Handwritten Digit Recognition

Author: Michał Maźniewski

E-mail: michmaz061@student.polsl.pl

Tutor: Marcin Wierzchanowski MSc

Table of Contents

Introduction

The project is to take a picture of a digit and process it up to recognize the image of that digit like a human brain recognize the various digits. The project contains the idea of the Image Processing techniques and the research area of machine learning and the main part of the machine learning called Neural Network

Training a model comes with the idea of to train a model by giving various sets of similar characters and to say them that the output of this digit is “that digit”. Like this idea one has to train the newly built neural network with so many characters.

Task Analysis

Convolutional neural networks are a specialized type of artificial neural networks that use a mathematical operation called convolution in place of general matrix multiplication in at least one of their layers. They are specifically designed to process pixel data and are used in image recognition and processing.

Using MNIST dataset of 60,000 small square 28×28 pixel grayscale images of handwritten single digits between 0 and 9. Building the model that will train and learn how to classify these images. While adding images to our model we need to make them from 28x28 dimensional to 1x784 dimensional so the input layer of the neural network can deal with it. Input and hidden layer consists of 128 neurons, output layer has 10 neurons.

The MNIST database is a large database of handwritten digits. It is also widely used for training and testing in the field of machine learning. It was created by the samples from NIST's original datasets. The creators felt that since NIST's training dataset was taken from employees, while the testing dataset was taken from high school students, it was not well-suited for machine learning experiments. Furthermore, the black and white images from NIST were normalized to fit into a 28x28 pixel bounding box and anti-aliased.

Selected tools:

Opencv (Open Source Computer Vision Library) is a python library which is very updated in the work of Image processing. One image file and pixel values can easily come into surface by this library. This library provides a common infrastructure and module related to computer vision technologies.

Numpy is a library adding support for large multidimensional array and metrices. This package contains a large collection of high level mathematical functions to operate on those arrays.

TensorFlow is an end-to-end open source package in python for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets developers easily build and deploy ML powered applications.

Software Specification

For my model I have used 4 layers. First one is for flattening the 28x28 image to 784x1 matrix, which is used as input. Then we have 2 simple dense layers. Last which is for output makes sure that all 10 neurons will add to one, where softmax gives us the probability for each digit to be the right answer. The model was trained using 50 epochs.

```
model = tf.keras.models.Sequential()
model.add(tf.keras.layers.Flatten(input_shape=(28, 28)))
model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu))
model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu))
model.add(tf.keras.layers.Dense(10, activation=tf.nn.softmax))

model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

For checking how the model works we have a loop that takes every .jpg with digit in directory and shows us the images and in console predicted number respectively to image file.

```
while os.path.isfile(f"dig/digit_{image_number}.jpg"):
    try:
        image = cv.imread(f"dig/digit_{image_number}.jpg")[:, :, 0]
        image = np.invert(np.array([image]))
        prediction = model.predict(image)
        print("{digit_", image_number, ")", f"Prediction of number is: {np.argmax(prediction)}")
        plt.imshow(image[0], cmap=plt.cm.binary)
        plt.show()
    except:
        print("Error")
    finally:
        image_number += 1
```

Experiments

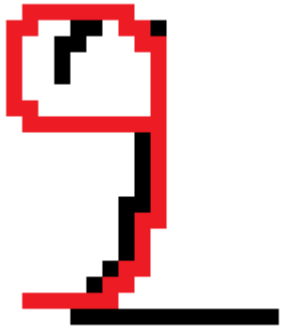
The things that I was experimenting with were number of epochs and images of handwritten digits.

I have tested a lot of different cases of epoch's value. That was done to get the best accuracy on my model and for me 50 epochs presented the best result, after getting over 50 I could observe the regression in accuracy of model.

Using images with handwritten digits I am able to say how my model is doing.
Images that I was using were drawn by my colleagues, so depending on the style of digit the results were different for digits from 1 to 9.

```
313/313 [==============================] - 0s 946us/step - loss: 0.2172 - accuracy: 0.9784
{digit_ 1 ) Prediction of number is: 1
{digit_ 2 ) Prediction of number is: 2
{digit_ 3 ) Prediction of number is: 3
{digit_ 4 ) Prediction of number is: 1
{digit_ 5 ) Prediction of number is: 5
{digit_ 6 ) Prediction of number is: 5
{digit_ 7 ) Prediction of number is: 8
{digit_ 8 ) Prediction of number is: 8
{digit_ 9 ) Prediction of number is: 2
```

Here we can see that program predicted 9 as 2 this is because the most part of 2 are in line with the arc of 9.



Based on results we can definetly say that my model is not perfect and that there a field to show off with getting it better.

Conclusion

For me, this project was a great introduction to the world of ML and a presentation of the basics of working with AI
Most of my time was spent researching CCN information.
Project can be improved by improving myself and neural network.
I doubt that my model can be applicable to any real-life situations, but it certainly is a basis for introduction into AI.

Links

GitHub:

https://github.com/michmaz061/BIAI_project