from keras.utils import to\_categorical

import numpy as np

import pandas as pd

from sklearn.model\_selection import train\_test\_split

data\_train = pd.read\_csv('C:/Users/achyu/OneDrive/Desktop/MNIST\_FASHION/fashion-mnist\_train.csv')

data\_test = pd.read\_csv('C:/Users/achyu/OneDrive/Desktop/MNIST\_FASHION/fashion-mnist\_test.csv')

img\_rows, img\_cols = 28, 28

input\_shape = (img\_rows, img\_cols, 1)

X = np.array(data\_train.iloc[:, 1:])

y = to\_categorical(np.array(data\_train.iloc[:, 0]))

#Here we split validation data to optimiza classifier during training

X\_train, X\_val, y\_train, y\_val = train\_test\_split(X, y, test\_size=0.2, random\_state=13)

#Test data

X\_test = np.array(data\_test.iloc[:, 1:])

y\_test = to\_categorical(np.array(data\_test.iloc[:, 0]))

X\_train = X\_train.reshape(X\_train.shape[0], img\_rows, img\_cols, 1)

X\_test = X\_test.reshape(X\_test.shape[0], img\_rows, img\_cols, 1)

X\_val = X\_val.reshape(X\_val.shape[0], img\_rows, img\_cols, 1)

X\_train = X\_train.astype('float32')

X\_test = X\_test.astype('float32')

X\_val = X\_val.astype('float32')

X\_train /= 255

X\_test /= 255

X\_val /= 255

import keras

from keras.models import Sequential

from keras.layers import Dense, Dropout, Flatten

from keras.layers import Conv2D, MaxPooling2D

from keras.layers.normalization import BatchNormalization

batch\_size = 256

num\_classes = 10

epochs = 50

#input image dimensions

img\_rows, img\_cols = 28, 28

model = Sequential()

model.add(Conv2D(32, kernel\_size=(3, 3),

activation='relu',

kernel\_initializer='he\_normal',

input\_shape=input\_shape))

model.add(MaxPooling2D((2, 2)))

model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), activation='relu'))

model.add(MaxPooling2D(pool\_size=(2, 2)))

model.add(Dropout(0.25))

model.add(Conv2D(128, (3, 3), activation='relu'))

model.add(Dropout(0.4))

model.add(Flatten())

model.add(Dense(128, activation='relu'))

model.add(Dropout(0.3))

model.add(Dense(num\_classes, activation='softmax'))

model.compile(loss=keras.losses.categorical\_crossentropy,

optimizer=keras.optimizers.Adam(),

metrics=['accuracy'])

model.summary()

history = model.fit(X\_train, y\_train,

batch\_size=batch\_size,

epochs=epochs,

verbose=1,

validation\_data=(X\_val, y\_val))

score = model.evaluate(X\_test, y\_test, verbose=0)

print('Test loss:', score[0])

print('Test accuracy:', score[1])

import matplotlib.pyplot as plt

accuracy = history.history['acc']

val\_accuracy = history.history['val\_acc']

loss = history.history['loss']

val\_loss = history.history['val\_loss']

epochs = range(len(accuracy))

plt.plot(epochs, accuracy, 'bo', label='Training accuracy')

plt.plot(epochs, val\_accuracy, 'b', label='Validation accuracy')

plt.title('Training and validation accuracy')

plt.legend()

plt.figure()

plt.plot(epochs, loss, 'bo', label='Training loss')

plt.plot(epochs, val\_loss, 'b', label='Validation loss')

plt.title('Training and validation loss')

plt.legend()

plt.show()

#get the predictions for the test data

predicted\_classes = model.predict\_classes(X\_test)

#get the indices to be plotted

y\_true = data\_test.iloc[:, 0]

correct = np.nonzero(predicted\_classes==y\_true)[0]

incorrect = np.nonzero(predicted\_classes!=y\_true)[0]

for i, correct in enumerate(correct[:9]):

plt.subplot(3,3,i+1)

plt.imshow(X\_test[correct].reshape(28,28), cmap='gray', interpolation='none')

plt.title("Predicted {}, Class {}".format(predicted\_classes[correct], y\_true[correct]))

plt.tight\_layout()