Homework IV

João Martins, Pedro Reis
Instituto Superior Técnico
Av. Rovisco Pais 1, 1049-001 Lisbon
Machine Learning 2019/2020
IST (MEIC-A / MEIC-T)
Group 68

Abstract

Using Tensorflow/Keras this work explores the use of different neural network architectures to achieve the highest performance on the test set of the famous MNIST dataset. More specifically it explores the use of Feed-Forward and Convolutional Neural Networks, experimenting on a different number of hidden layers and finally assesses the impact of different regularization methods.

1. Introduction

The MNIST dataset consists in a database of handwritten digits: a set of images and their corresponding labels. It poses the challenge of correctly and efficiently classifying the elements of the test set present in this database.

A valid solution to this problem is to use neural networks to help classifying the images present in the test set. These algorithms can specifically be used in image recognition and they are able to classify just about anything, from text to images, audio files, and videos.

The goal of this project is to use Tensorflow/Keras [1] to explore the use of different neural network architectures to correctly classify and achieve the highest performance on the test set of the MNIST dataset. This project focuses on exploring the use of Feed-Forward and Convolutional Neural Networks as well as exploring the number of hidden layers in the network's structure and assessing the impact of different regularization methods in the classification.

We begin in section 2 by briefly neural networks that were studied. Section 3 gives an overview of the MNIST dataset. In Section 4 we'll present and discuss the experimental results. The conclusion is then presented in section 6. All the computations done are present in a notebook in Colab (`https://colab.research.google.com/drive/19sjAna67uHlePjlIaOzdtgLkraC972e1?usp=sharing`).

2. Neural Networks

2.1. Feed-Forward Networks

Feed-forward neural networks, or multilayer perceptrons (MLPs), are the classic deep learning models. A feed-forward network defines a mapping y = f(x, θ) and learns the value of the parameters that result in the best function approximation. These models are called feed-forward because information flows through the function being evaluated from **x**, through the intermediate computations used to define f, and finally to the output y. There are no feedback connections in which outputs of the model are fed back into itself.

2.2. Convolutional Neural Networks

> *Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers.* [2]

Convolutional neural networks, or CNNs, are a specialized kind of neural network for processing datat hat has a known grid-like topology. Convolutional networks have been tremendously successful in practical applications. The name "convolutional neural network" indicates that the network employs a mathematical operation called convolution which is a kind of linear operation.

3. Dataset

This project focuses on the MNIST (`http://yann.lecun.com/exdb/mnist/`) dataset throughout this exercise, which is widely used for introducing image classification problems. This database contains 60,000 training images and 10,000 testing images. The images of the digits contain grey levels represented by a 28 x 28 matrix resulting in 784 dimensional input vectors. The dataset has 1 channel and 10 classes: [0-9].

4. Evaluation

4.1. Feed-Forward Networks

4.1.1 Baseline

The baseline network consists in a simple architecture without hidden layers. Thus, in addition to the input, it will only have a fully connected layer which, in the Keras API, is called a **Dense** layer. Since the dataset poses a multi-class classification problem, the layer will have a number of neurons equal to the number of classes and we will use the **softmax** activation.

4.1.2 Multilayer model

The baseline network architecture does not contain hidden layers. In order to trying to increase the performance of the network the experimentation with 1, 2 and 3 hidden layers was done. All the layers added were of type **Dense** with **'**tanh**'** activation function and output size 32.

4.1.3 Regularization

We applied the following regularization methods on the multilayer model with one hidden layer: **L2 regularization**, **Dropout Regularization** and **L2 + Dropout Regularization**.

4.2. Convolutional Neural Networks

4.2.1 Baseline

The baseline network architecure contains a convolutional layer which, in the keras API is called a **Conv2D** layer, with **'relu'** activation followed by a **Max pooling** layer. Since the image needs to be flattened before passing it to the output we use a **Flatten** layer. Finally, as the dataset poses a multi-class classification problem, the output **Dense** layer will have a number of neurons equal to the number of classes and we will use the **softmax** activation.

4.2.2 Multilayer model

The baseline network architecture contains only one hidden layer. In order to trying to increase the performance of the network the experimentation with 2, 3 and 4 hidden layers by adding convolutional layers was done. All the layers added were **Conv2D** layers with **'relu'** activation, containing 4 Kernels with size 4

4.2.3 Regularization

This project tests three different regularization methods on the baseline model: **L2 regularization**, **Dropout Regularization** and **L2 + Dropout Regularization**.

5. Results and Discussion

5.1. Feed-forward

As the number of hidden layers increased the accuracy increased and the loss decreased capping at a maximum of 2 hidden layers as can be seen in table 3. At 3 hidden layers the model probably started to overfit and memorization started to occur therefore the accuracy decreased and the loss increased. In table 4 we can see the Dropout regularization technique achieves the best results in solving the overfitting.

5.2. CNN's

Similiar to feed-forward networks, increasing the number of hidden layers increased the accuracy and decreased the loss until reaching 4 hidden layers as can be seen in table 2, where memorization and overfitting started to occur. In table 1 we can conclude that the baseline model with only one hidden layer doesn't suffer from overfitting since the regularization techniques only augmented the loss and damaged the accuracy .

6. Conclusion

Overall by comparing the results of both networks is evident that the CNN's outperforms the feed-forward networks. The results of the feed-forwards networks improve with the increment of number of layers but regularization is required to reduce overfitting and achieve better results. Convolutional networks also perform better with multiple layers, however, no regularization is required since there is no overfitting.

References

[1] TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] GOODFELLOW, I., BENGIO, Y., AND COURVILLE, A. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

A Appendix

	Baseline	Dropout	L2	L2 + Dropout
Accuracy (%)	0.9817	0.9808	0.9730	0.9713
Loss (%)	0.0587	0.0629	0.1437	0.1498
Parameters (n^o)	31,642	31,642	31,642	31,642

Table 1: Regularization CNN

	Baseline	2 Hidden	3 Hidden	4 Hidden
Accuracy (%)	0.9817	0.9834	0.9842	0.9820
Loss (%)	0.0587	0.0525	0.0503	0.0542
Parameters (n^o)	31,642	12,234	9,946	12,778

Table 2: Multilayer CNN

	Baseline	1 Hidden	2 Hidden	3 Hidden
Accuracy (%)	0.9168	0.9171	0.9257	0.9252
Loss (%)	0.2987	0.2982	0.2641	0.2641
Parameters (n^o)	7,850	250,954	252,010	253,066

Table 3: Multilayer Feed-forward

	1 Hidden	Dropout	L2	L2 + Dropout
Accuracy (%)	0.9171	0.9253	0.9210	0.9207
Loss (%)	0.2982	0.2658	0.3226	0.3177
Parameters (n^o)	250,954	250,954	250,954	250,954

Table 4: Regularization Feed-forward