Department of Computer Sciences
University of Salzburg

PS Natural Computation
SS 16

Deep Learning Experiments with Classification Benchmarks

October 11, 2016

Project Members:
Hana Salihodzic, 1320206, Hana.Salihodzic@stud.sbg.ac.at
Pauline Trung, 1320212, Pauline.Trung@stud.sbg.ac.at
Amrit Bhogal, 1320038, Amrit_Pal_Singh.Bhogal@stud.sbg.ac.at
Armin Rekic, 1320215, Armin.Rekic@stud.sbg.ac.at
Moritz Quotschalla, 1321554, Moritz.Quotschalla@stud.sbg.ac.at

Academic Supervisor:
Helmut MAYER
helmut@cosy.sbg.ac.at

Correspondence to:
Universität Salzburg
Fachbereich Computerwissenschaften
Jakob–Haringer–Straße 2
A–5020 Salzburg
Austria

Contents

1 Introduction **2**

2 Artificial Neural Networks **2**

3 Deep Learning Basics **4**

4 Classification Benchmark **5**

5 Experiments **6**

6 Summary **7**

Abstract

In this project we test how well Artificial Neural Networks (ANNs) perform on classifying a heart diagnosis benchmark. Instead of using a conventional ANN we implemented a specific learning technique, Deep Learning, to see if this kind of training would improve the accuracy of determining whether a patient is sick or not. We implemented a variant of Deep Learning using auto-encoders. In our experiments we tested two different types of sequential auto-encoders: one with varying size of hidden layers and one with constant size of hidden layers. The experiments results show that we get a sligthly better accuracy than with conventional ANNs.

1 Introduction

Artificial intelligence and machine learning are getting increasingly more attention and relevance within the last few years. AlphaGo, a computer program developed by the artificial intelligence company DeepMind, beat Lee Sedol, one of the world's best Go player, in the chinese board game using Artificial Neural Networks. Also Google is exploring the world of machine learning with introducing its new Cloud Machine Learning which will probably push this technology forward. Within this work we want to find out how Artificial Neural Networks perform on classifying classification benchmarks. Therefore we implemented a special way of learning, Deep Learning. With the help of so-called auto-encoders we want to show if this technique could lead to possible improvements in classifying a benchmark for heart diagnosis.

2 Artificial Neural Networks

The concept of an ANN is inspired by the way biological nervous systems, such as the brain, work and process information. In a human body the information processing is done with the help of neural networks. A neural network is just a web of millions interconnected neurons which enable the parallel processing in a human body. A neuron is a special biological cell that processes information from one neuron to another neuron with the help of electrical and chemical changes. Knowing what a neuron is and how neural networks work it is easy to comprehend how artificial neural networks work.

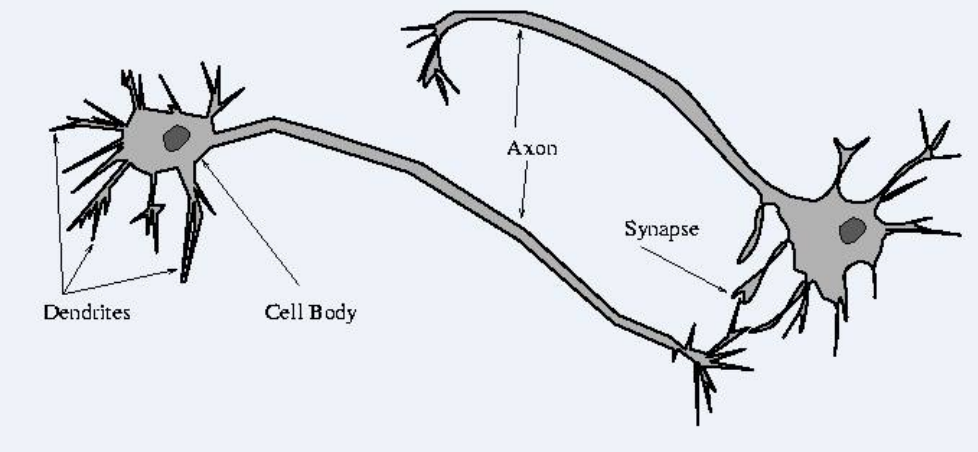

Figure 1: Two biological neurons [1].

An artificial neuron is basically just a mathematical model of a biological neuron. Artificial neurons basically consist of inputs, which are multiplied by weights, and then computed by a function which determines the activation of the neuron. The higher the weight of a connection between two artificial neurons is, the stronger the input which is multiplied by it will be. Weights can also be negative which has an inhibiting influence on the neuron. By adjusting the weights of an artificial neuron one can obtain the output for a specific input. There are many algorithms

[1] "Simplified biological neurons" by Albrecht Schmidt. 1996. Retrieved from: https://www.teco.edu/~albrecht/neuro/html/img1.gif

with which weights can be adjusted in order to obtain the desired output from the network. This process of adjusting the weights is called learning or training. There are many different training methods for ANNs but the most successful ones are based on back-propagation. An ANN is simply a net with many artificial neurons which are usually combined and organized in three different types of layers: the input layer, the hidden layer, and the output layer.

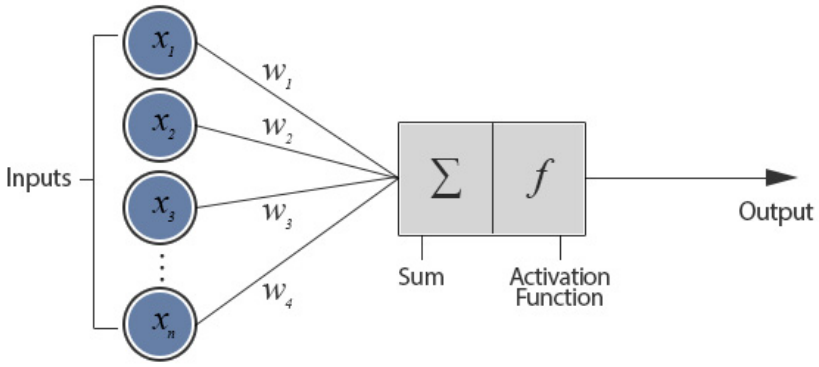

Figure 2: Neuron model [2].

The neurons in the input layer receive the data and transfer them to neurons in the first hidden layer through the weighted links. Data are processed and the result is transferred to the neurons in the next layer. The network's output is provided by the neurons in the last layer. The ith neuron in a hidden layer processes the incoming data.

$$input_{neuron} = \sum_j a_j w_{ij} = I_i$$

It is necessary to design a customized ANN for every purpose to guarantee proper functionality. An ANN is typically defined by three types of parameters:

1. The number of layers and the number of nodes in each of the layers.
2. The algorithm applied for updating the weights of the connections.
3. The activation functions used in various layers.

There are two ways for passing the signals from neurons. Recurrent networks allow connections between neurons to form cycles while feed-forward nets only have connections in one direction.

Choosing the right number of hidden neurons affects how well the network is able to separate the data. A large number of hidden neurons will ensure correct learning, and the network is able to correctly predict the data it has been trained on, but its performance on new data, its ability to generalize, may be compromised. With too few hidden neurons, the network may be unable to learn the relationships in the data and the error will fail to fall below an acceptable level. Thus, selection of the

[2] "Modeling Artificial Neurons" by Lee Jacobson. 2013. Retrieved from: http://www.theprojectspot.com/images/post-assets/an.jpg

[3] adapted from: https://upload.wikimedia.org/wikipedia/commons/thumb/3/3d/Neural_network.svg/2000px-Neural_network.svg.png

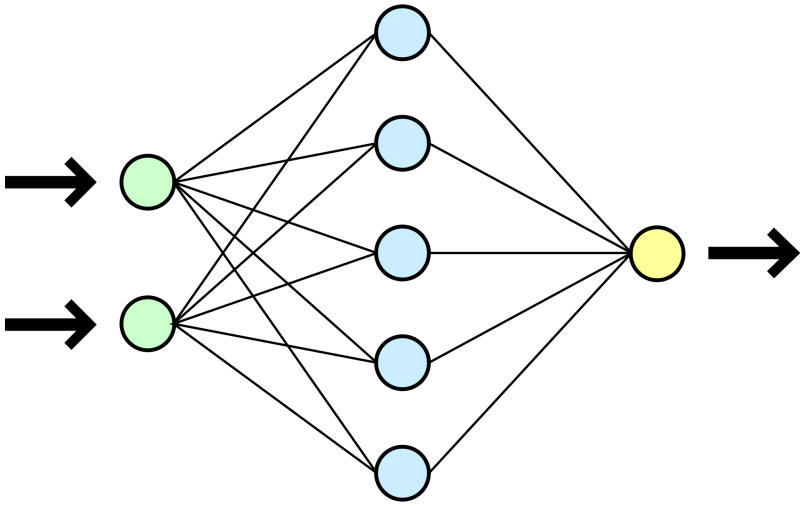

Figure 3: Input layer (left), hidden layer (middle), and output layer (right) of an ANN [3].

number of hidden neurons is a crucial decision. In most cases one hidden layer is completely sufficient. If accuracy is the main criterion for designing a neural network then hidden layers can be increased. There are some known rules-of-thumb used to determine the number of neurons in the hidden layer: [1]

- $\frac{2}{3}$ of the size of the input layer
- Less than twice of the number of neurons in input layer
- A number between the input layer size and the output layer size

3 Deep Learning Basics

Deep Learning is a powerful set of techniques for learning in ANNs. The basic principle is the transformation of the input to make the neural network learn much more easier. The ANN itself searches for a way it is learning the best.

There are two main methods for learning - supervised and unsupervised. The training data consist of a set of training examples. In supervised learning, each example is a pair consisting of an input object and the desired output value. The main difference to unsupervised learning is the task of inferring a function to describe hidden structure from unlabeled data. Since the examples given to the learner are labeled, there is no error or reward signal to evaluate a potential solution.

In our case we chose auto-encoders as technique for implementing Deep Learning. An auto-encoder is an ANN with a single hidden layer which is trained such that the input equals the output, so the input and output layers have the same number of neurons. The deviation of the output from the input is called reconstruction error. After the training the output of the hidden layer is obtained and it is used as an input for the next auto-encoder. This scheme is applied to every auto-encoder, which is connected in a series of auto-encoders. By eliminating redundant information (compression of data) or adding so called sparse data we may get better results than without Deep Learning [2]. Looking at the output of each layer (when training multiple auto-encoders in a row, i.e., the output of one auto-encoder is the input of the next auto-encoder) we get different encodings of the input which can be used for classification in a second ANN without hidden layer.

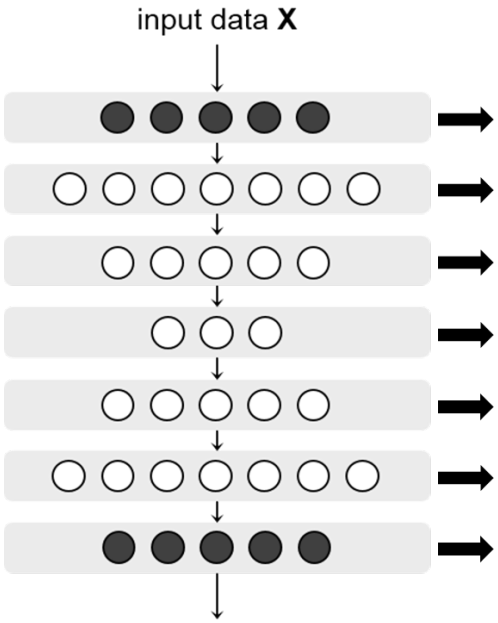

Figure 4: Getting Data from sequential auto-encoders with varying layer size [4].

In our experiment we tested two types of auto-encoders:

1. Auto-encoder with constant size of hidden layer's outputs/neurons (used as input for the next auto-encoder). We call it constant size.
2. Auto-encoder with various size of hidden layer's outputs/neurons (used as input for the next auto-encoder). To vary the neuron size we adjusted the hidden layer of the network. Firstly we varied the size of neurons decreasing with every auto-encoder in the series. Secondly we applied the same concept increasing. We call it various size.

4 Classification Benchmark

To test how the trained network performs we need some sample data we can use as input. We decided to choose a classification benchmark where our network should learn to classify data sets into a certain class. In order to demonstrate possible better results of the classification accuracy with our implementation a benchmark which has some room for improvement was necessary.

For our experiments we picked the heart data set out of the PROBEN1 benchmark collection. It is used for prediction of heart disease presence by classifying each data instance into two classes (sick or healthy). Each of the 303 instances has 35 different attributes thus our ANN has the exact same number of input neurons. While the most attributes stand for certain data of one patient (e.g. age, sex, heart rate) a few values just show whether a certain attribute is missing or present. On the other end we have two output neurons determining whether the patient is healthy or not. For a better reproducibility the data is already divided into training data, validation data, and test data [3].

[4] adapted from: http://stackoverflow.com/questions/20818774/implementation-of-autoencoder

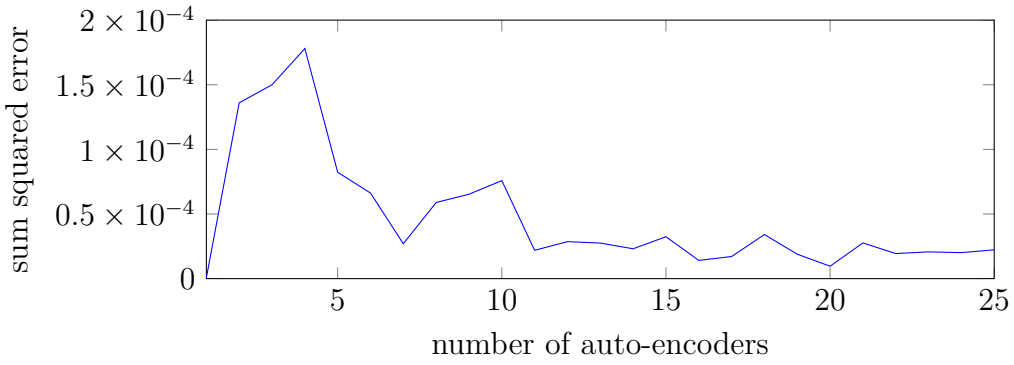

Figure 5: Reconstruction error for varying number of sequential auto-encoders

Previous experiments using this benchmark reached a classification accuracy of 77%.

5 Experiments

In the experiments we wanted to figure out, if we can achieve a high classification accuracy, when the network is able to decide how to encode the data by itself. Therefore we used an auto-encoder, which is implemented using the BOONE framework. BOONE is a JAVA framework, which provides tools for developing Artificial Neural Network applications [4].

For the classification we had to utilize another network. The input of that network was the coded data from the last auto-encoder in the series. That network was build up like the network of auto-encoders. So, it was also a fully connected feedforward network for the emerging layers. However, the network did not consist any hidden layers, because the work of those layers was already done by the auto-encoders earlier.

We used the data sets according to the heart benchmark. The benchmarks consists of training data to train, validation data to verify and test data to test the network.

To obtain the results depending on the reconstruction error we output the SSM (sum squared error) during the training of the network. In our case even the maximal reconstruction error was minimal. As a result, we did not consider it more closely.

As you can see we get the best result (84%) with eight hidden layers, with more layers there seems to be no improvement possible. In comparison to a conventional ANN without auto-encoders (ca. 80%) we get minor improvements.

With decreasing and increasing hidden layer size we get slightly worse results where the best ones are at the decreasing part.

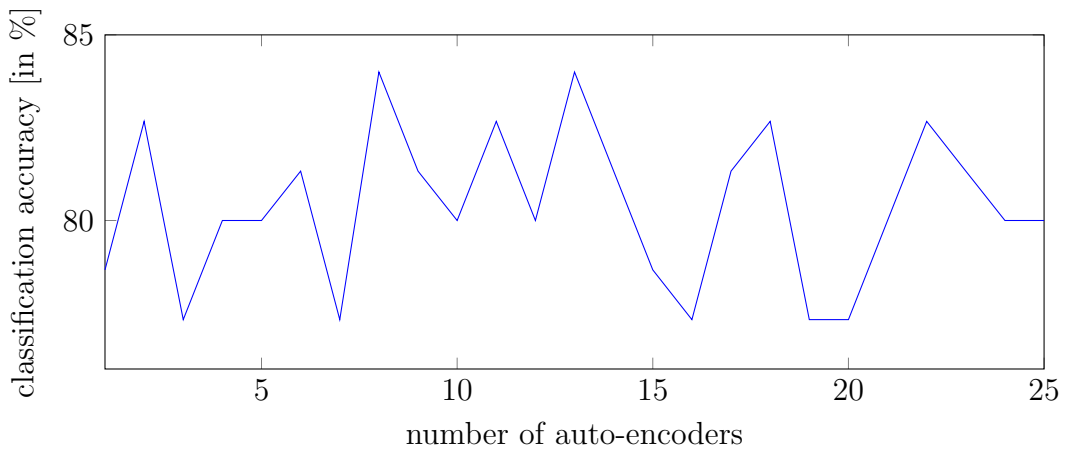

Figure 6: Classification accuracy for varying number of sequential auto-encoders (constant size)

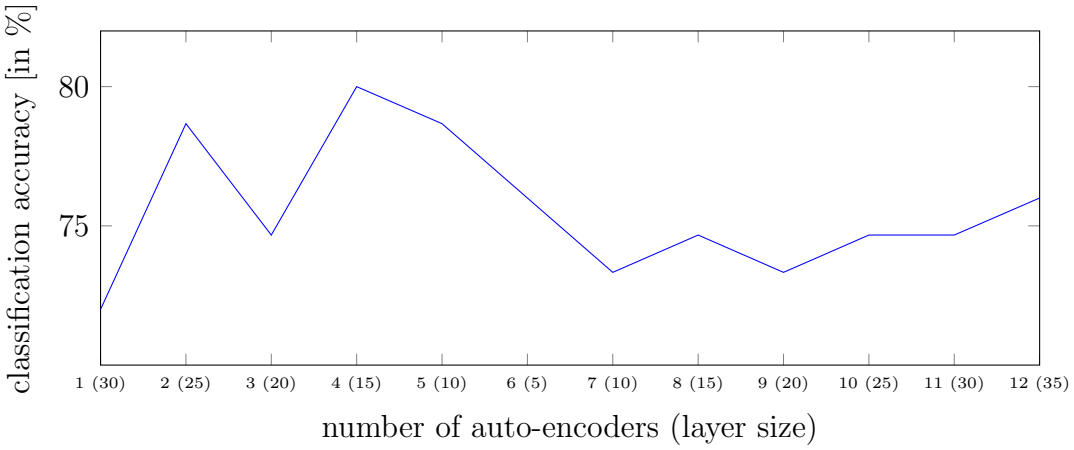

Figure 7: Classification accuracy for varying number of sequential auto-encoders (alternating size)

6 Summary

The results showed us that the Deep Learning technique with auto-encoders has potential in classifying benchmarks. With processing the data through sequential auto-encoders (in our case 8 auto-encoders was the best set up) we got slightly better results than with using a conventional ANN without auto-encoders and Deep Learning. But of course there is much more room for improvements and extensions. For example modifying and testing with other parameters may have more impact on increasing the accuracy. In future experiments it would be interesting to see, how the ANN performs in comparison with other benchmarks.

Links

- Project Page: `http://student.cosy.sbg.ac.at/~mquotsch/`
- PS Page: `http://www.cosy.sbg.ac.at/~helmut/Teaching/NaturalComputation/`

References

[1] Stergiou, C., Siganos, D.: Neural Networks. SURPRISE 96 Journal. 1996.

[2] Schmidhuber, J.: Deep Learning in Neural Networks: An Overview 2014.

[3] Prechelt, L.: PROBEN1 - A Set Of Neural Network Benchmark Problems and Benchmarking Rules. Technical Report 21/94. Universität Karlsruhe. 1994.

[4] Mayer., A., Mayer., M.: Boone: Basic Object-Oriented Neural Environment. 2003.