ANALYSE UND PROTOTYPISCHE IMPLEMENTIERUNG NEURONALER NETZE IN SAP

LEONHARD SCHUSTER
(BACHELORSTUDIUM WIRTSCHAFTSINFORMATIK)
Betreuer: Prof. Dr. Bernhard Holaubek, Prof. Dr. Andreas Krüger

Motivation

Zahlreiche Unternehmen verwenden derzeit Enterprise Resource Planning Software, um ihre Geschäftsprozesse abzubilden und die anfallende Datenverarbeitung zu automatisieren. Dabei werden häufig personenbezogene Daten verarbeitet, welche einen besonderen Schutz benötigen. Diese sensiblen Informationen werden durch die Datenschutzgrundverordnung (DSGVO) innerhalb der Europäischen Union geschützt. Für Unternehmen stellen diese Daten wiederum einen großen Wert dar, weil sie vor allem für verschiedene Analysen notwendig sind. Auch zum Testen von neuer Software werden sie benötigt, damit möglichst viele Fehler im Vorhinein identifiziert und zu behoben werden können. Personenbezogene Daten dürfen nach der DSGVO jedoch nicht permanent gespeichert werden, was langfristige Analysen stark einschränkt. Außerdem dürfen sie nicht als Testdaten an Dritte weitergegeben werden. Dies Problem kann gelöst werden, indem die Daten anonymisiert werden.
Die paricon AG in Rosenheim bietet eine solche Anonymisierung von Daten auf SAP-Systemen an. Unter dem Begriff der Anonymisierung von Daten versteht man, dass die Daten so verändert werden, dass sie inhaltlich konsistent bleiben, jedoch keine Schlüsse auf die realen Personen hinter den ursprünglichen Daten gezogen werden können. Dieser Prozess ist bei einer großen Datenmenge Laufzeit intensiv. Die Motivation dieser Arbeit liegt darin, dass ein neuronales Netz die Geschwindigkeit der Anonymisierung potenziell verbessern kann. Dies soll durch eine Optimierung der Erkennung von personenbezogenen Daten und deren Einordnung in einen Typ erfolgen. Typen sind in diesem Fall zum Beispiel Vorname, Nachname, Geburtsdatum, Wohnort usw.
Die paricon bietet ihre Softwarelösungen in Form von Programmen in SAP an. Diese werden durch die SAP interne Programmiersprache ABAP implementiert und können somit an Kunden ausgeliefert werden. Da es mit einem erheblichen Mehraufwand verbunden ist externe Programme bei einem Kunden einzusetzen, soll auch das neuronale Netz in ABAP programmiert werden.

Zielsetzung und Vorgehen

Das Ziel der Arbeit ist die prototypische Implementierung eines neuronalen Netzes in ABAP und das anschließende Testen. Als Testszenario soll dabei eine einfache Handschrifterkennung auf Grundlage der MNIST-Datenbank erfolgen. Die MNIST-Datenbank besteht aus Datensätzen, welche handschriftliche Ziffern von 0 bis 9 darstellen. Das Format ist 28 x 28 Pixel, wobei für jedes der 784 Pixel der Schwarzwert angegeben wird.
Das Vorgehen der Arbeit setzt sich aus fünf Phasen zusammen. Zuerst erfolgt eine Literaturrecherche, in welcher die Grundlagen von neuronalen Netzen erläutert werden und Best-Practices herausgearbeitet werden. Im Anschluss soll erforscht werden, welche Datenstrukturen sich in ABAP eignen, um ein neuronales Netz möglichst performant zu implementieren. Aus diesem Wissen wird dann der erste Prototyp erstellt. Dieser soll auch später zur Validierung von komplexeren Implementierungen dienen. Der erste Prototyp wird dann für verschiedene Anwendungsfälle, wie zum Beispiel das Training auf das logisches UND-Gatter oder auf die Handschrifterkennung, getestet. Auf Grundlage dieser Ergebnisse soll dann eine Überarbeitung der Implementierung erfolgen, um gegebenenfalls Fehler zu beheben oder eine Optimierung vorzunehmen. Zum Schluss sollen die verschiedenen Implementierungen verglichen werden, um Vor- und Nachteile zu analysieren. Für diesen Verglich werden die Netze auf die Daten aus der MNIST-Datenbank zur Erkennung von handschriftlichen Ziffern trainiert und dann gegeneinander gebenchmarkt.

Grundlagen

Künstliche neuronale Netze bestehen aus einer Zahl von Neuronen, welche durch gewichtete Verbindungen miteinander verknüpft werden. Die Verknüpfung erfolgt meist anhand von Schichten (Layern), das heißt jedes Neuron aus einem Layer wird mit jedem Neuron aus dem nächsten Layer verknüpft (Abbildung 1). Dabei existiert eine Eingabeschicht, welche den Input entgegennimmt und dann an die folgende Schicht weiterleitet. Die Ausgabeschicht stellt den letzten Verarbeitungsschritt dar und erzeugt einen Output. Dazwischen liegende Schichten werden als innere Schichten bezeichnet. Die Neuronen erhalten über die Verbindungen zwischen den Schichten ein Eingabesignal, aus welchem durch eine Aktivierungsfunktion eine Ausgabe erzeugt wird. Diese Ausgabe wird über die gewichteten Verknüpfungen an die nächste Neuronenschicht weitergeleitet. Das Training des Netzes erfolgt über die Anpassung dieser Gewichte. Trainingsdaten werden vom neuronalen Netz verarbeitet und auf Grund der Abweichung der tatsächlichen Ausgabe von der optimalen/erwarteten Ausgabe, werden die Anpassungen der Gewichtungen berechnet. Der Vorteil im Vergleich zu klassischen Algorithmen besteht darin, dass das neuronale Netz unterschiedliche Funktionen abbilden kann, für welche kein sinnvoller Algorithmus formuliert werden kann.

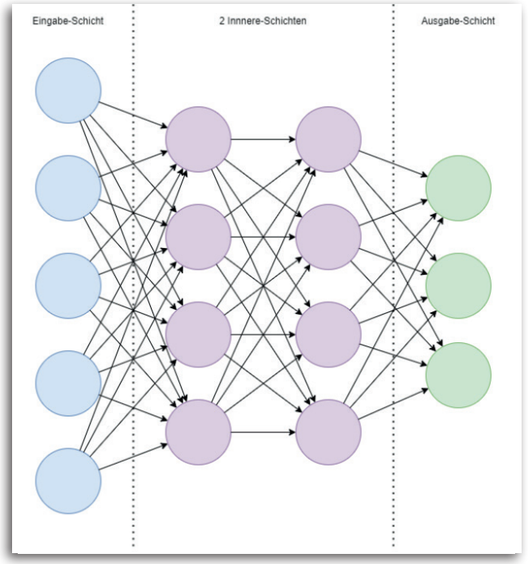

Abbildung 1 Vorwärtsgerichtetes neuronales Netz

Die Verarbeitung einer Eingabe bis hin zur Ausgabe eines neuronalen Netzes, kann mathematisch durch Matrizenmultiplikationen formuliert werden. Auch das Training benötigt diese mathematische Operation. Dieser Aspekt stellt einen wichtigen Punkt in der Implementierungs- und Optimierungsphase dar.

Ergebnisse

Der erste Prototyp stellte ein voll funktionsfähiges neuronales Netz mit einer variablen Anzahl an Neuronen und Layern dar. Die Implementierung selbst war im Vergleich zu einer Umsetzung mit Java oder anderen Programmiersprachen deutlich langsamer. Dies liegt daran, dass ABAP auf Datenverarbeitung anstelle von performanten Berechnungen optimiert ist. Deshalb liegen keine geeigneten Strukturen zur Umsetzung der Matrizenmultiplikation vor.
In der ersten Testphase konnte gezeigt werden, dass das neuronale Netz korrekt funktioniert. Nach einem Training auf die MNIST-Datenbank konnten handschriftliche Zahlen im vorgegebenen Format zuverlässig erkannt werden.
In der darauffolgenden Optimierungsphase wurden die Berechnung innerhalb des neuronalen Netzes durch eine komplexere Implementierung ersetzt. Beim Training und Testen der Daten aus der MNIST-Datenbank konnte folgendes erkannt werden. Das Erstellen und Initialisieren eines neuen Netzes war zwar 70% langsamer als im ersten Prototypen, jedoch konnte bei der Ausführung ca. 60% und dem Training ca. 64% der Ausführungszeit eingespart werden (Tabelle 1). Dies stellt eine Verbesserung dar, da die Erstellung nur einmal am Anfang erfolgt und somit vernachlässigt werden kann.
Zum Schluss wurde das optimierte Netz mit Hilfe der MNIST-Datenbank gebenchmarkt. Das neuronale Netz wurde durch Anpassung der Parameter auf diesen Anwendungsfall optimiert. Das beste Ergebnis erreichte eine Erkennungsrate von 95,84%.
Dieses Netz kann zukünftig auch auf andere Anwendungsfälle trainiert werden und somit als Grundlage für spätere Projekt in diesem Umfeld dienen.

	Optimierte Version (Ausführungszeit in Microsekunden)	Erster Prototyp (Ausführungszeit in Mikrosekunden)
Erstellen und Initialisieren eines Netzes	407.615	240.697
Training von 100 Datensätzen	10.117.735	27.859.220
Training eines Datensatzes	101.177	278.592
Ausführung eines Datensatzes	28.030	69.284

Tabelle 1 Laufzeitanalyse