Ein Machine Learning Ansatz zur Prognose der raumzeitlichen Verteilung von Mikromobilitätskonzepten

Bachelorarbeit

zur Erlangung des akademischen Grades „Bachelor of Science (B. Sc.)“ im Studiengang Wirtschaftsingenieur der Fakultät für Elektrotechnik und Informatik, Fakultät für Maschinenbau und der Wirtschaftswissenschaftlichen Fakultät der Leibniz Universität Hannover

vorgelegt von

Ricardo Andrés Ripphoff Castillo

[redacted] [redacted]

Prüfer: Prof. Dr. Breitner

Hannover, 28.09.2021

Inhaltsverzeichnis

1 Einleitung

Im Jahr 1950 eröffnete ALAN M. TURING seine wissenschaftliche Arbeit über Künstliche Intelligenz, *Computing Machinery and Intelligence*, mit seiner berühmten Fragestellung

„Can machines think?“

Der Versuch einer Antwort auf diese Frage ist seit jeher bekannt als *The Imitation Game*. Das Spiel handelt davon, dass ein Teilnehmer entweder eine Maschine oder ein Mensch sein kann und einer der Mitspieler durch gezieltes Fragen herausfinden soll, was von beidem zutrifft. Turing hat mit seiner Arbeit nicht nur eine philosophische Diskussion angestoßen, sondern für die damaligen Zeiten auch eine sehr technische, nämlich über die Entwicklung einer Maschine und die Nutzung dieser für Aufgaben, die Menschen nicht mehr bewältigen können (Turing 1950).

Was die Menschen im Jahr 1950 noch nicht wussten ist, dass Turing mit einem Team aus Spezialisten den Grundstein für die moderne Informatik und die Künstliche Intelligenz gelegt hatten. Die Motivation dazu war, dass das Enigma, die Verschlüsselungsmaschine der Deutschen im zweiten Weltkrieg, welche Funksprüche verschlüsselte, von den Allierten entschlüsselt werden sollte, um den Krieg zu beenden. Dazu hat Turing im Jahr 1940 seine *Turing-Bombe* entwickelt, was eine Maschine war, die unter Eingabe bekannter Werte, also bekannter Wörter aus einem Funkspruch, den Schlüssel ausgeben konnte, womit anschließend alle Funksprüche entziffert werden konnten. Die Existenz der Turing-Bombe war bis in die 1970er Jahre unter strenger Geheimhaltung (Wright 2017).

Die Turing-Bombe ist nach heutigem Verständnis noch weit weg von einer Künstlichen Intelligenz, kommt allerdings der Idee des Machine Learnings überraschend nah. Das Prinzip der Eingabe von bekannten Daten in eine Maschine, die daraus Zusammenhänge „lernt“ und diese anschließend ausgibt, war also bereits in den 1940er Jahren bekannt.

1.1 Einleitende Definitionen

Seit Turing und den 1940er Jahren haben sich die Möglichkeiten und das Verständnis von *Künstlicher Intelligenz* (KI) stark verändert. Während zur Zeit

Turings KI komplett aus Hardware, also buchstäblichen Maschinen bestand, ist KI heutzutage weitgehend softwarebasiert. Häufig genutzte Begriffe sind *Machine Learning* (ML) und *Deep Learning* (DL), welche, im Großteil der Literatur, Teilbereiche der KI sind (siehe Abbildung I). DL ist ein neueres Verfahren mit einer neuen Philosophie, welches erst in den 2010er populär wurde und als Teilmenge des MLs gilt (Wuttke 2021).

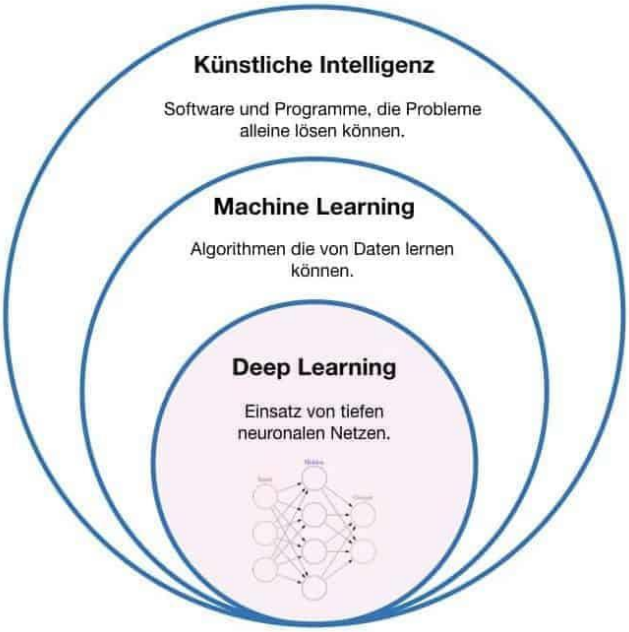

Abbildung I - Machine Learning vs. Deep Learning (Wuttke 2021)

ML arbeitet immer nach dem gleichen Prinzip; dem Modell werden Daten mit bekanntem Ergebnis eingegeben, womit das Modell dann Muster lernt, um mit neuen Daten ein Ergebnis auf Grundlage der gelernten Muster ausgeben kann. Die Menge an eingegebenen Daten hat direkte Auswirkungen auf die Qualität des Modells, allerdings nur bis zu einer bestimmten Grenze. Definitionsgemäß gehören normale *Artificial Neural Networks* (ANN), die ihre Anfänge ebenfalls bereits in den 1940er Jahren hatten, zum Machine Learning. Beispiele für ML im Alltag sind Produktempfehlungen in Online Shops, Vorhersagen des Kundenverhaltens, die Erkennung von Kreditkartenbetrugs und Verkehrsprognose (Bonetto und Latzko 2020; Wuttke 2021).

Deep Learning wurde ganz nach dem Vorbild des menschlichen Lernverhaltens entwickelt, also mit Neuronen und Verbindungen. Die Struktur eines DL Modells beginnt mit einer *input layer*, welche die Rohdateneingabe verarbeitet. Die aus der input layer ausgegebenen Daten durchlaufen dann mehrere *hidden layer*, in denen

die Daten weiterverarbeitet werden. Diese hidden layer sind technisch der Hauptunterschied zu einem normalen ANN, wo es nur eine hidden layer gibt. Ein DL Modell hat so tiefere Ebenen, woher der Name *Deep Learning* abgeleitet wurde. Die verarbeiteten Daten werden dann in eine *output layer* übergeben, wo die erlernten Muster ausgegeben werden können. Allerdings sind diese Muster für Menschen nur sehr schwer nachvollziehbar, da durch die Anzahl an hidden layers die Komplexität und die Anzahl der Berechnungen ins unfassbare ansteigt. Anders als bei ML Modellen können bei DL Modellen sehr viel größere Datenmengen verarbeitet und effizient analysiert werden. Deep Learning Modelle werden oft beim autonomen Fahren und in der Verkehrsprognose angewandt (Ai et al. 2018; Wuttke 2020).

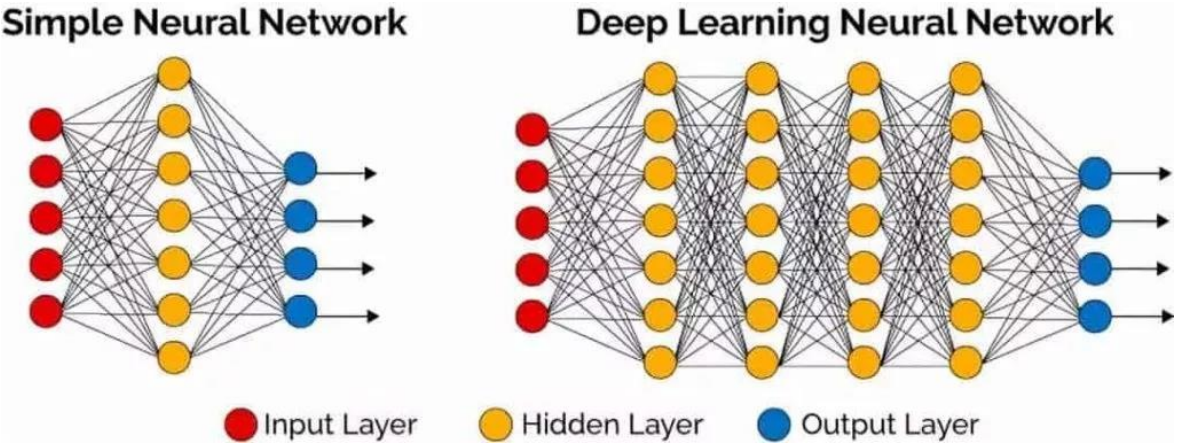

Abbildung II - Simple NN vs. Deep Learning NN (Wuttke 2020)

1.2 Relevanz

Seit einigen Jahren hat Mikromobilität immer weiter an Beliebtheit gewonnen. Vor allem, seitdem stationäre Systeme ersetzt werden durch dockless/free floating Systeme. Die Idee solcher dockless Systeme ist, dass Fahrzeuge, wie Fahrräder, Autos oder E-Scooter, in einer Stadt so verteilt sind, dass Menschen nur kurze Distanzen zurücklegen müssen, um leihweise eines dieser Fortbewegungsmittel nutzen zu können. Da vor allem Mikromobilität, also Fahrräder und E-Scooter, zusätzlich noch den Umweltgedanken mitbringen, untersucht diese Arbeit eine wichtige und notwendige technische Herausforderung, die es zu überwinden gilt; die Prognose der raumzeitlichen Verteilung von Mikromobilitätskonzepten. Wenn Betreiber in der Lage sind, die Nachfrage an Fortbewegungsmitteln raum- und zeitabhängig zu prognostizieren, können sie gezielter Fahrräder und E-Scooter positionieren, um die Nachfrage bedienen zu können. So würde sich nicht nur die

Kundenzufriedenheit und der Unternehmensumsatz steigern, sondern hätte auch positive Auswirkungen auf die Klimabilanz, wenn dadurch Menschen von Autos auf Fahrräder und E-Scooter umsteigen. Da vor allem E-Scooter ein neuer Trend mit großem Potenzial sind, wird in der zweiten Hälfte dieser Arbeit ein Prognosemodell untersucht, angepasst und getestet. Dieses wurde für dockless Systeme entwickelt und wird hier auf die E-Scooter Ökonomie in Berlin angewandt.

Um sich einen E-Scooter von einem Betreiber ausleihen zu können, müssen Nutzer sich die App des Betreibers herunterladen, dort eine Zahlungsmöglichkeit, wie ihre Kreditkarte oder PayPal, hinterlegen und dann an einem E-Scooter einen QR-Code scannen, um diesen freizuschalten. Außerdem kann man sich in der App eine Karte mit den zurzeit verfügbaren E-Scootern anzeigen lassen, um den nächsten finden zu können. Wenn man nun seine Fahrt beendet hat, kann man den E-Scooter wieder freigeben. Während ein E-Scooter vom Kunden gefahren, also neu positioniert, wird, taucht dieser nicht auf der Karte auf. Mit dieser Information können also einzelne Fahrten extrahiert wären, was in Kapitel 3.1 beschrieben wird.

Aus der Notwendigkeit das Nutzerverhalten prognostizieren zu müssen, lassen sich drei Forschungsfragen ableiten:

1. Welcher technische Ansatz ist für eine Prognose von dockless micromobility Modellen geeignet?
2. Welche Datengrundlage wird für eine genaue Prognose benötigt?
3. Wie performt der ausgewählte Ansatz mit der Datengrundlage dieser Arbeit?

Ziel dieser Arbeit ist es Antworten auf diese Fragen zu finden.

1.3 Struktur der Arbeit

Im folgenden Kapitel wird zuallererst der aktuelle Stand der Forschung von Machine Learning Ansätzen in der Mikromobilität vorgestellt. Grundlage dessen sind diverse wissenschaftliche Arbeiten, die bis zum 28.07.2021 veröffentlicht wurden. Dabei wird kurz auf traditionelle statistische Modelle eingegangen, welche in der Statistik immer noch große Bedeutung haben. Außerdem werden bekannte Machine Learning Ansätze, die bereits von Forschungsteams zur Prognose in der Mikromobilitätsbranche untersucht wurden, vorgestellt, sowie einige Deep Learning Ansätze, die vor allem in den letzten Jahren populär geworden sind. Als letztes wird ein hybrides Machine Learning Modell vorgestellt, welches zwar noch nicht auf

dockless Systeme abgestimmt ist, allerdings trotzdem Potenzial hat, aktuelle Ansätze abzulösen.

In Kapitel 3 wird die Datengrundlage analysiert und für das Prognosemodell präpariert. Die Rohdaten stammen aus *Data Mining*, was eine Methode zum Sammeln von Daten in sehr großen Mengen ist. Das Sammeln von Daten an sich ist kein Teil der Arbeit, da diese bereits von den Betreuern dieser Arbeit zur Verfügung gestellt wurde. Sehr wohl ist aber das Präparieren der Rohdaten und Aggregieren weiterer externer Daten Bestandteil der Arbeit.

Das in dieser Arbeit angewandte und angepasste Modell wird in Kapitel 4 vorgestellt und die Funktionsweise erläutert. Als Grundlage dieses Kapitels dienen unter anderem die Forschungsergebnisse von XU ET AL., welche ihr Modell im Jahr 2018 vorgeschlagen und es mit einem dockless Bike-Sharing System getestet haben. Die Ergebnisse aus Kapitel 3 bilden die Datengrundlage.

In einer Case Study, Kapitel 5, wird das angepasste Modell getestet und die Qualität des Modells numerisch festgestellt. Um die Qualität mit anderen Modellen vergleichen zu können, werden diverse Messmethoden unter Berücksichtigung verschiedener Ausbaustufen der Datengrundlage angewendet. Ziel ist es, die Wichtigkeit externer Einflüsse auf das Nutzerverhalten zu extrahieren und zu präsentieren.

In Kapitel 6 werden die Ergebnisse des Modells ausgewertet und diskutiert. Außerdem wird das Modell mit dem von XU ET AL. verglichen. Die Qualität dieser Arbeit wird analysiert, aber auch Grenzen werden aufgezeigt. Aus den Ergebnissen werden Implikationen abgeleitet.

Das persönliche und abschließende Fazit beendet diese Arbeit im siebten Kapitel.

Prognosegenauigkeit aufweist. Trotzdem soll klargestellt werden, dass Data Mining im größeren Stil eine deutlich elegantere Lösung wäre, wenn die Möglichkeit dazu besteht.

Eine letzte Implikation kann auf Grundlage der Erfahrung der technischen Ausarbeitung des Modells getroffen werden. Tensorflow und Keras wurden ursprünglich für Python entwickelt und funktionieren auch gut in Python. Allerdings ist diese Programmiersprache für das Data-Preprocessing aufgrund eines sehr geringen Datendurchsatzes nicht geeignet. Vor allem der Quellcode der pandas Bibliothek kostet das Modell ein hohes Maß an Performance. Empfohlen wird den gesamten Teil des Preprocessings in Java auszulagern, daraus eine fertige Datenbank zu erstellen und nur das Neuronale Netz in Tensorflow in Python zu realisieren. So wird die Effizienz auf ein Maximum gesteigert. Falls Anwender sehr häufig Neuronale Netze entwickeln wollen und diese ein hohes Maß an Komplexität erreichen, so wird empfohlen, entweder das Neuronale Netz mit den vorgefertigten Tensorflow Bibliotheken für Java auszulagern oder eigene Bibliotheken zu entwickeln. Auf jeden Fall wird die Auslegung auf multiprocessing, also das Verteilen der Rechenprozesse auf so viele Threads wie möglich, empfohlen, da dadurch die Rechenzeit verkürzt wird.

7 Abschließendes Fazit

Ziel dieser Arbeit war es die in Kapitel 1 gestellten Forschungsfragen zu beantworten. Es wurde ein geeignetes Modell, sogar das mit den höchsten Erfolgsaussichten, ausgewählt und getestet. Die Datengrundlage konnte für das Modell vorbereitet werden und der Prognoseerfolg mit dem Originalmodell verglichen werden. Obwohl dieses Modell in den Grundzügen funktioniert und unter den Umständen der kleinen Datenbank einen akzeptablen Erfolg erzielte, so konnte der persönliche Anspruch, ein Modell zu entwickeln, welches den gleichen oder sogar höheren Prognoseerfolg erzielt, nicht erreicht werden.

Trotzdem wurde ein Modell auf Grundlage von XU ET AL. entwickelt und sogar weiterentwickelt. Die Entwicklung in Python war dabei so hinderlich, dass bei einer erneuten Entwicklung eines Prognosetools so viel wie möglich in Java programmiert

werden sollte. *Eclipse* hat sich als IDE dafür sehr bewährt. Da mittlerweile von Tensorflow auch schon Bibliotheken für Java ausgegeben wurden, könnte auch der komplette Umstieg auf Java in Erwägung gezogen werden. Allerdings gibt es für diese noch keine Stabilitätsgarantie von Seiten des Google Brain Teams her. Auf jeden Fall hätte das Entwickeln des Modells in deutlich weniger Zeit realisiert werden können, wenn auf Python verzichtet worden wäre.

Abschließend kann klar gesagt werden, dass alle Forschungsfragen beantwortet werden konnten und das LSTM NN ein sehr erfolgsversprechender Ansatz zur Prognose der raumzeitlichen Verteilung von Mikromobilitätskonzepten ist.