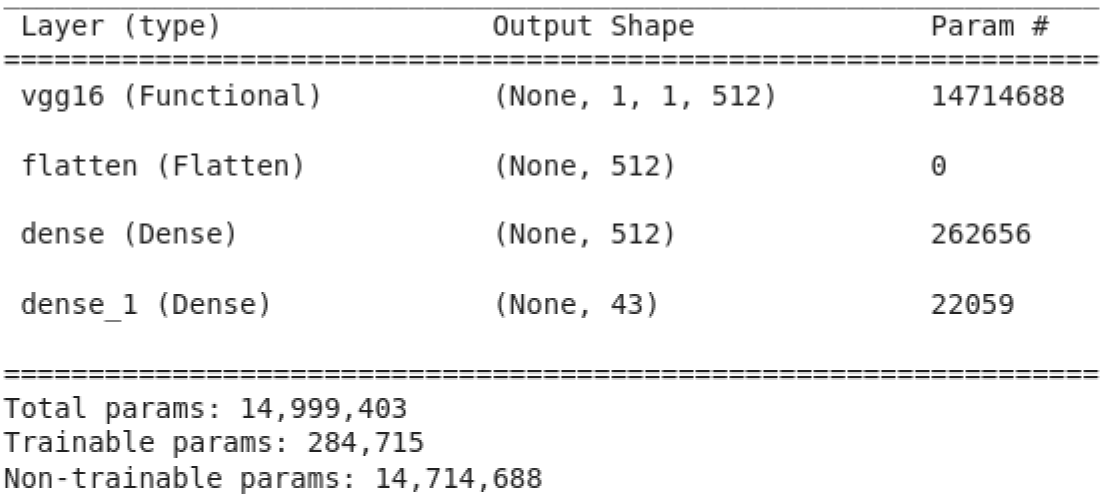
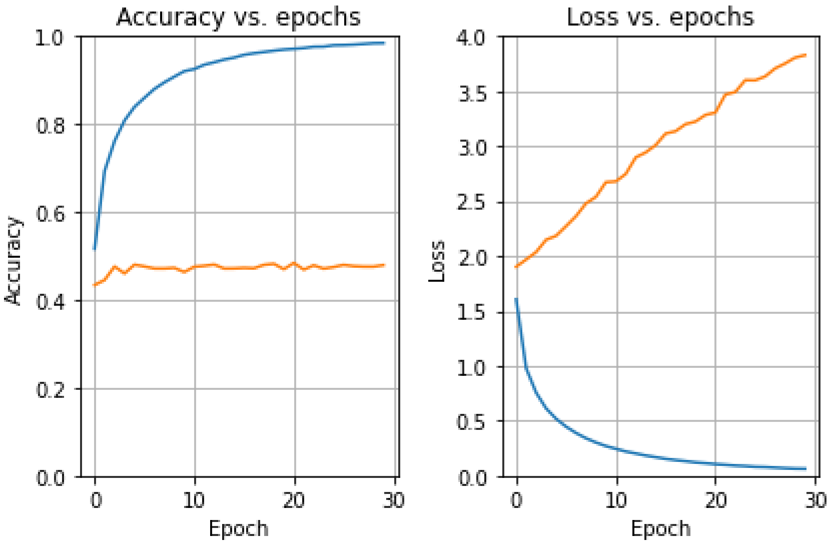
Wenguang Hu A01706648

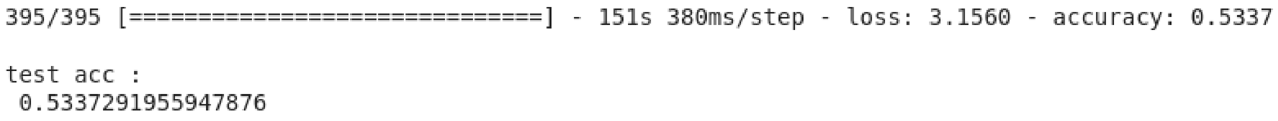
My model is going to do the classification of the Traffic Signal.

Model1
At beginning, I put vgg16 + 2 MLP
Then I saw while Loss of Training is going down, and Loss of Validation is going up.
It means we are having over fit.



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 1, 1, 512)         14714688

 flatten (Flatten)           (None, 512)               0

 dense (Dense)               (None, 512)               262656

 dense_1 (Dense)             (None, 43)                22059

=================================================================
Total params: 14,999,403
Trainable params: 284,715
Non-trainable params: 14,714,688
```





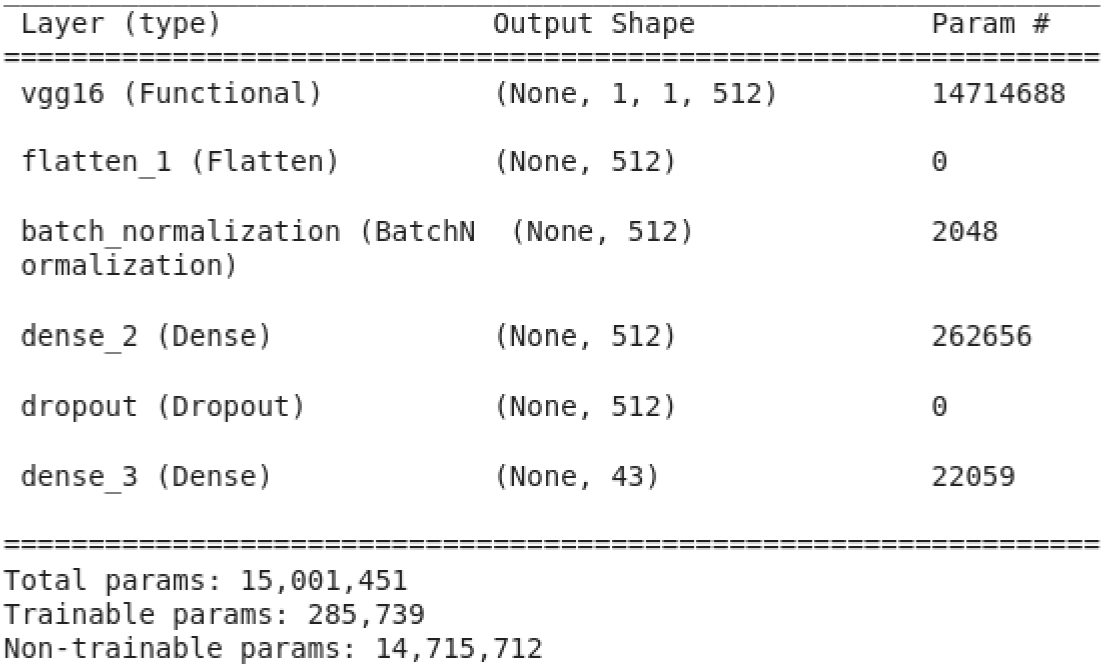
```
395/395 [==============================] - 151s 380ms/step - loss: 3.1560 - accuracy: 0.5337

test acc :
 0.5337291955947876
```

Wenguang Hu A01706648

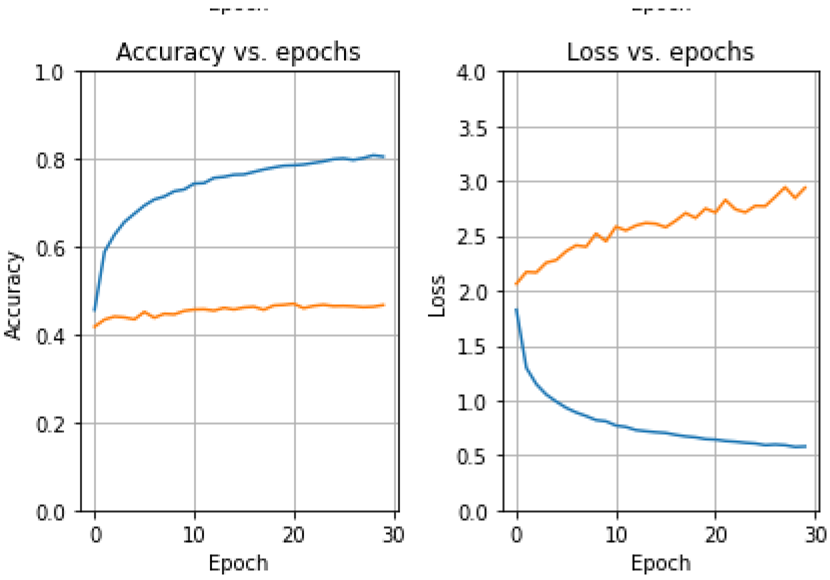
2, I tried to decrease the overfit, then I added a normalization after vgg16, and add one dropout layer between the Dense

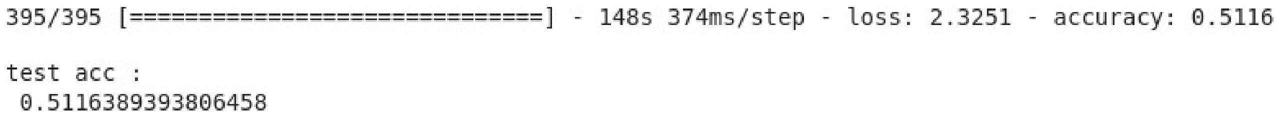
I used VGG16 + Normalization + Flatten + Dense + Drop + Dense

The Loss of validation seems not increase that much, but loss of validation is still increasing a lot while the training loss is decrease, so the model is still overfit.



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 1, 1, 512)         14714688

 flatten_1 (Flatten)         (None, 512)               0

 batch_normalization (BatchN  (None, 512)              2048
 ormalization)

 dense_2 (Dense)             (None, 512)               262656

 dropout (Dropout)           (None, 512)               0

 dense_3 (Dense)             (None, 43)                22059

=================================================================
Total params: 15,001,451
Trainable params: 285,739
Non-trainable params: 14,715,712
```





```
395/395 [==============================] - 148s 374ms/step - loss: 2.3251 - accuracy: 0.5116

test acc :
 0.5116389393806458
```

Wenguang Hu A01706648

3, After analysis,
I think due to our Image Input size is 32 X 32,
and VGG16 has 5 layers of pooling,
after pooling, we only have one pixel for classification,

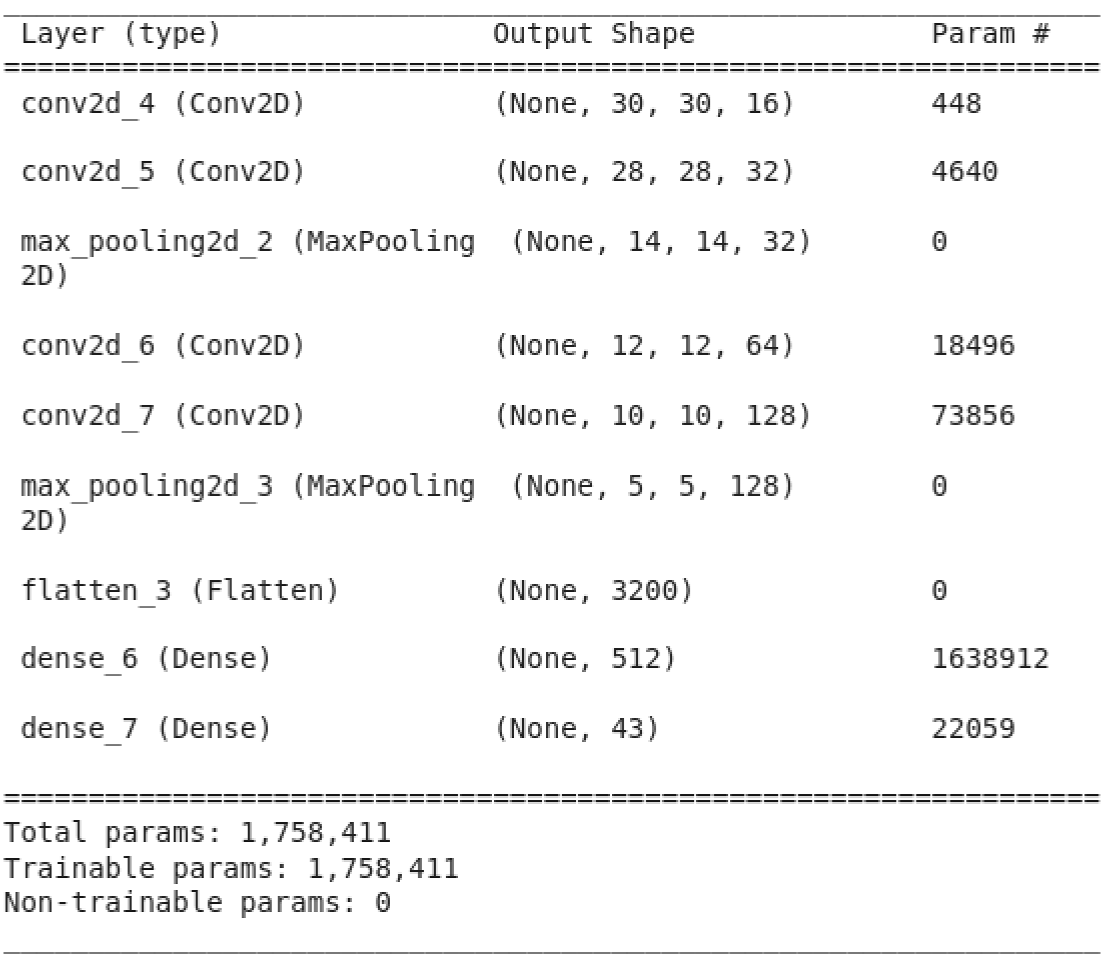
There is too little information for the classification,
it will cause the overfit by the architecture.

Change the Architecture, 2 x (2 x CNN + pooling) + flatten + 2 x MLP

So we changed the architecture,
Reduce the Pooling Layer and CCN layer. This time we only do pooling twice

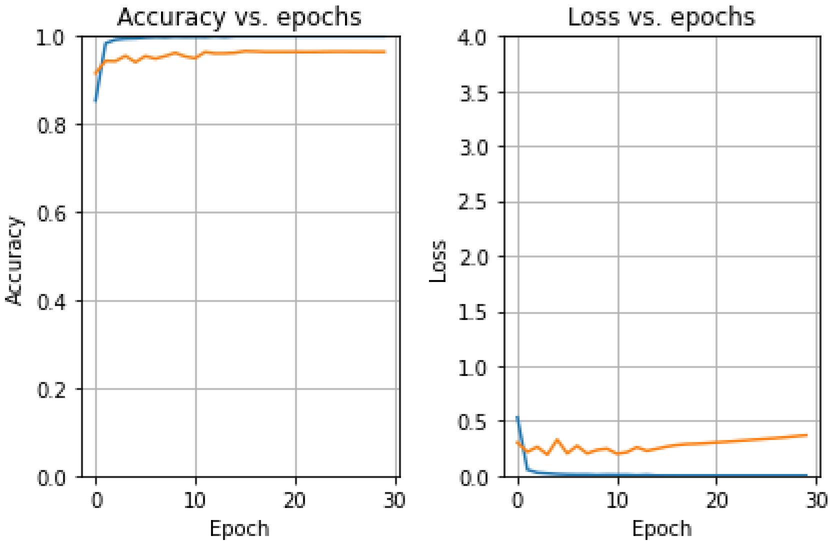
This time, we have much better validation accuracy, and the loss of validation only increase slightly after 10 epochs

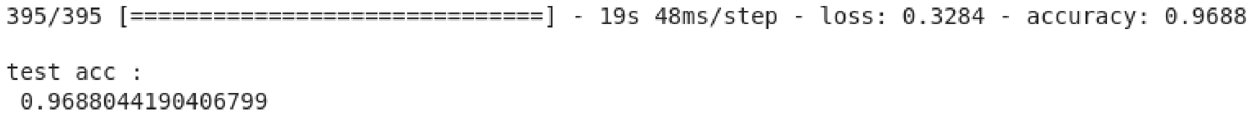
So we have slightly overfit in this module, but the accuracy of Training and Validation have been improved a lot.



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 30, 30, 16)        448

 conv2d_5 (Conv2D)           (None, 28, 28, 32)        4640

 max_pooling2d_2 (MaxPooling  (None, 14, 14, 32)       0
 2D)

 conv2d_6 (Conv2D)           (None, 12, 12, 64)        18496

 conv2d_7 (Conv2D)           (None, 10, 10, 128)       73856

 max_pooling2d_3 (MaxPooling  (None, 5, 5, 128)        0
 2D)

 flatten_3 (Flatten)         (None, 3200)              0

 dense_6 (Dense)             (None, 512)               1638912

 dense_7 (Dense)             (None, 43)                22059

=================================================================
Total params: 1,758,411
Trainable params: 1,758,411
Non-trainable params: 0
_________________________________________________________________
```

Wenguang Hu A01706648





```
395/395 [==============================] - 19s 48ms/step - loss: 0.3284 - accuracy: 0.9688

test acc :
 0.9688044190406799
```

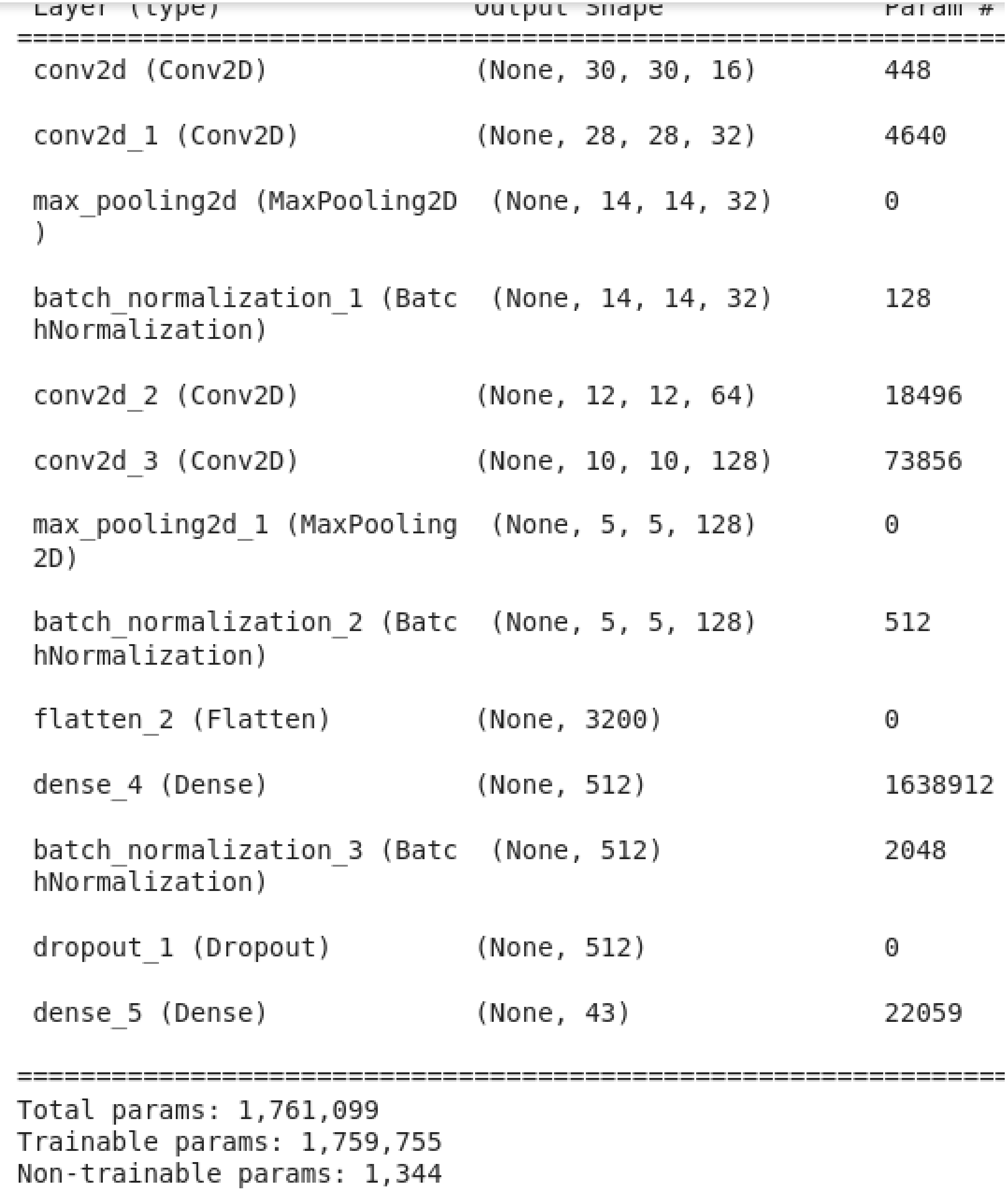
Wenguang Hu A01706648

4, In order to reduce the slightly overfit from pervious model,
I added normalization after each Pooling Layer,
I added a normalization and dropout layer before the last dense layer.

(2 x CNN + pooling) + normalization + (2 x CNN + pooling) + normalization + flatten
+ MLP + normalization + dropout + MLP

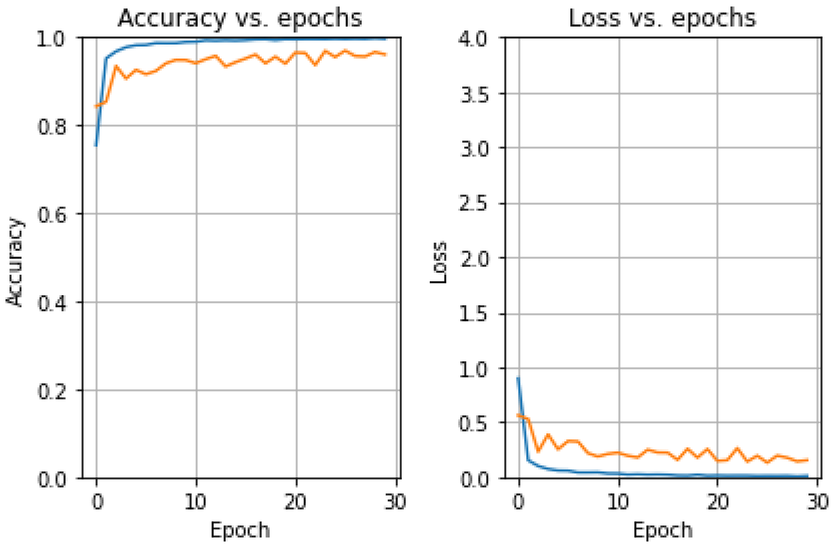
After the training,
The Accuracy of Training and Validation are both high, above 95%
The Accuracy of Validation is increasing while Accuracy of Validation is increasing
and the Loss of Validation is decreasing while Loss of Training is decreasing

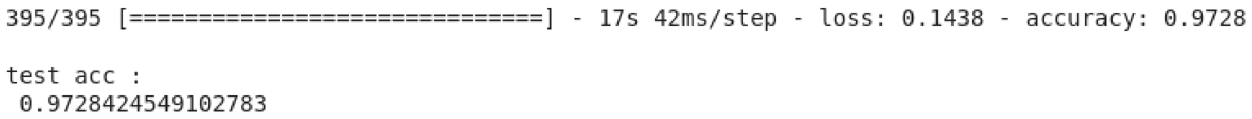
So we having a model just fit now.



```
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 30, 30, 16)        448

 conv2d_1 (Conv2D)           (None, 28, 28, 32)        4640

 max_pooling2d (MaxPooling2D  (None, 14, 14, 32)       0
 )

 batch_normalization_1 (Batc  (None, 14, 14, 32)       128
 hNormalization)

 conv2d_2 (Conv2D)           (None, 12, 12, 64)        18496

 conv2d_3 (Conv2D)           (None, 10, 10, 128)       73856

 max_pooling2d_1 (MaxPooling  (None, 5, 5, 128)        0
 2D)

 batch_normalization_2 (Batc  (None, 5, 5, 128)        512
 hNormalization)

 flatten_2 (Flatten)         (None, 3200)              0

 dense_4 (Dense)             (None, 512)               1638912

 batch_normalization_3 (Batc  (None, 512)              2048
 hNormalization)

 dropout_1 (Dropout)         (None, 512)               0

 dense_5 (Dense)             (None, 43)                22059

=================================================================
Total params: 1,761,099
Trainable params: 1,759,755
Non-trainable params: 1,344
```

Wenguang Hu A01706648





```
395/395 [==============================] - 17s 42ms/step - loss: 0.1438 - accuracy: 0.9728

test acc :
 0.9728424549102783
```

So our final model has improve the test accuracy from 3rd model’s 96.88% to 97.28% and reduce the slightly overfit.