Document Analysis - Report

Exercise 4

Aebischer Nadia (nadia.aebischer@unifr.ch)
Luyet Gil (gil.luyet@unifr.ch)

Introduction

The goal of this exercise session was to discover how neural networks work by changing some of the available parameters. The dataset of MNIST (which is composed of digit images of 28x28 pixels) was used to train and test the neural network. This exercise is splitted in two part parts. In the first part, only pixel values intensity are used as feature. In the second part, we are going to use custom features (the same used with exercise session 3) and see if we can have better results.

Features

Before extracting any features, all the samples (train and test sets) have to be binarized. As the different samples are of good quality (i.e. clear background and foreground), the binarization is done as follows: pixels having value greater than 127 are set to 1.0 (meaning it's white), otherwise they are set to 0.0 (meaning it's black).
Once this binarization has been done, features extraction can begin.

Basic

In Task 1, it has been asked to used as feature only the **pixel values intensity** of the samples. For each sample, pixels values are extracted line by line (from top to bottom, and from left to right) and appended to each other. This results in 784 (= 28 * 28) feature values for each sample.
Features for train set and test set are stored in `train_pv.txt` and `test_pv.txt` respectively.

Custom

In Task 2, it has been asked to used other features. For this implementation we use the **method suggested by Rath et al. [1].** This method consists of a sliding window of 1 pixel on which we compute four features. Those features are the **projection profile** (number of black pixels in one window divided by the height of the window), the **word profiles** (also known as the upper and lower profiles) which consist on taking the number of white pixels until the first appearance of the first black pixel (from top and from bottom) and the **number of black/white transitions**. For the word profiles, in some case there can be some "holes" in the sample (i.e. at the beginning and the end). To deal with this problem, we set the number of white pixel counted for the current window to half of the image height (in case no black pixel was encountered).

For each sample, the four features are computed and appended to each other. This results in 112 (= 28 * 4) feature values for each sample.
Features for train set and test set are stored in `train.txt` and `test.txt` respectively.

Tests and results

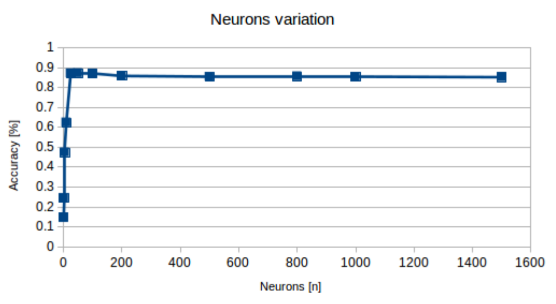
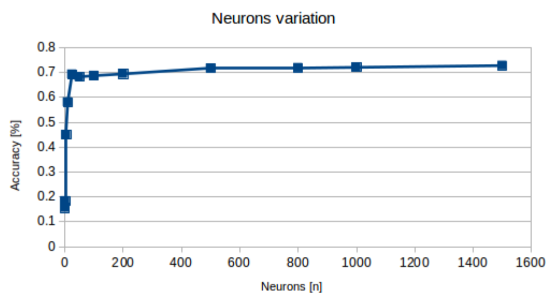
To test this neural network, we'll observe how one parameter will affect its results (i.e. the accuracy).
Default values of the parameters:

- neurons = 500
- epochs = 10
- learning rate = 0.001
- samples = 60'000

For each test, two runs will be done and the two accuracy values will be averaged.

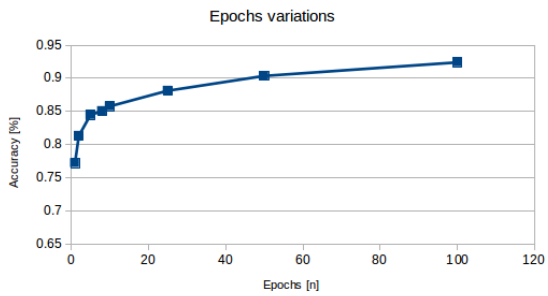
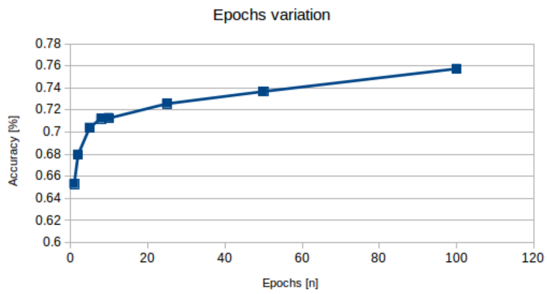
Neurons

Here we want to see how the number of neurons will impact the results.
We will only vary the neurons parameter with the following values: 1, 2, 5, 10, 25, 50, 100, 200, 500, 800,1000 and 1500.

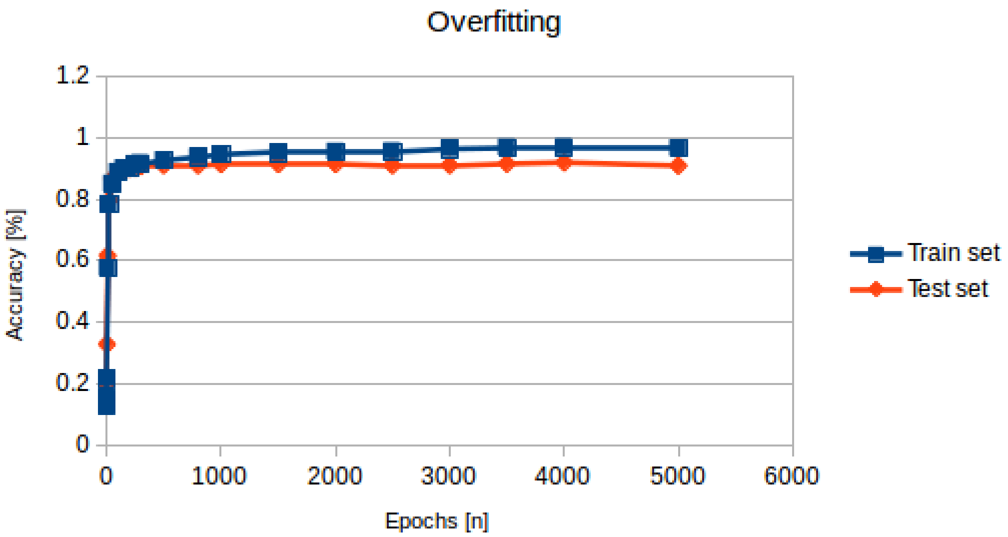
Pixels values	Custom features
	

Epochs

Here we want to see how the number of epochs will impact the results.
We will only vary the epochs parameter with the following values: 1, 2, 5, 8, 10, 25, 50, 100.

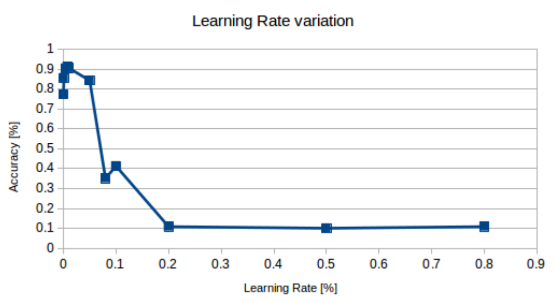
<table>
<tr><th>Pixels values</th><th>Custom features</th></tr>
<tr><td>
</td><td>
</td></tr>
</table>

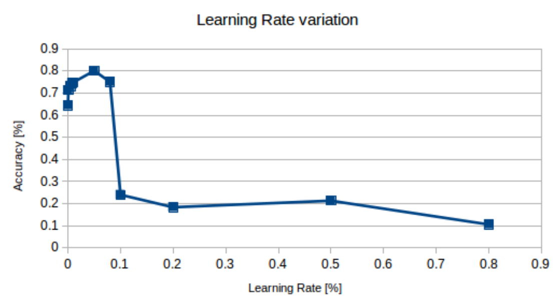
It is also possible to spot an overfitting by looking at the epochs variation. We can compare the accuracy on the training set and on the test set and see if the neural network tends to become less performant for the test set. This behavior occurs when the neural network begins to fit more to the train. The following graphic is produced by training the neural network of 25 neurons over 10'000 samples with the learning rate set to 0.001. The following graph use the pixels values to perform classification.



Learning rate

Here we want to see how the learning rate will impact the results.
We will only vary the learning rate parameter with the following values: 0.0001, 0.001, 0.005, 0.008, 0.01, ,0.05, 0.08, 0.1, 0.5 and 0.8

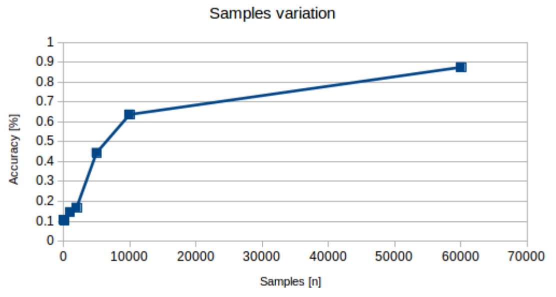
<table>
<tr><th>Pixels values</th><th>Custom features</th></tr>
<tr><td>


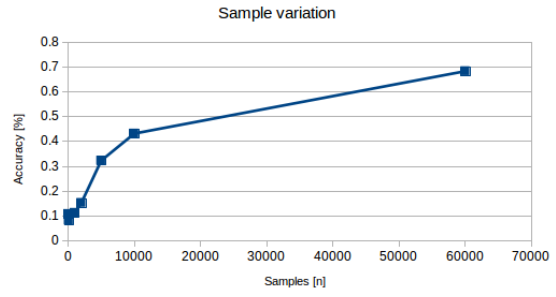
</td><td>


</td></tr>
</table>

Samples

Here we want to see how the number of samples will impact the results.
To do this, different training sets will be prepared by picking randomly a certain number (specified by the parameter) of samples from the already computed features file (`train.txt` and `train_pv.txt`). In order to be to have samples for each digit, we picked randomly parameter/10 samples for each digit.
We will only vary the samples parameter with the following values: 20, 100, 1000, 2000, 5000, 10000, 60000.
Note: we used only 25 neurons to run the tests here.

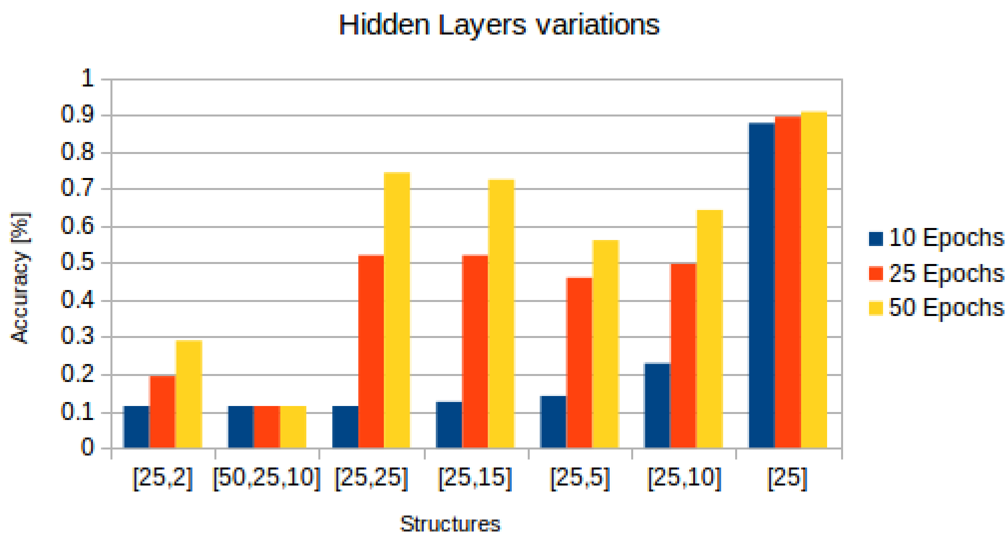
<table>
<tr><th>Pixels values</th><th>Custom features</th></tr>
<tr><td>


</td><td>


</td></tr>
</table>

Hidden layer variations

Here we want to see how the number of hidden layers will impact the results. We are going to consider only the pixels values (because more accurate). As the number of layers and the number of neurons within these layers can vary. The test are done using 0.001 as learning rate and performed over the whole test set. The epochs are here increasing (it seems to us to be the most important parameter).



Conclusion

The pixels values are as expected the most accurate features to use and need a lot of computations regarding for example our custom features. We can also detect overfitting in the learning rate variation graphs. The more the neural network learns from the test set the less it is accurate.
In contrary, the more samples we use the more accurate the NN will be. This behavior was expected and make total sense. It is the same for the epochs. The more epochs are used to train the NN the more the structure will be able to perform recognition with good accuracy.
The number of total needed hidden neuron in the case we only care about one hidden layer is quite low regarding to the size of the input. We only need 25 neurons in average at the hidden layer to perform a very good recognition. This results is quite astonishing and was not expected at all.
For the runs with more than one hidden layer we can see that the NN needs more epochs to learn a structure but we can expect a better recognition after the learning process is over. The variation of accuracy when the number of epochs grown is much more significant than the variation observed with only one hidden layer. This is why we can expect a lot from those structures.
But it is clear that some structure [25 -> 5], for example, was not a good choice. This is due to the reduction performed by the layer containing only 2 neurons. This was also an expected behaviour we thought about.

References

[1] Rath, Toni M., and Raghavan Manmatha. "Word image matching using dynamic time warping." In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, vol. 2, pp. II-521. IEEE, 2003.