Joanna Masikowska

B9TB1710

CAPS12_B9TB1710.m

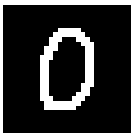
```
#loading data
fid=fopen('t10k-images-idx3-ubyte','r','b');
fread(fid,4,'int32');
data=fread(fid,[28*28,10000],'uint8');
fclose(fid);

#loading my_numbers
sample1=imread('my_number.png');
sample2=imread('my_number2.png');
sample1=mean(sample1,3);
sample2=mean(sample2,3);

#labels
fid=fopen('t10k-labels-idx1-ubyte','r','b');
fread(fid,2,'int32');
label=fread(fid,10000,'uint8');
fclose(fid);

#training
tr_label = label(1:5000);
tr_data = data(:,1:5000);
model = train(tr_label,sparse(tr_data)');

#test1
te_label = label(5001:6000);
te_data = data(:,5001:6000);
pred_label=predict(te_label,sparse(te_data)',model);

pred1=predict([0],sparse(reshape(sample1',1,28*28)),model)
pred2=predict([5],sparse(reshape(sample2',1,28*28)),model)
```

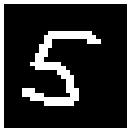
I use functions **fopen** and **fread** to open and read the files with data that I will be using. In **fopen**, I specify file name, the purpose of opening (‘r’ for read), and the format of data (‘b’ for IEEE big endian format). **Fopen** returns an integer (fid) that is used to refer to the file. If it returns -1, it means that the error occurred. To make sure that it is not a case, I tried the code in the command window as below.

```
>> fid=fopen('t10k-images-idx3-ubyte','r','b')
fid =  8
>> fread(fid,4,'int32')
ans =

    2051
   10000
      28
      28

>> data=fread(fid,[28*28,10000],'uint8');
>> fclose(fid);
>> fid=fopen('t10k-labels-idx1-ubyte','r','b')
fid =  8
>> fread(fid,2,'int32')
ans =

    2049
   10000

>> label=fread(fid,10000,'uint8');
>> fclose(fid);
```

In **fread** I specify 3 arguments. Fid refers to the file opened by **fopen**. 2nd argument is the size of an output which in the first case is a matrix of a size 784x10000, and in the second case it is a column vector 1x10000. 3rd argument stands for the type of data to be read. 'uint8' is an 8-bit unsigned integer.

After each extraction of the data I need (to variables *data* and *label*), I close each file with **fclose**. I also load the images I will be using in the code. 'my_number' is 0 and 'my_number2' is 5.





Before running the code, I add a path to liblinear as it is in a different directory that my script.

```
>> addpath('C:\Users\Asia\Desktop\zajecia_online\Exercises in Computer-Aided
 Problem Solving\liblinear-2.30\matlab')
>>
```

I train the algorithm to recognise handwritten digits (from 28x28 pixels pictures). I convert the data from *data* and *labels* into its sparse matrix representation which does not include zeros (in order to use less memory and make the algorithm faster) and use it in the function **train** from liblinear. Then, with a function **predict** (from liblinear), I make my algorithm try to recognize the digits. First, I run it on the data from mnist. In the output I can see that the accuracy of the algorithm on that data was 86%. Next, I run it on my handwritten digits. It accurately found that first image represents 0, but failed to recognize 5.

```
Objective value = -0.131785
nSV = 1336
Accuracy = 86.3% (863/1000)
Accuracy = 100% (1/1)
pred1 = 0
Accuracy = 0% (0/1)
pred2 =  8
>>
```