

Neuronale Netze

Projektbericht

Alexandra Zarkh, Sui Yin Zhang,
Lennart Leggewie, Alexander Schallenberg

Hochschule Bonn-Rhein-Sieg

9. November 2021

Inhaltsverzeichnis

1	Einleitung	3
1.1	Aufgabenstellung & Zielsetzung	3
1.2	Aufgabenkontext & externe Vorgaben	3
1.3	Nicht vorgegeben aber notwendigerweise von uns festgelegt	3
1.4	Literaturarbeit: Verweis auf Vorarbeiten	3
2	Methoden	4
2.1	Util	4
2.2	Klassenstruktur Network & Neuron	4
2.3	Konstruktoren & Initialisierung	4
2.4	Kalkulation	4
2.4.1	Forward Propagation	4
2.5	Training	4
2.5.1	Backpropagation	4
2.6	Speichern & Laden	4
2.7	toString	4
3	Ergebnisse	5
3.1	Begründung der Korrektheit der Umsetzung	5
3.2	Performance-Überlegungen	5
4	Diskussion & Fazit	6
4.1	Vor- & Nachteile der gewählten Umsetzung	6
4.2	Was fehlt ? Was könnte erweiternd gemacht werden?	6
5	Verwendete Literatur & Anhang	7

1 Einleitung

1.1 Aufgabenstellung & Zielsetzung

Die Aufgabenstellung lautete zunächst, die Grundlagen eines künstlichen neuronalen Netzes (knN) in Java zu implementieren, sodass mit diesem schon zu einfachen Eingaben eine korrekte Ausgabe kalkuliert wird (Forward Propagation). Diese weiteten sich darauf aus, das Netz trainieren zu können (Backpropagation) und die trainierten Einstellungen des Netzes zu speichern und zu laden.

Das Ziel war, die erste Hälfte der Aufgabenstellung in den ersten zwei Wochen und die zweite Hälfte in den folgenden zwei Wochen umzusetzen.

1.2 Aufgabenkontext & externe Vorgaben

Vorgegeben war, ein knN erstellen zu können, dem man bei seiner Erstellung Gewichtungen sowie Biases übergeben kann. Außerdem soll das Netz Ausgaben abhängig von den Eingaben berechnen können und die Gewichtungen und Biases sollen trainiert werden können. Außerdem ist ein Format zur Abspeicherung der Gewichte und Biases vorgegeben worden, welche in diesem Format in eine CSV-Datei gespeichert werden soll.

1.3 Nicht vorgegeben aber notwendigerweise von uns festgelegt

Nicht vorgegeben, aber notwendigerweise festgelegt wurde, dass beim Erstellen eines knN die Anzahl der Neuronen für jede Neuronenschicht vom Benutzer festgelegt wird. Implizit wird damit auch die Anzahl der versteckten Schichten verlangt. Außerdem ist die Struktur des Netzes frei gewählt.

1.4 Literaturarbeit: Verweis auf Vorarbeiten

- An Introduction to Neural Networks, Kroese, B., a Van der Smagt, P., 1996
- Neural Networks, 3Blue1Brown, 2018, YouTube

2 Methoden

2.1 Util

Die Klasse *Util* bietet die abstrakten Hilfsmethoden *random(int)*, *addToVec1(double[], double[])* und *mulToVec(double, double[])* für das kreieren und für den Umgang mit Vektoren bzw. Arrays. Da diese Methoden mit ausreichend JavaDoc ausgestattet sind, wodurch sie selbsterklärend sind, wird auf diese hier nicht näher eingegangen.

2.2 Klassenstruktur Network & Neuron

2.3 Konstruktoren & Initialisierung

2.4 Kalkulation

2.4.1 Forward Propagation

2.5 Training

Die Methode *train(double[[[]], double, double[[[]], int)* in der Klasse *Network* bietet dem Benutzer die Möglichkeit das knN bzw. dessen Gewichte und Biases zu trainieren, so dass das Netz nach ausreichend Training immer das gewünschte Ergebnis errechnet. Dazu nimmt die Methode mehrere Eingabevektoren und passende Zielausgabevektoren, sowie eine Lernrate sowie die Anzahl der Wiederholungen des Trainingsprozesses benötigt. Auf die Eingabevektoren (EV) wird erst die Forward Propagation angewendet um die Ergebnisse der Neuronen zu erhalten, welche dann für den eigentlichen Lernprozess durch die Backpropagation benutzt werden. Die Kosten des Durchlaufs werden für eine Trainingsdurchlauf summiert und danach zur Trainingskontrolle ausgegeben. Die Methode *cost(double[], double[])* errechnet diese durch

$$C_0 = \sum_{i=0}^n (o_i - y_i)^2 \quad (1)$$

mit o und y = tatsächliche und gewünschte Ergebnisse und n = Anzahl der Ergebnisse in o .

2.5.1 Backpropagation

Die Umsetzung der Backpropagation ist an die Mathematik von 3Blue1Brown angelehnt.

2.6 Speichern & Laden

2.7 toString

»

3 Ergebnisse

3.1 Begründung der Korrektheit der Umsetzung

3.2 Performance-Überlegungen

4 Diskussion & Fazit

4.1 Vor- & Nachteile der gewählten Umsetzung

4.2 Was fehlt ? Was könnte erweiternd gemacht werden?

5 Verwendete Literatur & Anhang