Assignment 1: Multi Layer Perceptron

February 21, 2023

0 INTRODUCTION

A grocery-robot has no knowledge of what kind of product an apple is. However, it can determine the features of a product using its sensors. Based on features such as the shape, color, weight, and wrapping of a product, it can recognize specific classes.

The problem can be simplified into **classification**, which can be performed by an Artificial Neural Network (ANN). The ANN could be able to recognize products by detecting a number of separate features:

- Roundness
- Color
- Weight
- etc ...

Based on the value of these features, the ANN could then categorize the product into various classes such as:

- Fruit
- Meat
- Candy
- etc ...

ASSIGNMENT

You and your team will have to code a fully functioning neural network. You will also have to write a report explaining what you did. Please include answers to all the questions from Section 1 in a **structured way** by numbering your answers. Where possible, include plots and graphics to support your answers and explain these properly.

You will create a network that can classify an input of **10 features** into one of **7 classes**. The data you will need to train and test your network can be found on Brightspace and on Vocareum. There are three files containing comma-separated values:

- `features.txt`: 7854 samples of 10 features
- `targets.txt`: 7854 target classes of these samples
- `unknown.txt`: 784 samples of 10 features, with no classes known

It's your job to load the data, divide it into training sets and use it to train a neural network. Finally, you should use your network to recognize the classes of the unknown samples. You cannot know how well your network performed on these samples until after the deadline. Your grade will be based partly on the performance of your network on this set.

You are expected to write the entire network from scratch: **a feed-forward neural network** with a **back-propagation training algorithm**. Write clear code, indent where needed and add comments to clarify what is happening. We have provided you with a Jupyter Notebook that can efficiently create and train neural networks. You will compare the results of this notebook with your own neural network. The following paragraphs contain instructions for the assignment, read them very carefully.

OUTPUT

The main method of your neural network should generate a list of predictions for the classes of unknown samples. The predictions should be outputted as comma-separated values to a file called `<group_number>_classes.txt` (where XX is your group number). It must be a **text file (.txt)** with all values in a single line. Example: `7,2,4,5,6,1,3,1, ...`.

Note that grading of this part of the assignment will be automated. This means that deviation from the required format will result in 0 points for this component.

DELIVERABLES

Create a report to answer the questions from Section 1. Besides this, you have to clean up your code and **upload its final working version to the course Vocareum** in the corresponding assignment, **including a Jupyter Notebook** which (1) attempts to train your perceptron on the XOR problem and (2) executes the entire training and prediction sequence. You should check if the notebook works as expected before pressing the `submit` button. Furthermore, we ask you to **deliver the following files via Vocareum**:

- a report in PDF format with answers to all questions. Your report **must be no longer than 7 pages** (including visuals) + the title page and the references. Please structure your report based on the numbered questions in this document (i.e. `Question 1.2.1` – `Answer 1.2.1` – `Question 1.2.2` – `Answer 1.2.2` ...):

  `"<group_number>_report.pdf"`

- a file with predictions for the unknown samples in format as described in section `Output`:

  `"<group_number>_classes.txt"`

Note 1: Your code will not be directly graded - do not rely on it to support your answers, but it will be checked for irregularities.

Note 2: Remember to keep your code clean, use proper indentation, provide useful comments, name your variables logically, etc. as the graders will inspect it.

To speed up the grading process, we want you to deliver your work exactly as outlined above. **If (a) your files are not of this format or (b) the main notebook does not run, your work will not be graded.**

You will use Python for this assignment. You are also allowed to use NumPy and pandas as linear algebra/data analysis libraries - these are the only libraries to develop the neural network that are allowed for this assignment. You may however use other libraries for tasks unrelated to the functionality of your neural network (for instance for plotting).

Fraud will not be tolerated. You are allowed to discuss concepts and ideas with colleagues from other groups, but you are not allowed to share code outside your own group. The same applies to submissions from previous years. You are highly encouraged to make use of the provided Gitlab repositories.

You may ask questions on Answers-EWI (while keeping in mind that it's a public forum, so please don't share partial solutions), or to a TA during the labs.

The deadline for submission is Monday **6th of March 2023 at 18:00**.

1 QUESTIONS

The following sections can help you through the development of the neural network. Each of the questions need to be answered explicitly in the report. It is very much recommended to take some time to think about the network architecture before you start development – the central part of this assignment is to code a fully working neural network with the back-propagation training algorithm (which should work for at least one hidden layer).

Hint: A good rule of thumb when reporting on your work is to:

1. Observe the patterns – e.g. describe what you noticed with regards to the behavior of your algorithm or results of your experiment, possibly referring to data in visual aids.
2. Explain your observations – e.g. state what you think impacts the particular behavior of your algorithm (data, hyper-parameters, ...) or influences your results.
3. Elaborate on your explanation – e.g. put it into a broader context, link to the knowledge from lectures, conduct further experiments to give evidence for your statements.

If you provide clear explanations with supporting evidence (also in form of meaningful visual aids discussed in the text), we will be able to see that you really mastered the material!

Note: Please make sure all the plots, diagrams and other images are of high quality. Furthermore, all plots should only present relevant information, have clear labels of the axes, be properly scaled, have titles, and are referenced in the text. Please avoid unnecessary graphs and make sure your report is well-formatted and without grammatical mistakes.

1.1 SET-UP

Working code is a **knock-out criterion**. If your code does not run, your submission will not be graded. Make sure to extend the main Jupyter Notebook so that when it is ran, it trains a neural network (with hyper-parameters of your choice), predicts the labels of unknown instances, and saves the predictions into a file. You will submit this notebook – along with the rest of your code and any data files – to the course Vocareum.

In case there are any problems running your code, your group will be contacted via email.

1.2 ARCHITECTURE

The basic element of a neural network is a perceptron. A perceptron is able to learn from its errors and improve after training.

1. (1 point) Develop (program) a single perceptron first. Show with three graphs (one per logic function) presenting the error over epochs that your perceptron is able to learn the `OR` and the `AND` functions but not the `XOR` function.

An ANN is much more powerful than a single perceptron. In the book and during the lectures the multi-layer network for an `XOR`-operator was introduced, which might help you to create your - more sophisticated - network. You will now design the topology of your network. For every question, briefly discuss how you arrived at your answers.

2. (0.5 points) How many input neurons are needed for this assignment?
3. (0.5 points) How many output neurons do you require?
4. (0.5 points) How many hidden neurons and layers will your network have? (Give an initial guess, later you will tune these values and analyze the network's performance).
5. (0.5 points) Which activation function(s) will you use?
6. (1 point) Give a schematic diagram of your complete network.

1.3 TRAINING

Now develop (program) your entire MLP network. Remember that the weight update rule for hidden neurons in an MLP is slightly different than it is for a perceptron. Also remember to take into account the derivative of your activation function. The network you've developed should now be trained to get accustomed to the type of data it will be processing. It's customary to divide data into a training, validation and test set.

7. (2 points) How do you divide your data to ensure unbiased estimate of performance?
8. (1 point) How do you evaluate the performance of your network? Justify your answer.
9. (2 points) When and why do you decide to end the training?
10. (2 points) Train your network 10 times, each with different initial weights. How does the initialization impact the performance? Plot your results.

1.4 Optimization

You will now use cross-validation to choose the 'optimal' number of hidden neurons. It is important to keep all other hyper-parameters fixed (learning rate, momentum etc.) to ensure that the results only portray the influence of the number of hidden neurons. Because the performance of a trained network is somewhat random due to initializations, perform each training 10 times and use the average as an indicator for performance.

11. (3 points) Train your network with different amounts of hidden neurons. At least 3 times chosen within the range of 7-30 hidden neurons. Generate a plot of the final performance versus the number of hidden neurons in the network. Explain what you observe and what might be the cause of these observations.

12. (2 points) Pick the architecture with the best results and plot its performance on the training set and the validation set during training, across epochs. Justify your choice.

1.5 Evaluation

It is now time to evaluate your network, and use visual aids to present its performance.

13. (2 points) What is the success rate of your network on the test set? How does it compare to the results of the validation set?

14. (2 points) Show and discuss a confusion matrix constructed on your test set. How should it be read? Where does your network make the most mistakes?

15. (10 points) Feed the unknown set (provided on Brightspace) to the network. Export the resulting classes as a comma-separated file exactly as in section 0.

1.6 Scikit-learn

The Python library `scikit-learn` provides algorithms for training neural networks, among other things. We made a Jupyter notebook which uses these features to create and train an ANN, similar to the one you just built. You can find the notebook on Brightspace.

A useful option in `scikit-learn` is grid search, which lets you exhaustively search a few points in the hyper-parameter space for optimized values. The notebook uses grid search to train and estimate the performance of a neural network for you.

16. (1 point) Download and open the Jupyter notebook. Tweak the settings of the grid search, and run all cells to find optimized parameters according to `scikit-learn`. Are the values of these parameters different from the ones you chose? What differences can you see in the training behaviour and performance of the two networks?

17. (2 points) Finally, take the optimal hyper-parameters found by `scikit-learn` and plug them into your own network. Does this give you better performance? Why do you think these parameters are better / worse?

1.7 Reflection

For this question you do not need to keep your specific assignment in mind, rather we ask you to reflect on Artificial Neural Networks and classification in general. A grocery-robot may have no knowledge of what kind of product an apple is and if it grabs a pear and classifies it as an apple, it would be unpractical, but no real harm is done. However, we use classification in a lot of other places too. Bias may carry some consequences for a certain group of individuals or introduce problems into society that are unwanted.

18. (2 points) Can you think of an example of misclassification and some potentially harmful consequences resulting from bias? Write a short paragraph (at most 5 sentences) explaining these. Example from the lecture slides is not accepted.

19. (2 points) Do you know of any ways to mitigate the harm done by unjust classification? Give two. Can you be sure that the problem is solved now?

1.8 Pen and paper

Finally, we have two questions for you to be solved on paper which serve as the preparation for the final exam. They are independent from the rest of the questions in this document. You are asked to submit only one set of answers but we strongly suggest that each team member solves them separately so that you all can practice with the calculations.

20. (3 points) When gathering data about the products, our robot may want to take a photo of the object and extract the features from the intensity of pixels in that photo. Imagine that the process works as follows. First, the robot uses `2 x 2` max pooling filter, reducing the image dimensionality to `4 x 4`. Then, on the transformed photo it uses the *sharpen* kernel (Figure 1.1) **without** padding. Finally, it adds the values per row (such that x_1 is the sum of the top row, x_2 is the sum of the bottom row), resulting in two numbers that will be used as input into a small neural network. Find the inputs for the photo shown in Figure 1.2 below. Present all steps of your work.

$$\begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$

Figure 1.1: *Sharpen kernel*

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 2 & 6 & 2 & 1 \\ 1 & 6 & 8 & 4 & 1 \\ 1 & 2 & 4 & 2 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Figure 1.2: *Photo taken by the robot*

21. (2 points) Use the inputs x_1 and x_2 found in the previous exercise and the network architecture presented in Figure 1.3 to find the output on neuron y. Assume that neurons z_1, z_2, and y use ReLU as the activation function. You can also assume that both bias terms – x_0 and z_0 – have value 0.

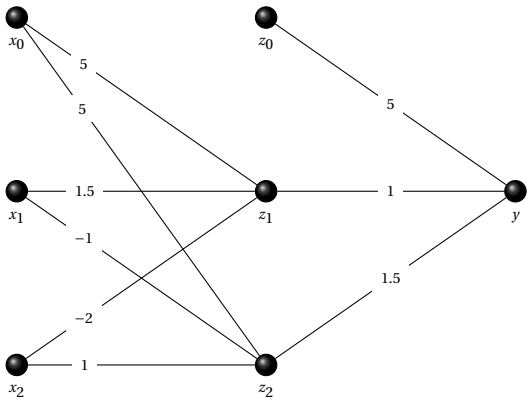

Figure 1.3: *A neural network used by the robot*