

```
In [1]: import numpy as np
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.utils import np_utils
from keras import backend as K
```

Using TensorFlow backend.

```
In [2]: # Set that the color channel value will be first
K.set_image_data_format("channels_first")

# Set seed
np.random.seed(0)

# Set image information
channels = 1
height = 28
width = 28

# Load data and target from MNIST data
(data_train, target_train), (data_test, target_test) = mnist.load_data()

# Reshape training image data into features
data_train = data_train.reshape(data_train.shape[0], channels, height, width)

# Reshape test image data into features
data_test = data_test.reshape(data_test.shape[0], channels, height, width)

# Rescale pixel intensity to between 0 and 1
features_train = data_train / 255
features_test = data_test / 255

# One-hot encode target
target_train = np_utils.to_categorical(target_train)
target_test = np_utils.to_categorical(target_test)
number_of_classes = target_test.shape[1]

# Start neural network
network = Sequential()

# Add convolutional layer with 64 filters, a 5x5 window, and ReLU activation function
network.add(Conv2D(filters=64,
                    kernel_size=(5, 5),
                    input_shape=(channels, width, height),
                    activation='relu'))

# Add max pooling layer with a 2x2 window
network.add(MaxPooling2D(pool_size=(2, 2)))

# Add dropout layer
network.add(Dropout(0.5))

# Add layer to flatten input
network.add(Flatten())

# Add fully connected layer of 128 units with a ReLU activation function
network.add(Dense(128, activation="relu"))

# Add dropout layer
network.add(Dropout(0.5))
```

```

# Add fully connected layer with a softmax activation function
network.add(Dense(number_of_classes, activation="softmax"))

# Compile neural network
network.compile(loss="categorical_crossentropy", # Cross-entropy
                optimizer="rmsprop", # Root Mean Square Propagation
                metrics=["accuracy"]) # Accuracy performance metric

# Train neural network
network.fit(features_train, # Features
            target_train, # Target
            epochs=2, # Number of epochs
            verbose=0, # Don't print description after each epoch
            batch_size=1000, # Number of observations per batch
            validation_data=(features_test, target_test)) # Data for eval
uation

```

Downloading data from <https://s3.amazonaws.com/img-datasets/mnist.npz>  
11493376/11490434 [=====] - 2s 0us/step

WARNING:tensorflow:From c:\users\safar\documents\github\safariel103\bellevue university\courses\dsc550\week9and10\env\lib\site-packages\tensorflow\_core\python\ops\resource\_variable\_ops.py:1630: calling BaseResourceVariable.\_\_init\_\_ (from tensorflow.python.ops.resource\_variable\_ops) with constraint is deprecated and will be removed in a future version.

Instructions for updating:

If using Keras pass \*\_constraint arguments to layers.

WARNING:tensorflow:From c:\users\safar\documents\github\safariel103\bellevue university\courses\dsc550\week9and10\env\lib\site-packages\keras\backend\tensorflow\_backend.py:4070: The name tf.nn.max\_pool is deprecated. Please use tf.nn.max\_pool2d instead.

WARNING:tensorflow:From c:\users\safar\documents\github\safariel103\bellevue university\courses\dsc550\week9and10\env\lib\site-packages\keras\backend\tensorflow\_backend.py:422: The name tf.global\_variables is deprecated. Please use tf.compat.v1.global\_variables instead.

Out[2]: <keras.callbacks.callbacks.History at 0x1d8ad6eba08>

In [ ]: