# Part 2

We are a team of:

* Hussein Taha -1937542 Group: 03
* Olivier Samson- Group: 01

Python code is attached as “TP2\_2.py”

## Comparing of Full connected NN and Convolution NN with MNIST data

Our code contains Two different architecture of Neural Network:

1. Fully Connected two layered Neural Network (28\*28 input🡪512,

512🡪10 output)

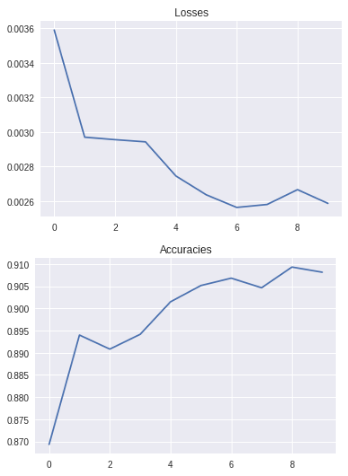
1. Convolution Neural Network architecture using two convolution layers (two convolution layers with 5 kernel and one full connected layer) [1🡪16, 16🡪32🡪10 output]

Tabel.1 shows the average loss and accuracy for a Fully Connected NN and a Convolution NN with the same validation data and with a number of epochs=10 and Learning rate of lr=0.001

*Table .1: Average losses and accuracy of Full Connected and Convolution NN*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epochs | Full connected NN | | Convolution NN | | |
| Average Losses | Accuracy % | | Average Losses | Accuracy % |
| 1 | 0.0040 | 85 | | 0.0032 | 88 |
| 2 | 0.0036 | 87 | | 0.0031 | 89 |
| 3 | 0.0034 | 87 | | 0.0028 | 90 |
| 4 | 0.0033 | 88 | | 0.0025 | 91 |
| 5 | 0.0032 | 88 | | 0.0026 | 91 |
| 6 | 0.0031 | 89 | | 0.0026 | 91 |
| 7 | 0.0031 | 89 | | 0.0029 | 90 |
| 8 | 0.0032 | 88 | | 0.0026 | 91 |
| 9 | 0.0031 | 89 | | 0.0028 | 90 |
| 10 | 0.0031 | 89 | | 0.0028 | 90 |

So, it is concluded that, for image processing, the Convolution Neural Network is more accurate and has lower losses.



1. (b)

*Figure.1: Average losses and accuracy for validation data with 10 training Epochs and Learning rate of lr=0.001 for a) full-connected NN, b) Convolution NN*

Testing data with convolution network returns as: Average loss: 0.0029, Accuracy: 8999/10000 (89%).

This is in support of the theory that implies that a preprocessing of the inputs by convolution layers in a smaller number of more significant features, help with the performances of the NN.

## Experiments for convolution network (CNN) to check accuracy and losses with different architecture

In this part we use different CNN architectures and obtain the losses and accuracy for the validation data set and test the best model on our testing data set.

*Table.2 losses and accuracy of different CNN architecture*

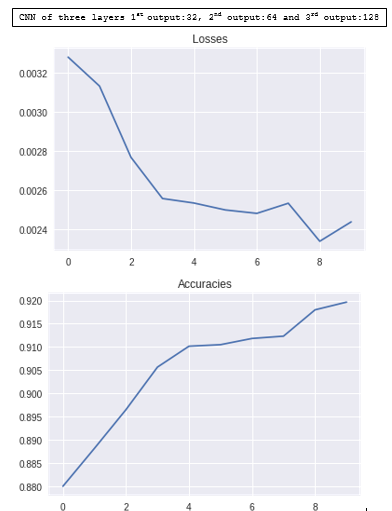
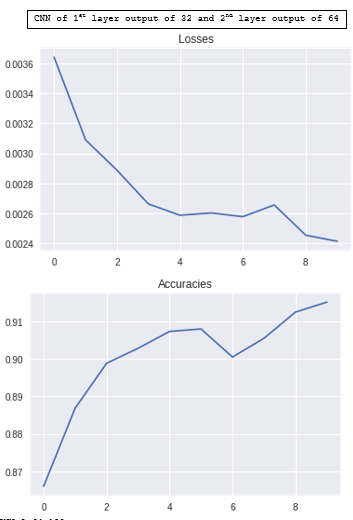
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Epoch | Two Convolution layers | | | | | | Three Convolution layers | | | |
| CNN\_16\_32 | | CNN\_32\_64 | | CNN\_64\_128 | | CNN\_16\_32\_64 | | CNN\_32\_64\_128 | |
| Losses | Accu% | Losses | Accu% | Losses | Accu% | Losses | Accu% | Losses | Accu% |
| 1 | 0.0032 | 88 | 0.0036 | 87 | 0.0035 | 87 | 0.0034 | 88 | 0.0033 | 88 |
| 2 | 0.0031 | 89 | 0.0031 | 89 | 0.0032 | 88 | 0.0032 | 88 | 0.0031 | 89 |
| 3 | 0.0028 | 90 | 0.0029 | 90 | 0.0029 | 89 | 0.0028 | 90 | 0.0028 | 90 |
| 4 | 0.0025 | 91 | 0.0027 | 90 | 0.0027 | 90 | 0.0028 | 90 | 0.0026 | 91 |
| 5 | 0.0026 | 91 | 0.0026 | 91 | 0.0027 | 90 | 0.0028 | 90 | 0.0025 | 91 |
| 6 | 0.0026 | 91 | 0.0026 | 91 | 0.0027 | 90 | 0.0026 | 90 | 0.0025 | 91 |
| 7 | 0.0029 | 90 | 0.0026 | 90 | 0.0026 | 91 | 0.0025 | 91 | 0.0025 | 91 |
| 8 | 0.0026 | 91 | 0.0027 | 91 | 0.0026 | 91 | 0.0026 | 91 | 0.0025 | 91 |
| 9 | 0.0028 | 90 | 0.0025 | 91 | 0.0025 | 91 | 0.0025 | 91 | 0.0023 | 92 |
| 10 | 0.0028 | 90 | 0.0024 | 92 | 0.0026 | 91 | 0.0024 | 91 | 0.0024 | 92 |

The architecture of the CNN with three layers (1st output:32, 2nd output:64 and 3rd output:128) is more stable in training phase and is considered as the best one, where losses keep decreasing and accuracy increases with epochs of training. However, the CNN model with two layers (1st output:32, 2nd output:64) shows good performances and nearly as much as our best model although the learning process is a bit slower.

Figure 2 presents validation losses and accuracy according to training epochs for (a) our best CNN architecture and (b) two layered CNN (1st output:32, 2nd output:64).

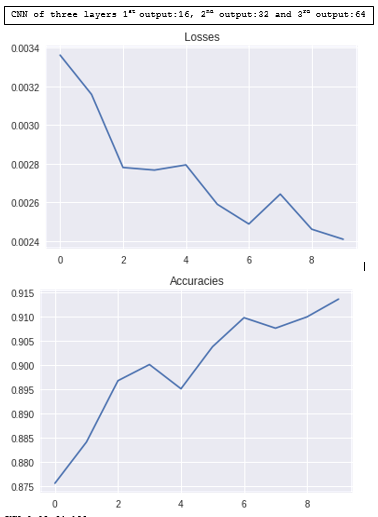
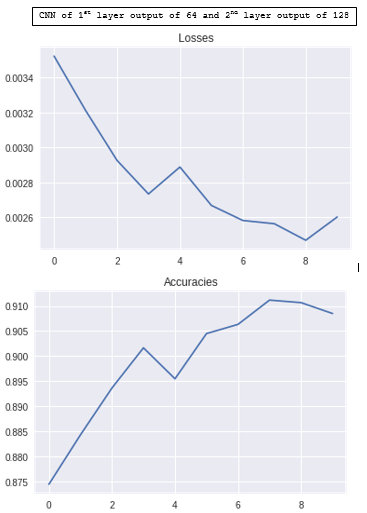
Figure 3 depicts validation losses and accuracy for (a) three layered CNN (1st output:16, 2nd output:32 and 3rd output:64), (b) two layered CNN (1st output:64, 2nd output:128) according to number of epochs. and it is noticeable that they are showing nearly the same performances, but it is faster to achieve with the three layered CNN than two layered one.

Best model test results gotten were: Average loss: 0.0025, Accuracy: (91%) with data from FashionMNIST.



1. (b)

*Figure. 2: losses and accuracy according to training epochs for (a) Best CNN architecture, (b)Two CNN layers with (1st output:32, 2nd output:64).*



1. (b)

*Figure. 3: losses and accuracy according to training epochs for (a) Three CNN layers with (1st output:16, 2nd output:32 and 3rd output:64) , (b)Two CNN layers with (1st output:64, 2nd output:128).*

A CNN has one or more layers of convolution filters. They apply a convolution with a specific stride on the input. Each filter/channel in a layer is learning to detect a specific feature and is known as feature map h(l). This is a clear advantage to only using a Fully Connected NN as it reduces the number of features needed to be processed by our FC layer. Also, the resulting feature map in output (or input of our FC network) depends on the number of filters/channels present in a layer as well as on the number of convolution layers used.



Where,

n is the number of convolution filter,

 is 2-D convolution operation,

(l) are hidden layers,

is channel input and

b is the biases

The weights of each filter are different as they are training to detect certain features of the input image like edges, lines, and other geometry. The more filters a convolution layer has, the more features it will be able to detect. So, for a CNN with two layers (1st output:64, 2nd output:128), we get better performance than for a CNN with two layers (1st output:32, 2nd output:64), which has less filters. Also, the same can be seen for a three convolution layers as our three-layer CNN (1st output:32, 2nd output:64 and 3rd output:128) has better training performances than the three-layer CNN (1st output:16, 2nd output:32 and 3rd output:64).

A CNN also has a pooling unit for each convolution layer to reduce the dimension of each feature map which reduces the computational process time. It applies the Maximum, Minimum or Mean functions over the contents of the window by dividing the window of data to strides of [x, y]. Strides minimize the dimension of the window. For example, if the input window is 6X6 and the stride 2x2, the output window will be 3x3. This down-sampling is considered another advantage of the CNN as it exploits the correlation between the adjacent inputs. So, extra layers will offer more down-sampling, which can help reduce the number of insignificant features as we can see with three-layer CNN (1st output:32, 2nd output:64 and 3rd output:128) training faster than the two-layer CNN (1st output:64, 2nd output:128).

We also use a dropout unit is to reduce overfitting and ReLU on each of our convolution layers as activation which is preventing the vanishing gradient problem.