Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

FAKULTÄT
FÜR MATHEMATIK, INFORMATIK
UND NATURWISSENSCHAFTEN

Bachelorthesis

Evaluierung neuronaler Architekturen zur Klassifizierung des Operanten Motivtests

Tjorben Gschwander

4gschwan@informatik.uni-hamburg.de
Studiengang Informatik
Matrikelnummer 6680651

Erstgutachter: Prof. Dr. Chris Biemann
Zweitgutachter: Dr. Timo Baumann

Abgabe: 12.2019

Zusammenfassung

Implizite Motive steuern das unbewusste Verhalten von Menschen. Dieses Wissen kann in vielen Bereichen eingesetzt werden, bspw. für die Heilung mentaler Krankheiten. Diese impliziten Motive lassen sich mit Motivtests messen. Vor allem der Operante Motivtest gilt als einer der modernsten. Jedoch sind Motivtests mit einem sehr hohen Zeitaufwand und dementsprechend auch mit einem hohen Kostenaufwand verbunden, da diese nur von ausgebildeten Psychologen ausgewertet werden können. Es wurden acht Modelle evaluiert, wovon das Long Short-Term Memory, kombiniert mit Bag of Words, die besten Ergebnisse erzielte. Dabei wurde für die Klassifizierung der Motiv-Ebene-Kombinationen ein F1-Score von 0,6016 erreicht. Weiterhin wurde ein F1-Score von 0,6561 für die Klassifizierung der Ebenen und ein F1-Score von 0,8202 für die Klassifizierung der Motive gemessen. Ebenfalls zeigt sich, dass das Capsule Network schlechte Ergebnisse auf diesem Datensatz hervorbringt. Weiterhin konnten keine Indizien für eine Abhängigkeit zwischen Motiven und Ebenen gefunden werden.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

B Batch Size

BiGRU Bidirectional Gated Recurrent Unit

BoW Bag of Words

CapsNet Capsule Network

Caps Capsule Network

CBoW Continous Bag of Words

CNN Convolutional Neural Network

DensCaps Dense Capsules

FT fastText

H Hidden Units

LSTM Long short-term memory

LR Learning Rate

MLP Multilayer Perceptron

NLPsych Natural Language Processing for Predicting psychological traits

OOV Out-Of-Vocabulary

OMT Operanter Motivtest

PrimCaps Primary Capsules

PSI-Theorie Theorie der Persönlichkeits-System-Interaktionen

RNN Recurrent Neural Network

SG Skip-Gram

TAT Thematischer Apperzeptionstest

TF-IDF Term Frequency - Inverse Document Frequency

W2V Word2Vec

1 Einleitung

In diesem Kapitel wird der psychologische Kontext dieser Bachelorthesis kurz erklärt und das grundsätzliche Problem dargestellt. Auch die Methode für die Lösung des Problems wird genannt.

1.1 Problemstellung

Menschen handeln aufgrund ihrer Motivation. Dabei ist die Motivation von vielen Eigenschaften abhängig, unter anderem auch von den impliziten Motiven. Implizite Motive (Kapitel 2.1) steuern die Motivation und damit auch das Verhalten von Menschen [Scheffer, 2001]. Es gibt drei implizite Grundmotive, auch *Big Three* genannt [Heckhausen und Heckhausen, 2006]: Das Anschlussmotiv (*need for affiliation*), das sich durch das Aufbauen und Erweitern von Kontakten und Beziehungen charakterisiert, das Leistungsmotiv (*need for achievement*), welches die eigene Leistung und Leistungssteigerung in den Mittelpunkt stellt und das Machtmotiv (*need for power*), das sich durch die positive und negative Einflussnahme auf Mensch und Umwelt beschreiben lässt. Obwohl diese impliziten Motive schwer zu verbalisieren sind [McClelland und Pilon, 1983], gibt es Motivtests (Kapitel 2.3) zur Messung impliziter Motive. Dies geschieht durch operante Methoden, welche keine bestimmte Antwort erzwingen und ein spontanes Verhalten ermöglichen. Dadurch wird der Antwort mehr Raum an Möglichkeiten gegeben. Der *Thematische Apperzeptionstest* (TAT) [Murray, 1943] ist einer der ersten impliziten Motivtests. Beim TAT müssen Testpersonen eine Geschichte zu einem Bild schreiben. Dies nimmt viel Zeit in Anspruch und auch der Informationsgehalt verringert sich nach sechs Bildern [Atkinson, 1958]. Aufbauend auf dem TAT wurde der *Operante Motivtest* (OMT) entwickelt [Kuhl und Scheffer, 1999]. Der OMT präzisiert die Messung dahingehend, dass zu jedem Motiv fünf Ebenen existieren. Diese Ebenen beschreiben die Art und Weise, wie ein implizites Motiv ausgelebt wird. Diese Differenzierung basiert auf der *Theorie der Persönlichkeits-System-Interaktionen* (PSI-Theorie, Kapitel 2.2) [Kuhl, 2001]. Motive und Ebenen existieren unabhängig voneinander. Die Probanden müssen keine Geschichten zu Bildern schreiben, sondern lediglich vier Fragen in kurzen Sätzen oder Stichpunkten beantworten. Trotz der schnelleren Durchführungszeit für die Probanden, bleibt die Auswertung eine zeit- und kostenintensive Tätigkeit [Schüler et al., 2015; Heckhausen und Heckhausen, 2006]. Dieses Zeitproblem gilt generell für Motivtests, sodass die Auswertung von 100 Picture Exercise Story - Tests [Koestner und McClelland, 1992] 50 Stunden dauert [Schultheiss, 2008]. Zum OMT lassen sich keine genauen Zahlen finden, jedoch beschreibt Scheffer

[2001], dass für seine Untersuchungen der OMT von 20 auf sieben Fragen gekürzt werden musste, damit die Untersuchungsdauer deutlich verringert wird.

Diese Zeitintensität ist ein Grund für die geringe Nutzung eines Motivtests, obwohl bspw. eine Relation zwischen den impliziten Motiven und dem Karriereerfolg womöglich existiert [McClelland und Boyatzis, 1982]. Auch existiert eine Verbindung zwischen dem langfristigen sportlichen Erfolg und dem Leistungsmotiv [Wegner und Schüler, 2015]. Die Durchführungszeit lässt sich durch die automatische Bewertung (Kapitel 2.4) mittels neuronaler Netze verringern. Die Auswertungszeit reduziert sich dadurch von einigen Stunden auf einige Sekunden. Diese Bachelorthesis verfolgt diesen Ansatz und verwendet Deep Learning (Kapitel 2.5) als Methode. Bereits in der Vergangenheit wurden mit Deep Learning gute Resultate in schwierigen Problemen errungen, sei es in der Bildverarbeitung, Sprachverarbeitung oder Textverarbeitung [LeCun et al., 2015]. Welche Modelle sich dafür eignen, lässt sich, aufgrund der fehlenden Forschung bezüglich der automatischen Auswertung von psychologischen Motivtests mit Deep Learning (Kapitel 2.6), nicht beantworten [Johannßen und Biemann, 2018].

1.2 Forschungsfragen

Diese Bachelorthesis beschränkt sich auf die Evaluierung einiger Modelle, welche sich in anderen Kontexten bereits bewährt haben. Die unterschiedlichen Modelle bestehen aus einer von drei unterschiedlichen neuronalen Architekturen (Kapitel 3.2), in Kombination mit einer von drei verschiedenen Wort-/Satzrepräsentationen (Kapitel 3.1). Folgende Fragen werden in dieser Bachelorthesis beantwortet:

- Können Deep Learning-Methoden zur automatischen Auswertung des OMTs genutzt werden?
- Kann die Unabhängigkeit zwischen Motiven und Ebenen beobachtet werden?

Am Ende dieser Durchführung soll ebenfalls eine Aussage getroffen werden können, wie akkurat die betrachteten Modelle ungesehene Instanzen vorhersagen.

Mit der Hilfe impliziter Motive lässt sich teilweise das Verhalten von Menschen vorhersagen. Diese Erkenntnis kann in vielen Branchen eingesetzt werden, sei es bei der Strukturierung optimaler Personengruppen, als auch möglicherweise für die Heilung mentaler Krankheiten [Weindl und Lueger-Schuster, 2016]. Diese Arbeit wird nicht alle Arten von Architekturen und Methoden zur Generierung von Wort-/Satzrepräsentationen betrachten können, dadurch würde der Rahmen dieser Arbeit überschritten werden.

1.3 Struktur

In Kapitel 2 werden Grundlagen geschaffen, um den psychologischen Kontext besser zu verstehen. Deep Learning und Textklassifizierung, welches die grundsätzliche Aufgabe dieser Arbeit darstellt, werden ebenfalls in diesem Kapitel behandelt. Für ein besseres technisches Verständnis werden im 3. Kapitel die einzelnen Methoden und Architekturen vorgestellt. Dazu wird zunächst auf die Methoden zur Generierung von Wort-/Satzrepräsentationen eingegangen (Kapitel 3.1), welche die erste Instanz eines Modells darstellen. Danach werden die einzelnen neuronalen Architekturen grundlegend vorgestellt (Kapitel 3.2). Ab Kapitel 4 beginnt der praktische Teil, sodass zunächst der Datensatz analysiert wird und Besonderheiten hervorgehoben werden. Die Art und Weise, wie eine Evaluierung stattfindet, wird im Methodik-Kapitel (Kapitel 5) erörtert. Zuerst wird auf die Vorverarbeitung der Daten eingegangen. Gefolgt von der Beschreibung der Art und Weise, wie Modelle evaluiert werden und welche Eigenschaften grundsätzlich gelten. Bevor jedoch eine Evaluierung stattfinden kann, wird eine Baseline geschaffen. In Kapitel 6 werden die Ergebnisse der Modelle präsentiert. Die Evaluierung dieser Modelle wird in Kapitel 7 behandelt. Dazu werden einige Modelle in einzelnen Unterkapitel gemeinsam betrachtet. Diese Unterkapitel betrachten dabei einen bestimmten Schwerpunkt. Auf der Basis der Evaluierung werden die Forschungsfragen im Diskussions-Kapitel (Kapitel 8) beantwortet und die Kernpunkte aufgegriffen. Im letzten Kapitel wird das Ergebnis dieser Bachelorthesis kurz zusammengefasst (Kapitel 9).

2 Literatur-Review

In diesem Kapitel wird der Kontext dieser Arbeit erläutert. Eine Auseinandersetzung mit dem psychologischen Kontext ist dahingehend wichtig, dass das Messen eines Motivs bzw. einer Ebene durch geschulte Psychologen geschieht und es zum Verständnis, weshalb die Modelle den OMT auswerten können, beitragen kann. Das technische Wissen, welches in diesem Kapitel behandelt wird, beschreibt die Grundaufgabe und die grundlegende Methode dieser Bachelorthesis.

2.1 Motive

Die Motivforschung ist ein Teilbereich der Motivationsforschung [Heckhausen und Heckhausen, 2006]. Die Motivationsforschung „*versucht, die Richtung, Persistenz und Intensität von zielgerichtetem Verhalten zu erklären*“ (ebd. S. 4). Die Motivation einer Person ergibt sich aus den situativen Anreizen und den persönlichen Präferenzen (ebd.). Als situativer Anreiz gilt die Tätigkeit selbst, das Ergebnis einer Tätigkeit oder auch verschiedene Arten von Ergebnis-Folgen (Konsequenzen). Zu den personenbezogenen Faktoren gehören: Bedürfnisse, Motivdispositionen (implizite Motive) und Zielsetzungen (explizite Motive) [Heckhausen und Heckhausen, 2006]. Explizite Motive werden durch rationale Anreize aktiviert und ergeben sich aus den bewussten Werten, Erwartungen und Zielvorstellungen. Implizite Motive sind unbewusste Ziele, welche den Lebensbereich definieren und dahingehend auch das Ziel der investierten Zeit, Energie und Hoffnung beschreiben [Scheffer, 2001]. Sie sind Verhaltenstendenzen, welche in der frühkindlichen Phase durch emotionale Einflüsse gebildet werden. Aus diesem Grund sind implizite Motive auch schwer zu verbalisieren [McClelland und Pilon, 1983]. Implizite Motive lassen sich nicht bewusst steuern, sie werden durch affektive Anreize aktiviert [Schüler et al., 2015]. Aus dieser Erkenntnis entstand die Idee, operante Tests zur Messung impliziter Motive zu nutzen.

Anschlussmotiv Das grundsätzliche Bedürfnis des Anschlussmotivs ist der natürliche Anreiz, mit Menschen einen persönlichen und verlässlichen Kontakt zu etablieren und zu erweitern. Auch die Vermeidung der Einsamkeit charakterisiert das Anschlussmotiv [Baumeister und Leary, 1995]. Dieses implizite Motiv wird auch Bindungsmotiv genannt. Der Fokus der Wahrnehmung und Verhaltenssteuerung dieses Motivs richtet sich auf das Suchen, Austauschen und Aufrechterhalten von Kontakten. Alle anschlussmotivierten

Aktionen sind mit den Gefühlen des Vertrauens, der Sympathie und der Zuneigung verbunden [Heckhausen und Heckhausen, 2006]. Das Motiv kann bspw. durch kooperative Gruppenaktivitäten befriedigt werden. Auch ein einfaches Treffen mit Freunden oder Bekannten befriedigt das Anschlussmotiv. Menschen mit diesem Motiv besitzen bestimmte Merkmale in ihrem Arbeitsstil: Sie sind gesellig, sobald die Arbeit unter entspannten Verhältnissen stattfindet [Winter und Carlson, 1988], aber auch zurückhaltend, sobald die Arbeit unter stressigen Bedingungen zu verrichten ist. Vor allem die Unterschiede im Verhalten gegenüber sympathischen Menschen, im Gegensatz zu negativ fremd wirkenden Menschen, ist deutlich anders, als bei den anderen impliziten Motiven. Menschen mit einer hohen Tendenz zum Anschlussmotiv werden als kompliziert erachtet und als unbeliebt beschrieben, aufgrund ihrer Ängste in sozialen Beziehungen. Anschlussmotiverte agieren oft unsicher und fordern häufiger eine Rückversicherung ihrer Interaktionspartner. Weiterhin lässt sich beobachten, dass die Leistung von Anschlussmotivierten zunimmt, sobald ein Gefallen mit dieser Tätigkeit verbunden ist. Das Ziel hinter solchen Aktionen ist dabei der Ausbau von Beziehungen [Atkinson und O'Connor, 1966].

Leistungsmotiv Das Leistungsmotiv ist das am intensivste untersuchte implizite Motiv [Heckhausen und Heckhausen, 2006]. Murray [1938] beschreibt dieses Motiv mit folgenden Merkmalen: *„To accomplish something difficult"*,*„To do this as rapidly, and as independently as possible"*, *„To overcome obstacles and attain a high standard"*, *„To rival and surpass others"*. Leistungsmotivierte setzen sich mit einem Maßstab auseinander, um eine Verbesserung der eigenen Leistung und die damit verbundene Verschiebung der erreichbaren Leistungsgrenzen zu erzielen [Heckhausen und Heckhausen, 2006]. Leistungssportler besitzen meist ein stark ausgeprägtes Leistungsmotiv. Leistungsmotivierte lassen sich vor allem daran erkennen, dass deren Arbeitsstil sehr *professionell* ist [Heckhausen und Heckhausen, 2006]. Es werden schwierige, aber auch erreichbare Ziele gesetzt, welche sehr konkret definiert sind [Locke und Latham, 1990] und bei welchen die Ergebnisse durch eigene Wirksamkeit erzielt werden können [Scheffer, 2001]. Sie können sich einfacher auf ihre Ziele fokussieren und diese dementsprechend klarer voneinander trennen. Die Maximierung der Effizienz ist ebenso ein wichtiges Ziel. Trotz der leistungsorientierten Arbeit erreichen Leistungsmotivierte, im Gegensatz zu Machtmotivierte, kaum die höchsten hierarchischen Positionen [McClelland und Boyatzis, 1982]. Dies liegt an deren besseren Fähigkeit, Menschen managen zu können.

Machtmotiv Das Kerncharakteristikum des Machtmotives ist, Einfluss auszuüben. Dabei ist sowohl die positive, als auch die negative Art des Einflusses gemeint. Machtmotivierte Menschen sind hilfsbereiter und eher gewillt anderen zu vergeben [Heckhausen und Heckhausen, 2006]. Auch das Begeistern, Motivieren und Beeindrucken gehören zur Einflussnahme [Scheffer, 2001]. Die Einflussnahme kann ebenfalls negativ manipulativ geschehen. Auch impulsives und aggressives Verhalten, als auch extreme Risikobereit-

schaft beschreiben das Machtmotiv [McClelland, 1975]. Hemmungslosigkeit und exzessiver Alkoholkonsum sind ebenfalls zuzuordnen [Scheffer, 2001]. Das Aufsteigen einer sozialen Hierarchie gehört zu den machtmotivierten Aktionen. Je höher die eigene Ebene in einer sozialen Hierarchie, desto mehr und einfacher kann Einfluss ausgeübt werden. Folglich streben Machtmotivierte auch nach Karriereaufstiegen. McClelland und Boyatzis [1982] zeigten, dass in großen US-amerikanischen Konzernen die Machtmotivierten höher aufgestiegen waren, als Personen mit einem vergleichsweise niedrigen Machtmotiv. Diese Messung wurde in einem Intervall von acht Jahren vollzogen (0, 8, 16).

2.2 PSI-Theorie

Die Theorie der *Persönlichkeits-System-Interaktionen* (PSI-Theorie, [Kuhl, 2001]) bringt verschiedene Annahmen aus der Persönlichkeitstheorie zusammen und beschreibt das Verhalten von Menschen. Dabei differiert diese Theorie von der klassischen Psychologie [Kuhl, 2004][1]. In dieser Theorie wird das Wechselspiel vier verschiedener kognitiver Makrosysteme beschrieben. Dabei legt die PSI-Theorie den Fokus auf die Variabilität der Konfiguration dieser Makrosysteme und deren Verbindungen [Schlebusch et al., 2006]. Diese lassen sich in bestimmten Gehirnarealen verorten [Kuhl und Strehlau, 2014]. Die intuitive Verhaltenssteuerung ist das erste kognitive Makrosystem aus der PSI-Theorie. Dieses System befindet sich im hinteren Teil der rechten Hemisphäre. Es ist als Einziges dazu fähig, Aktionen ausführen zu können. Es agiert im Unterbewusstsein und speichert alle intuitiven Handlungsabläufe. Weiterhin besitzt es ein eigenes Wahrnehmungssystem, wodurch Informationslücken in einer intuitiven Handlung gefüllt werden [Kuhl, 2004]. Die intuitive Verhaltenssteuerung ist bspw. grundsätzlich in einer Unterhaltung aktiv. Dieses System steht in Konkurrenz zum Intentionsgedächtnis. In Konkurrenz deshalb, da nur eines der beiden Makrosysteme zurzeit aktiviert sein kann. Sowohl das Denken und Planen, als auch das Speichern dieser Informationen geschehen im Intentionsgedächtnis [Kuhl und Strehlau, 2014]. Dieses System befindet sich im linken präfrontalen Cortex. Es kann erst dann aktiviert werden, sobald eine Aktion nicht unmittelbar ausgeführt werden muss oder kann, aber das Ziel für eine längere Zeit aufrechterhalten werden muss [Kuhl und Strehlau, 2014]. Eine unmittelbare Ausführung ist gehindert, sobald Schwierigkeiten oder Zielkonflikte auftreten. Um eine Intention ausführen zu können, muss diese an die intuitive Verhaltenssteuerung gesendet werden. Dazu muss eine Art Aktivierungsenergie erbracht werden. Diese Hemmung ist notwendig um die Ausführung von impulsiven und unüberlegten Intentionen zu verhindern. Explizite Motive sind dem Intentionsgedächtnis zuzuordnen [Schlebusch et al., 2006].

Um alle passenden Erfahrungen abrufen zu können, ist das Extensionsgedächtnis nötig. Das Extensionsgedächtnis kann als ein semantisches Netzwerk betrachtet werden,

[1] https://www.psi-theorie.com/app/download/11284415293/Eine%20neue%20Pers%C3%B6nlichkeitstheorie%20PSI-Theorie-light%20Julius%20Kuhl-1.pdf?t=1552669290

welches passende Erfahrungen zur jeweiligen Situation zur Verfügung stellt. Die impliziten Repräsentationen von Erfahrungen sind oftmals schwer zu verbalisieren [Schlebusch et al., 2006], da es sich auch um emotionale Bedeutungen handeln kann. Das Extensionsgedächtnis agiert, wie die intuitive Verhaltenssteuerung, im Unterbewusstsein. Die impliziten Motive sind dem Extensionsgedächtnis zuzuordnen [Schlebusch et al., 2006]. Es befindet sich im vorderen Teil der rechten Hemisphäre. Das vierte und letzte kognitive Makrosystem ist das Objekterkennungssystem, welches fähig ist, den Fokus auf konkrete Einzelheiten setzen zu können. Dadurch lassen sich bspw. einzelne Sinneseindrücke wahrnehmen [Kuhl, 2001]. Dieses System befindet sich in der linken hinteren Hemisphäre und rückt besondere Einzelheiten in den bewussten Vordergrund. Dies ist vor allem wichtig, um auf Gefahren aufmerksam zu machen. Damit diese Einzelheiten abgespeichert werden, müssen die Informationen ins Extensionsgedächtnis gelangen. Somit stehen das Objekterkennungssystem und das Extensionsgedächtnis in Konkurrenz zueinander, sodass das Zusammenspiel beider Systeme durch den Verlust des Fokus realisiert wird. Dadurch können erkannte Einzelheiten im Extensionsgedächtnis gespeichert werden und in anderen Kontexten wiedererkannt werden. In Abbildung 2.1 werden die vier Makrosysteme dargestellt und anhand eines Gehirns lokalisiert.

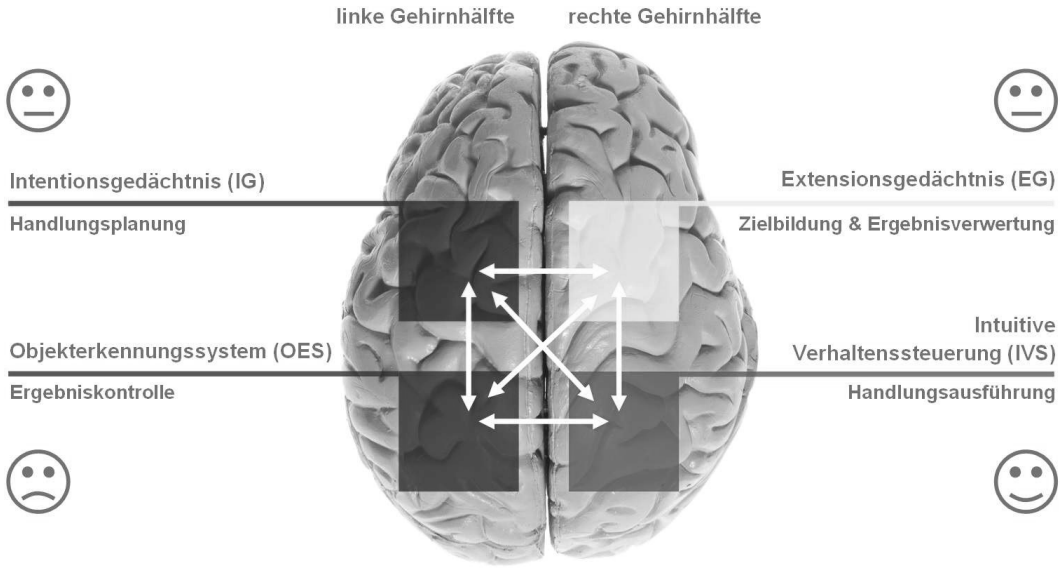

Abbildung 2.1: Dieses Bild stellt die vier Makrosysteme dar und in welchem Bereich des Gehirns diese sich befinden. Dieses Bild stammt von `https://meck.de/methoden.php`, Aufgerufen: 07.11.2019.

Eine zentrale Aussage der PSI-Theorie ist, dass die Aktivierung der kognitiven Makrosysteme von Affekten (Emotionen) abhängig ist. Ein positiver Affekt aktiviert die intuitive Verhaltenssteuerung. Sobald dieser gehemmt wird, bspw. durch das plötzliche Auftreten einer schwierigen Situation, aktiviert sich das Intentionsgedächtnis. Hält man den Verlust des positiven Affekts aus (Frustrationstoleranz), dann ist genug Zeit vor-

handen, um eine Absicht zu bilden. Wird das Gefühl verstärkt, dann aktiviert sich die intuitive Verhaltenssteuerung und die Intention kann ausgeführt werden. Diese Verstärkung kann vom Individuum selbst vorgenommen werden, bspw. durch Selbstmotivation. Ist ein negativer Affekt vorhanden, so aktiviert sich das Objekterkennungssystem, welches den Fokus auf Unstimmigkeiten lenkt. Dadurch geht der Zugang zum Extensionsgedächtnis verloren. Wird dieser Affekt gehemmt, so erlangt man Zugriff auf das Extensionsgedächtnis. Die Fähigkeit, positive oder negative Affekte zu hemmen oder zu verstärken, wird im persönlichen Wachstum erlernt, vor allem im Kindesalter [Kuhl, 2001; Solzbacher und Calvert, 2016]. Positive und negative Affekte können koexistieren. In Abbildung 2.2 wird die PSI-Theorie dargestellt.

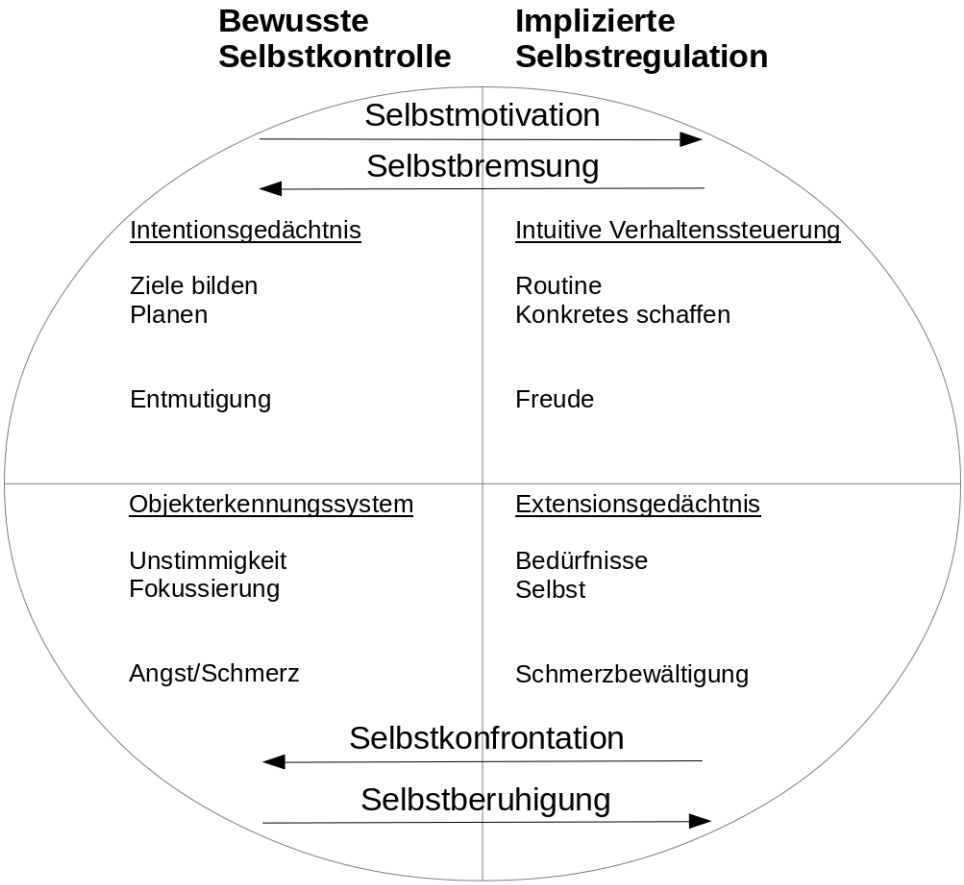

Abbildung 2.2: Diese Darstellung zeigt die vier Makrosysteme, welche in einen bewussten und unbewussten Bereich aufgeteilt werden (Vertikale Teilung). Die horizontale Teilung bezieht sich auf den benötigten Affekt. Die vier Makrosysteme werden durch zwei Stichpunkte und einer möglicherweise vorhandenen Stimmung beschrieben. Dieses Bild ist angelehnt an Abbildung 1 von Kuhl [2004].

2.3 Operanter Motivtest

Um implizite Motive messen zu können, werden operante Motivtests verwendet [Heckhausen und Heckhausen, 2006]. Die Messung geschieht mittels der Interpretation gegebener Antworten. Diese entstehen durch die Beantwortung bestimmter Fragen, welche mehr Offenheit für unterschiedliche Antwortmöglichkeiten bieten. Diese Offenheit wird durch vieldeutiges Testmaterial ermöglicht.

Als Testmaterial werden vor allem Bilder benutzt, welche eine mehrdeutige Situation darstellen. Das Erfragen von Gefühlen oder Gedanken einzelner Personen im Bild ermöglicht ebenfalls einen großen Raum für Antwortmöglichkeiten, aufgrund der niedrigen Beschränktheit. Die gegebenen Antworten basieren auf Erfahrungen und des daraus resultierenden impliziten Motivs [Schüler et al., 2015]. Während der Ausführung darf der Testperson nicht bewusst sein, dass Rückschlüsse über das implizite Motiv gezogen werden, denn mangels stabiler Selbsteinschätzung würden die impliziten Motive weniger zutreffen [Heckhausen und Heckhausen, 2006]. Je spontaner eine Antwort ist, desto eher spiegelt die Antwort das implizite Motiv wider [Heckhausen und Heckhausen, 2006]. Dies ist auch der Unterschied zu den expliziten Fragebögen, welche zum Messen expliziter Motive genutzt werden. Bei diesen Tests ist der Testperson bewusst, wozu der Fragebogen dient. Die Selbsteinschätzung eines expliziten Motivs ist stabil und daher mittels direkter Fragen messbar.

Als eines der ersten operanten Motivtests gilt der TAT. In diesem Test wurden den Testpersonen Bilder gezeigt, zu denen jeweils eine Geschichte geschrieben werden soll. Es werden 20 Bilder empfohlen [Murray, 1943], wobei meistens nur sechs präsentiert werden konnten, da der Informationsgehalt sich danach verringert [Atkinson, 1958]. Im TAT wird nach den drei Grundmotiven klassifiziert. Kritisiert wird, dass dieser Motive deckungsgleich mit Bedürfnissen sieht [Scheffer, 2001]. Eine weitere Kritik ist die Inkonsistenz von Motivmessungen, welches laut Scheffer [2001] an der Vermischung von stabilen und aktuellen Bedürfnissen liegt. Der stabile Bedürfniskern bildet die Basis der impliziten Motive. Diese Vermischung wird durch die verwendeten Bilder hervorgerufen. Auf diesen Motivtest baut der OMT auf und gilt als eines der modernen Motivtests [Heckhausen und Heckhausen, 2006]. Wie beim TAT wird beim OMT eine Unterscheidung der Grundmotive vorgenommen. Darüber hinaus wird beim OMT der Persönlichkeitsstil durch fünf Ebenen unterschieden. Diese Ebenen basieren auf der PSI-Theorie.

Die Ebenen beschreiben einen Persönlichkeitsstil, während die Motive das Ziel einer Handlung beschreiben. Diese Persönlichkeitsstile ergeben sich aus den Wechselwirkungen der Makrosysteme. Im Folgenden werden die Ebenen kurz beschrieben. Die erste Ebene beschreibt die aktive Bewältigung von Problemen. Dabei zielt die Handlung auf die Befriedigung des Selbst ab, da dieser Persönlichkeitsstil einen negativen Affekt

hemmt und dadurch das Extensionsgedächtnis aktiviert, welches die eigenen Bedürfnisse vermittelt. Gleichzeitig wird auch ein positiver Affekt generiert, welcher die intuitive Verhaltenssteuerung aktiviert und dadurch eine Handlung unmittelbar vollzogen wird. Anreizgesteuert und ein extrinsischer Charakter beschreiben die zweite Ebene. Es handelt sich ebenfalls um eine aufsuchende Variante. Als aufsuchend werden Ebenen beschrieben, welche eine Befriedigung eines impliziten Motivs aufsuchen und diese nicht verweigern. Diese Ebene ist aber nicht gesteuert durch das eigene Bewusstsein, sondern geschieht intuitiv (unbewusst). Dies liegt daran, dass nur die intuitive Verhaltenssteuerung aktiviert ist. Die dritte Ebene beschreibt einen Persönlichkeitsstil, welcher vor allem das Extensionsgedächtnis als dominierendes Makrosystem benutzt. Die Umsetzung einer Handlung gestaltet sich schwierig, aufgrund der nicht aktivierten intuitiven Verhaltenssteuerung. Diese Konstellation führt zu einer optimistischen Beschreibung einer Schwierigkeit, aber nicht zu einer aktiven Bewältigung dieser. Bei der vierten Ebene ist eine negative Stimmung zu erkennen. Diese negative Stimmung wird durch das Hemmen des positiven Affekts hervorgerufen. Dadurch ist das Intentionsgedächtnis aktiviert, welches durch Denken und Anstrengung das Problem bewältigen möchte, sodass möglicherweise doch noch ein positiver Affekt gewonnen werden kann. Vor allem die aktive Suche nach dieser Möglichkeit charakterisiert diesen Persönlichkeitsstil und unterscheidet sich von der fünfte Ebene. In dieser Ebene wird keine Aktion unternommen die vorherrschende negative Stimmung zu bewältigen. Dadurch entsteht keine Bedürfnisbefriedigung und es entsteht ein Gefühl der *Leere* und *Hilflosigkeit* [Scheffer, 2001]. Eine visuelle Abbildung der Ebenen kombiniert mit den Motiven findet sich in Tabelle 2.1.

Dominant Motive	Dominant Macrosystem	Description
Affiliation	EM + IBC	*1.1 There are always people in my life with whom I can share really intimate feelings.*
	IBC	*1.2) I enjoy being physically near to others.*
	EM	*1.3) When a relationship becomes difficult I can generate new energy.*
	IM	*1.4) I expect a lot from a romantic relationship.*
	OR	*1.5) When I meet someone new I am often afraid that they will not like me.*
Achivement	EM + IBC	*2.1) I need to like new tasks otherwise I don't bother.*
	IBC	*2.2) I can fully identify with most of the tasks I take on.*
	EM	*2.3.) The most difficult tasks attract me most.*
	IM	*2.4) My will to achieve is insatiable.*
	OR	*2.5) A failure leaves me sapped of energy.*
Power	EM + IBC	*3.1) I just know intuitively how I can show my style.*
	IBC	*3.2) I enjoy feelings of superiority.*
	EM	*3.3) If somebody interrupts me in an impolite or reprimanding manner I'll give them a proper response.*
	IM	*3.4) If I want to convince somebody of something, I carefully consider what will work best with them.*
	OR	*3.5) If somebody acts in a very self-confident manner I keep myself in the background.*

Note: EM = extension memory; IBC = Intuitive behavior control; OR = discrepancy-sensitive object recognition; IM = intention memory (explicit representation of difficult intentions)

Tabelle 2.1: Diese Tabelle beschreibt die Ebenen innerhalb eines Motivs. Dabei ist auch zu erkennen, welche Makrosysteme aktiviert sind. *IBC* ist dabei die intuitive Verhaltenssteuerung, *IM* das Intentionsgedächtnis, *EM* das Extensionsgedächtnis und *OR* das Objekterkennungssystem. Sie wurde zu Darstellungszwecken leicht abgeändert, sodass sie vereinheitlicht wurde. Verwendet wird Tabelle 2 aus dem OMT Manual [Kuhl und Scheffer, 1999].

Falls ein Motiv nicht ermittelt werden kann, so wird das Null-Motiv verwendet. Die Verwendung des Null-Motivs ist dahingehend wichtig, sodass Ebenen und Motive unabhängig voneinander existieren können. Heißt, ein nicht existierendes Motiv schließt keine Ebene aus. Analog dazu existiert auch die Null-Ebene. Beim OMT werden den Testpersonen 15-20 Bilder gezeigt, mit jeweils vier Fragen. Abbildung 2.2 zeigt Beispielbilder und Beispielantworten. Bevor die Fragen beantwortet werden, müssen die Testpersonen eine Hauptfigur markieren und sich eine Geschichte zu jedem Bild ausdenken. Danach werden folgende vier Fragen beantwortet:

1. *Was ist für die Person in dieser Situation wichtig und was tut sie?*
2. *Wie fühlt sich die Person?*
3. *Warum fühlt sich die Person so?*
4. *Wie endet die Geschichte?*

Diese vier Fragen sollen in kurzen Sätzen oder Stichpunkten beantwortet werden. Da im Gegensatz zum TAT nur vier Fragen beantwortet werden sollen, benötigt die Ausführung des OMTs weniger Zeit, bei gleicher Validität [Kuhl und Scheffer, 1999]. Trotzdem ist die Auswertungszeit weiterhin kosten- und zeitintensiv [Schüler et al., 2015]. Scheffer [2001] führte eine Untersuchung durch, bei dem der OMT zum Einsatz kam. Die Untersuchung pro Testperson hat zwischen zwei und sieben Stunden benötigt, wobei die Auswertung des OMTs nur einen Teil der Untersuchung ausmachte. Der OMT müsste aber einen starken Einfluss auf die Untersuchung gehabt haben, da die Anzahl der verwendeten Bilder von 20 auf sieben reduziert wurde. Eine solche Analyse kann nur von geschulten Psychologen vorgenommen werden, sodass der Kostenfaktor in Abhängigkeit von der Zeit steigt. Im Allgemeinen benötigen Motivtests viel Zeit für die Auswertung, sodass bspw. die Auswertung von 100 Probanden bei der Picture Exercise Story 50 Stunden dauert [Schultheiss, 2008].

	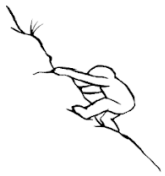	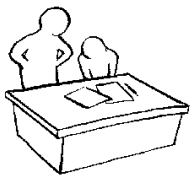	
Was ist für die Person in dieser Situation wichtig und was tut sie?	Halt zu kriegen, nach oben zu kommen, Sie klettert den berg hoch.	Sie beurteilt ein Arbeitsergebnis	Sie will, das das Kind ihr Gesellschaft leistet, und sie erzählt ihm Geschichten
Wie fühlt sich die Person?	angestrengt	mächtig, kontrollierend, unabhängig	Die Anwesenheit des Kindes tut ihr gut
Warum fühlt sich die Person so?	Der Weg ist steil und sie muss viel Kraft aufwenden	Sie trägt die Verantwortung und Entscheidung.	Sie ist nicht gerne allein
Wie endet die Geschichte?	Nach einiger Zeit ist Sie oben angekommen und fühlt sich erleichtert, gut und stolz.	Nach der Kontrolle wird das Arbeitsergebnis modifiziert und die zweite Person führt die besprochenen weiteren Arbeitsschritte aus.	Das Kind bleibt eine Zeitlang
Motiv + Ebene	L3	M3	A4

Tabelle 2.2: Diese Tabelle zeigt drei Beispielbilder aus dem OMT [Kuhl und Scheffer, 1999], welche beantwortet wurden. Die Bewertungen dieser Antworten sind in der letzten Zeile zu finden. In der linken Spalte sind die Fragen aus dem OMT zu sehen. Es sei darauf hingewiesen, dass diese Antworten unverändert aus dem Datensatz stammen. *A* steht für das Anschlussmotiv, *L* für das Leistungsmotiv und *M* für das Machtmotiv.

2.4 Textklassifizierung

80% der Informationen liegen in Textform zur Verfügung, was die Bedeutung der Textklassifizierung begründet. Die Anfänge der Textklassifizierung finden sich in den 1960er Jahren [Sebastiani, 2002]. Vor allem die besser gewordene Hardware ist ein wesentlicher Grund dafür. Anwendung findet die Textklassifizierung bspw. in der Genanalyse, in der Gesundheitsinformatik oder in der Finanzbranche [Xing et al., 2010]. Die Textklassifizierung kann für die Generierung von Metadaten, zur Dokumentenfilterung oder zur Stimmungsanalyse verwendet werden. In dieser Bachelorthesis werden psychologische Motivtests ausgewertet. Die formale Definition der Textklassifizierung kann als eine Zuordnung beschrieben werden:

Sei D die Menge aller Dokumente, ferner sei C die Menge aller Klassen, dann existiert eine Zielfunktion $\phi : D \times C \rightarrow \{True, False\}$. Diese Zielfunktion ordnet einem Dokument und einer Klasse einen Wahrheitswert zu, welcher die Zugehörigkeit eines Dokuments zu einer Klasse beschreibt. Das Ziel der Textklassifizierung ist eine möglichst gute Annäherung an die Zielfunktion, sodass jedem Tupel der korrekte Wert zugeordnet werden kann [Sebastiani, 2002].

Es gibt drei Arten der Textklassifizierung: die binäre Klassifikation, die Multiklassen-Klassifikation und die Multilabel-Klassifikation. In der binären Klassifikation gibt es lediglich zwei Klassen, nach denen klassifiziert werden soll. Jedes Dokument kann nur einer Klasse angehören. Somit gilt für $|C| = 2$ $(\forall d \in D : \exists! c \in C : (d, c) = True)$. In der Multiklassen-Klassifikation gibt es mehr als zwei Klassen ($|C| > 2$), trotzdem kann ein Dokument nur einer Klasse angehören. Diese Art der Textklassifizierung wird in dieser Bachelorthesis angewendet. Bei der Multilabel-Klassifikation kann ein Dokument zu keiner oder mehreren Klassen gehören. Somit gilt $(\forall d \in D : \forall c \in C : \exists! b \in \{True, False\} : (d, c) = b)$.

2.5 Deep Learning

Bis in den späten 1980er war *Knowledge Engineering* im Bereich des Machine Learnings ein häufig genutzter Ansatz [Sebastiani, 2002]. Bei diesem Ansatz wurden Regeln definiert, welche eine Klassifizierung ermöglichen. Diese Regeln benötigen das Wissen von Domänenexperten. Die Anfänge der heutigen datengetriebenen Ansätze finden sich Ende der 1980er, Anfang der 1990er Jahre. Datengetriebene Machine Learning-Ansätze waren, im Gegensatz zu den regelbasierten Ansätzen, fähig, auf der Basis von Rohdaten oder auf extrahierten Features dieser Rohdaten zu klassifizieren. Diese Features wurden gegebenenfalls von Domänenexperten extrahiert. Als Feature werden messbare Eigenschaften beschrieben, welche häufig als Zahlenwert dargestellt werden. Alle genutzten Features

bilden zusammen einen Feature-Vektor, welcher den Eingabewert für die Modelle darstellt. Dennoch waren Machine Learning-Techniken limitiert hinsichtlich der Abhängigkeit dieser Feature-Vektoren, da diese entweder die Rohdaten waren oder ebenfalls das Domänenwissen von Experten benötigten [Alom et al., 2018]. Der Eingabewert kann als eine Repräsentation der zu klassifizierenden Daten angesehen werden. Um diese Limitierung zu umgehen, wurden Modelle entwickelt, welche Repräsentationen lernen konnten, die dann zur Klassifizierung genutzt wurden. Deep Learning ist eine repräsentationslernende Methode und damit eine Disziplin innerhalb des Machine Learnings [LeCun et al., 2015].

Das grundlegende Prinzip bei Deep Learning ist, dass es mehrere Schichten (engl. Layer) gibt, welche den Ausgabewert des vorherigen Schritts weiter abstrahieren, beginnend mit den Rohdaten. Eine Abstraktion ist eine Hervorhebung wichtiger Eigenschaften und eine Vernachlässigung unwichtiger Eigenschaften. Dank dieser Idee wurden Techniken entwickelt, welche damalige State-of-the-Art Machine Learning-Methoden übertroffen haben, bspw. in der Bilderkennung oder Spracherkennung [LeCun et al., 2015]. Laut Alom et al. [2018] wird Deep Learning, aufgrund der Dominanz im Gegensatz zum klassischen Machine Learning, oftmals als universelle Lernmethode angesehen.

Es existieren drei Ansätzen zum Trainieren von Deep Learning-Modellen. Überwachtes Lernen, semi-überwachtes Lernen und unüberwachtes Lernen. Das verstärkende Lernen (engl. Reinforcement Learning) gehört dem semi-überwachten und unüberwachten Lernen an [Alom et al., 2018]. Im Folgenden gilt das Textdokument als universeller Eingabewert eines Modells, aber auch visuelle oder auditive Daten können als Beispiel dienen. Beim unüberwachten Lernen ist während des Trainings nicht bekannt, welche Klassen existieren. Dies impliziert auch, dass während des Trainings nicht bekannt ist, ob ein Dokument zu einer vorhergesagten Klassen gehört. Beim überwachten Lernen ist während des Trainings bekannt, welche Klassen existieren und zu welchen Klassen ein Dokument gehört. Diese Art des Lernens wird in dieser Bachelorthesis angewendet. Beim semi-überwachten Lernen kann die Klasse eines Dokuments während des Trainings bekannt sein. Jedoch muss dies nicht zwangsläufig der Fall sein. Diese Trainingsart vereint das überwachte Lernen und unüberwachte Lernen.

Beim überwachten Lernen wird mittels einer Loss Function (dt. Verlustfunktion; manchmal auch Fehlerfunktion genannt) der Loss (dt. Verlust) errechnet, welcher den Unterschied zwischen dem erwarteten Ausgabewert und dem vorhergesagten Ausgabewert als ein Wert darstellt. Dieser Loss wird genutzt, um die Gewichtungen innerhalb des Modells, mittels Backpropagation [Werbos, 1974], anzupassen. Backpropagation berechnet die Relevanz jedes Gewichtes für den Ausgabewert und inwiefern ein Gewicht angepasst werden muss, um eine Verringerung des Loss-Wertes zu ermöglichen. Dadurch wird, falls eine ähnliche Situation auftritt, die in der Vergangenheit falsch

entschieden wurde, mit einer etwas geringeren Wahrscheinlichkeit noch einmal die gleiche Entscheidung fallen. Das Modell kann als eine ganze Funktion gesehen werden. Bei der Backpropagation wird die Modell-Funktion mit der Loss Function konkateniert, aus der sich eine Funktion ergibt, für die eine partielle Ableitung für jedes Gewicht berechnet wird. Auf der Basis dieser werden die Gradientenvektoren berechnet, welche den steilsten Weg zu einem Minimum zeigen. Um die Gewichtungen anzupassen, wird ein Optimierer (engl. Optimizer) genutzt.

Der Datensatz wird aufgeteilt in eine Trainingsmenge (engl. trainings set), Testmenge (engl. test set), und Validierungsmenge (engl. validation set). Die Trainingsmenge wird verwendet, um die Gewichtungen eines Modells zu trainieren. Die Validierungsmenge wird für das Ermitteln der Hyperparameter genutzt. Dies sind Parameter, welche vor dem Training eines Modells eingestellt werden können. Die Testmenge wird lediglich zum Testen eines Modells genutzt, dabei dürfen keine Änderungen auf der Basis dieser Ergebnisse vorgenommen werden. Dies ist notwendig, um die konkrete Anpassung (overfitting) des Modells an die Trainingsmenge und Validierungsmenge zu vermeiden und die Generalisierung zu erhöhen [Goodfellow et al., 2016]. Die eigenen Modelle werden mit einer Baseline verglichen. Eine Baseline ist das Ergebnis eines einfachen Modells, welches durch das eigene Modell (meist kompliziertere) übertroffen werden soll.

2.6 NLPsych

Der Begriff *NLPsych* findet das erste Mal Benutzung in der Forschungsarbeit von Johannßen und Biemann [2018]. Diese Forschungsarbeit untersucht den aktuellen Forschungsstand im Bereich des „*natural language processing for predicting psychological traits*„ (NLPsych, [Johannßen und Biemann, 2018]). Schon zu Beginn weisen die Autoren darauf hin, dass die Forschung in diesem Bereich sehr fragmentiert ist. Alle gleich vorgestellten Zweige innerhalb des NLPsych sind nach ihrer Forschungsintensität sortiert. Der am intensivste untersuchte Zweig ist *Mental Health*, welcher im Allgemeinen mentale Krankheiten, Suizidversuche oder Krisen behandelt. *Dreams language in dream narratives*, ein weiterer Zweig des NLPsych, beschäftigt sich mit den Beziehungen zwischen Träumen und Charaktereigenschaften und inwiefern sich Traumerzählungen von normalen Geschichten sprachlich unterscheiden. *Mental changes* analysiert die natürlich mentalen Veränderungen, in dem die Sprache von alternden Menschen oder nach lebensverändernden Ereignissen benutzt wird. *Motivation and emotion* ist ein weniger untersuchter Bereich. Zum Thema Motivation wird nur eine Forschungsarbeit genannt, welche lediglich die Veröffentlichung eines Datensatzes thematisiert [Pérez-Rosas et al., 2016]. Es ist hervorzuheben, dass implizite Motive nur einen Teil der Motivation ausmachen und sich diese Bachelorthesis dadurch in einem konkreteren psychologischen Kontext befindet. Alle anderen Forschungsarbeiten, in diesem Zweig,

beziehen sich auf das Thema Emotion. Prinzipiell werden in diesem Bereich Gefühle analysiert. Aber auch Hassrede (engl. Hate Speech) lässt sich wiederfinden. Als Hate Speech werden Aussagen bezeichnet, welche Personen oder Personengruppen erniedrigen, beleidigen oder einschüchtern. Der am wenigsten intensivste untersuchte Zweig ist *Academic Success*. Hier wird untersucht, ob sich akademischer Erfolg aus Texten vorhersagen lässt. Drei von den 15 vorgestellten Forschungsarbeiten verwenden Deep Learning, von denen zwei Textklassifizierungsaufgaben bearbeiten.

Eine weitere erwähnenswerte Forschungsarbeit ist die von Mowery et al. [2016], welche Tweets dahingehend klassifiziert, ob diese einen Hinweis für Depressionen sein können und welche Symptome darin auftreten. Benutzt wurden sechs verschiedene Machine Learning-Klassifizierer. Als Baseline wurde der Keyword-Ansatz gewählt, welcher eine Klassifizierungs-Methode auf Basis von Keywords beschreibt. Tritt ein bestimmtes Wort auf, so wird der gesamte Text in Abhängigkeit dieses Wortes klassifiziert. Ihrer Aussage nach haben sie mit ihrer Arbeit eine gute neue Baseline geschaffen und den Keyword-Ansatz übertroffen.

Rashkin et al. [2018] untersuchten die mentalen Zustände von Charakteren in Kurztexten mit der Hilfe von Künstlicher Intelligenz (KI). Sie entwickelten ebenfalls eine neue Annotationsmethode für mentale Zustände in Kurztexten. Die mentalen Zustände ändern sich durch Reaktionen, auf vorherigen Aktionen (Handlungen), welche aus einer Motivation resultieren. Die Annotationsmethode basiert auf drei psychologischen Theorien: *Hierarchy of needs* [Maslow, 1943], *Basic Motives* [Reiss, 2004] und *Wheel of Emotions* [Plutchik, 1980]. Ferner beschäftigt sich diese Forschungsarbeit mit der Textgenerierung und Textklassifizierung. Bei der Textklassifizierung wurden Term Frequency - Inverse Document Frequency (TF-IDF) [Luhn, 1957], GloVe [Pennington et al., 2014], Convolutional Neural Network (CNN) [Fukushima, 1980], Long Short-Term Memory (LSTM) [Hochreiter und Schmidhuber, 1997], Recurrent Entity Network [Henaff et al., 2016] und Neural Process Network [Bosselut et al., 2018] als Encoder und ein Logistic Regression Classifier als Decoder verwendet. Als Encoder wird der repräsentationslernende Teil eines Modells bezeichnet, während der Decoder auf der Basis dieser Repräsentation klassifiziert.

3 Neuronale Architekturen zur Klassifizierung

Im Folgenden werden die verwendeten neuronalen Architekturen für die Klassifizierung erläutert. Ein Modell kann in zwei Schichten unterteilt werden, sodass in der ersten Schicht die Word Embeddings genutzt werden und in der zweiten Schicht eine neuronale Architektur. Zur Darstellung dieser Modellstruktur dient Abbildung 3. Zuerst werden die Word Embeddings vorgestellt. Da in dieser Arbeit keine Word Embeddings trainiert werden, bildet die zweite Schicht den zu trainierenden Bereich eines Modells. Dieser Bereich enthält die trainierbaren Gewichtungen, welche durch die neuronalen Architekturen definiert werden.

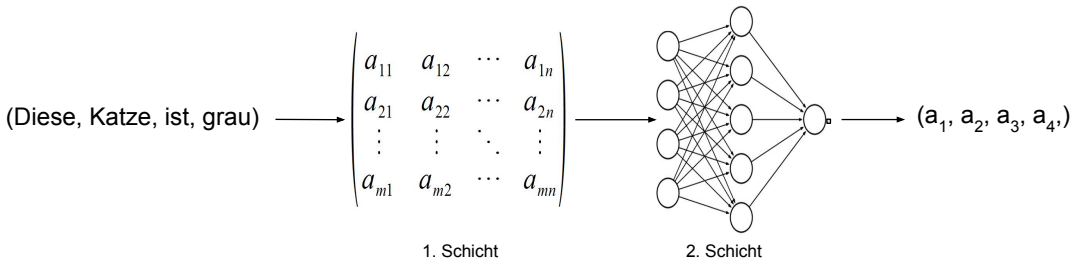

Abbildung 3.1: Diese Darstellung zeigt die verwendete Modellstruktur dieser Bachelorthesis. In der ersten Schicht werden die Word Embeddings verwendet und in der zweiten Schicht die neuronalen Architekturen. Als Eingabewert dienen vorverarbeitete Texte und der Ausgabewert enthält die Wahrscheinlichkeiten für die Klassen der Motiv-Ebene-Kombinationen.

3.1 Word Embeddings

Word Embeddings können Wörter in Wortvektoren konvertieren. Diese Embeddings können sowohl durch mathematische Operationen, als auch durch repräsentationslernende Methoden generiert werden. In diesem Kapitel werden die Methodendetails der Word Embeddings erläutert.

3.1.1 Bag of Words

Bag of Words (BoW) generiert Satzvektoren durch das Zählen von Häufigkeiten einzelner Wörter in einem Dokument [Gaussier und Yvon, 2013]. Bevor ein Satzvektor erstellt werden kann, muss ein Vokabular fixiert werden, welches aus einer Liste der im Datensatz existierenden Wörter besteht. Jeder Index in diesem Vokabular ist einem speziellen Wort zugeordnet. Bei der Konvertierung eines Dokuments zu einem Satzvektor stellt jeder Index in diesem Vektor die Häufigkeit des Wortes dar. Diese Methode führt zu Vektoren mit vielen Dimensionen. Diese Vektoren werden auch One-Hot-Vektoren genannt, da viele Indizes den Wert null haben, weil einzelne Daten lediglich eine kleine Teilmenge aller Wörter im Vokabular enthalten. Aufgrund der Verwendung eines Satzvektors existieren keine Informationen zur Satzstellung.

3.1.2 Word2Vec

Word2Vec-Methoden werden verwendet, um Word Embeddings zu berechnen. Diese repräsentationslernenden Methoden verwenden einen großen Textkorpus. Für jedes einzelne Wort in diesem Korpus wird ein Vektor trainiert. Bevor dies geschieht, wird aus dem Textkorpus ein Vokabular V generiert. Es gibt zwei Arten der Word2Vec-Methode, insofern, dass ein Word Embedding mittels Skip-Gram (SG) oder Continuous Bag of Words (CBoW) berechnet werden kann. Sowohl SG als auch CBoW sind neuronale Netze und bestehen jeweils aus zwei Fully-Connected-Layer. Ein Fully-Connected-Layer besteht aus Neuronen, welche jeweils den gesamten Vektor als Eingabewert bekommen. Als Kontext C ist eine Menge von Wörtern $c_0, \ldots, c_i$ gemeint, welche die Bedeutung eines Wortes beschreiben sollen. Normalerweise sind dies Wörter, die sich unmittelbar beim Zielwort befinden. Der maximale Abstand zum Zielwort ist ein Parameter und muss vor dem Training festgelegt werden. Die Notation in Abbildung 3.2 wird für ein einfacheres Verständnis durch folgende Notation ersetzt: Die Kontextwörter $w(t+i) \in C : j \neq 0$ werden als $c_i \in C$ dargestellt. Das Zielwort $w(t)$ wird als w_j dargestellt.

In Abbildung 3.2 ist zu erkennen, dass CBoW aus einem gegebenen Kontext das Zielwort vorhersagt. Die Kontextwörter $c_0, \ldots, c_i$ werden in die One-Hot-Vektoren $x_{0k}, \ldots, x_{ik}$ konvertiert, welche aus dem vorher generierten Vokabular stammen. Aufgrund der Tatsache, dass diese Architektur aus zwei Fully-Connected-Layer besteht, existieren die zwei Gewichtungsmatrizen $W \in R^{|V| \times N}$ und $W' \in R^{N \times |V|}$. Die Dimension N gibt die Dimension der Wortvektoren an und muss vor dem Training festgelegt werden. Im ersten Berechnungsschritt wird der Vektor h berechnet, in dem die One-Hot-Vektoren der Kontextwörter mit der Matrix W multipliziert werden und der Durchschnitt daraus gebildet wird: $h = \frac{1}{|C|} W^T (x_{0k}, + \cdots +, x_{ik})$. Im nächsten Schritt wird der Vektor h mit W' multipliziert, woraus der One-Hot-Vektor x_j berechnet wird: $x_j = W'h$. Dieser Vektor wird mit der *Softmax*-Funktion zu einer Wahrscheinlichkeitsverteilung berechnet. Der Index mit dem höchsten Wert repräsentiert das Zielwort. Mittels

Loss Function wird der Loss berechnet und die Gewichtungen angepasst. Am Ende des Trainings ist die Gewichtungsmatrix W das später verwendete Word Embedding.

SG benutzt einen gegenteiligen Ansatz. In dieser Architektur wird das Zielwort w_j bzw. der One-Hot-Vektor x_j als Eingabewert verwendet. Die Größe der Gewichtungsmatrizen bleiben die gleichen, wie bei der CBoW-Architektur. Im ersten Schritt wird der Vektor h berechnet: $h = W^T x_j$. Im nächsten Schritt wird der Vektor h mit der Gewichtungsmatrix W' multipliziert um einen One-Hot-Vektor zu erhalten, welcher die Wahrscheinlichkeiten der Kontextwörter beinhaltet: $x_i = W'h$. Der Vektor x_i wird ebenfalls mit der *Softmax*-Funktion zu einer Wahrscheinlichkeitsverteilung konvertiert. Für jeden One-Hot-Vektor der Kontextwörter $x_{0k}, \ldots, x_{jk}$ wird der Loss berechnet und die Gewichtungen angepasst [Rong, 2014].

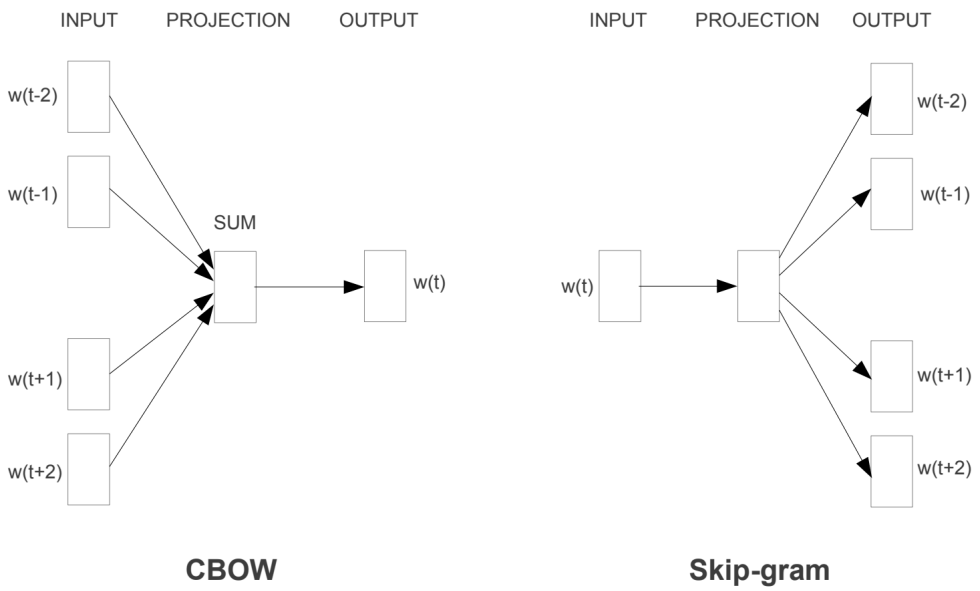

Abbildung 3.2: Diese Darstellung zeigt die CBoW- und SG-Architektur. Dabei repräsentiert $w(t)$ das Zielwort und $w(t+i)$ die Kontextwörter. Diese Architekturen stammen von Mikolov et al. [2013].

Die SG-Architektur eignet sich gut, wenn man eher einen kleineren Textkorpus zum Trainieren verwendet [Mikolov et al., 2013]. Weiterhin generiert diese Architektur qualitativ bessere Wortvektoren für seltene Wörter, wobei die CBoW-Architektur bessere Wortvektoren für häufige Wörter generiert. Ebenfalls ist SG rechenintensiver als CBoW [Mikolov et al., 2013]. Dank diesen Word2Vec-Methoden entstehen Vektorrepräsentationen, die eine gewisse Semantik aufweisen. Beispielsweise ist der Wortvektor von *Deutschland* ähnlich dem Wortvektor *Frankreich*. Ebenfalls kann man auf diesen Vektoren algebraische Operationen anwenden, sodass $vector[\text{"}biggest\text{"}] - vector[\text{"}big\text{"}] + vector[\text{"}small\text{"}] = vector[\text{"}smallest\text{"}]$ ergibt, zumin-

dest wenn das Word Embedding ausreichend gut trainiert wurde. Word2Vec hat den Nachteil, dass während des Trainings nicht alle Wörter verfügbar sind. Es entstehen nur Wortvektoren für alle Wörter aus dem verwendeten Datensatz. Fehlende Wörter in einem Word Embedding werden Out-Of-Vocabulary (OOV) Wörter genannt. Dies ist ein Problem, da OOV-Wörter in der Testmenge auftreten können und dadurch das Modell das Wort nicht erkennen kann. Dies kann zur Nichterkennung von Schlüsselwörter führen. Meist wird ein Null-Vektor für alle OOV-Wörter benutzt. Repräsentationslernende Methoden haben meist weniger Dimensionen als zählende Methoden. Word2Vec ist eine etablierte Word Embedding Methode in der Textklassifizierung. Für die Wahl eines geeigneten Word2Vec-Modells wurden zwei Word Embeddings in Kapitel 5.2 verglichen.

3.1.3 FastText

FastText [Bojanowski et al., 2017] ist ebenfalls eine repräsentationslernende Methode zur Generierung von Word Embeddings. Grundsätzlich wird SG als neuronales Netz verwendet. Der Unterschied zum Word2Vec, liegt in der Verwendung einer anderen Abstraktionsebene. Statt Wortvektoren werden Zeichenkettenvektoren verwendet, wobei das Wort selbst ebenfalls in der Menge enthalten ist. Die Länge der Zeichenketten wird vor dem Training bestimmt. Sollte ein Wort im fertigen Word Embedding nicht vorhanden sein, so wird die Summe der Zeichenkettenvektoren gebildet und als Wortvektor verwendet. Das Problem mit den OOV-Wörtern kann im Allgemeinen ignoriert werden, da die Wahrscheinlichkeit für nicht gesehene Zeichenketten gering ist. Da ein Wortvektor aus der Summe aller Zeichenkettenvektoren besteht, müssten alle Zeichenketten nicht existieren, damit ein Wort als OOV-Wort gilt. Die Wortvektoren aus fastText haben die Eigenschaft, dass die Ähnlichkeit zwischen Wörtern auch in den Wortvektoren vorhanden sein kann. Vor allem bei einer Sprache mit vielen Präfixen und Suffixen kann diese Eigenschaft von fastText als ein Vorteil gelten. Dabei sei zu beachten, dass auch Wörter existieren, welche ähnlich geschrieben werden, aber keine ähnliche Bedeutung besitzen. FastText gilt momentan als State-of-the-Art Word-Embedding-Methode. Das in dieser Arbeit verwendete fastText-Modell [Grave et al., 2018] wurde auf einen Datensatz bestehend aus Wikipedia-Artikeln und anderen freien zugänglichen Datenquellen trainiert. Grave et al. [2018] trainierten Word Embeddings für 157 Sprachen. Diese Vielfalt an Modellen existiert momentan nicht für Word2Vec. Es gilt zu erwähnen, dass bereits weitere repräsentationslernende Methoden existieren, die in manchen Bereichen besser sind als die gerade eben vorgestellten Methoden. Darunter zählen ELMo [Peters et al., 2018], BERT [Devlin et al., 2019] und XLNet [Yang et al., 2019]. Diese Methoden bilden Wortvektoren in Abhängigkeit zum Kontext. Beispielsweise hat das Wort "Bank" mehrere Bedeutungen (Geldinstitut, Parkbank). Bei Word2Vec und fastText würden beide Bedeutungen durch einen Wortvektor repräsentiert werden. Bei ELMo, BERT und XLNet können diese Bedeutungen durch verschiedene Wortvektoren (für dasselbe Wort) repräsentiert werden.

3.2 Neuronale Architekturen

3.2.1 Multilayer Perceptron

Ein Multilayer Perceptron (MLP) ist eine einfache neuronale Architektur [Goodfellow et al., 2016], welche standardgemäß aus zwei Fully-Connected-Layer besteht. Das MLP ist ein Feedforward Netz, dadurch sind in dieser Architektur keine Zyklen existent und Informationen können nicht über Iterationen hinweg gespeichert werden. Das *universal approximation theorem* [Hornik et al., 1989] besagt, dass sich ein Feedforward Netz jeder Funktion annähern kann. Mindestens ein Hidden Layer mit einer Aktivierungsfunktion wird benötigt, um sich jeder Funktion annähern zu können. Es ist aber nicht garantiert, dass jede Architektur sich der Zielfunktion stark annähert, da dies abhängig von der Anzahl und Qualität der Testdaten ist, als auch von der Größe der Architektur [Goodfellow et al., 2016]. Das MLP akzeptiert nur Vektoren (Tensoren mit Rang *1*) als Eingabewert. In dieser Bachelorthesis wird die Standard-Implementierung benutzt. Als Aktivierungsfunktion wird ReLu [Hahnloser et al., 2000] verwendet.

3.2.2 Long Short-Term Memory

Das Long Short-Term Memory (LSTM) ist eine Weiterentwicklung des Recurrent Neural Network (RNN). Das RNN ist ein anderer Architekturtyp, als das Feedforward Netz. Das RNN besitzt, anders als ein Feedforward Netz, einen Zyklus. Dieser Zyklus wird durch eine Schleife realisiert. Dadurch wird das Speichern von Informationen über Iterationen hinweg ermöglicht. Die Länge der Sequenz bestimmt die Anzahl der Iterationen. Dabei wird dem RNN in der ersten Iteration die erste Entität aus der Sequenz als Eingabewert gegeben. In der nächsten Iteration kriegt das RNN den Ausgabewert der vorherigen Iteration und die zweite Entität aus der Sequenz als Eingabewert. Der Ausgabewert der letzten Iteration gilt als Ausgabewert der gesamten Sequenz. Der Ausgabewert einer Iteration wird in der nächsten Iteration als Hidden State bezeichnet. Dieses Konzept eignet sich besonders für Sequenzen verschiedener Längen. Das RNN hat den Nachteil, dass das Vanishing-Gradient-Problem [Hochreiter, 1991] auftritt. Dieses Problem beschreibt eine exponentielle Verringerung, abhängig von der Anzahl der Iterationen, des Gradienten, welcher für die Anpassung der Gewichtungen notwendig ist. Es ist vor allem der Gradient für die vorderen Layer, der extrem klein werden kann, da dieser in der Ableitung abhängig ist von den hinteren Layer.

Um das Vanishing-Gradient-Problem zu lösen, wurde das LSTM entwickelt. Es wird ein sogenannter Cell State (dt. innere Zelle) eingeführt, welcher ebenfalls Informationen über Iterationen hinweg speichert. Der Cell State ermöglicht eine stärkere Relation zwischen zwei weit distanzierte Eingabewerten. In jeder Iteration bekommt das LSTM drei Eingabewerte: die aktuelle Entität einer Sequenz x_t, der Ausgabewert der vorherigen Iteration h_{t-1} und c_{t-1}, den Cell State der vorherigen Iteration. Bevor der Aus-

gabewert dieser Iteration berechnet werden kann, muss der Cell State berechnet werden. Es gibt drei Gates (dt. Tore), welche die Veränderung des Cell States und die Einflussnahme auf den Ausgabewert kontrollieren. Das erste Gate ist das Forget Gate. Dieses wurde für das Löschen bzw. Behalten von Informationen im Cell State entwickelt. $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ beschreibt die Berechnung des Forget-Vektors f_t. Dieser Vektor wird später für die Berechnung des neuen Cell States c_t benutzt. Das zweite Gate ist das Input Gate, welches bestimmt, welche neuen Informationen in den Cell State einfließen sollen. Dazu werden die möglichen Informationen $\hat{c}_t$ durch folgende Gleichung berechnet: $\hat{c}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$. Anschließend wird der Input-Vektor i_t folgendermaßen berechnet: $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$. Mit dem Forget-Vektor und dem Input-Vektor lässt sich nun der neue Cell State c_t bilden: $c_t = f_t \cdot c_{t-1} + i_t \cdot \hat{c}_t$. Das Output Gate generiert den Ausgabewert einer Iteration, indem es zuerst den Output-Vektor berechnet: $o_t = \sigma(W_h \cdot [h_{t-1}, x_t] + b_h)$, welcher mit dem neuen Cell State verrechnet wird, um den Ausgabewert h_t zu erhalten: $h_t = o_t \cdot tanh(c_t)$. Der nun gebildete Ausgabewert und der aktuelle Cell State werden der nächsten Iteration übergeben. In Abbildung 3.3 ist eine Darstellung einer LSTM-Zelle zu sehen. Solch eine LSTM-Zelle wird in jeder Iteration durchlaufen. Dank dem Cell State und seinen Gates, speziell das Forget und Input Gate, ist die rekursive Ableitung eines bestimmten Terms gleich eins, sodass der Gradient auch für die früheren Iterationen nicht *verschwindet*.

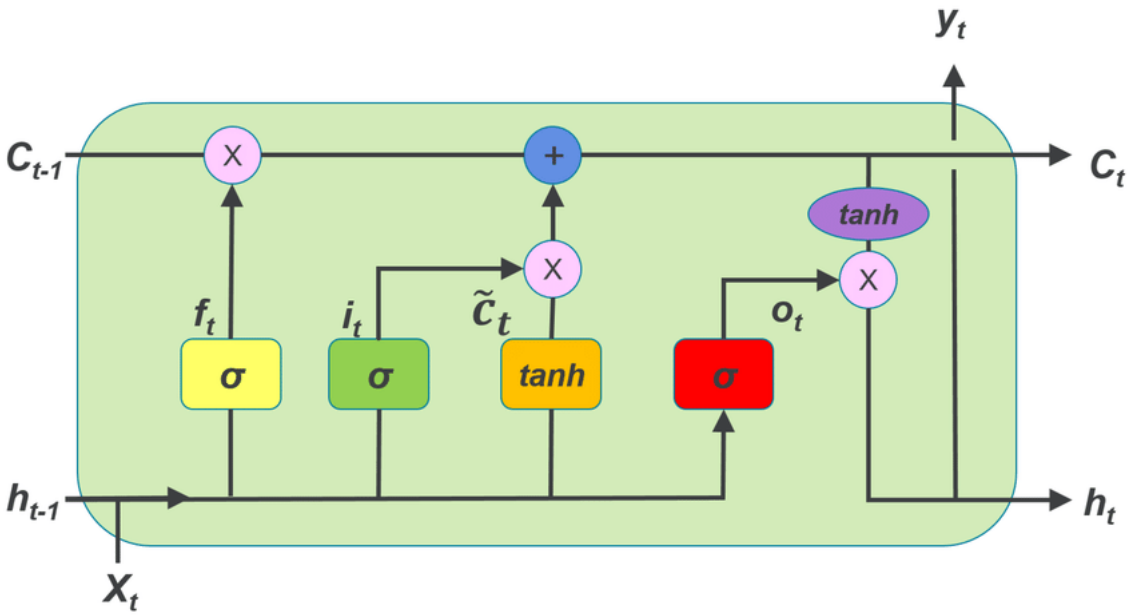

Abbildung 3.3: Diese Darstellung stammt aus Ismail et al. [2018]. Sie zeigt eine LSTM-Zelle, welche in jeder Iteration durchlaufen wird.

Levy et al. [2018] untersuchten den Beitrag des Cell State und seine Gates zum Ergebnis des gesamten LSTMs. Ihre Annahme war es, dass der Cell State mit seinen Gates lediglich eine Unterstützung ist, damit das Vanishing-Gradient-Problem in einem RNN

verringert wird. Die Autoren entwickelten verschiedene Architekturen aus einem LSTM. Es entstand die *LSTM - S-RNN-*, die *LSTM - S-RNN - OUT-*, die *LSTM - S-RNN - HIDDEN* und die *LSTM - GATES*-Architektur. Dabei ersetzten die Autoren das sRNN mit einer normalen linearen Transformation oder entkoppelten das sRNN von dem Cell State. Mit sRNN ist ein simple RNN gemeint, welcher dem RNN in dieser Arbeit gleicht. Das Resultat war, dass die *LSTM - S-RNN*-Architektur, welches statt einem sRNN, eine lineare Transformation besitzt, teilweise ein besseres Ergebnis erzielt hat als ein normales LSTM. Dies deutet darauf hin, dass die Gates und der dazugehörige Cell State mehr als nur eine unterstützende Funktion sind, laut den Autoren.

Diese Architektur gilt als ein Standard in der Textklassifizierung und bildet einen Bestandteil für heutige State-of-the-Art Modelle, bspw. im Modell von Howard und Ruder [2018], welches sehr gute Ergebnisse in verschiedenen Datensätze erreicht hat, unter anderem im AG-Dataset[1]. In dieser Bachelorthesis wird eine Architektur genutzt, welche aus einem LSTM-Layer besteht, gefolgt von einem Dropout-Layer und einem Fully-Connected-Layer. Ein Dropout wird in vielen LSTM-Modellen verwendet, um dem Overfitting entgegenzuwirken [Srivastava et al., 2014]. Der Fully-Connected-Layer dient als Decoder in dieser Architektur, sodass der LSTM-Layer den Encoder darstellt. Diese Aufteilung ist ebenfalls häufig in LSTM-Modellen zu finden.

3.2.3 Capsule Network

Kapseln (engl. capsules) wurden vor allem für die Bildverarbeitung erfunden [Hinton et al., 2011] und sollen ein grundlegendes Problem heutiger CNNs lösen: Irrelevante Eigenschaften eines Bildes werden zur Reduzierung der Speicher- und Laufzeitkomplexität ignoriert. Für manche CNNs ist die bloße Präsenz einzelner kleinerer Entitäten ausreichend, um eine große Entität zu erkennen. Wenn bspw. die einzelnen Entitäten eines Gesichtes vorhanden sind, dann wird das Gesicht als solches erkannt, unabhängig von der Größe des Mundes, die Position der Nase oder die räumliche Beziehung zwischen Augen und Ohren. Um diese Transformationen ebenfalls erkennen zu können, benötigen CNNs exponentiell mehr Filter oder Trainingsdaten [Zhao et al., 2018]. Ein Filter hat eine bestimmte Größe, wandert über die Matrix eines Bildes und analysiert das Bild danach, ob es in diesem bestimmten Areal ein bestimmtes Muster aufweist. Die Analyse geschieht durch Berechnungen mit den Gewichtungen innerhalb des Filters, dadurch entstehen mehrere kleinere Matrizen.

Eine Kapsel wird durch gruppierte Neuronen dargestellt und repräsentiert eine Entität nicht als ein Skalar, sondern als ein Vektor. Dieser Vektor enthält Instanziierungsparameter, welche Eigenschaften der repräsentierenden Entität beschreiben. Je länger der

[1] `http://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html`, Aufgerufen: 11.11.2019

Ausgabewertvektor einer Kapsel ist, desto höher ist die Konfidenz der Erkennung einer Instanz. Die Bedeutung der Instanziierungsparameter und die Entität einer Kapsel wird während des Trainings festgelegt. Instanziierungsparameter können zum Beispiel die Position, die Orientierung, die Größe, usw. darstellen. Der Ausgabewert einer Kapsel ist invariant. Das bedeutet, dass bei einer Veränderung des Blickwinkels (auf ein Bild) sich zwar die spezifischen Instanziierungsparameter verändern, aber nicht die Länge des Ausgabewertvektors einer Kapsel.

Sabour et al. [2017] entwickelten das Capsule Network (CapsNet), eine Architektur, welche die Kapseln verwendet. Diese Architektur erreicht in der Bildverarbeitung gute Werte. Auf dem MNIST-Datensatz wurde ein Test Error von 0,25% erreicht. Das CapsNet bildet damit eine State-of-the-Art Architektur, laut den Autoren. Es gibt aber mindestens ein Modell, welches bessere Werte erreicht hat[2]. Der Ansatz ähnelt dem menschlichen Konzept der Erkennung von Entitäten, sodass eine Entität aus mehreren kleineren Entitäten besteht. So existieren im CapsNet zwei Arten von Kapseln, die *Primary Caps* (PrimCaps) und die *Dense Caps* (DensCaps). Die PrimCaps stellen die niedrigeren Entitäten dar, während die DensCaps die höheren Entitäten darstellen. Beispielsweise kann ein Gesicht eine höhere Entität sein und die Nase die niedrigere Entität. Dadurch ist das CapsNet robuster gegenüber Rotation, Reflexion, Skalierung, usw. Die Informationsweitergabe von den PrimCaps zu den DensCaps geschieht durch das Dynamic Routing. Das Dynamic Routing ersetzt das Max Pooling des CNNs. Pooling verkleinert die Dimension einer Matrix, indem die Werte innerhalb der Matrix zu einem Wert konvertiert werden. Max Pooling konvertiert eine Matrix zu dem höchsten Wert innerhalb dieser Matrix. Beim Max Pooling gehen räumliche Informationen verloren, aufgrund dieser alleinigen Auswahl des besten Features aus einem Areal. Laut den Autoren soll Dynamic Routing effektiver sein als Max Pooling.

Als Dynamic Routing verwendeten die Autoren das eigens entwickelte *Routing-by-agreement*. Sei c_{prim} die Anzahl der PrimCaps-Kapseln und c_{dense} die Anzahl der DensCaps-Kapseln. Ferner sei M die Menge aller Klassen, dann gilt $c_{dense} = |M|$. u_i beschreibt den Ausgabewert der PrimCaps-Kapsel i mit der Dimension d_{prim}. Bevor das Dynamic Routing beginnt, wird die Gewichtungsmatrix $W_{prim|dense} \in R^{c_{dense} * c_{prim} * d_{dense} * d_{prim}}$ initialisiert. Für PrimCaps-Kapsel i wird der Vektor $\hat{u}_{j|i} = u_i * W_{prim|dense}$ berechnet. Der Vektor $\hat{u}_{j|i}$ beinhaltet die vorhergesagten Ausgabewerte für jede DensCaps-Kapsel j. Nun kann die Iteration des Dynamic Routings beginnen. Es wird der Ausgabewert der DensCaps-Kapseln berechnet: $v_j = squash(\sum_i c_{j|i} \hat{u}_{j|i})$. Die Kopplungsmatrix $c_{j|i}$ gibt die Relevanz von $\hat{u}_{j|i}$ für v_j an. *Squash(x)* skaliert den Ausgabewert nicht-linear zwischen null und eins, sodass die Länge als Wahrscheinlichkeit für die Existenz einer Entität steht. Sofern keine Iteration mehr ansteht, ist v_j der Ausgabewert der

[2] `http://yann.lecun.com/exdb/mnist/`, besucht am 11.11.2019

DensCaps-Kapseln und v der Ausgabewert des DensCaps. Ist die maximale Anzahl an Iterationen noch nicht erreicht, so wird die Ähnlichkeit $b_j|i$ zwischen $\hat{u}_{j|i}$ und v_j berechnet: $b_{j|i} \leftarrow b_{j|i} + \hat{u}_{j|i} * v_j$. Daraufhin wird die Kopplungsmatrix $c_{j|i}$ aktualisiert $c_{j|i} = softmax(b_{j|i})$. *Softmax(x)* skaliert die Werte zu einer Wahrscheinlichkeitsverteilung. Damit ist diese Iteration beendet. Dieser Algorithmus wird im Pseudocode-Block 1 dargestellt.

Algorithmus 1 : Der Routing-Algorithmus nach Sabour et al. [2017]

Result : v_j

Initialization: $\forall i \in PrimCaps.\forall j \in DensCaps : b_{j|i} \leftarrow 0$

for *r iterations* **do**

$\forall i \in PrimCaps : v_j \leftarrow squash(\sum_i c_{j|i}\hat{u}_{j|i})$

$\forall i \in PrimCaps.\forall j \in DensCaps : b_{j|i} \leftarrow b_{j|i} + \hat{u}_{j|i} * v_j$

$\forall i \in PrimCaps.\forall j \in DensCaps : c_{j|i} \leftarrow softmax(b_{j|i})$

end

Dieser Algorithmus berechnet Vektoren aus den PrimCaps-Kapseln, welche den Ausgabewert der DensCaps-Kapseln vorhersagen und je ähnlicher die Vorhersage ist, desto mehr Einfluss hat eine PrimCaps-Kapsel auf eine DensCaps-Kapsel in der nächsten Iteration. Da der Ausgabewert einer DensCaps-Kapsel durch die vorhergesagten Vektoren $\hat{u}_{j|i}$ beeinflusst wird, ist eine Konvergenz garantiert. In vielen Implementationen hat das Dynamic Routing mit drei Iterationen die besten Werte erreicht [Zhao et al., 2018; Sabour et al., 2017; Ren und Lu, 2018]. Aufgrund der großen Tensoren und Berechnungen nimmt das CapsNet viel Arbeitsspeicher in Anspruch und benötigt viel Rechenzeit zum Trainieren. Während des Trainings wurden teilweise mehr als elf Gigabyte GPU-RAM benötigt und eine Trainingszeit von bis zu sechs Stunden.

Sabour et al. [2017] konstruierten ebenfalls eine eigene Loss Function, die sog. Margin Loss Function, welche die Länge eines Vektors beachtet, da dieser ausschlaggebend ist, für die Existenz einer Entität. Ebenfalls wurde das Rekonstruktions-Modul konstruiert. Es dient als Decoder und kriegt als Eingabewert den Ausgabewert der DensCaps. Dieses Modul rekonstruiert aus den Instanziierungsparametern den Eingabewert des Encoders. Der Ausgabewert des Rekonstruktions-Moduls wird ebenfalls in der Margin Loss Function beachtet, aber nur zu einem sehr geringen Prozentsatz. Aus diesem Grund ist das Rekonstruktions-Modul in der Loss Function von geringer Bedeutung und wird in einigen Implementationen für die Textklassifizierung auch nicht beachtet [Zhao et al., 2018; Dobert, 2019]. Das Rekonstruktions-Modul dient zur Regularisierung des CapsNet.

Zhao et al. [2018] entwickelten und zeigten eine Möglichkeit, wie mit einem CapsNet Texte klassifiziert werden können. Es wird angenommen, dass die Instanziierungspara-

meter Eigenschaften wie Lokalität oder auch Semantik darstellen könnten. Sie erreichten State-of-the-Art Werte im Bereich der Textklassifizierung. Weiterhin entwickelten sie zwei andere Arten des Dynamic Routings und verwendeten Word2Vec als Word Embedding. Auch Ren und Lu [2018] entwickelten eine Möglichkeit, das CapsNet zur Textklassifizierung zu nutzen. Darüber hinaus haben sie ebenfalls ihren eigenen Dynamic Routing Algorithmus konstruiert, welcher auf den K-Means Algorithmus basiert. Sie verwendeten das Compositional Coding, um den Speicherverbrauch von Word Embeddings zu verringern. Ihr Modell besteht aus einem normalen CapsNet verbunden mit einem Bidirectional Gated Recurrent Unit (BiGRU) [Cho et al., 2014]. Dieses Modell erreichte in einigen Aufgaben beinahe State-of-the-Art Werte. Auch Aly et al. [2019] verwendeten ebenfalls das CapsNet für eine Multilabel-Klassifizierung. Dabei war das CapsNet besser als ein LSTM-Modell.

In dieser Arbeit wird eine Implementierung genutzt, welche weniger Arbeitsspeicher verbraucht und weniger Zeit zum Trainieren benötigt [Dobert, 2019]. In der ursprünglichen Implementierung fehlt der erste CNN-Layer, welcher jedoch wieder hinzugefügt wurde. Dieser Layer verringert die Dimensionalität in den Caps-Layern, sodass ein geringerer Ressourcenverbrauch erreicht wird. Dieser Layer wird ebenfalls bei Sabour et al. [2017] und Zhao et al. [2018] verwendet. Diese Implementierung wurde gegen das MNIST Dataset[3] getestet. Die Ergebnisse sind vergleichbar mit dem CapsNet von Sabour et al. [2017], sodass die Funktionalität gewährleistet werden kann. In Abbildung 3.4 sind die Ergebnisse zu sehen. Dieses Ergebnis entspricht der Aussage aus Dobert [2019], dass die Implementierung vergleichbare Ergebnisse liefert.

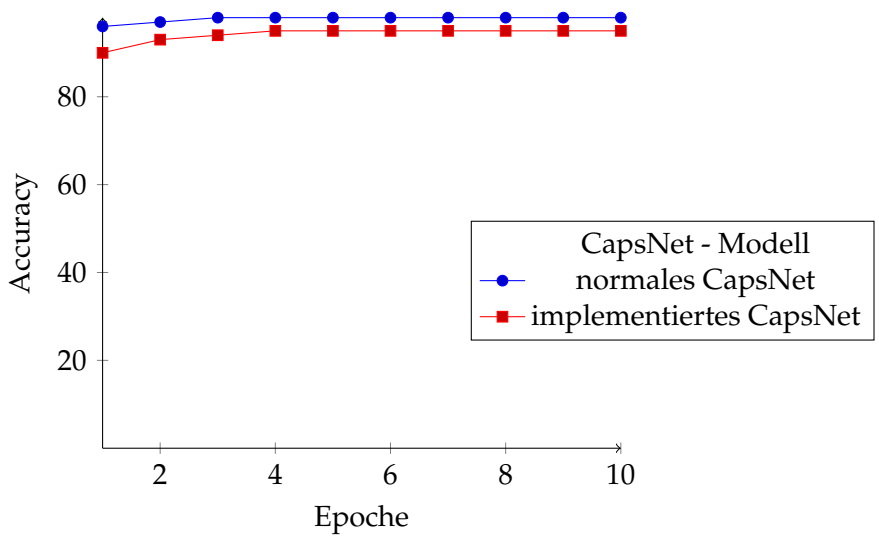

Abbildung 3.4: Ein Vergleich zwischen der CapsNet-Implementierung von Sabour et al. [2017] und der in dieser Bachelorthesis genutzten CapsNet-Implementierung.

[3] http://yann.lecun.com/exdb/mnist/

4 Datensatz

In dieser Bachelorthesis wird ein bisher unveröffentlichter Datensatz verwendet, welcher von der Universität Trier stammt. Auch wenn der OMT Variabilität in der Teststruktur bietet [Kuhl und Scheffer, 1999], so wurde dieser Datensatz mit dem normalen OMT erstellt, welcher 15 Bilder mit jeweils vier Fragen beinhaltet. Der Datensatz besteht aus 214.037 beantworteten Bildern von 14.775 Testpersonen. Die Sprachverteilung in diesem Test lässt sich in Tabelle 4.1 erkennen. Die Erkennung dieser Sprachen wird näher in Kapitel 5.1 beleuchtet. Darüber hinaus wurden weitere Sprachen erkannt, welche jedoch aufgrund invalider Daten erkannt wurden. Die Invalidität entsteht unter anderem durch Wiederholungen von Zeichenketten oder durch Antworten mit weniger als drei Buchstaben, wobei diese Wörter nicht existent sind. Als invalide würden folgende Antworten gelten: „sd“, „abcdabcdabcd“ oder „-“. In Tabelle 4.2 wird die Motiv-Ebene-Verteilung anhand der deutschsprachigen Testdaten dargestellt. Diese Bachelorthesis wird sich ausschließlich mit den deutschsprachigen Testdaten befassen. In Tabelle 4.2 ist ebenfalls zu erkennen, dass neben dem Macht-, Leistungs- und Anschlussmotiv auch das Freiheitsmotiv genutzt wurde. Dieses Freiheitsmotiv ist in der Motivforschung noch nicht wirklich etabliert. Weiterhin ist dieses implizite Motiv eine echte Untermenge des Machtmotivs (D. Scheffer, persönliche Kommunikation, 23. September 2019). Für eine bessere Vergleichbarkeit mit der bestehenden Forschung wird das Freiheitsmotiv als Machtmotiv gewertet. Weiterhin kann aus dem Datensatz entnommen werden, dass vor allem bestimmte Wörter für bestimmte Motive relativ häufig oft vorkommen. Für das Anschlussmotiv sind das häufig Wörter wie: "Gruppe", "unterhalten", "fühlt", "Gespräch", "zusammen". Für das Leistungsmotiv sind das Wörter wie: "Berg", "Aufgabe", "arbeiten", "Ziel", "konzentriert". Für das Machtmotiv kommen folgende Wörter häufig vor: "überlegen", "wichtig", "möchte", "helfen", "stark".

Deutsch	Französisch	Englisch	Russisch	Italienisch
$\approx$ 94,87 %	$\approx$ 2,53 %	$\approx$ 1,17 %	$\approx$ 0,03 %	$\approx$ 0,01 %

Tabelle 4.1: Diese Tabelle zeigt die Sprachverteilung innerhalb des Datensatzes an. Einige Sprachen wurden nicht mit aufgenommen, da der Detektor in diesen Fällen Spam-Daten detektiert hatte.

Wie bereits beschrieben, ist für die Evaluierung eine Trainings-, Test- und Validierungsmenge notwendig. In dieser Arbeit beinhaltet die Trainingsmenge 60% der Daten aus

dem Datensatz. Die Test- und Validierungsmenge beinhalten jeweils 20% der Daten. Bei der Verteilung wurde auf die Klassenverteilung geachtet, sodass die Mengen die gleiche relative Verteilung der Klassen haben. Es existieren die Klassen A0-A5, L0-L5, M0-M5 und *0*. Die Klasse *0* fasst alle Entitäten aus dem Null-Motiv zusammen, sodass nur die Null-Ebene dafür angenommen wird. Die Klassen 01, 02 und 03 würden mindestens in einer Menge nicht auftauchen, da es zu wenige Entitäten gibt. Weiterhin ist die Klasse 00 zu 99,85% dominierend im Null-Motiv. Da 50% der Ebenen im Null-Motiv nicht klassifizierbar sind, aufgrund des niedrigen Aufkommens und die Klasse 00 sehr dominierend ist, wurden die Entitäten aus den Motiv-Ebene-Kombinationen 01, 02, 03, 04 und 05 nicht verwendet. Somit bleibt lediglich die Klasse *0* bzw. 00 vorhanden. Vor der Aufteilung der Entitäten wurden diese zufällig gemischt, sodass die Antworten einzelner Testpersonen in mehreren Mengen auftauchen können. Dies hat den Vorteil, dass die Klassen durch mehrere Kontexte beschrieben werden und sich diese Kontexte in eventuell jeder Menge wiederfinden. Als Kontext kann bspw. die Wahl der Hauptperson oder auch die Wahrnehmung der Umgebung gelten.

	Anschluss	**Leistung**	**Macht**	**Freiheit**	**Null**
Ebene 0	10	7	25	7	9844
Ebene 1	3464	2885	11601	2096	0
Ebene 2	11651	15351	8305	6639	2
Ebene 3	1630	7601	12967	5217	2
Ebene 4	9181	9053	32806	10873	4
Ebene 5	8382	4554	17260	11428	7

Tabelle 4.2: Diese Tabellen zeigen die Verteilung der Motiv-Ebene-Kombinationen der deutschsprachigen Daten.

5 Methodik

In diesem Kapitel wird die Art und Weise der Evaluierung beschrieben. Als Erstes wird die Vorverarbeitung des Datensatzes erläutert. Danach wird die verwendete Evaluationsmetrik genannt und inwiefern eine Parametrisierung stattfindet. Zuletzt wird die Baseline definiert und die Art und Weise der isolierten Klassifizierung dargestellt.

5.1 Vorverarbeitung

Die Vorverarbeitung beginnt mit dem Filtern von nicht-deutschsprachigen Antworten. Dafür wird ein Framework[1] verwendet, welches eine Portierung der Spracherkennungssoftware von Google ist. Die Verteilung der Sprachen im Datensatz lässt sich in Tabelle 4.1 sehen. Durch diese Filterung wurden ebenfalls invalide Daten entfernt. Alle vier Antworten pro Bild werden konkateniert, da keine Antwort speziell für ein gemessenes Motiv oder eine gemessene Ebene verantwortlich ist. Die Umlaute werden durch lateinische Buchstabenkombinationen ersetzt, sofern dies für das Word Embedding notwendig ist. In Kapitel 5.3 werden zwei Word Embeddings miteinander verglichen, wovon eines Wörter mit Umlauten hat und das andere Wörter ohne Umlaute besitzt. Im nächsten Schritt werden Interpunktionszeichen entfernt. Es gibt mehrere Gründe, wieso dies von Nutzen ist: Das in dieser Bachelorthesis verwendete Word2Vec-Modell hat ebenfalls bei der Vorverarbeitung Interpunktionszeichen entfernt, sodass unter anderem Punkte ("."), Kommata (","), Bindestriche ("-"), usw. entfernt wurden. Ein weiterer Grund ist die häufige und falsche Nutzung von Satzzeichen im Datensatz, sodass bspw. die Zeichenkette ”””” 4.854 Mal auftaucht. Weiterhin empfiehlt die Instruktion zum OMT, dass die Antworten in Stichpunkten geschrieben werden sollen.

Die maximale Länge einer konkatenierten Antwort wird auf 22 Wörter gekürzt. Sie entspricht der durchschnittlichen Länge aller konkatenierter Antworten. Eine feste Länge ist notwendig, da das CapsNet nicht mit variablen Sequenzen benutzbar ist. Laut Kuhl und Scheffer [1999] sollten die ersten Wörter der Antwort ausschlaggebend für das gefundene Motiv und die gefundene Ebene sein. Es gilt auch die *Primacy Rule*, welche besagt, dass das erste gefundene Motiv oder die erste gefundene Ebene nicht geändert werden darf.

[1] `https://pypi.org/project/langdetect/`

Sollte eine Antwort weniger als 22 Wörter besitzen, so wird die Sequenz mit leeren Worten aufgefüllt (Padding). Der Wortvektor dieser leeren Wörter ist der Null-Vektor. Da das MLP nur mit Tensoren des Ranges *1* (Vektoren) verwendet werden kann, ist diese Architektur davon nicht betroffen. Das LSTM kann zwar mit variablen Sequenzen arbeiten, doch in Abbildung 5.1 ist zu erkennen, dass das LSTM mit Sequenzen, welche aufgefüllt wurden, bessere Ergebnisse erreicht, als ohne Auffüllung. Das LSTM-Modell mit Padding hat den besten Wert mit einem F1-Score von 0,5936 erreicht. Das LSTM-Modell ohne Padding erreichte einen Wert von 0,5895.

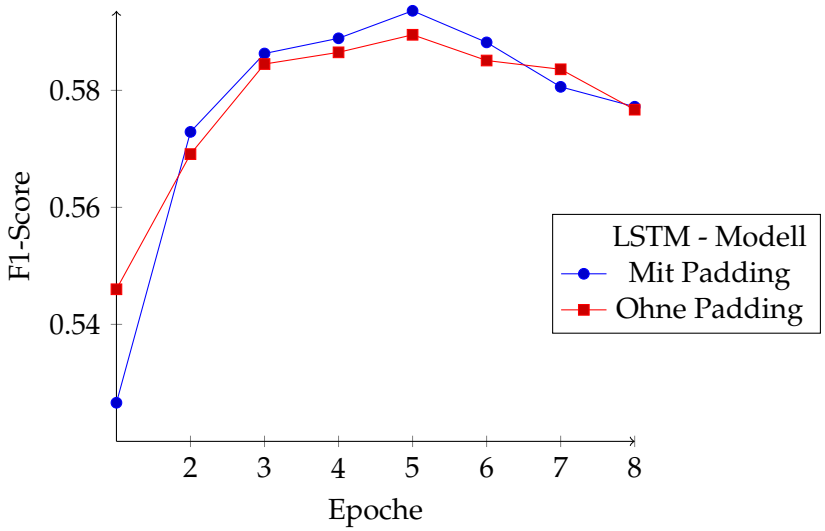

Abbildung 5.1: In dieser Abbildung wird ein LSTM-Modell mit und ohne Padding verglichen. Dabei beginnt das Koordinatensystem bei 0,52 für eine bessere Vergleichbarkeit. Diese Ergebnisse basieren auf der Validierungsmenge. Dabei wurde eine Learning Rate von 0,0001, 700 Hidden Units und eine Batch Size von Eins als Parameter festgelegt.

5.2 Evaluationsmetrik

In dieser Arbeit wird der F1-Score [Sasaki, 2007] und die Genauigkeit (engl. Accuracy) zum Vergleichen der Modelle genutzt. Der F1-Score ist das harmonische Mittel zwischen Precision und Recall und wird folgendermaßen berechnet: $F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$. Der Vorteil der Accuracy gegenüber dem F1-Score ist die einfache Verständlichkeit dieses Wertes, da diese Metrik vor allem in der alltäglichen Praxis genutzt wird. Sie besitzt aber ein Problem, welches im folgendem Beispiel deutlich wird: Bei einer binären Klassifizierung besteht der Testdatensatz aus 100 Entitäten. 80 Entitäten gehören zu Klasse A und 20 Entität zu Klasse B. Wenn das Modell jede Entität als Klasse A klassifiziert, dann beträgt die Accuracy zwar 80%, aber die konkretere Accuracy für Klasse B beträgt 0%. Dies führt zum Problem, dass aufgrund eines ungleich verteilten Datensatzes die Reliabilität für das Modell für die wenig aufkommenden Klassen niedrig ist, aber die allgemeine Accuracy

dies nicht stark genug aussagt. Der F1-Score liegt in diesem Fall bei 71,2%.

Es gibt drei Arten für den F1-Score in einer Multiklassen-Klassifikation. Der *macro*-F1-Score, der *micro*-F1-Score und der *weighted*-F1-Score. Der *macro*-F1-Score berechnet den F1-Score in dem es den Durchschnitt der einzelnen F1-Scores bildet, dadurch würde jede Klasse den gleichen Einfluss auf den gesamten F1-Score haben, welches in einem ungleich verteilten Datensatz eventuell erwünscht ist. Dies kann aber auch zu schlechten Ergebnissen führen, sobald es Klassen gibt, die kaum im Datensatz vorkommen, sodass diese speziell einen schlechten F1-Score erreichen und aufgrund des großen Einflusses den gesamten F1-Score verschlechtern. Der *micro*-F1-Score berechnet den globalen F1-Score (Ohne vorher die einzelnen Klassen zu betrachten). Dies hat den Vorteil, dass keine Klasse im einzelnen betrachtet wird, sondern das gesamte Modell an sich. Bei diesem Wert haben vor allem Klassen, zu denen viele Entitäten in einem ungleich verteilten Datensatz gehören, größeren Einfluss auf den F1-Score. Der *weighted*-F1-Score ähnelt dem *macro*-F1-Score, nur das jeder einzelne Wert mit einer Gewichtung multipliziert wird. Die Gewichtungen entsprechen meist der prozentualen Häufigkeit einer Klasse. In dieser Bachelorthesis wird der *weighted*-F1-Score genutzt. Sowohl die Accuracy, als auch der F1-Score, werden häufig in der Textklassifizierung verwendet.

5.3 Parametrisierung

Für einen fairen Vergleich der Modelle ist es notwendig, deren Hyperparameter zu tunen. Dies bedeutet, dass verschiedene Parameterkombinationen genutzt werden, um bestmögliche Werte für ein Modell zu erreichen. Dabei wird das Modell auf der Validierungsmenge getestet. Es wird eine gierige Grid-Search-Methode angewendet. Grid-Search beschreibt die Verwendung jeder Parameterkombination aus vorher festgelegten Parameterkategorien und Werten, sodass endlich viele Parameterkombinationen vorhanden sind. Die hier verwendete Methode ist deshalb gierig, da jede Parameterkategorie isoliert und nach einander betrachtet wird. Dadurch wird zuerst der beste Wert für die Learning Rate gesucht, wobei für die anderen Parameterkategorien jeweils ein fester Wert genommen wird. Nachdem der beste Wert für die Learning Rate gefunden wurde, wird eine andere Parameterkategorie getunt, wobei der Wert für die Learning Rate durch das vorherige Tuning feststeht. Zwar führt diese Herangehensweise nicht zu den besten Ergebnissen, dafür ist die Rechenzeit deutlich kleiner. Wie bereits in Kapitel 3.2.3 vorgestellt, kann ein Training bis zu sechs Stunden dauern. In Tabelle 5.1 ist zu sehen, welche Parameterkategorien trainiert werden und welche Werte gewählt wurden. Es sei zu erwähnen, dass die Anzahl der PrimCaps-Kapseln c_{prim} sich aus der Multiplikation der *Prime caps num.* und der *Prime caps dim.* ergeben. Dies geht aus der ursprünglichen Implementierung hervor.

Parameter	**MLP**	**LSTM**	**Caps**
Batch Size	{8, 16, 32}		
Learning Rate	{0,001, 0,0001, 0,00005}		
Hidden Units	{500, 700, 1000}		
Prime caps num.			{8, 16}
Prime caps dim.			{8, 16}
Dense caps dim.			{8, 16}

Tabelle 5.1: Die hier dargestellten Parameterkategorien werden für die speziellen Modelle mit bestimmten Werten getunt. Die Parameterkategorie *Hidden Units* existiert nur für die MLP- und LSTM-Modelle. *Prime caps num., Prime caps dim.* und *Dense caps dim.* existieren nur für die CapsNet-Modelle.

Das Training geschieht über 30 Epochen, wobei es stoppt, nachdem dreimal hintereinander der F1-Score schlechter wurde, als in der vorherigen Epoche. Als Optimizer wurde ADAM [Kingma und Ba, 2015] gewählt, da dieser in vielen Textklassifizierungen vorkommt und als einer der Standard-Optimizer gilt. Als Loss Function wird grundsätzlich Cross Entropy verwendet, wobei beim CapsNet die Margin Loss Function verwendet wird. Die Margin Loss Function beachtet die spezifische Eigenschaft des Ausgabewertes, sodass die Länge der Vektoren ausschlaggebend ist. Darüber hinaus führt die Margin Loss Function auch zu besseren Ergebnissen, sodass in Abbildung 5.2 erkennbar ist, dass bereits ab der ersten Epoche die Margin Loss Function bessere Werte erzielt als die Cross Entropy Function. In der Darstellung ist ein Sprung des F1-Scores bei der Cross Entropy Function zu erkennen. Dieser tritt auch bei einem zweiten Versuch auf. Es sind keine besonderen Änderungen des Loss-Wertes festzustellen, sodass dies eher für eine Inkompatibiltät zwischen der Cross Entropy Function und des CapsNets spricht. Um möglichst viel Potential aus dem CapsNet rauszuholen, wird die Margin Loss Function für das Training verwendet.

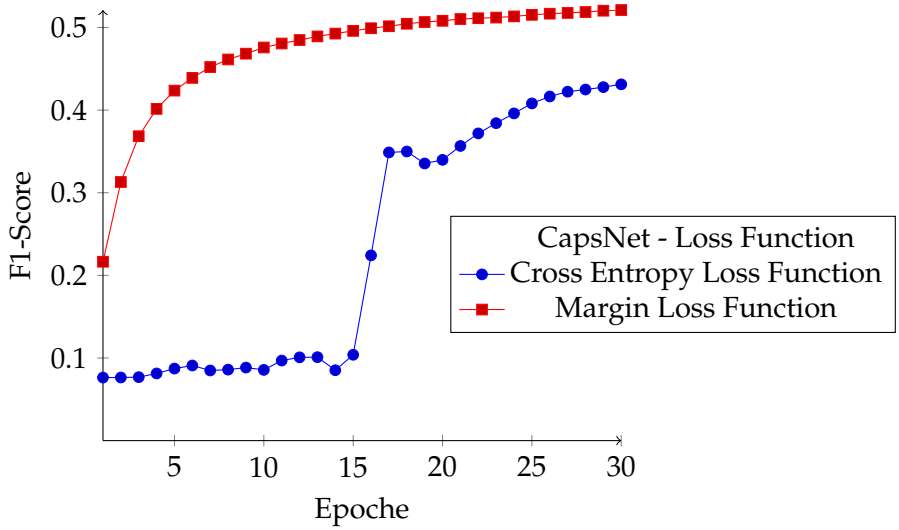

Abbildung 5.2: Diese Tabelle zeigt einen Vergleich zwischen der Cross Entropy Loss Function und der Margin Loss Function in einem CapsNet, mit fastText als Word Embedding.

Es wurden zwei Word Embeddings basierend auf der Word2Vec-Methode miteinander verglichen. Das WikiNews-Modell[2] und das Twitter-Modell[3]. Das Twitter-Modell basiert auf einen 9,6 GB großen Korpus, während der Korpus des WikiNews-Modells 7,49 GB groß ist. Vor dem Training wurden beim WikiNews-Modell Stoppwörter und Sonderzeichen aus dem Datensatz entfernt. Als Stoppwort werden Wörter bezeichnet, welche sehr häufig auftreten und normalerweise keine Relevanz haben. Dazu zählt unter anderem das Wort "und" oder "ist". Eine Vorverarbeitung des Datensatzes beim Twitter-Modell wurde nicht vorgenommen. Für die Evaluierung beider Modelle wurde der eigene vorverarbeitete Datensatz genutzt (s. Kapitel 5.1). Falls notwendig, wurden die Umlaute durch die lateinischen Buchstabenkombinationen ersetzt. Weiterhin wurden die Modelle sowohl mit als auch ohne Beachtung von Groß- und Kleinschreibung verglichen. Durch die nicht Beachtung von Groß- und Kleinschreibung wird der Wortvektor der anderen Schreibweise genutzt, sofern dieser existiert und kein Wortvektor für die originale Schreibeweise vorhanden ist. Zur Evaluierung wurde das LSTM-Modell verwendet. In Abbildung 5.3 lässt sich erkennen, dass das WikiNews-Modell in Kombination mit der Nichtbeachtung von Groß- und Kleinschreibung die besten Werte erreicht hat.

[2] https://devmount.github.io/GermanWordEmbeddings/
[3] https://www.cl.uni-heidelberg.de/english/research/downloads/data/GermanTwitterEmbeddings/twitter-de_d100_w5_min10.bin.gz

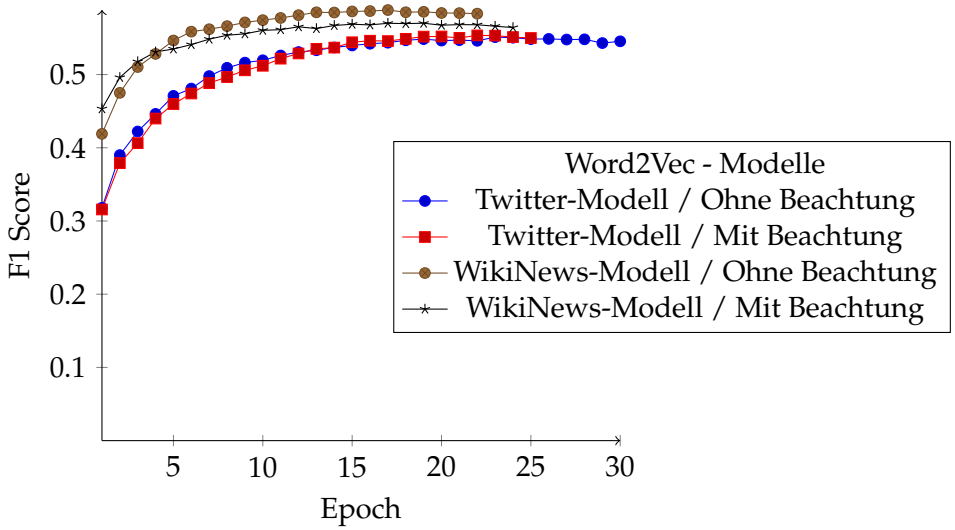

Abbildung 5.3: Dieses Koordinatensystem zeigt einen Vergleich zwischen dem WikiNews-Modell und dem Twitter-Modell. Beides sind Word Embeddings aus der Word2Vec-Methode. Die Evaluierung geschah sowohl mit als auch ohne Beachtung von Groß- und Kleinschreibung. Als Modell wurde ein LSTM-Modell genutzt mit einer Learning Rate von 0,0001, 700 Hidden Units und einer Batch Size von 32.

Hervorzuheben ist die Nichtrelevanz der Schreibweise beim Twitter-Modell. Der Grund dafür kann die geringe Anzahl an OOV-Wörter sein. Im WikiNews-Modell sind viele Wörter nicht existent, aufgrund der Entfernung von Stoppwörtern. Dieser Effekt tritt aber nur bei der Beachtung der Schreibweise auf, denn das Wort “und“ ist zwar nicht existent, aber das Wort “Und“ existiert weiterhin und wurde nicht entfernt.

5.4 Baseline

Als Baseline verwendet diese Arbeit einen Naïve Bayes Klassifikator, genauer einen Multinomial Naïve Bayes Klassifikator. Dieser basiert auf den Satz von Bayes [Bayes, 1763]: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$. Dieses Theorem beschreibt die Berechnung des Auftretens von Ereignis A, wenn Ereignis B eingetreten ist mittels der bedingten Wahrscheinlichkeit von B, wenn Ereignis A gilt. Zusätzlich wird angenommen, dass die Features (B) unabhängig voneinander sind. Aus diesem Grund werden die Klassifikatoren naïve (dt: naiv) genannt. Zusammen mit dem BoW-Modell erreicht dieses Modell eine Accuracy von 0,5582 und einen F1-Score von 0,5556. Dabei wurde die Trainingsmenge zum Trainieren der Werte genutzt und die Testmenge zur Evaluation dieses Modells. Dieser Klassifikator hat den Vorteil, dass die Gewichtungen für jedes Feature separat gelernt werden und keine Abhängigkeiten zwischen Features existieren. Zumindest ist dies ein Vorteil, sofern auch eine Unabhängigkeit dieser Features vorhanden ist. Verglichen mit anderen Naïve Bayes Klassifikatoren ist der Multinomial Naïve Bayes Klassifikator vor allem

besser, wenn große Eingabewerte verwendet werden [McCallum et al., 1998]. Aufgrund dessen, dass die Features unabhängig sind, bildet sich vor allem für Klassen mit langen Texten der Vorteil, dass mehr Wörter für diese Klasse sprechen als es bei anderen Klassen der Fall wäre [Kim et al., 2006]. Dieser Vorteil ist, durch Kürzung der Texte des eigenen Datensatzes auf 22 Wörter, verringert worden. Auch die Seltenheit eines Wortes spielt eine große Rolle, da jedes Wort den gleichen Wert hat, werden die wichtigen Wörter eines Textes, welche für die Klassifizierung notwendig wären, durch gewöhnliche Wörter (Auch Stoppwörter) *überschattet*. Mit *überschattet* ist gemeint, dass häufig auftretende Wörter für eine häufig auftretende Klasse sprechen, obwohl diese nicht relevant seien [Kim et al., 2006]. Nichtsdestotrotz wird dieser Klassifikator ebenfalls für die Textklassifizierung eingesetzt und dient aufgrund seiner Einfachheit als gute Baseline, da bisher noch keine vergleichbaren Ergebnisse existieren.

5.5 Isolierte Klassifizierung

Nachdem die Evaluierung der Modelle beendet ist, werden zwei gute Modelle genutzt, um die Motive und die Ebenen unabhängig voneinander zu klassifizieren. Dadurch soll eine mögliche Abhängigkeit aufgezeigt werden. Es werden die Ergebnisse der isolierten Klassifizierung mit den Ergebnissen der abhängigen Klassifizierung (Klassifizierung der 19 Klassen) verglichen. Sollten die unabhängigen Modelle dabei schlechtere Ergebnisse hervorbringen, dann könnte dies ein Indiz für eine Abhängigkeit sein. Für dieses Vorgehen wird dieselbe Trainings- und Testmenge verwendet.

6 Ergebnisse

In diesem Kapitel werden die Ergebnisse der einzelnen Modelle vorgestellt und Besonderheiten hervorgehoben. Es werden sowohl die Ergebnisse, welche auf der Validierungsmenge, präsentiert, als auch die finalen Resultate, welche auf der Testmenge basieren. Auch die Ergebnisse aus der isolierten Klassifizierung werden hier vorgestellt. Alle Berechnungen wurden auf einer GPU vom Typ *NVIDIA GeForce GTX 1080 TI* mit elf GB GPU-RAM durchgeführt. In der Tabelle 6.4 wurden die Ergebnisse auf der Testmenge zusammengefasst. Die Ergebnisse der isolierten Klassifizierung lassen sich in Tabelle 6.5 erkennen. Weiterhin wurden diese Ergebnisse mit Ergebnissen aus anderen Forschungsarbeiten verglichen.
Folgende Abkürzungen werden eingeführt: W2V für Word2Vec, FT für fastText, Caps für das CapsNet, B für die Batch Size, LR für die Learning Rate, H für die Hidden Units, PC für die *Prime caps num.*, PD für die *Prime caps dim.*, und DD für die *Dense caps dimension*.

6.1 Ergebnisse MLP-Modelle

In den Abbildungen 6.1, 6.2 und 6.3 sind die Ergebnisse des Hyperparameter-Tunings zu sehen. Weiterhin zeigt die Tabelle 6.1 die daraus resultierenden Parameter für die finalen Modelle. Beim Hyperparameter-Tuning des MLP-BoW-Modells besitzen alle Parameterkombinationen ähnlich maximale F1-Scores. Von den maximal 30 Epochen hat dieses Modell maximal vier Epochen gebraucht, um den Hochpunkt zu erreichen. Folgende Parameter stellten die beste Parameterkombination dar: B = 8, LR = 0,0001 und H = 1000. Auf der Testmenge erreicht dieses Modell einen F1-Score von 0,6005 $\pm$ 0,0013 und eine Accuracy von 0,6084 $\pm$ 0,0006. Das Modell besitzt 75.180.019 trainierbare Parameter und braucht ca. 40 Minuten zum Trainieren und Evaluieren.

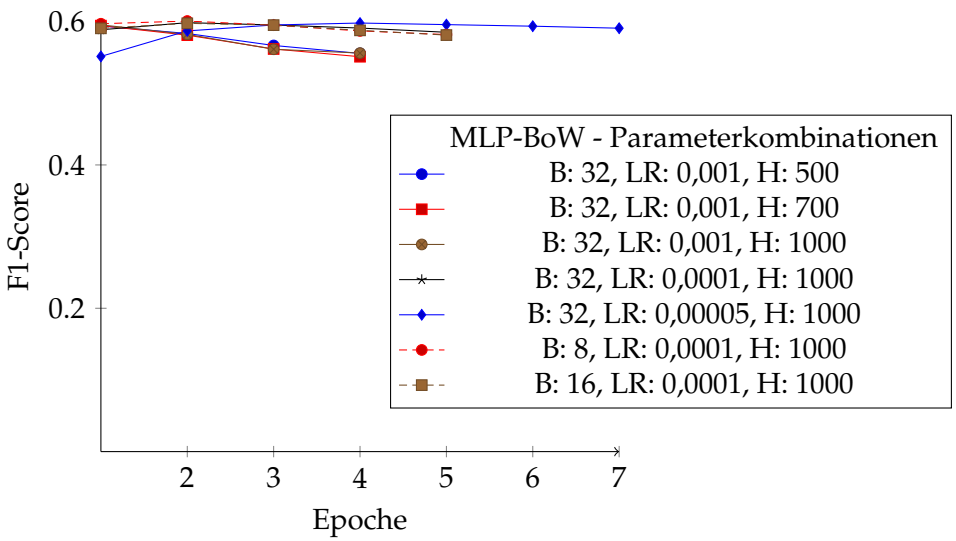

Abbildung 6.1: Tuning des MLP-BoW-Modells. Die Ergebnisse wurden alle auf der Validierungsmenge erreicht. *B* steht für die Batch Size, *LR* für die Learning Rate und *H* für die Anzahl der Hidden Units.

Das MLP-W2V-Modell und das MLP-FT-Modell klassifizieren Texte, indem der Durchschnittsvektor aus den Ausgabewerten der einzelnen Wörter generiert wird. Daher haben diese Modelle einzelne Wörter klassifiziert. Es zeigt sich, dass dieser Ansatz relativ schlechte Ergebnisse hervorbringt. In beiden Modellen beträgt der Loss während des Trainings zwischen 2,27 und 2,33. Beim MLP-BoW-Modell liegt der Loss zwischen 1,69 und 0,13. Es scheint, als würde die Initialisierung der Parameter, zu Beginn des Trainings und die ersten Anpassungen dieser, ein wesentlicher Faktor für das Ergebnis dieser Modelle sein. Dies lässt sich daran sehen, dass bei einem MLP-W2V-Modell mit denselben Parametern ein F1-Score von 0,2167 und 0,2726 gemessen wurde, während der F1-Score in der ersten Epoche bei 0,1614 und 0,2086 lag. Auch beim MLP-FT-Modell wurde während des Hyperparameter-Tunings mit derselben Parameterkombination ein F1-Score von 0,2167 (In der ersten Epoche: 0,2087) und 0,2734 (In der ersten Epoche: 0,2689) gemessen. Generell lässt sich ebenfalls nicht vorhersagen, wie viele Epochen ungefähr benötigt werden zum Trainieren. Beide Modelle besitzen 224.019 trainierbare Gewichtungen.

Beim MLP-W2V-Modell schien die Batch Size einen relevanten Unterschied zu machen, da während des Hyperparameter-Tunings vor allem die Veränderung dieser Parameterkategorie mit Abstand den besten Wert erreichte. Auf der Testmenge wurde ein F1-Score von 0,2520 $\pm$ 0,0251 gemessen. Weiterhin wurde eine Accuracy von 0,3308 $\pm$ 0,0173 erreicht.

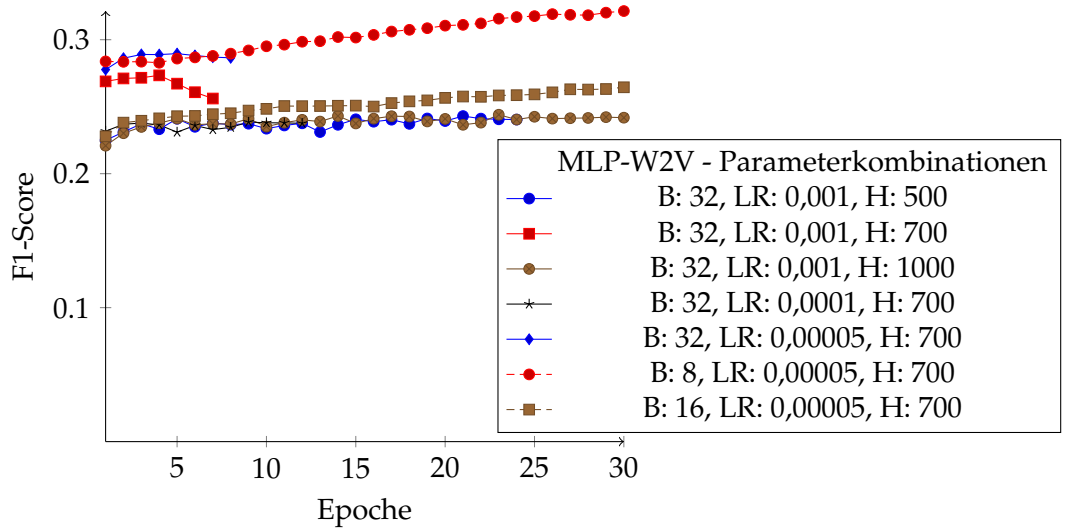

Abbildung 6.2: Hier wurde das MLP-W2V-Modell auf der Validierungsmenge getunt. *B* steht für die Batch Size, *LR* für die Learning Rate und *H* für die Anzahl der Hidden Units.

Das MLP-FT-Modell hat mit fast jeder Parameterkombination die vollen 30 Epochen zum Trainieren ausgenutzt. Es zeigt sich, dass bei einer LR von 0,00005 die Modelle nie einen Hochpunkt erreicht haben. Auf der Testmenge wurde ein maximaler F1-Score von 0,2511 $\pm$ 0,0008 und eine Accuracy von 0,3321 $\pm$ 0,0003 erreicht.

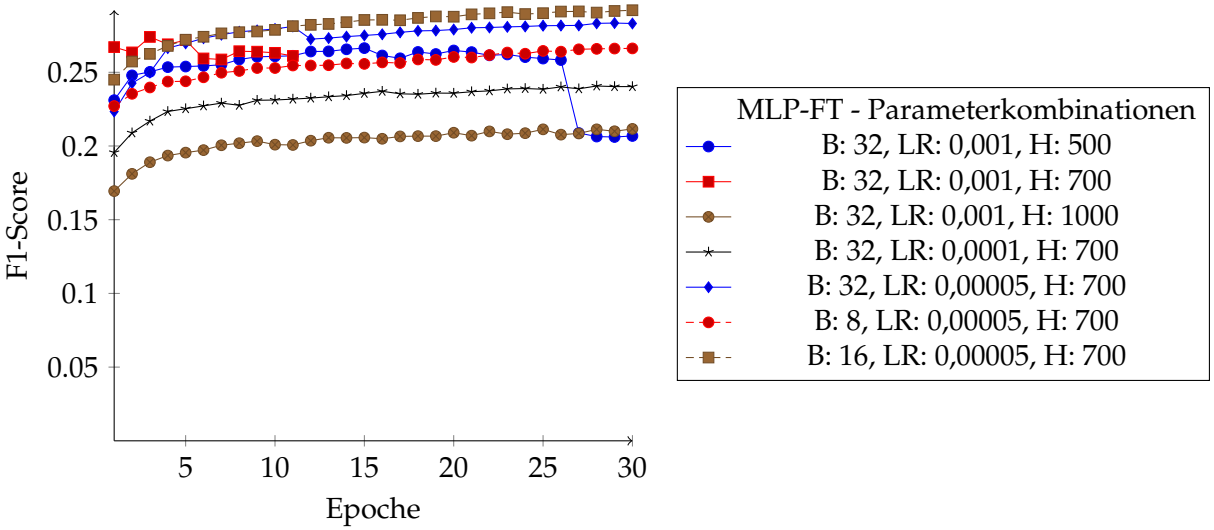

Abbildung 6.3: Evaluation der MLP-FT-Modelle. Die Ergebnisse wurden alle auf der Validierungsmenge erreicht. *B* steht für die Batch Size, *LR* für die Learning Rate und *H* für die Anzahl der Hidden Units.

Parameter	**MLP-BoW**	**MLP-W2V**	**MLP-FT**
Batch Size	8	8	16
Learning Rate	0,0001	0,00005	0,00005
Hidden Units	1000	700	700

Tabelle 6.1: Diese Tabelle zeigt die Werte der Parameterkategorien für die MLP-Modelle an, welche aus dem Hyperparameter-Tuning stammen. Diese Ergebnisse basieren auf der Validierungsmenge.

6.2 Ergebnisse LSTM-Modelle

In den Abbildungen 6.4, 6.5 und 6.6 sind die Ergebnisse des Hyperparameter-Tunings dargestellt. In Tabelle 6.2 werden die gewählten Parameter gezeigt. Beim LSTM-BoW-Modell ist zu erkennen, dass die F1-Scores aus dem Hyperparameter-Tuning ebenfalls alle ähnlich sind. Die Anzahl der Epochen steht in Abhängigkeit zur LR, dennoch wurden die 30 Epochen nicht vollständig genutzt. Es wurde ein F1-Score von 0,6016 $\pm$ 0,0008 und eine Accuracy von 0,6085 $\pm$ 0,0015 auf der Testmenge gemessen. Dieses Modell hat damit den besten F1-Score erreicht. Es enthält 304.667.019 trainierbare Parameter und benötigt ca. vier bis fünf Stunden zum Trainieren und Evaluieren. Dieses Modell wurde genutzt, um die Motive und die Ebenen isoliert voneinander zu klassifizieren. Dabei wurde für die Klassifizierung der Motive ein F1-Score von 0,8202 und für die Klassifizierung der Ebenen ein F1-Score von 0,6561 gemessen.

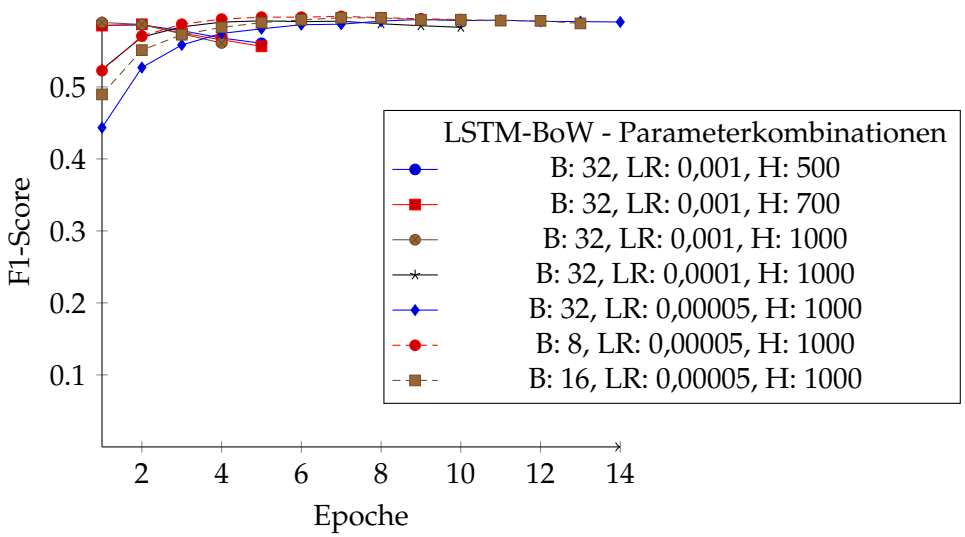

Abbildung 6.4: Diese Darstellung zeigt die Ergebnisse vom Tuning des LSTM-BoW-Modells. *B* steht für die Batch Size, *LR* für die Learning Rate und *H* für die Anzahl der Hidden Units.

Das LSTM-W2V-Modell erreichte auf der Testmenge einen F1-Score von 0,5847 $\pm$ 0,0019 und eine Accuracy von 0,5915 $\pm$ 0,0030. Hier ist nicht nur die Anzahl der Epochen abhängig von der LR, sondern auch der F1-Score (Auf der Validierungsmenge). Je kleiner die LR, desto höher der F1-Score. Insgesamt besitzt das Modell 2.818.919 Gewichtungen und braucht ca. 20 Minuten für das Training und die Evaluation. Dieses Modell wurde ebenfalls für die isolierte Klassifizierung der Motive und Ebenen verwendet. Dabei wurde ein F1-Score von 0,8026 für die Klassifizierung der Motive und ein F1-Score von 0,6183 für die Klassifizierung der Ebenen erreicht.

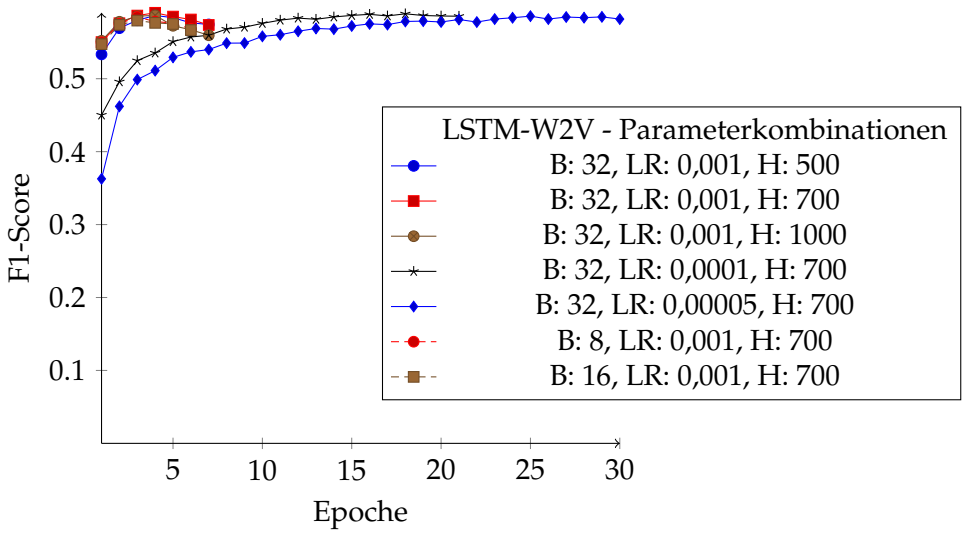

Abbildung 6.5: Die Darstellung zeigt die Ergebnisse des LSTM-W2V-Modells, welche während des Tunings erreicht wurden. *B* steht für die Batch Size, *LR* für die Learning Rate und *H* für die Anzahl der Hidden Units.

Beim LSTM-FT-Modell ist noch stärker zu erkennen, dass der F1-Score abhängig von der LR ist, sodass eine höhere LR zu einem besseren Ergebnis führt. Es darf aber nicht vergessen werden, dass das Training auf 30 Epochen beschränkt ist und daher keine Aussage getroffen werden sollte, welche darüber hinaus geht. Daher beschränkt sich die Aussage bezüglich der LR auf die 30 Epochen. Das LSTM-FT-Modell erreicht auf der Testmenge ein F1-Score von 0,5753 $\pm$ 0,0033 und eine Accuracy von 0,5860 $\pm$ 0,0016. Dieses Modell besitzt 5.227.019 trainierbare Gewichtungen und benötigt ca. 30 Minuten für das Training.

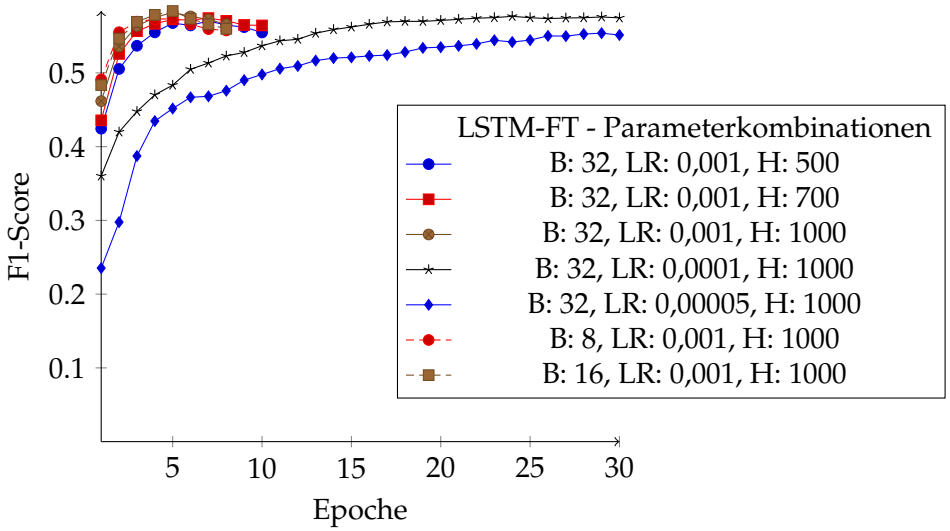

Abbildung 6.6: Es werden die Ergebnisse aus dem Hyperparameter-Tuning des LSTM-FT-Modell präsentiert. *B* steht für die Batch Size, *LR* für die Learning Rate und *H* für die Anzahl der Hidden Units.

Parameter	**LSTM-BoW**	**LSTM-W2V**	**LSTM-FT**
Batch Size	8	32	16
Learning Rate	0,00005	0,001	0,001
Hidden Units	1000	700	1000

Tabelle 6.2: Diese Tabelle zeigt die Werte der Parameterkategorien für die LSTM-Modelle, welche aus dem Hyperparameter-Tuning stammen.

6.3 Ergebnisse Caps-Modelle

In den Abbildungen 6.7 und 6.8 werden die Ergebnisse des Hyperparameter-Tunings für die Caps-Modelle vorgestellt. Die daraus resultierenden Parameter finden sich in Tabelle 6.3. Der Verlauf der Caps-Modelle ähnelt dem Verlauf der LSTM-Modelle. Das Caps-W2V-Modell hat einen F1-Score von 0,5392 $\pm$ 0,0027 und eine Accuracy von 0,5536 $\pm$ 0,003 erreicht. Das Caps-FT-Modell liegt mit einem F1-Score von 0,519 $\pm$ 0,0071 und einer Accuracy von 0,5184 $\pm$ 0,0045 unter dem Caps-W2V-Modell. Die Parameterkombination beim Caps-W2V-Modell beinhaltet die niedrigsten Werte für die Parameterkategorien aus der Grid-Search-Tabelle. In beiden Modellen zeigt lediglich die Veränderungen der LR sichtbare Änderungen an der F1-Score/Epoche-Kurve. Das Caps-W2V-Modell hat mit 5.492.288 trainierbaren Gewichtungen ca. eine Stunde für das Training gebraucht, dies ist deutlich weniger als das Caps-FT-Modell mit 19.251.840 trainierbaren Gewichtungen und drei Stunden Trainingszeit.

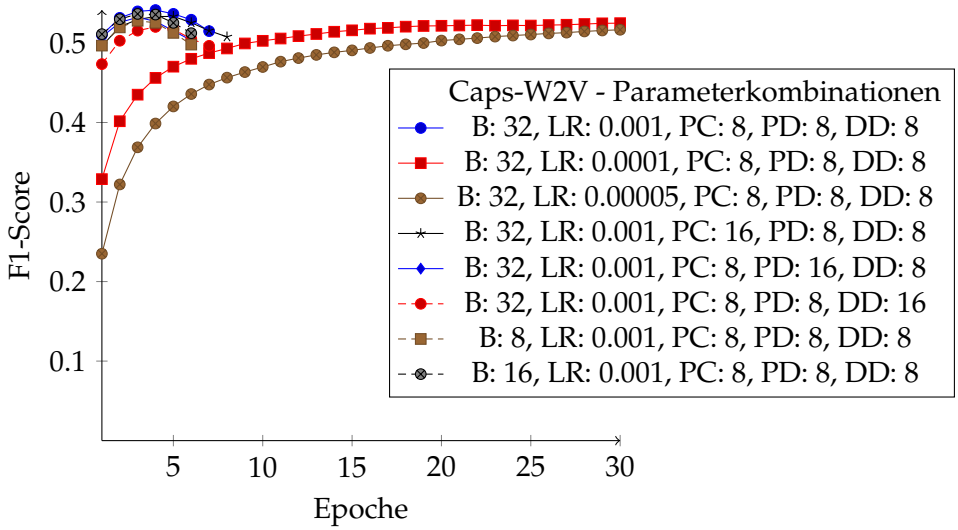

Abbildung 6.7: Es werden die Ergebnisse des Hyperparameter-Tunings für das Caps-W2V-Modell vorgestellt. *B* steht für die Batch Size, *LR* für die Learning Rate und *PC* für die *prime caps num.*, *PD* für die *Prime caps dim.* und *DD* für die *Dense caps dimension*.

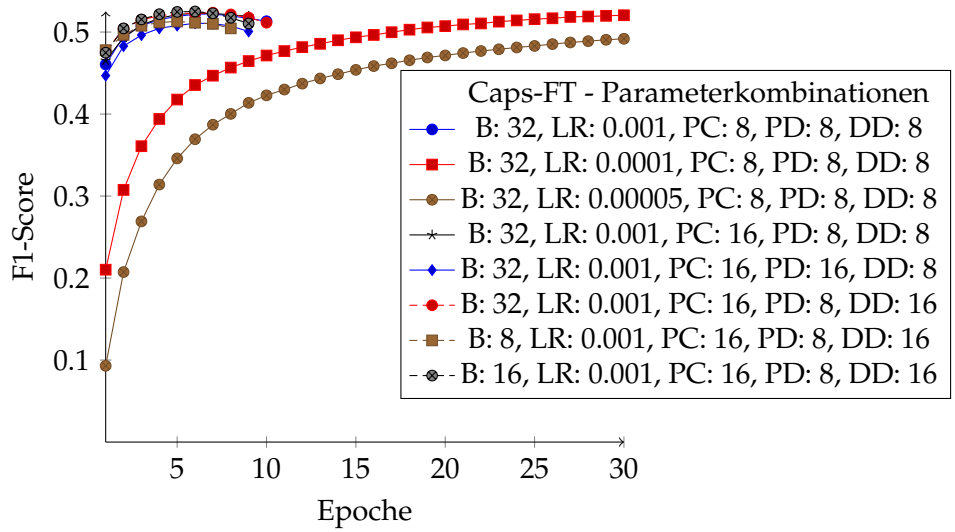

Abbildung 6.8: Es werden die Ergebnisse des Hyperparameter-Tunings für das Caps-FT-Modell vorgestellt. *B* steht für die Batch Size, *LR* für die Learning Rate und *PC* für die *prime caps num.*, *PD* für die *Prime caps dim.* und *DD* für die *Dense caps dimension*.

Parameter	**Caps-W2V**	**Caps-FT**
Batch Size	32	16
Learning Rate	0,001	0,001
Prime caps num.	8	16
Prime caps dim.	8	8
Dense caps dim.	8	16

Tabelle 6.3: Diese Tabelle zeigt die Werte der Parameterkategorien für die Caps-Modelle, welche aus dem Hyperparameter-Tuning stammen.

Modell	**F1-Score**	**Accuracy**
LSTM-BoW	0,6016 $\pm$ 0,0008	0,6085 $\pm$ 0,0015
MLP-BoW	0,6005 $\pm$ 0,0013	0,6084 $\pm$ 0,0006
LSTM-W2V	0,5847 $\pm$ 0,0019	0,5915 $\pm$ 0,0030
LSTM-FT	0,5753 $\pm$ 0,0033	0,5860 $\pm$ 0,0016
Baseline	0,5556	0,5582
Caps-W2V	0,5392 $\pm$ 0,0027	0,5536 $\pm$ 0,003
Caps-FT	0,519 $\pm$ 0,0071	0,5348 $\pm$ 0,0045
MLP-W2V	0,2520 $\pm$ 0,0251	0,3308 $\pm$ 0,0173
MLP-FT	0,2511 $\pm$ 0,0008	0,3321 $\pm$ 0,0003

Tabelle 6.4: Diese Tabelle fasst die F1-Scores, welche auf der Testmenge erreicht wurden, zusammen. Die Auflistung ist in absteigender Reihenfolge, gemessen am F1-Score.

Modell	**Motivklassifizierung F1-Score**	**Ebenenklassifizierung F1-Score**
BERT [Meyer, 2019]	0,8443	
LSTM-BoW	0,8202	0,6561
LSTM-W2V	0,8026	0,6183
LMT [Johannßen et al., 2019]	0,801	0,654
zeroR	0,5877	0,3052

Tabelle 6.5: Diese Tabelle fasst die erreichten Ergebnisse der isolierten Klassifizierung zusammen, wobei ebenfalls Ergebnisse aus anderen Forschungsarbeiten dargestellt werden, für einen besseren Vergleich. Die Sortierung geschieht in absteigender Reihenfolge anhand des F1-Scores der Motivklassifizierung. Die Modelle, welche in dieser Arbeit evaluiert wurden, sind fett gedruckt.

7 Evaluation

Im folgenden Kapitel werden die Ergebnisse evaluiert. Es werden teilweise Worthäufigkeiten als Evaluierungsmethodik genutzt. Zuerst werden die drei besten Modelle analysiert und evaluiert, sodass Schwächen und Stärken dieser Modelle hervorgehoben werden. Danach wird das MLP-W2V- und das MLP-FT-Modell, welche einzelne Wörter klassifizieren, evaluiert. Weiterhin werden die Caps-Modelle evaluiert und die Unterschiede zu den anderen Modellen hervorgehoben. Als Letztes werden die Ergebnisse der isolierten Klassifizierungen evaluiert und verglichen.

7.1 Evaluation MLP-BoW & LSTM-BoW/W2V

In dieser Arbeit hat das LSTM-BoW-Modell die besten Ergebnisse erzielt. Dabei wurde ein F1-Score von 0,6016 $\pm$ 0,0008 und eine Accuracy von 0,6085 $\pm$ 0,0015 gemessen. Aber auch das MLP-BoW- und das LSTM-W2V-Modell haben ähnliche Ergebnisse erzielt. In den Abbildungen 7.1 und 7.2 ist die Confusion Matrix des LSTM-BoW-Modells und des LSTM-W2V-Modells dargestellt. Die Confusion Matrix des MLP-BoW-Modells ist ähnlich zu den anderen beiden. Die folgende Analyse wurde sowohl mit als auch ohne Stoppwörter erstellt. Auf der einen Seite haben Stoppwörter normalerweise keine Relevanz für die Bedeutung eines Satzes, aufgrund ihres häufigen Auftretens. Auf der anderen Seite hat Pennebaker et al. [2014] aufgezeigt, dass für die Vorhersage des akademischen Erfolgs von Studenten, die Art und Weise wie ein Satz geschrieben wurde, wichtig ist. Aus diesem Grund werden in diesem Unterkapitel beide Methoden angewendet. Als Stoppwortliste wurde der Stoppwortkorpus von *NLTK*[1] [Bird et al., 2009] genutzt.

In den Confusion Matrizen ist zu erkennen, dass vor allem die Klassen A2 und A5 im Bereich des Anschlussmotivs gute Werte erzielt haben. Die A0-Entitäten wurden nie als A0 klassifiziert. Dies liegt wahrscheinlich an der niedrigen Anzahl der Entitäten im Datensatz. Die A1-Entitäten wurden häufig als A2 klassifiziert. Die Klasse A1 ist deutlich weniger als die Klasse A2 im Datensatz vorhanden. Betrachtet man die 20 häufigsten Wörter, so haben beide Klassen 15 (Mit Stoppwörter) bzw. 14 (Ohne Stoppwörter) gemeinsame Wörter. Zum Vergleich haben die Klassen A2 und A5 zehn von 20 gemeinsame Wörter (Mit Stoppwörter). Diese Ähnlichkeit kann ein Grund für die falsche Klassifizierung sein. Die Klasse A3 ist ebenfalls selten im Datensatz vorhanden, dennoch fällt auf,

[1] https://www.nltk.org/

dass die Klasse häufig als A4 oder M4 klassifiziert wird. Bei den als A3 klassifizierten Entitäten kommt das Wort "Gruppe" in 42,5% der Entitäten vor. Auch das Wort "möchte" kommt in 30% dieser Entitäten vor. In der gesamten Testmenge ist "Gruppe" in 4,27% und "möchte" in 7,16% Entitäten vorhanden. Die Wörter "Person" (40,18%) und "Gruppe" (16,56%) kommen in den meisten A3-Entitäten vor (nicht zu verwechseln mit den als A3 klassifizierten Entitäten). Generell ist das Wort "Person" in 29,52% aller Entitäten aus der Testmenge vorhanden. Dieses Wort wird deshalb so häufig verwendet, da die Fragen sich meist auf eine Hauptperson beziehen. Aus diesem Grund werden A3-Entitäten häufig als M4 klassifiziert, da diese Klasse oft im Datensatz vorhanden ist und somit häufig auftretende Wörter eher die Klasse M4 repräsentieren. Zwischen den A3- und A4-Entitäten existieren einige gemeinsame, anschlussmotivierte, Wörter. Beispielsweise "Gespräch", "traurig" und "unsicher". Da die Klasse A4 deutlich öfter im Datensatz existiert, als A3, werden Entitäten eher der Klasse A4 zugeordnet. Sofern eine anschlussmotivierte Entität falsch klassifiziert wurde, so ist die Wahrscheinlichkeit hoch, dass diese als ein Machtmotiv klassifiziert wurde, aufgrund der Dominanz des Machtmotivs im Datensatz.

Beim Leistungsmotiv wurden ebenfalls die L0-Entitäten falsch klassifiziert. Diese kommen ebenso selten im Datensatz vor. Auch in diesem Motiv lässt sich erkennen, dass die L1-Entitäten häufig als L2-Entität klassifiziert wurden. Werden die ersten 20 häufigsten Wörter betrachtet, so ist zu erkennen, dass beide Klassen 14 (Mit Stoppwörter) bzw. elf (ohne Stoppwörter) gemeinsame Wörter haben. Aufgrund der Häufigkeit der L2-Klasse und die hohe Ähnlichkeit beider, gemessen anhand der gemeinsamen Wörter, könnte dies ein Grund für die falsche Klassifizierung sein. Die L3-Entitäten wurden häufig als L2-Entitäten klassifiziert. Auch hier ist die hohe Ähnlichkeit der Klassen dafür verantwortlich, als auch der deutliche Unterschied in den Häufigkeiten beider Klassen innerhalb des Datensatzes. Zwischen den Klassen L5 und M5 existieren 15 von 20 (Mit Stoppwörter) bzw. zwölf von 20 (Ohne Stoppwörter) gemeinsame Wörter. Aufgrund des großen Unterschieds in den Häufigkeiten dieser Klassen und der Ähnlichkeit wurden L5-Entitäten oftmals als M5 klassifiziert. Analog dazu wurden M5-Entitäten selten als L5 klassifiziert.

Das Machtmotiv ist deutlich häufiger als die anderen Motive im Datensatz vorhanden. Aufgrund dessen wurden viele Entitäten als Machtmotiv klassifiziert. Innerhalb des Machtmotivs sind die Klassen M4 und M5 besonders gut zu klassifizieren. Die beiden Klassen sind auch am häufigsten im Datensatz vorhanden. Die Klassifizierung der M0-Entitäten besitzt dieselbe Problematik, wie bei den Klassen A0 und L0. Diese Klassen sind zu selten im Datensatz vorhanden. Die Klasse M3 wurde häufig als M4 klassifiziert. Zwischen beiden Klassen existieren 18 von 20 gemeinsame Wörter (Mit Stoppwörter) bzw. zwölf von 20 gemeinsame Wörter (Ohne Stoppwörter). Obwohl die Gemeinsamkeit beider Klassen höher ist, als die zwischen L1 und L2, so ist die Wahrscheinlichkeit für ei-

ne falsche Klassifizierung niedriger. Es wurden 22% der M3 Entitäten als M4 klassifiziert, analog dazu wurden 32% der L1-Entitäten als L2 klassifiziert, weil die Klasse M3 häufiger im Datensatz vorkommt, als L2, sodass dies zu einer besseren Differenzierbarkeit führt. M2-Entitäten werden oftmals als L2 klassifiziert. Auf Basis der Worthäufigkeiten, ohne Stoppwörter, lässt sich keine verlässliche Aussage treffen. Die 30 häufigsten Worte beider Klassen haben zwar zehn Worte gemeinsam, wovon aber die Wörter "stark" und vor allem "stolz" spezifischer für die Klassen gelten, da diese nicht häufig in der Testmenge vorkommen. Generell lässt sich feststellen, dass vor allem im Bereich des Machtmotivs viele Entitäten klassifiziert wurden. Dies lässt sich auf das Ungleichgewicht des Datensatzes zurückführen, da 58% der Entitäten einer machtmotivierten Klasse angehören. Eine weitere Auffälligkeit ist, dass in beiden Confusion Matrizen ähnliche Ergebnisse erkennbar sind, sodass der relative Abstand von zwei Werten in beiden Matrizen ungefähr gleich sind.

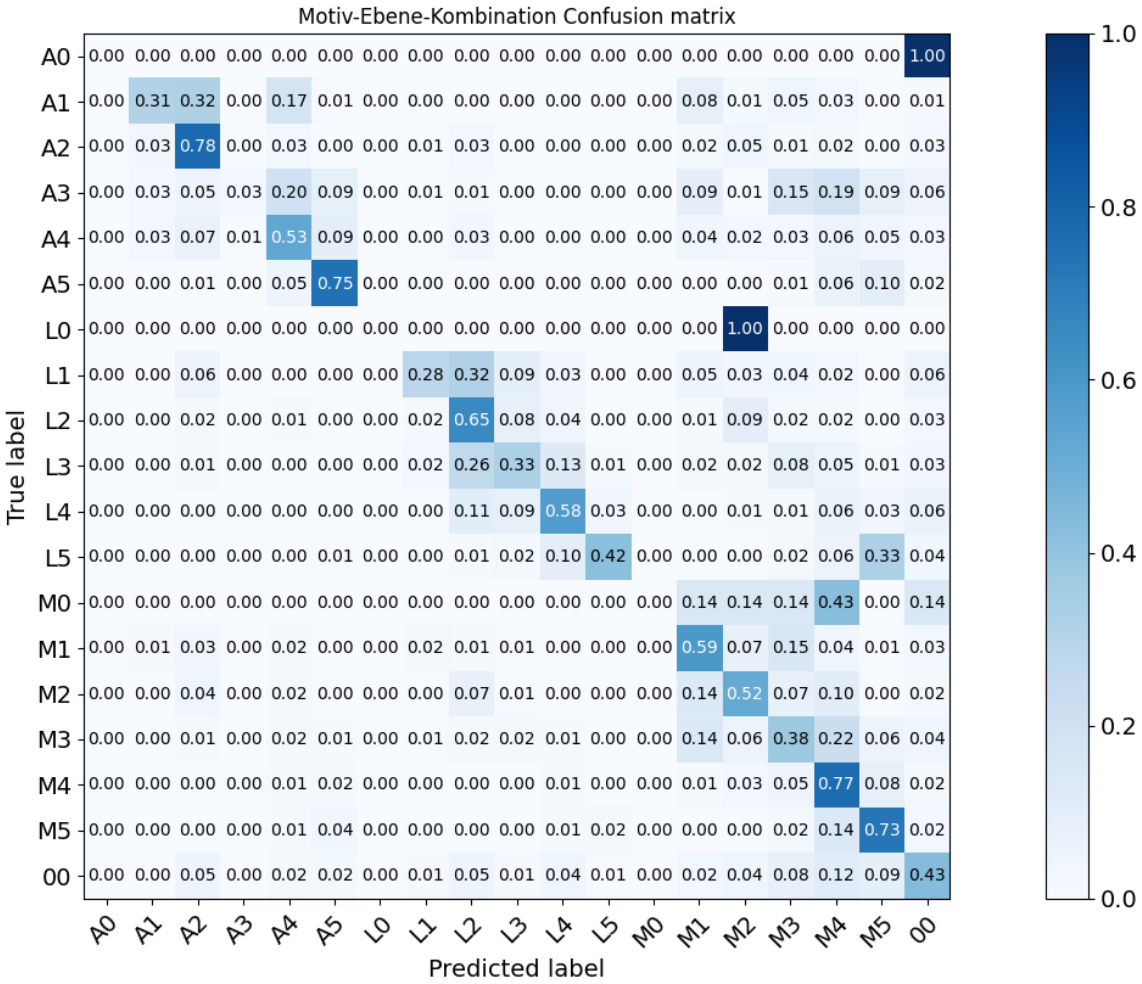

Abbildung 7.1: Diese Darstellung zeigt die Confusion Matrix des LSTM-BoW-Modells. Die Werte sind normalisiert. Die Labels sind die einzelnen Klassen, nach denen klassifiziert wurde.

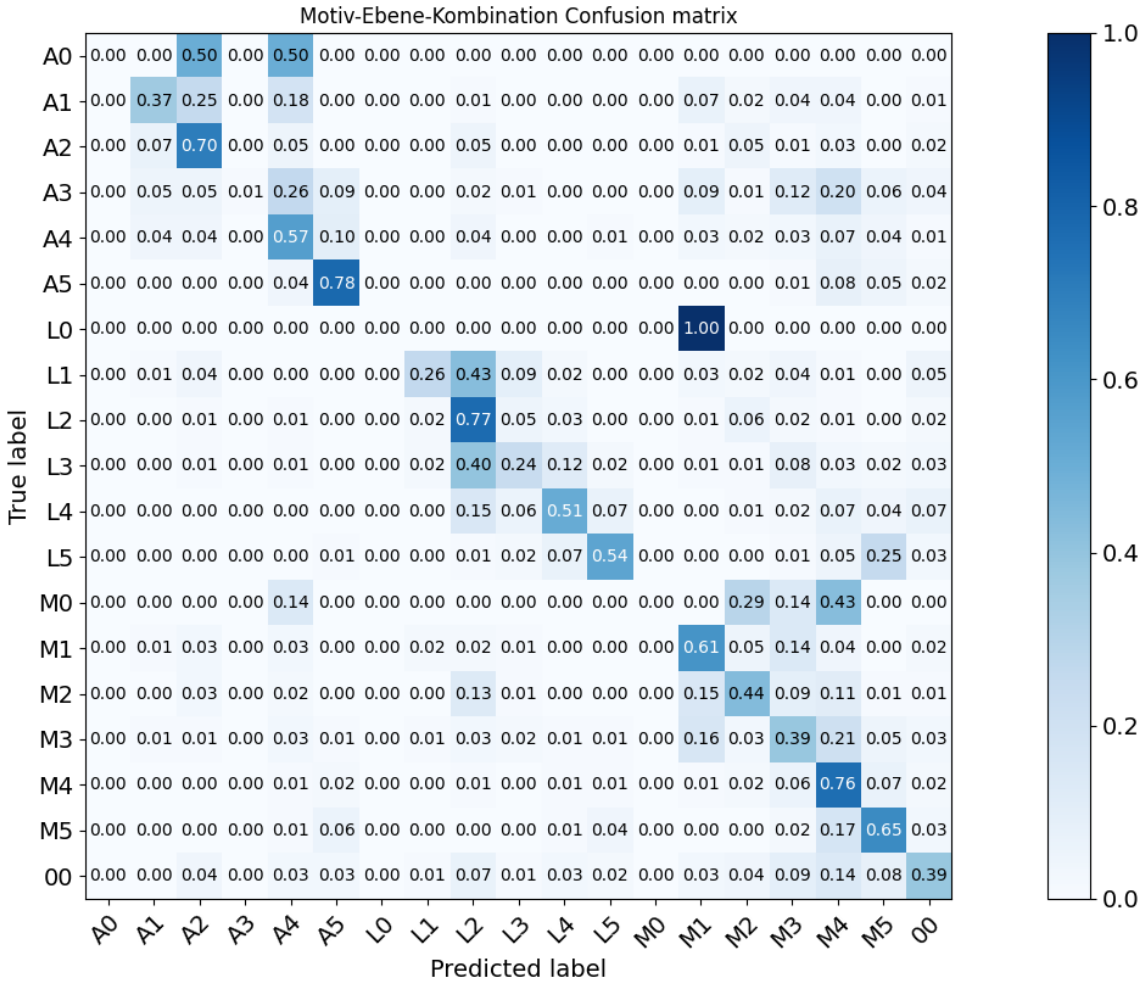

Abbildung 7.2: Diese Darstellung zeigt die Confusion Matrix des LSTM-W2V-Modells. Die Werte sind normalisiert. Die Labels sind die einzelnen Klassen, nach denen klassifiziert wurde.

Die Analyse zeigt, dass bestimmte Klassen viele gemeinsame Wörter miteinander teilen und so eine Klassifizierung für die häufig aufkommende Klasse ausfällt, wie dies in den Beispielen erkennbar ist. Prinzipiell lässt sich in den Confusion Matrizen eine Diagonale feststellen, welche die korrekte Klassifizierung der Klassen angibt. Ob Stoppwörter mit in die Analyse einbezogen werden oder nicht, hätte in diesem Fall keinen Unterschied des Ergebnisses herbeigeführt.

Wenn A1-Entitäten falsch klassifiziert wurden, dann meistens als M1. Die falsch klassifizierten A2-Entitäten wurden häufig als M2 klassifiziert. Für die Klassen A4 und A5 ist dies ebenfalls zu erkennen. Auch beim Leistungsmotiv ist diese Auffälligkeit zu sehen. Dies gilt auch für die falsch klassifizierten L3-Entitäten. Die Modelle haben Schwierigkeiten zwischen Ebene-drei und Ebene-vier zu klassifizieren. Dies führt dazu, dass die falsch klassifizierten A3-Entitäten häufig als M4 klassifiziert wurden. Nichtsdestotrotz

wurden viele dieser Entitäten auch als M3 klassifiziert. Diese Auffälligkeit spricht für eine Unabhängigkeit der Ebenen.

Das MLP-BoW-Modell besitzt 75.180.019, das LSTM-BoW-Modell 304.667.019 und das LSTM-W2V-Modell 2.818.919 trainierbare Gewichtungen. Nichtsdestotrotz ist die Anzahl der Parameter nicht ausschlaggebend für bessere Ergebnisse, da beim LSTM-W2V-Modell eine Hidden Size von 700 bessere Ergebnisse erzielt hat als mit einer Hidden Size von 1000, welches zu mehr Gewichtungen geführt hätte. Durchschnittlich dauerte das Training des MLP-BoW-Modells 45 Minuten. Beim LSTM-BoW-Modell dauert es ungefähr vier bis fünf Stunden und beim LSTM-W2V-Modell sind es ca. 20 Minuten. Es zeigt sich, dass das LSTM-W2V-Modell sehr effizient bezüglich Größe und Trainingsdauer ist.

7.2 Evaluation MLP-W2V/FT-Modell

Es ist deutlich zu erkennen, dass die Klassifizierung einzelner Wörter schlechte Ergebnisse hervorgebracht hat, als die Klassifizierung ganzer Texte. Im Datensatz sind die Motive, als auch die Ebenen ungleichmäßig verteilt. Diese Tatsache ermöglicht es, dass Stoppwörter eine Bedeutung für häufig vorkommende Motive und Ebenen darstellt, welche dahingehend auch klassifiziert werden. Das MLP-FT-Modell hat ca. 75% der Testdaten als M4 klassifiziert, wobei nur ca. 22% der Testdaten der Motiv-Ebene-Kombination angehören. Klassen, welche weniger als 2% im Datensatz auftauchen, wurden nicht klassifiziert. Das MLP-FT-Modell hat einen Precision Score von 0,535 und einen Recall Score von 0,3321 erreicht, während das MLP-BoW-Modell einen Precision Score von 0,5963 und einen Recall Score von 0,6075 hat. Dies zeigt, dass das MLP-FT-Modell, wenn es eine andere Klasse, als M4, klassifiziert hat, dann war die Klassifizierung häufig korrekt. Dasselbe gilt auch für das MLP-W2V-Modell.

7.3 Evaluation CapsNet-Modelle

Weshalb die CapsNet-Modelle, im Gegensatz zu den meist anderen Modellen, schlechter sind, ist schwierig zu beantworten. In vielen Textklassifizierungsaufgaben erreichte das CapsNet State-of-the-Art Werte, wie bspw. bei Zhao et al. [2018]. Es gibt aber auch Forschungsarbeiten, bei denen das CapsNet ebenso schlechte Ergebnisse hervorbrachte. Beispielsweise bei Kim et al. [2019], dort erreichte das CapsNet auf dem IMDb-Datensatz [Maas et al., 2011] einen F1-Score von 0,898. Der F1-Score des State-of-the-Art Modells liegt bei 0,941. Vergleicht man die Confusion Matrix des Caps-W2V-Modells, Abbildung 7.3, mit den anderen Confusion Matrizen, so zeigt sich, dass die Entitäten häufiger als Machtmotiv klassifiziert werden. Generell werden die Klassen A2, L2, M1, M2, M4 und M5 häufiger klassifiziert, als dies in den anderen Modellen der Fall ist. Es ist festzustellen, dass diese Klassen auch sehr häufig im Datensatz vorkommen. Zwar gilt dies auch

für die Klasse M3, doch die vorherige Analyse hat gezeigt, dass es eine große Gemeinsamkeit mit der Klasse M4 gibt. Ähnliche Ergebnisse wurden auch beim Caps-FT-Modell erreicht. Es sollte aber auch festgehalten werden, dass die Klassen, welche häufiger klassifiziert werden, dennoch keine höhere Accuracy besitzen, im Gegensatz zu den anderen Modellen, welche einen besseren F1-Score erreicht haben.

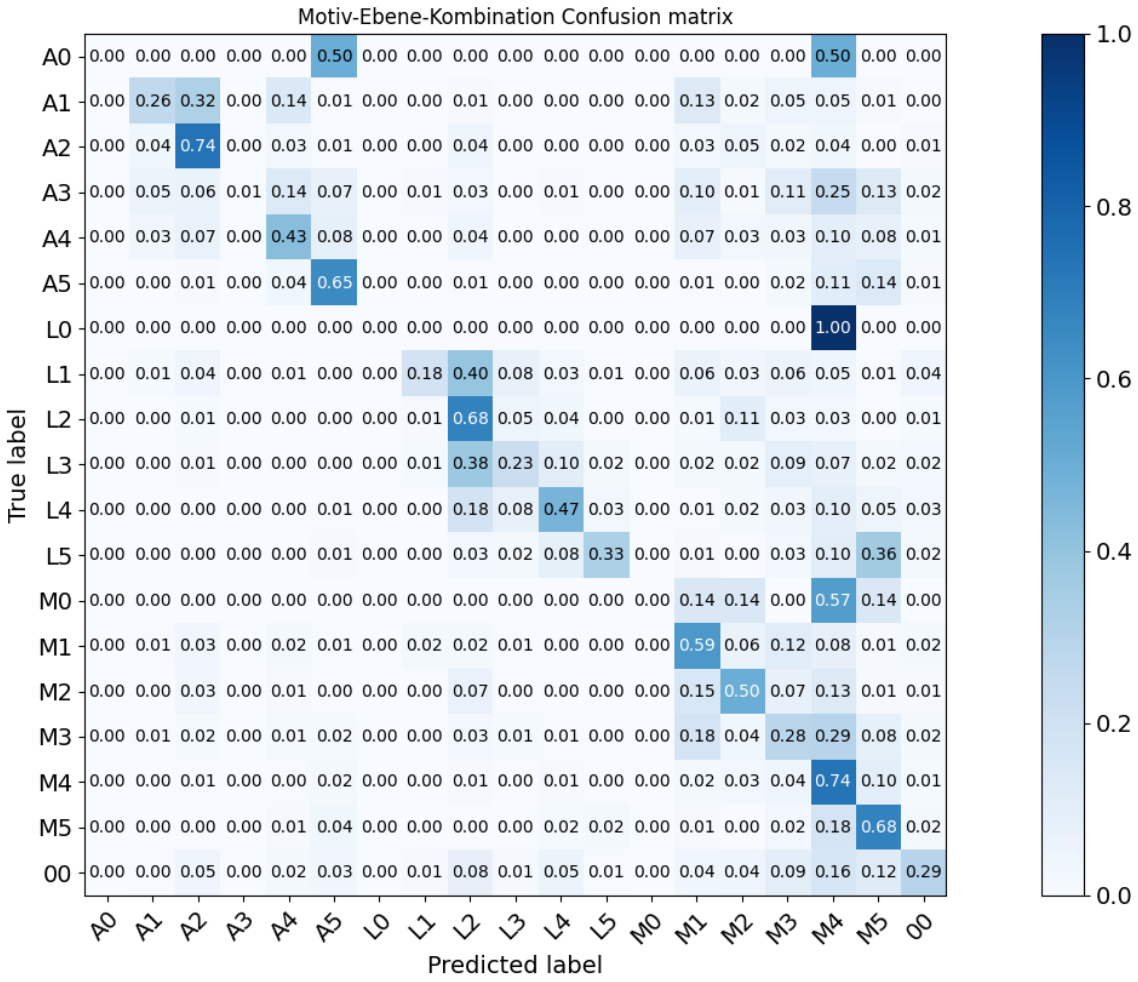

Abbildung 7.3: Diese Darstellung zeigt die Confusion Matrix des Caps-W2V-Modells. Die Werte sind normalisiert. Die Labels sind die einzelnen Klassen, nach denen klassifiziert wurde.

7.4 Evaluation isolierte Klassifizierung

Das LSTM-BoW- und das LSTM-W2V-Modell werden für die isolierte Klassifizierung verwendet. Diese Klassifizierung soll aufzeigen, ob auf Basis der Daten eine Abhängigkeit zwischen den Kategorien (Motiv und Ebene) existiert. Sollten die Ergebnisse schlechter sein, als die bereits vorgestellten Ergebnisse, bei denen die Kategorien zusammen betrachtet wurden, dann ist dies als ein Indiz für eine Abhängigkeit dieser Kategorien zu sehen. Es kann hier auch von einer einseitigen Abhängigkeit gesprochen werden, sofern

nur ein Wert schlechtere Ergebnisse bei der isolierten Klassifizierung hervorbringt. Zur Veranschaulichung zeigen die Abbildungen 7.4 und 7.5 die Confusion Matrizen der isolierten Klassifizierung der Motive und Ebenen. In der Tabelle 6.5 sind die Ergebnisse der isolierten Klassifizierung zu erkennen. Die F1-Scores beider isolierter Klassifizierungen sind über dem F1-Score, welcher für die Klassifizierung der Motiv-Ebene-Kombinationen erreicht wurde. Das LSTM-BoW-Modell erreicht dabei einen F1-Score von 0,8202 für die Klassifizierung der Motive und ein F1-Score von 0,6561 für die Klassifizierung der Ebenen. Beim LSTM-W2V-Modell wurde für Klassifizierung der Motive ein F1-Score von 0,8026 und für die Ebenen ein F1-Score von 0,6183 erreicht. Anhand dieser Ergebnisse lässt sich also kein Indiz für eine Abhängigkeit zwischen Motiven und Ebenen feststellen.

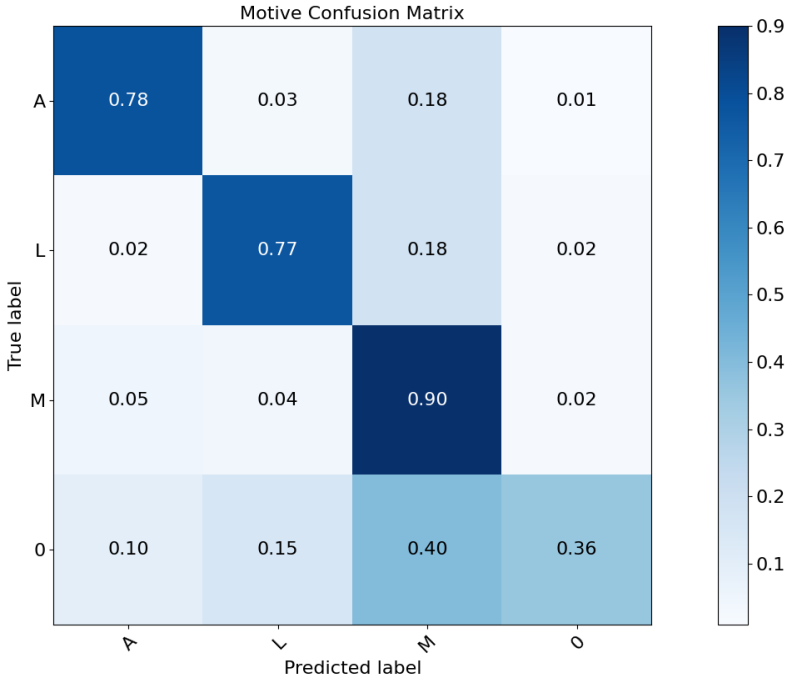

Abbildung 7.4: Diese Darstellung zeigt die Confusion Matrix des LSTM-BoW-Modells für die isolierte Klassifizierung der Motive. Die Werte sind normalisiert. Die Labels sind die einzelnen Klassen, nach denen klassifiziert wurde. Dabei steht *A* für das Anschlussmotiv, *L* für das Leistungsmotiv, *M* für das Machtmotiv und *0* für das Null-Motiv.

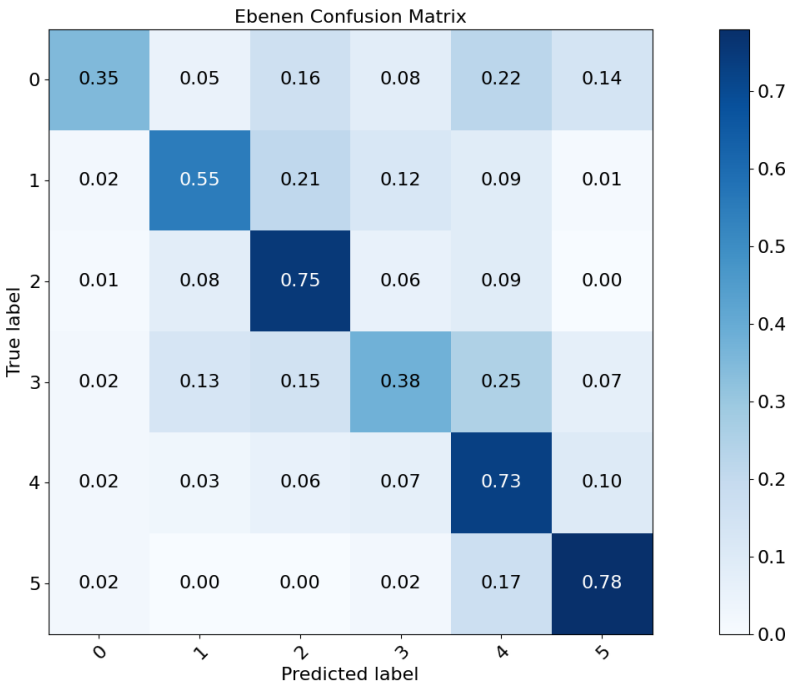

Abbildung 7.5: Diese Darstellung zeigt die Confusion Matrix des LSTM-BoW-Modells für die isolierte Klassifizierung der Ebenen. Die Werte sind normalisiert. Die Labels sind die einzelnen Klassen, nach denen klassifiziert wurde.

Diese Analyse wurde mit der Verwendung von Stoppwörter erstellt. Die gemeinsamen Worthäufigkeiten innerhalb der klassifizierten Motive und Ebenen zeigen auch keine offensichtliche Abhängigkeit. Beispielsweise haben die Entitäten, welche als Ebene-vier klassifiziert wurden 84 von 100 gemeinsame Wörter mit dem Machtmotiv. Beim Leistungsmotiv sind das 75 und beim Anschlussmotiv 68 gemeinsame Wörter. Es sei zu erwähnen, dass 76,55% der Ebene-vier-Entitäten zum Machtmotiv gehören. Daher kann nicht von einer Abhängigkeit gesprochen werden. Wird die Gemeinsamkeit zwischen der Ebene-zwei und den Motiven verglichen, so hat das Anschlussmotiv 71, das Leistungsmotiv 72 und das Machtmotiv 55 gemeinsame Wörter, wobei nur ca. 27,78% der Ebene-zwei-Entitäten dem Anschlussmotiv, 36,59% dem Leistungsmotiv und 36,61% dem Machtmotiv angehören. Es fällt zwar auf, dass kaum gemeinsame Wörter mit dem Machtmotiv existieren, dennoch kann hier nicht von einer Abhängigkeit gesprochen werden.

Es lässt sich der F1-Score für die Klassifizierung der Motive und der Ebenen aus der Klassifizierung der Motiv-Ebene-Kombinationen berechnen. Es stellt sich heraus, dass der F1-Score für die Klassifizierung der Motive bei 0,8251 liegt. Für die Klassifizierung der Ebenen wurde ein F1-Score von 0,6661 erreicht. Beide F1-Scores sind höher, als diese, welche bei der isolierten Klassifizierung erreicht wurden. Dabei muss erwähnt werden, dass das Hyperparameter-Tuning auf der Basis der Motiv-Ebene-Kombinationen

gemacht wurde. Da beide F1-Scores höher liegen, kann hier nicht von einer einseitigen Abhängigkeit gesprochen werden. Prinzipiell sollte auch nicht von einer allgemein geltenden Abhängigkeit gesprochen werden, da beide Kategorien sich unabhängig von der anderen Kategorie klassifizieren lassen und einen besseren F1-Score erreichten, als bei der Klassifizierung der Motiv-Ebene-Kombinationen.

Johannßen und Biemann [2019] haben ebenfalls den OMT nach den Motiven klassifiziert. Dabei kamen die Autoren auf einen F1-Score von 0,8155 $\pm$ 0,0009. Dafür wurde ein LSTM mit einem Attention-Mechanismus [Young et al., 2018] verwendet. Meyer [2019] hat ebenfalls die Motive isoliert klassifiziert und einen F1-Score von 0,8443 erreicht. Dafür hat der Autor BERT als Methode verwendet. Es zeigt sich, dass die, in dieser Bachelorthesis, verwendeten Modelle gute Ergebnisse hervorbringen. In der Klassifizierung der Ebenen wurden State-of-the-Art Werte erreicht, da der bisherige F1-Score bei 0,6540 lag [Johannßen et al., 2019], während in dieser Bachelorthesis ein F1-Score von 0,6561 gemessen wurde.

8 Diskussion

Die Klassifikation der Motiv-Ebene-Kombinationen, stammend aus dem OMT, stellt sich als eine schwierige Herausforderung dar. In dem verwendeten Datensatz sind die Klassen ungleich verteilt. 21,53% der Entitäten gehören der Klasse M4 an, was für insgesamt 19 Klassen ein hoher Anteil ist. 58% der Entitäten gehören dem Machtmotiv an. Weiterhin besitzt dieser Datensatz eine hohe Anzahl an seltenen Wörter. Als seltenes Wort gelten in dieser Arbeit Wörter, die weniger als dreimal im Datensatz vorkommen. Ungefähr 70% aller Wörter sind seltene Wörter. Dies liegt unter anderem an den vielen Rechtschreibfehlern, als auch an den Fragen selbst, da diese einen großen Spielraum an Antworten zulassen und so die Vielfalt zustande kommt. Um diese Probleme möglicherweise aus dem Weg zu gehen, wäre es möglich, die Verteilung mittels Oversampling anzupassen, sodass die Klassen gleichmäßiger verteilt sind. Durch Oversampling werden die Entitäten von Klassen, welche selten auftreten, mehr oder weniger dupliziert [Ah-Pine und Soriano-Morales, 2016]. Als weitere mögliche Maßnahme könnten die falsch geschriebenen Wörter verbessert werden, sodass die Anzahl der selten vorkommenden Wörter geringer ist.

Für die Klassifizierung der Motiv-Ebene-Kombinationen wurde ein F1-Score von 0,6016 $\pm$ 0,001 und eine Accuracy von 0,6085 $\pm$ 0,0015 gemessen. Das MLP-BoW und alle LSTM-Modelle haben die aufgestellte Baseline, welche einen F1-Score von 0,5556 und eine Accuracy von 0,5582 erreicht hat, übertroffen. Es zeigt sich deutlich, dass die State-of-the-Art Architekturen oder Word Embeddings nicht immer die besseren Ergebnisse erreichen. Es gilt festzuhalten, dass das Word2Vec-Modell immer bessere Ergebnisse erzielt hat, als das fastText-Modell, sofern die gleiche neuronale Architektur verwendet wurde. Diese Arbeit hat die ersten Ergebnisse für die Klassifizierung der Motiv-Ebene-Kombinationen gemessen, sodass das LSTM-BoW-Modell das State-of-the-Art Modell darstellt. Weiterhin hat das LSTM-BoW-Modell auch State-of-the-Art Werte in der Klassifizierung der Ebenen erreicht. Das CapsNet hat keine guten Ergebnisse auf diesen Datensatz mit diesem Vorgehen hervorgebracht. Es zeigte sich, dass dieses Modell mit dem Ungleichgewicht des Datensatzes Schwierigkeiten hatte.

Deep Learning-Methoden können zur automatischen Auswertung des OMTs genutzt werden. Es sollte aber differenziert werden, für welchen Kontext diese Modelle genutzt werden sollen. Für die reelle Praxis sind diese, in dieser Arbeit, entwickelten Modelle nicht geeignet. Die Accuracy für diese Modelle ist zu gering,

als dass Entscheidungen über Menschen dafür getroffen werden können. Weiterhin sind Deep Learning-Methoden sehr schwierig zu interpretieren, sodass die Art und Weise wie es zu einer Klassifizierung kam, nicht ersichtlich ist. Dennoch zeigt diese Bachelorthesis, dass sich Deep Learning als Methodik eignen kann. Es zeigt aber auch, dass vor allem einfache Modelle gute Ergebnisse erreichen. Beispielsweise haben die Satzrepräsentationen des BoWs die beiden besten Ergebnisse erzielt. Meyer [2019] zeigt, dass auch etwas kompliziertere Modelle, wie BERT, zu State-of-the-Art Ergebnissen kommen können.

Diese Bachelorthesis konnte kein Indiz für eine Abhängigkeit zwischen Motive und Ebenen feststellen. Es zeigte sich, dass die Klassifizierung beider Kategorien zu einem schlechteren F1-Score führt, verglichen mit den F1-Scores aus der isolierten Klassifizierung. Auch bei der Analyse der klassifizierten Entitäten konnten keine Indizien gefunden werden. Eher sind Indizien für eine Unabhängigkeit zu sehen, sodass in der Klassifizierung der Motiv-Ebene-Kombinationen bspw. den falsch klassifizierten Entitäten häufig die richtige Ebene, aber das falsche Motiv zugeordnet wurde. Es zeigt sich, dass das Trainieren der Modelle für die Klassifizierung der Motiv-Ebene-Kombinationen zu leicht besseren Ergebnisse führen kann, für die isolierte Klassifizierung der Motive oder Ebenen.

9 Zusammenfassung

Für die Klassifizierung der Motiv-Ebene-Kombinationen wurde ein F1-Score von 0,6016 $\pm$ 0,0008 und eine Accuracy von 0,6085 $\pm$ 0,0015 erreicht. Aufgrund der Tatsache, dass keine anderen Ergebnisse existieren, ist das LSTM-BoW-Modell das State-of-the-Art Modell. Auch das LSTM-W2V- und das MLP-BoW-Modell erreichten gute Ergebnisse. Die LSTM-Modelle haben bessere Ergebnisse als die Baseline erreicht, ebenso das MLP-BoW-Modell. Die Caps-Modelle konnten keine besseren Ergebnisse erzielen. Die Analyse zeigt, dass vor allem das Ungleichgewicht des Datensatzes ein Problem darstellt. Es zeigt sich auch, dass BoW besser funktioniert hat, als die repräsentationslernenden Methoden. Die Klassifizierung der einzelnen Wörter erweist sich als besonders ineffektiv. Für diese Klassifizierung wurde das MLP-W2V- und das MLP-FT-Modell verwendet. Es zeigt sich, dass Deep Learning als Methode geeignet ist für die automatische Auswertung des OMTs, wobei keines der Modelle in der realen Praxis angewendet werden sollte.

Weiterhin konnte kein Indiz für eine Abhängigkeit zwischen Motiven und Ebenen gefunden werden, zumindest nicht in diesem Datensatz. Es wurden bessere Ergebnisse für die isolierte Klassifizierung der Motive und Ebenen erreicht. Auch die klassifizierten Entitäten zeigten keine Auffälligkeit, welche eine Abhängigkeit aufweisen. Für die isolierte Klassifizierung der Motive wurde ein F1-Score von 0,8202 mit dem LSTM-BoW-Modell erreicht. Der State-of-the-Art Wert liegt bei 0,8443 [Meyer, 2019]. Für die Klassifizierung der Ebenen hat das LSTM-BoW-Modell einen F1-Score von 0,6561 erreicht, welches ein neuer State-of-the-Art Wert ist, verglichen mit dem vorherigen Wert von Johannßen et al. [2019].

Literaturverzeichnis

Ah-Pine, J. und Soriano-Morales, E.-P. (2016). A study of synthetic oversampling for twitter imbalanced sentiment analysis. In *Workshop on Interactions between Data Mining and Natural Language Processing (DMNLP 2016)*, pages 17–24, Riva del Garda, Italy.

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Hasan, M., Esesn, B. C. V., Awwal, A. A. S., und Asari, V. K. (2018). The history began from AlexNet: A comprehensive survey on deep learning approaches. *CoRR*, abs/1803.01164.

Aly, R., Remus, S., und Biemann, C. (2019). Hierarchical multi-label classification of text with capsule networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, page 323–330, Florence, Italy.

Atkinson, J. W. (1958). *Motives in fantasy, action, and society: A method of assessment and study.* Van Nostrand, Princeton, New York, USA.

Atkinson, J. W. und O'Connor, P. (1966). Neglected factors in studies of achievement-oriented performance: Social approval as an incentive and performance decrement. *A theory of achievement motivation. New York: Wiley*, pages 299–325.

Baumeister, R. F. und Leary, M. R. (1995). The need to belong: desire for interpersonal attachments as a fundamental human motivation. *Psychological Bulletin*, 117(3):497–529.

Bayes, T. (1763). An essay towards solving a problem in the doctrine of chances. *Phil. Trans. of the Royal Soc. of London*, 53:370–418.

Bird, S., Klein, E., und Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media, Inc., Newton, Massachusetts, USA.

Bojanowski, P., Grave, E., Joulin, A., und Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Bosselut, A., Levy, O., Holtzman, A., Ennis, C., Fox, D., und Choi, Y. (2018). Simulating action dynamics with neural process networks. In *6th International Conference on Learning Representations, ICLR 2018, Conference Track Proceedings*, Vancouver, British Columbia, Canada.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bougares, F., Schwenk, H., und Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734, Doha, Qatar.

Devlin, J., Chang, M., Lee, K., und Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 4171–4186, Minneapolis, Minnesota, USA.

Dobert, T. (2019). Sentiment analysis of informal online texts with neural networks. Master's thesis, Universität Hamburg, Hamburg, Germany. Retrieved from: `https://www.inf.uni-hamburg.de/en/inst/ab/lt/teaching/theses/completed-theses/2019-ma-dobert.pdf`.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202.

Gaussier, E. und Yvon, F. (2013). *Textual information access: statistical models*. John Wiley & Sons, Hoboken, New Jersey, USA, 1 edition.

Goodfellow, I., Bengio, Y., und Courville, A. (2016). *Deep learning*. MIT press, Cambridge, Massachusetts, USA, 1 edition.

Grave, E., Bojanowski, P., Gupta, P., Joulin, A., und Mikolov, T. (2018). Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, pages 3483–3487, Miyazaki, Japan.

Hahnloser, R. H., Sarpeshkar, R., Mahowald, M. A., Douglas, R. J., und Seung, H. S. (2000). Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*, 405(6789):947–951.

Heckhausen, J. und Heckhausen, H. (2006). *Motivation und Handeln*. Springer-Verlag, Heidelberg, Germany, 3 edition.

Henaff, M., Weston, J., Szlam, A., Bordes, A., und LeCun, Y. (2016). Tracking the world state with recurrent entity networks. *CoRR*, abs/1612.03969.

Hinton, G. E., Krizhevsky, A., und Wang, S. D. (2011). Transforming auto-encoders. In *Artificial Neural Networks and Machine Learning - ICANN 2011 - 21st International Conference on Artificial Neural Networks, Proceedings, Part I*, pages 44–51, Espoo, Finland. Springer.

Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut für Informatik, Technische Universität München, München, Germany. Retrieved November 3, 2019, from: `http://people.idsia.ch/~juergen/SeppHochreiter1991ThesisAdvisorSchmidhuber.pdf`.

Hochreiter, S. und Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hornik, K., Stinchcombe, M., und White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

Howard, J. und Ruder, S. (2018). Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia.

Ismail, A. A., Wood, T., und Bravo, H. C. (2018). Improving long-horizon forecasts with expectation-biased lstm networks. *CoRR*, abs/1804.06776.

Johannßen, D. und Biemann, C. (2018). Between the lines: Machine learning for prediction of psychological traits - A survey. In *Machine Learning and Knowledge Extraction - Second IFIP TC 5, TC 8/WG 8.4, 8.9, TC 12/WG 12.9 International Cross-Domain Conference, CD-MAKE 2018, Proceedings*, pages 192–211, Hamburg, Germany.

Johannßen, D., Biemann, C., und Scheffer, D. (2019). Reviving a psychometric measure: Classification and prediction of the operant motive test. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pages 121–125, Minneapolis, Minnesota, USA.

Johannßen, D. und Biemann, C. (2019). Neural classification with attention assessment of the implicit-association test omt and prediction of subsequent academic success. In *Preliminary proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019): Long Papers*, pages 68–78, Erlangen, Germany.

Kim, J., Jang, S., Park, E., und Choi, S. (2019). Text classification using capsules. *Neurocomputing*.

Kim, S.-B., Han, K.-S., Rim, H.-C., und Myaeng, S.-H. (2006). Some effective techniques for naive bayes text classification. *Knowledge and Data Engineering, IEEE Transactions on*, 18:1457–1466.

Kingma, D. P. und Ba, J. (2015). Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*, San Diego, California, USA.

Koestner, R. und McClelland, D. C. (1992). The affiliation motive. In *Motivation and Personality: Handbook of Thematic Content Analysis*. Cambridge University Press, Cambridge, Great Britain.

Kuhl, J. (2001). *Motivation und Persönlichkeit: Interaktionen psychischer Systeme*. Hogrefe Verlag, Göttingen, Germany, 1 edition.

Kuhl, J. (2004). Eine neue Persönlichkeitstheorie. Retrieved August 13, 2019, from `https://www.psi-theorie.com/app/download/11284415293/Eine%20neue%20Pers%C3%B6nlichkeitstheorie%20PSI-Theorie-light%20Julius%20Kuhl-1.pdf?t=1552669290`.

Kuhl, J. und Scheffer, D. (1999). *Der Operante Multi-Motiv-Test (OMT): Manual [The operant multi-motive-test (OMT): Manual]*. University of Osnabrück, Osnabrück, Germany.

Kuhl, J. und Strehlau, A. (2014). *Handlungspsychologische Grundlagen des Coaching: Anwendung der Theorie der Persönlichkeits-System-Interaktionen (PSI)*. VS Verlag für Sozialwissenschaften, Wiesbaden, Germany.

LeCun, Y., Bengio, Y., und Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

Levy, O., Lee, K., FitzGerald, N., und Zettlemoyer, L. (2018). Long Short-Term Memory as a dynamically computed element-wise weighted sum. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Volume 2: Short Papers*, pages 732–739, Melbourne, Australia.

Locke, E. A. und Latham, G. P. (1990). *A theory of goal setting & task performance.* Prentice-Hall, Inc, Englewood Cliffs, New Jersey, USA.

Luhn, H. P. (1957). A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of Research and Development*, 1(4):309–317.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., und Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150, Portland, Oregon, USA.

Maslow, A. H. (1943). A theory of human motivation. *Psychological Review*, 50(4):370–396.

McCallum, A., Nigam, K., et al. (1998). A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48.

McClelland, D. C. (1975). *Power: The inner experience.* Irvington Publishers, New York City, New York, USA.

McClelland, D. C. und Boyatzis, R. E. (1982). Leadership Motive Pattern and Long-Term Success in Management. *Journal of Applied Psychology*, 67(6):737–743.

McClelland, D. C. und Pilon, D. A. (1983). Sources of adult motives in patterns of parent behavior in early childhood. *Journal of Personality and Social Psychology*, 44(3):564–574.

Meyer, F. (2019). Classification of the Answers of the OMT. Bachelor Thesis, Universität Hamburg, Hamburg, Germany. Retrieved November 5, 2019, from: `https://www.inf.uni-hamburg.de/en/inst/ab/lt/teaching/theses/completed-theses/2019-ba-meyer-omt.pdf`.

Mikolov, T., Chen, K., Corrado, G. S., und Dean, J. (2013). Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Workshop Track Proceedings*, Scottsdale, Arizona, USA.

Mowery, D. L., Park, A., Bryan, C., und Conway, M. (2016). Towards automatically classifying depressive symptoms from twitter data for population health. In *Proceedings of the Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media (PEOPLES)*, pages 182–191, Osaka, Japan.

Murray, H. A. (1938). *Explorations in personality: A clinical and experimental study of fifty men of college age.* Oxford Univ. Press, Oxford, England.

Murray, H. A. (1943). *Thematic apperception test.* Harvard University Press, Cambridge, Massachusetts, USA.

Pennebaker, J. W., Chung, C. K., Frazee, J., Lavergne, G. M., und Beaver, D. I. (2014). When small words foretell academic success: The case of college admissions essays. *PloS one*, 9(12):e115844.

Pennington, J., Socher, R., und Manning, C. (2014). GloVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, Doha, Qatar.

Pérez-Rosas, V., Mihalcea, R., Resnicow, K., Singh, S., und An, L. (2016). Building a motivational interviewing dataset. In *Proceedings of the Third Workshop on Computational Linguistics and Clinical Psychology*, pages 42–51, San Diego, California, USA.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., und Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, USA.

Plutchik, R. (1980). A general psychoevolutionary theory of emotion. *Theories of emotion*, 1(4):3–31.

Rashkin, H., Bosselut, A., Sap, M., Knight, K., und Choi, Y. (2018). Modeling naive psychology of characters in simple commonsense stories. *CoRR*, abs/1805.06533.

Reiss, S. (2004). Multifaceted nature of intrinsic motivation: The theory of 16 basic desires. *Review of General Psychology*, 8(3):179–193.

Ren, H. und Lu, H. (2018). Compositional coding capsule network with k-means routing for text classification. *CoRR*, abs/1810.09177.

Rong, X. (2014). word2vec parameter learning explained. *CoRR*, abs/1411.2738.

Sabour, S., Frosst, N., und Hinton, G. E. (2017). Dynamic routing between capsules. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 3856–3866, Long Beach, Carlifonia, USA.

Sasaki, Y. (2007). The truth of the f-measure. *Teach Tutor mater*, 1(5):1–5.

Scheffer, D. (2001). *Entwicklungsbedingungen impliziter Motive: Bindung, Leistung & Macht*. PhD thesis, Universität Osnabrück, Osnabrück, Germany. Retrieved from: `https://repositorium.ub.uni-osnabrueck.de/bitstream/urn:nbn:de:gbv:700-2001092518/1/E-Diss150_thesis.pdf`.

Schlebusch, P., Kuhl, J., Breil, J., und Püschel, O. (2006). Alkoholismus als Störung der Affektregulation. *Perspektiven Klärungsorientierter Psychotherapie*, page 60–119.

Schüler, J., Brandstätter, V., Wegner, M., und Baumann, N. (2015). Testing the convergent and discriminant validity of three implicit motive measures: PSE, OMT, and MMG. *Motivation and Emotion*, 39(6):839–857.

Schultheiss, O. C. (2008). *Implicit motives.* The Guilford Press, New York City, New York, USA.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)*, 34(1):1–47.

Solzbacher, C. und Calvert, K. (2016). *Ich schaff das schon...: Wie Kinder Selbstkompetenz entwickeln können*. Verlag Herder GmbH, Freiburg im Breisgau, Germany.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., und Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Wegner, M. und Schüler, J. (2015). Implizite Motive–Perspektiven im Kontext Sport und Bewegung. *Zeitschrift für Sportpsychologie*, 22(1):2–5.

Weindl, D. und Lueger-Schuster, B. (2016). Institutional Abuse (IA) and implicit motives of power, affiliation, and achievement - an alternative perspective on trauma-related psychological responses. In *ISTSS 32nd Annual Meeting. Trauma and Public Health: Innovative Technology and Knowledge Dissemination*, Dallas, Texas, USA.

Werbos, P. (1974). *Beyond regression: New tools for prediction and analysis in the behavioral sciences*. PhD thesis, Harvard University, Cambridge, Massachusetts, USA.

Winter, D. G. und Carlson, L. A. (1988). Using motive scores in the psychobiographical study of an individual: The case of Richard Nixon. *Journal of Personality*, 56(1):75–103.

Xing, Z., Pei, J., und Keogh, E. (2010). A brief survey on sequence classification. *ACM Sigkdd Explorations Newsletter*, 12(1):40–48.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J. G., Salakhutdinov, R., und Le, Q. V. (2019). XLNet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Young, T., Hazarika, D., Poria, S., und Cambria, E. (2018). Recent trends in deep learning based natural language processing. *IEEE Computational Intelligence Magazine*, 13(3):55–75.

Zhao, W., Ye, J., Yang, M., Lei, Z., Zhang, S., und Zhao, Z. (2018). Investigating capsule networks with dynamic routing for text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3110–3119, Brussels, Belgium.

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe angefertigt und mich anderer als der im beigefügten Verzeichnis angegebenen Hilfsmittel nicht bedient habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und die eingereichte schriftliche Fassung der auf dem elektronischen Speichermedium entspricht.
Ich bin mit einer Einstellung in den Bestand der Bibliothek des Fachbereiches einverstanden.

Hamburg, den _____ Unterschrift: _____