Universidade Federal de Pernambuco – UFPE

Centro de Informática – CIn

Neural Networks

Digits recognition with MLPs

Recife – 2019

Neural Networks

Digits recognition with MLPs

Analysis of the impact of parameter variation on the success rate in digits recognition with MLPs

Written by:

Jacqueline Alves Barbosa - jab3@cin.ufpe.br,

Lucca França Gomes Ferreira - lfgf@cin.ufpe.br.

Summary

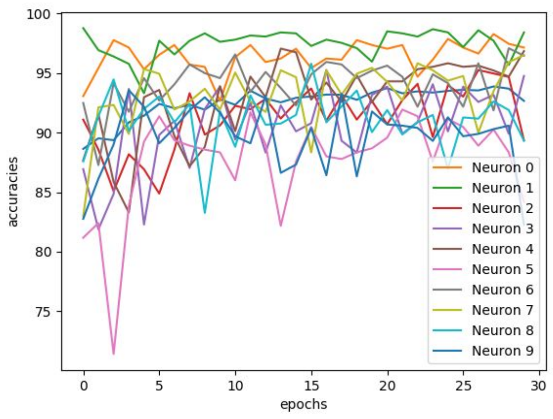
1. Base execution

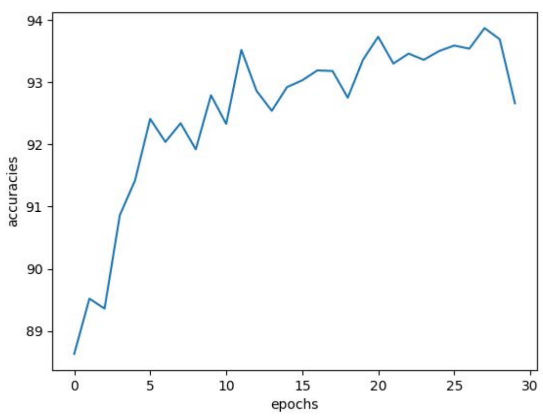
We started training and varying the parameters using as standard values the ones used by Michael Nielsen on his online book *Neural Networks and Deep Learning*.

Execution with:

- 3 layers (img_sizeˆ2, 30, 10)
- 30 epochs
- Mini-batch size of 10
- Learning rate of 3.0

With these parameters we achieved a final success rate of 92.66%.





2. Varying the number of epochs

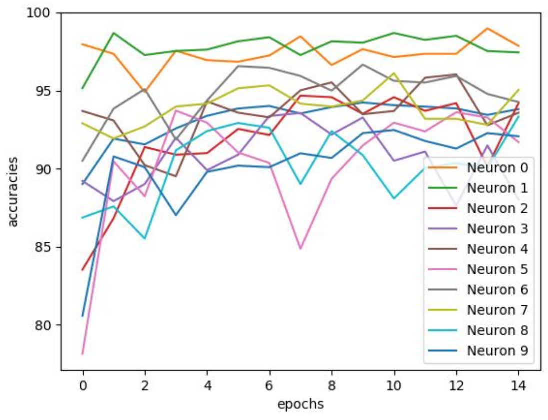
Using the base training, we decided to vary the number of epochs, with higher and lower values, in order to understand the impact of this variation on the training performance.

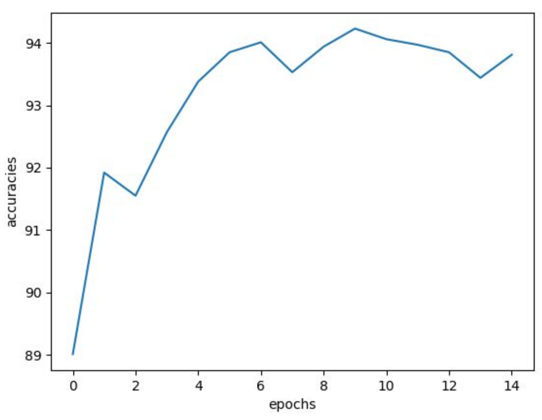
2.1. Reducing by half the number of epochs

Execution with:

- 3 layers (img_sizeˆ2, 30, 10)
- 15 epochs
- Mini-batch size of 10
- Learning rate of 3.0

With these parameters we achieved a final success rate of 93.81%



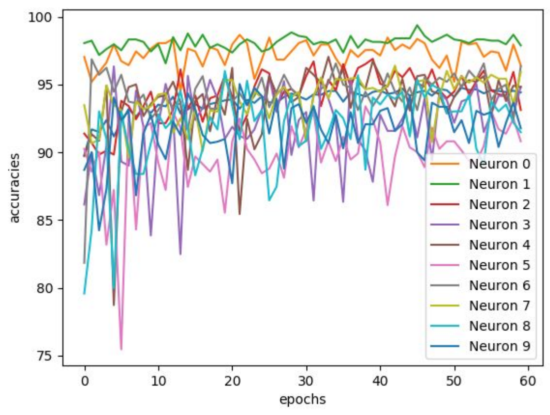


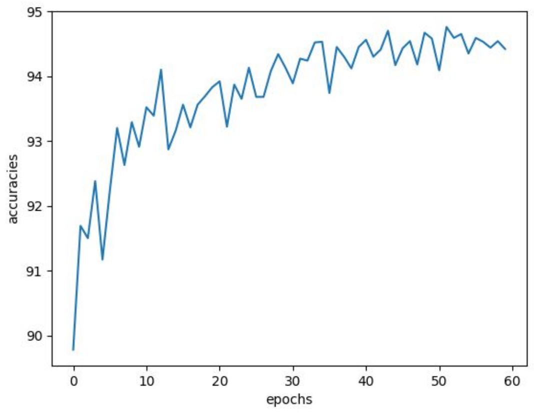
2.2. Doubling the number of epochs

Execution with:

- 3 layers (img_sizeˆ2, 30, 10)
- 60 epochs
- Mini-batch size of 10
- Learning rate of 3.0

With these parameters we achieved a final success rate of 94.42%.





With the variation of the number of epochs results, we could conclude that changes made to this parameter individually have no significant impact on the final success rate. However it is possible to notice that raising the number of epochs, the success rate on each class tend to vary less and have an improvement in the average accuracy variation when the training is close to the end.

3. Varying the mini-batch size

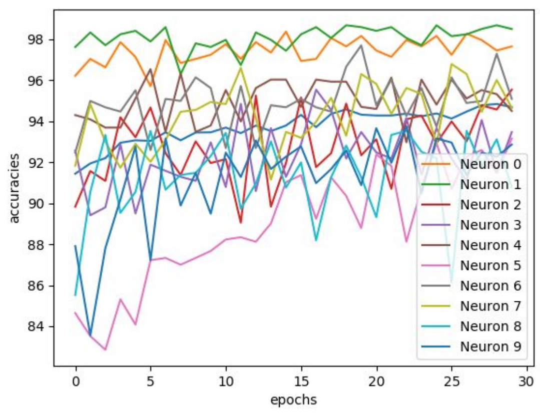
With the previous results achieved varying the number of epochs, we decided to vary another parameter individually, with higher and lower values, in order to understand the impact of this variation on the training performance.

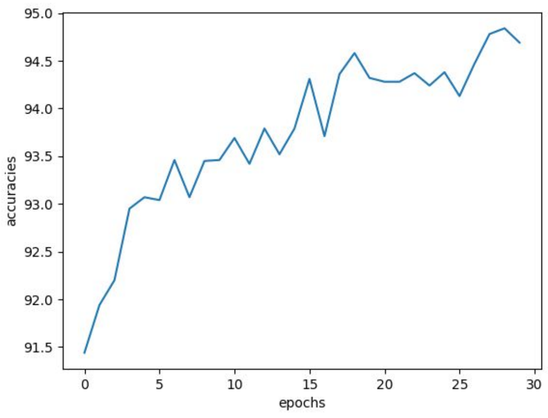
3.1. Increasing by 10 times the mini-batch size

Execution with:

- 3 layers (img_sizeˆ2, 30, 10)
- 30 epochs
- Mini-batch size of 100
- Learning rate of 3.0

With these parameters we achieved a final success rate of 94.69%.



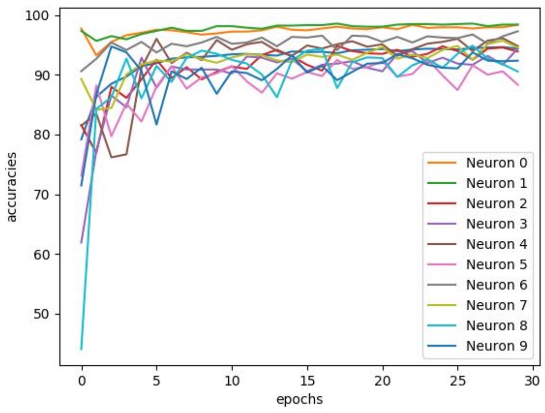


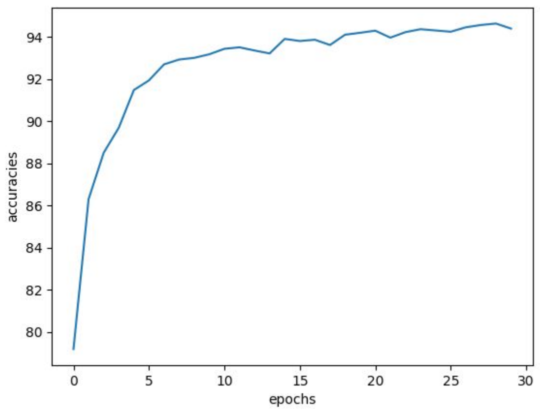
3.2. Increasing by 100 times the mini-batch size

Execution with:

- 3 layers (img_sizeˆ2, 30 e 10)
- 30 epochs
- Mini-batch size of 1000
- Learning rate of 3.0

With these parameters we achieved a final success rate of 94.4%.





With the variation of the mini-batch size results, we could conclude that: with a bigger batch size (in this case, 0.2% of the training set) the overall performance results are similar with the training with smaller size (batch size of 0.002% or batch size of 0.02% of the training set), however after analysing the performance per class individually, we could notice that bigger sizes ensure a smoother convergence, with less sudden variation on the success rate between epochs.

We could also observe that using bigger batch sizes, we have a worse performance per neuron individually on the first epochs, as we can see on neurons 3 and 8.

4. Varying the learning rate

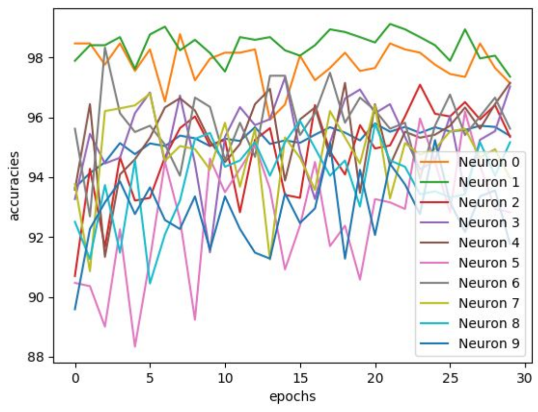
Using the base training, we decided to vary the learning rate, with higher and lower values, in order to understand the impact of this variation on the training performance.

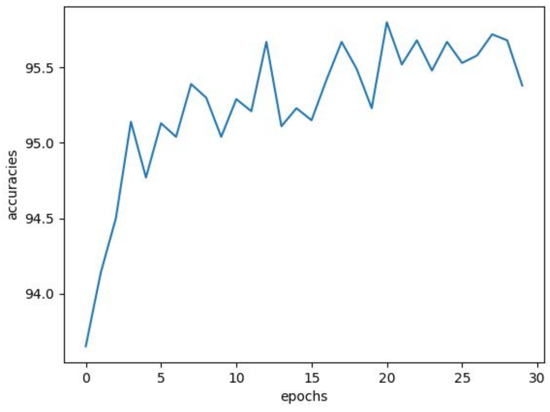
4.1. Learning rate of 1.0

Execution with:

- 3 layers (img_sizeˆ2, 30, 10)
- 30 epochs
- Mini-batch size of 10
- Learning rate of 1.0

With these parameters we achieved a final success rate of 95.38%.



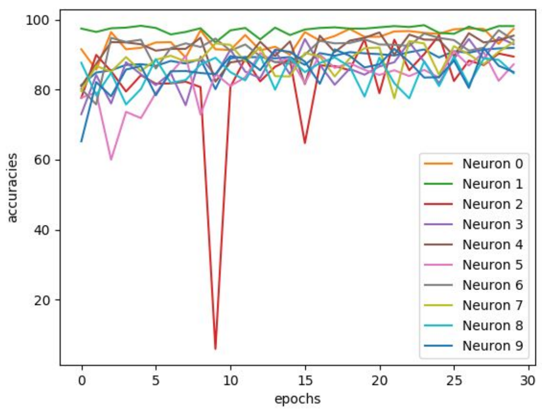


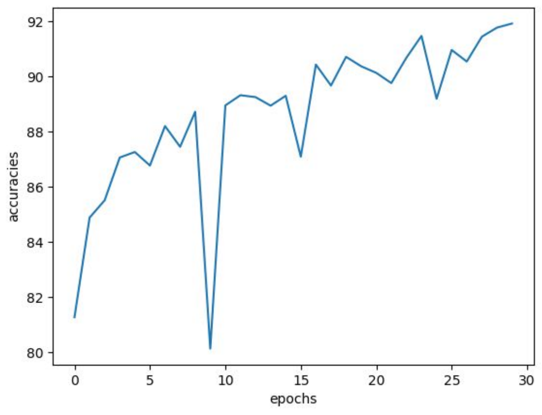
4.2. Learning rate of 5.0

Execution with:

- 3 layers (img_sizeˆ2, 30, 10)
- 30 epochs
- Mini-batch size of 10
- Learning rate of 5.0

With these parameters we achieved a final success rate of 91.2%.





We could conclude that with a higher learning rate, the initial performance was worse compared to a lower learning rate and, with a smaller variation, made the overall final success rate lower.

5. Increasing number of epochs and mini-batch size and decreasing learning rate

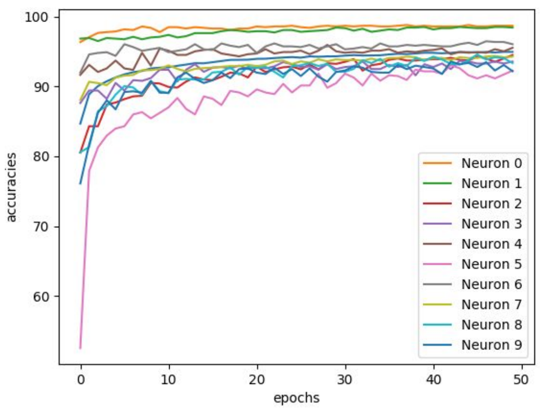
We noticed that the increase in the mini-batch size itself resulted in a decrease in the variation of the success rate and the decrease in the learning rate provided a better overall final accuracy, despite having a larger variation. We then decided to train with these two parameters in conjunction with increasing the number of epochs to analyze the joint impact of these three parameters on the final result.

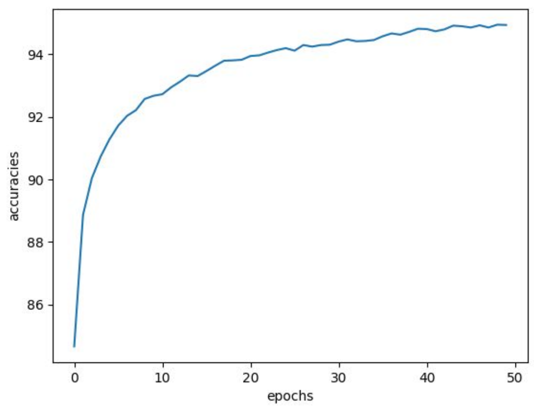
5.1. With 50 epochs

Execution with:

- 3 layers (img_sizeˆ2, 30, 10)
- 50 epochs
- Mini-batch size of 1000
- Learning rate of 1.0

With these parameters we achieved a final success rate of 94.93%.



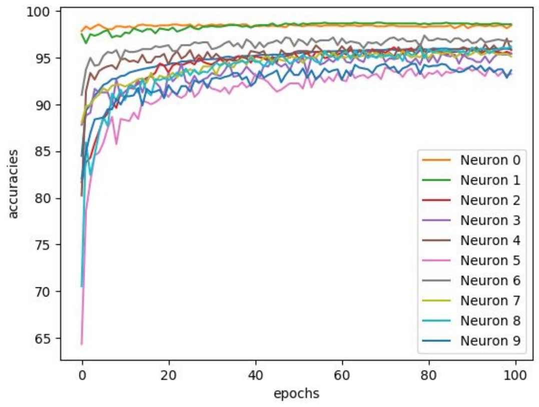


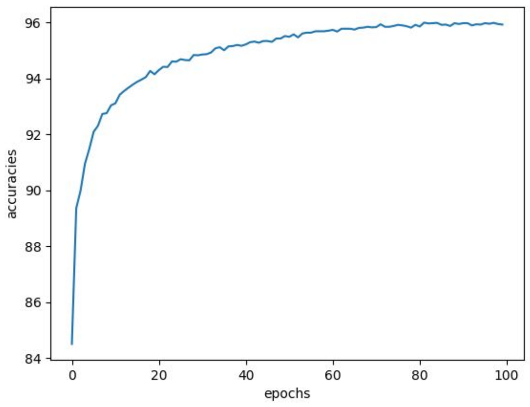
5.2. With 100 epochs

Execution with:

- 3 layers (img_sizeˆ2, 30, 10)
- 100 epochs
- Mini-batch size of 1000
- Learning rate of 1.0

With these parameters we achieved a final success rate of 95.93%.





We could conclude that with more epochs there was more time for the success rate to grow and to reach a better result at the end of the execution.

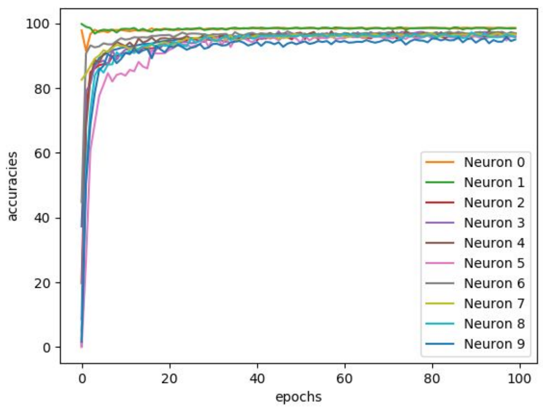
6. Adding a layer

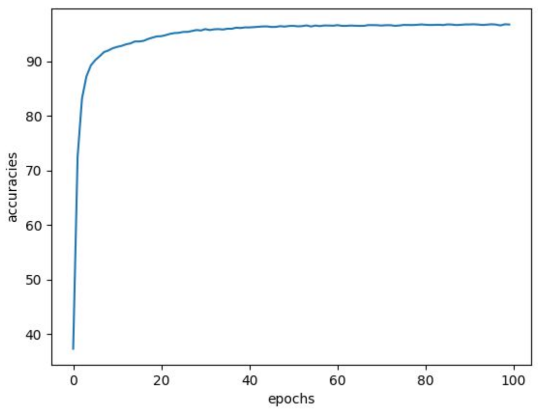
From our analysis, we took the best result we could achieve (with 100 epochs, mini-batch size of 1000 and learning rate of 1.0) and added a new layer to try to achieve a greater smoothing of the function. We added a layer with 30 extra neurons and we got these results:

Execution with:

- 4 layers (img_sizeˆ2, 30, 30, 10)
- 100 epochs
- Mini-batch size of 1000
- Learning rate of 1.0

With these parameters we achieved a final success rate of 96.77%.





At first glance we can see that success rate have increased faster with one more layer if we look at all classes, and also have a smaller variation. The final result of the total accuracy was also better, having an increase of almost 1% compared to the results obtained with 3 layers.

7. Increasing the intermediate layer

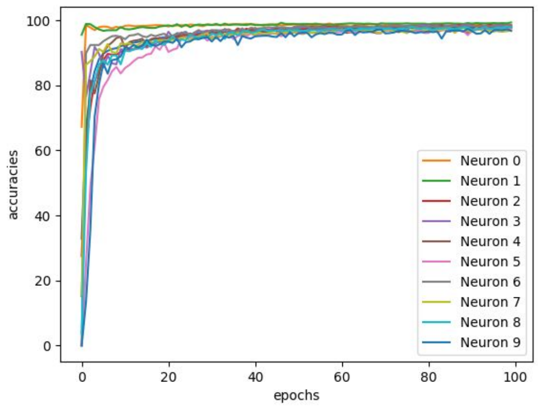
As a result of the addition of a new layer, we decided to slightly increase the intermediate layer so that the reduction of neurons from one to the other would be smoother. With this we made the following execution:

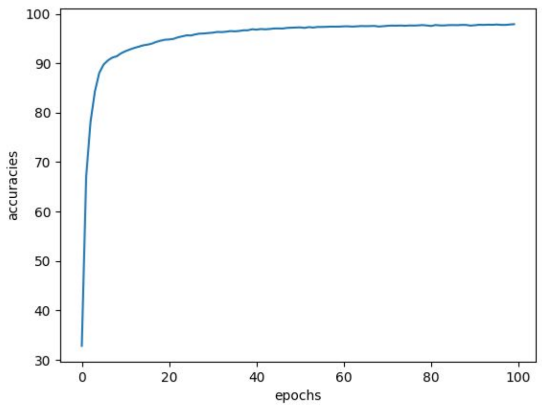
7.1. Intermediate layer with 100 neurons

Execution with:

- 4 layers (img_sizeˆ2, 100, 30, 10)
- 100 epochs
- Mini-batch size of 1000
- Learning rate of 1.0

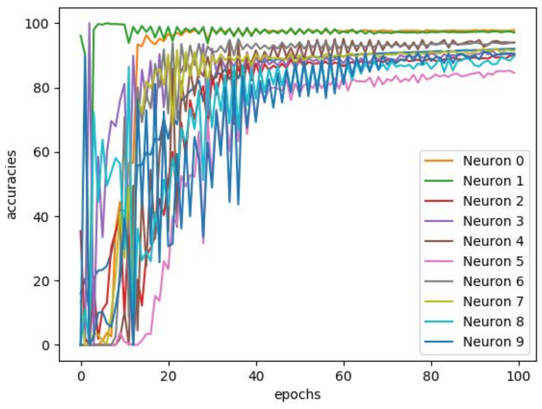
With these parameters we achieved a final success rate of 97.89%.

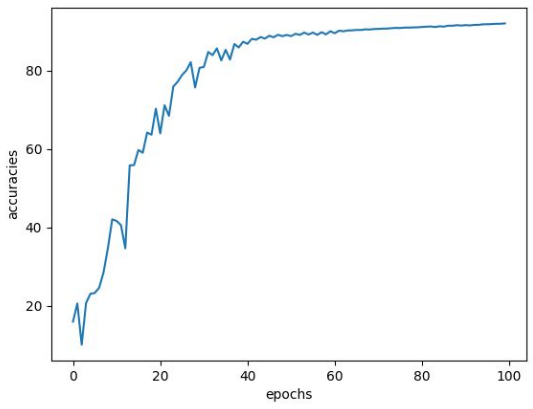




7.2. Two 100 neurons intermediate layers

Increasing the mini-batch size to 10,000 and adding an additional 100 neuron intermediate layer (img_sizeˆ2, 100, 100, 10) we achieved an overall performance of 92.09%.





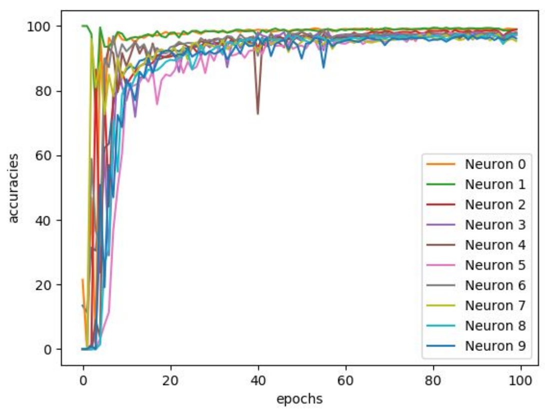
7.3. 500 neurons on the second layer

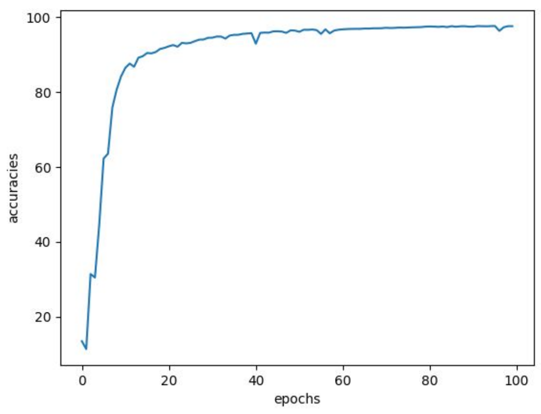
We then tried to enlarge the intermediate layer in order to smooth the passage from one layer to the other.

Execution with:

- 4 layers (img_sizeˆ2, 500, 100, 30, 10)
- 100 epochs
- Mini-batch size of 1000
- Learning rate of 1.0

With these parameters we achieved a final success rate of 97.63%.





We can see that there was no significant change in the final result, there was a slight decrease in the total final accuracy.

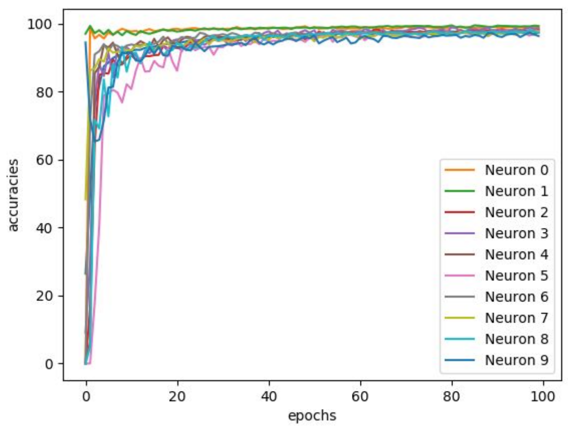
8. Uniformity of intermediate layers

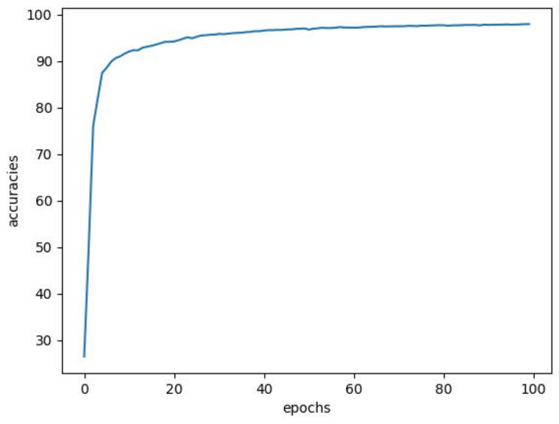
With this result we could assume that training with more than one intermediate layer results in better values if combined with a larger batch_size (0.2 to 0.4% of the dataset). With this in mind we applied two intermediate layers, each with 200 neuron.

Execution with:

- 4 layers (img_sizeˆ2, 200, 200, 10)
- 100 epochs
- Mini-batch size of 1000
- Learning rate of 1.0

With these parameters we achieved a final success rate of 97.91%.





9. Conclusion

Following our tests, we concluded that the addition of more intermediate layers allows the network to have a smoother convergence, with the individual accuracy of each class experiencing fewer abrupt variations. We also realized that we were able to achieve this convergence using a low (1.0) learning rate and a batch size of 0.2% of the total dataset. These conclusions apply specifically to the problem under analysis and may have different results depending on the problem to be addressed.