Karlsruhe Institute of Technology

Prädiktion von Aktienkursen mit Neuronalen Netzen

Bachelorarbeit
von

Daniel Handloser

An der Fakultät für Informatik
Institut für Anthropomatik und Robotik

Erstgutachter:	Prof. Dr. Alexander Waibel
Zweitgutachter:	Prof. Dr.-Ing. Tamim Asfour
Betreuender Mitarbeiter:	Dr.-Ing. Jan Niehues

Bearbeitungszeit: 11. Januar 2017 – 10. Mai 2017

Ich versichere hiermit, die vorliegende Arbeit selbstständig und ohne fremde Hilfe angefertigt zu haben. Die verwendeten Hilfsmittel und Quellen sind im Literaturverzeichnis vollständig aufgeführt.

Karlsruhe, den 5. Mai 2017

Abstract

Diese Arbeit untersucht Kurzmitteilungen der Plattform Twitter über Unternehmen aus dem Dow Jones Industrial Average (DJIA) und dem Deutschen Aktienindex (DAX). Mit den Daten soll die Entwicklung des Aktienkurses dieser Unternehmen prognostiziert werden.

Zwischen Mitte Dezember 2016 und Anfang Februar 2017 wurden circa 20 Millionen Kurzmitteilungen über die 30 größten amerikanische Aktienkunternehmen aus dem DJIA und die 30 größten deutschen Aktienunternehmen aus dem DAX gesammelt. Mit Hilfe von verschiedenen Neuronalen Netzen wurde versucht, mit Twitterdaten auf eine Vorhersage für den Aktienkurs an einem Folgetag zu schließen.

Dazu wurden zwei Ansätze verfolgt. Im ersten Ansatz wurde bei der Prädiktion mittels Sentimentanalyse bestimmt, wie positiv oder negativ die Aussage eines Tweets ist. Mit dieser Information wird überprüft, ob es einen Zusammenhang zwischen dem durchschnittlichen Sentiment an einem Tag und dem Verlauf des Aktienkurs am nächsten Tag gibt.

Beim zweiten Ansatz – der direkten Prädikation – wird der Datensatz in einen Trainings- und Testdatensatz aufgeteilt. Mit den Zusammenhängen aus Tweets und Aktienkursen im Trainingsdatensatz wird versucht, mit den Tweets im Testdatensatz auf die Kurse in diesem Zeitraum zu schließen.

Mit der Prädiktion über die Sentimentanalyse wird in 49% der Fällen richtig vorhergesagt, ob der Aktienkurs am nächsten Tag fällt oder steigt. Bei der direkten Prädiktion wird in 61% der Fälle richtig prognostiziert, ob der Aktienkurs vier Tage später steigt oder fällt.

Inhaltsverzeichnis

1. Einleitung

1.1 Motivation

Dass sich Mitteilungen auf Twitter auf den Aktienmarkt auswirken können, wurde während der Präsidentschaftswahl 2016 in den USA wiederholt in den Medien diskutiert.[1][2] Grund dafür war Donald Trump, der vermehrt durch seine Äußerungen über verschiedene Unternehmen aufgefallen ist. Bei positiven Tweets des 45. Präsidenten der Vereinigten Staaten über die Unternehmen Ford und ExxonMobil stieg der Aktienkurs zum Beispiel am selben Tag um je 4,6% und 1,8% an.[1] Äußerte er sich zum Beispiel negativ über Lockheed Martin, so fiel der Kurs am selben Tag um 1,3%.[1] Obwohl sich alle Kurse über einen längeren Zeitraum anschließend wieder in die entgegengesetzte Richtung entwickelten, ist zumindest kurzfristig ein Zusammenhang zwischen Donald Trumps Tweets und den Aktienkursen der Unternehmen zu erkennen.

Naheliegend ist die Vermutung, dass solche Zusammenhänge nicht nur zwischen Donald Trumps Tweets und dem Aktienmarkt existieren. Daher untersucht diese Arbeit den Zusammenhang zwischen der öffentlichen Meinung über ein Unternehmen und dessen Aktienkurs auf Basis von Twitterdaten.

1.2 Überblick

Das *Kapitel 2* verschafft einen Überblick über den aktuellen Forschungsstand der Textklassifikation. Im darauffolgenden Grundlagenteil werden Neuronale Netze eingeführt.

In *Kapitel 4* werden verschiedene Analysemethoden vorgestellt. Im ersten Teil "Prädiktion mittels Sentimentanalyse" wird für jede Nachricht auf Twitter ein Sentiment berechnet. Dieses sagt aus, wie positiv oder negativ der Inhalt ist. Damit wird untersucht, ob sich positive oder negative Meinungen auf Twitter auf den Aktienkurs eines Unternehmens auswirken. Der zweite vorgestellte Ansatz ist die direkte Prädiktion. Hier soll von Neuronalen Netzen direkt anhand von Tweets entschieden werden, ob der Aktienkurs an einem späteren Tag fällt oder steigt.

Kapitel 5 beschäftigt sich mit der Evaluation der Arbeit. Hier werden verschiedene Optimierungen und ihre Auswirkungen vorgestellt, sowie die Ergebnisse interpretiert und miteinander verglichen.

Im letzten Kapitel werden einige mögliche Verbesserungen für die Zukunft vorgestellt und die Ergebnisse der Arbeit werden zusammengefasst.

2. Stand der Forschung

Natural Language Processing ist ein Teilbereich der Informatik und Linguistik, der sich mit der Interaktion von Computern und natürlicher Sprache beschäftigt.

Mittels Data Mining und Natural Language Processing eröffnen sich viele Möglichkeiten, die Flut an Textdaten aus dem Internet zu sammeln, zu filtern und daraus Informationen zu gewinnen. Textklassifikation und Sentimentanalyse stellen zwei zentrale Bausteine zur Bewältigung dieser Herausforderung dar.[3]

2.1 Textklassifikation

Die Klassifikation von Texten lässt sich mittels Machine Learning durchführen. Oft verwendete Ansätze sind Naive Bayes, Maximum Entropy Classification (MEC), Support Vector Machines (SVM)[4] und Neuronale Netze[5].
Abhängig von der Domäne und den verwendeten Parametern werden unterschiedliche Ergebnisse erzielt, so dass keine Methode die eindeutig besten Resultate liefert.[4][5]. Ein Vorteil von der SVM gegenüber Neuronaler Netze ist die effizientere Berechnung.[6]. Bei einer Kombination beider Verfahren, in der das Neuronale Netz die Vektorrepräsentation des Texte festlegt und die SVM die Klassifikation übernimmt, können so zum Beispiel bessere Ergebnisse als mit den einzelnen Verfahren erreicht werden.[6].

2.1.1 Sentimentanalyse

Ein verbreiteter Anwendungsfall von Textklassifikation ist die Sentimentanalse. Ziel der Analyse ist es, die geäußerte Haltung (oder Sentiment) in einem Text als positiv oder negativ zu klassifizieren.[7]

Die aktuellen Methoden beruhen hauptsächlich darauf, dass der Verfasser in Teilen seines Textes explizit seine Meinung und emotionalen Standpunkt ausdrückt. Werkzeuge, die erkennen, wenn implizit ein Sentiment geäußert wird, sind noch relativ wenig erforscht.[3][8]

Die Ergebnisse einer Sentimentanlyse auf einem großen Datenbestand lassen sich durch aufsummieren von Sentimentwerten recht einfach zusammenfassen. Allerdings existieren auch komplexere Methoden, mit denen Sentiments strukturierter zusammengefasst werden. Diese sind Gegenstand der aktuellen Forschung.[9] Beispiele dafür sind vor allem probabilistische Modelle wie Probabilistic Latent Semantic Analysis (PLSA) und Latent Dirichlet Analysis (LDA).[9]

2.2 Vorhersagen mit sozialen Netzwerken

Vorhersagen mit Hilfe von Daten aus sozialen Netzwerken wurden bereits in sehr vielen Bereichen erstellt.

Mit Hilfe des Wörterbuchansatzes Linguistic Inquiry and Word Count (LIWC) wurden zum Beispiel Vorhersagen zu den Bundestagswahlen getroffen (Tumasjan et al. 2010[10]).

Zudem wurde die Ausbreitung von Krankheiten untersucht (Culotta, 2010[11]), indem überprüft wurde, ob in Tweets spezielle Schlüsselwörter vorkommen. Bei der Vorhersage der Influenzaausbreitung in den Vereinigten Staaten wurde eine Korrelation von 95% mit der nationalen Gesundheitsstatistik erreicht.

Zwischen dem Umsatz von Kinofilmen und den Einträgen auf Blogs im Internet über Filme konnte ebenfalls eine positive Korrelation gezeigt werden (Mishne und Glance, 2006[12]).

2.2.1 Prognosen auf Aktienmärkten

Auch Prognosen auf Aktienmärkten wurden bereits mit einer Vielzahl von Datenquellen durchgeführt. Unter anderem wurden Blogs, Empfehlungen von Sicherheitsanalysten, Suchanfragen im Internet, Foren zum Thema Aktienmarkt oder Nachrichten in der Finanzbranche als Grundlage für Prognosen auf dem Aktienmarkt zu Rate gezogen.[13] Im Folgenden soll allerdings auf die Arbeiten eingegangen werden, die sich mit der Analyse von Informationen auf Twitter beschäftigten.

Die wohl bekannteste Arbeit zur Bestimmung von Aktienkursen mit Hilfe von Twitter stammt aus dem aus dem Jahr 2011 von Johan Bollen und Huina Mao.[14] Untersucht wurden hier die Auswirkungen von Tweets auf den DJIA. Dabei wurde ein Zusammenhang zwischen Tweets und Aktienindex nach drei bis vier Tagen festgestellt. Zur Analyse der Tweets wurden zwei Werkzeuge verwendet. Zum einen der OpinionFinder, welcher Texte nach positivem und negativem Sentiment klassifizert. Zum anderen das Google-Profile of Mood States (GPOMS), welches die Haltung eines Textes in sechs Dimensionen (”Calm”, ”Alert”, ”Sure”, ”Vital”, ”Kind” und ”Happy”) analysiert. Während sich Ersteres für diese Arbeit als wenig aussagekräftig herausstellte, wurde für das GPOMS mit der Dimension ”Calm” eine positive Korrelation mit dem DJIA drei bis vier Tage später festgestellt. Bei der Vorhersage, ob der Schlusskurs des Index steigt oder fällt, wurde eine Accuracy von 87.6% erreicht.

Die Arbeit ”Tweets and Trades: the Information Content of Stock Microblogs”[15] untersucht erstmals nur Nachrichten, die sich auch spezifisch auf den Finanzmarkt beziehen. Berücksichtigt werden nur die Tweets die ein Dollarzeichen gefolgt von einem Aktien-Symbol enthalten. Das sorgt dafür, dass der Datensatz vor allem Nachrichten mit direktem Bezug zum Aktienmarkt enthält. Mittels Naïve Bayesian wurden die Texte in drei Kategorien klassifiziert: Kaufen, behalten und verkaufen. In der Arbeit wurde lediglich eine statistisch unsignifikante Korrelation von 0,5% zwischen Tweets und dem Aktienkurs des genannten Unternehmen einen Tag später festgestellt. In die umgekehrte Richtung – vom Aktienkurs auf die Twitterdaten einen Tag später – wurde eine signifikante Korrelation von 13,2% festgestellt. Zudem zeigte sich für den betrachteten Datensatz, dass Nutzer mit hoher Anzahl an Followern und vielen Retweets die besseren Prognosen über den Aktienmarkt abgeben.[15] Außerdem wurde eine Korrelation von 44,1% zwischen der Anzahl an Nachrichten zu einem Unternehmen auf Twitter und dem Transaktionsvolumen der zugehörigen Aktie an der Börse festgestellt.[15]

Im gleichen Jahr wurde in der Arbeit ”Predicting Stock Market Indicators Through Twitter ”[16] ebenfalls ein Zusammenhang zwischen der allgemeinen Stimmung auf Twitter und

einem Aktienindex festgestellt. Die Arbeit fand eine positive Korrelation zwischen Tagen, an denen viel Angst, Freude und Sorge auf Twitter ausgedrückt wird, und einem sinkenden Kurs des DJIA am nächsten Tag. An Tagen, an denen weniger emotionale Mitteilungen verschickt werden, steigt der Index laut den Autoren. Für die Dimension "Hope" wurde eine Korrelation von 38,1% mit der Entwicklung des DJIA festgestellt.

Eine Arbeit von Sam Paglia aus dem Jahr 2013 untersucht die Tweets von Nachrichtenunternehmen und versucht damit, den Wert des Aktienindex Standard & Poor's 500 (S&P 500) zu prognostizieren. Für die Vorhersage, ob der Tagesschlusskurse am nächsten Börsentag fällt oder steigt, wurde eine Accuracy von 59% erreicht.[17]

Die Vorhersage eines Aktienindexes mit Hilfe von der öffentlichen Meinung auf Twitter wurde bereits ausführlich in verschiedenen Arbeiten behandelt. Für die feingranularere Prädiktion von Aktienkursen einzelner Unternehmen existieren bisher nur wenige Veröffentlichungen. In dieser Arbeit werden daher Twitterdaten verwendet, die explizit ein Aktienunternehmen nennen. Mit Hilfe dieser Daten wird analysiert, ob der Aktienkurs an einem späteren Tag fällt oder steigt. Betrachtet wird demnach nicht die allgemeine öffentliche Meinung auf Twitter, sondern die öffentliche Meinung zu konkreten Unternehmen. Damit soll der Aktienkurs von spezifischen Unternehmen prognostiziert werden.

3. Grundlagen Neuronale Netze

Ein künstliches Neuronales Netz besteht aus Schichten mit künstlichen Neuronen. Die Struktur und Funktion stellt eine Abstraktion des Nervensystems und speziell des Gehirns von Tieren und Menschen dar.

Die Einsatzbereiche von neuronalen Netzen befinden sich vor allem in Feldern wie Sprach- und Bilderkennung, in denen Lösungen mit traditioneller logischer Programmierung nur schwer erreichbar sind.[18]

3.1 Künstliches Neuron

Ein Netz besteht aus mindestens einem oder mehreren parallel arbeitenden Neuronen.[19] Diese senden sich Informationen in Form von Aktivierungssignalen über gerichtete Verbindungen zu.[20] Jedes Neuron erhält eine oder mehrere Eingaben und gibt eine Ausgabe zurück.

Das McCulloch-Pitts-Neuron (Abbildung 3.1) ist ein einfaches und weit verbreitetes Neuronenmodell, das 1943 vorgestellt wurde. Die Eingabesignale x_i werden mit den entsprechenden Gewichten w_i multipliziert, bevor sie das Neuron erreichen. Die gewichteten Eingaben $x_i * w_i$ werden aufsummiert und ergeben die Aktivierungsspannung Σ. θ ist der Schwellwert (*engl.* Bias), ab dem eine bestimmte Ausgabe erzeugt wird. Das Aktivierungspotenzial u ergibt sich als Differenz von Σ und θ. Ist $u > \theta$, wird ein erregendes Signal erzeugt. Andernfalls ist das Signal hemmend. Die Aktivierungsfunktion g sorgt dafür, dass die Ausgabe des Neurons innerhalb eines bestimmten Wertebereichs liegt. Daraus entsteht das endgültige Ausgabesignal y, das wiederum als Eingabe für weitere Neuronen dienen kann.[18]

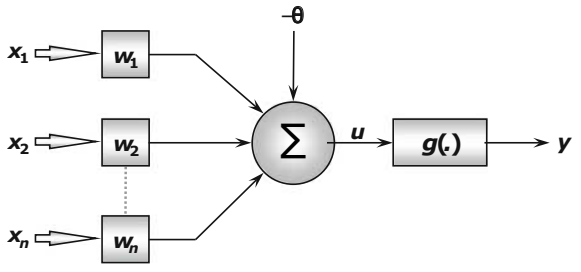

Abbildung 3.1: McCulloch-Pitts-Neuron[18]

3.2 Aufbau

Der Aufbau eines Neuronalen Netzes legt fest, wie die verschiedenen Neuronen miteinander verbunden werden.[18] Typischerweise besteht ein Netz aus einer Eingabeschicht, eventuellen verstecken Schichten und einer Ausgabeschicht.

Die Eingabeschicht nimmt Informationen entgegen. Diese Eingaben werden in der Regel einheitlich normalisiert. Auch die Ausgabeschicht besteht aus Neuronen. Diese erzeugt die endgültigen Ergebnisse des Netzes. Dazwischen können sich beliebig viele versteckte Schichten befinden, die ebenfalls aus Neuronen bestehen.[18]

Die Anzahl der Neuronen in der Ausgabeschicht hängt von der Problemstellung ab. Bei Regressionsproblemen enthält die Ausgabeschicht in der Regel ein Neuron, während bei Klassifizierungsproblemen die Anzahl der Neuronen der Anzahl der Klassen entspricht.

Generell wird zwischen einschichtigen und mehrschichtigen Feedforward-Netzen, rekurrenten neuronalen Netzen und Mesh-Netzen unterschieden. Auf die für diese Arbeit relevanten Varianten wird im Folgenden eingegangen.[18]

3.2.1 Feedforward Netz

Feedforward-Netze zeichnen sich dadurch aus, dass eine Schicht immer nur mit der nachfolgenden Schicht verbunden ist.

Ein einschichtiges Feedforward-Netz besteht aus einer Eingabeschicht mit n Eingaben und m Ausgaben. Die Informationen fließen in eine Richtung – von der Eingabe- zur Ausgabeschicht.[18]

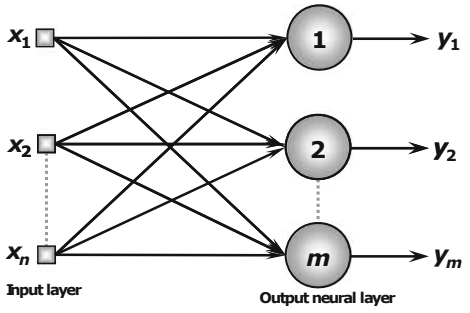

Abbildung 3.2: Einschichtiges Feedforward[18]

Mehrschichtige Feedforward-Netze besitzen mindestens eine versteckte Schicht. Abbildung 3.3 zeigt ein solches Netz mit einer Eingabeschicht mit n Eingabesignalen, zwei versteckten Schichten mit n_1 und n_2 Neuronen und einer Ausgabeschicht mit m Neuronen. Die Variable m entspricht dabei auch der Anzahl der Ausgabewerte des Netzes.[18]

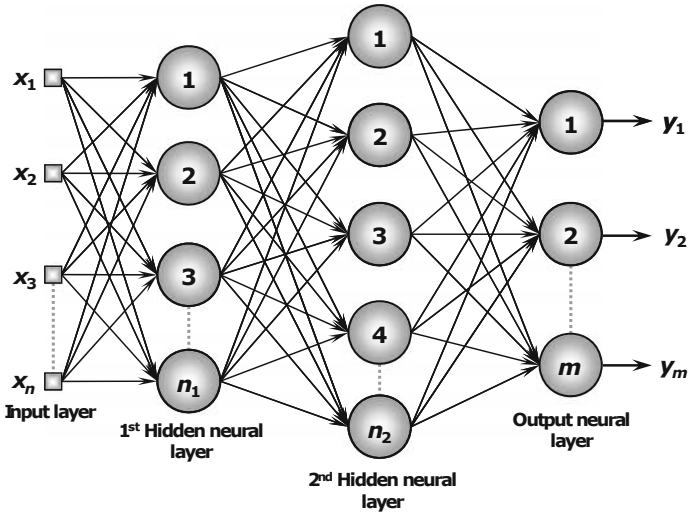

Abbildung 3.3: Dreischichtiges Feedforward[18]

3.2.2 Rekurrente Netze

Bei rekurrenten neuronalen Netzen werden die Ausgaben der Neuronen als Rückmeldung an Neuronen der vorherigen Schichten gegeben. Diese Rückkopplung wird durch rückgerichtete (rekurrente) Kanten erzeugt. So ist es möglich, ein Ergebnis an vorherige Schichten weiterzuleiten.[18]

Die Motivation für rekurrente Netze rührt vor allem von einer Schwäche der Feedforward-Netze. Wenn ein Feedforward-Netz zum Beispiel ein Wort in einem Satz betrachtet, stehen diesem keine Informationen über die vorherigen Wörter in dem Satz zur Verfügung. Durch die Rückkopplung in rekurrenten Neuronalen Netzen ist es möglich, Informationen über die vorherigen Eingaben zu speichern.

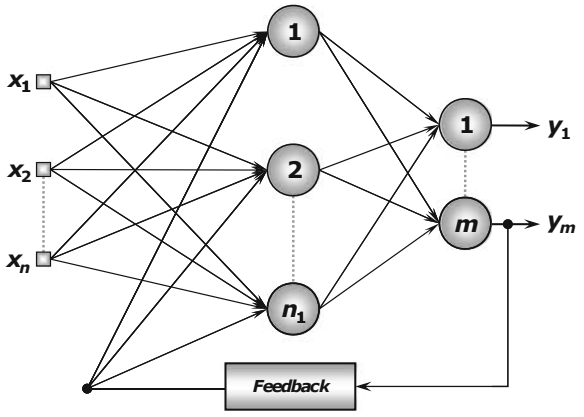

Abbildung 3.4: Rekurrentes Netz[18]

3.3 Schichten

Zwischen der Ein- und Ausgabeschicht kann ein Neuronales Netz eine Vielzahl verschiedener Schichten enthalten. Im Folgenden werden vier Schichten vorgestellt, die in dieser Arbeit verwendet wurden.

3.3.1 Feedforward-Netze

3.3.1.1 Reshape-Schicht

Eine Reshape-Schicht L_i wird eingesetzt, um die Ausgabe der vorherigen Schicht L_{i-1} an die Eingabe der darauffolgenden Schicht L_{i+1} anzupassen. Die Schicht nimmt einen Tensor entgegen und wandelt ihn in einen Tensor mit der gleichen Anzahl an Elementen um.[21]

Dabei wird die Dimension der Eingabe verändert ohne die Daten zu verändern. Ein Beispiel wäre das Überführen einer 2×2-Matrix in eine 4×1-Matrix.

3.3.1.2 Convolutional-Schicht

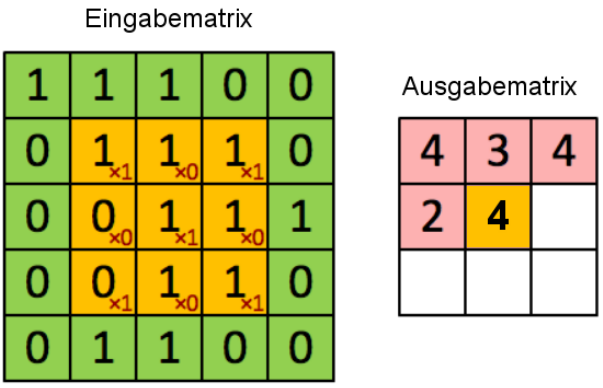

Abbildung 3.5: Die fünfte Anwendung einer Maske in einer Convolutional-Schicht[22]

Eine Convolutional-Schicht verringert die Anzahl der Ausagaben im Verhältnis zur Eingabe mit Hilfe einer Faltung (*engl.* Convolution). Dabei wird schrittweise eine Maske (*engl.* Kernel), die kleiner oder gleich groß wie die Eingabematrix ist, über die Eingabe bewegt. Die Elemente der Matrix werden mit den Gewichten der Maske verrechnet, aufsummiert und ergeben dann ein Element in der Ausgabematrix. Die Anzahl, wie oft die Maske auf die Eingabe angewandt wurde, entspricht so der Anzahl der Ausgabeelemente.

In einer Schicht können mehrere Masken angewandt werden. Eine Maske wird bei der gesamten Eingabe mit den gleichen Gewichten verwendet. Man spricht hier auch von *Shared weights*, da die Gewichte bei allen Anwendungen der Maske gleich bleiben. In der Bildverarbeitung sorgt das dafür, dass im gesamten Bild die gleichen Muster erkannt werden. Ein weiterer Vorteil ist, dass weniger Parameter optimiert werden müssen, was die Verarbeitung beschleunigt.[23]

3.3.1.3 Pooling-Schicht

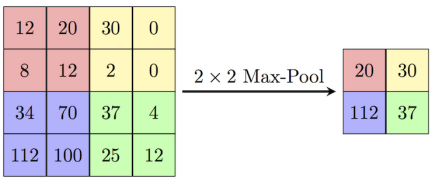

Abbildung 3.6: Max-Pooling [24]

Mit Pooling werden Informationen reduziert. Die Reduktion der Daten bringt verschiedene Vorteile mit sich. Zum einen verringert sich der Speicherbedarf und die Berechnungsgeschwindigkeit wird erhöht. Zum anderen wird verhindert, dass sich die Gewichte und

Schwellwerte des Netzes zu sehr an die Trainingsdaten anpassen und dadurch nicht mehr generalisieren können. Tritt letzteres doch auf, spricht man von *Overfitting*. Verwendet wird die Pooling-Schicht oft nach einer Convolutional-Schicht. Dadurch wird die Vielzahl an Parametern, die die Convolutional-Schicht ausgibt, reduziert. Im Idealfall werden so Störfaktoren beseitigt und relevante Informationen behalten.[18]

Eine Variante des Poolings ist das Max-Pooling. Dabei wird, wie bei allen Pooling-Schichten, eine Fenstergröße festgelegt. Die Eingabematrix wird in Fenster unterteilt und von jedem Fenster wird lediglich das maximale Element in die Eingabe übernommen. Ist die Größte der Eingabe durch die Fenstergröße teilbar, so sind alle Fenstern gleich groß. Ansonsten ist das letzte Fenster echt kleiner als die Fenstergröße.[25] In Abbildung 3.6 wird Max-Pooling auf eine 4×4-Matrix mit einem 2×2-Fenster angewandt.

3.3.1.4 Vollvernetzte Schicht

In einer vollvernetzten Schicht L_i dient die Ausgabe jedes Neurons als Eingabe aller Neuronen in der Schicht L_{i+1}. Ein Beispiel dafür ist Abbildung 3.3.

3.3.2 Rekurrente Neuronale Netze

Klassiche rekurrente Neuronale Netze eignen sich zum Beispiel, um das letzte Wort in dem Satz "Die Bäume stehen im *Wald*" vorherzusagen. Möchte man jedoch das letzte Wort im Text "Ich bin in England aufgewachsen [...] Ich spreche fließend *Englisch*" vorhersagen, so lässt sich aus den unmittelbar bevorstehenden Informationen ableiten, dass es sich bei dem letzten Wort um eine Sprache handelt. Theoretisch ist ein rekurrentes Netz auch in der Lage, Langzeitabhängigkeiten zu erkennen. Doch je größer der Abstand zwischen den Informationen ist, desto höher die Wahrscheinlichkeit, dass das Netz keinen Zusammenhang zum anfänglichen Satz herstellen kann.[26].

Die Möglichkeit, Abhängigkeiten über mehrere Eingaben hinweg zu erkennen, lässt sich mit dem Time Delay Neural Network (TDNN), einem Feedforward-Netz, umsetzen.[27] Ein rekurrentes Netz mit dieser Funktionalität ist das Long Short-Term Memory (LSTM).

3.3.2.1 Long Short-Term Memory (LSTM)

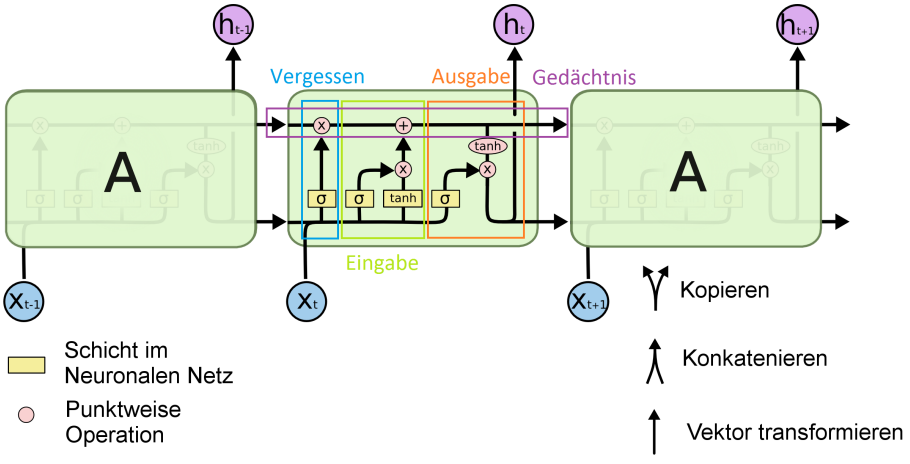

Abbildung 3.7: LSTM[28]

Die Abbildung 3.7 illustriert ein LSTM mit Eingabe x_t und Ausgabe h_t für Berechnung t.

Charakteristisch ist vor allem das Gedächtnis (horiziontaler Pfeil oben in der Abbildung), das eine Liste von Zuständen speichert. Die drei Gatter "Vergessen", "Eingabe" und "Ausgabe" (in der Abbildung in blau, grün und orange) legen fest, ob und inwiefern der Zustand verändert wird. Diese bestehen jeweils aus einer Sigmoid-Schicht (σ) und einer punktweisen Multiplikation (x). Die Aktivierungsfunktion Sigmoid (siehe 3.7.2) sorgt dafür, dass die Ausgabe zwischen 0 und 1 liegt. Für das Gatter bedeutet 0 komplett geschlossen und 1 komplett geöffnet.[28]

Das *Vergessen*-Gatter entscheidet anhand von x_t und h_t, welche Informationen des Gedächtnisses verworfen werden sollen. Die Ergebnisse werden mit dem aktuellen Inhalt des Gedächtnis multipliziert. Im nächsten Schritt wird entschieden, welche neuen Informationen im Gedächtnis gespeichert werden sollen. Das geschieht in zwei Teilen (*Eingabe* in Abbildung 3.7.2). Zuerst entscheidet die Schicht σ, welche Werte wie stark gewichtet werden sollen. Dann wird in der Schicht *tanh* für jeden Eintrag im Gedächtnis ein neuer Wert berechnet. Anschließend werden die Ergebnisse multipliziert. Das Gedächtnis wird dann durch eine Addition verändert.[28]

Abhängig vom Gedächtnis wird eine Ausgabe erzeugt. Die Schicht σ (in der Abbildung im Bereich *Ausgabe*) entscheidet, welche Teile des Gedächtnis für die Ausgabe verwendet werden. Die Werte aus dem Gedächtnis werden mit *tanh* auf den Wertebereich -1 bis 1 abgebildet. Die Ausgabe ergibt sich dann nach der Gewichtung mit den Ergebnissen von σ. [28]

Neben dem vorgestellten Modell, existieren zahlreiche Modifikationen und Verbesserungen für verschiedene Anwendungsbereiche.[28] Allgemein lässt sich festhalten, dass es für Probleme wie Sprachkomprimierung[29] und Handschriftenerkennung[30] LSTM aktuell die besten Ergebnisse liefert.

3.4 Kostenfunktion

Die Kostenfunktion (*engl.* Loss function) gibt die Abweichung zwischen der Ausgabe des Netzes und dem gewünschten Resultat an. Ziel ist es, diese Funktion zu minimieren.

3.4.1 Kreuzentropie

Kreuzentropie ist eine Kostenfunktion, die oft für Klassifizierungsprobleme verwendet wird.[31]

Für die prognostizierte Wahrscheinlichkeitsverteilung y und die tatsächliche Verteilung y' ist sie definiert als[32]

$$H_{y'}(y) = -\sum_i y'_i \log(y_i).$$

3.5 Lernen

Die charakteristische Eigenschaften von Neuronalen Netzen ist die Fähigkeit, aus Mustern in den eingegebenen Daten systematisch zu lernen. Nachdem ein Netz die Zusammenhänge von Ein- und Ausgabe erlernt hat, soll durch Generalisieren auch eine Lösung für beliebige Eingabewerte produziert beziehungsweise angenähert werden können.[18]

Das Lernen geschieht, indem im Training die Gewichte und Schwellenwerte der Neuronen verändert werden. Der bekannteste Algorithmus zum Trainieren von neuronalen Netzen ist die Backpropagation (oder auch Fehlerrückführung).

3.5.1 Lernverfahren

Grundsätzlich existieren drei verschiedene Lernverfahren für neuronale Netze: überwachtes (*supervised*), unüberwachtes (*unsupervised*) und bestärkendes Lernen (*reinforced learning*). Für diese Arbeit wird ein überwachtes Lernverfahren eingesetzt.

3.5.1.1 Überwachtes Lernen

Beim überwachten Lernen erhält das Netz eine Eingabe. Die zugehörige Ausgabe wir dann mit dem Wert verglichen, der tatsächlich hätte erzeugt werden sollen. Durch Vergleich der gewünschten und der tatsächlichen Ausgabe werden Änderungen an den Gewichten und Schwellwerten des Netzes vorgenommen.[33]

3.5.2 Lernrate

Die Lernrate legt fest, wie stark die Gewichte beim Training verändert werden. Die Anpassung der Gewichte wird mit Algorithmen wie der Backpropagation durchgeführt.

Die Lernrate ist essenziell für ein gutes Ergebnis eines Neuronalen Netzes. Ist sie zu gering, so ist die Gefahr hoch, dass ein lokales Minimum statt dem globalen angenähert wird. Ist die Lernrate zu hoch und die Gewichte werden in jeder Iteration sehr stark verändert, ist es möglich, dass ein globales Minimum übersprungen wird oder dass um ein globales Minimum herum oszilliert wird.[34]

Um zu vermeiden, dass eine zu extreme Lernrate gewählt wird, kann die Lernrate über verschiedene Epochen verändert werden. Ein Ansatz ist es, mit einer großen Lernrate zu beginnen, um möglichst schnell an das globale Minimum zu kommen, und diese dann zu verkleinern, um das globale Minimum nicht zu überschreiten.[34]

3.5.3 Backpropagation

Für diese Arbeit wurde ein überwachtes Lernverfahren mit dem Backpropagations-Algorithmus verwendet, weshalb hier auf diesen eingegangen wird.

Backpropagation ist ein weitverbreiteter Algorithmus für die Optimierung von Neuronalen Netzen.[35] Der Algorithmus kann in drei Schritte unterteilt werden.

1. Im ersten Schritt wird durch alle Schichten des Netzes von der Eingabe- bis zur Ausgabeschicht propagiert (oder verbreitet). Das heißt, eine Eingabe wird in das Neuronale Netz gegeben, jedes Neuron berechnet zu einer Eingabe eine entsprechende Ausgabe und am Ende gibt die letzte Schicht das Ergebnis zur Eingabe aus.

2. Im zweiten Schritt wird mittels Kostenfunktion (*engl.* Loss function) die Abweichung zwischen der Ausgabe des Netzes und dem tatsächlichen Ergebnis berechnet. Für jedes Neuron in der Ausgabeschicht ergibt sich so ein Fehler.

3. Daraufhin wird der Fehler so lange rückpropagiert, bis jedes Neuron jeder Schicht einen Fehlerwert erhalten hat.

Mit diesen Fehlerwerten wird der Gradient der Kostenfunktion berechnet. Der Gradient gibt die Richtung der größten Steigung an. Da der Fehler minimiert werden soll, wird das Vorzeichen umgedreht. Anschließend wird der Wert an die sogenannte Updatefunktion weitergereicht. Diese nimmt als Eingabe die Richtung, in die verändert werden soll, und die Lernrate entgegen und passt die Gewichte so an, dass die Kostenfunktion minimiert wird.[35]

3.5.3.1 Größe des Batches

Die Größe des Batches legt fest, wie viele Tweets beim Training auf einmal durch das Neuronale Netz propagiert werden, bevor die Gewichte der künstlichen Neuronen angepasst werden.

Je kleiner ein Batch ist, desto mehr Störfaktoren wirken sich auf die Gewichte in den künstlichen Neuronen aus. Je größer ein Batch ist, desto höher ist die Berechnungszeit für den Gradienten.[36]

3.6 Updatefunktion

Bei der Backpropagation werden die Gewichte und Schwellenwerte durch eine Updatefunktion angepasst. Ziel ist es statt einem lokalen Minimum das globale Minimum zu finden. Im Folgenden werden Stochastic Gradient Descent (SGD) und einige Modifikationen davon, die im Bereich des maschinellen Lernens häufiger verwendet werden, vorgestellt.[35]

3.6.1 Stochastic Gradient Descent

SGD ist eine Approximation des Gradientenverfahrens. Hier wird ein Näherungswert für den Gradienten bestimmt. Von diesem aus wird in Richtung des negativen Gradienten gegangen. Es wird der negative statt dem positiven Gradienten angenähert, da die Kostenfunktion minimiert und nicht maximiert werden soll. Der Gradient g wird über den Fehler des Neurons berechnet. Ein Parameter p wird mit einer Lernrate l und dem Gradienten λ wie folgt angepasst:

$$p = p - l * \lambda$$

Die Lernrate legt fest, um welchen Faktor das Gewicht verändert werden soll.

3.6.2 Adadelta

Adadelta ist eine Updatefunktion, bei der die Lernrate adaptiv bestimmt wird. Daher muss diese nicht mehr manuell angepasst werden.[37]

3.6.3 Root Mean Square Propagation

Bei Root Mean Square Propagation (RMSProp) wird die Lernrate für jeden Parameter individuell angepasst. Um die nächste Lernrate zu berechnen, wird das gleitende Mittel der Größen der letzten Gradienten für jedes Gewicht berechnet. Die Lernrate wird dann durch diesen Wert dividiert.[35]

Im ersten Schritt wird das gleitende Mittel $E[g^2]_t$ berechnet. Sei γ der Prozentwert, der bestimmt wie viel Information verworfen wird und g der Gradient. Dann ist das gleitende Mittel für den Zeitpunkt t wie folgt definiert:

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1 - \gamma) g_t^2$$

Damit ergibt sich die Updatefunktion θ:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

3.6.4 Adam

Adaptive Moment Estimation (Adam) ist ein Verfahren, um für jeden Parameter adaptiv Lernraten zu berechnen. Das Verfahren speichert den Durchschnitt m_t von den vorherigen Gradienten und den Durchschnitt v_t von den vorherigen quadrierten Gradienten.[35] Diese sind Schätzungen des ersten und zweiten Momentums.

Das Momentum ist die Summe der vorherigen Gradienten, die in einem Vektor gespeichert wird. Dieser Vektor beschleunigt das Gradientenverfahren, wenn sich der Gradient nur wenig verändert. Wenn der Gradient stark oszilliert, verlangsamt der Momentums-Vektor jedoch das Gradientenverfahren.[38] Die Variablen β_1, β_2 geben an, um wie viel die Parameter jeweils abklingen sollen:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t,$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2.$$

Da m_t und v_t als Nullvektoren initialisiert werden, tendieren diese gegen Null. Um das zu verhindern, wird das erste und zweite Momentum wie folgt abgeschätzt:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t},$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}.$$

Die Updatefunktion θ ergibt sich dann als[38]

$$\theta_{t+1} = w_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t.$$

[35]

3.6.5 Nadam

Die bereits vorgestellte Funktion Adam verwendet Momentum. Nesterov Adam Optimizer (Nadam) ist eine Erweiterung dazu, die stattdessen Nesterov-Momentum verwendet.

3.7 Aktivierungsfunktion

Die Aktivierungsfunktion entscheidet über die Form der Ausgabe eines künstlichen Neurons. Die Funktion ist nicht linear, was erst zur Nichtlinearität eines Neuronalen Netzes führt.3.1 Die Ausgabe kann in unterschiedlichen Wertebereichen liegen und zum Beispiel alle reellen Zahlen enthalten oder alle Zahlen im Intervall $[0, 1]$. Vor allem für die Ausgabeschicht spielt die Aktivierungsfunktion eine große Rolle, da sie das Format der Ausgabe festlegt.

3.7.1 ReLU

Die Aktivierungsfunktion ReLU ist definiert als

$$g(x) = \max(0, x).$$

Die Funktion ist für eine Eingabe x null, falls $x \leq 0$ und entspricht ansonsten x. Als einfacher Schwellwert ist ReLu die simpelste der hier vorgestellten Aktivierungsfunktionen, was Vorteile für die Dauer des Trainings eines Neuronalen Netzes mit sich bringt.

3.7.2 Sigmoid

Die Funktion Sigmoid ist definiert als

$$g(x_i) = \frac{1}{1 + e^{-x_i}}.$$

Mathematisch betrachtet nimmt die Funktion einen reellen Wert entgegen und gibt einen Wert im Intervall $[0, 1]$ zurück.

3.7.3 Softmax

Für einen Vektor der Größe k sei Softmax definiert als

$$g(x_i) = \frac{e^{x_i}}{\sum_{j=0}^{k} e^{x_j}}.$$

Die Funktion berechnet die Wahrscheinlichkeit für jede Klasse eines Klassifizierungsproblems.

Wie bei Sigmoid liegt die Ausgabe im Intervall von $[0, 1]$. Die Summe aller Wahrscheinlichkeiten ergibt immer 1. Bei Klassifikationsproblemen enthält die Ausgabeschicht mit Aktivierungsfunktion Softmax so viele Neuronen wie Klassen bestimmt werden sollen. Der ausgegebene Wert gibt für die Eingabe an, mit welcher Wahrscheinlichkeit sie in der angegebenen Klasse liegt.

3.7.4 tanh

Die Ausgabe der tanh-Aktivierungsfunktion liegt im Intervall zwischen $[-1, 1]$.

$$\tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$

Die Funktion hat in der Nähe von null und eins einen flachen Verlauf. Werden viele kleine Werte multipliziert, um den Gradienten zu berechnen, spricht man vom *Vanishing Gradient*-Problem, was bei tanh auftreten kann.[26]

3.8 Regularisierung

In Neuronalen Netzen mit vielen Parametern ist *Overfitting* ein häufiges Problem. Mit Regularisierung wird versucht das Problem zu lösen.

3.8.1 Dropout

Eine Möglichkeit der Regularisierung sind Dropout-Schichten. Diese sorgen dafür, dass eine zufällig bestimmter Prozentsatz der Neuronen ausgeschaltet wird und im nächsten Berechnungsschritt nicht berücksichtigt wird. Dadurch wird verhindert, dass die Neuronen sich zu stark an die Trainingsdaten anpassen und dadurch nicht mehr generalisieren können.[39]

Auf wie viel Prozent der Neuronen der Dropout angewendet wird, muss anhand der Ergebnisse des Netzes entschieden werden. Ist der Prozentsatz zu klein, kann das Netz zu stark auf die Trainingsdaten abgestimmt werden und erzielt dadurch schlechtere Ergebnisse auf den Testdaten. Wenn der Prozentsatz des Dropouts zu groß ist, verschlechtern sich die Ergebnisse auf den Trainings- und den Testdaten.

3.8.2 L2-Regularisierung

Bei der L2-Regularisierung wird der Kostenfunktion der sogenannten Regularisierungsterm hinzugefügt. Dieser besteht aus der Summe der quadrierten Gewichte des gesamten Netzes multipliziert mit $\frac{\lambda}{2n}$. Dabei ist λ der Regularisierungsparameter und n die Größe des Trainingsdatensatzes.[40]

Für die Kreuzentropie ergibt sich dadurch folgende Gleichung:

$$H_{y'}(y) = -\sum_i y'_i \log(y_i) + \frac{\lambda}{2n} \sum_w w^2.$$

4. Implementierung und Methodik

4.1 Motivation

Ziel dieser Arbeit ist es, mit Hilfe von Tweets vorherzusagen, ob der Aktienkurs eines Unternehmens an einem Folgtetag fällt oder steigt. Dazu wurden zwei verschiedene Ansätze verwendet.
Bei der Prädiktion mittels Sentimentanalyse wird mit einer bestehenden Anwendung bestimmt, wie positiv oder negativ die Aussage eines Tweets ist. Mit dieser Information wird überprüft, ob es einen Zusammenhang zwischen dem durchschnittlichen Sentiment an einem Tag und dem Verlauf des Aktienkurs am nächsten Tag gibt.
Bei der direkten Prädikation wird der Datensatz in einen Trainings- und Testdatensatz aufgeteilt. Mit den Zusammenhängen aus Tweets und Aktienkursen im Trainingsdatensatz wird versucht, mit den Tweets im Testdatensatz auf die Kurse in diesem Zeitraum zu schließen.

Die anfängliche Implementierungsarbeit bestand daraus, Daten zu sammeln. Das Vorgehen dazu wird im Abschnitt *Daten* erklärt.

4.2 Prädiktion mittels Sentimentanalyse

Mit dem Werkzeug Stanford Natural Language Processing (SNLP) lassen sich umfangreiche Analysen auf Texten ausführen. SNLP allgemein unterstützt mehrere Sprachen, die Sentimentanalyse ist jedoch nur in englischer Sprache verfügbar.[41]

Für jeden Tweet in englischer Sprache wurde mit Hilfe eines Recursive Neural Tensor Network (RNTN) ein Sentiment errechnet. Dieses reicht von sehr negativ über negativ, neutral, positiv bis sehr positiv.

Das RNTN nimmt Sätze variabler Länge als Eingabe entgegen. Hier werden Sätze immer zusammenhängend betrachtet. Daraus ergeben sich einige Vorteile im Gegensatz zur Betrachtung einzelner Wörter. Zum Beispiel kann erkannt werden, auf was sich Negationen beziehen, was bei *Bag-of-Words*-Ansätzen oft ein Problem darstellt.[42] Ein *Bag-of-Words* ist eine Repräsentation eines Textes, bei der Grammatik und Reihenfolge der Wörter ignoriert werden. Ein Text besteht in dieser Repräsentation aus der Menge der enthaltenen Wörter mit der Information, wie oft ein Wort vorkommt.[43]

Während *Bag-of-Words*-Ansätze bei längeren Dokumenten vor allem durch Wörter mit starkem Sentiment wie ”Super” oder ”Fantastisch” gute Ergebnisse erzielen, so erzielt der

Ansatz nur begrenzt gute Ergebnisse, wenn es darum geht, kürzere Texte (wie zum Beispiel Tweets) in mehr als eine Klasse einzuteilen. Hier können mit dem RNTN deutlich bessere Ergebnisse erzielt werden.[44]

Das Training für das RNTN basiert auf einem Datensatz der "Stanford Sentiment Treebank", einem komplett-annotierten Syntaxbaum. Dieser baut auf Nutzerbewertungen zu Filmen von der Plattform Rotten Tomatoes auf.[45] Der Datensatz wurde von drei Personen manuell annotiert.

4.2.1 Aufbau des Netzes

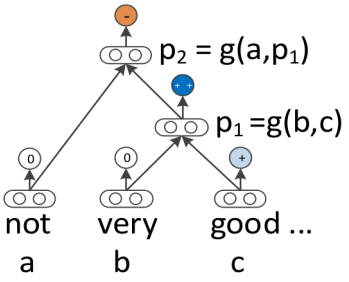

Abbildung 4.1: Text als Binärbaum repräsentiert

Mit einem rekursiven Neuronenmodell (*engl.* recursive neural model) lassen sich Texte variabler Länge mit Hilfe von Vektoren darstellen. Dafür wird die Eingabe in einen Binärbaum gewandelt. Jedes Blatt des Baumes ist ein Vektor, der ein Wort in der Eingabe repräsentiert. Mit Hilfe verschiedener Kompositionen g werden von den Blättern aus die Elternknoten berechnet. Das Berechnen der Vektoren $p_i \in \mathbb{R}^d$ für Vektoren der Länge d mit einem Rekurrente Neuronalen Netz (RNN) funktioniert für das Beispiel in Abbildung 4.1 wie folgt:

$$p_1 = f\left(W \begin{bmatrix} b \\ c \end{bmatrix}\right), \; p_2 = f\left(W \begin{bmatrix} a \\ p_1 \end{bmatrix}\right), \tag{4.1}$$

dabei sei f die Aktivierungsfunktion *tanh* und $W \in \mathbb{R}^{d \times 2d}$ die Gewichte, die trainiert werden sollen.

Bei einem rekurrenten Neuronalen Netz (RNN) können sich Eingabevektoren nur implizit über die nicht-lineare Aktivierungsfunktion austauschen. Um die Interaktion zwischen den Eingabevektoren zu erhöhen, verwendet das RNTN die gleiche Komposition für alle Knoten im Binärbaum.

Sei $V^{[1:d]} \in \mathbb{R}^{2d \times 2d \times d}$ ein Tensor, der mehrere Bilinearformen definiert, dann lässt sich das Produkt zweier Tensoren $h \in \mathbb{R}^d$ wie folgt berechnen:

$$h_i = \begin{bmatrix} b \\ c \end{bmatrix}^T V^{[1:d]} \begin{bmatrix} b \\ c \end{bmatrix}. \tag{4.2}$$

Damit berechnet das RNTN p_1 wie folgt:

$$p_1 = f\left(\begin{bmatrix} b \\ c \end{bmatrix}^T V^{[1:d]} \begin{bmatrix} b \\ c \end{bmatrix} + W \begin{bmatrix} b \\ c \end{bmatrix}\right). \tag{4.3}$$

Der zweite Elternvektor in Abbildung 4.1 wird mit den gleichen Gewichten bestimmt:

$$p_2 = f\left(\begin{bmatrix} a \\ p_1 \end{bmatrix}^T V^{[1:d]} \begin{bmatrix} a \\ p_1 \end{bmatrix} + W \begin{bmatrix} a \\ p_1 \end{bmatrix}\right). \tag{4.4}$$

Anhand der Formeln lässt sich erkennen, dass das RNN ein Spezialfall des RNTN ist, bei dem $V = 0$ ist. Im Gegensatz zum vorher vorgestellten rekursiven Modell kann so ein Zusammenhang zwischen Eingabevektoren hergestellt werden.

4.2.2 Transformation für die Prädiktion der Aktienkurse

Die Sentimentanalyse teilt jeden Tweet in eine der fünf Klassen ein: sehr negativ (0), negativ (1), (2), positiv (3) und sehr positiv (4). Um zu bestimmen, ob der Aktienkurs fällt oder steigt, wird dieses Ergebnis auf zwei Klassen reduziert: Kurs steigt (0) und Kurs fällt (1). Dafür wird der Grenzwert g gebildet, für den die meisten Prädiktionen im Trainingsdatensatz korrekt sind. Damit lässt sich jeder Tweet t mit der Funktion $b(t)$ in einer der beiden Klassen einteilen:

$$b(t) = \begin{cases} 0, & \text{falls } \mathit{Sentiment} \geq g \\ 1, & \text{sonst} \end{cases}.$$

4.3 Direkte Prädiktion

Ein Ansatz, um die Prädiktion mittels Sentimentanalyse zu verbessern, wäre es, das RNTN mit Tweets zu trainieren, anstelle von Filmbewertungen. Dafür müssten jedoch die Satzbestandteile der Tweets manuell annotiert werden. Deshalb werden im Folgenden drei Neuronale Netze vorgestellt, die sich mit der Information trainieren lassen, ob der Aktienkurs des erwähnten Unternehmens am nächsten Tage gestiegen oder gefallen ist. Dadurch wird eine direkte Prädiktion des Aktienkurses möglich.

Bei dem ersten neuronalen Netz handelt es sich um ein rudimentäres Feedforward-Netz mit einer Max-Pooling und zwei vollvernetzten Schichten. Das zweite Netz ist mit einer Convolutional-Schicht ein komplexeres Feedforward-Netz. Bei dem letzten Netz handelt es sich um ein Rekurrentes Neuronales Netz (RNN) mit einer LSTM-Schicht.

Die Wörten in den Tweets werden mit Word Embeddings repräsentiert, was im Folgenden Abschnitt näher erläutert wird. Darauf folgt eine Übersicht über den Aufbau der Netze und ein anschließender Überblick über die Hyperparameter, die gewählt wurden, um die Ergebnisse der drei Neuronalen Netze möglichst zu maximieren.

4.3.1 Word Embeddings

Wörter lassen sich für die Berechnungen in Neuronalen Netzen als Vektoren von reellen Zahlen darstellen. Diese Art von Darstellung nennt sich auch *Word Embeddings*.

Mathematisch betrachtet handelt es sich dabei um eine Einbettung, also eine Abbildung, bei der ein Objekt als Teil eines anderen Objekts aufgefasst werden kann. Die Abbildung geht von einem Raum mit einer Dimension pro Wort auf einen kontinuierlichen Vektorraum mit wesentlich geringerer Dimension.

Mit Word Embeddings ist es möglich, semantische Bedeutungen in einen geometrischen Raum zu übertragen. Die Vektoren für einzelne Wörter werden so gewählt, dass der Abstand zwischen zwei Vektoren eine Aussage über die Semantik der beiden Wörter darstellen kann.[46]

An einem Beispiel lassen sich diese semantische Zusammenhänge wie folgt erklären. Sei $v(x)$ der zugehörige Vektor für das Wort x, dann gilt[47]

$$v(Paris) - v(Frankreich) \approx v(Deutschland) - v(Berlin)$$

Neben der Möglichkeit Word Embeddings in der ersten Schicht zu erstellen, lassen sich auch bereits trainierte Vektoren verwenden. Die bereits trainierten Vektoren, die für diese Arbeit verwendet wurden, wurden mit dem Global Vectors for Word Representation (GloVe)-Algorithmus erstellt.

4.3.1.1 GloVe

GloVe ist ein Algorithums, mit dem sich Wörter als Vektoren repräsentieren lassen.[48]

Die verwendeten Vektoren haben eine Dimension von 100 und stammen aus einem Datensatz mit 400.000 Wörter aus der englischen Wikipedia.[49] Aus dem Datensatz mit den bereits trainierten Embeddings wird ein Wörterbuch W_{GloVe} erstellt, das für jedes Wort den zugehörigen Vektor $w \in \mathbb{R}^{1\times d}$ enthält. Aus den Eingabedaten wir ein Wörterbuch $W_{Eingabe}$ mit den am häufigsten vorkommenden Wörtern erstellt. Jedes Wort wird dabei auf einen der Indizes $1, ..., \mid W_{Eingabe} \mid$ abgebildet. Mit Hilfe dieser beider Wörterbücher lässt sich eine Embedding-Matrix $E \in \mathbb{R}^{d\times|W_{Eingabe}|}$ erstellen. Jede Spalte in E steht für ein Wort aus $w_{Eingabe}$ der Eingabedaten. Kommt dieses Wort in w_{GloVe} vor, enthält die Zeile den entsprechenden Vektor, falls nicht, enthält die Zeile den Nullvektor mit der Länge 100.

4.3.2 Aufbau der Neuronalen Netze

4.3.2.1 Max Pooling-Netz

Das erste Neuronale Netz für die direkte Prädiktion ist ein Feedorward-Netz bestehend aus Max Pooling- und vollvernetzten Schichten. Im Folgenden wird näher auf dessen Aufbau eingegangen.

Auf die Embedding-Schicht folgt eine eindimensionale Max Pooling-Schicht mit Fenstergröße vier. Damit wird versucht, aus der Embedding-Matrix Störfaktoren zu entfernen.

Nach einer Dropout und einer Flatten-Schicht folgen zwei vollvernetzte Schichten. Zwischen diesen liegt wieder eine Dropout-Schicht. In der ersten vollvernetzten Schicht befinden sich 128 Neuronen. Die zweite vollvernetzte Schicht ist die Ausgabeschicht des Netzes. Sie besitzt zwei Neuronen, die durch die Aktivierungsfunktion Softmax die Wahrscheinlichkeit für jede der zwei Klassen angeben.

4.3.2.2 Convolution-Netz

Das Convolutional-Netz ist einer Erweiterung des zuvor vorgestellten Max Pooling-Netzes. Durch eine weitere Schicht wird die Komplexität des Netzes erhöht, was es im Idealfall ermöglicht, Muster in den Trainingsdaten zu erkennen, die das Max Pooling-Netz noch nicht erkannt hat.

Auf die Convolutional-Schicht folgt eine Pooling-Schicht. Das Hauptziel der Convolutional-Schicht ist es, sich wiederholende Muster in den Trainingsdaten zu erkennen. Die Faltoperation (*engl.* Convolution) wendet hierzu einen Filter auf die Eingabematrix an. Anstelle eines einzelnen Filters können auch parallel mehrere Filter, von denen jeder eine Feature Map erzeugt, angewandt werden. Diese ergeben wiederum eine Feature Map Matrix.

Auf die Ausgabe der Convolutional-Schicht wird elementweise eine Aktivierungsfunktion angewandt. Nach dieser Funktion, werden die Daten als Eingabe in die Pooling-Schicht

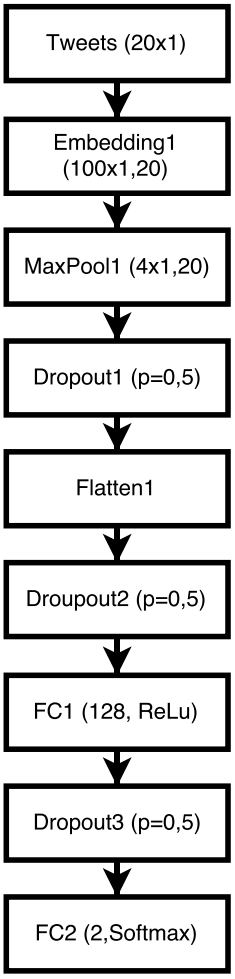

Abbildung 4.2: Aufbau des Max-Pooling-Netzes

gegeben. Hauptziel dieser Schicht ist es, Daten zusammenzufassen und den Umfang der Daten zu verkleinern. Genauer wird hier das Verfahren Max-Pooling angewandt.

Nach der Convolutional- und Pooling-Schicht folgt eine Reshape-Schicht gefolgt von einer vollvernetzten Softmax-Schicht. Diese berechnet jeweils die Wahrscheinlichkeit, dass der Aktienkurs am nächsten Tag fällt oder steigt.

Im Bereich des maschinellen Lernen spricht man von ”Feature Extraction”, wenn es darum geht, aus Daten Informationen (Features) abzuleiten, die nicht redundant sind. Diese Vereinfachung kommen den folgenden Lerniterationen und der Gerneralisierung zu Gute. Die Convolutional-Schicht in Verbindung mit der Aktivierungsfunktion und der Pooling-Schicht stellt einen nicht-linearen Feature Extractor dar. Dadurch kann das Netz aus den Eingabedaten eine umfangreiche Repräsentation in Form von Features erstellen.[50]

4.3.2.3 LSTM-Netz

Statt einer Convolutional-Schicht enthält dieses Netz eine LSTM-Schicht. Damit lassen sich auch Zusammenhänge erkennen, die zeitlich weiter auseinanderliegen.

Alle Eingabedaten wurden auch hier auf die gleiche Länge gebracht. Alternativ wäre es auch möglich, Daten unterschiedlicher Länge an das LSTM zu geben.

Die rekurrente LSTM-Schicht besitzt ein ”Gedächtnis” der Größe 128. Diese entscheidet auch über die Ausgabedimensionalität. Nach einer Dropout- folgt die erste vollvernetz-

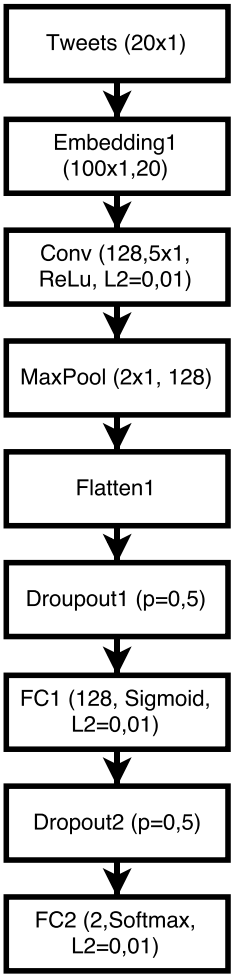

Abbildung 4.3: Aufbau des Convolution-Netzes

te Schicht mit Sigmoid-Aktivierungsfunktion. Wie bei den vorherigen beiden Netzen besitzt diese 128 Neuronen. Nach einer Dropout-Schicht folgt dann wieder eine vollvernetzte Schicht mit Softmax-Aktivierungsfunktion.

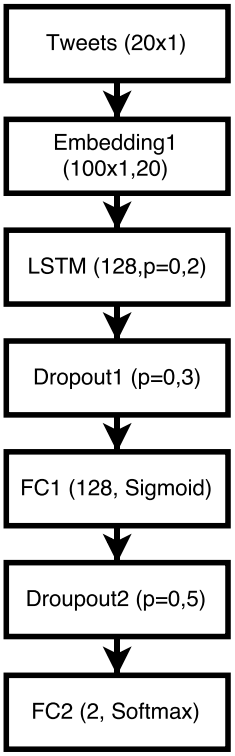

Abbildung 4.4: Aufbau des LSTM-Netzes

4.4 Daten

4.4.1 Tweets

Mit über 300 Millionen aktiven Nutzern ist Twitter eine der größten Informationsquellen im Internet.[51] Wegen der hohen Anzahl an Nutzern und der Vielzahl an Nachrichten wurde die Plattform benutzt, um für diese Arbeit Stimmungsbilder für verschiedene Unternehmen erstellen zu können.

Über den Zeitraum vom 21. Dezember 2016 bis zum 3. Februar 2017 wurden von der Plattform circa 20 Millionen Tweets gesammelt. Twitter stellt dafür die sogenannten *Stream API* zur Verfügung. Damit lässt sich über einen HTTP-Request nach Tweets suchen, die verschiedene Schlagwörter enthalten. In Echtzeit werden dann entsprechende Tweets zurückgegeben. Alternativ stellt Twitter die Serach API zur Verfügung. Diese liefert Tweets der letzten sieben Tage. Die Auswahl dieser ist eher auf Relevanz als auf Vollständigkeit ausgelegt. Eine offizielle API die Tweets liefert, die noch weiter zurück liegen, existiert jedoch nicht.[52]
Jede Antwort auf die Anfrage beinhaltet neben dem Text mit maximal 140 Zeichen weitere Informationen. Dazu zählt der Name des Autors, die Anzahl der Personen, die ihm folgen, und ein Zeitstempel. Außerdem wird optional der Standort des Autors geliefert, falls er diesen aktiviert hat.

Die Anfragen wurden über ein Java-Programm gestellt. Dieses enthält eine Liste von allen DOW30- und DAX30-Unternehmen. Diese werden mit der Anfrage an die API übergeben. Groß-/Kleinschreibung spielt bei den Anfragen keine Rolle, weshalb die Antworten zusätzlich gefiltert wurden. So wird zum Beispiel erreicht, dass das Symbol ”BEI” für die Beiersdorf AG auch nur in Großbuchstaben vorkommt und nicht zum Beispiel als deutsches Wort ”bei”. Weiter muss das übergebene Wort nicht alleinstehend sein. Deshalb entfernt

das Programm auch alle Tweets, die auf die Suchanfrage zu "Bayer" zutreffen, aber tatsächlich zum Beispiel das Wort "Bayern" beinhalten.

Die Aktienunternehmen können in den Tweets in einer von drei Schreibweisen vorkommen. Zum einen, indem der Name eines der 60 Unternehmen enthalten ist ($Name$). Außerdem kann das Aktiensymbol, über das sich das Unternehmen an der jeweiligen Börse eindeutig identifizieren lässt, im Tweet vorkommen ($Symbol$). Oder es wird ein Dollarzeichen, gefolgt von dem jeweiligen Aktiensymbol, erwähnt ($\$Symbol$). Letztere Erwähnung wird üblicherweise auf Twitter benutzt, wenn sich über Aktienkurse ausgetauscht wird.[53].

Ein Vorteil der $\$Symbol$-Tweets ist es, dass mit sehr hoher Wahrscheinlichkeit die Rede von dem Unternehmen ist, während ohne das Dollarzeichen viele andere Bedeutungen möglich sind. Ein Nachteil ist, dass im Vergleich zu den anderen Schreibweisen deutlich weniger Tweets geschrieben werden. Am 10. Januar waren für das Symbol "MMM" zum Beispiel über 78.000 Tweets gesammelt, von denen lediglich 685 die Zeichenkette "$MMM" enthielten.

Für das Trainieren und Überprüfen der Neuronalen Netze zur direkten Prädiktion und für das Testen der Prädiktion mittels Sentimentanalyse wurden zwei gleich große, disjunkte Datensätze verwendet. In jedem Datensatz gehört eine Hälfte der Tweets zur Klasse "Kurs steigt" (0) und die andere zu "Kurs fällt" (1). Da die Klassen im ursprünglichen Datensatz ungleich groß sind, wurden aus der größeren Klasse mit einem Pseudo-Zufallsverfahren so viele Tweets entnommen, dass beide Klassen gleich stark vertreten sind.

Zudem wurden alle Texte auf die gleiche Länge gebracht. Der längste Tweet im Datensatz enthält 70 durch Leerzeichen getrennte Zeichen. Da aber nur wenige Randfälle mehr als 20 Wörter enthalten, wurde die maximale Länge der als Zahlen repräsentierten Wörter auf 20 beschränkt. An kürzere Zeichenketten wurde so oft die Zahl Null vorne angehängt, bis dieser aus 20 Zahlen bestand, längere Texte wurden abgeschnitten. Die Länge 20 im Vergleich zu 70 brachte keine Einbußen bei der *Accuracy* mit sich, beschleunigte jedoch das Training um ein Vielfaches.

Die Repräsentation von Wörtern als Zahlen sei an einem Beispiel wie folgt erklärt: Angenommen das englische Wort "I" käme in allen Texten am häufigsten vor und das Wort "am" am dritthäufigsten. Wenn ein Wörterbuch mit den 20.000 am häufigsten auftretenden Wörtern gewählt wird, dann sähe der Tweet "I am" in Ganzzahlen übersetzt zum Beispiel so aus: "0 0 0 ... 0 20000 19998" und bestände aus 20 Zahlen.

4.4.2 Aktienkurse

Über die Yahoo Finance API[54] lassen sich Anfragen mit Symbolen und einem Zeitraum machen. Auf diese Anfrage wird mit einer CSV-Datei geantwortet, die für das Unternehmen in dem gegebenen Zeitraum Informationen zum Aktienkurs liefert. Die Datei enthält für jeden Tag, an dem die Börse geöffnet war, verschiedene Informationen: Kurs zur Öffnung und Schluss der Börse und die Höchst- und Tiefstwerte des Tages.
Diese Daten wurden nach dem Auslesen (wie die Twitter-Daten) in einer Datei im JSON-Format abgespeichert.

Jeder Tweet wird mit Hilfe der Daten in zwei Klassen eingeteilt: Kurs fällt und Kurs steigt. Das geschieht indem der Schlusskurs des Tages, an dem der Tweet verfasst wurde, mit dem Schlusskurs eines späteren Tages verglichen wird. Überprüft wurden die Abstände von ein bis vier Tagen. Angenommen das Zeitfenster ist eins und ein Tweet wurde am Freitag erstellt. Da die Börse am Samstag und Sonntag geschlossen ist, wurde der Tweet für die Prognose vom nächsten Börsentag genutzt (in diesem Fall Montag), statt ihn nicht in die Prognose mit einfließen zu lassen.

5. Evaluation

Die Methoden zur direkten Prädiktion und der Prädiktion mittels Sentiment dienen als binäre Klassifikatoren, die jeden Tweet in eine der zwei Klassen "Kurs steigt" oder "Kurs fällt" einordnen. Aus der relativen Häufigkeit der Fehler, bei denen ein Tweet in die falsche Klasse eingeordnet wurde, lassen sich quantitative Maße ableiten. Diese ermöglichen es, die Netze zu vergleichen und zu beurteilen.

Die Ausgabe der Neuronalen Netze lässt sich in vier Kategorien einteilen:

- Richtig positiv (RP): der steigende Kurs wurde korrekt bestimmt
- Falsch positiv (FP): vorhergesagt wurde ein steigender Kurs, dieser fällt jedoch
- Richtig negativ (RN): Kurs wurde korrekt als fallend bestimmt
- Falsch negativ (FN): Kurs wurde als fallend bestimmt, tatsächlich steigt er

Mit diesen Informationen über die Ausgabe der neuronalen Netze zu den Testdaten lässt sich die Accuracy (auch Korrektsklassifikationsrate) berechnen:

$$Accuracy = \frac{RP + RN}{RP + RN + FP + FN}$$

Die Trainings- und Testdaten enthalten für jede Klasse gleich viele Tweets. So wird vermieden, dass ein Großteil der Eingaben zu einer Klasse gehört und ein Neuronales Netz daraus lernt, generell viel in diese Klasse einzuordnen.

Sind beide Klassen gleich stark vertreten, ist es zum Beispiel dennoch möglich, dass 70% der Tweets über Microsoft zur Klasse der fallenden Kurse gehören und 70% der Tweets über Apple zu der Klasse der steigenden Kurse gehören. Teilt ein Klassifizierer alle Tweets über Mircrosoft in die Klasse der fallenden Kurse ein und alle Apple-Tweets in die Klasse der steigende Kurse, erzielt der eine Accuracy von 70%. Daher sollte vor allem bei einer hohen Accuracy überprüft werden, auf welchen genauen Daten das Ergebnis zu Stande kam. Deshalb werden im Folgenden weitere quantitaive Maße eingeführt.

Die Precision (auch positiver Vorhersagewert oder Relevanz genannt) gibt an, wie viele der positiv klassifizierten Ergebnisse korrekt als positiv klassifiziert wurden. Für diese Arbeit entspricht das dem Anteil der als steigend klassifizierten Tweets unter allen als positiv klassifizierten Tweets.

$$Precision = \frac{RP}{RP + FP}$$

Wenn keine *falsch-positiven (FP)* Entscheidungen getroffen wurden, sei $Precision = 1$.

Der Recall (auch Richtig-positiv-Rate, Sensitivität oder Trefferquote) gibt den Anteil der korrekt als Kurs steigend (TP) klassifizierte Tweets von allen Tweets, die tatsächlich zur Klasse der steigenden Kurse gehören, an:

$$Recall = \frac{RP}{RP + FN}$$

Für den Fall, dass keine *richtig-positiven (RP)* Ergebnisse erzielt wurden, sei $Recall = 1$.

Um die Netze anhand eines Werts vergleichen zu können, wird das kombinierte Maß $F1$ verwendet. $F1$ bildet das harmonische Mittel von Precision und Recall.

$$F1 = 2 * \frac{1}{\frac{1}{Recall} + \frac{1}{Precision}} = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Für den Randfall, dass der Nenner Null wird, sei $F1 = 1$.

Alle Tweets wurden in drei Datensätze aufgeteilt:

- *Symbol* enthält Tweets, die das Börsensymbol eines Unternehmens erwähnen
- Tweets in $\$Symbol$ erwähnen das Symbol mit vorgestelltem Dollarzeichen
- In *Name* sind alle Tweets enthalten, die eines der Aktienunternehmen mit dem Unternehmensname erwähnen.

Da die $\$Symbol$-Tweets anfänglich die besten Resultate lieferten, wurden die Optimierungen auf diesem Datensatz vorgenommen. Dieser wird im Folgenden für alle Neuronalen Netze im Detail betrachtet und anschließend mit den restlichen Datensätzen verglichen.

Jeder Datensatz wurde in zwei gleichgroße, disjunkte Trainings- und Testdatensätze aufgeteilt. Die Ergebnisse im Folgenden beziehen sich immer auf den Testdatensatz. Die Neuronalen Netze zur direkten Prädiktion wurden auf dem Trainingsdatensatz trainiert.

5.1 Prädiktion mittels Sentimmentanalyse

Im Durchschnitt ist das Sentiment der Tweets negativ, wenn diese mit dem RNTN analysiert werden. Dafür kann es mehrere Gründe geben. Zum einen ist das Neuronale Netz mit Filmbewertungen trainiert, was sich thematisch nur wenig mit Tweets über Aktienunternehmen überschneidet. Zum anderen sind Tweets oft nicht grammatisch korrekt und enthalten viele Abkürzungen, wofür das Werkzeug ebenfalls nicht trainiert ist.

Um die Prädiktion mittels Sentimentanalyse mit der direkten Prädiktion vergleichen zu können, wurde eine einfache Anlagestrategie zur Evaluation eingesetzt. Für die Sentiments der Tweets wurde ein Grenzwert bestimmt. Liegt das Sentiment unter diesem Grenzwert, ist die Aussage eher negativ und die Prognose lautet "Kurs fällt". Ist das Sentiment größer oder gleich wie der Grenzwert, ist die Aussage positiv und es wird ein steigender Kurs prädiziert.

Sei ein positiver Tweet über ein Unternehmen ein Indikator für einen steigenden Aktienkurs am nächsten Tag und ein negativer Tweet ein Indikator für einen fallenden Kurs, so erreicht

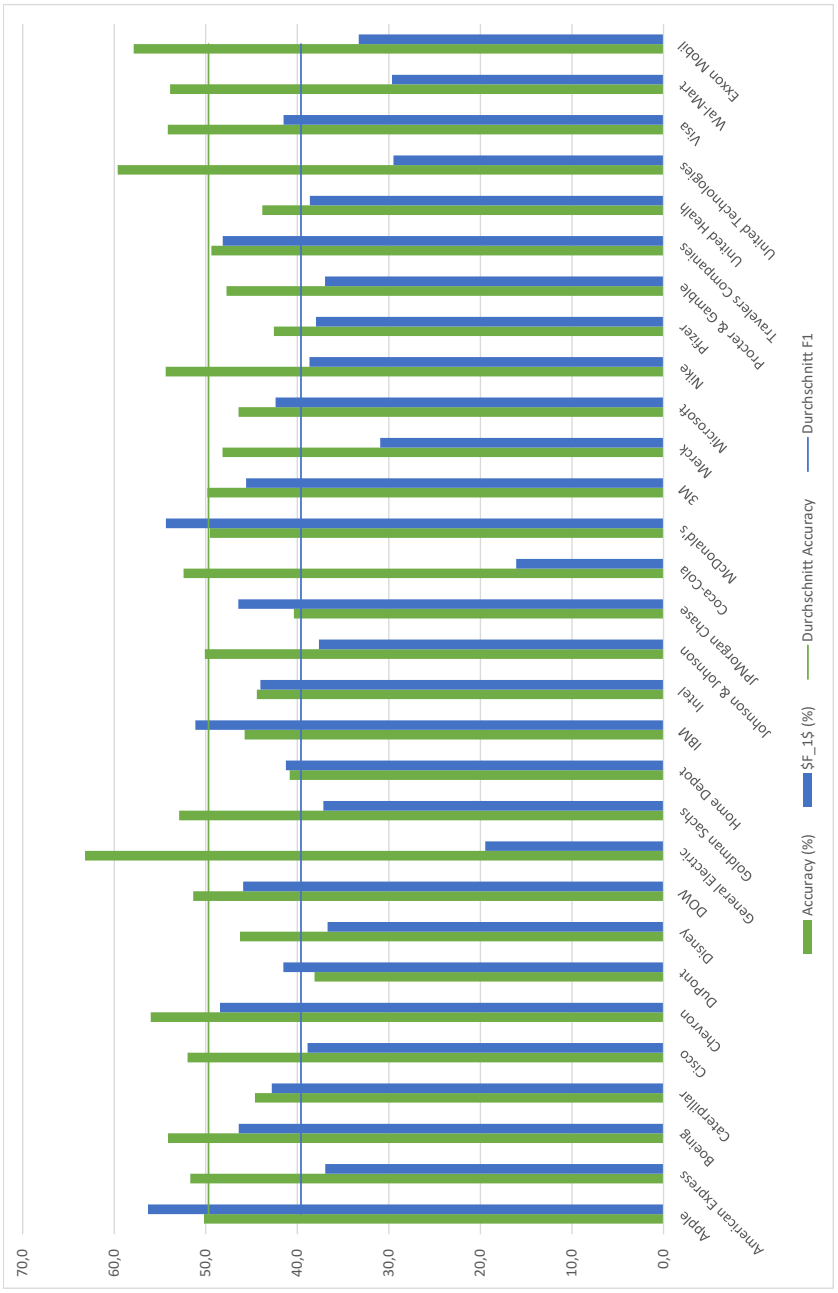

Abbildung 5.1: Accuracy und F1 pro Unternehmen für das RNTN

die Vorhersage auf den gesammelten Testdaten zu $*Symbol* vom 11. Januar bis zum 3. Februar 2017 eine *Accuracy* von 49,7% (siehe Abbildung 5.1).

Im Durchschnitt wurde die beste Accuracy mit 49,7% auf dem Datensatz $*Symbol* erzielt. Wird die Accuracy für einzelne Unternehmen betrachtet, können einige Unterschiede festgestellt werden. Während General Electric (GE) zum Beispiel mit 63% einen vergleichsweise hohen Wert erreicht, lag DuPont lediglich bei 38%. Für die Unternehmen treten beide Klassen ungefähr ausgeglichen auf und es wurden nicht übermäßig viele Tweets in eine Klasse eingeteilt. Das legt nahe, dass für den Datensatz nicht behauptet werden kann, dass positive Tweets mit einer positiven Entwicklung des Aktienkurses zusammenhängen. Obwohl für einige Unternehmen ein solcher Zusammenhang besteht, so ist für andere Unternehmen wie DuPont ein positiver Tweet im Durchschnitt eher ein Indiz für einen fallenden Kurs.

Unternehmen mit durchschnittlicher Accuracy haben nicht unbedingt einen durchschnittlichen Recall oder eine durchschnittliche Precision. Coca-Cola zum Beispiel liegt mit einer Accuracy von 52% leicht über dem Durchschnitt. Precision und Recall liegen mit 17% und 16% jedoch deutlich unter dem jeweiligen Durchschnitt. Die Erklärung dafür sind sehr wenige richtig-positive Entscheidungen, dafür aber viele richtig-negative Entscheidungen. Tatsächlich waren auch nur wenige richtig-positive Bewertungen möglich, da der Coca-Cola-Kurs im Zeitraum des Testdatensatz (11. Januar bis 3. Februar) öfter gefallen als gestiegen ist (siehe Abbildung 5.2).

Auch bei General Electric (GE) existieren große Unterschiede zwischen einzelnen Metriken. Mit 63 % ist GE das Unternehmen mit der höchsten Accuracy, besitzt aber gleichzeitig mit 20% den zweitniedrigsten Wert für F1. Hier wurden wesentlich mehr richtig-negative als richtig-positive Ergebnisse für das Unternehmen erzielt. Die Börsenschlusskurse sind bis auf zwei Tage im Zeitraum des Testdatensatzes immer gefallen.[55] Die Korrelation mit dem tendenziell negativen Sentiment ist hier vergleichsweise hoch. Gleichzeitig zeigt das Ergebnis, dass steigende Kurse nur schlecht vorhergesagt werden können.

Abbildung 5.2: Entwicklung der Coca-Cola Aktie im Testdatenzeitraum

Die Accuracy liegt im Durchschnitt sehr nahe an 50%, was auch dem Prozentsatz einer zufällig ausgewählten Entscheidung über fallende oder steigende Kurse entspricht. Das Ergebnis motivierte die *direkte Prädiktion*, mit der versucht wird, bessere Resultate zu erzielen.

5.1.0.1 Anpassung des Datensatzes

Wie eingangs bereits erwähnt, enthalten Tweets oft Störfaktoren, was die Sentimentanalyse erschwert. Das Bereinigen der Tweets lässt die Accuracy der Sentimentanalyse mit dem

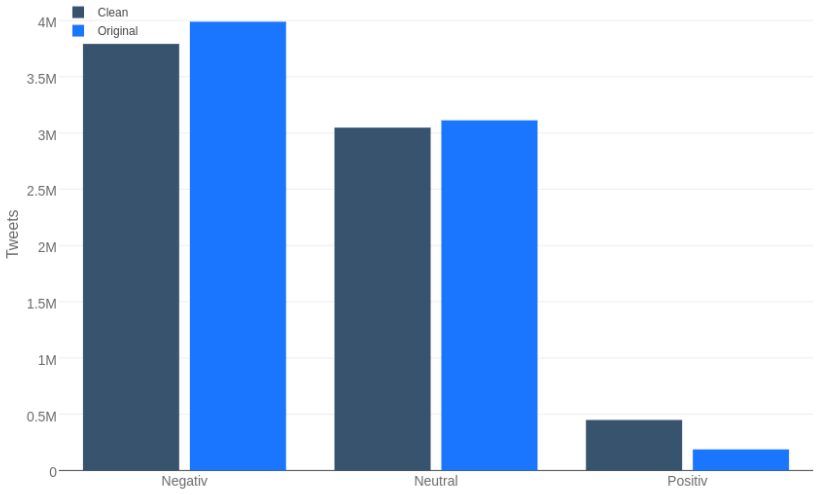

Abbildung 5.3: Sentiments vor und nach der Anpassung der Tweets

RNTN für das ”IMDB dataset” zum Beispiel um 3,9% auf 85.5% steigern.[56]

Bei den Daten dieser Arbeit wurde daher auch versucht, Störfaktoren zu beseitigen. Dabei wurden URLs, Nutzernahmen und Hashtags aus den Texten entfernt. Dies führte dazu, dass das durchschnittliche Sentiment neutraler wurde, wie in Abbildung 5.3 zu sehen ist. Die Accuracy erhöhte sich dadurch jedoch lediglich um knapp ein Prozent und die Tweets haben immer noch eine starke negative Tendenz. Mögliche Störfaktoren, die diese Tendenz auslösen können, sind Rechtschreibfehler und Umgangssprache.

5.1.1 Korrelation mit den Aktienkursen

Wie im vorherigen Kapitel bereits erwähnt wurde, teilt die Sentimentanalyse mit dem RNTN jeden Tweet in fünf Klassen von sehr negativ bis sehr positiv ein. Da jedoch nur bestimmt wird, ob der Aktienkurs am nächsten Tag fällt oder steigt, werden diese fünf Klasse auf lediglich zwei Klassen überführt. Das führt zum Informationsverlust. Um den Zusammenhang der fünf Klassen und dem Aktienkurs einzelner Unternehmen feststellen zu können, wurde mit Hilfe einer linearen Regression der Korrelationskoeffizient zwischen Sentiment und Aktienkurs berechnet. Dafür würde für jedes Unternehmen an jedem Tag ein durchschnittliches Sentiment gebildet und mit den jeweiligen Aktienkursen an dem Tag gebildet. Hier wurden die tatsächlichen Kurswerte verwendet und nicht nur die Information, ob der Kurs steigt oder fällt.

Die Aussage des Korrelationskoeffizienten soll an einem Beispiel verdeutlicht werden. Würde der Aktienkurs eines Unternehmens sich jeden Tag um den gleichen Prozentsatz wie das durchschnittliche Sentiment des Unternehmens verändern, dann wären diese mit 100% perfekt positiv korreliert. Entwickelt sich das durchschnittliche Sentiment immer um den negativen Prozentsatz in die andere Richtung, so läge die Korrelation bei -100%.

Die höchste Korrelation ergibt sich mit 7,13% für Tweets aus dem Datensatz *Symbol*. Auch hier konnten wieder große Unterschiede zwischen den einzelnen Unternehmen festgestellt werden. Beispielsweise korrelieren Tweets, die das Symbol ”AAPL” für Apple enthalten zu 51,36% für alle Tage an denen die Börse geöffnet war. Die Kurznachrichten mit den Zeichen ”KO” für Coca-Cola korrelieren dagegen stark negativ mit -25,03%. Fast keine Korrelation ergab sich bei ”UTX” (United Technologies) mit 0,46% und bei ”PFE” (Pfizer)

Tabelle 5.1: (Absolute) Korrelationen des RNTN-Sentiments mit den Aktienkursen

	$Symbol	Symbol	Name
Korrelation [%]	1,53	5,57	7,13
absolute Korrelation [%]	17,84	18,87	15,19

mit 0,18%.
Für alle gesammelten Tweets der Unternehmen des DJIA lässt sich feststellen, dass der Durchschnitt der Sentimente nur wenig mit dem Kurs des DJIA korreliert.

Die Betrachtung des Durchschnitts der absoluten Korrelationen in Tabelle 5.1 zeigt einen mehr als doppelt so großen Zusammenhang. Das heißt die Stimmung der Kurzmitteilungen ändert sich auch, wenn sich der Aktienkurs verändert. Allerdings geht die Veränderung je nach Unternehmen teilweise in die entgegengesetzte Richtung. Das heißt, wenn das durchschnittliche Sentiment steigt, fällt der Aktienkurs oder umgekehrt.

Die Korrelationen der $*Dollar*-Tweets lagen zwischen -10,1% und 7,86%. Die höchste Korrelation ergab sich für Tweets über Microsoft mit dessen Aktienkurs am Folgenden Tag mit 52,78%

5.1.1.1 Diskussion der Ergebnisse

Wie vorhergehend bereits erwähnt, kann das schlechte Ergebnis der Prädiktion mittels Sentiment damit zusammenhängen, dass das verwendete Werkzeug nicht auf Twitter-Daten trainiert wurde.

Angenommen das Sentiment wäre jedoch akkurat, so ist noch nicht gesagt, dass ein Zusammenhang zwischen einem positiven Tweet und einem steigenden Aktienkurs besteht. In der Arbeit von Bollen et. al.[57] wurde beispielsweise kein signifikanter Zusammenhang zwischen positiver Stimmung auf Twitter und dem DJIA festgestellt. Allerdings wurde ein Zusammenhang zwischen der Gelassenheit (”Calm”) in Tweets und dem Aktienindex festgestellt.

Für alle drei Datensätze liegt die absolute Korrelation zwischen Sentiment und Aktienkurs bei über 15%. Das lässt die Schlussfolgerung zu, dass Veränderungen des durchschnittlichen Sentiments mit Marktbewegungen korrelieren. Offen ist die Frage, warum für manche Unternehmen ein sehr positives Sentiment zu fallenden Kursen und für andere zu steigenden Kursen führt.

Angenommen es besteht eine signifikante Korrelation zwischen den Nachrichten im Datensatz und den Aktienkursen der Unternehmen. Dann sollte es auch Neuronale Netze geben, die diese Zusammenhänge erlernen und auf andere Texte anwenden können. Daher wird im Folgenden versucht, solche Zusammenhänge zu erkennen, ohne den Fokus auf das positive oder negative Sentiment der Tweets zu legen.

5.2 Direkte Prädiktion

5.2.1 Max Pooling-Netz

Mit dem Max Pooling-Netz konnten bessere Ergebnisse erzielt werden als mit dem RNTN. Das Netz erreichte bei der Vorhersage des folgenden Tages eine Accuracy von 55% (siehe Tabelle 8.2 im Anhang).

5.2.1.1 Optimierungen und ihre Auswrikungen

Die Ergebnisse von Neuronalen Netzen lassen sich mit einer Vielzahl von Hyperparametern beeinflussen. Im Folgenden werden einige verschiedene Optimierungen und deren Ergebnisse vorgestellt.

Word Embeddings

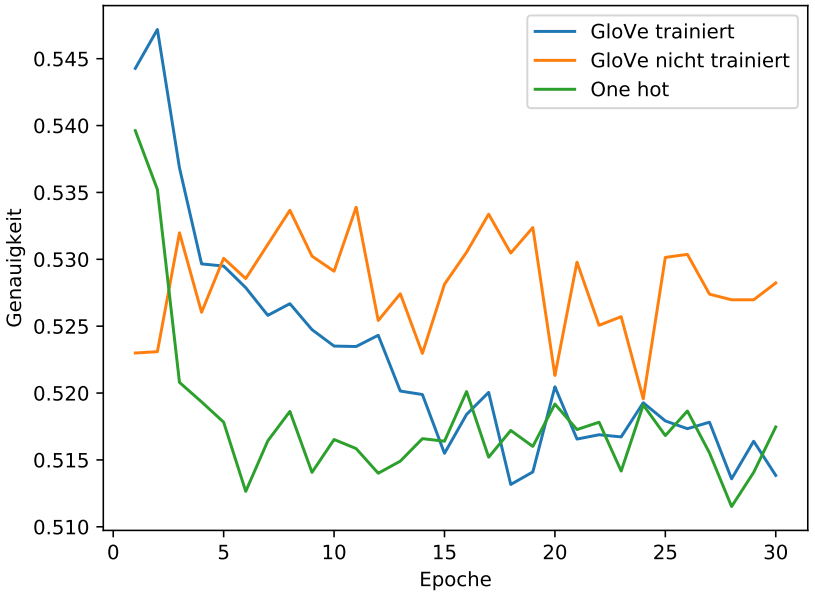

Abbildung 5.4: Unveränderte und veränderte GloVe-Embeddings im Vergleich mit Embeddings ohne GloVe

Zum Erstellen der Word Embeddings wurden drei Ansätze verglichen. Im ersten Ansatz wurde jedes Wort auf einen Vektor mit zufälligen, gleichverteilten Werten abgebildet. Die Repräsentation wurde in der Embedding-Schicht anschließend beim Training des Neuronalen Netz angepasst. Die zwei weiteren Ansätze verwenden bereits vortrainierte Word Embeddings, die mit dem Algorithmus GloVe erstellt wurden. Diese wurden einmal unverändert übernommen und einmal – wie im ersten Ansatz – im Training weiter angepasst.

Wie in Abbildung 5.4 zu sehen ist, wurde die höchste Accuracy mit den GloVe-Embeddings in der zweiten Epoche erreicht, wenn diese beim Lernen vom Netz angepasst werden. Über 30 Epochen hinweg wurde kein besseres Resultat erzielt. Es fällt zusätzlich auf, dass mit den unveränderten GloVe-Embeddings die konstantesten Ergebnisse erzielt werden.

Pooling-Schicht

Das Reduzieren mehrerer Werte auf einen Wert beim Pooling kann auf mehrere Arten geschehen. Neben dem maximalen Wert kann beispielsweise der Durchschnittswert gebildet werden. Damit wird versucht, Ausreißer zu eliminieren. Mit dem sogenannten *Average Pooling* wurde nach ein und fünf Epochen je eine *Accuracy* erreicht, die um ein Prozent geringer war als die beim *Max Pooling*.

Beim *Global Max Pooling* wurde nach einer und fünf Epochen die gleiche *Accuracy* wie beim *Max Pooling* erreicht, wenn die Nachkommastellen der Prozentzahl vernachlässigt werden. Verwendet wurde Schlussendlich Max Pooling.

Max Pooling Fenstergröße

In der Max Pooling-Schicht wurde mit kleineren Fenstergrößen eine höhere Accuracy erzielt, wie in Tabelle 5.2.1.1 zu sehen ist. Nach fünf Epochen ergab sich die höchste Accuracy für die Fenstergröße 20 mit 54%, was zwei Prozent höher ist als die Accuracy der Fenstergrößen zwei, vier und acht.

Fenstergröße	**2**	**4**	**8**	**20**
Genauigkeit [%]	51,99	52,13	51,89	54,42
Laufzeit [Sek.]	36,70	33,58	32,03	32,21

Tabelle 5.2: Verschiedene Fenstergrößen im Vergleich

Alle Konfigurationen wurden auf acht Prozessorkernen ausgeführt. In der Tabelle ist vor allem zwischen Fenstergröße zwei und 20 ein deutlicher Unterschied zu sehen. Das zeigt, wie durch Datenreduktion die Rechenzeit in Neuronalen Netzen verringert werden kann.

Aktivierungsfunktion

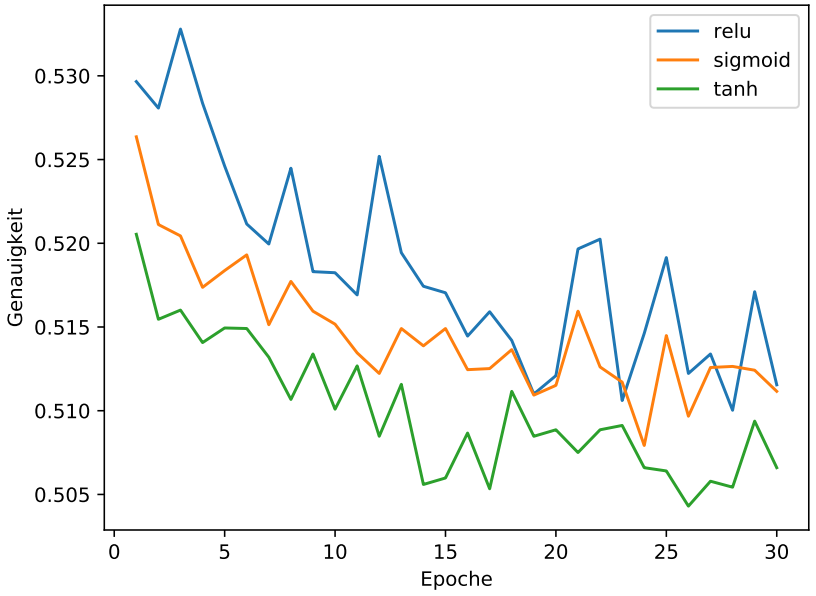

Abbildung 5.5: Verschiedene Aktivierungsfunktionen bei der ersten vollvernetzten Schichten

Für die erste vollvernetzte Schicht wurden verschiedene Aktivierungsfunktionen getestet. Die Ergebnisse sind in Abbildung 5.5 dargestellt. Die höchste Accuracy erzielte in fast allen Epochen ReLu. Das beste Ergebnis wurde in der dritten Epoche erreicht. Für die Ausgabeschicht wird Softmax verwendet, um für das binäre Klassifikationsproblem die Wahrscheinlichkeit für jede Klasse als Ausgabe zu erhalten.

Overfitting

Mit dem beschriebenen Aufbau des Netzes und den Anpassungen an den Hyperparametern wurde auf den Testdaten eine *Accuracy* von 54% nach einer Epoche erzielt. Über zehn Epochen sank die Accuracy bis auf 52%. Auf den Trainingsdaten lag der Wert nach einer Epoche bei 56% und nach zehn Epochen bei 84%.

Abbildung 5.6 zeigt, wie Dropout Overfitting vermeiden kann. Verwendet wurden drei Schichten mit jeweils 50% Dropout. Die rote und orangene Linie zeigen die Accuracy auf den Trainingsdaten mit und ohne Dropout. Wie zu erkennen ist, sinkt die Accuracy hier mit dem Dropout. Auf den Testdaten wurde ohne Dropout eine sehr geringe Accuracy im Vergleich zu den Trainingsdaten erzielt, siehe grüne Linie. Die darüberliegende blaue Linie

zeigt, wie Dropout die Ergebnisse auf den Testdaten vebessert und dafür sorgt, dass das Netz mehr generalisiert als zuvor.

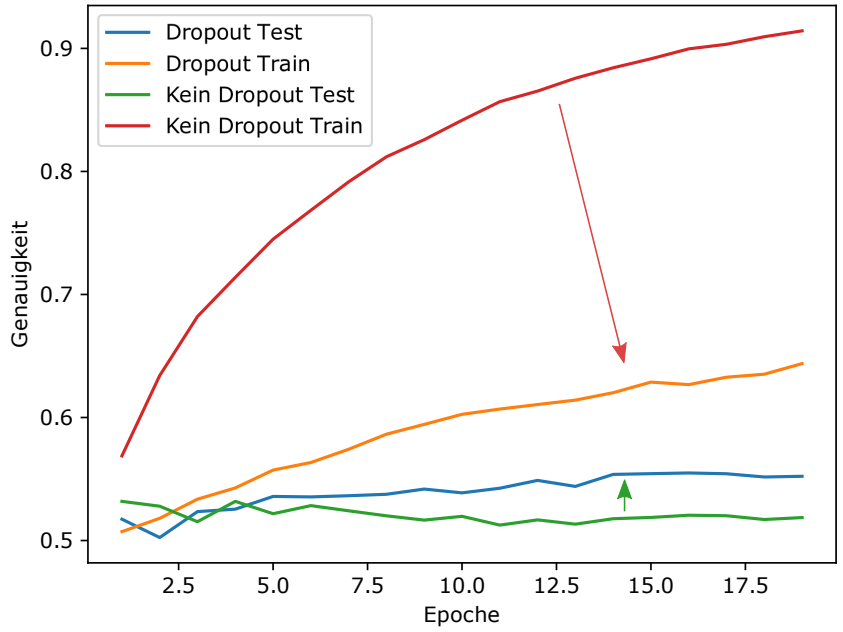

Abbildung 5.6: Entwicklung der Accuracy auf den Test- und Trainingsdaten über mehrere Epochen

5.2.1.2 Ergebnisse

Die Accuracy aller Tweets liegt bei 55% (siehe auch Abbildung 5.7). Zusätzlich wurde die durchschnittliche Accuracy betrachtet, wenn alle Unternehmen gleich gewichtet werden. Der Schnitt pro Unternehmen liegt zwei Prozent unter diesem Wert. Demnach ist die Accuracy für Unternehmen im Durchschnitt höher, wenn für diese mehr Tweets existieren. Tatsächlich werden 27% aller Tweets Apple zugeordnet. Alle anderen Unternehmen haben einen wesentlich kleineren Anteil an der Menge der Tweets und keines besitzt über 10% an der Gesamtmenge. Mit einer Accuracy von 60% erzielt Apple auch ein höheres Ergebnis als die anderen Unternehmen im Schnitt und hebt damit das Gesamtergebnis an.

Die höchste Accuracy erreichte Chevron mit 84% gefolgt von General Electric mit 83%. Durch nähere Betrachtung der Ergebnisse beider Unternehmen soll im Folgenden eine Erklärung für das überdurchschnittliche Ergebnis gefunden werden.

Im Fall von Chevron fiel der Aktienkurs an 16 von den 21 betrachteten Tagen. Das Netz klassifizierte nur 2% der 635 Tweets über Chevron in die Klasse ”Kurs steigend” und erzielte so eine sehr hohe Accuracy. Precision und Recall sind null, da kein Tweet richtig-positiv eingeordnet wurde. Bei Betrachtung der Trainingsdaten zu Chevron zeigt sich, dass auch hier mit 71% ein großer Teil der Tweets mit ”Kurs fallend” klassifiziert wurde.

Hier zeigt sich, dass eine hohe Accuracy nicht zwingend mit dem Inhalt der Tweets zusammenhängen muss. Denn wenn der Kurs sich im Trainings- und Testzeitraum gleich entwickelt, kann das Netz überdurchschnittliche Ergebnisse erzielen, indem es nur anhand des Unternehmens klassifiziert und den Rest der Informationen vernachlässigt. Im Fall von Chevron wäre es möglich, dass das Neuronale Netze die negative Entwicklung in den Trainingsdaten auf die Testdaten übertragen hat.

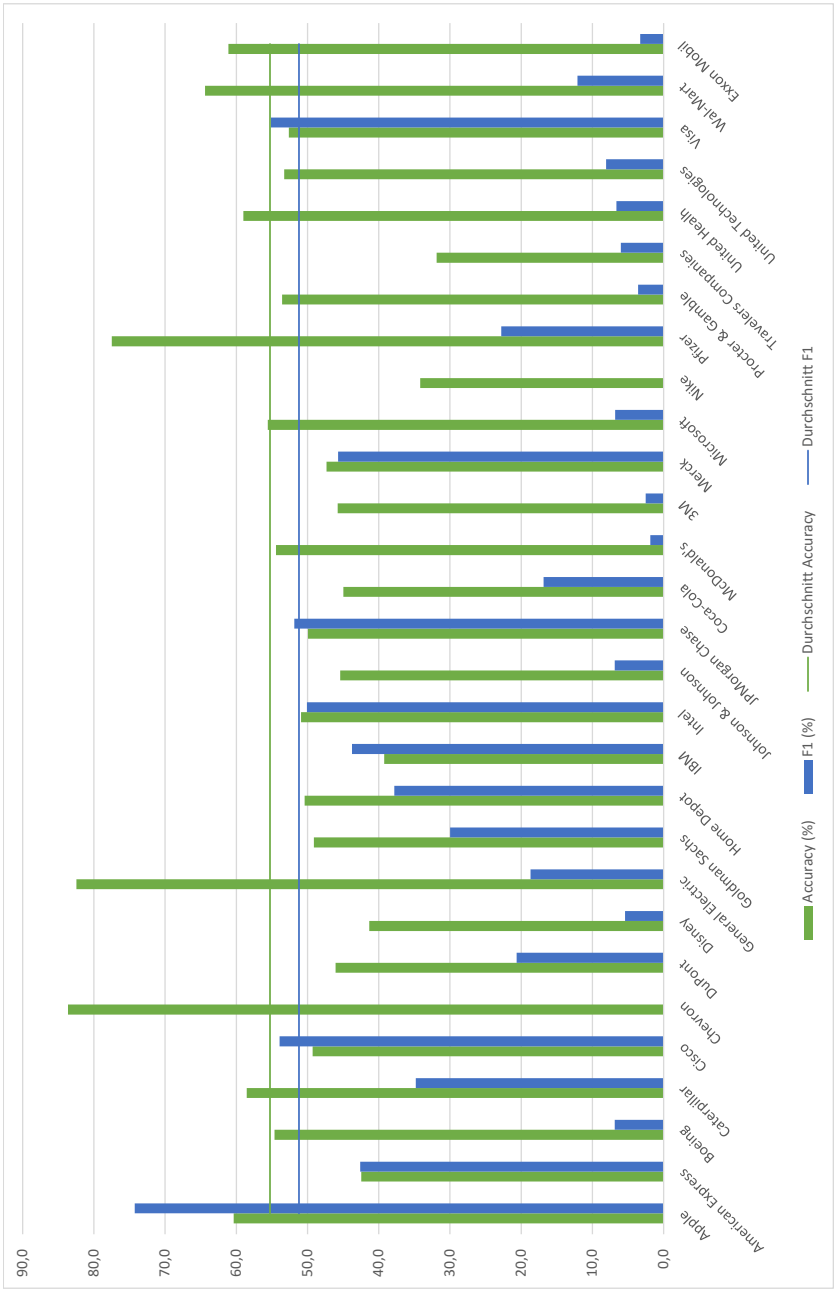

Abbildung 5.7: Accuracy und F1 pro Unternehmen für das Max Pooling-Netz

Bei dem Unternehmen mit der zweithöchsten Accuracy ist dem nicht mehr so. General Electric erreichte eine Accuracy von 83%. Im Testdatensatz ist der Aktienkurs an 19 von 21 Tagen gefallen. Im Trainingsdatensatz gehören lediglich 40% der Tweets zur Klasse "Kurs fällt". Das Neuronale Netze konnten hier demnach nicht aus einer Mehrheit einer Klasse in den Trainingsdaten auf eine Mehrheit in den Testdaten schließen. Dennoch klassifizierte das Netz 91% der Tweets mit "Kurs fallend" und erzielte so eine hohe Accuracy. An den zwei Tagen, an denen der Kurs von General Electric stieg, wurden 16% der Tweets richtig klassifiziert (Recall). Von allen Klassifizierungen für fallende Kurse waren nur 23% richtig (Precision). Relativ betrachtet wurden so mehr falsche Prädiktionen für steigende als für fallende Kurse gemacht.

Die durchschnittliche Precision liegt bei 57%, der durchschnittliche Recall bei 47%. Die richtig-positiven Prognosen sind demnach im Verhältnis zu allen positiven Kursentwicklungen niedriger als im Verhältnis zu allen positiven Prognosen. In andern Worten existieren weniger falsch-positive als falsch-negative Prognosen, da gilt:

$$Precision < Recall = \frac{RP}{RP + FP} < \frac{RP}{RP + FN} \Rightarrow FN < FP$$

Der größte Unterschied zwischen Precision und Recall existiert zwischen den Unternehmen 3M und Travelers Company. Für beide Unternehmen stellt sich daher die Frage, warum öfter falsche negative als falsche positive Prognosen geliefert wurden. Für beide Unternehmen wurden über 97% negative Prognosen gemacht. Von den sehr wenigen positiven Prognosen waren alle korrekt, weshalb die bestmögliche Precision erreicht wurde. Da der Kurs der beiden Unternehmen nicht übermäßig stark fiel, waren sehr viele der Prädiktionen für sinkende Kurse nicht korrekt. Dadurch kommt es zu einem sehr niedrigen Recall.

5.2.2 Convolution-Netz

Das Convolutional-Netz erreichte eine Accuracy von 54% für die Vorhersage des nächsten Tages. Vor mehreren Optimierungen war diese um zwei Prozent geringer. Diese Verbesserungen werden im folgenden Abschnitt vorgestellt. Anschließend wird näher auf die Ergebnisse eingegangen.

5.2.2.1 Optimierungen und ihre Auswirkungen

Anzahl der Convolutional-Schichten

Ein "Deep Convolutional Neural Network" kann aus sehr vielen aufeinanderfolgenden Convolutional- und Max-Pooling-Schichten bestehen. Daher wurde mit ein bis drei Kombinationen aus beiden Schichten getestet, wie sich die Anzahl der Schichten auf die Accuracy auswirkt. Für jede zusätzliche Kombination aus Convolutional- und Max-Pooling-Schicht sank die Accuracy auf den Test- und Trainingsdaten um circa ein Prozent.

Duch die zusätzlichen Schichten bekommt das Neuronale Netz zusätzliche Gewichte und wird dadurch komplexer. Jedoch verschlechterten zusätzliche Schichten die Accuracy. Das kann damit zusammenhängen, dass der Trainingsdatensatz zu klein und das Problem nicht komplex genug für das Neuronale Netz ist.

Anzahl Filter in Convolutional-Schicht

Für verschiedene Anzahl an Filtern ergab sich die höchste Accuracy mit 53% für 128 Filter. Im Vergleich dazu wurde mit 64 und 256 Neuronen eine um jeweils ein Prozent geringere Accuracy erzielt. Mit mehr Filtern steigt die Anzahl der Parameter im Netz. Dadurch kam es zum Overfitting. Mit weniger Filter sank sowohl die Accuracy auf den Trainingsdaten- als auch auf den Testdaten.

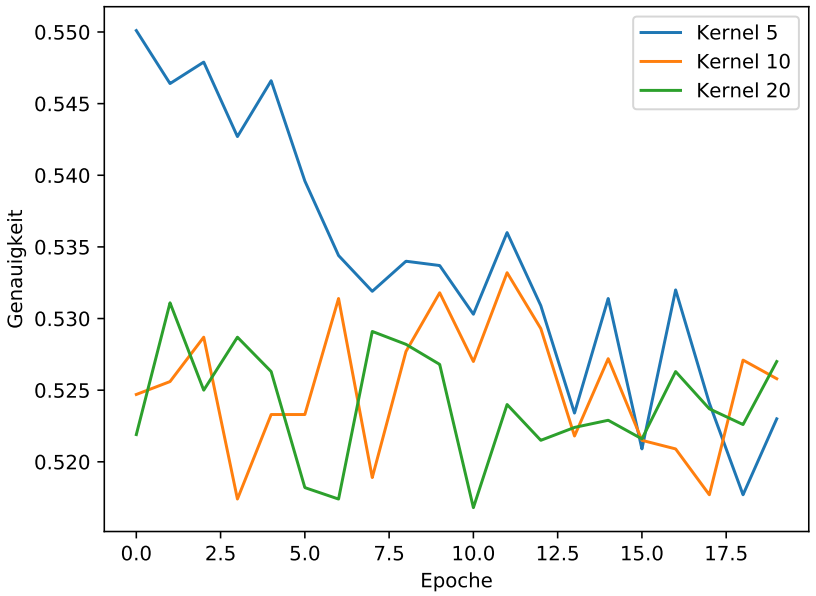

Abbildung 5.8: Verschiedene Maskengrößen (Kernel) in der Convolutional-Schicht

Größe der Maske in Convolutional-Schicht

Die Maske, die in einer Convolutional-Schicht schrittweise über die Eingabe bewegt wird, kann unterschiedlich groß sein. Die Eingabe ist eine Matrix, in der jede Spalte ein Wort enthält.

Bei verschiedenen Größen der Maske (*engl.* Kernel) wurden die besten Ergebnisse mit der Größe fünf erzielt, blaue Linie in Abbildung 5.8. Das heißt, der Filter wird je über drei Wörter gelegt. Andere Größen führten zu einem ein- bis dreiprozentigem Rückgang der Accuracy. Außerdem zeigte sich ein starker Abfall der Accuracy bereits nach der ersten Epoche.

Updatefunktion

Die Updatefunktionen Adadelta, Adam und Nadam erzielten ähnliche Ergebnisse über 20 Epochen hinweg. RMSProp liefert sehr schwankende und durchweg schlechtere Resultate als die anderen Funktionen. Da das beste Ergebnis nach 20 Epochen mit Adam erzielt wurde, wird diese als Updatefunktion verwendet.

Aktivierungsfunktion

Wie beim Max Pooling-Netz wurde für die erste vollvernetzten-Schicht Sigmoid als Aktivierungsfunktion verwendet und Softmax für die zweite.

Größe des Batches

Mit einer kleinen Batchgröße lassen sich teilweise bessere Ergebnisse erzielen. Das liegt daran, dass die Gewichte so öfter angepasst werden und das Netz schneller lernt. Mit weniger Daten wirken jedoch auch mehr Störfaktoren auf die Gewichte. Daher muss – abhängig vom Datensatz – ein Mittelweg zwischen zu großem und zu kleinem Batch gefunden werden.

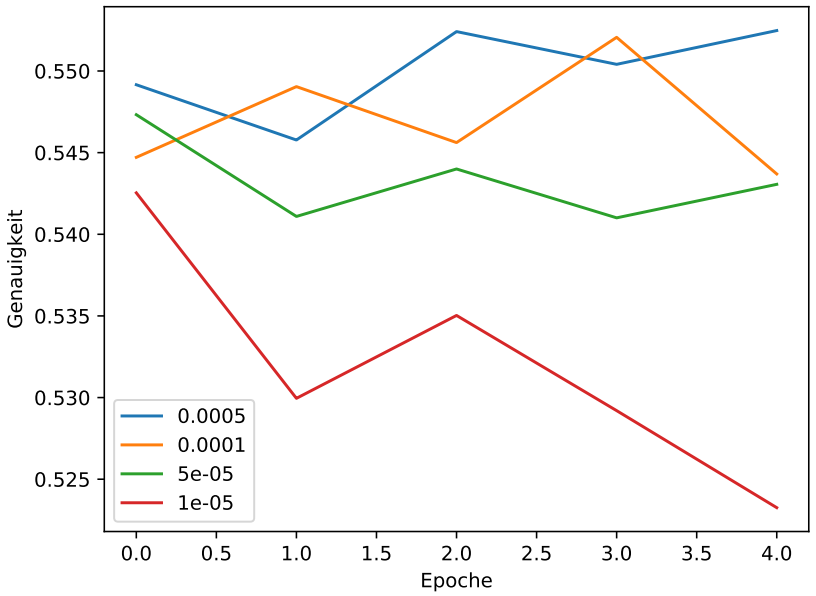

Abbildung 5.9: Verschiedene L2-Regularisierungswerte im Vergleich

Im Vergleich zu größeren Batches wurde die höchste Accuracy nach zehn Epochen mit 60 Tweets pro Batch erreicht. Nach einer Epoche wurde jedoch mit der Größe 120 eine höhere Accuracy erzielt. Das Problem, dass keine eindeutig bessere Optimierung über alle Epochen festgelegt werden konnte, trat bei einigen Parametern auf. Denn für viele Optimierungen lag das beste Ergebnis in unterschiedlichen Epochen. In solchen Fällen mussten Kompromisse eingegangen werden. Das Convolutional-Netz erreichte nach allen Optimierungen die höchste Accuracy nach sieben Epochen mit der Batchgröße 60.

L2-Regularisierung

Bei Anpassung der L2-Regularisierung erreichte der Wert $0,0005$ die höchste Accuracy. Abbildung 5.9 zeigt, dass kleinere Regularisierer schlechtere Ergebnisse erbrachten. Größerer Werte für die L2-Regularisierung führten zu mehr Overfitting und schlechteren Ergebnissen auf den Testdaten.

5.2.2.2 Ergebnisse

Um die Ergebnisse für alle Unternehmen zusammenzufassen, werden zwei Durchschnitte betrachtet. Der erste Schnitt gibt das durchschnittliche Ergebnis aller Tweets an. Der zweite Durchschnitt wird gebildet, indem alle Unternehmen gleich gewichtet werden. Die beiden Werte unterscheiden sich, da nicht alle Unternehmen in gleich vielen Tweets vorkommen.

Das Convolutional-Netz klassifizierte 54% der Tweets korrekt (Schnitt 1). Für die einzelnen DJIA-Unternehmen ergab sich im Schnitt eine geringere Accuracy von 52% (Schnitt 2). Die Ergebnisse sind in Abbildung 5.10 grafisch dargestellt. Demnach erreichten Unternehmen mit mehr Tweets im Durchschnitt eine höhere Accuracy. Für den Wert von $F1$ gilt, dass der Schnitt pro Unternehmen höher ist als der pro Tweet. Die größte Abweichung bei den zwei Durchschnittswerten zeigt sich bei dem Recall.

Der hohe Recall pro Tweet von 48% wird stark von Apple beeinflusst. Das Unternehmen macht 27% der Tweets aus und hat mit 89% einen vergleichsweise hohen Recall. Gleichzeitig wird der durchschnittliche Recall pro Unternehmen stark von den vielen Unternehmen

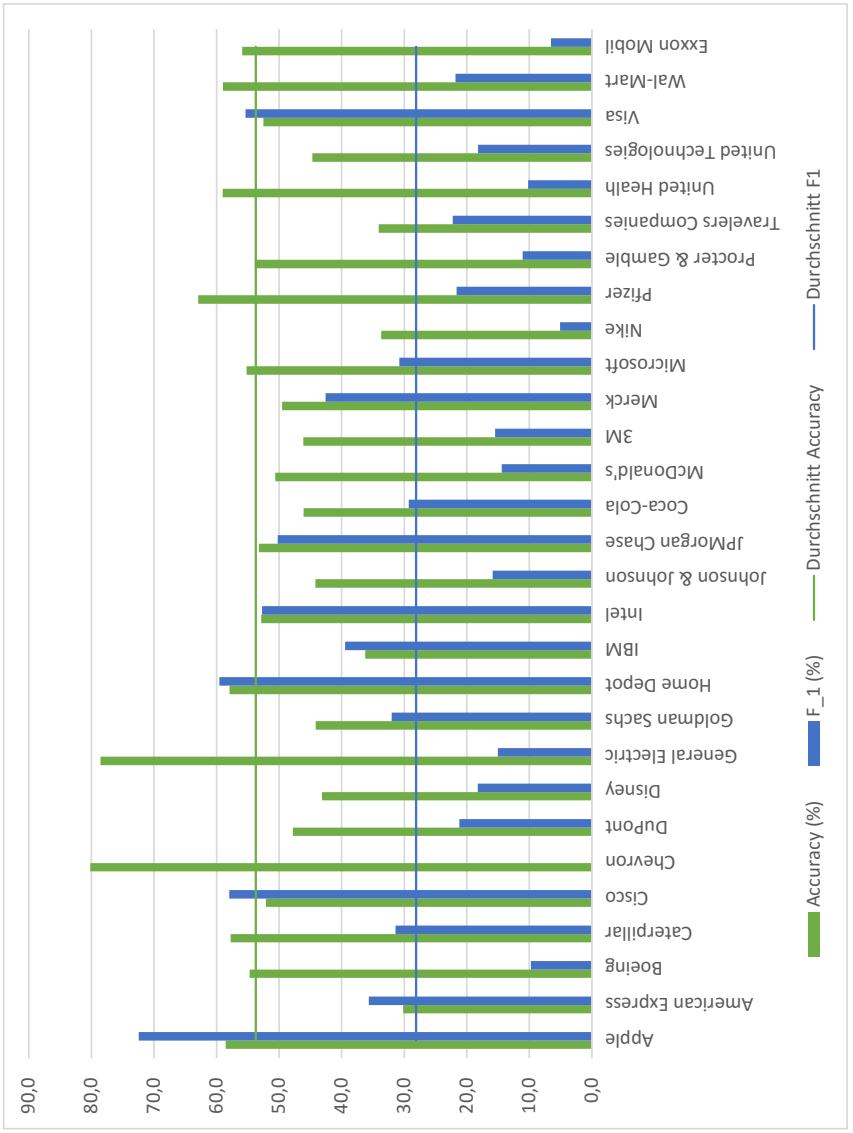

Abbildung 5.10: Accuracy und F1 pro Unternehmen für das Convolutional-Netz

mit niedrigem Recall beeinflusst, die nur in wenigen Tweets erwähnt werden. Chevron, Nike und Exxon Mobil haben den geringsten Recall unter allen Unternehmen. Für alle drei Unternehmen wurden weniger als 10% Prädiktionen für steigende Kurse gemacht. Gleichzeitig sind für alle drei Unternehmen über 70% der Tweets in den Testdaten mit fallendem Kurs klassifiziert.

Das Netz klassifizierte für die drei Unternehmen viele Tweets als negativ. Dadurch wurden nur wenige steigende Kurse richtig prädiziert. Und im Verhältnis dazu wurden viele fallende Kurse falsch eingeordnet. Dies führte zu dem geringen Recall. Gleichzeitig wurde in zwei der drei Fälle so viele richtige Klassifikationen für fallende Kurse gemacht, dass die Accuracy über dem Durchschnitt lag. Im dritten Fall (Nike) stieg der Kurs in den meisten Fällen und die Mehrheit der negativen Klassifikationen war falsch.

Für Precision und F1 zeigt sich wie beim Recall, dass Unternehmen, die selten in den Tweets vorkommen, in der Regel schlechtere Ergebnisse erzielen als jene, die oft erwähnt werden.

5.2.3 LSTM-Netz

Das LSTM erreichte anfangs lediglich eine Accuracy von 52%. Durch die im Folgenden vorgestellten Optimierungen konnte dieser Wert auf 54% gesteigert werden. Im Anschluss daran werden die Ergebnisse, die mit dem Netz erzielt wurden, näher vorgestellt.

5.2.3.1 Optimierungen und ihre Auswirkungen

Die Hyperparameter wurden teilweise parallel optimiert. Daher steigt das Ergebnis im Folgenden nicht notwendigerweise mit jeder Optimierung.

Updatefunktion

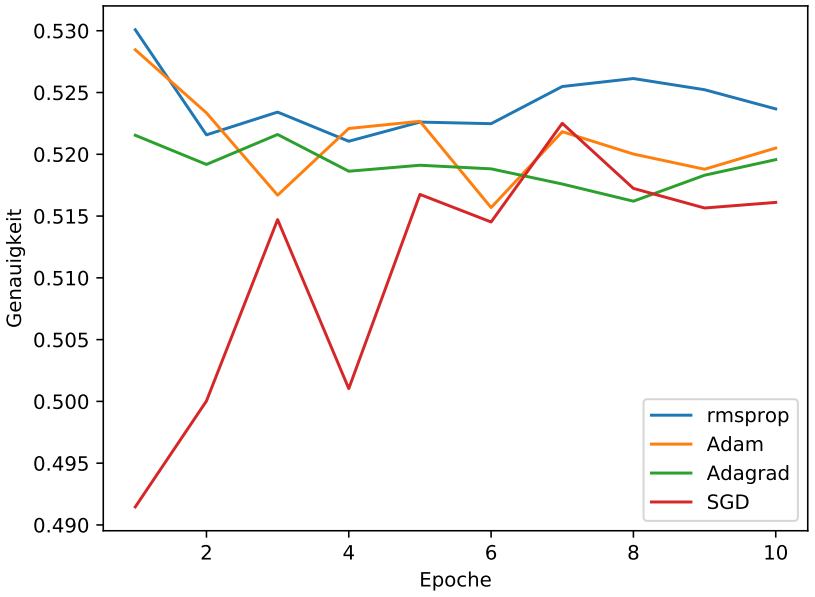

Abbildung 5.11: Updatefunktionen im Vergleich für das LSTM-Netz

Als Updatefunktion wurde der Algorithmus RMSProp verwendet. Nach zehn Epochen lieferte dieser leicht bessere Ergebnisse als die Funktionen Adam, Adagrad und SGD. Bis auf die ersten drei Epochen beim SGD schneiden jedoch alle vier Funktionen sehr ähnlich ab und unterschieden sich meist nur um 0,5%.

Aktivierungsfunktion

Für die erste vollvernetzte Schichte wurde die Aktivierungsfunktionen Sigmoid verwendet, die beim Max Pooling-Netz die höchste Genaugikeit erreichte. Die Ausgabeschicht verwendet wie auch bei den vorherigen Netzen Softmax.

Lernrate

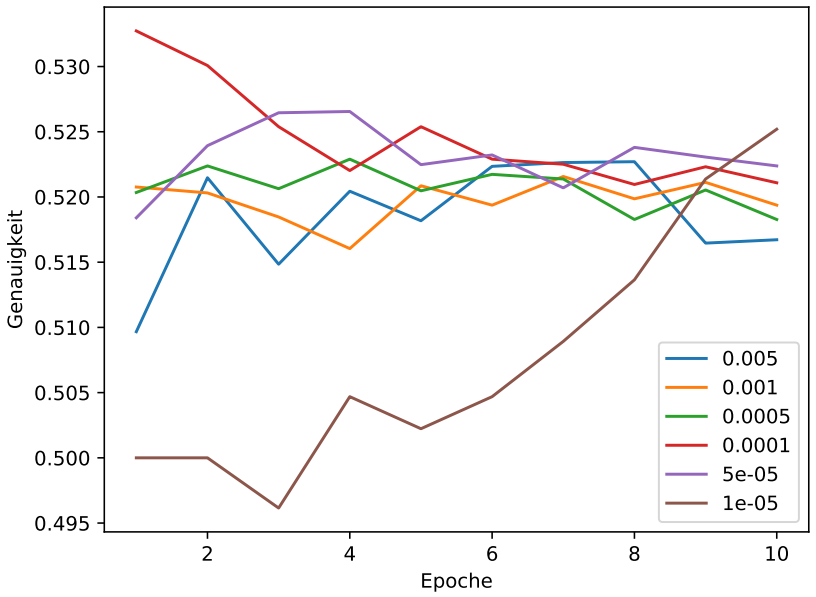

Abbildung 5.12: Lernraten über zehn Epochen

In einem ersten Test von verschiedenen Lernrate wurde mit dem kleinsten Wert (0,00001) nach zehn Epochen der beste Ergebnis erzielt, wie in Abbildung 5.12 zu sehen ist. Da die Accuracy mit diesem Wert ab Epoche fünf monoton stieg, wurde untersucht, wie sich dieser und noch kleinere Werte über weitere Epochen entwickeln. Abbildung 5.13 zeigt, dass zwischen den Epochen 60 und 90 die maximale Accuracy bei leicht über 54% erreicht wird. Außerdem ist zu sehen, dass das Netz bei zu kleiner Lernrate (rote Linie in Abbildung 5.13), selbst über hundert Epochen nur sehr wenig lernt.

Je kleiner die Lernrate, desto länger muss das Netz trainiert werden. Zudem birgt das die Gefahr, auf einem Plateau stehen zu bleiben und nie ein Minimum der Kostenfunktion zu erreichen, weshalb diese nicht kleiner gewählt wird.

Dropout-Schicht

Das Modell wird lediglich vier Epochen trainiert, da es später zum Overfitting kommt. Zusätzlich wurde Dropout als Möglichkeit betrachtet, um Überanpassung zu vermeiden. Getestet wurde Dropout in der LSTM-Schicht, eine Dropout-Schicht nach der LSTM-Schicht und eine Dropout-Schicht nach der ersten vollvernetzten Schicht. Getestet wurde jeweils 20% und 50% Dropout.

Die besten Ergebnisse wurden jedoch nach vier Epochen ohne Dropout erzielt. Die Accuracy auf den Trainingsdaten steigt in dem Zeitraum lediglich bis 62%. Dieser Wert ist ähnlich wie beim Max Pooling- und dem Convolutional-Netz mit Dropout. Ohne Dropout ist die Accuracy bei diesen Netzen auf den Trainingsdaten wesentlich höher. Dass LSTM-Netz kann demnach ohne Dropout besser generalisieren als die anderen beiden Netze.

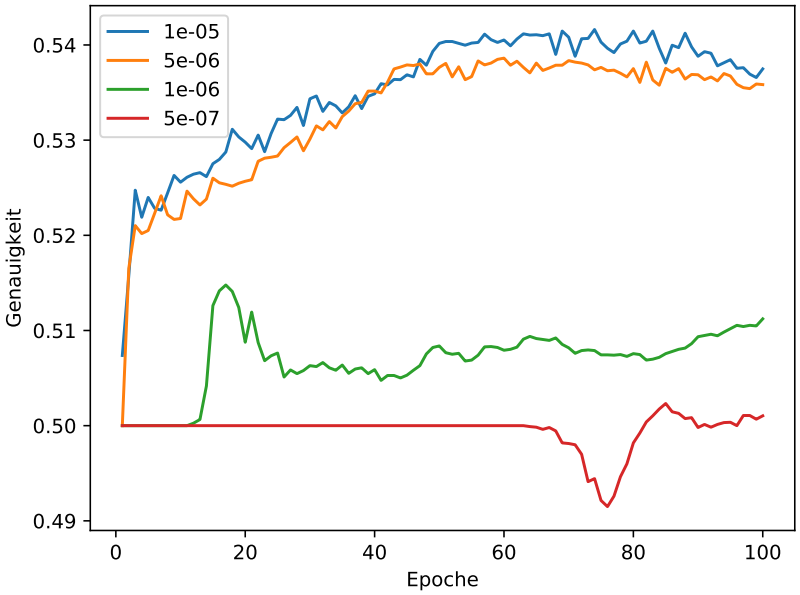

Abbildung 5.13: Lernraten über 100 Epochen

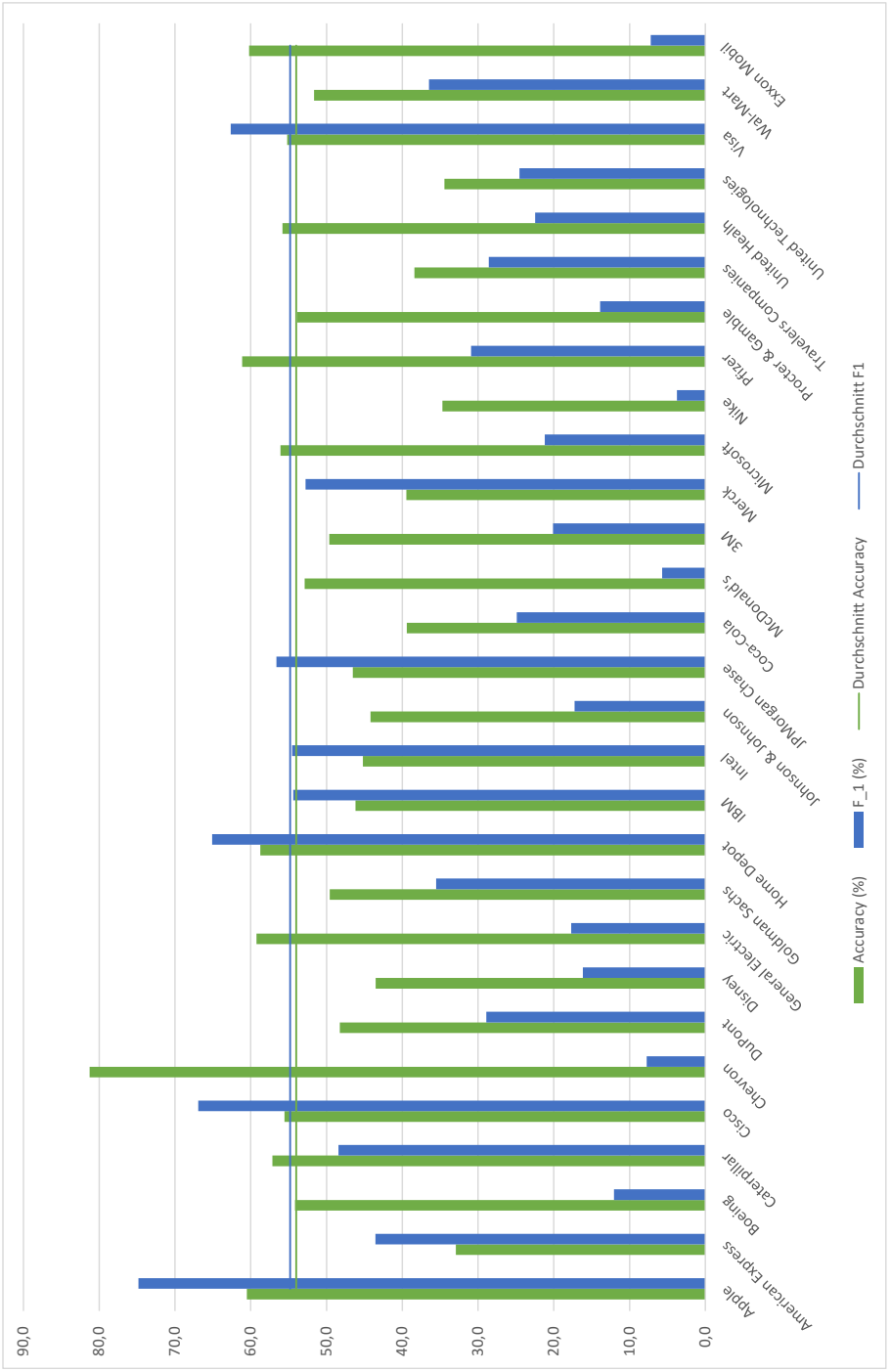

Abbildung 5.14: Accuracy und F1 pro Unternehmen für das LSTM-Netz

5.2.3.2 Ergebnisse

Das Neuronale Netz mit einer LSTM- gefolgt einer vollvernetzten Schicht erreichte pro Tweet eine Accuracy von 54% und einen F1-Wert von 55%.

Die Ergebnisse der einzelnen Unternehmen sind in Abbildung 5.14 zu sehen. Diese sind im Durchschnitt, wenn jedes Unternehmen gleich gewichtet wird, geringer als die durchschnittlichen Metriken für alle Tweets gesamt betrachtet. Demnach werden die besseren Ergebnisse von den Unternehmen erzielt, für die mehr Tweets vorliegen.

Vor allem beim Recall ist der Unterschied der beiden Durchschnitte auffallend groß, was auch zu einer großen Diskrepanz der Werte bei der Metrik F1 führt. Werden alle Unternehmen gleich gewichtet, liegt der Recall bei 37%. Wenn jeder Tweet gleich gewichtet wird, liegt der Recall bei 56%. Das Unternehmen mit dem niedrigsten Recall von zwei Prozent wird mit einem Prozent auch in unterdurchschnittlich wenigen Tweets erwähnt.

Allerdings lässt sich nicht für alle Unternehmen aus vielen Tweets auf hohe Ergebnisse in den vier Metriken folgern. Beispielsweise hat American Express den zweithöchsten Recall, wird aber auch nur in einem Prozent der Tweets erwähnt. Für dieses Unternehmen lagen im Trainingszeitraum 84% der Tweets in der Klasse "Kurs steigt". Das Neuronale Netz klassifizierte den Großteil der Tweets in den Testdaten ebenfalls mit "Kurs steigt", was zu vielen richtig-positiven Klassifizierungen führte. Dadurch kam es auch zu einem hohen Recall.

Hier zeigt sich erneut, dass die Neuronale Netze die Möglichkeit haben, den Namen eines Unternehmens generell mit fallenden oder steigenden Kursen zu assoziieren, wenn es in den Trainingsdaten sehr oft gestiegen oder gefallen ist. Daher wurden in einem Test alle Aktiensymbole aus den Tweets entfernt (siehe 5.2.5 *Selektion der Daten*).

Ein Beispiel für ein unterdurchschnittliches Ergebnis trotz vieler Tweets ist Microsoft. Das Unternehmen macht sechs Prozent der betrachteten Tweets im Trainingsdatensatz und acht Prozent im Testdatensatz aus, was deutlich über dem Durchschnitt liegt. Dennoch wurde in den Trainingsdaten lediglich ein F1-Wert von 21% erreicht. Der Wert ist so gering, da nur 11% der Tweets am folgenden mit "Kurs steigt" klassifiziert wurden. Im Testdatensatz gehören 58% der Tweets zu dieser Klasse, also nicht deutlich viel mehr als die Hälfte. Dass das übermäßig hohe Auftreten einer Klasse für das Unternehmen in den Trainingsdaten von dem Neuronalen Netz auf die Testdaten übertragen wurde, scheint daher weniger wahrscheinlich. Eine andere Begründung für diese Entwicklung ist, dass die Tweets zu Microsoft im Testzeitraum wenig mit dem Aktienkurs korrelieren. In diesem Fall würde das bedeuten, dass es in den Tweets viele Indikatoren für fallende Kurse gab, der Kurs aber tatsächlich nicht gefallen ist.

Zusammenfassend lässt sich sagen, dass auch bei dem LSTM-Netz zwischen den Unternehmen große Unterschiede existieren. Dabei scheinen extreme Ergebnisse von sehr vielen Faktoren abzuhängen und eine eindeutige Begründung für sehr hohe und sehr niedrige Werte ist nur schwer zu bestimmen.

5.2.4 Anpassung des Datensatzes

5.2.4.1 Störfaktoren

Im Vergleich zu oft für die Sentimentanalyse verwendendeten Datensätzen wie "The Large Movie Review Dataset" (oft auch kurz "IMDB dataset" genannt) und dem "Rotten Tomatoes Dataset" enthält dieser Datensatz sehr viele URLs, Nutzernamen (mit "@" markiert), Hashtags (mit "#" markiert) und Emojis.

Um möglichst viele dieser Störfaktoren vor der Sentimentanalyse zu entfernen, wurden alle URLs, Nutzernamen und Hashtags entfernt.

Zudem wurden nur Nutzer mit über 100 Follower berücksichtigt, um Spam zu vermeiden. Außerdem wurden englisch- und deutschsprachige Tweets nur getrennt betrachtet.

5.2.5 Selektion der Daten

Die einzelnen Wörter in den Texten werden anstelle von Buchstaben durch je eine Ganzzahl repräsentiert. Diese steht für die Häufigkeit, mit der das Wort über alle Tweets verteilt vorkommt, wobei das Wort mit der höchsten Zahl am häufigsten vorkommt. Für das Training und die Überprüfung des Neuronalen Netzes wurden lediglich die 20.000 am häufigsten vorkommenden Wörter betrachtet. Ein größeres Wörterbuch brachte keine Verbesserung der Accuracy mit sich, ein kleineres Wörterbuch mit 5000 und 2000 Wörtern verschlechterte die Accuracy beim Max Pooling-Netz um ein beziehungsweise drei Prozent.

Eine mögliche Verbesserung der Ergebnisse der Neuronalen Netz ist das Löschen der Erwähnung der Aktienunternehmen, sodass zum Beispiel für ein Aktienunternehmen, dessen Kurs in den Trainingsdaten immer gefallen ist, das Netz nicht den Zusammenhang zwischen Name des Unternehmens und fallendem Kurs erlernt. Für diesen Datensatz ergab sich jedoch mit dem Max Pooling-Netz eine Einbuße von drei Prozent bei der *Accuracy*.

Für jeden Tweet ist die Anzahl der Follower des Autors im Datensatz enthalten, also die Anzahl der Personen, die den Autor auf Twitter abonniert hat. Werden nur Tweets berücksichtigt, deren Autoren mindestens 100 Follower haben, so lässt sich die *Accuracy* beim Max Pooling-Netz über fünf Epochen um ein Prozent auf 53% steigern.

5.3 Vergleich der Prädiktionsmöglichkeiten

Im Folgenden wird die Prädiktion mittels Sentimentanalyse mit der direkten Prädiktion verglichen. Wie im vorherigen Teil der Evaluation basieren die Ergebnisse auf den DJIA-Unternehmen im Datensatz $\$Symbol$.

Die Tabelle 5.3 zeigt, dass die direkte Prädiktion besser abgeschnitten hat als die Bestimmung mittels Sentimentanalyse. Dass mit der Sentimentanalyse schlechtere Ergebnisse erzielt wurden als mit einem Zufallsexperiment, kann verschiedene Gründe haben. Zum einen ist das RNTN auf einer anderen Domäne trainiert worden als Kurzmitteilungen. Zum anderen muss keine positive Korrelation zwischen dem durchschnittlichen Sentiment auf Twitter über ein Unternehmen und dessen Aktienkurs bestehen.

Die drei Neuronalen Netze zur direkten Prädiktion schnitten mit 54% und 55% Accuracy besser ab als das vorgestellte Zufallsexperiment. Zumindest bei der Accuracy hat sich gezeigt, dass mit komplexeren Netzen auf den verwendeten Daten nicht unbedingt bessere Ergebnisse erzielt werden. Hier war vor allem Overfitting ein Problem. Betrachtet man genauer, wie die endgültigen Metriken zu Stande kamen, so fällt auf, dass das Max

Prädiktionsart	Name	Pro Tweet		Pro Unternehmen	
		Accuracy [%]	F1 [%]	Accuracy [%]	F1 [%]
Mittels Sentimentanalyses	RNTN	49,7	39,6	49,0	22,5
Direkt	Max Pooling	55,3	51,2	53,2	26,4
	Convolution	53,7	51,0	53,2	29,6
	LSTM	54,0	54,8	51,9	37,8

Tabelle 5.3: Endergebnisse aller Neuronalen Netze im Vergleich

Pooling-Netz über den Testzeitraum viele Tweets über ein Unternehmen hauptsächlich in eine Klasse eingeteilt hat. So wurden vor allem gute Ergebnisse erzielt, wenn die Aktie tatsächlich hauptsächlich gestiegen oder gefallen ist. Dahingegen waren die Klassifizierungen von den letzten beiden Netzen ausgeglichener. Das lässt den Schluss zu, dass das Convolutional- und das LSTM-Netz ihr Potenzial auf einem größeren Datensatz, bei dem idealerweise die Klassen pro Unternehmen gleich oft auftreten, besser ausschöpfen könnten und hier bessere Ergebnisse als das Max-Pooling-Netz erzielen könnten.

Bei der Metrik F1 schneidet das LSTM-Netz am Besten ab. Der Wert wird durch die meisten richtig der Klasse der steigenden Kurse zugeordneten Klassifizierungen erreicht. Sowohl im Verhältnis zu allen Klassifizierungen "Kurs steigt" als auch im Verhältnis zu den tatsächlich zur Klasse "Kurs steigt". Die Prädiktionen des Netzes sind im Vergleich zu den anderen Netzen am ausgeglichensten. Die restlichen Netze haben mehr Tweets zur Klasse der fallenden Kurse zugeordnet. Tatsächlich sind jedoch beiden Klassen exakt gleich viele Tweets zugeordnet.

Wird der Durchschnitt aus Accuracy und F1 zur Bewertung der vier Methoden verwendet, so erzielt das LSTM-Netz die besten Ergebnisse mit 54%. Darauf folgt das Max Pooling-Netz mit 53%. Das Convolutional-Netz erreicht 52% und die Prädiktion mittels Sentimentanalyse lediglich 45%.

5.4 Vergleich der drei Datensätze

	Accuracy [%]	F1 [%]	Anzahl Tweets [Tsd.]
$\$Symbol$	54,0	54,8	60
$Symbol$	52,9	48,4	1.396
$Name$	50,1	51,9	2.806

Tabelle 5.4: Ergebnisse auf den verschiedenen Datensätzen im Vergleich

Im vorhergehenden Teil der Evaluation wurden die Ergebnisse vorgestellt, die für Tweets aus dem Datensatz $\$Symbol$ erreicht wurden. Das sind jene Tweets, die ein Dollarzeichen, gefolgt von dem Symbol eines Aktienunternehmens, enthalten. Zum Vergleich werden im Folgenden die Ergebnisse der restlichen betrachteten Datensätze vorgestellt.

Dass durch die drei verschiedenen Schreibweisen sehr unterschiedliche Tweets gesammelt wurden, soll an zwei Beispielen verdeutlicht werden. Ein relevanter Tweet im Datensatz $\$Symbol$ ist zum Beispiel folgender:

> Appeals court revives lawsuit claiming allegedly created iOS app monopoly. Read more: [Link] $AAPL

Nicht relevant für das Unternehmen Visa mit dem Aktiensymbol V ist beispielsweise:

> New on Ebay! Grand Theft Auto GTA V 5 (PS4) Brand New Sealed Game UK.

Der Vergleich der Schreibweisen basiert auf den Ergebnissen des LSTM-Netzes, welches im vorherigen Abschnitt im Durchschnitt die besten Resultate geliefert hat.

Wie Tabelle 5.4 zeigt, wurden die besten Ergebnisse auf dem kleinsten Datensatz erzielt. Im vorherigen Kapitel wurde bereits erwähnt, dass dieser vermutlich die wenigsten Störsignale und den relevantesten Inhalt enthält, da wenig andere Bedeutungen für die Schreibweise mit dem Dollarzeichen existieren. Die Vermutung liegt nahe, dass die Ergebnisse für den Datensatz $\$Symbol$ weiter verbessert werden könnten, wenn dieser so viele Daten enthalten würde, wie die anderen Datensätze.

5.5 Zeitfenster der Vorhersage

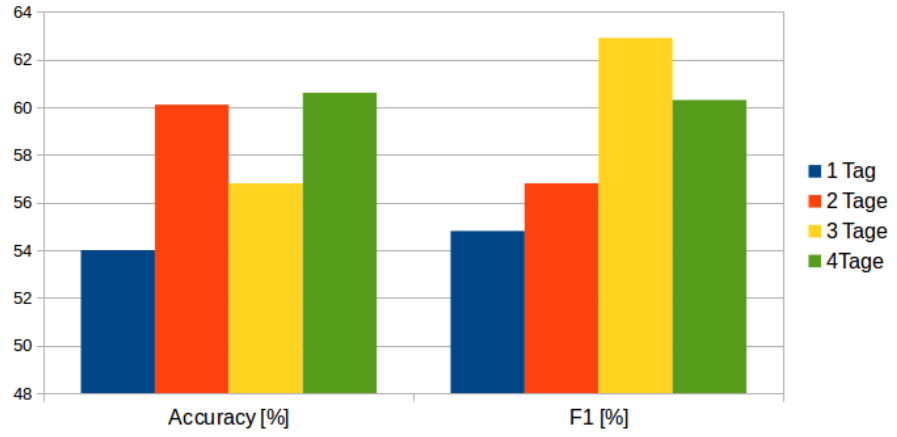

Abbildung 5.15: Ergebnisse für die Prädiktion des Aktienkurses für verschieden viele Tage im Voraus

Die Ergebnisse im vorhergehenden Teil betrachteten jeweils den Zusammenhang zwischen einem Tweet und dem Aktienkurs am folgenden Tag. Zusätzlich werden im Folgenden die Resultate zwei, drei und vier Tage nach dem Erstellen des Tweets untersucht. Die Ergebnisse stammen von dem LSTM-Netz.

In Abbildung 5.15 ist zu sehen, dass für zwei, drei und vier Tage im Voraus bessere Resultate erzielt wurden als für einen Tag im Voraus.

Diese Tendenz ist die gleich wie in der Arbeit von Bollen[14], in der ein Zusammenhang zwischen der Stimmung auf Twitter und dem Atkienkurs drei bis vier Tage später gefunden wurde.

Sei der Durchschnitt aus Accuracy und F1 das Maß für den Vergleich anhand eines Werts, dann wird das beste Ergebnis für die Prädiktion vier Tage nach der Erstellung des Tweets erzielt. Hier wurde eine Accuracy von 61% und ein F1-Wert von 60% erreicht.

5.6 Alternative Analagestrategie

Würde die Accuracy für jeden Tweet als Anlagestrategie auf dem Aktienmarkt verwendet werden, würde das bei 20 Millionen Tweets bis zu 20 Millionen Transaktionen bedeuten. Da diese Strategie unpraktikabel wäre und um besser analysieren zu können, wie die Accuracy jeweils zu Stande kommt, wird das LSTM-Netz im Folgenden zusätzlich anhand einer einfacheren aber realistischeren Anlagestrategie verglichen. Hierbei soll für jedes Unternehmen für jeden Tag entscheiden werden, ob die Aktie des Unternehmens am nächsten Tag steigt oder fällt. Diese Entscheidung ist für jeden Tag wahr oder falsch. Die Ergebnisse lassen sich wieder mit den bereits vorgestellten Metriken bewerten. Diese Strategie sei im Folgenden ”Prädiktion pro Unternehmen” genannt.

Die Prädiktion pro Unternehmen liefert eine Accuracy von 52%, wenn für jeden Tag entschieden wird, ob der Kurs steigt oder fällt. Im Vergleich dazu wird bei jedem Unternehmen im Schnit 51% Accuracy erzielt, wenn für jeden Tweet entschieden wird, wie sich der Kurs entwickelt. Erwartungsgemäß sind die Vorhersagen, die alle Tweets an einem Tag betrachten, öfter korrekt als die Prädiktionen, die nur auf einem Tweet basieren.

Auch beim F1-Wert erzielt die Prädiktion pro Unternehmen mit 38% bessere Ergebnisse als die Prädiktion pro Tweet für jedes Unternehmen. Dieser Wert liegt bei 33%.

Bei der Prädiktion pro Unternehmen steigt der Kurs im Schnitt an 9,2 Tagen und fällt an 11,8 Tagen. Im Vergleich zum gesamten Datensatz sind die beiden Klassen pro Unternehmen nicht gleich stark vertreten.

In den vorherigen Teilen wurde für jeden Tweet einzeln bestimmt, ob der Aktienkurs fällt oder steigt. Hier wurde als Vergleich eine Strategie betrachtet, die zufällig entscheidet, zu welcher Klasse ein Kurs gehört. Damit wird eine Accuracy von 50% erzielt. Da die beiden Klassen bei der Prädiktion pro Unternehmen nicht mehr gleich groß sind, kann die 50%-Marke nicht mehr zum Vegleich genutzt werden.

Daher wird statt dem Zufall eine andere einfache Anlagestrategie verwendet, mit der die Ergebnisse verglichen werden können. Dabei wird für jeden Tag die gleiche Kursveränderung prognostiziert wie am aktuellen Tag. Das heißt, wenn der Kurs heute steigt, wird prognostiziert, dass er auch morgen steigt. Anhand dieser Strategie lässt sich das Ergebnis der Prädiktion pro Unternehmen bewerten.

Die Vergleichsstrategie erreicht eine Accuracy von 64% und ein F1-Wert von 60%. Demnach scheint die Vorhersage pro Unternehmen mit 52% wenig vielversprechend. Allerdings besitzt die Vergleichsstrategie den Vorteil, dass sie Informationen über die Aktienkurse vom Vortag im Testzeitraum besitzt. Die Neuronalen Netze hingegen, erhalten lediglich Informationen aus dem Trainingszeitraum, der zeitlich vor dem Testzeitraum liegt.

5.7 Ergebnisse für DAX- und DJIA-Unternehmen im Vergleich

Der Anteil an Tweets über DJIA-Unternehmen ist etwa zehn mal größer als der an Anteil an Tweets über DAX-Unternehmen. Da die Größe des Datensatzes Auswirkungen auf die Leistungsfähigkeit eines Neuronalen Netzes hat, wurden die Netze mit den Daten zu den DJIA-Unternehmen optimiert. Mit dem LSTM-Netz, welches die besten Resultate lieferte, werden im Folgenden die Ergebnisse für die DAX-Unternehmen vorgestellt.

Die Prädiktion der Kurse der DAX-Unternehmen basierte auf deutschsprachigen Tweets. Die Ergebnisse sind in Tabelle 5.7 dargestellt. Für die drei Datensätze ergab sich die höchste Accuracy für den Datensatz *Name*. Auch dieser lag nur knapp über den 50%, die mit der Entscheidung per Zufall getroffen werden könnte. Hauptsächlich kann das an der geringen Anzahl an Daten im Vergleich zu den DJIA-Unternehmen liegen.

In Tabelle 5.7 sind die Ergebnisse für verschieden viele Tage im Voraus für den Datensatz $\$Symbol$ aufgelistet. Für zwei und drei Tage nach dem Erstellen der Tweets wurde im Schnitt der höchste F1-Wert und die höchste Accuracy erreicht. Interessant ist die Beobachtung, dass auch hier die beste Prognose nicht für einen Tag im Voraus gemacht wurde, wie es vielleicht naheliegend wäre.

	DAX		DJIA	
Datensatz	Accuracy	F1	Accuracy	F1
$\$Symbol$	45,8	34,3	54,0	54,8
Symbol	50,9	42,1	52,9	48,4
Name	51,5	44,0	50,1	51,9

Tabelle 5.5: Die Ergebnisse für DAX- und DJIA-Unternehmen im Vergleich

	DAX		DJIA	
Tage Differenz	Accuracy	F1	Accuracy	F1
1	45,8	34,3	54,0	54,8
2	51,5	63,0	60,1	56,8
3	57,5	52,4	56,8	62,9
4	55,0	62,4	60,6	60,3

Tabelle 5.6: Prognosen für verschieden viele Tage im Voraus

5.8 Vergleich Arbeiten der Literatur

Die besten Ergebnisse dieser Arbeit wurden bei der Prädiktion der Aktienkurse aller Unternehmen im DJIA über vier Tage im Voraus erzielt. Diese werden daher zum Vergleich mit anderen Arbeiten genutzt. Da der verwendete Datensatz für diese Arbeit erstellt wurde, lassen sich die Ergebnisse nur begrenzt mit anderen Arbeiten vergleichen. Twitter unterbindet die Veröffentlichung von Daten, die über deren Plattform gesammelt wurden. Daher sind zu den betrachteten anderen Arbeiten in diesem Bereich keine Datensätze veröffentlicht worden oder diese wurden bereits gelöscht. Im Folgenden wird dennoch versucht, die Ansätze und Ergebnisse anderer Arbeiten mit dieser zu vergleichen.

Eine Arbeit von Sam Paglia aus dem Jahr 2013 versucht mit den Tweets von Nachrichtenunternehmen den Aktienindex S&P 500 am nächsten Börsentag zu prognostizieren. Die Arbeit untersucht die Auswirkungen auf den nächsten Börsentag. Zudem wird in der Arbeit von Paglia versucht, einen Index zu prognostizieren, während in dieser Arbeit jedes Unternehmen eines Indexes separat prognostiziert wird. Mit einem Naive-Bayes-Klassifikator wurde eine Accuracy von 59% erreicht. Precision und Recall lagen bei 50,2% und 20,8%.[17] Das entspricht einem F1-Wert von 29%. Während die Accuracy von dieser Arbeit nur leicht über der von Paglia lag, so wurde mit 60% ein deutlich höherer F1-Wert erreicht.

Die Arbeit von Bollen[14] untersuchte beliebige Tweets, die nicht unbedingt ein Aktienunternehmen erwähnen. Das Sentiment der Tweets wird als Indikator für die Stimmung in der Öffentlichkeit gewählt. Daraus wird für jeden betrachteten Tag prognostiziert, ob der Schlusskurs des DJIA steigt oder fällt. Der Kurs konnte mit einer *Accuracy* von 86,7% vorhergesagt werden. Betrachtet wurden in der Arbeit von Bollen beliebige Tweets, die keine Verbindung zum DJIA oder den darin enthaltenen Unternehmen besitzen müssen. Da so nicht die Kurse der einzelnen Unternehmen im DJIA bewertet werden können, ist ein direkter Vergleich auch hier nur eingeschränkt sinnvoll. Die Accuracy ist jedoch um 26% höher als der in dieser Arbeit erreichte Wert. Das kann an einer Kombination der folgenden drei Erklärungen liegen:

- Die allgemeine Stimmung auf Twitter ist ein besserer Indikator für den Kurs des DJIA, als die Tweets über ein Unternehmen ein Indikator für den Kurs dieses Unternehmens sind.
- Der größere Beobachtungszeitraum der Tweets führt zu einer besseren Klassifizierung. Während diese Arbeit knapp zwei Monate betrachtet, betrachtet die von Bollen über neun Monate.[14]
- Die Sentimentanalyse mit dem GPOMS für die Dimension ”Calm” ist erfolgsversprechender für die Prädiktion von Aktienkursen als die direkte Prädiktion mit dem hier vorgestellten Neuronalen Netz.

6. Zusammenfassung und Ausblick

In der *Einleitung* wurden Donald Trumps Äußerungen auf Twitter und deren Auswirkungen auf den Aktienmarkt erwähnt. Daraufhin wurde die Vermutung aufgestellt, dass auch die Mitteilungen aller Twitter-Nutzer einen Einfluss auf diesen Markt haben können. In der Tat, ließen sich Aktienkurse mit Hilfe von Twitter-Daten zu einem gewissen Maß vorhersagen.

Die vorgestellten Neuronalen Netze erzielten eine Genauigkeit von bis zu 61%. Dabei wurde prognostiziert, ob der Aktienkurs eines Unternehmen in vier Tagen fällt oder steigt.

Es zeigte sich, dass die Vorhersage für zwei- bis drei Tage in die Zukunft bessere Ergebnisse erzielt als für den folgenden Tag.

Zudem stellte sich heraus, dass mit Tweets, die explizit das Aktiensymbol eines Unternehmens erwähnen, eine bessere Prognose geliefert werden kann, als wenn lediglich der Name des Unternehmens genannt wird.

Im Vergleich zu englischsprachigen Tweets, die schon in mehreren Arbeiten untersucht wurden, zeigt sich, dass mit deutschsprachigen Tweets auch Prädiktionen für den deutschen Aktienmarkt möglich sind. Hier wurde für die Prognose drei Tage in die Zukunft eine Accuracy von 58% erreicht. Jedoch ist festzuhalten, dass das Volumen der Tweets in englischer Sprache bedeutend größer ist, als das in deutscher Sprache.

Im Folgenden werden einige Vorschläge gemacht, wie die in dieser Arbeit erzielten Ergebnisse verbessert werden könnten. Bei der Prädiktion mittels Sentimentanalyse ist ein erster möglicher Verbesserungsansatz das Trainieren des RNTN mit Texten, die ebenfalls Störfaktoren wie Umgangssprache und Rechtschreibfehler enthalten. Wenn dadurch ein ausgeglicheneres Sentiment mit weniger negativer Tendenz auf dem Datensatz entsteht, so könnten eventuell auch bessere Ergebnisse erreicht werden.

Für die Verbesserung der direkten Prädiktion werden im Folgenden einige Verbesserungsvorschläge gemacht. Einmal wäre es denkbar, ein Neuronales Netz zu konstruieren, welches mehr Informationen erhält als Tweets und die binäre Klasse des Tweets. Mögliche Zusatzinformationen wären die Aktienkurse vergangener Tage, ein Zeitstempel, wann der Tweet erstellt wurde und weitere Informationen, von denen der Aktienkurs abhängen könnte. Ein Beispiel für letzteren Vorschlag ist der Ölpreis bei Ölunternehmen oder Autoherstellern. Zudem könnte die Anzahl der Follower und der Retweets in die Vorhersage mit einfließen.

Dass diese weitere Informationen zu vielversprechenden Ergebnissen führen könnten, hat eine Vergleichsstrategie in der *Evaluation* gezeigt. Bei der wurde die Entwicklung des Aktienkurs am vorherigen Tag als Prognose für den kommenden Tag gewählt. Damit wurde eine Accuracy von 64% erreicht.

Eine weitere Verbesserung könnte durch Vergrößern des Datensatzes erreicht werden. Denkbar ist es, dass die bestehenden Neuronalen Netze mit einem größeren Datensatz bessere Resultate erzielen. Bei dem vorgestellten Convolutional-Netz hat sich beispielsweise gezeigt, dass ein größeres Netz mit mehr Convolutional-Schichten schnell zu komplex für den verhältnismäßig kleinen Datensatz wird. Mit einem größeren Datensatz könnte das Potenzial eines größeren Netzen eventuell besser genutzt werden.

Der dritte Verbesserungsvorschlag für die direkte Prädiktion richtet sich an die Repräsentation der Tweets. Durch Word Embeddings wurde in dieser Arbeit jedes Wort als Vektor dargestellt. Eine andere Möglichkeit wäre es, einen ganzen Textes als Vektor fester Länge zu repräsentieren. Dieser Vektor enthält Informationen über das gesamte Dokument. Mit diesem Vorgehen lassen sich bei der Sentimentanalyse von Tweets teilweise bessere Resultate erzielen, als mit Word Embeddings.[58]

7. Abkürzungsverzeichnis

Adam	Adaptive Moment Estimation
DJIA	Dow Jones Industrial Average
GloVe	Global Vectors for Word Representation
GPOMS	Google-Profile of Mood States
LDA	Latent Dirichlet Analysis
LIWC	Linguistic Inquiry and Word Count
LSTM	Long Short-Term Memory
MEC	Maximum Entropy Classification
Nadam	Nesterov Adam Optimizer
PLSA	Probabilistic Latent Semantic Analysis
RMSProp	Root Mean Square Propagation
RNN	Rekurrentes Neuronales Netz
RNTN	Recursive Neural Tensor Network
SGD	Stochastic Gradient Descent
SNLP	Stanford Natural Language Processing
S&P 500	Standard & Poor's 500
SVM	Support Vector Machines
TDNN	Time Delay Neural Network

8. Anhang

8.1 Prädiktion mittels Sentimentanalyse

Tabelle 8.1: Bewertung des RNTN pro Tweet

Name	*Accuracy* (%)	*Precision* (%)	*Recall* (%)	F_1 (%)
Apple	50,2	45,2	74,6	56,3
American Express	51,7	43,4	32,1	37,0
Boeing	54,1	38,3	58,8	46,4
Caterpillar	44,6	36,5	51,7	42,8
Cisco	52,0	34,2	45,1	38,9
Chevron	56,0	40,3	60,8	48,4
DuPont	38,1	28,7	75,4	41,5
Disney	46,3	38,7	34,9	36,7
General Electric	63,2	24,6	16,1	19,5
Goldman Sachs	52,9	39,6	35,0	37,1
Home Depot	40,8	34,1	52,2	41,3
IBM	45,8	38,4	76,4	51,1
Intel	44,4	33,9	62,6	44,0
Johnson & Johnson	50,1	29,4	52,4	37,6
JPMorgan Chase	40,4	33,7	74,5	46,4
Coca-Cola	52,4	16,7	15,5	16,1
McDonald's	49,6	46,3	65,9	54,4
3M	49,8	38,5	55,9	45,6
Merck	48,2	23,9	43,8	30,9
Microsoft	46,4	31,4	64,9	42,4
Nike	54,4	38,8	38,6	38,7
Pfizer	42,6	27,9	59,5	38,0
Procter & Gamble	47,7	29,3	50,0	37,0
Travelers Companies	49,4	35,1	76,5	48,1
United Healh	43,8	28,5	59,9	38,6
United Technologies	59,6	29,0	30,0	29,5
Visa	54,1	36,4	48,3	41,5
Wal-Mart	53,9	30,3	29,1	29,7
Exxon Mobil	57,9	38,6	29,3	33,3
Schnitt	49,7	34,1	50,7	39,6

8.2 Max Pooling-Netz

Tabelle 8.2: Ergebnisse des Max Pooling-Netzes pro Unternehmen

Name	*Accuracy* (%)	*Precision* (%)	*Recall* (%)	F_1 (%)
Apple	60,4	61,7	93,2	74,3
American Express	42,5	29,2	78,7	42,6
Boeing	54,6	85,7	3,6	6,9
Caterpillar	58,6	50,0	26,7	34,8
Cisco	49,3	60,7	48,5	53,9
Chevron	83,6	0,0	0,0	0,0
DuPont	46,1	47,1	13,2	20,6
Disney	41,3	44,8	2,9	5,4
General Electric	82,5	23,3	15,6	18,7
Goldman Sachs	49,1	35,1	26,2	30,0
Home Depot	50,4	65,5	26,6	37,8
IBM	39,2	80,3	30,1	43,8
Intel	50,9	43,3	59,3	50,1
Johnson & Johnson	45,4	85,7	3,6	6,9
JPMorgan Chase	49,9	46,8	58,1	51,8
Coca-Cola	45,0	88,2	9,3	16,9
McDonald's	54,4	40,0	1,0	1,9
3M	45,8	100,0	1,3	2,5
Merck	47,3	39,3	54,7	45,7
Microsoft	55,6	48,8	3,7	6,8
Nike	34,2	0,0	0,0	0,0
Pfizer	77,5	36,1	16,7	22,8
Procter & Gamble	53,6	11,1	2,1	3,6
Travelers Companies	31,9	100,0	3,1	6,0
United Healh	59,0	15,6	4,2	6,6
United Technologies	53,3	6,3	11,1	8,1
Visa	52,6	66,9	46,9	55,1
Wal-Mart	64,4	45,5	7,0	12,1
Exxon Mobil	61,1	46,2	1,7	3,3
Schnitt pro Unternehmen	**53,1**	**48,4**	**22,4**	**23,1**
Schnitt pro Tweet	**55,3**	**56,7**	**46,6**	**51,2**

8.3 LSTM-Netz

Tabelle 8.3: Resultate des LSTM-Netzes

Name	*Accuracy* (%)	*Precision* (%)	*Recall* (%)	F_1 (%)
Apple	60,5	61,4	95,6	74,8
American Express	32,9	28,2	95,4	43,6
Boeing	54,2	58,4	6,7	12,0
Caterpillar	57,1	48,3	48,5	48,4
Cisco	55,5	61,4	73,5	66,9
Chevron	81,3	12,5	5,6	7,8
DuPont	48,2	53,3	19,8	28,9
Disney	43,5	60,0	9,3	16,2
General Electric	59,2	12,0	33,9	17,7
Goldman Sachs	49,6	38,0	33,4	35,5
Home Depot	58,7	62,6	67,8	65,1
IBM	46,2	81,5	40,8	54,4
Intel	45,2	41,6	79,1	54,5
Johnson & Johnson	44,2	52,0	10,3	17,2
JPMorgan Chase	46,5	45,4	75,2	56,6
Coca-Cola	39,4	48,2	16,8	24,9
McDonald's	52,9	31,0	3,1	5,7
3M	49,6	78,3	11,5	20,1
Merck	39,5	38,6	83,3	52,8
Microsoft	56,1	51,8	13,3	21,2
Nike	34,7	55,6	1,9	3,8
Pfizer	61,1	23,9	43,6	30,9
Procter & Gamble	53,9	28,3	9,2	13,9
Travelers Companies	38,4	77,3	17,5	28,6
United Healh	55,8	28,6	18,5	22,4
United Technologies	34,4	15,6	57,8	24,5
Visa	55,2	65,0	60,5	62,7
Wal-Mart	51,6	33,9	39,5	36,5
Exxon Mobil	60,2	37,8	4,0	7,2
Schnitt pro Unternehmen	**50,5**	**45,9**	**37,1**	**32,9**
Schnitt pro Tweet	**54,0**	**54,1**	**55,6**	**54,8**

8.4 Convolutional-Netz

Tabelle 8.4: Metriken für das Convolutional-Netz pro Tweet

Name	*Accuracy* (%)	*Precision* (%)	*Recall* (%)	F_1 (%)
Apple	58,5	61,2	88,7	72,4
American Express	30,2	23,8	71,3	35,6
Boeing	54,7	71,4	5,2	9,7
Caterpillar	57,7	48,0	23,3	31,4
Cisco	52,1	62,6	54,0	57,9
Chevron	80,2	0,0	0,0	0,0
DuPont	47,8	53,3	13,2	21,2
Disney	43,1	56,3	10,9	18,2
General Electric	78,6	15,4	14,7	15,0
Goldman Sachs	44,2	32,5	31,6	32,0
Home Depot	57,9	65,5	54,5	59,5
IBM	36,2	77,7	26,4	39,5
Intel	52,9	45,2	63,4	52,7
Johnson & Johnson	44,2	52,2	9,3	15,9
JPMorgan Chase	53,2	49,6	50,8	50,2
Coca-Cola	46,1	68,2	18,6	29,3
McDonald's	50,6	33,6	9,2	14,4
3M	46,1	56,0	9,0	15,5
Merck	49,5	39,5	46,1	42,6
Microsoft	55,2	48,8	22,5	30,8
Nike	33,7	41,2	2,7	5,1
Pfizer	62,9	18,7	25,6	21,6
Procter & Gamble	53,9	25,0	7,1	11,0
Travelers Companies	34,1	65,0	13,4	22,2
United Healh	59,0	21,1	6,7	10,2
United Technologies	44,7	12,5	33,3	18,2
Visa	52,5	66,5	47,4	55,4
Wal-Mart	59,0	33,0	16,3	21,8
Exxon Mobil	55,9	18,4	4,0	6,6
Schnitt pro Unternehmen	**51,5**	**43,5**	**26,9**	**28,1**
Schnitt pro Tweet	**53,7**	**54,5**	**48,0**	**51,0**

Abbildungsverzeichnis

Literaturverzeichnis

[1] N. Popper. A little birdie told me: Playing the market on trump tweets. [Online]. Available: https://www.nytimes.com/2017/02/16/business/dealbook/trump-tweets-stock-market-trading-bots.html?_r=0

[2] S. Greenstone. When trump tweets, this bot makes money. [Online]. Available: http://www.npr.org/2017/02/04/513469456/when-trump-tweets-this-bot-makes-money

[3] X. H. Cambria, Schuller, "New avenues in opinion mining and sentiment analysis," 2013. [Online]. Available: http://sentic.net/new-avenues-in-opinion-mining-and-sentiment-analysis.pdf

[4] V. Pang, Lee, "Thumbs up? sentiment classification using machine learning techniques," 2002. [Online]. Available: http://www.cs.cornell.edu/home/llee/papers/sentiment.pdf

[5] J. Y. W. J. C. C. D. M. A. Y. N. u. C. P. Richard Socher, Alex Perelygin, "Recursive deep models for semantic compositionality over a sentiment treebank," 2013. [Online]. Available: http://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf

[6] Y. Cao, R. Xu, and T. Chen, *Combining Convolutional Neural Network and Support Vector Machine for Sentiment Classification.* Singapore: Springer Singapore, 2015, pp. 144–155. [Online]. Available: http://dx.doi.org/10.1007/978-981-10-0080-5_13

[7] "Sentiment analysis and ontology engineering : An environment of computational intelligence," Cham, 2016. [Online]. Available: http://swbplus.bsz-bw.de/bsz468158650cov.htmhttp://dx.doi.org/10.1007/978-3-319-30319-2

[8] S. Greene and P. Resnik, "More than words: Syntactic packaging and implicit sentiment," in *Proceedings of human language technologies: The 2009 annual conference of the north american chapter of the association for computational linguistics.* Association for Computational Linguistics, 2009, pp. 503–511.

[9] S. Z. Kim, Ganesan, "Comprehensive review of opinion summarization," 2011. [Online]. Available: https://s3-us-west-2.amazonaws.com/mlsurveys/134.pdf

[10] A. Tumasjan, T. O. Sprenger, P. G. Sandner, and I. M. Welpe, "Predicting elections with twitter: What 140 characters reveal about political sentiment." *ICWSM*, vol. 10, no. 1, pp. 178–185, 2010.

[11] A. Culotta, "Detecting influenza outbreaks by analyzing twitter messages," *arXiv preprint arXiv:1007.4748*, 2010.

[12] G. Mishne, N. S. Glance *et al.*, "Predicting movie sales from blogger sentiment." in *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, 2006, pp. 155–158.

[13] E. M. C. Francesco Corea, "The power of micro-blogging: How to use twitter for predicting the stock market," 2015. [Online]. Available: https://ideas.repec.org/a/ejn/ejefjr/v3y2015i4p1-7.html

[14] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," *Journal of computational science*, vol. 2, no. 1, pp. 1–8, 2011.

[15] P. G. S. I. M. W. Timm O. Sprenger, Andranik Tumasjan, "Tweets and trades: the information content of stock microblogs," 2013. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1111/j.1468-036X.2013.12007.x/full

[16] X. Zhang, H. Fuehres, and P. A. Gloor, "Predicting stock market indicators through twitter "i hope it is not as bad as i fear"," *Procedia - Social and Behavioral Sciences*, vol. 26, pp. 55 – 62, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877042811023895

[17] S. PAGLIA, "Using twitter to gauge news effect on stock market moves."

[18] I. N. da Silva, "Artificial neural networks : A practical course," Cham, 2017. [Online]. Available: http://swbplus.bsz-bw.de/bsz477158021cov.htmhttp://dx.doi.org/10.1007/978-3-319-43162-8

[19] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015.

[20] C. B. F. K. C. M. M. S. Rudolf Kruse, Christian Borgelt, *Computational Intelligence, Seite 7*. Springer Fachmedien Wiesbaden, 2015. [Online]. Available: http://link.springer.com/book/10.1007%2F978-3-658-10904-2

[21] (2017) Tensorflow documentation: Reshape. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/reshape

[22] (2017) Grafik convolution-schicht. [Online]. Available: http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/

[23] D. Britz. (2017) Understanding cnn for nlp. [Online]. Available: http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/

[24] (2017) Max pool illustration. [Online]. Available: https://cambridgespark.com/content/tutorials/convolutional-neural-networks-with-keras/index.html

[25] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," *Artificial Neural Networks–ICANN 2010*, pp. 92–101, 2010.

[26] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen netzen," *Master's thesis, Institut fur Informatik, Technische Universitat, Munchen*, 1991.

[27] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE transactions on acoustics, speech, and signal processing*, vol. 37, no. 3, pp. 328–339, 1989.

[28] (2017) Understanding lstms. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[29] (2016) Large text compression benchmark. [Online]. Available: http://www.mattmahoney.net/dc/text.html#1218

[30] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 5, pp. 855–868, 2009.

[31] (2017) Why you should use cross-entropy error instead of classification error or mean squared error for neural network classifier training. [Online]. Available: https://jamesmccaffrey.wordpress.com/2013/11/05/why-you-should-use-cross-entropy-error-instead-of-classification-error-or-mean-squared-error-for-neura

[32] (2017) Mnist for ml beginners. [Online]. Available: https://www.tensorflow.org/get_started/mnist/beginners

[33] (2017) Supervised learning. [Online]. Available: https://www.coursera.org/learn/machine-learning/lecture/1VkCb/supervised-learning

[34] K. W. Günter Daniel Rey. (2017) Neuronale netze. [Online]. Available: http://www.neuronalesnetz.de/downloads/neuronalesnetz_de.pdf

[35] S. Ruder, "An overview of gradient descent optimization algorithms," *CoRR*, vol. abs/1609.04747, 2016. [Online]. Available: http://arxiv.org/abs/1609.04747

[36] M. Li, T. Zhang, Y. Chen, and A. J. Smola, "Efficient mini-batch training for stochastic optimization," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 661–670.

[37] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," *ArXiv e-prints*, Dec. 2012.

[38] T. Dozat, "Incorporating nesterov momentum into adam," Stanford University, Tech. Rep., 2015.[Online]. http://cs229. stanford. edu/proj2015/054 report. pdf, Tech. Rep., 2015.

[39] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2627435.2670313

[40] M. Nielsen, "Neural networks and deep learning," 2017. [Online]. Available: http://neuralnetworksanddeeplearning.com/chap3.html#overfitting_and_regularization

[41] Using stanford corenlp on other human languages. [Online]. Available: http://stanfordnlp.github.io/CoreNLP/human-languages.html

[42] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, C. Potts *et al.*, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, vol. 1631. Citeseer, 2013, p. 1642.

[43] Nyu lecture: Bag-of-words models. [Online]. Available: http://cs.nyu.edu/~fergus/teaching/vision_2012/9_BoW.pdf

[44] H. Wang, D. Can, A. Kazemzadeh, F. Bar, and S. Narayanan, "A system for real-time twitter sentiment analysis of 2012 us presidential election cycle," in *Proceedings of the ACL 2012 System Demonstrations*. Association for Computational Linguistics, 2012, pp. 115–120.

[45] B. Pang and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ser. ACL '05. Stroudsburg, PA, USA: Association for Computational Linguistics, 2005, pp. 115–124. [Online]. Available: https://doi.org/10.3115/1219840.1219855

[46] (2016) Using pre-trained word embeddings in a keras model. [Online]. Available: https://blog.keras.io/using-pre-trained-word-embeddings-in-a-keras-model.html

[47] D. Selivanov, "Glove word embeddings," 2016. [Online]. Available: https://cran.r-project.org/web/packages/text2vec/vignettes/glove.html

[48] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[49] (2017) Stanford nlp data sets. [Online]. Available: https://nlp.stanford.edu/data/

[50] A. Severyn and A. Moschitti, "Twitter sentiment analysis with deep convolutional neural networks," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2015, pp. 959–962.

[51] (2017) About twitter inc. [Online]. Available: https://about.twitter.com/company

[52] (2017) Twitter serach api documentation. [Online]. Available: https://dev.twitter.com/rest/public/search

[53] (2009) On twitter, $ is the new #. [Online]. Available: https://www.wired.com/2009/02/on-twitter-is-t/

[54] (2017) Finance quotes api for yahoo finance (java). [Online]. Available: https://github.com/sstrickx/yahoofinance-api

[55] (2017) Historische kurse von general electric bei yahoo finance. [Online]. Available: https://de.finance.yahoo.com/quote/GE/history?p=GE

[56] Y. Bao, C. Quan, L. Wang, and F. Ren, *The Role of Pre-processing in Twitter Sentiment Analysis*. Cham: Springer International Publishing, 2014, pp. 615–624. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-09339-0_62

[57] X. Z. Johan Bollena, Huina Maoa, "Twitter mood predicts the stock market," 2010. [Online]. Available: https://arxiv.org/pdf/1010.3003.pdf

[58] S. Vosoughi, P. Vijayaraghavan, and D. Roy, "Tweet2vec: Learning tweet embeddings using character-level cnn-lstm encoder-decoder," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 2016, pp. 1041–1044.