import keras

from keras.datasets import mnist

from keras.models import Sequential

from keras.layers import Dense, Dropout, Flatten

from keras.layers import Conv2D, MaxPooling2D

from keras import backend as K

batch\_size = 128

num\_classes = 10

epochs = 12

# input image dimensions

img\_rows, img\_cols = 28, 28

# the data, split between train and test sets

(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()

if K.image\_data\_format() == 'channels\_first':

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

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

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

else:

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)

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

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

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

x\_train /= 255

x\_test /= 255

print('x\_train shape:', x\_train.shape)

print(x\_train.shape[0], 'train samples')

print(x\_test.shape[0], 'test samples')

# convert class vectors to binary class matrices

y\_train = keras.utils.to\_categorical(y\_train, num\_classes)

y\_test = keras.utils.to\_categorical(y\_test, num\_classes)

model = Sequential()

model.add(Conv2D(6, kernel\_size=(3, 3),

activation='relu',

input\_shape=input\_shape))

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

model.add(Conv2D(16, (3, 3), activation='relu'))

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

model.add(Flatten())

model.add(Dense(120, activation='relu'))

model.add(Dense(84, activation='relu'))

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

# <https://keras.io/optimizers/>

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

optimizer=keras.optimizers.Adadelta(lr=0.1, rho=0.95, epsilon=None, decay=0.0),

metrics=['accuracy'])

model.fit(x\_train, y\_train,

batch\_size=batch\_size,

epochs=epochs,

verbose=1,

validation\_data=(x\_test, y\_test))

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

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

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