MNIST

Rui Melo, Group 36[1]

***Abstract*— This document contains works as a report done for Homework 5, where it was asked to compare the performance of Feed-Forward and Convolutional Neural Networks and assess the impact of changing some characteristics.**

***Keywords*— MNIST, Convolutional Neural Network, Feed-Forward, Machine Learning**

I. Introduction

The aim of this report is to compare Feed-Forward and Convolutional Neural Networks and assess the impact of changing the following characteristics:

Feed-Forward Neural Networks:

- Number of neurons in the hidden layer
- Activation function of the hidden layer
- Number of hidden layers
- No regularization vs L1/L2 regularization

Convolutional Neural Networks:

- Number of kernels/filters
- Pooling vs No Pooling
- Number of convolutional layers
- Fully connected layer between the convolutional and output layers
- No regularization vs Dropout

II. Feed-Forward Neural Networks:

A. Number of Neurons in the hidden layer

During this project, it was tested different amounts of neurons in an Hidden Layer. Usually, the number of Hidden Units usually varies between the number of units in the input layer and the output layer. It was used as the baseline model a model with a (28,28) input layer shape and an output layer using softmax activation. Between both layers, it was added a layer on which it was varied the number of neurons.

Table I
Number of Neurons in the Hidden layer

#Neurons	Accuracy w/ Training Set	Accuracy w/ Testing Set
20	0.9208	0.9051
50	0.8831	0.8769
88	0.9205	0.9064
200	0.8238	0.8135
400	0.9274	0.9188
700	0.8284	0.8257

The accuracy for the different amount of neurons in the hidden layer do not vary too much unless we compare it to the 700 and 200 record.

Work done as part of Machine Learning Course Homework at IST on May 2021

The main difference when training was the amount of time that a high number of neurons on a hidden layer model takes to train. This might imply that when having few neurons on an hidden layer, there is an higher generalization whereas in the opposite, we might expose the model to overfitting (which did not occur).

B. Activation Function

In this project, it was tried 4 different activation function in a multi-layer network. The base network was composed by the input layer, a layer where the activation was tested and the output one (similar to when varying the number of neurons in an hidden layer). The activation functions were: Relu, SoftPlus, Tanh and Sigmoid. All the activation functions provided a similar result, having an accuracy with the testing dataset around 0,975 .

Table II
Activation Function in the Hidden Layer

Activation Function	Accuracy w/ Training Set	Accuracy w/ Testing Set
Sigmoid	0.9900	0.9743
Tanh	0.9948	0.9758
ReLu	0.9944	0.9764
SoftPlus	0.9944	0.9739

C. Number of Hidden Layers

Increasing the number of hidden layers, improved the accuracy, especially from 1 hidden layer to 2. After that increment, it would not be necessary increasing even more the number of layers. In this scenario it was used 88 neurons in which layer to allow for a faster training process, but still maintaining a good performance.

Table III
Number of Hidden layers

#Hidden Layers	Accuracy w/ Training Set	Accuracy w/ Testing Set
0	0.9275	0.9246
1	0.9205	0.9064
2	0.9900	0.9743
3	0.9945	0.9726
4	0.9925	0.9635

D. Regularization

When it comes to regularization, L1 does not perform well in this scenario whereas L2 can be used, but it does not outperform having no regularization.

TABLE IV
REGULARIZATION

#Regularization	Accuracy w/ Training Set	Accuracy w/ Testing Set
No Reg.	0.9944	0.9775
L1	0.1124	0.1135
L2	0.9424	0.9398

III. CONVOLUTIONAL NEURAL NETWORKS

A very usual Convolutional Neural Network use is to detect patterns in images. This, in a practical use, allows the model to recognize edges, curves, etc. in early layers. The deeper the network is, the more complex these patterns are.

A. *Number of kernels*

In this study, it was used kernels where the window is assumed to be squared. The size of the square window was tested from 1 to 6, meaning the window would be (N x N)

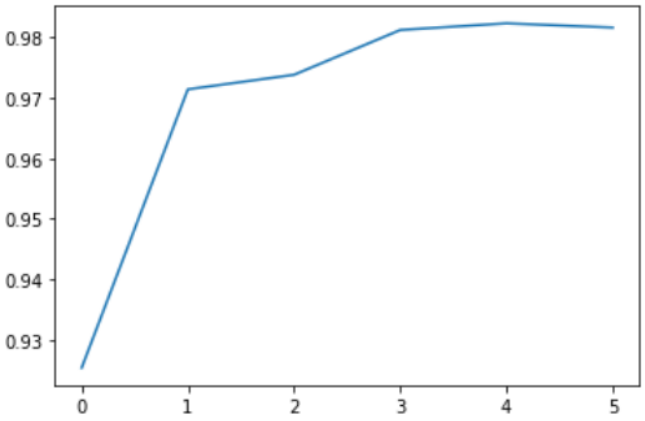

Fig. 1. Accuracy on Testing Dataset for each kernel Square Size

As seen, the accuracy on its own starts around 0.97, having a subtle increase from a window with 2x2 to one 3x3. The accuracy, however does not increase substantially on other iterations.

B. *Size of Filters*

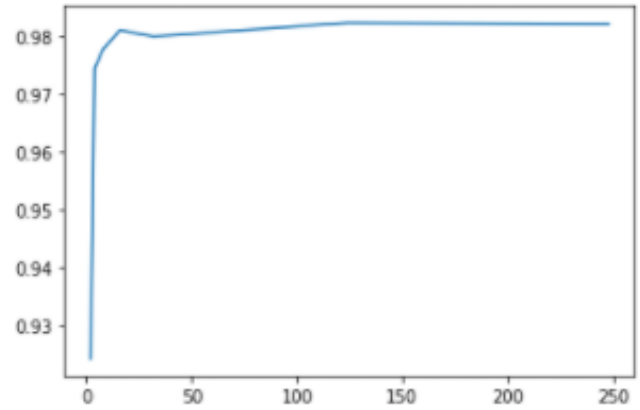

Fig. 2. Accuracy on Testing Dataset for each Filter Size

Regarding the size of the filters, it was tested with filter sizes equals to 2,4,8,16,32,128 and 256.The increasing size of the filters makes the training process to be incrementally longer. From the tests done, having the size range from 4 to 16 would be ideal, managing to have either good accuracy and decent training time.

C. *Pooling*

In order to test the impact of Pooling and not Pooling it was used a model with a Convolutional 2D layer with 16 filter and the kernel size (4,4). In order to test the difference that Pooling would actually do, it was inserted a Pooling layer. This layer would vary from being a Average Pooling Layer, a max Pooling Layer, a Global Average Pooling Layer or a Global Pooling Layer.

TABLE V
POOLING

Pooling	Accuracy w/ Training Set	Accuracy w/ Testing Set
No Pooling	0.9856	0.9809
Max Pool. Layer	0.9840	0.9811
Average Pool. Layer	0.9733	0.9733
Global Avg. Pool. Layer	0.1986	0.1973
Global Max Pool. Layer	0.8022	0.8115

From the results, it can be retrieved that Global Pooling does not perform well in this scenario, whereas both Max Pooling and Average Pooling perform relatively well, not surpassing the non pooling model, but having a very similar performance, specially between the model with no Pooling layer and the Max Pooling model.

D. *Number of Fully-Connected Convolutional Layers*

In order to access the impact of incresing the number of Convolutional Layers, it was used a baseline model that uses a Convolutional 2D layer with 16 filter and the kernel size (4,4). After that, to increase the number of Convolutional Layers, it was added extra layers, reducing the kernel size progressively.

TABLE VI
NUMBER OF CONVOLUTIONAL LAYERS

Convolutional Layers	Accuracy w/ Training Set	Accuracy w/ Testing Set
1	0.9856	0.9856
2	0.9844	0.9829
3	0.9834	0.9805
4	0.9873	0.9812
5	0.9845	0.9818

E. *Dropout vs No Regularization*

TABLE VII
NUMBER OF CONVOLUTIONAL LAYERS

Regularization	Acc. w/ Training Set	Acc. w/ Testing Set
No Reg. 1 Conv. Layer	0.9856	0.9856
Dropout 1 Conv. Layer	0.9812	0.9785
No Reg. 2 Conv. Layer	0.9844	0.9829
Dropout 2 Conv. Layer	0.9902	0.9855
No Reg. 3 Conv. Layer	0.9834	0.9805
Dropout 3 Conv. Layer	0.9916	0.9862
No Reg. 4 Conv. Layer	0.9873	0.9812
Dropout 4 Conv. Layer	0.9934	0.9871
Batch Norm. 1 Conv. Layer	0.9955	0.9831
Batch Norm. 2 Conv. Layer	0.9964	0.9870
Batch Norm. 3 Conv. Layer	0.9958	0.9856
Batch Norm. 4 Conv. Layer	**0.9967**	**0.9878**

IV. CONCLUSION

Overall, it is possible to conclude that having a Convolutional Neural Network increase the performance when comparing to a Feed-Forward Neural Network. Also in this study, when applied a Convolutional Neural Network, the difference in accuracy between the Testing and Training dataset is smaller, might revealing that when applying to a context similar to this study's dataset, the occurrence of overfitting might be rarer.

It is good to note that having multiple layers perceptron helps in getting knowledge when it comes to non-linear relations. If a problem has mostly linear relations, a perceptron model with only an input and output layer would probably be enough. In this context, we have a set of images. Not all the images are well-centered or have the same amount of brightness. This might be a reason why multi-layer model can turn out to be beneficial over the single-layer model.

When looking into more detail into Convolutional Neural Networks present in this study, is it safe to assume that is more beneficial increasing the number of layers than just increasing its dimension and having a regularization in this Neural Networks are useful, increasing the overall accuracy. This effect is particularly noticeable when applying regularization. When discussing the regularization in particular, applying a batch Normalization provides a better accuracy than using dropout.

V. CONCLUSION'S TEST

After gathering information about the different Neural Networks and access the impact of changing different characteristics, it was tested a final model. It was tested a Convolutional Neural Network with 5 Convolutional Layers. Those layers started with 6x6 kernel until 2x2. Also, it was applied Max Pooling and Batch Normalization. It turned out with an accuracy of 0.9953 regarding the training dataset and an accuracy of 0.9895 regarding the testing dataset.

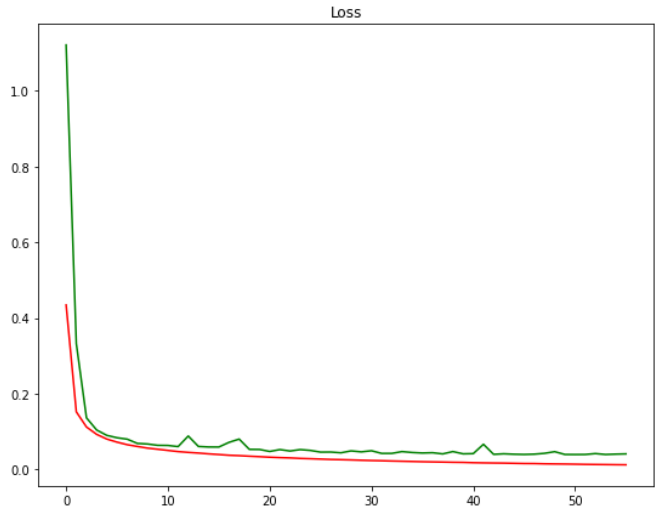

Fig. 3. Final Model Loss

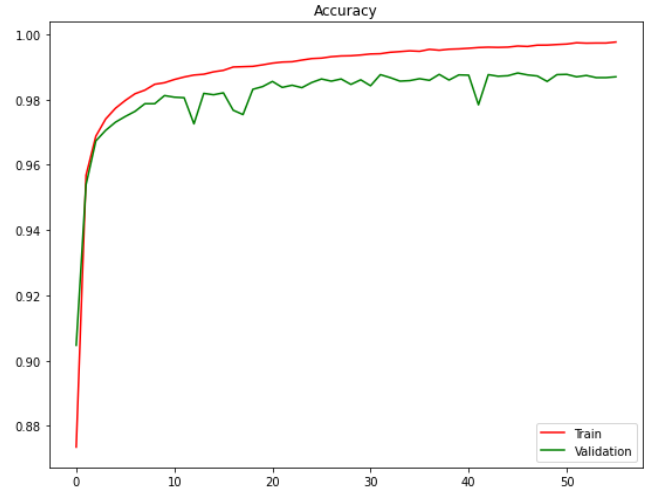

Fig. 4. Final Model Accuracy

VI. LINKS

Feed-Forward Neural Networks:
For accessing the notebook, see:
`https://colab.research.google.com/drive/1-S0eb95FEIItsXWiz6Q0CnRXGRAVZ5GQ?usp=sharing`
or:
`https://github.com/rufimelo99/MNIST/blob/main/MNIST_FFN.ipynb`

Convolutional Neural Networks:
For accessing the notebook, see:
`https://colab.research.google.com/drive/1EZ38zKfTEzLauGAhJVoSjFQZbVr5Ymgk?usp=sharing`
or:
`https://github.com/rufimelo99/MNIST/blob/main/MNIST_CNN.ipynb`