from\_future\_import print\_function

import numpy as np

from keras.datasets import mnist

from keras.models import Sequential

from keras.layers.core import Dense, Activation

from keras.optimizers import SGD

from keras.utils import np\_utils

np.random.seed(1671) # for reproducibility

#network and training

NB\_EPOCH = 20

BATCH\_SIZE = 128

VERBOSE =1

NB\_CLASSES = 10 # number of outputs = number of digits

OPTIMIZER = SGD() # optimizer

N\_HIDDEN = 128

VALIDATION\_SPLIT = 0.2 # how much TRAIN is reserved for Validation

# data shuffled and split between train and test sets

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

#X\_train is 6000 rows of 28\*28 values -🡪 reshaped in 60000 \* 784

RESHAPED = 784

#

X\_train = X\_train .reshape(60000, RESHAPED)

X\_test = X\_test.reshape(10000, RESHAPED)

X\_train = X\_train.astype(‘float 32’)

X\_test = X\_test.astype(‘float 32’)

#normalized

X\_train /= 255

X\_test /= 255

print(X\_train.shape[0], ‘train samples’)

print(X\_test.shape[0], ‘test samples’)

# convert class vectors to binary class matrices

Y\_train = np\_utils.to\_categorical(y\_train, NB\_CLASSES)

Y\_test = np\_utils.to\_categorical(y\_test, NB\_CLASSES)

#M\_HIDDEN hidden layers

#10 outputs  
#final stage is softmax

model = Sequential()

model.add(Dense(N\_HIDDEN, input\_shape = (RESHAPED,)))

model.add(Activation(‘relu’))

model.add(Dense(N\_HIDDEN))

model.add(Activation(‘relu’))

model.add(Dense(NB\_CLASSES))

model.add(Activation(‘softmax’))

model.summary()

model.compile(loss = ‘categorical\_crossentropy’, optimizer = OPTIMIZER, metrics = [‘accuracy’])

history = model.fit(X\_train, Y\_train, batch\_size = BATCH\_SIZE, epochs = NB\_EPOCH,

verbose = VERBOSE, validation\_split = VALIDATION\_SPLIT)

score = model.evaluate(X\_test, Y\_test, verbose = VERBOSE)

print(“Test score: “, score[0])

print(“Test accuracy: “, score[1

])