(1024),
        nn.ReLU(),
)
```

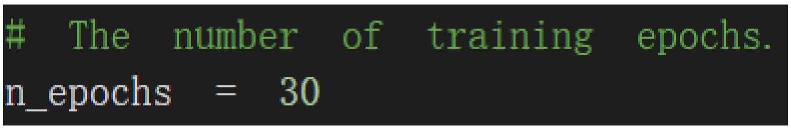
→將 Kernel size 由 3 改為 2，以捕捉更細節的特徵。並增加一 Dropout 層避免過度擬合，最後再加一層卷積層進一步提取特徵。



```
self.fc = nn.Sequential(
        nn.Linear(1024*5*5, 1024),
        nn.ReLU(),
        nn.Linear(1024, 512),
        nn.ReLU(),
        nn.Linear(512, 256),
        nn.ReLU(),
        nn.Linear(256, 128),
        nn.ReLU(),
        nn.Linear(128, 11)
)
```

→增加神經網路層數及神經元個數。

3. 提升訓練週期



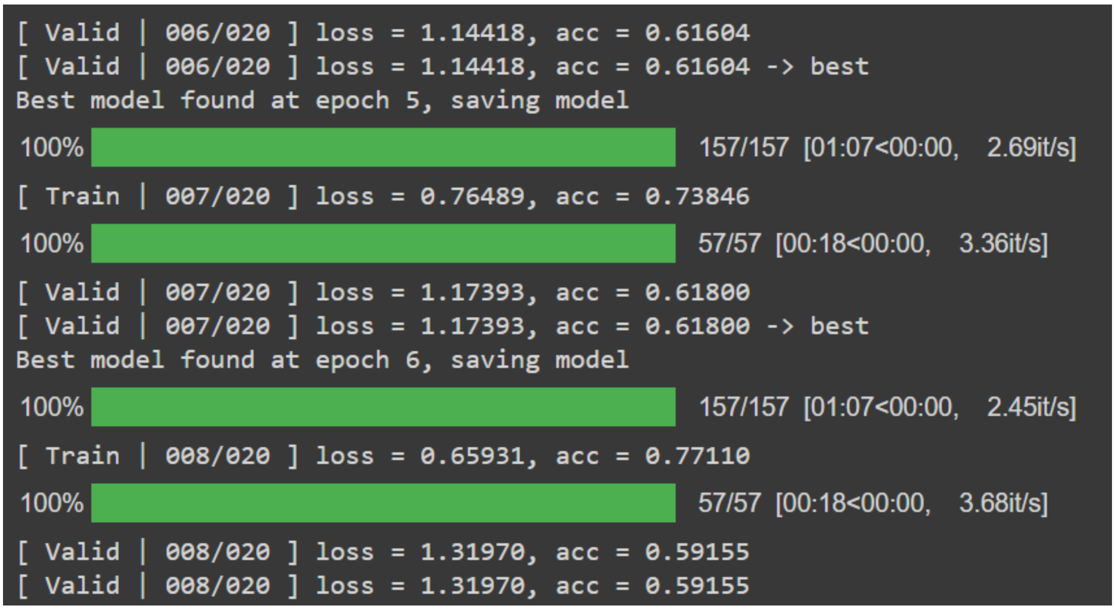
```
# The number of training epochs.
n_epochs = 30
```

→將訓練週期由原先的 8 提升到 30。

- 訓練結果：

➢ 修改前

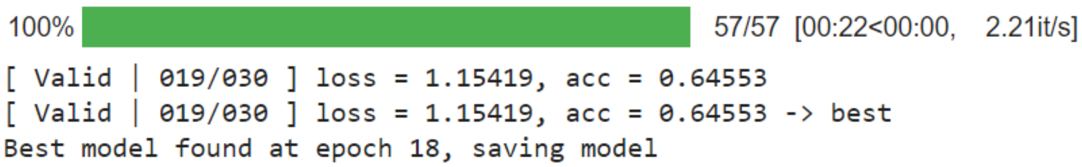
最高 val_acc 為 0.61800。



```
[ Valid | 006/020 ] loss = 1.14418, acc = 0.61604
[ Valid | 006/020 ] loss = 1.14418, acc = 0.61604 -> best
Best model found at epoch 5, saving model
100% ████████████████████ 157/157 [01:07<00:00, 2.69it/s]
[ Train | 007/020 ] loss = 0.76489, acc = 0.73846
100% ████████████████████ 57/57 [00:18<00:00, 3.36it/s]
[ Valid | 007/020 ] loss = 1.17393, acc = 0.61800
[ Valid | 007/020 ] loss = 1.17393, acc = 0.61800 -> best
Best model found at epoch 6, saving model
100% ████████████████████ 157/157 [01:07<00:00, 2.45it/s]
[ Train | 008/020 ] loss = 0.65931, acc = 0.77110
100% ████████████████████ 57/57 [00:18<00:00, 3.68it/s]
[ Valid | 008/020 ] loss = 1.31970, acc = 0.59155
[ Valid | 008/020 ] loss = 1.31970, acc = 0.59155
```

➢ 修改後

最高 val_acc 為 0.64553。



```
100%  57/57 [00:22<00:00,  2.21it/s]
[ Valid | 019/030 ] loss = 1.15419, acc = 0.64553
[ Valid | 019/030 ] loss = 1.15419, acc = 0.64553 -> best
Best model found at epoch 18, saving model
```

→最高準確度提升約 0.02753。