Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Proseminar Softwareentwicklung in der Wissenschaft

Grundlagen Neuronaler Netze

vorgelegt von

Jonah Lüdemann

Fakultät für Mathematik, Informatik und Naturwissenschaften
Fachbereich Informatik
Arbeitsbereich Wissenschaftliches Rechnen

Studiengang: Informatik
Matrikelnummer: 7311749

Betreuer: Jannek Squar

Wedel, 30.08.2020

Contents

1 Aufbau eines neuronalen Netz

Der Aufbau ein künstlichen neuronales Netz ist dem eines biologischen nachempfunden, indem künstliche Neuronen miteinander vernetzt sind. Die Darstellung eines neuronalen Netzes entspricht der eines Graphen, wobei die Neuronen den Knoten und die Verbindungen den Kanten in einem Graphen entsprechen (siehe Figure 1.1). Je nach Topologie kann ein Netz in bis zu drei verschieden Schichten aufgebaut sein, die Eingabeschicht, beliebig viel Versteckte Schichten und die Ausgabeschicht (siehe Section 1.3).

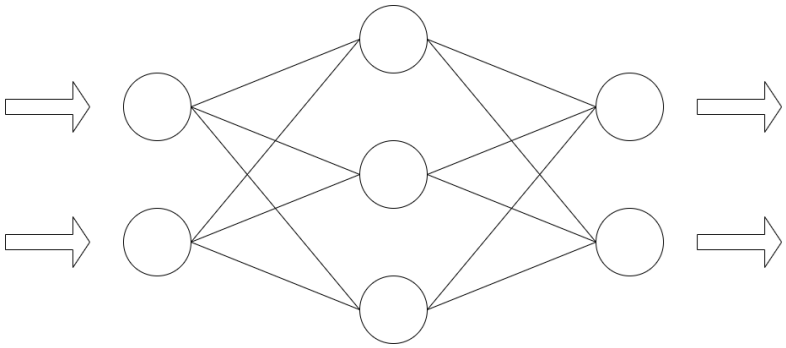

Figure 1.1: Einfaches neuronales Netz

1.1 Neuronen

Die Neuronen in einem neuronalen Netz bestehen aus drei Funktionen, die nacheinander ausgeführt werden. Zuerst nimmt die Propagierungsfunktion alle Eingaben, die ein Neuron als Ausgabe von anderen Neuronen erhält und verarbeitet diese auf die in der Propagierungsfunktion festgelegten Weise. Danach wird der durch die Propagierungsfunktion ermittelt Wert in der Aktivierungsfunktion verarbeitet (siehe Section 1.1.1). Zum Schluss wird dann von Ausgabefunktion das Ergebnis der Aktivierungsfunktion an alle anliegenden Verbindungen ausgegeben.

1.1.1 Aktiverungsfunktionen

Die Aktivierungsfunktion bestimmt, welchen Zustand ein Neuron bei einer gegebenen Eingabe annimmt. Um alle möglichen Aufgaben bei neuronalen Netzen zu erfüllen, gibt es viele verschiedene Aktiverungsfunktionen, die auf Eingaben anders reagieren. Hier beschränke ich mich auf drei Beispiele:

1. Die Schwellenwert- oder Heaviside-Funktion. Eine einfache Funktion, die an das biologische Original angelehnt ist (siehe Figure 1.2)

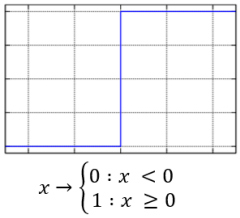

Figure 1.2: Schwellenwertfunktion

2. Die Sigmoidfunktion. Die Steigung der Funktion ist variierbar und sie ist für die Backpropagation geeignet (siehe Figure 1.3)

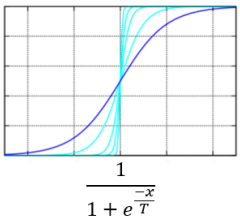

Figure 1.3: Sigmoidfunktion

3. Die Rectifier Funktion. Die Ausgabe beschränkt sich auf positive Zahlen und Null (siehe Figure 1.4)

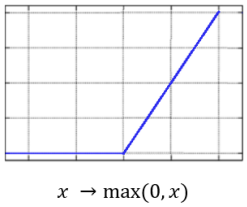

Figure 1.4: Rectifier Funktion

1.1.2 Bias-Neuron

Das Bias-Neuron ist eine Sonderform der Neuronen. Im Gegensatz zu normalen Neuronen besitzt das Bias-Neuron keine Propagierungs- oder Aktivierungsfunktion, sondern es gibt immer den Wert 1 aus. Das Bias-Neuron wird dazu verwendet, den Schwellwert einer Aktivierungsfunktion zu verändern, ohne die Aktivierungsfunktion zu verändern. Die tatsächliche Stärke des Signals eines Bias-Neurons lässt sich durch die Gewichtung der Verbindungen bestimmen.

1.2 Verbindungen

Eine Verbindung zwischen zwei Neuronen leiten die Ausgaben der Ausgabefunktionen des einen Neurons an das andere weiter. Zusätzlich hat jede Verbindung noch eine eigene Gewichtung zwischen -1 und 1. Diese Gewichtung wird in der Propagierungsfunktion des empfangenden Neurons mit der Ausgabe des sendenden Neurons multipliziert. Eine Verbindung mit dem Gewicht 0 ist eine ungenutzte Verbindung.

1.3 Topologien

1.3.1 Feed-Forward-Netz

Ein Feed-Forward-Netz besteht aus drei verschiedenen Schichten, bei denen die Verbindungen immer nur von einer höheren in eine tiefere Schicht verlaufen können, meistens nur in die direkt darunter liegende Schicht:

- Die Eingabeschicht. Sie liegt am Anfang des Neuronalen Netzes und nimmt die Eingaben in das Netz entgegen. Die Eingabeschicht kann auch gleichzeitig die Ausgabeschicht sein.
- Beliebig viele versteckte Schichten zwischen der Eingabe- und der Ausgabeschicht. Mit diesen Schichten kann von außerhalb des Netzes nicht direkt interagiert werden.
- Die Ausgabeschicht. Sie liegt am Ende des Neuronalen Netzes und gibt die vom Netz verarbeiteten Daten aus. Die Ausgabeschicht kann auch gleichzeitig die Eingabeschicht sein.

1.3.2 Rückgekoppeltes Netz

In einem rückgekoppelte Netz sind im Gegensatz zum Feed-Forward-Netz keine festgelegten Schichten vorgesehen. Stattdessen kann jedes Neuron mit jedem Neuron des Netzes verbunden sein und sich so mit seiner eigenen Ausgabe selber beeinflussen. Dies kann entweder durch direkte Rückkopplung, also eine Verbindung von der Ausgabe eines Neurons zurück zur Eingabe, oder durch indirekte Rückkopplung über andere Neuronen vorkommen. Durch den schichtenlosen Aufbau einen rückgekoppelten Netzes gibt es in der Regel keine festgelegten Neuronen, die für die Ein- oder Ausgabe zuständig sind.

1.3.3 Vollständig verbundenes Netz

In einem vollständig verbundenen Netz gibt es wie im rückgekoppelten Netz keine festgelegten Schichten. Das vollständig verbundene Netz zeichnet sich dadurch aus, dass jedes Neuron des Netzes mit jedem anderen Neuron des Netzes verbunden ist, es gibt aber im Normalfall keine direkte Rückkopplung. Ebenso wie im rückgekoppelten Netz gibt es im vollständig verbundenen Netz keine festgelegten Ein- und Ausgabeneuronen.

2 Lernprozess

Ein neuronales Netz ist zu Beginn noch nicht auf die erwartete Aufgabe angepasst. Es muss erst mit Trainingsbeispielen an die Aufgabe angepasst werden. Beispielsweise müssten dafür einem Netz, dass handschriftlich geschrieben Zahlen erkennen soll, viel Bilder von ebenfalls handschriftlich geschrieben Zahlen vorgegeben werden. Dabei lässt sich das Lernen in einem neuronalen Netz durch verschiedene Änderungen am Netz realisieren. Diese Änderungen lassen sich grob in drei Kategorien einteilen:

1. Änderung der Verbindungsgewichte. Bei der Änderung der Verbindungsgewichte kann man eine Vielzahl an Veränderungen an einem Netz vornehmen. Man kann neue Verbindungen schaffen und unnötige Verbindungen löschen, da theoretisch jedes Neuron mit jedem verbunden ist, aber ein Verbindungsgewicht von 0 eine Verbindung abschaltet und ein Verbindungsgewicht ungleich 0 eine Verbindung aktiviert. Auch lassen sich durch Bias-Neuronen erzeugte Schwellenwerte für Aktivierungsfunktionen anpassen, da diese auch durch Verbindungsgewichte bestimmt werden.

2. Hinzufügen oder Entfernen von Neuronen. Dies ist ein aufwändigerer Prozess als die Änderung von Verbindungsgewichten und kann nur in einer neuen Generation eines neuronalen Netzes durchgeführt werden.

3. Änderung von Propagierungs-, Aktivierungs-, und/oder Ausgabefunktion. Dies ist eine aufwändige und eher unübliche Art ein Netz lernen zu lassen, da dies auch eher schwierig umzusetzen ist.

2.1 Supervised und Unsupervised Learning

Der Lernvorgang in einem neuronalen Netz kann entweder supervised oder unsupervised stattfinden. Der Unterschied zwischen den beiden Vorgehensweisen liegt darin, ob dem Netz zu einem Beispiel noch die erwartete Ausgabe dazugegeben wird. Beim supervised Learning wird dem Netz die erwartete Ausgabe nachdem das Netz seine eigene Ausgabe mitgeteilt. Diese Information wird dann im neuronalen Netz dazu genutzt, um mittels Backpropagation (siehe Section 2.2) die bestmöglichen Änderungen an allen Verbindungen zu berechnen.

Dabei gibt es des weiteren noch die Einteilung in online und offline Lernen beim supervised Learning. Beim online Lernen wird nach jedem Beispiel die Änderung im Netz berechnet und angewendet. Beim offline Lernen wird hingegen erst eine festgelegte Menge an Beispielen zur Berechnung der Änderung genutzt.

Das unsupervised Learning ist dagegen am natürlichen Vorbild orientiert. Hier erhält das neuronale Netz kein Feedback, wie die richtige Ausgabe für ein gegebenes Beispiel sein soll. Dadurch muss das neuronale Netz eigene Muster in den gegebenen Daten finden und diese dann entsprechend ordnen.

2.2 Backpropagation

Unter Backpropagation versteht man das Verfahren, mit dem in einem neuronalen Netz berechnet wird, wie alle Verbindungsgewichte geändert werden müssen, um näher zu den erwünschten Ausgaben zu gelangen. In diesem Verfahren wird mittels Gradientenabstieg (siehe Section 2.2.1) für jedes Neuron berechnet, wie stark jedes einzelne Verbindungsgewicht geändert werden soll.

Dazu beginnt man mit den Neuronen der letzten Schicht und vergleicht den erwarteten Ausgabewert mit dem erhaltenen Wert Aus diesen beiden Werten wird dann die Verlustfunktion (siehe Section 2.2.1) gebildet. Danach wird für jedes Neuron der davor liegenden Schicht der Durchschnitt aller erwarteten Änderungen mitgeteilt und dort als erwarteter Ausgabewert verwendet. Dieses Verfahren wird solange rekursiv immer mit der nächst höheren Schicht durchgeführt, bis alle Schichten erreicht wurden.

2.2.1 Verlustfunktion und Gradientenabstieg

Die Verlustfunktion mit dem Ausgabewert a_i und dem erwarteten Wert e_i lautet

$$\sum_{i}^{n}(a_i - e_i)^2$$

Je näher das Ergebnis dieser Summe an 0 ist, desto genauer entsprechen die Ausgabewerte den erwarteten Werten.

Um einen guten Zustand für das neuronale Netz zu finden muss man in der mathematischen Funktion des neuronalen Netzes einen Tiefpunkt finden. Dazu nutzt man das Verfahren des Gradientenabstiegs. Für den Gradientenabstieg wird der über die Verlustfunktion errechnete Wert als Gradient genutzt. Der Gradient ist für jeden Punkt einer Funktion als Vektor g in die Richtung des steilsten Anstiegs definiert. Folglich ist der negative Gradient in die Richtung des steilsten Abstiegs. Dem negativen Gradienten folgt man dann in Richtung des Tiefpunktes, dabei wird die Schrittlänge durch den Steigungsgrad, der als Betrag des Gradienten bestimmt ist.

3 Anwendung

Neuronale Netze finden heutzutage schon in vielen Bereichen Anwendungsmöglichkeiten. Sie werden in der Bilderkennung genutzt, um dort Text, Zahlen, Gesichter oder auch bestimmte Bilder, beispielsweise Tiere zu erkennen. Sie werden auch im Bereich der Robotik und künstlichen Intelligenz genutzt, um etwa einen Roboter bestimmte Aufgaben lernen zu lassen. Ebenfalls sind Sprachassistenten durch neuronale Netze realisierbar, da diese auch zur Spracherkennung genutzt werden. Auch wird noch an neuronalen Netzen selber geforscht, um diese weiter zu verbessern.

3.1 Vor- und Nachteile

Neuronale Netze können, wenn sie richtig genutzt werden, viele Aufgaben vereinfachen. Sie benötigen weniger konkretes Vorwissen, da sie auch selber anhand von selbst erkannten Mustern die eingegebenen Daten einordnen können. Auch können Daten, die verzerrt oder unvollständig sind, noch von neuronalen Netzen erkannt und zugeordnet werden.[2]

Wenn man für ein neuronales Netz einen guten Trainigsdatensatz zusammenstellen kann, so kann ein Netz anhand von diesem auch unbekannte Daten im nachhinein sehr gut einordnen. Ein schlechter Trainingsdatensatz kann hingegen dafür sorgen, dass das Netz beim Lernvorgang diesen auswendig lernt oder anhand von unerwünschten Kriterien (beispielsweise Helligkeit) sortiert. Dieser Vorgang nennt sich Overfitting. Einen Trainingsdatensatz zu erstellen kann sich auch als schwierig herausstellen, da man für ein neuronales Netz so viele verschiedene Daten wie möglich benötigt und es je nach Art der Daten sehr aufwändig sein kann, diese zu sammeln.

Auch die Vorgänge innerhalb eines neuronalen Netzes sind schwierig nachzuvollziehen, wodurch es sich als schwierig erweisen kann, einen Fehler in einem Netz zu finden und zu beheben. Es gibt jedoch mittlerweile Bestrebungen dieses Problem zu beheben.[4]

3.2 Frameworks

Es gibt viele verschiedene Frameworks, mit denen sich neuronale Netze realisieren lassen. Im folgenden stelle ich zwei für Python entwickelte Frameworks näher vor.

3.2.1 Tensorflow

Das Framework Tensorflow [6] wurde vom Google-Brain-Team für die Entwicklung von Firmeneigenen Programmen als Nachfolger von DistBelief entwickelt. Heutzutage wird Tensorflow für eine Vielzahl von Google Diensten, wie Maps, Gmail, die Google-Suche und andere benutzt. Das Framework kann auch einfach auf der eigenen Seite heruntergeladen und zum eigenen Gebrauch genutzt werden.

3.2.2 PyTorch

Das Frameworck PyTorch [7] wurde vom Facebook AI Research lab entwickelt. Es wurde auf der Basis der Bibliothek Torch aufgebaut, die im Gegensatz zu PyTorch mittlerweile nicht mehr fortgeführt wird. PyTorch ist wie Tensorflow Open-Source und kann somit von jedem benutzt werden.

Bibliography

[1] D. Kriesel. Ein kleiner Überblick über Neuronale Netzes
http://www.dkriesel.com/_media/science/neuronalenetze-de-zeta2-2col-dkrieselcom.pdf

[2] M.Traeger, A. Eberhart, G. Geldner, A. M. Morin, C. Putzke, H.Wulf. L. H. J. Eberhart. Künstliche neuronale Netze Theorie und Anwendungen in der Anästhesie, Intensiv- und Notfallmedizin
https://sci-hub.st/10.1007/s00101-003-0576-x

[3] Spektrum Akademischer Verlag, Heidelberg. Lexikon der Neurowissenschaft neuronale Netze
https://www.spektrum.de/lexikon/neurowissenschaft/neuronale-netze/8653

[4] Frauenhofer Gesellschaft. Der Blick in neuronale Netze
https://www.fraunhofer.de/de/presse/presseinformationen/2019/juli/der-blick-in-neuronale-netze.html

[5] Wikipedia. Künstliches neuronales Netz
https://de.wikipedia.org/wiki/Künstliches_neuronales_Netz

[6] Tensorflow Website
https://www.tensorflow.org/

[7] PyTorch Website
https://pytorch.org/