操作方法：執行 104503530.py 的檔案即可

內容說明：此作業有兩個 python 檔案，其中的 104503530_pretrain.py 將模型以 mnist 資料集訓練 15 個 epoch，產生的 top_model_weight.h5 用意是提供訓練新模型的參數初始值，研究指出一個給定初始值的模型會比隨機值開始有著更好的表現。

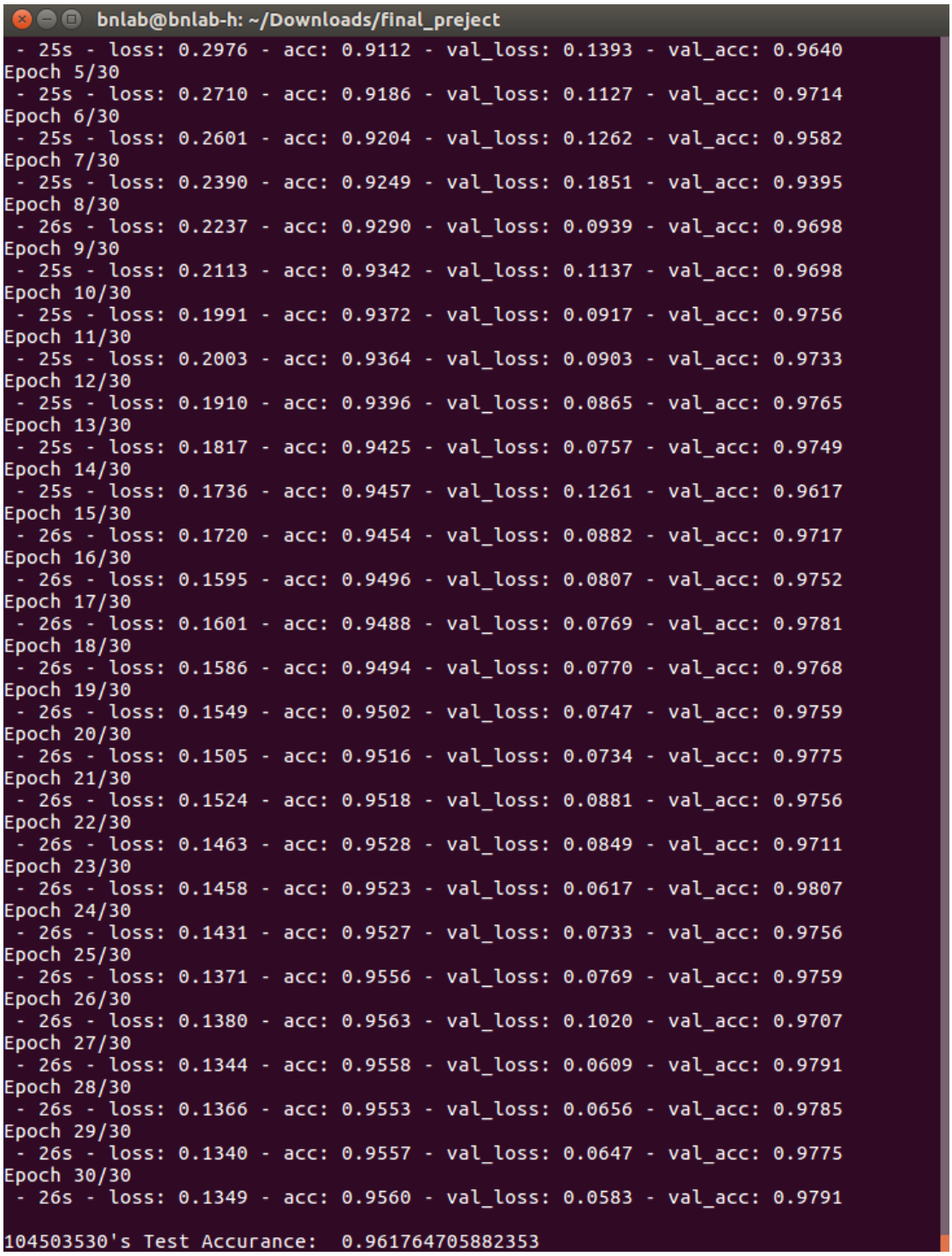
下圖為在 mnist 上訓練的結果圖：

```
/usr/local/lib/python3.5/dist-packages/h5py/__init__.py:36: FutureWarning: Conve
rsion of the second argument of issubdtype from `float` to `np.floating` is depr
ecated. In future, it will be treated as `np.float64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
2018-06-08 22:41:42.668685: I tensorflow/core/platform/cpu_feature_guard.cc:137]
 Your CPU supports instructions that this TensorFlow binary was not compiled to
use: SSE4.1 SSE4.2
2018-06-08 22:41:42.926192: I tensorflow/stream_executor/cuda/cuda_gpu_executor.
cc:895] successful NUMA node read from SysFS had negative value (-1), but there
must be at least one NUMA node, so returning NUMA node zero
2018-06-08 22:41:42.926631: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1
105] Found device 0 with properties:
name: GeForce GTX 1060 6GB major: 6 minor: 1 memoryClockRate(GHz): 1.759
pciBusID: 0000:02:00.0
totalMemory: 5.93GiB freeMemory: 5.52GiB
2018-06-08 22:41:42.926660: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1
195] Creating TensorFlow device (/device:GPU:0) -> (device: 0, name: GeForce GTX
 1060 6GB, pci bus id: 0000:02:00.0, compute capability: 6.1)
Train on 60000 samples, validate on 10000 samples
Epoch 1/15
 - 12s - loss: 0.3918 - acc: 0.9492 - val_loss: 0.2319 - val_acc: 0.9704
Epoch 2/15
 - 10s - loss: 0.1677 - acc: 0.9796 - val_loss: 0.1252 - val_acc: 0.9806
Epoch 3/15
 - 10s - loss: 0.1054 - acc: 0.9851 - val_loss: 0.0976 - val_acc: 0.9829
Epoch 4/15
 - 10s - loss: 0.0720 - acc: 0.9893 - val_loss: 0.0746 - val_acc: 0.9838
Epoch 5/15
 - 11s - loss: 0.0540 - acc: 0.9918 - val_loss: 0.0726 - val_acc: 0.9819
Epoch 6/15
 - 10s - loss: 0.0402 - acc: 0.9939 - val_loss: 0.0739 - val_acc: 0.9827
Epoch 7/15
 - 10s - loss: 0.0320 - acc: 0.9949 - val_loss: 0.0569 - val_acc: 0.9836
Epoch 8/15
 - 10s - loss: 0.0254 - acc: 0.9963 - val_loss: 0.0605 - val_acc: 0.9825
Epoch 9/15
 - 10s - loss: 0.0220 - acc: 0.9964 - val_loss: 0.0572 - val_acc: 0.9840
Epoch 10/15
 - 10s - loss: 0.0168 - acc: 0.9974 - val_loss: 0.0603 - val_acc: 0.9818
Epoch 11/15
 - 10s - loss: 0.0148 - acc: 0.9976 - val_loss: 0.0530 - val_acc: 0.9839
Epoch 12/15
 - 10s - loss: 0.0132 - acc: 0.9978 - val_loss: 0.0526 - val_acc: 0.9840
Epoch 13/15
 - 10s - loss: 0.0132 - acc: 0.9975 - val_loss: 0.0530 - val_acc: 0.9840
Epoch 14/15
 - 10s - loss: 0.0099 - acc: 0.9983 - val_loss: 0.0540 - val_acc: 0.9847
Epoch 15/15
 - 10s - loss: 0.0082 - acc: 0.9988 - val_loss: 0.0574 - val_acc: 0.9834
```

104503530_pretrain_mnist.png

接著是主程式104503530.py，由於此次訓練資料集數量稀少且圖片解析度只有28*28，設計模型結構上以簡單為主，共只用了兩個卷積層Conv2D、一個池化層MaxPooling2D和兩個全連結層Dense，並在每個線性層與非線性轉換層(Activation Function)加入歸一化層BatchNormalization，原因為神經網路除了輸出層外，其它層因為低層網路在訓練的時候更新了參數，而引起後面層輸入數據分佈的變化，因此若在神經網路的每一層輸入前插入了一個歸一化層，也就是將數據歸一化至均值為0、方差為1，再進入網路的下一層訓練，可有效減少後層網路輸入的劇烈變動，在不會過擬合狀況下增加了準確率。

在訓練資料的處理上，使用了Data Augmentation將訓練資料數量增大，將fit函數取代成fit_generator，並透過keras內建的ImageDataGenerator函數隨機變換圖片(旋轉、平移、縮放……)，如此可大大增加訓練資料量和隨機性，以下為訓練了30個epoch後的結果。



104503530.png

上圖驗證資料的 val_acc 比訓練資料的 acc 比高的原因是，模型中加入了 Dropout 層，雖然稍微降低對訓練資料的擬合，卻提升了模型的泛化能力。