Mini-Project 4: Reproducibility in ML
Reproducing LeNet
COMP 551: Applied Machine Learning

Antoine Bonnet
260928321
McGill University

Dragos Secrieru
260923248
McGill University

Cyril Saidane
260706157
McGill University

Abstract

This paper aims to reproduce the results obtained through the LeNet convolutional neural network model as published by Yann LeCun et al. in 1998. We reproduced the same architecture and trained the model on the MNIST dataset. We performed experiments to test two claims made by the authors of the paper regarding the LeNet performance and its unusual robustness to overtraining. We found that three versions of this model, called LeNet-1, LeNet-4 and LeNet-5, yielded approximately test accuracies of 98.43, 98.72 and 98.71% respectively, which are similar to the reported accuracies of 98.3, 98.8 and 99.05%. We also validated that the LeNet model is robust to overtraining. However, we obtained data contradicting the authors' hypothesis regarding the causes of this robustness.

1 Introduction

This report aims to reproduce the results obtained through the LeNet architecture used for image classification on the MNIST dataset, as published in the paper "Gradient-Based Learning Applied to Document Recognition" by Yann LeCun, Leon Bottou, Patrick Haffner and Yoshua Bengio in 1998 [2].

The publication of this paper was a pivotal moment in machine learning, because it introduced the one of the earliest convolutional neural networks (CNN). The paper goes over at least a dozen different gradient based methods of learning for handwritten text recognition, culminating in their own state-of-the-art model for the time; LeNet-5. Our main goal was to verify the performance of multiple versions of the LeNet model as showcased in the paper. We also tested the validity of claims made in the paper that were not substantiated with data.

2 Scope of reproducibility

The authors remark that, while the common phenomenon of over-training yields decreasing test accuracy when training a model for too long, they observe that no such phenomenon occurred for LeNet-5. They posit as hypothesis that this may be due to the fact that the learning rate was kept relatively large throughout training, thus preventing the stagnation of the weights in a harmful local minimum. All in all, we decided to test two specific claims:

Claim 1: LeNet-1, LeNet-4 and LeNet-5 reach test accuracies on MNIST of 98.3%, 98.8% and 99.05% respectively.

Claim 2: The test error rate of LeNet-5 stabilizes after 10 training epochs on MNIST. The authors remark that the problem of most classifiers is that their recognition accuracy is determined by the ability of the scientist to come up with good features. Their main point is that for it to be applicable to more problems, one should focus instead on automatic learning. To that end they devised a series of different CNN models, culminating in LeNet-5.

3 Methodology

3.1 Model descriptions

We analysed 3 different models, which were variations of convolutional neural networks; LeNet-1, LeNet-4 and LeNet-5. Since training these networks on the MNIST dataset was manageable, we did not use pretrained models from the paper. The architecture of each model is detailed below.

LeNet-1 contains a total of 5 layers, for a total of 3,246 trainable parameters.

1. 2D convolutional layer with 4 filters of size 5×5, without paddingand with tanh activation.
2. Average pooling layer with stride 2.
3. 2D convolutional layer with 12 filters of size 5×5.
4. Average pooling layer with stride 2.
5. Fully-connected layer of size 10, with softmax activation.

LeNet-4 contains a total of 6 layers, for a total of 51,050 trainable parameters.

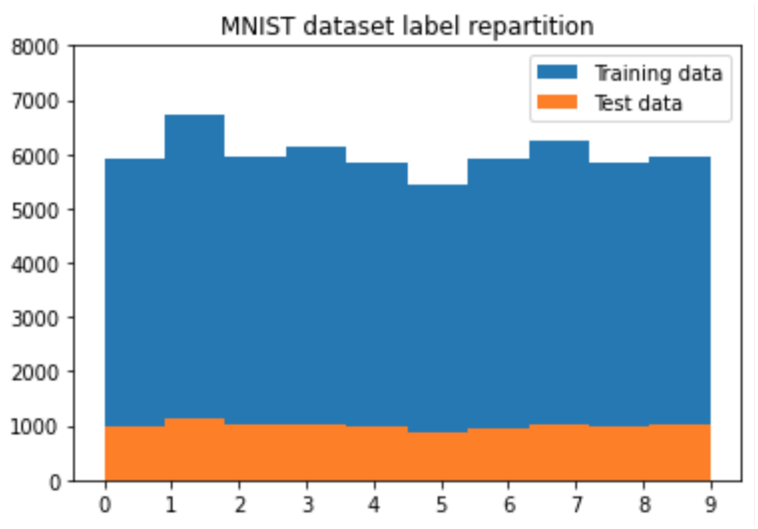
1. 2D convolutional layer with 4 filters of size 5×5 with tanh activation.
2. Average pooling layer with stride 2.
3. 2D convolutional layer with 16 filters of size 5×5 with tanh activation.
4. Average pooling layer with stride 2.
5. 2D convolutional layer with 120 filters of size 5×5 with tanh activation.
6. Fully-connected layer of size 10 with softmax activation.

LeNet-5 contains a total of 7 layers, for a total of 61,706 trainable parameters.

1. 2D convolutional layer with 6 filters of size 5×5 with tanh activation.
2. Average pooling layer with stride 2.
3. 2D convolutional layer with 16 filters of size 5×5 with tanh activation.
4. Average pooling layer with stride 2.
5. 2D convolutional layer with 120 filters of size 5×5 with tanh activation.
6. Fully-connected layer of size 84 with tanh activation.
7. Fully-connected layer of size 10 with softmax activation.

3.2 Datasets

The MNIST dataset is a dataset composed of 70,000 gray-scaled images showing articles of clothing. This dataset was first designed by the authors of this paper, and has since remained a benchmark for image classification models. The images are low resolution and encoded into 28x28 matrices of pixel values. They are labelled with one of 10 classes (dress, coat, trousers, etc.). The dataset is split into 60K instances for training and 10K for testing. The class repartitions for both training and testing sets are shown below.



As designed by the authors, the pixel values of each image in the training and test sets were normalized within a range of $[0, 1]$. All of the LeNet models have a last layer of size 10×1, which corresponds to each of the digits to be classified. The cross-entropy loss function is applied on this layer to back-propagate the gradients and train the whole network.

3.3 Hyperparameters

All three models were trained using the Adam optimization algorithm. We followed the architecture design of the authors by mimicking their choice of layers, filter sizes, padding and activation functions. We used the same hyperparameters, except for the learning rates which we changed in order to test their hypotheses. For more information consult 4.

The learning rates used by the authors to train LeNet-5 were "0.0005 for the first two passes, 0.0002 for the next three, 0.0001 for the next three, 0.00005 for the next 4, and 0.00001 therafter." We used these learning rates hyperparameters for most of our experiments, but we also conducted additional experiments by modifying the final learning rate.

3.4 Experimental setup and code

The models were coded from scratch with Keras from TensorFlow by reading through the paper's detailed description of the model architectures, with some help from experimental blogposts (see [1] and [3]). The models were trained with regards to the accuracy measure, i.e. the proportion of correctly labelled images.

3.5 Computational requirements

As the computational requirements for training these models were low, we used the Google Colab environment directly. The longest training times were for LeNet-5, for which training 20 epochs took approximately 30 minutes.

4 Results

We found that we were able to reproduce Claim 1 of the paper related to the performance of LeNet-1, 4 and 5. We also support their claim regarding the fact that the test accuracy remains stable as we increase the number of training epochs, thus validating Claim 2. However, we obtained data contradicting the authors' hypothesis regarding the reason why Claim 2 holds.

4.1 Results reproducing original paper

4.1.1 Result 1

We trained the LeNet-1, LeNet-4 and LeNet-5 models for 20 training epochs, as the authors had indicated. We found that these three models achieved a final test accuracy of 98.43%, 98.72% and 98.71% respectively on MNIST. If we compare these performances to the paper results (98.3%, 98.8% and 99.05%), we find that obtained very similar accuracies (except for LeNet-5 which was slightly lower) and so Claim 1 is validated. The training and test accuracies over 20 training epochs of each model are shown below in Figures 1, 2 and 3 respectively.

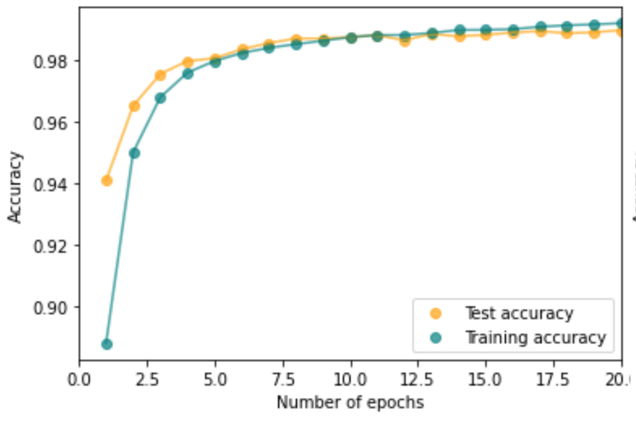


Figure 1: Accuracy of LeNet-1 on MNIST

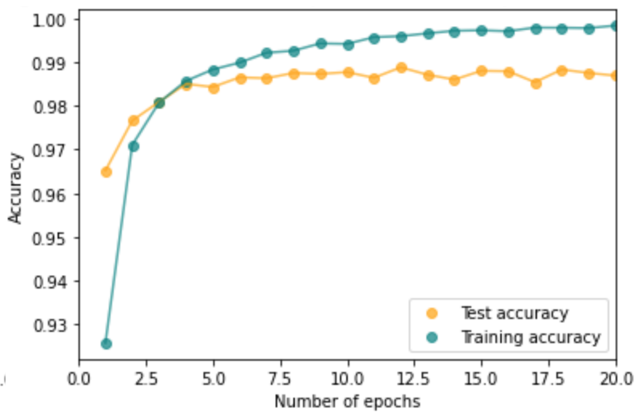


Figure 2: Accuracy of LeNet-4 on MNIST

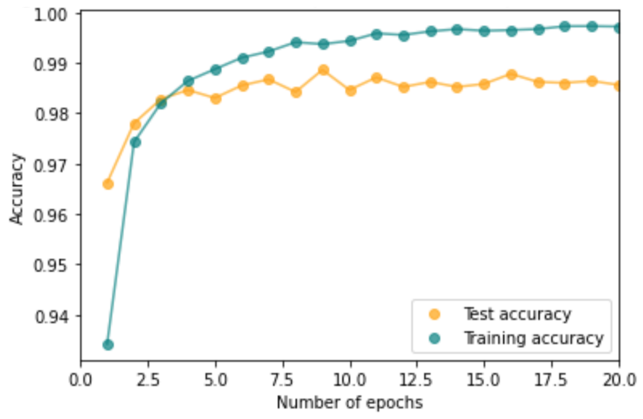


Figure 3: Accuracy of LeNet-5 on MNIST

4.1.2 Result 2

We also tested Claim 2, which states that the LeNet-4 test accuracy does not decrease after reaching a peak as we train the model over larger amount of training epochs. As shown in Figure 3, we observed that the test accuracy tends to stagnate after reaching its peak, and does not start decreasing as we overtrain our model. Thus, we achieved the same results as in the paper, and Claim 2 was also validated. However, the authors' hypothesis regarding why this behavior occurs was unsubstantiated, so we decided to design an experiment to use different learning rates so as to test this hypothesis.

4.2 Results beyond original paper

We focused on two main claims that the paper made that were unsupported by any data. We will now go over each of them in detail.

4.2.1 Additional Result 1

On page 10 of the paper, the authors discuss the fact that for most CNN models, when over-training occurs the training errors keeps decreasing with subsequent epochs but the test error starts slowly increasing after going through a minimum. The authors found that this did not happen with LeNet-5. A possible explanation they gave for that was they used relatively high learning rates, thus preventing the optimization algorithm from getting stuck in local minima.

Our first additional experiment can be formulated as follows:

Experiment 1: Does the test error rate still stabilize even with lower learning rates?

We performed this experiment by using the same architecture as the authors, but modifying the learning rates used after the ninth epoch onwards. They used learning rates of "0.0005 for the first two epochs, 0.0001 for the next three, 0.00005 for the next 4 and 0.00001 thereafter". We tested the following learning rates: 10^{-6}, $5 \cdot 10^{-7}$ and 10^{-7}.

In the 4 plots, below we compared the accuracy yielded by the LeNet-5 model with the learning rate $\alpha = 10^{-5}$ from the paper (see Figure 4) with 3 other learning rates that we decided on (see Figures 5, 6 and 7 respectively). We found that decreasing the learning rate did not yield a decreasing test accuracy as the authors had hypothesized. Even with the lowest learning rate of 10^{-7}, we saw that the test accuracy remained stable. Thus, the hypothesis that relatively high learning rate explains why the test accuracy remains high despite overtraining is not validated by our data, since smaller learning rates yield the same results. This phenomenon deserves more investigation since the explanation given in the paper was shown to not be true empirically.

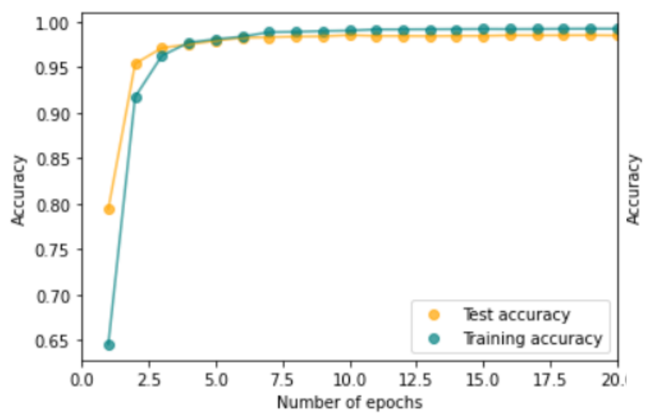


Figure 4: Accuracy for learning rate: $\alpha = 10^{-5}$ (Paper)

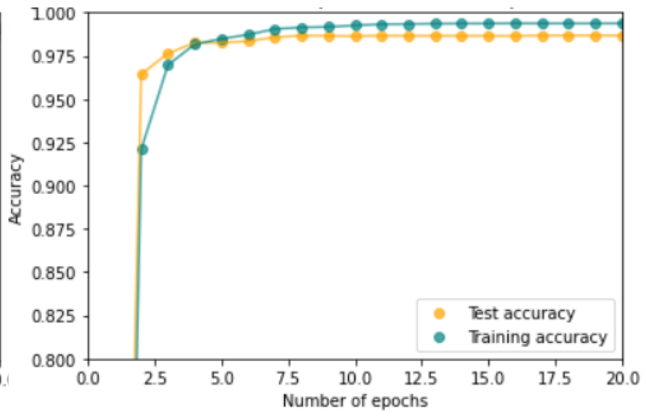


Figure 5: Accuracy for learning rate: $\alpha = 10^{-6}$

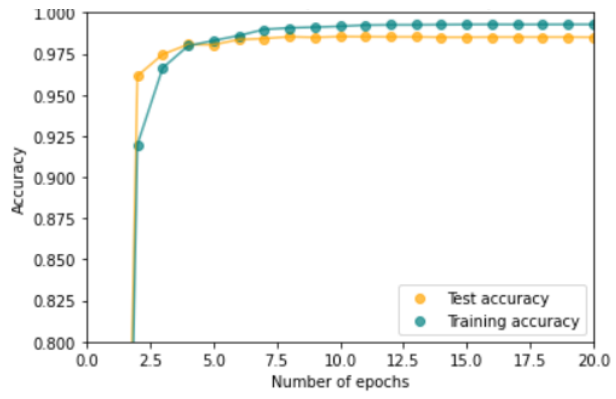


Figure 6: Accuracy for learning rate: $\alpha = 5 \cdot 10^{-7}$

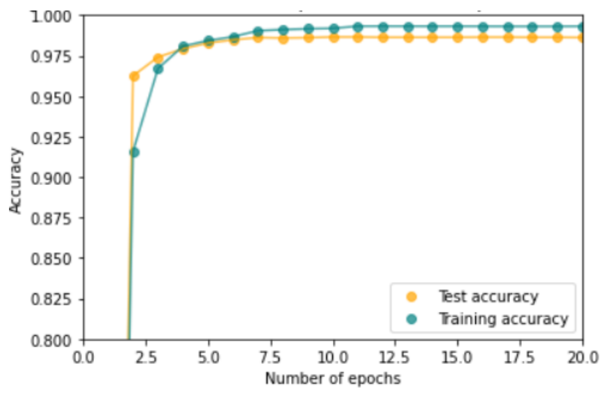


Figure 7: Accuracy for learning rate: $\alpha = 10^{-7}$

4.2.2 Additional Result 2

The paper made an emphasis on using tanh as an activation function in their LeNet models (see Appendix A in the paper). We decided to experiment with that, and try to see what would happen if one would remplace it with a different activating function. In order to do that we ran 2 separate models, one with relu activation functions, and another with sigmoid activation functions.

Our second additional experiment can be formulated as follows:

Experiment 2: What is the effect on the MNIST test accuracy of different activation functions for LeNet-5 such as ReLu and sigmoid?

We performed this experiment by training a model with the same architecture but with the ReLu activation function on every single layer (except the last one, which keeps the sigmoid activation). We found that the model with ReLu activation functions clearly outperformed sigmoid activation function with a final test accuracy of 98.85% vs 95.55%, as shown in Figures 8 and 9. Given that the ReLu function yielded about the same accuracy as the tanh function, it is unclear to us why the paper chose the latter.

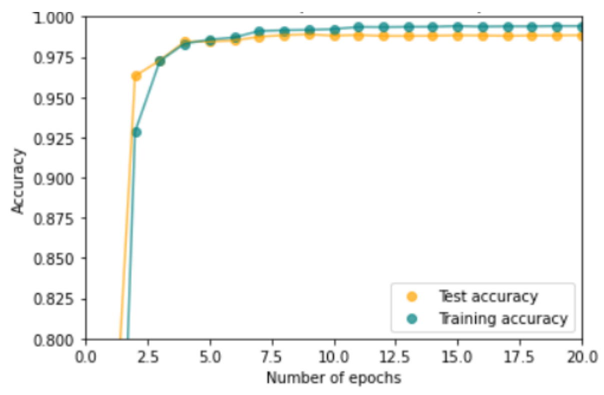


Figure 8: Accuracy of ReLu LeNet-1 on MNIST

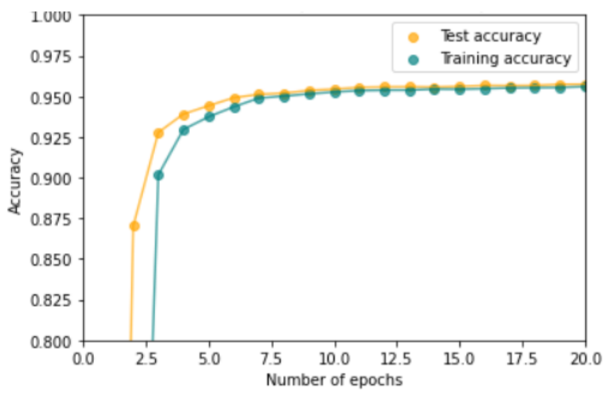


Figure 9: Accuracy of sigmoid LeNet-5 on MNIST

5 Discussion

We found that the MNIST test accuracy reported in the paper for LeNet-1, 4 and 5 were accurate, thus validating Claim 1. However, one of the weakness of our analysis is that we did not average out over multiple trained models in order to decrease the variance. This choice was made due to the lack of sufficient computational resources.

Our experiments also validated Claim 2, which stated that the test error does not start decreasing when overtraining. However, we also found that the authors' hypothesis regarding why the overtraining phenomenon occur based on their choice of large learning ratse was contradicted by our results. We found that decreasing the learning rate did not affect the tendency of LeNet-5 to have a stable test accuracy with subsequent epochs. While the number of unique learning rate that we tried were limited, it nonetheless presents a counterarguments to the hypothesis presented in the paper.

5.1 What was easy

Since we worked on reproducing a seminal paper with over 40K citations, it was easy to find a lot of resources online regarding the implementation of the different LeNet models, as well as online discussions based on their results. The sources we consulted were added to the bibliography. Also using Keras, we found it really easy to implement some ideas in the paper such as changing the learning rate depending on the epoch.

5.2 What was difficult

One of the difficult things that we found was that with the paper totaling 46 pages, we were unsure about which results to test. We had no choice but to focus on a small subset of claims made in the paper. The authors carried other experiments regarding the test accuracy depending on data augmentation techniques through an expansion of the MNIST dataset with image modifications. We did not have the knowledge regarding how to apply such data augmentation techniques to image data, so we decided not to reproduce this particular experiment.

Furthermore, we could not understand the choice of the tanh activation function. The paper goes into this in detail in Appendix A but we could not find argumentative reasons on why this particular choice was adopted, although we know that the ReLu activaiton function came into fashion after this paper was published.

5.3 Communication with original authors

We did not communicate with the authors.

References

[1] Azel Daniel. Understanding lenet: A detailed walkthrough, Sep 2020. `https://towardsdatascience.com/understanding-lenet-a-detailed-walkthrough-17833d4bd155`, Last accessed on 2022-04-25.

[2] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition, 1998.

[3] Sik-Ho Tsang. Review: LeNet-1, LeNet-4, lenet-5, boosted LeNet-4, Mar 2019. `https://sh-tsang.medium.com/paper-brief-review-of-lenet-1-lenet-4-lenet-5-boosted-lenet-4-image-classification-1f5f809dbf17` Last accessed on 2022-04-25.