WPF Verteilte Systeme

A distributed Architecture for training a Deep Learning Agent

Jonas Faßbender

Amjad ...

Hamza ...

David ...

Contents

List of Figures

1 Introduction

This document records our attempt of building a distributed application for training a Deep Learing Agent to master a specific environment.

It should be noted that the focus of this project lies on "distributed", which means the main goal of this application is speeding up a task which requires lots of computing power using technologies and algorithms for concurrent computing in order to horizontally scale up and reduce the overall time the task takes for completing rather than an attempt to find a optimized algorithm that reduces time complexity of training a Deep Learning Agent.

We will begin with introducing the technologies we used, starting with a brief introduction of Neural Networks followed by a summary of the Python programming language we have used for building our application. After that we will outline the frameworks and libraries we added in order to build our Machine Learning Infrastructure, before finally presenting the concept of a Message Broker, which is one crucial part of our application's architecture.

After that we will present the architecture and the concepts of our application with emphasis on how we achieve concurrency on different levels. This is followed by a chapter which outlines our test results. The document is concluded by giving an overwiew over improvements and optimizations which can further increase the results of our application.

The whole project can be found at:

https://github.com/jofas/wpfvs

2 Technologies

2.1 Neural Nets

2.1.1 Derivation from biology

"Neural networks are an attempt to model the insights gained in brain research on the interaction between nerve cells (neurons) and connections (synapses)".[**?**]

Here, an artificial neuron mimics the functioning of a nerve cell . A biological nerve cell is connected to other nerve cells via synapses. At these connections, stimuli are transmitted by means of chemical messengers (neurotransmitters) and converted into an electrical signal within the nerve cell.

The synapses of a nerve cell are located at so-called dendrites (branches), from which the transmitted stimuli are transmitted to the soma (cell body). It does matter how far the synapse is from the soma. The closer a synapse is to the soma, the stronger the stimulus transmission.[**? ?**]

In addition, it should be the case that multiple synapses stimuli at the same time which adds incoming stimuli within the nerve cell.[**?**] If the stimulus transmitted exceeds a threshold within the cell, then an action potential is triggered and the cell forwards the signal to other cells, which in turn are connected via synapses to the stimulating transmitting cell. Nerve cells form associations in the human brain, which develop the ability to recognize complex structures. Artificial neural networks are an attempt to imitate these associations of nerve cells.[**?**]

2.1.2 Structure of an artificial neuron

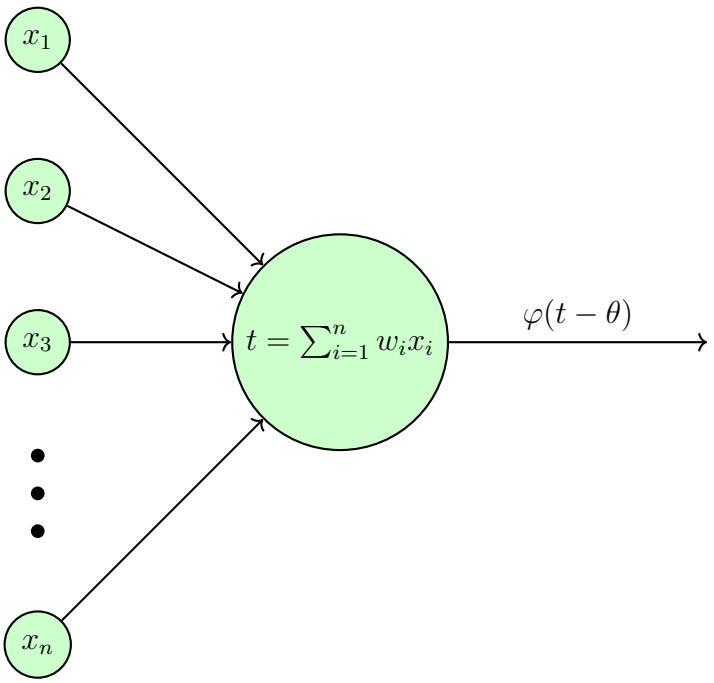

Figure 1: Scheme of an artificial neuron

From chapter 2.1.1, the components and functioning of an artificial neuron can be derived.

In an artificial neuron the following happens:

1. An artificial neuron is connected to other neurons or the direct value input via input connections, which are supposed to represent the synapses of the nerve cell (described in Figure 1 with x_i). These inputs can be discreet or continuous.[**?**]

   If the data comes from other neurons, the weighting of the connection is additionally transferred to the value ω_i . Each connection of a neuron has an individual weighting. The greater the value of this weighting, the more important the transmitted value x_i for the network output.

2. The value and weight of each input connection is added by the transfer function $\sum_{i=1}^{n} w_i x_i$ to calculate the net input value t.

3. Every artificial neuron, like a nerve cell, has a threshold. An artificial neuron is a value θ. The threshold θ is subtracted from the net input value t to determine the activation potential of the neuron.

   The threshold can also be represented by the bias. The bias is a further input, which transmits the constant value 1 and as weighting the value $b = -\theta$.

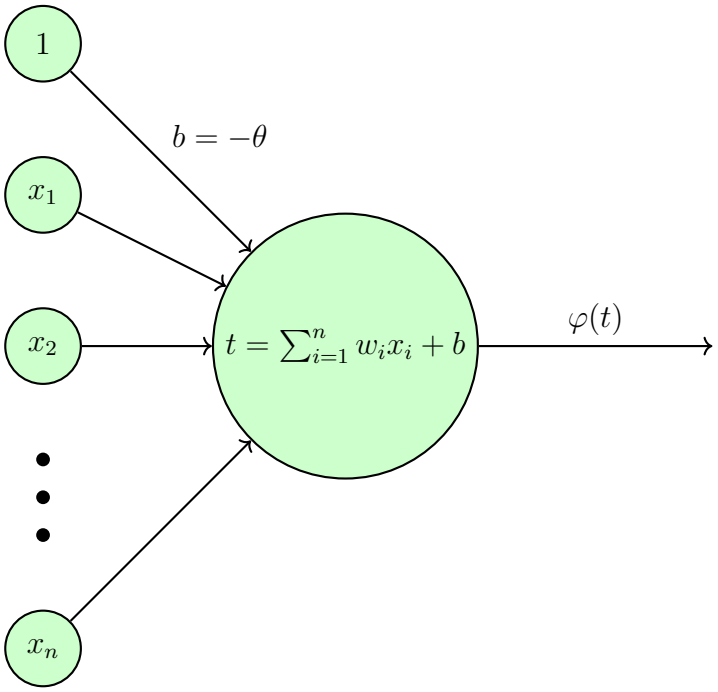

Figure 2: An artificial neuron with bias

4. The determined value of $t - \theta$ is used as an input in the activation function to determine the output of the neuron.

   From this process, the basic elements of a neuron can be determined:

   - Weighting
   - Threshold
   - Transfer function
   - Activation function

2.1.3 Construction of a neural network

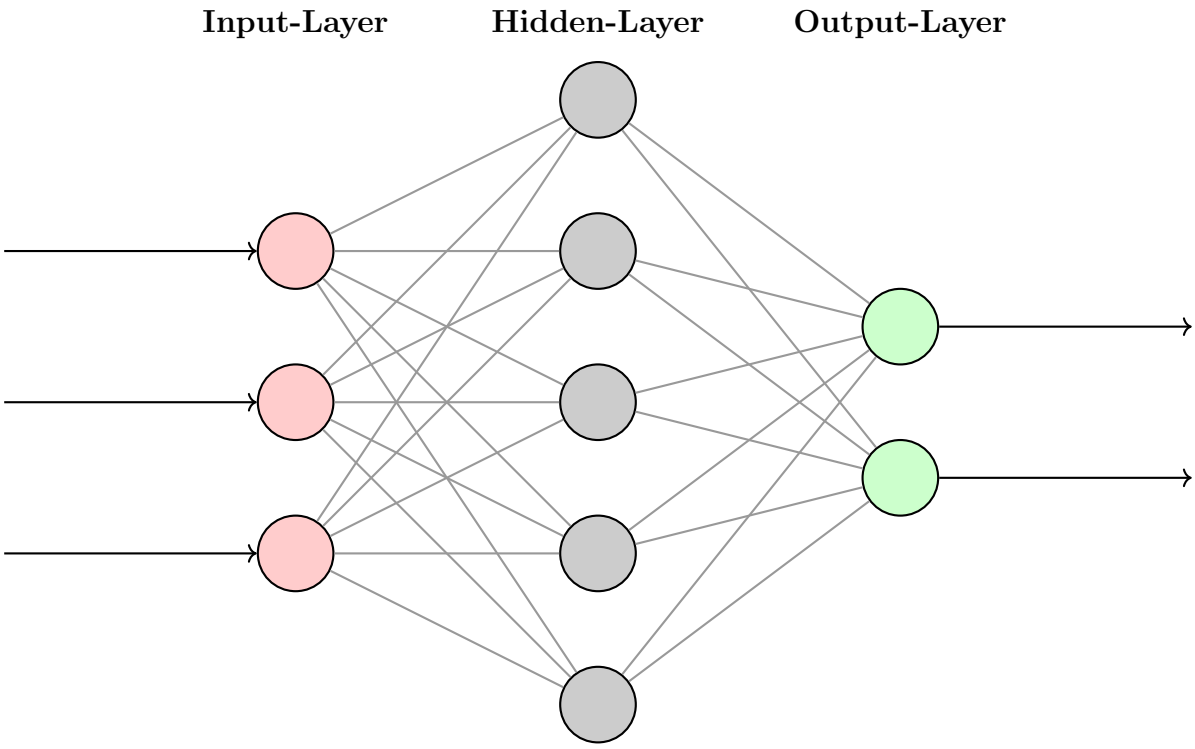

Figure 3: Neural network with hidden layer

2.2 Python

...

2.3 Tensorflow

2.4 Gym

2.5 Message Broker (RabbitMQ)

3 Application

intro here

3.1 Architecture

intro here

3.1.1 Executioner

3.1.2 Worker

here comes the data sanitation part

3.1.3 Network

3.1.4 Queues

3.2 Results

3.3 Where to go

(dynamic data generation, loadbalancing, further optimizations like thread pool exec in worker (not spawning so often), destroying the bottleneck in executioner (data parsing))

hallo [**?** **?**]

References

[1] Erdmann, A., Erdmann, U., and Martens, A. Grüne Reihe. Materialien für den Sekundarbereich II. Schroedel Verlag GmbH, Braunschweig, 2004.

[2] Rey, G. D., and Beck, F. Neuronale Netze - Eine Einführung. http://www.neuronalesnetz.de/downloads/neuronalesnetz_de.pdf. 17.07.2018.

[3] Trinkwalder, A. Netzgespinste. c't 6 (2016), 130–135.