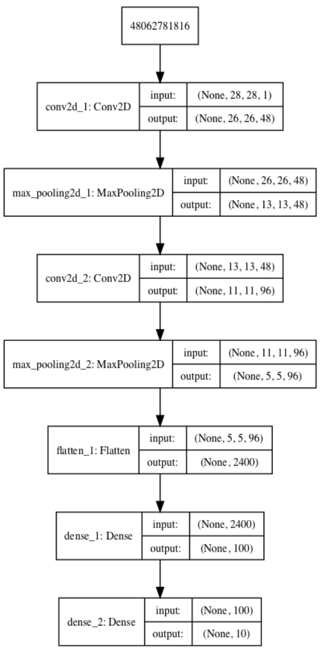
Task 3: Big CNN

```
cnn_model = models.Sequential()
cnn_model.add(Conv2D(48, kernel_size=(3,
3),activation='relu',input_shape=(img_rows,img_cols,1),padding='valid',strides=(1, 1)))
cnn_model.add(MaxPooling2D((2, 2)))
cnn_model.add(Conv2D(96, kernel_size=(3,
3),activation='relu',input_shape=(img_rows,img_cols,1),padding='valid',strides=(1, 1)))
cnn_model.add(MaxPooling2D((2, 2)))
cnn_model.add(Flatten())
cnn_model.add(layers.Dense(100, activation = "relu"))
cnn_model.add(layers.Dense(10, activation = "softmax"))
cnn_model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 26, 26, 48)        480
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 13, 13, 48)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 11, 11, 96)        41568
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 5, 5, 96)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 2400)              0
_________________________________________________________________
dense_1 (Dense)              (None, 100)               240100
_________________________________________________________________
dense_2 (Dense)              (None, 10)                1010
=================================================================
Total params: 283,158
Trainable params: 283,158
Non-trainable params: 0
_________________________________________________________________
```

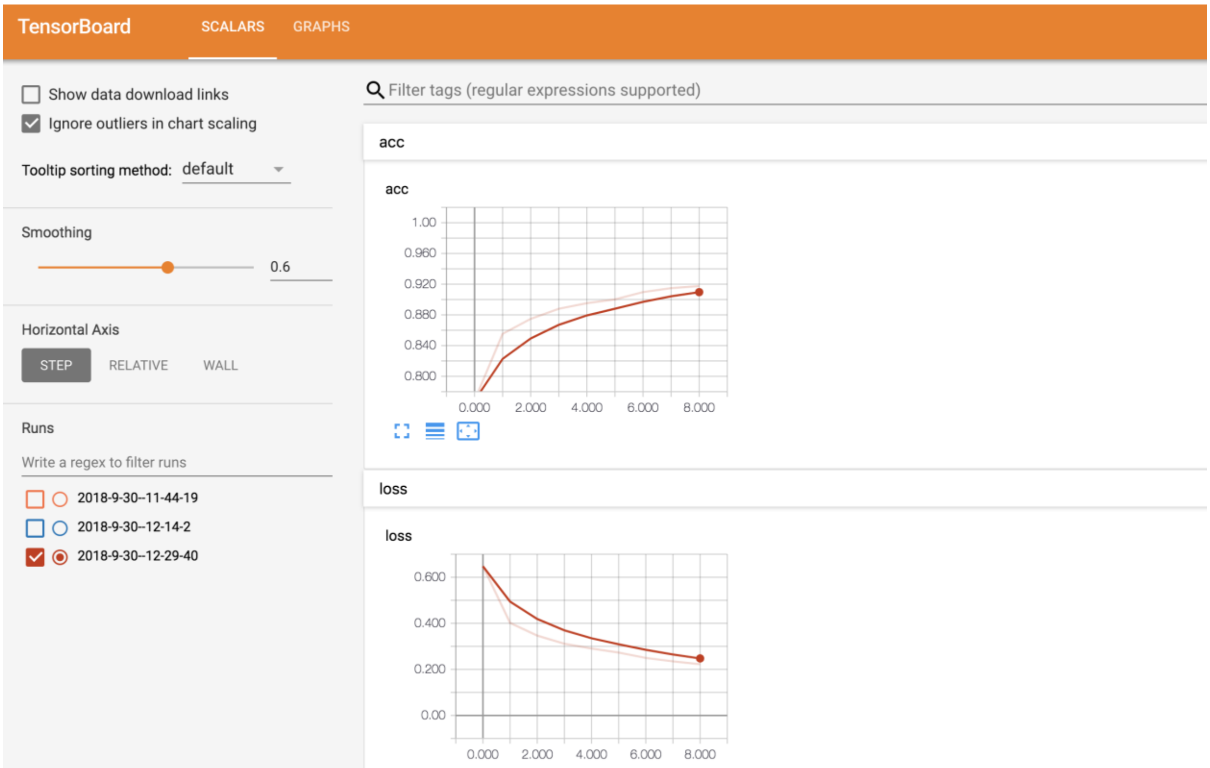


```
import time
t = time.localtime(time.time())
timeStamp = str(t.tm_year) + '-' + str(t.tm_mon) + '-' + str(t.tm_mday) + '--' + str(t.tm_hour) + '-'+str(t.tm_min) + '-
'+str(t.tm_sec)
tBoard = TensorBoard(log_dir='logs/{}'.format(timeStamp))
cnn_model.compile(
 optimizer = "adam",
 #loss = "categorical_crossentropy",
 loss = "sparse_categorical_crossentropy",
 metrics = ["accuracy"]
)
cnn_result = cnn_model.fit(train_x,target_train_y,
              batch_size=minibatch,
              epochs=epochs,
              validation_split = 0.2,
              shuffle=True,
              callbacks=[tBoard])
```

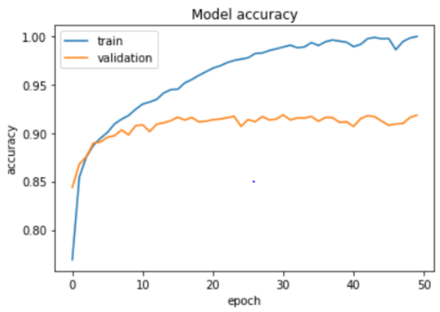
```
Train on 38400 samples, validate on 9600 samples
Epoch 1/50
38400/38400 [==============================] - 80s 2ms/step - loss: 0.6
470 - acc: 0.7694 - val_loss: 0.4468 - val_acc: 0.8439
Epoch 2/50
38400/38400 [==============================] - 80s 2ms/step - loss: 0.4
035 - acc: 0.8545 - val_loss: 0.3650 - val_acc: 0.8678
Epoch 3/50
38400/38400 [==============================] - 80s 2ms/step - loss: 0.3
484 - acc: 0.8754 - val_loss: 0.3428 - val_acc: 0.8754
Epoch 4/50
38400/38400 [==============================] - 81s 2ms/step - loss: 0.3
124 - acc: 0.8873 - val_loss: 0.3146 - val_acc: 0.8896
Epoch 5/50
38400/38400 [==============================] - 81s 2ms/step - loss: 0.2
914 - acc: 0.8946 - val_loss: 0.3043 - val_acc: 0.8907
Epoch 6/50
38400/38400 [==============================] - 80s 2ms/step - loss: 0.2
731 - acc: 0.9005 - val_loss: 0.2888 - val_acc: 0.8956
Epoch 7/50
38400/38400 [==============================] - 80s 2ms/step - loss: 0.2
511 - acc: 0.9092 - val_loss: 0.2851 - val_acc: 0.8972
Epoch 8/50
38400/38400 [==============================] - 80s 2ms/step - loss: 0.2
372 - acc: 0.9141 - val_loss: 0.2786 - val_acc: 0.9031
Epoch 9/50
38400/38400 [==============================] - 79s 2ms/step - loss: 0.2
246 - acc: 0.9180 - val_loss: 0.2844 - val_acc: 0.8981
Epoch 10/50
38400/38400 [==============================] - 80s 2ms/step - loss: 0.2
088 - acc: 0.9244 - val_loss: 0.2655 - val_acc: 0.9075
Epoch 11/50
38400/38400 [==============================] - 81s 2ms/step - loss: 0.1
949 - acc: 0.9297 - val_loss: 0.2602 - val_acc: 0.9084
Epoch 12/50
38400/38400 [==============================] - 80s 2ms/step - loss: 0.1
848 - acc: 0.9319 - val_loss: 0.2747 - val_acc: 0.9015
Epoch 13/50
38400/38400 [==============================] - 80s 2ms/step - loss: 0.1
804 - acc: 0.9347 - val_loss: 0.2629 - val_acc: 0.9091
Epoch 14/50
38400/38400 [==============================] - 79s 2ms/step - loss: 0.1
585 - acc: 0.9412 - val_loss: 0.2570 - val_acc: 0.9105
```

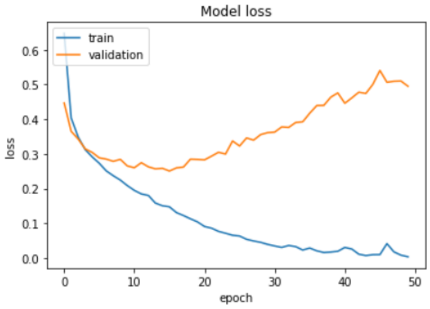
```
Epoch 15/50
38400/38400 [==============================] - 79s 2ms/step - loss: 0.1
507 - acc: 0.9446 - val_loss: 0.2586 - val_acc: 0.9126
Epoch 16/50
38400/38400 [==============================] - 79s 2ms/step - loss: 0.1
476 - acc: 0.9451 - val_loss: 0.2506 - val_acc: 0.9162
Epoch 17/50
38400/38400 [==============================] - 79s 2ms/step - loss: 0.1
312 - acc: 0.9516 - val_loss: 0.2597 - val_acc: 0.9133
Epoch 18/50
38400/38400 [==============================] - 80s 2ms/step - loss: 0.1
225 - acc: 0.9551 - val_loss: 0.2622 - val_acc: 0.9159
Epoch 19/50
38400/38400 [==============================] - 79s 2ms/step - loss: 0.1
131 - acc: 0.9594 - val_loss: 0.2849 - val_acc: 0.9115
Epoch 20/50
38400/38400 [==============================] - 79s 2ms/step - loss: 0.1
043 - acc: 0.9631 - val_loss: 0.2842 - val_acc: 0.9121
Epoch 21/50
38400/38400 [==============================] - 81s 2ms/step - loss: 0.0
910 - acc: 0.9668 - val_loss: 0.2829 - val_acc: 0.9136
Epoch 22/50
38400/38400 [==============================] - 82s 2ms/step - loss: 0.0
861 - acc: 0.9692 - val_loss: 0.2936 - val_acc: 0.9144
Epoch 23/50
38400/38400 [==============================] - 77s 2ms/step - loss: 0.0
770 - acc: 0.9724 - val_loss: 0.3047 - val_acc: 0.9156
Epoch 24/50
38400/38400 [==============================] - 47s 1ms/step - loss: 0.0
715 - acc: 0.9748 - val_loss: 0.2996 - val_acc: 0.9172
Epoch 25/50
38400/38400 [==============================] - 47s 1ms/step - loss: 0.0
657 - acc: 0.9761 - val_loss: 0.3373 - val_acc: 0.9068
Epoch 26/50
38400/38400 [==============================] - 48s 1ms/step - loss: 0.0
635 - acc: 0.9775 - val_loss: 0.3228 - val_acc: 0.9139
Epoch 27/50
38400/38400 [==============================] - 49s 1ms/step - loss: 0.0
541 - acc: 0.9818 - val_loss: 0.3460 - val_acc: 0.9116
Epoch 28/50
38400/38400 [==============================] - 48s 1ms/step - loss: 0.0
493 - acc: 0.9824 - val_loss: 0.3397 - val_acc: 0.9169
Epoch 29/50
38400/38400 [==============================] - 47s 1ms/step - loss: 0.0
454 - acc: 0.9849 - val_loss: 0.3556 - val_acc: 0.9134
Epoch 30/50
38400/38400 [==============================] - 46s 1ms/step - loss: 0.0
395 - acc: 0.9867 - val_loss: 0.3612 - val_acc: 0.9143
Epoch 31/50
38400/38400 [==============================] - 45s 1ms/step - loss: 0.0
349 - acc: 0.9886 - val_loss: 0.3628 - val_acc: 0.9186
Epoch 32/50
38400/38400 [==============================] - 48s 1ms/step - loss: 0.0
307 - acc: 0.9905 - val_loss: 0.3777 - val_acc: 0.9134
Epoch 33/50
38400/38400 [==============================] - 46s 1ms/step - loss: 0.0
363 - acc: 0.9879 - val_loss: 0.3763 - val_acc: 0.9155
Epoch 34/50
```

```
38400/38400 [==============================] - 46s 1ms/step - loss: 0.0
329 - acc: 0.9887 - val_loss: 0.3905 - val_acc: 0.9153
Epoch 35/50
38400/38400 [==============================] - 48s 1ms/step - loss: 0.0
232 - acc: 0.9932 - val_loss: 0.3925 - val_acc: 0.9170
Epoch 36/50
38400/38400 [==============================] - 47s 1ms/step - loss: 0.0
289 - acc: 0.9902 - val_loss: 0.4173 - val_acc: 0.9121
Epoch 37/50
38400/38400 [==============================] - 46s 1ms/step - loss: 0.0
210 - acc: 0.9940 - val_loss: 0.4393 - val_acc: 0.9159
Epoch 38/50
38400/38400 [==============================] - 48s 1ms/step - loss: 0.0
161 - acc: 0.9959 - val_loss: 0.4397 - val_acc: 0.9158
Epoch 39/50
38400/38400 [==============================] - 48s 1ms/step - loss: 0.0
174 - acc: 0.9947 - val_loss: 0.4634 - val_acc: 0.9109
Epoch 40/50
38400/38400 [==============================] - 47s 1ms/step - loss: 0.0
196 - acc: 0.9935 - val_loss: 0.4758 - val_acc: 0.9116
Epoch 41/50
38400/38400 [==============================] - 46s 1ms/step - loss: 0.0
307 - acc: 0.9890 - val_loss: 0.4459 - val_acc: 0.9067
Epoch 42/50
38400/38400 [==============================] - 48s 1ms/step - loss: 0.0
261 - acc: 0.9913 - val_loss: 0.4614 - val_acc: 0.9148
Epoch 43/50
38400/38400 [==============================] - 47s 1ms/step - loss: 0.0
111 - acc: 0.9973 - val_loss: 0.4778 - val_acc: 0.9178
Epoch 44/50
38400/38400 [==============================] - 52s 1ms/step - loss: 0.0
072 - acc: 0.9985 - val_loss: 0.4742 - val_acc: 0.9168
Epoch 45/50
38400/38400 [==============================] - 51s 1ms/step - loss: 0.0
097 - acc: 0.9971 - val_loss: 0.5005 - val_acc: 0.9122
Epoch 46/50
38400/38400 [==============================] - 48s 1ms/step - loss: 0.0
100 - acc: 0.9974 - val_loss: 0.5405 - val_acc: 0.9079
Epoch 47/50
38400/38400 [==============================] - 44s 1ms/step - loss: 0.0
416 - acc: 0.9859 - val_loss: 0.5065 - val_acc: 0.9093
Epoch 48/50
38400/38400 [==============================] - 45s 1ms/step - loss: 0.0
179 - acc: 0.9941 - val_loss: 0.5093 - val_acc: 0.9098
Epoch 49/50
38400/38400 [==============================] - 45s 1ms/step - loss: 0.0
084 - acc: 0.9979 - val_loss: 0.5100 - val_acc: 0.9158
Epoch 50/50
38400/38400 [==============================] - 45s 1ms/step - loss: 0.0
038 - acc: 0.9995 - val_loss: 0.4949 - val_acc: 0.9184
```




```
plt.plot(cnn_result.history['acc'])
plt.plot(cnn_result.history['val_acc'])
plt.title('Model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'validation'], loc='upper left')
plt.show()
# summarize history for loss
plt.plot(cnn_result.history['loss'])
plt.plot(cnn_result.history['val_loss'])
plt.title('Model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'validation'], loc='upper left')
plt.show()
```






```
test_result = cnn_model.evaluate(test_x,target_test_y, verbose=0)
print('Classification loss:', test_result[0])
print('Test Accuracy:', test_result[1])
```

```
Classification loss: 0.35
Test Accuracy: 0.91
```

```
images,labels = ld.load('test')
test_x = images
test_x = test_x.reshape(test_x.shape[0], img_rows, img_cols, 1)
target_test_y = labels
test_x = test_x.astype('float32')
test_x = test_x / 255
test_y = to_categorical(target_test_y)

y_pred = cnn_model.predict(test_x)
y_pred = np.argmax(np.round(y_pred),axis=1)
from sklearn.metrics import classification_report
print(classification_report(target_test_y, y_pred))
cm = confusion_matrix(target_test_y, y_pred)
print(cm)
```

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]] [9 2 1 ... 8 1 5]
             precision    recall  f1-score   support

          0       0.86      0.83      0.85      1000
          1       0.99      0.97      0.98      1000
          2       0.86      0.87      0.87      1000
          3       0.91      0.91      0.91      1000
          4       0.84      0.90      0.87      1000
          5       0.98      0.97      0.98      1000
          6       0.75      0.72      0.73      1000
          7       0.95      0.98      0.96      1000
          8       0.97      0.98      0.97      1000
          9       0.98      0.96      0.97      1000

avg / total       0.91      0.91      0.91     10000
```

```
[[829   0  18  16   6   1 119   0  11   0]
 [  3 975   1  12   4   0   3   0   2   0]
 [ 14   1 870   8  44   0  62   0   1   0]
 [ 18   6  11 909  27   0  25   0   3   1]
 [  5   1  39  18 903   0  33   1   0   0]
 [  0   0   0   0   0 974   0  17   1   8]
 [ 88   0  70  25  83   0 722   1  11   0]
 [  0   0   0   0   0  12   0 978   0  10]
 [  2   0   1   6   2   2   5   4 978   0]
 [  1   0   0   0   0   7   0  31   0 961]]
```

```
y_classes = ["T-shirt/top","Trouser","Pullover","Dress","Coat","Sandal","Shirt","Sneaker","Bag","Ankle boot"]
import seaborn as sns
sns.heatmap(cm, annot=True, fmt='d',xticklabels=y_classes, yticklabels=y_classes)
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
```