Software project 1 (Part 4)

Buciu Ștefan

User Manual

Installation of the needed Python libraries:

pip install numpy

pip install pandas

pip install plotly

pip install sklearn

pip install tensorflow

Dataset Exploration

We use the MNIST Handwritten Digits Dataset which can be found in the keras package.

from tensorflow.keras.datasets import mnist

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

The training input data consists of 60000 grayscale images of dimensions 28x28. The training output data is an array of 60000 numbers ranging from 0 to 9 representing the correct digit. The test data contains 10000 samples.

We need to normalize the input data, and that can be done simply by dividing by 255 (the maximum value of a pixel), or we can use some tools like keras.utils.normalize.

Building the model

We are going to use the Sequential model from keras. After initializing such an object, we can add multiple types of layers to build a CNN. We are going to use the following layers: Conv2D, MaxPool2D, Flatten, Dense.

model = keras.models.Sequential()

model.add(keras.layers.Conv2D(32, kernel\_size=(3, 3), activation=tf.nn.relu, kernel\_initializer=keras.initializers.he\_uniform, input\_shape=(28, 28, 1)))

model.add(keras.layers.MaxPool2D(input\_shape=(2, 2)))

model.add(keras.layers.Flatten())

model.add(keras.layers.Dense(100, activation=tf.nn.relu))

model.add(keras.layers.Dense(10, activation=tf.nn.softmax))

After defining our model, we need to compile it. When doing so, we need to specify some parameters, such as the optimizer that we are going to use, the loss function we are going to minimize and what metrics to consider when building the error.

model.compile(

optimizer='adam',

loss='sparse\_categorical\_crossentropy',

metrics=['accuracy']

)

Starting the training

We can train our model by using the fit function where we need to pass an array of samples as input and an array of results as output.

history = model.fit(trainX, trainY, epochs=EPOCHS, batch\_size=32, validation\_data=(testX, testY), verbose=0)