h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

Hochschule Darmstadt

– Fachbereich Informatik –

Klassifizierung der Textkomplexität von Chatbot-Antworten mittels Transformer-Modellen

Abschlussarbeit zur Erlangung des akademischen Grades

Master of Science (M.Sc.)

vorgelegt von

Ruben Klepp

Matrikelnummer: 751223

Referent	:	Prof. Dr. Ute Trapp
Korreferent	:	Prof. Dr. Melanie Siegel

ERKLÄRUNG

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, 24. Oktober 2022

Ruben Klepp

ABSTRACT

Many people in Germany have difficulty reading and understanding texts. With the increasing popularity of chatbots and the requirements with regard to the accessibility of digital offerings, it is important to reduce linguistic hurdles when operating a chatbot. An indicator of the readability of a chatbot response can help operators optimize their offerings.

The aim of this master thesis is to find a method that allows chatbot responses to be automatically classified based on their readability. The language classes, easy language, plain language, everyday language and special language serve as a classification into a language level. For this purpose, we will answer the question whether Transformer models are suitable for classification into language levels of chatbot responses and how powerful they are compared to a classical approach based on linguistic features and a trained Support Vector Machine model (SVM model).

For the training of the Transformer models and the SVM model, an extensive corpus of texts from all four language categories was created. Different Transformer language models were trained, each with two hyperparameter configurations, and their performance was compared with each other and with an SVM model that had also been trained.

The evaluation shows that Transformer models are excellent for classifying language levels. All trained Transformer models deliver very good results and are also significantly more powerful compared to the SVM model.

ZUSAMMENFASSUNG

Viele Menschen in Deutschland haben Schwierigkeiten beim Leseverständnis von Texten. Mit der zunehmenden Popularität von Chatbots und den Anforderungen im Hinblick auf die Barrierefreiheit von digitalen Angeboten ist es wichtig, sprachliche Hürden bei der Bedienung eines Chatbots zu reduzieren. Ein Indikator zur Lesbarkeit einer Chatbot-Antwort kann dem Betreiber dabei helfen, sein Angebot zu optimieren.

Ziel dieser Masterarbeit ist es, ein Verfahren zu finden, das es ermöglicht, Chatbot-Antworten anhand ihrer Lesbarkeit automatisch zu klassifizieren. Die Sprachklassen, leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache dienen als Einordnung in ein Sprachniveau. Dazu soll die Frage beantwortet werden, ob sich Transformer-Modelle zur Klassifizierung in Sprachniveaus von Chatbot-Antworten eignen und wie leistungsstark sie gegenüber einem klassischen Ansatz auf Basis von linguistischen Merkmalen und einem trainierten Support Vector Machine Modell (SVM-Modell) sind.

Für das Training der Transformer-Modelle und des SVM-Modells wurde ein umfangreiches Korpus aus Texten aller vier Sprachkategorien angelegt. Es wurden verschiedene Transformer-Sprachmodelle mit jeweils zwei Hyperparameter-Konfigurationen trainiert und deren Leistungsfähigkeit untereinander und gegenüber einem ebenfalls trainierten SVM-Modell verglichen.

Die Evaluation zeigt, dass sich Transformer-Modelle hervorragend für die Klassifizierung von Sprachniveaus eignen. Alle trainierten Transformer-Modelle liefern sehr gute Ergebnisse und sind im Vergleich zum SVM-Modell außerdem deutlich leistungsstärker.

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

ABKÜRZUNGSVERZEICHNIS

NLP Natural Language Processing
ML Machine Learning

Teil I

THESIS

1

EINLEITUNG

In vielen Bereichen des Internets sind Chatbots ein fester Bestandteil der Kommunikation. Der Chatbot ist ein Software-System, dass Anfrage in natürlicher Sprache verarbeiten kann und mit dem Benutzer interagiert. Dabei versucht der Chatbot die Kommunikation mit einem Menschen zu imitieren. Unternehmen, Organisationen und Behörden nutzen diese Art der Kommunikation, um ihren Kunden und den Bürgern eine vergleichsweise direkte Art der Kontaktaufnahme anzubieten. Beispielsweise können in der Kundenbetreuung oder in Behörden mithilfe eines Chatbots automatisiert wiederkehrende Fragen beantwortet werden. Der Benutzer muss dabei nicht selbst zu der gesuchten Information navigieren, sondern kann eine Frage stellen und bekommt idealerweise direkt eine passende Antwort geliefert.

Die Kommunikation mit einem Chatbot kann je nach Implementierung über eine Webseite, direkt über einen Messenger oder auch verbal beispielsweise über einen Smart Speaker erfolgen. Mögliche Antworten eines Chatbots werden von Domänenexperten in einem Backend-System hinterlegt.

Durch die voranschreitende Digitalisierung und aufgrund der Verbreitung von Chatbots kommen zunehmend auch Bevölkerungsgruppen damit in Kontakt, denen eine Kommunikation mit diesen bisher nur eingeschränkt möglich ist.

In Deutschland leben im Jahr 2019 laut dem Statistischen Bundesamt (Bundesamt, 2021) 7,6 Millionen schwerbehinderten Menschen. Davon haben 14% eine geistige oder seelische Behinderungen und sind potenziell eingeschränkt oder nicht in der Lage, Texte zu lesen und zu verstehen. Ein Teil dieser Gruppe sind beispielsweise prälingual Gehörlose. Sie sind bereits gehörlos, bevor sind eine Sprache entwickeln konnten und haben große Schwierigkeiten mit der Schriftsprache. Der Wortschatz dieser Menschen ist häufig sehr klein und es fällt ihnen schwer, komplexe Sätze oder den Bezug des Personalpronomens zu entschlüsseln (Maaß, 2015).

Auch der funktionale Analphabetismus betrifft viele Menschen. Die LEO Studie der Universität Hamburg kommt zu dem Schluss, das ca. 12% der erwerbsfähigen Bevölkerung in Deutschland nicht oder nur unzureichend lesen und schreiben kann (Grotlüschen und Buddeberg, 2020). Viele Geflüchtete verfügen zwar über eine Lesekompetenz in ihrer Muttersprache, lernen die deutsche Sprache aber vorerst meist ohne Unterweisung und ausschließlich in der kommunikativen Praxis (Maaß, 2015). Gerade in der Anfangsphase ihres Spracherwerbs sind sie aber darauf angewiesen, mit Behörden auch schriftlich, beispielsweise per Chatbot zu kommunizieren.

Aus dem Bedarf heraus wurden Ansätze entwickelt, die sprachliche Ausdrucksweise besonders verständlich auszurichten. Der Verein Netzwerk Leichte Sprache e. V. gibt ein Regelwerk heraus, das Regeln zur Sprache, Rechtschreibung und Empfehlungen zur Darstellung von Texten enthält. Texte, die die den Vorgaben des Regelwerks folgen, werden als Leichte Sprache bezeichnet. Zugleich gibt es Bemühungen, Sprache und Texte möglichst einfach zu gestalten, ohne dies jedoch strikt zu regeln. Dies wird als einfache Sprache bezeichnet und liegt näher an der Alltagssprache. Im Rahmen der „Barrierefreie-Informationstechnik-Verordnung“ (BITV) sind die Behörden und Institutionen des Bundes seit 2014 verpflichtet, ihrer Webseiten auch in Leichter Sprache anzubieten.

Im Rahmen dieser Arbeit beziehen sich die Ausdrücke Sprachniveau, Textkomplexität und Lesbarkeit synonym auf die Eigenschaft eines Textes von Menschen der jeweiligen Zielgruppen für leichte Sprache, einfachen Sprache, Alltagssprache und Fachsprache gelesen und verstanden zu werden.

1.1 MOTIVATION

Viele Institutionen, Behörden und Unternehmen möchten gerne Chatbots anbieten und wollen dabei Menschen nicht ausschließen, die Verständnisprobleme bei der Verwendung der normalen Schriftsprache haben. Ein Indikator im Backend eines Chatbots zur Lesbarkeit bzw. zur Komplexität einer Antwort würde es dem Domänenexperten ermöglichen, beim Schreiben der Antwort Rücksicht zu nehmen. Die Sprachkategorien: Leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache helfen dabei, das gewünschte Sprachniveau zu erreichen. So können Antworten auf besonders gute Verständlichkeit auch für die Zielgruppe der leichten und einfachen Sprache optimiert oder Antworten im „Bürokratendeutsch“ reduziert werden.

Die vorliegende Masterarbeit ist als Teil eines Kooperationsprojektes der Hochschule Darmstadt mit der MakeIT Consulting GmbH & Co. KG entstanden. Im Rahmen des Projektes Smart-Multi-Purpose-Emergency-Bot-Tools (SMEBT) werden verschiedene Werkzeuge rund um Chatbot-Platformen entwickelt. Als Bestandteil dieses Projektes soll ein Verfahren entwickelt werden, dass dem Domänenexperten beim Schreiben einer Antwort im Backend einer Chatbot-Plattform eine Rückmeldung zur Lesbarkeit gibt.

1.2 ZIEL DER ARBEIT

Ziel dieser Masterarbeit ist es, ein Verfahren zu finden, das es ermöglicht, Chatbot-Antworten anhand ihrer Lesbarkeit automatisch zu klassifizieren. Die Sprachklassen: Leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache sollen als Einordnung in ein Sprachniveau dienen.

Dazu wird ein Ansatz vorgestellt, der ohne zuvor extrahierte, linguistische Merkmale einer Antwort auskommt und Transformer-Modelle zur Klassifi-

zierung einsetzt. Die Eignung und die Leistungsfähigkeit dieses neuen Ansatzes wird untersucht und zur Einordnung mit einem klassischen Ansatz der Lesbarkeitsforschung auf Basis linguistischer Merkmale und eines Support Vector Machine Models, verglichen.

1.3 AUFBAU DER ARBEIT

Das Kapitel 2 beinhaltet die notwendigen Grundlagen zur Unterscheidung der Sprachniveaus der eingesetzten Werkzeuge zu Entwicklung eines Crawlers und zu Transformer-Modellen sowie zu Support Vector Machines.

Das Kapitel 3 beschreibt den Stand der Technik auf dem Gebiet der klassischen Lesbarkeitsforschung. Dazu werden verschiedene vergleichsweise einfache Lesbarkeitsformeln und Machine Learning (ML)-Ansätze jeweils auf Basis linguistischer Merkmale vorgestellt. Im Kapitel 4 wird der Aufbau eines umfassenden Korpus und das Training mehrere Transformer-Modelle beschrieben. Das Kapitel 5 geht auf die Evaluation der trainierten Modelle. Es wird die Leistungsfähigkeit der unterschiedlichen Ansätze ermittelt und verglichen. Zum Abschluss folgt in Kapitel 6 ein Fazit und ein Ausblick.

2

GRUNDLAGEN

In diesem Kapitel werden die Grundlagen zu Transformer-Modellen und Support-Vector-Machine-Algorithmen erklärt. Außerdem werden die verwendeten Werkzeuge für den Aufbau des Korpus vorgestellt und auf die unterschiedlichen Sprachniveaus wird eingegangen.

2.1 SPRACHNIVEAUS

Im Rahmen der Forschungsfrage dieser Arbeit sollen Chatbot-Antworten hinsichtlich ihrer Sprachniveaus klassifiziert werden. Die Unterschiede der Textkategorien werden nachfolgend dargestellt.

2.1.1 *Leichte Sprache*

Leichte Sprache wurde für Menschen entwickelt, die Lernschwierigkeiten und Schwierigkeiten mit dem Leseverständnis haben. Der Verein Netzwerk Leichte Sprache e.V. hat ein umfassendes Regelwerk erarbeitet. Es sieht Regeln zur Grammatik, Satzbau und auch zur Gestaltung von Texten vor. Die Texte in leichter Sprache müssen von Menschen mit Lernschwierigkeiten auf Verständlichkeit kontrolliert werden, bevor sie als leichte Sprache bezeichnet werden dürfen. Die Übersetzung und den Prüfprozess übernehmen in der Regel Übersetzungsbüros. Mit der Einführung der „Barrierefreie-Informationstechnik-Verordnung“ (BITV) sind staatliche Institutionen verpflichtet, ihre Webseiten auch in leichter Sprache anzubieten.

2.1.2 *Einfache Sprache*

Im Vergleich zur leichten Sprache ist die einfache Sprache etwas komplexer. Sie wird als Sammelbegriff für Texte verwendet, deren Autoren für viele Menschen möglichst verständlich schreiben möchten. Dabei richtet sich einfache Sprache explizit nicht nur an Menschen mit Lernschwierigkeiten, sondern auch an Menschen mit Migrationshintergrund, Flüchtlinge oder funktionale Analphabeten. Obwohl es kein festes Regelwerk gibt, lassen sich folgende Grundsätze feststellen: Auf Fremdwörter sollte, soweit Möglich, verzichtet werden. Wenn sie doch verwendet werden, dann müssen sie erklärt werden. Sätze sollten tendenziell kurz und einfach gehalten werden und klar strukturiert sein. Zudem ist es wichtig, auf Ironie sowie Metaphern zu verzichten und klare eindeutige Aussagen zu vermitteln.

2.1.3 *Alltagssprache*

Die Alltagssprache entspricht der geschriebenen Sprache nach den Regeln der deutschen Rechtschreibung. Zu diesem Sprachniveau gehören Zeitungsartikel, Bücher usw.

2.1.4 *Fachsprache*

Im Vergleich zur Alltagssprache zeichnet die Fachsprache unter anderem einen komplexen Satzbau, abstrakte Formulierungen und die Verwendung von Fach- und Fremdworten aus. Texte dieses Sprachniveaus stammen unter anderem aus wissenschaftlichen Veröffentlichungen und aus dem Rechtskontext.

2.2 WERKZEUGE

Die folgenden Werkzeuge eignen sich für den Aufbau eines Textkorpus. Mit Puppeteer und Cheerio ist es möglich Webscraper zu entwickeln, die in der Lage sind, Inhalte auch von dynamischen Webseiten zu kopieren. Mit spaCy können die kopierten Texte auf vielfältige Weise bereinigt und aufbereitet werden.

2.2.1 *Puppeteer*

Puppeteer wird als NPM-Paket für das JavaScript Ökosystem entwickelt und ermöglicht es, aus dem Code heraus die Funktionen eines *Headless Browsers* oder eines klassischen Webbrowsers automatisiert zu verwenden. Ein *Headless Browser* funktioniert analog zu einem klassischen Webbrowser. Die grafische Ausgabe der Webseiten und die Bedienoberfläche werden allerdings deaktiviert. Über das DevTools Protokolls kann der *Headless Browser* ferngesteuert werden.

Moderne Webseiten setzen häufig auf JavaScript, um im Hintergrund Inhalte nachzuladen oder um die Darstellung einer Webseite gezielt zu steuern. Durch die alleinige Analyse des HTML-Codes, ohne eine Interpretation durch einen Browser und ohne die Ausführung der zugehörigen JavaScript-Dateien, besteht die Möglichkeit, auf Inhalte nicht zugreifen zu können. Aufgrund dessen bietet es sich an, zur Entwicklung von Webscrapern einen *Headless Browser* zu verwenden, der eine Webseite vollständig lädt, interpretiert und deren JavaScript ausführt. Erst dann stehen dem Webscraper möglicherweise alle Inhalte zur Verfügung, wie sie auch ein üblicher Webbrowser darstellen würde. Puppeteer ermöglicht außerdem die Interaktion mit der Webseite und den Zugriff auf den DOM-Baum. Wenn aufgrund einer Interkation mit der Webseite, wie einem Klick auf einen Button, Inhalte dynamisch nachgeladen werden können, kann diese über Puppeteer automatisiert ausgelöst werden.

2.2.2 *Cheerio*

Cheerio ist ein Open-Source-Projekt zur Entwicklung eines NPM Paketes zum Parsen von HTML-Quellcode in der Programmiersprache JavaScript. Das Paket bietet Methoden an, um durch die Knoten des DOM-Baumes zu navigieren und um Text oder Attribute zu extrahieren, zu ändern und zu löschen. Die Syntax der Selektoren zur Auswahl von HTML-Elementen in der Baumstruktur ist die gleiche, wie sie auch bei CSS oder bei jQuery Anwendung findet.

2.2.3 *spaCy*

spaCy ist ein Natural Language Processing (NLP)-Framework für die Verarbeitung und zur Analyse von Texten in Python. Es bietet unter anderem die Möglichkeit Texte zu segmentieren, Wortarten zuzuordnen und automatisiert Objekte, Personen oder Orte aus Texten zu extrahieren.

2.3 SUPPORT VECTOR MACHINE

Eine Support Vector Machine (SVM) ist ein vergleichsweise einfacher Machine Learning (ML)-Algorithmus, der vor allem für Klassifizierungsaufgaben verwendet wird. Als Trainingsdaten werden numerische Merkmale verwendet. Das Ziel des Algorithmuses ist es, eine Hyperplane (Trennebene) in einem n-dimensionalen Raum zu finden, die die Datenpunkte der Merkmale in Klassen separiert. Dabei wird der Abstand (Margin) der Datenpunkte zur Hyperplane maximiert. Um zwischen den Klassen der Datenpunkte zu unterscheiden, wird die Hyperplane als Entscheidungsgrenzen im Klassifikator verwendet.

Die Dimension der Hyperplane hängt von der Anzahl der Merkmale ab. Wenn ein SVM-Modell mit nur zwei Merkmalen trainiert wird, dann ist die Hyperplane nur ein Strich. Bei drei Merkmalen wird die Hyperplane eine zwei-dimensionale Fläche in einem drei-dimensionalen Raum. Wenn die Anzahl der Merkmale drei übersteigt, dann ist der Körper der Hyperplane schwer vorstellbar. Als Support Vectors werden Datenpunkte bezeichnet, die nahe an der Hyperplane liegen und deren Position und Ausrichtung beeinflusst. Mittels dieser Support Vector Datenpunkte kann der Spielraum des Klassifikators vergrößert werden. Wenn die Support Vector Datenpunkte entfernt werden, ändert sich die Position der Hyperplane.

2.4 TRANSFORMER

Die grundlegende Transformer-Architektur wurde im Jahr 2017 im Paper „Attention Is All You Need“ (Vaswani u. a., 2017) vorgeschlagen. Die ursprüngliche Veröffentlichung hatte sich auf Übersetzungsaufgaben fokussiert. Bald darauf wurden weitere Modelle auf Basis der Transformer-Architektur

vorgestellt, die sich auch für andere Aufgaben eignen. Zu den bekanntesten Modellen zählt unter anderem BERT (Devlin u. a., 2019) und GPT (Radford u. a., 2018). Alle diese Transformer-Modelle wurden als sogenannte Sprachmodelle trainiert. Dabei werden unter großem Aufwand sehr große Textkorpora, viel Rechenleistung und viel Zeit für einen selbstüberwachten Lernprozess verwendet. Nach dem Training repräsentieren die Sprachmodelle ein statistisches Verständnis der Sprache, mit deren Texten sie trainiert wurden. In diesem Zustand kann das Modell nicht für eine bestimmte Aufgabe verwendet werden. Das selbstüberwachte Lernen ist eine Art des Trainings, bei dem das Ziel automatisch aus den Eingaben eines Modells berechnet wird. Die Eingabedaten müssen dabei nicht manuell annotiert werden.

2.4.1 *Transfer Learning*

Erst im nächsten Schritt wird ein sogenanntes Fine-Tuning des Sprachmodells für eine spezifische Aufgabe, wie die Klassifizierung von Texten, durchgeführt. Dafür werden annotierte Daten, die zur gewünschten Aufgabe des zukünftigen Modells passen, in einem überwachten Lernprozess nachtrainiert. Weil das grundsätzliche statistische Verständnis des Sprachmodells im Fine-Tuning Prozess wieder verwendet werden kann, wird dieser Prozess als *Transfer Learning* bezeichnet. Im Vergleich zum initialen Training des Sprachmodells werden weniger Texte, weniger Rechenleistung und damit weniger Trainingszeit benötigt. Dafür können zugrundeliegende Sprachmodelle für unterschiedliche Aufgaben verwendet werden. Somit muss der aufwendige initiale Trainingsprozess nicht für jede Aufgabe neu durchgeführt werden. Durch den Austausch der Sprachmodelle über Plattformen wie Hugging Face, können unter ökologisch und ökonomisch nachhaltiger Perspektive Transformer-Modelle für eine Vielzahl an Aufgaben nachtrainiert (Fine-Tuning) werden.

2.4.2 *Decoder und Encoder*

Ein Transformer-Modell kann sich aus zwei Einheiten zusammensetzen: dem Encoder und dem Decoder. Ein Encoder verarbeiten die Eingabedaten und extrahiert Merkmale, die die Eingabedaten repräsentieren. Mit der Verwendung eines Encoders ist das Transformer-Modell optimiert auf die Gewinnung eines Verständnisses der Eingabedaten. Der Decoder verwendet diese Merkmale, zusammen mit weiteren Eingabedaten, um damit eine Ausgabe zu generieren. Mit der Verwendung eines Decoders wird das Transformer-Modell auf die Generierung von Ausgaben optimiert. Beide Einheiten können getrennt voneinander in einem Transformer-Modell verwendet werden. Modelle, die nur einen Encoder verwenden, eignen sich für Aufgaben wie die Textklassifizierung. Für Aufgaben, wie die Textgenerierung, kommen dagegen Modelle zum Einsatz, die nur einen Decoder einsetzen. Die Kombination eines Encoders und eines Decoders eignet sich für Übersetzungsauf-

gaben. Für die Klassifikation der Sprachniveaus werden Encoder-Modelle verwendet.

2.4.3 *Attention Layer*

Ein Kernbestandteil der Transformer-Architektur sind die sogenannten *Attention Layer*. Bei der Verarbeitung der Eingabedaten sorgen sie dafür, dass das Modell bei der Übersetzung der Worte zu Repräsentationen bzw. zu Merkmalen besondere Aufmerksamkeit auf bestimmte Worte richtet und die anderen Worte mehr oder weniger ignoriert werden. Die Bedeutung eines Wortes hängt maßgeblich von seinem Kontext, einem oder mehreren Worten vor oder nach dem gerade betrachtetem Wort, ab. Das lässt sich am Beispiel des Satzes „Das Haus ist groß." verdeutlichen. Die Worte „Haus" und „groß" stehen in Beziehung zueinander. Die verbleibenden Worte sind für diese Beziehung nicht relevant und werden von den *Attention Layern* entfernt, während die Beziehung zwischen den Worten „Haus" und „groß" erfasst wird. Mithilfe einer sogenannten *Attention Mask* werden die *Attention Layer*, der Encoder und Decoder, angewiesen bestimmte Worte zu ignorieren. Dies wird für das Padding von Eingabedaten verwendet. Wenn mehrere Sätze stapelweise verarbeitet werden sollen, aber nicht die gleiche Länge aufweisen, dann werden die kürzeren Sätze um spezielle Füllwörter erweitert. Diese werden mithilfe der *Attention Mask* von der Aufmerksamkeit des Modells ausgeschlossen.

2.4.4 *Nomenklatur*

Im Transformer-Kontext werden die folgenden Bezeichnungen verwendet: Architektur, Checkpoint und Modell. Die Architektur bezeichnet die Grundlage eines Modells. Alle verwendeten Komponenten wie Encoder, Decoder oder beide zusammen und die Ebenen, sowie die internen Prozesse des Transformer-Modells, werden in der Architektur implementiert. Die Gewichte des neuronalen Netzes einer Architektur werden als Checkpoints bezeichnet. Als Modell wird manchmal die Architektur, aber auch die Checkpoints bezeichnet. Der Begriff ist nicht eindeutig definiert.

2.4.5 *Hugging Face und Simple Transformer*

Die Firma hinter der Plattform Hugging Face entwickelt ein populäres Framework[1] für die Arbeit mit und an Transformer-Modellen. Das Framework wird als Transformers bezeichnet. Außerdem wird die Plattform für den Austausch von Transformer-Modellen verwendet. Im Rahmen dieser Arbeit wird das Training der Sprachmodelle mithilfe des Framework Simple Transformer[2] implementiert. Das Framework abstrahiert die Schnittstellen des Fra-

1 https://huggingface.co/docs/transformers/
2 https://simpletransformers.ai/

mework Transformers von Hugging Face und reduziert die Komplexität des Fine-Tunings eines Sprachmodells.

3

STAND DER TECHNIK

Bereits seit den 1920er Jahren beschäftigt sich die Wissenschaft mit der Messung der Komplexität von Texten (Dubay, 2004). Die Initiative ging von Wissenschaftlern in den USA aus. Auf der Suche nach Kriterien zur Auswahl von Büchern für den Unterricht in Schulen wurde im Jahr 1923 eine erste Lesbarkeitsformel von Bertha Lively und S. L. Pressey entwickelt. Die Formel sollte als ein objektives Kriterium eingesetzt werden, um zu entscheiden, welche Bücher sich für welche Jahrgangsstufe eigneten. Dafür wurde gemessen, wie viel neues Vokabular ein Text verwendet. Als neues Wort galt ein Wort, das nicht in einer Liste des damaligen Standardvokabulars aufgeführt wurde. Darüber hinaus wurde auch die Anzahl der nur einmal vorkommenden Worte innerhalb eines 1000-Wort-Blockes gezählt und in der Formel verrechnet. (Dubay, 2004, S. 14). In den nächsten Jahrzehnten beschäftigte sich eine zunehmende Zahl an Wissenschaftlerinnen und Wissenschaftlern mit der Einschätzung der Lesbarkeit von Texten. Ein neues Forschungsfeld entstand.

Rudolf Flesch entwickelte 1948 mit dem Flesch-Reading-Ease eine erste Lesbarkeitsformel, die auf Basis der durchschnittlichen Satzlänge, der durchschnittlichen Silbenzahl pro Wort und statistisch ermittelten Konstanten berechnet wird (Flesch, 1948). Seine Formel begründet eine ganze Kategorie an weiteren Formeln und Indizes, die sich der sogenannten oberflächlichen Merkmalen eines Textes bedient. Als oberflächliche werden sie deshalb bezeichnet, weil sie sich auf einfache syntaktische Merkmale, wie auf die durchschnittliche Satzlänge eines Textes, beschränken. Eine tiefergehende Untersuchung, beispielsweise zur Analyse der semantischen Struktur eines Satzes, findet hingegen nicht statt. Diese Formeln werden als klassische Formeln bezeichnet und im Kapitel 3.1 genauer betrachtet. Der Flesch-Reading-Ease und zahlreiche andere klassische Lesbarkeitsformeln werden auch heute noch zur Einschätzung der Lesbarkeit verwendet. Sie waren bis Anfang der 2000er Jahre führend in der Prognosegenauigkeit.

Die klassischen Formeln mussten sich bei der Analyse der Lesbarkeit auf wenige oberflächliche Merkmale verlassen. Ab den 2000er Jahren und mit der Verfügbarkeit immer größerer Rechenleistung sowie durch die Entwicklungen im Bereich des Natural Language Processing (NLP) und Machine Learning (ML) wurde es möglich, deutlich größere Textmengen effizient automatisiert auszuwerten. So konnten auch tiefgehende Merkmale, beispielsweise auf der semantischen Ebene, automatisiert untersucht werden. Das Kapitel 3.2 beschäftigt sich mit Methoden zur Messung von Textkomplexi-

tät unter Verwendung von Natural Language Processing (NLP) und Machine Learning (ML).

Abseits der Methoden der Computerlinguistik gibt es auch andere Ansätze zur Erfassung der Lesbarkeit eines Textes. Dazu gehört beispielsweise das Hamburger Verständlichkeitsmodell (Langer, Thun und Tausch, 2019). Die Autoren stellen keine Formeln oder Indizes vor, sondern Fragebögen für die Durchführung von empirische Studien in der Zielgruppe eines Textes oder zur Selbstkontrolle der Autorin oder des Autors. Es werden vorwiegend subjektive Kriterien zur Wahrnehmung des Textes durch die Fragebögen erfasst. Die Auswertung einer solchen Studie liefert differenzierte Ergebnisse und bietet die Möglichkeit zur individuellen Verbesserung eines Textes. Da sich diese Ansätze außerhalb der Computerlinguistik befinden und keine Möglichkeit zur automatisierten Kategorisierung eines Textes in leichte Sprache, einfache Sprache, Alltagssprache oder Fachsprache bieten, sind sie kein weiterer Bestandteil dieses Kapitels zum Stand der Forschung.

Je besser die Zusammenhänge eines Textes vom Leser erfasst und verstanden werden können, desto geringer ist dessen Textkomplexität. Über die Verständlichkeit oder Unverständlichkeit eines Textes entscheiden die Charakteristik des Textes, sowie auch die individuellen Fähigkeiten des Lesers (Feng, Elhadad und Huenerfauth, 2009). Zu den individuellen Fähigkeiten eines Lesers zählen unter anderem dessen thematisches Vorwissen, der Umfang des Wortschatzes, sowie dessen Motivation und Aufmerksamkeit. Darüber hinaus sind auch seine Lesekompetenz und die Größe seines Arbeitsgedächtnisses wichtig (Friedrich, 2020). Die vorliegende Arbeit konzentriert sich auf die Untersuchung der Charakteristik eines Textes und nicht auf individuelle Fähigkeiten des Lesers.

In der Literatur werden die Ausdrücke Textverständlichkeit und Lesbarkeit häufig synonym für Textkomplexität verwendet. Dies wird in der Forschung kontrovers diskutiert (Friedrich, 2020, S. 61). Aus Gründen der Vereinfachung und Lesbarkeit der vorliegenden Arbeit wird die synonyme Verwendung fortgeführt.

Die Forschung zur Komplexität von Texten hat ihren Ursprung im angloamerikanischen Sprachraum. Auch heute findet der Großteil der Forschung im Kontext der englischen Sprache statt. Dementsprechend wurden und werden viele Formeln und Modelle auf die englische Sprache zugeschnitten. Die deutsche Sprache spielt in diesem Forschungsfeld im Verhältnis bisher eine tendenzielle kleine Rolle (Friedrich, 2020). Neue Erkenntnisse, Formeln und Modelle sind auch für den deutschen Sprachraum wichtig. Sie lassen sich allerdings nicht ohne weiteres übernehmen und müssen an die deutsche Sprache angepasst werden (Friedrich, 2020, S. 61). Aufgrund dessen konzentriert sich die vorliegende Arbeit und das Kapitel Stand der Technik auf die deutsche Sprache und für diese geeigneten Formeln, Konzepte und Modelle.

3.1 KLASSISCHE LESBARKEITSFORMELN

Seit Beginn der Suche nach Methoden zur präzisen Einschätzung von Textkomplexität wurden über 200 Formeln alleine bis 1980 entwickelt (Dubay, 2004, S. 42). Dabei werden verschiedene Textmerkmale gewichtet miteinander verrechnet. Als Ergebnis errechnet eine solche Formel einen Index, der Auskunft über die Komplexität des untersuchten Textes gibt. Die meisten der Formeln verwenden einen Index, der der Jahrgangsstufe entspricht, in der ein Schüler in der Regel die Lesekompetenz besitzt, um den untersuchten Text zu verstehen. Aber auch Indizes in frei gewählten Wertebereichen sind üblich. Zu den bekanntesten Lesbarkeitsformeln zählt der Flesch-Reading-Ease (Flesch, 1948), Dale und Chall (Dale und Chall, 1948), Gunning Fog (Gunning, 1952), SMOG (McLaughlin, 1969), Flesch-Kinceid (Kincaid u. a., 1975) und Coleman-Liau (Coleman und Liau, 1975). Viele Formeln sind speziell für das amerikanische Schulsystem und die englische Sprache entwickelt worden. Ohne Anpassung an die deutsche Sprache und das deutsche Schulsystem nimmt die Prognosegenauigkeit der Formeln ab.

Zur Berechnung der Indizes werden die sogenannten traditionellen Textmerkmale verwendet. Zu den einfachsten Merkmalen gehört beispielsweise die durchschnittliche Satzlänge. Darüber hinaus werden aber auch andere statistische Faktoren, wie die Wortlänge oder die Wortkomplexität mit einbezogen (Martinc, Pollak und Robnik-Šikonja, 2021, S. 143). Aufgrund der unterschiedlichen Sprachcharakteristiken zwischen der englischen und der deutschen Sprache liefern viele der Formeln, die für die englische Sprache entwickelt wurden, unpräzise Ergebnisse, wenn sie auf einen deutschen Text angewendet werden. Beispielsweise sind deutsche Wörter in der Regel länger als englische Wörter. Formeln, die die Wortlänge mit einbeziehen, müssen demnach in der Gewichtung dieses Merkmals angepasst werden (vgl. (Amstad, 1978)). In der Vergangenheit wurden einige Formeln an die deutsche Sprache oder, in Bezug auf die Indizes, an das deutsche Schulsystem angepasst. Darüber hinaus wurden aber auch eigene Formeln für den deutschen Sprachraum entwickelt.

Zu den bekanntesten Formeln, die für den deutschen Sprachraum angepasst wurden, gehören:

Flesch-Amstad Index

Der Flesch-Reading-Ease (Flesch, 1948) wurde von Toni Amstad (Amstad, 1978) an die deutsche Sprache angepasst, da deutsche Worte im Durchschnitt länger sind als englische Worte. Das Ergebnis der Formel liegt zwischen 0 und 100. Je niedriger der Index, desto schwieriger ist der Text zu lesen. Der Flesch-Amstad Index wird wie folgt berechnet:

$$FleschAmstad = 180 - \frac{\text{Anzahl der Wörter}}{\text{Anzahl der Sätze}} - 58,5 * \frac{\text{Anzahl aller Silben im Text}}{\text{Anzahl der Wörter}}$$

Lesbarkeitsindex (LIX)

Eine der populärsten Formeln zur Berechnung der Lesbarkeit eines Textes ist der LIX Index. Sie wurde von Carl-Hugo Björnsson (Bjornsson, Stockholm und Pedagogiskt centrum, 1968) ursprünglich für die schwedische Sprache entwickelt und lässt sich ohne Anpassung auch auf die deutsche Sprache anwenden. Der Index beginnt bei 15 und geht bis 80. Werte zwischen 15 und 30 sind als sehr leicht, zwischen 30 und 40 als leicht, zwischen 40 und 50 als mittel, zwischen 50 und 60 als schwer und zwischen 60 und 80 als sehr schwer zu interpretieren. Der LIX Index wird wie folgt berechnet:

$$LIX = \frac{\text{Zahl der Wörter}}{\text{Zahl der Sätze}} + 100 * \frac{\text{Zahl der Wörter mit > 6 Buchstaben}}{\text{Zahl der Wörter}}$$

German Simple Measure of Gobbledygook (gSMOG)

Der gSMOG (German SMOG) Index ist die Adaption des Simple Measure of Gobbledygook (SMOG) Index von Harry McLaughlin (McLaughlin, 1969) für die deutsche Sprache durch Bamberger (Bamberger und Vanecek, 1984). Das Ergebnis der Formel entspricht ungefähr der Jahrgangsstufe, die zum Verständnis des untersuchten Textes notwendig ist. Die Formel wird wie folgt berechnet:

$$gSMOG = \sqrt{\frac{\text{(Anzahl der Wörter mit } >= \text{ 3 Silben) * 30}}{\text{Zahl der Sätze}}} - 2$$

Wiener Sachtextformeln

Richard Bamberger und Erich Vanecek haben die sogenannten Wiener Sachtextformeln (Bamberger, 2006) entwickelt. Insbesondere die 4. Wiener Sachtextformel ist weit verbreitet, da sie keinen abstrakten Index berechnet. Ähnlich wie die anderen Formeln gibt sie die Zahl der Schuljahre an, die zum Verständnis des Textes notwendig sind. Der Index beginnt bei 4 und endet bei 15. Ab der Stufe 12 ist der Index allerdings eher als Schwierigkeitsstufe und nicht als Jahrgangsstufe zu verstehen. Der Formel liegt eine Regressionsanalyse von „einigen hundert Jugendbüchern“ zugrunde (Bamberger, 2006, S. 285). Die Formel wird wie folgt berechnet:

$$WSTF_4 = 0,2744 * MS + 0,2656 * SL - 1,693$$

MS: Prozentanteil der Wörter mit drei oder mehr Silben, SL: mittlere Satzlänge (Anzahl Wörter)

Regensburger Index (RIX)

Der Regensburger Index wurde von Johannes Wild und Markus Pissarek entwickelt (Wild und Pissarek, 2020) und bezieht neben den üblichen Parametern, wie der durchschnittlichen Satzlänge usw., auch weitere Schwierigkeitsparameter mit ein. Zu diesen gehören der Anteil der Passivsätze, der Anteil

der Sätze, die Nebensätze beinhalten und der Anteil der Normalisierungen im Text. Das Ergebnis der Formel entspricht ungefähr der Jahrgangsstufe, die für das Verständnis des Textes notwendig ist.

$$SWP = \text{Anteil Sätze im Passiv} + \text{Anteil Sätze mit NS} + \text{Anteil Nominalisierungen}$$

$$RIX = \sqrt{\frac{\text{Anzahl der Wörter}}{\text{Anzahl der Sätze}} + \frac{\text{Anzahl der Wörter mit > 6 Buchstaben}}{\text{Anzahl der Wörter}}} + \sqrt{SWP - 0,260}$$

Neben den aufgeführten Lesbarkeitsformeln existieren zwei weitere Indizes. Sie wurden speziell für die deutsche Sprache entwickelt, aber deren Formel nicht frei veröffentlicht. Zu diesem gehört der Hohenheimer Verständlichkeitsindex (HIX) (Hohenheim, o. D.) und der Bremer Erstlese-Index (BRELIX) (Brügelmann und Brinkmann, 2021).

Der Hohenheimer Verständlichkeitsindex kombiniert vier gängige Lesbarkeitsformeln (Flesch-Amstad Index, 1. neue Wiener Sachtextformel, gSMOG Index und LIX Lesbarkeitsindex) mit sechs eigenständigen Parametern. Zu diesen zählen die durchschnittliche Satzlänge in Wörtern, die durchschnittliche Satzteillänge in Wörtern, die durchschnittliche Wortlänge in Buchstaben, der Anteil der Wörter mit mehr als sechs Buchstaben, der Anteil der Satzteile mit mehr als 12 Wörtern und der Anteil der Sätze mit mehr als 20 Wörtern. Das Ergebnis des Index liegt zwischen 0 (geringe Verständlichkeit) und 20 (hohe Verständlichkeit).

Der Bremer Erstlese-Index wurde in sechs aufeinander aufbauenden Formeln entwickelt. Die Autoren wollten die unterschiedlichen Faktoren, die die Lesbarkeit eines Textes beeinflussen, dediziert abbilden (Brügelmann und Brinkmann, 2021, S. 13). Jede Formel verrechnet dabei das Ergebnis der Vorgängerformel mit zusätzlichen Merkmalen (BRELIX1 bezieht sich auf den BRELIX0, BRELIX2 auf den BRELIX1, usw.). Folgende Merkmale werden von den Formeln verwendet (vgl. (Brügelmann und Brinkmann, 2021, S. 14) für eine Zuordnung der Merkmale zu den Formeln): neben den klassischen Merkmalen wie der Satzlänge, Wortlänge, dem Anteil verschiedener Wörter und dem Anteil der Nebensätze werden auch Merkmale bezüglich der Darstellung des untersuchten Textes, wie die Schriftgröße oder die Textmenge je Seite und die Wortschwierigkeit in den Formeln verwendet. Die Textdarstellungsmerkmale und die Wortschwierigkeit sind besonders hervorzuheben, da sie, soweit dem Autor bekannt, in keiner anderen klassischen Lesbarkeitsformel verwendet werden. Zur Messung der Wortschwierigkeit wird die Anzahl der seltenen Buchstaben (ß, ä, y, q usw.), die Anzahl der mehrgliedrigen Grapheme (st, ie, sch und eu) und die Anzahl der Konsonantenhäufungen am Silbenanfang (str, kl, usw.) gemessen.

Die klassischen Lesbarkeitsformeln haben ihren festen Platz in der Linguistik und Didaktik. Sie bieten ein schnelles Verfahren zur Einschätzung der Lesbarkeit eines Textes, dank der einfach zu berechnenden Formeln. Zur Bewertung der Textkomplexität und zur Zuordnung in eine der Sprachkategorien einer Chatbot-Antwort eignen sich die klassischen Lesbarkeitsformeln aus den folgenden Gründen nicht oder nur eingeschränkt.

Die vergleichsweise kleine Zahl an oberflächlichen Merkmalen, die in den klassischen Lesbarkeitsformeln verwendet werden, reichen nicht für eine präzise Komplexitätseinschätzung aus (Collins-Thompson, 2014). Die folgenden Merkmale sind für das Leseverständnis ebenfalls essenziell, werden aber durch die Formeln nicht berücksichtigt. Dazu gehört die Dichte der Aussagen im Text, Vorkenntnisse, Semantik und Syntax und die Lesemotivation, sowie die Kohäsion des Textes (Collins-Thompson, 2014; Feng, 2010). Allerdings unterscheiden sich leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache, insbesondere bei Betrachtung dieser Merkmale. Einzig der Bremer Erstlese-Index berücksichtigt die Darstellung und Struktur eines Textes neben den traditionellen Merkmalen. Die anderen Lesbarkeitsformeln ignorieren diese Merkmale, obwohl sie eine wichtige Rolle für das Textverständnis spielen (Pearson und Hiebert, 2014).

Die klassischen Lesbarkeitsformeln wurden für vergleichsweise große Texte entwickelt, wie sie für Bücher und Zeitungsartikel typisch sind. Im Gegensatz dazu untersucht die vorliegende Arbeit kurze Texte, wie sie für Chatbot-Antworten verwendet werden. Das führt zu unzuverlässigen Ergebnissen, da viele Formeln auf eine Mindesttextlänge für aussagekräftige Ergebnisse angewiesen sind (Kidwell, Lebanon und Collins-Thompson, 2009).

Des Weiteren gehen klassische Lesbarkeitsformeln von vollständigen und grammatikalisch korrekten Sätzen ohne Sonderzeichen aus. Eine Reihe von Studien (Si und Callan, 2001; Collins-Thompson und Callan, 2004; Feng, Elhadad und Huenerfauth, 2009) kommt zu dem Schluss, dass Texte von Webseiten, die möglicherweise Emojis, Elemente zur Textstrukturierung (Aufzählungslisten mit Bulletpoints usw.) oder Sonderzeichen enthalten, zu unzuverlässigen Ergebnisse der klassischen Lesbarkeitsformeln führen. Für ein möglichst leicht zu erfassende Textbild sehen die Regel der leichten Sprache und einfachen Sprache allerdings genau jene Elemente zur Textstrukturierung, wie Absätze oder formatierte Aufzählungslisten, vor (Maaß, 2019).

Aufgrund dessen und aufgrund der oberflächlichen Parameter sowie vergleichsweise kurzen Länge einer Chatbot-Antwort eigenen sich die klassischen Lesbarkeitsformeln nur eingeschränkt für die Unterscheidung zwischen leichter Sprache, einfacher Sprache, Alltagssprache und Fachsprache. Als zusätzliches Textmerkmal können sie hingegen als Teil einer tiefgehenden Analyse nützlich sein (Deutsch, Jasbi und Shieber, 2020).

3.2 ML-BASIERTE LESBARKEITSEINSCHÄTZUNG

Zu Beginn der Entwicklung der klassischen Lesbarkeitsformeln standen weder die benötigten Werkzeuge für die maschinelle Verarbeitung von natürlicher Sprache zur Verfügung, noch gab es ausreichend leistungsstarke Computer, um diese auf große Textmengen anzuwenden. Inzwischen wurden zahlreiche Werkzeuge im Bereich Natural Language Processing (NLP) entwickelt und auch in der Forschung zum maschinellen Lernen (Machine Learning (ML)) wurden große Fortschritte erzielt. Diese Entwicklungen ermöglichten es, neue Ansätze zur Einschätzung der Lesbarkeit von Texten zu entwickeln. Die daraus entstehenden ML-Modelle erzielen präzisere Ergebnisse, als das die klassischen Lesbarkeitsformeln bisher konnten (Mühlenbock, 2013).

Die klassischen Lesbarkeitsformeln verwenden eine Auswahl der traditionellen Parameter in einer Formel, die in der Regel leicht implementiert oder manuell berechnet werden können. Als Ergebnis erhält man häufig die ungefähre Anzahl der Schuljahre, die ein Leser absolvieren sollte, um den Text lesen und verstehen zu können. Abseits der Schuljahre berechnen manche Formeln auch einen abstrakten Index (bspw. LIX).

Im Gegensatz dazu verwenden die ML-basierten Ansätze zur Einschätzung der Lesbarkeit eine breite Auswahl an Textmerkmalen als Eingabeparameter für die Algorithmen zum Training der Modelle. Dabei scheint es weniger relevant für die Genauigkeit des Ergebnisses, welcher Algorithmus ausgewählt, als welche Auswahl der Textmerkmale verwendet wird (Kate u. a., 2010). Neben den klassischen Textmerkmalen, wie der durchschnittlichen Satzlänge oder der durchschnittlichen Wortlänge, können Dank der umfassenden Möglichkeiten der NLP-Werkzeuge auch zahlreiche tiefgehende Merkmale aus einem Text extrahiert werden. Nicht jede Studie verwendet jedoch alle theoretisch möglichen Merkmale. Im Gegenteil, (Kate u. a., 2010) schreibt, dass die Ermittlung der wirklich relevanten Merkmale und die Beschränkung auf diese einen signifikanten Einfluss auf die Prognosegenauigkeit eines Modells zur Komplexitätseinschätzung hat. In den wissenschaftlichen Veröffentlichungen der letzten Jahre (vgl. Collins-Thompson, 2014; Naderi u. a., 2019; Martinc, Pollak und Robnik-Šikonja, 2021), hat sich die folgende Gruppierung der verwendeten Textmerkmale durchgesetzt:

- traditionelle Merkmale
- lexikalisch-semantische Merkmale
- syntaktische Merkmale (Grammatikalische Strukturen)
- Sprachmodelle
- morphologische Merkmale
- Merkmale der diskursive Kohäsion

Nicht jedes Modell verwendet auch jede Merkmalgruppe. Teilweise konzentrieren sich manche Modelle ganz bewusst nur auf einen Teil der Merkmale. Nachfolgend werden die einzelnen Merkmalgruppen genauer erläutert.

Traditionelle Merkmale

Traditionell wird die Lesbarkeit eines Textes mithilfe der klassischen Lesbarkeitsformeln gemessen. Diese Formeln sind so konstruiert, dass sie einfach zu berechnen sind und deren Ergebnis eine möglichst hohe Korrelation zur durchschnittlichen empfundenen Lesbarkeit eines Textes aufweist (Martinc, Pollak und Robnik-Šikonja, 2021, S. 143). Die traditionellen Merkmale umfassen die klassischen Lesbarkeitsformeln, wie den LIX-Index oder die Wiener Sachtextformeln, aber auch deren zugrunde liegenden Merkmale. Zu diesen, zumeist statischen, Merkmalen gehören zum Beispiel die durchschnittliche Wortlänge, die durchschnittliche Satzlänge oder die durchschnittliche Silbenzahl pro Wort.

Lexikalisch-semantische Merkmale

Als ein wichtiger Faktor, der die Lesbarkeit eines Textes beeinflusst, gilt das verwendete Vokabular. Die lexikalisch-semantischen Merkmale sind Merkmale, die im Zusammenhang mit der Schwierigkeit oder der Bekanntheit der verwendeten Worte und Sätze stehen (Collins-Thompson, 2014). Zu den einfachsten Merkmalen, die die lexikalische Schwierigkeit eines Wortes erfassen, gehört die relative Frequenz der Verwendung eines Wortes in Bezug auf die alltägliche Verwendung. Die relative Frequenz eines Wortes wird initial mithilfe eines großen und repräsentativen Textkorpus bestimmt. Auch werden häufig Wortlisten mit bekannten oder einfachen Worten auf Basis wissenschaftlicher Untersuchungen angelegt und die Worte im untersuchten Text abgeglichen. In dieser Gruppe können sehr viele Merkmale über die Anwesenheit oder Abwesenheit bestimmter Worte oder Wortgruppen erzeugt werden. Ein weiteres Beispiel ist die Type-Token Ratio. Sie beschreibt das Verhältnis von allen eindeutigen Wörter (Wörter, die nur einmal im Text vorkommen) zu allen Wörtern eines Textes. Die Type-Token Ratio ist ein eher generelles Merkmal,s das die lexikalische Reichhaltigkeit eines Textes beschreibt (Collins-Thompson, 2014).

Syntaktische Merkmale

Die syntaktische Komplexität eines Satzes wird laut Friederici mit der Verarbeitungszeit des Gehirns korreliert (Friederici, 2015). Unter der Prämisse, dass eine längere Verarbeitungszeit eine schlechtere Lesbarkeit zur Folge hat, werden Merkmale dieser Gruppe als aussagekräftige Indikatoren betrachtet. Mit der Hilfe leistungsfähiger NLP-Parser werden die Sätze in ihre Bestandteile zerlegt und die Wörter entsprechend ihrer Wortart (bspw. Nomen, Adverb, Präposition usw.) mit sogenannten Part-of-Speech Tag (POS-Tag) klassifiziert. Anschließend ermöglichen aktuelle NLP-Werkzeuge eine tiefgreifende

Analyse des Satzbaus, der strukturellen Beziehung zwischen den Wörtern und der grammatikalischen Strukturen.

Sprachmodelle

Ein statistisches Sprachmodell ist eine weitere Quelle für lexikalische Merkmale. Ein Sprachmodell entspricht dabei einem Histogramm, dass die relative Wahrscheinlichkeit des Vorkommens eines Wortes oder einer Wortsequenz in einem Text zeigt. Mit der Hilfe eines solchen Modells können Muster in der Wortwahl abgebildet werden (Collins-Thompson, 2014). Entsprechend dem zugrundeliegenden Korpus sind die Sprachmodelle kontextspezifisch. Basierend auf den Wahrscheinlichkeiten eines Wortes oder einer Wortsequenz sowie durch ein kontextbewusstes Training des Sprachmodells lassen sich Merkmale zur Textkomplexität ermitteln.

Morphologische Merkmale

Die Merkmale dieser Gruppe beziehen sich auf seltene oder komplexe morphologische Einheiten in den Wörtern des untersuchten Textes. Sie geben Auskunft über die innere Struktur von Wörtern. Es ist anzunehmen, dass die morphologischen Eigenschaften signifikant zur Lesbarkeit eines Textes beitragen. Die Wissenschaftlerin Hancke konnte nachweisen, dass sich die Genauigkeit der Lesbarkeitseinschätzung für die deutsche Sprache mit der Berücksichtigung dieser Merkmale signifikant verbessern lässt (Hancke, Vajjala und Meurers, 2012).

Kohäsive Merkmale

Ein Text ist mehr als die Aneinanderreihung von beliebigen Sätzen. Sprache besitzt übergreifende und übergeordnete Strukturen aufgrund der Beziehungen und Abhängigkeiten ihrer Elemente. Oft bezieht sich die Interpretation eines Elementes eines Textes auf ein anderes Element. Diese Eigenschaft wird als Kohäsion bezeichnet (M. A.K. Halliday, 1976). Ob ein Text als kohäsiv betrachtet werden kann oder ob er nur aus einer Sammlung von Bezug losen Sätzen besteht, ist Gegenstand der Forschung. Diese Unterscheidung kann unter anderem durch die folgenden Merkmale getroffen werden. Durch die Anzahl der Wiederholungen von inhaltlichen Wörtern (bspw. das Subjekt eines Satzes) oder durch die Analyse der Worte, die durch ihre Bedeutung einen expliziten Bezug (bspw. aufgrund dessen, demnach, deshalb usw.) zwischen unterschiedlichen Textteilen herstellen, lassen sich Merkmale ermitteln, mit denen die Kohäsion eines Textes quantifiziert werden kann.

Anders ausgedrückt, werden die semantischen Zusammenhänge, wie der Textzusammenhang, der Textfluss und die Textbezüge mithilfe einer Analyse oberflächlicher Merkmale untersucht. Dafür bietet sich die Suche nach Bindegliedern und anderen kohäsiven Eigenschaften zur Verdeutlichung eines Übergangs oder einer Beziehung im Text an.

Für das Training von Modellen zur Lesbarkeitseinschätzung kommen klassischerweise Algorithmen des überwachten Lernens (Supervised Machine Learning) zum Einsatz. Insbesondere Algorithmen zur Textklassifizierung und zur Modellierung von Regressions- (continuous-valued levels) oder Ranking-Problemen bieten sich an. Folgendes Vorgehen lässt sich abstrakt aus den Veröffentlichungen zur ML-basierten Lesbarkeitseinschätzung ableiten (Balakrishna, 2015, S. 19):

1. *Korpus:* Auswahl eines umfangreichen und hochwertigen Korpus, der Texte einer Schwierigkeitsstufe zuordnet.
2. *Merkmale:* Auswahl von aussagekräftigen linguistischen Merkmalen, von denen auf die Textkomplexität geschlossen werden kann.
3. *Training:* Auswahl eines Algorithmus und ggf. eines Basismodells für das Finetuning. Anschließend muss mithilfe der ausgewählten Merkmale, dem Korpus und dem Algorithmus ein Modell trainiert werden.
4. *Evaluation:* Die Leistungsfähigkeit des Modells wird untersucht. Üblicherweise wird diese über die Messung der Vorhersagegenauigkeit (prediction accuracy) des Modells gemessen.

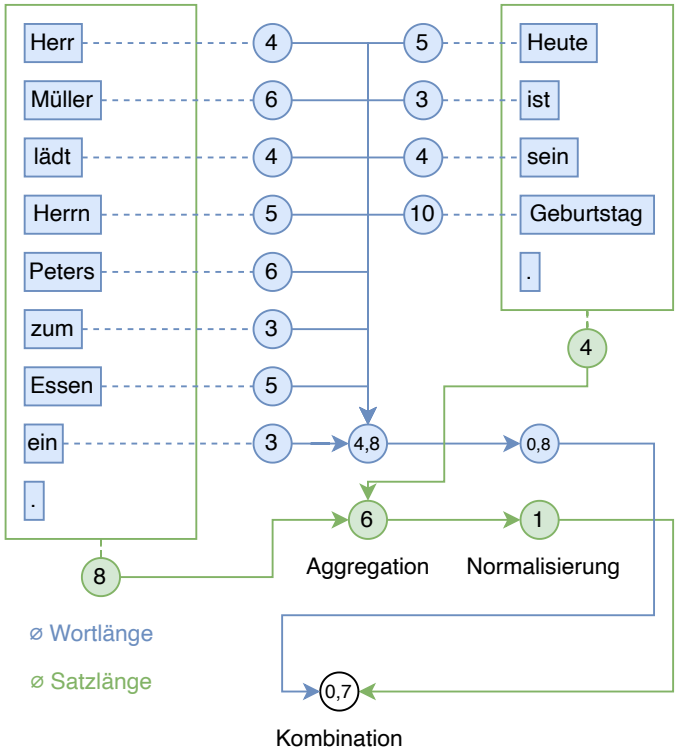

Abbildung 3.1: Beispiel Ablaufdiagramm des „DeLite readability checker“ (vor der Brück und Leveling, 2007) mit den Merkmalen durchschnittliche Wortlänge und durchschnittliche Satzlänge

Zu den ersten ML-basierten Ansätzen zur Textkomplexitätseinschätzung gehört der „DeLite readability checker“ (vor der Brück und Leveling, 2007;

vor der Brück, Hartrumpf und Helbig, 2008). Der DeLite berechnet einen Lesbarkeitsindex in fünf Schritte (vgl. Abbildung 3.1):

1. *Segmentierung:* Mit Hilfe des WOCADI-Parser (WOrd ClAss based DIsambiguating parser) (Hartrumpf, 2003) wird der untersuchte Text in Wörter, Phrasen und Sätze zerlegt und für den nächsten Verarbeitungsschritt aufbereitet.

2. *Berechnung:* Auf Basis der Ergebnisse des WOCADI-Parsers werden anschließend 48 morphologische, lexikalische, syntaktische und semantische Merkmale berechnet. Die Merkmale werden für ihr jeweils zugehöriges Segment (Wörter, Phrasen, Sätze) berechnet.

3. *Aggregation:* Innerhalb der Segmente wird jedes Merkmal gemittelt. Beispielsweise wird die Länge aller Sätze erhoben und die mittlere Satzlänge berechnet.

4. *Normalisierung:* Die gemittelten Merkmale werden nun auf den Wertebereich 0 bis 1 normalisiert.

5. *Kombination:* Im letzten Schritt wird der Lesbarkeitsindex berechnet. Dazu werden alle gemittelten und normalisierten Merkmale entsprechend ihres Einflusses auf die Lesbarkeit gewichtet und anschließend summiert. Die einzelnen Gewichte sind nicht-negativ und ergeben in Summe eins.

Zur Entwicklung des Algorithmus wurden die Parameter der Normalisierungsfunktion und die Gewichtung der Merkmale in der abschließenden Summierung durch den Einsatz von ML-Methoden berechnet (vor der Brück und Leveling, 2007). Die Methode zur Bestimmung der Parameter der Normalisierungsfunktion ist durch eine Analyse bedingter Wahrscheinlichkeiten ermittelt worden. Für die Bestimmung der Parameter zur Gewichtung wurden zwei Ansätze getestet: Eine exakte Methode auf Basis robuster Regression (vor der Brück, Hartrumpf und Helbig, 2008, S. 21) und eine approximative Methode basierend auf linearer Regression (vor der Brück, Hartrumpf und Helbig, 2008, S. 22).

Zum Training und zur Evaluation des „DeLite readability checker“ wurde ein manuell annotierter Textkorpus, bestehend aus 515 Texten aus dem kommunalen Kontext, verwendet. Die Lesbarkeit der Texte ist von 315 Testpersonen auf einer sieben Punkte Likert-Skala (Likert, 1932) bewertet worden. Daraus folgte ein Datensatz aus 2800 individuellen Lesbarkeitsbewertungen. Die Evaluation hat gezeigt, dass im Vergleich zum Flesch-Amstad Index (0.245 Root mean square error (RMSE)) der DeLite eine höhere Prognosegenauigkeit (0.157 - 0.159 Root mean square error (RMSE)) aufweist.

Im Gegensatz zu DeLite verfolgte Hancke (Hancke, Vajjala und Meurers, 2012) das Ziel einer binären Entscheidung zwischen für Kinder und für Erwachsene geeigneten Texten. Dafür kommen ML-Algorithmen nicht nur für die Konstruktion einer Lesbarkeitsformel zum Einsatz, sondern sind integraler Bestandteil der Lesbarkeitseinschätzung selbst. Als Grundlage dienen

eine breite Auswahl an Textmerkmalen. Neben lexikalischen, syntaktischen und morphologischen Merkmalen setzte Hancke auch auf klassische Lesbarkeitsformeln und Sprachmodelle als Merkmale. Diese werden als Eingabeparameter für den Sequential Minimal Optimization (SMO) Algorithmus (Hall u. a., 2009) im Training und auch für die Inferenz mit dem Modell verwendet. Für das Training und die Evaluation des ML-Modells wurde ein Korpus aus 4603 Artikeln aufgebaut. Diese stammen aus GEOlino, einem Magazin für Kinder. Und GEO, einem Magazin für Erwachsene. Das Training des Modells wurde getrennt, sowie auch mit kombinierten Merkmalgruppen, durchgeführt. Das Modell, für dessen Training alle Gruppen mit in Summe 155 Merkmalen verwendet wurden, erzielte die höchste Prognosegenauigkeit von 89,7%.

Weiß und Meurers haben den Ansatz von Hancke fortgeführt (Weiß und Meurers, 2018). Dafür wurde der bereits existierenden GEO/GEOlino-Korpus aktualisiert, auf 8.263 Artikel erweitert und darüber hinaus ein zusätzlicher Korpus angelegt. Der zweite Korpus besteht aus 836 Texten, basierend auf den Untertiteln der Nachrichtensendungen „Tagesschau“der ARD und „Logo!“des ZDF. Das ZDF produziert die Nachrichtensendungen speziell für Kinder. Es wurden 400 Merkmale verteilt über alle Merkmalgruppen extrahiert und jedes Merkmal auf seinen Informationsgewinn in Bezug auf die Klassifizierung untersucht. Je nach Korpus wurden 316 (GEO/GEOlino) bzw. 353 (Tagesschau/Logo) Merkmale als klassifizierungsrelevant identifiziert. Für das Training des binären Klassifizierungsmodells ist, wie bereits bei (Hancke, Vajjala und Meurers, 2012), der Sequential Minimal Optimization (SMO) Algorithmus verwendet worden. Über beide Korpusse hinweg konnte eine Prognosegenauigkeit von 89,4 % und 98,9 % erreicht werden.

Anderer Autoren (Herzberg, 2019; Naderi u. a., 2019) legen den Fokus auf die Suche nach besonders geeigneten ML-Algorithmen. Dafür wurde für beide Veröffentlichungen jeweils ein eigener manuell annotierter Korpus angelegt. Anschließend sind die folgenden etablierten ML-Algorithmen auf ihre Leistungsfähigkeit getestet worden:

- Gaussian Naive Bayes (Manning, Raghavan und Schütze, 2008)
- Support Vector Machine Classifier (Cortes und Vapnik, 1995)
- Decision Tree Classifier (James u. a., 2013)
- Random Forest Classifier (Breiman, 2001)
- Gradient Boosting Classifier (Friedman, 2002)
- Logistische Regression (Allison, 2001)
- Support Vector Regression (Chang und Lin, 2002)
- Random Forest Regression (Breiman, 2001)
- Polynomial Regression

- Linear Regression

Herzberg (Herzberg, 2019), wie auch Naderi (Naderi u. a., 2019) führen ihre Trainings mit einer jeweils unterschiedlichen Anzahlen an Merkmalen durch. Beide Untersuchungen konnten die besten Ergebnisse mit der höchsten Anzahl der verwendeten Merkmale und den Algorithmen der Random Forest Klasse (Breiman, 2001) erreichen. Bei einer schrittweisen Reduktion der Anzahl der Merkmale konnte eine Verringerung der Prognosegenauigkeit algorithmenübergreifend beobachtet werden. Die jeweils besten Ergebnisse erzielten dann unterschiedliche Algorithmen.

Die Forschung im Bereich der ML-basierten Lesbarkeitseinschätzung hat sich in der Vergangenheit auf die Kombination vielfältiger linguistisch motivierter Merkmale, kombiniert mit etablierten und vergleichsweise einfachen ML-Algorithmen konzentriert. Im Kontrast hat Martinc (Martinc, Pollak und Robnik-Šikonja, 2021) gezeigt, dass sich mit dem Einsatz von Deep Neural Networks sehr gute Ergebnisse ohne explizite Extraktion von Merkmalen erzielen lassen. Der Autor untersucht in der Studie überwachte und unüberwachte neurale Ansätze und deren Leistungsfähigkeit bei der Lesbarkeitseinschätzung von Texten.

Die Studie zeigt, dass Deep Neural Networks sich grundsätzlich für die Lesbarkeitseinschätzung mittels überwachter und unüberwachter Trainingsverfahren eigenen. Die Leistungsfähigkeit eines Modells wird über die Wahl eines passenden Trainingsverfahrens zu den Eigenschaften des zugrunde liegenden Korpus bestimmt. In der Kategorie der unüberwachten Trainingsverfahren wurden die folgenden Algorithmen getestet: Long Short-term Memory Network (LSTM) (Kim u. a., 2015), Temporal Convolutional Networks (TCN) (Bai, Kolter und Koltun, 2018) und Transfer Learning auf Basis einen vortrainierten BERT Transformer-Modells (Devlin u. a., 2019). Die ermittelte Prognosegenauigkeit der trainierten Modelle liegt allerdings unter derer, die auf Basis extensiver Merkmalsextraktion in der Vorverarbeitung der Texte trainiert wurden.

Im Gegensatz dazu liefern überwachte Trainingsverfahren, die einzig Texte als Eingabeparameter verwenden, wiederum bessere Ergebnisse als die merkmalsbasierten Verfahren. Die Studie (Martinc, Pollak und Robnik-Šikonja, 2021) hat dafür folgende Trainingsverfahren getestet: Bidirectional Long Short-term Memory Network (BiLSTM) (Conneau u. a., 2017), Hierarchical Attention Networks (HAN) (Nguyen, 2018) und ein Transfer Learning auf Basis eines vortrainierten BERT Transformer-Modells (Devlin u. a., 2019). Beim Training eines Modells auf Basis eines Korpus aus vergleichsweise kurzen Dokumenten, zwischen 200 und 300 Wörter im Vergleich zu 600 und 700 Wörtern je Dokument, konnten mittels Transfer Learning auf Basis des BERT Transformermodells die besten Ergebnisse in der Prognosegenauigkeit erzielt werden.

Deutsch hat in ihrer Arbeit (Deutsch, Jasbi und Shieber, 2020) beide Ansätze kombiniert. Dafür wurde eine Reihe von Deep Neural Networks ohne zu-

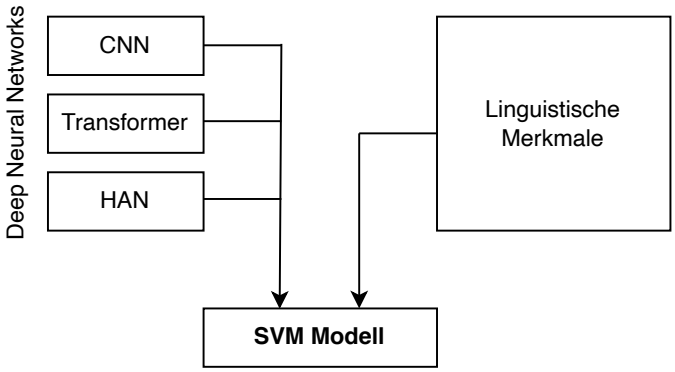

Abbildung 3.2: Verwendung neuraler und linguistischer Merkmale als Eingabeparameter für ein SVM Modell (Deutsch, Jasbi und Shieber, 2020)

vor extrahierte Merkmale eingesetzt und deren Klassifikationsergebnisse als Merkmale interpretiert. Diese Merkmale wurden neben einer Auswahl an linguistischen Merkmalen wiederum als Eingabeparameter für ein Support Vector Machine (SVM) Modell (Cortes und Vapnik, 1995) verwendet (vgl. Abbildung 3.2). In der Evaluation konnten keine signifikanten Verbesserungen der Prognosegenauigkeit, verglichen mit Deep Neural Networks ohne Bezugnahme von linguistischen Merkmalen (Martinc, Pollak und Robnik-Šikonja, 2021), festgestellt werden. Die folgenden Deep Neural Networks wurden als neurale Merkmale trainiert und eingesetzt:

- Convolutional Neural Network (CNN) (Kim, 2014)
- Transformermodell von Martinic (Martinc, Pollak und Robnik-Šikonja, 2021)
- Hierarchical Attention Network (HAN) (Nguyen, 2018)

Die Trainings und die Evaluation wurden mithilfe zweier Korpusse (Vajjala und Meurers, 2012; Xia, Kochmar und Briscoe, 2016) durchgeführt.

4

METHODE

Im Rahmen der Forschungsfrage dieser Arbeit soll untersucht werden, ob die Lesbarkeit eines Textes mithilfe von Transformer-Modellen als leichte Sprache, einfache Sprache, Alltagssprache oder Fachsprache klassifiziert werden kann. Zur Beantwortung dieser Frage wird ein Fine-Tuning auf mehrere Transformer-Modelle durchgeführt. Dies wird in Kapitel 4.2 beschrieben.

Im Vergleich zu den traditionellen Ansätzen, wie in Kapitel 2 beschrieben, ist für das Fine-Tuning eines Transformer-Modells für Klassifizierungsaufgaben kein extensives Feature-Engineering notwendig. Anstelle dessen wird ein umfassendes Korpus, der Texte aller vier Sprachkategorien enthält, benötigt. Der Aufbau eines solchen Korpus wird in Kapitel 4.1 beschrieben.

Um die der Leistungsfähigkeit des transformersbasierten Ansatzes mit den traditionellen, merkmalbasierten Ansätzen vergleichen zu können, wird exemplarisch ein SVM-Modell trainiert. Das Training wird im Kapitel 4.3 beschrieben.

4.1 AUFBAU EINES KORPUS

Eine umfassende und hochwertige Datenbasis ist die Grundlage für jedes Training eines Deep-Learning Modells. Für die Untersuchung der Forschungsfrage dieser Thesis ist es notwendig, ML-Modelle zu trainieren bzw. ein Fine-Tuning (vgl. 4.2) auf bestehende Modelle anzuwenden. Dafür ist eine umfassende Sammlung von Texten in den vier Sprachkategorien leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache notwendig. Im Folgenden wird die Textsammlung als Korpus bezeichnet.

Soweit vorhanden, würde sich die Verwendung von bestehenden Korpora anbieten. Für die Kategorien leichte und einfache Sprache gibt es bereits mehrere Korpora, die sich allerdings nicht oder nur eingeschränkt für einen Trainingsprozess eignen. Das Korpus *LeiKo* (Jablotschkin und Zinsmeister, 2020) besteht aus einer zu kleinen Anzahl von 216 Texten aus frei verfügbaren Quellen. Die Texte des Korpus *KED* (Jach, 2020) stammen aus Quellen, die teilweise nicht eindeutig einer Sprachkategorie zugeordnet werden können. Auf andere Korpora, wie dem umfangreichen Korpus *Geasy* (Hansen-Schirra, Nitzke und Gutermuth, 2021), dem Korpus von Bock (Bock, 2019), oder dem Korpus von Stephan (Stephan, 2014) konnte aufgrund von urheberrechtlichen Problemen nicht zugegriffen werden. Soweit dem Autor bekannt, gibt es bisher keinen umfassenden Korpus, der hochwertige Texte in

ausreichender Anzahl und in allen vier Sprachkategorien vereint. Aufgrund dessen ist es notwendig, einen eigenen Korpus anzulegen.

Allerdings können dafür die Texte bestehender Korpora verwendet werden. So kommen beispielsweise die Texte des *LHA* Korpus (Spring, Rios und Ebling, 2021) und Auszüge des *KED* Korpus, die sich eindeutig zuordnen lassen, zum Einsatz. Die Quellen der Texte des Korpus werden im Kapitel 4.1.2 aufgeführt.

Für die beiden Sprachkategorien Alltagssprache und Fachsprache existieren ebenfalls bereits Korpora. Allerdings beziehen sich diese auf ausschließlich eine Themenkategorie: News Artikel. Aufgrund dessen muss das Korpus um weitere Texte anderer Kategorien ergänzt werden, um einem Selection Bias (Shah, Schwartz und Hovy, 2020) vorzubeugen.

Weil die bestehenden Korpora eine zu geringe Anzahl an Texten enthalten oder diese nur einer Themenkategorie zugeordnet werden können, muss der aufzubauenden Korpus um weitere Texte aus unterschiedlichen Quellen ergänzt werden. Dafür wurden Texte von verschiedenen, öffentlich zugänglichen Webseiten kopiert. Zur Extraktion der Texten aus den Quellen wurden eigene Crawler-Skripte entwickelt. Dies wird im Abschnitt 4.1.1 genauer erläutert. Der Abschnitt 4.1.3 geht auf die Vorverarbeitung der Texte aus den Quellen und der Korpora ein. Anschließend legt der Abschnitt 4.1.2 die Auswahl der Quellen dar.

4.1.1 *Crawler-Entwicklung*

Für den Aufbau des Korpus werden sehr viele Texte benötigt. Es bieten sich digitale Quellen wie PDF-Dateien oder Webseiten zur Gewinnung der Texte an, da sie ohne zusätzlichen Aufwand bereits maschinell verarbeitet werden können. Texte lassen sich vergleichsweise einfach aus Webseiten extrahieren. Aufgrund dessen werden sie neben den bestehenden Korpora als Quellen verwendet. Zum Extrahieren von Texten aus den Webseiten, auch als Web-Scraping oder Crawling bezeichnet, stehen viele unterschiedliche Frameworks und Anwendungen zur Verfügung. Da jede Webseite ihre Inhalte im Quellcode individuell strukturiert, ist ein automatisierter Export von bestimmten Textteilen, wie beispielsweise einem Newsartikel, eine nicht triviale Aufgabe. Wiederkehrende relevante Textelemente müssen bei, dem Projektumfang angemessenen einfachen, Crawlern manuell im Quellcode der Webseite ausgewählt werden. Es gibt viele verschiedene Frameworks[1] und Anwendungen,[2] [3] [4] [5] die dafür die passenden Werkzeuge liefern und die Möglichkeit bieten, aus den einmal selektierten Elementen des Quellcodes effizient Texte wiederkehrend zu extrahieren. In diesem Kontext stel-

1 https://github.com/BruceDone/awesome-crawler
2 https://www.octoparse.com/
3 https://apify.com/
4 https://www.parsehub.com/pricing
5 https://nutch.apache.org/

len Frameworks Werkzeuge zur softwaretechnischen Entwicklung von Crawlern auf eigener Infrastruktur zur Verfügung. Anwendungen hingegen bieten ebenfalls Werkzeuge an, die es auf abstrahierter Ebene, beispielsweise grafisch, ermöglichen, Crawler zu entwickeln und diese gegebenenfalls auf fremder Infrastruktur auszuführen.

Die dem Autor bekannten Anwendungen sind für den Projektumfang ungeeignet, da sie wie Apify[6] und Parsehub[7] teuer, wie Nutch[8] komplex in der Anwendung sind oder wie Octoparse[9] nicht zuverlässig funktionieren. Die Frameworks zur Crawler-Entwicklung[10] bieten dagegen eine gute und einfache Alternative. Einige der Frameworks, wie Scrapy[11] oder Cheerio[12] werden seit Jahren von viele Firmen produktiv eingesetzt und funktionieren stabil. Viele Crawler-Frameworks werden zudem quelloffen entwickelt und sind sehr gut dokumentiert.

Die Programmiersprache JavaScript ist fest in die Entwicklung von Webseiten integriert. Seit einigen Jahren kann sie auch serverseitig eingesetzt werden. Aufgrund dessen bietet es sich an, für die Crawler-Entwicklung die etablieren JavaScript-basierten Frameworks Puppeteer (vgl. Kapitel 2.2.1) und Cheerio (vgl. Kapitel 2.2.2) serverseitig einzusetzen. Sie sind in der Entwicklergemeinschaft weit verbreitet, gut dokumentiert und bieten einen einfachen Zugriff auf die notwendigen Funktionalitäten.

Die Webseiten, die als Quellen ausgewählt wurden und deren Texte nicht über einen bereits bestehenden Korpus zur Verfügung stehen, weisen alle eine ähnliche Struktur auf. Auf einer Übersichtsseite werden Links zu den Detailseiten aufgeführt. Die Detailseiten beinhalten den eigentlichen Inhalt. Am Beispiel einer Nachrichtenseite, wie nachrichtenleicht.de, werden auf der Übersichtsseite die Newsartikel mit einem Kurztext, einem Bild und dem Link zum eigentlichen Artikel, der Detailseite, aufgeführt.

Ziel des Crawlers ist es, von der Übersichtsseite die Links zu den Detailseiten zu extrahieren, in einer Liste zu speichern und anschließend die Detailseiten aufzurufen und deren Texte zu extrahieren. Da die Webseiten der Quellen jeweils einen individuell strukturierten Quellcode aufweisen, wird für jede Webseite ein eigenes Crawler-Skript implementiert.

Die Crawler rufen die Übersichtsseite mittels Puppeteer, wie in Kapitel 2.2.1 beschrieben, auf und der Quellcode der Webseite wird an Cheerio (vgl. 2.2.2) übergeben. Im Hintergrund verwendet Puppeteer für den Aufruf der Webseite den Chromium Browser. Aufgrund dessen wird der Quellcode der Webseite vollständig interpretiert und auch dynamische Änderungen am Quellcode durch den Einsatz von JavaScript berücksichtigt. Cheerio parst den

6 https://apify.com/
7 https://www.parsehub.com/pricing
8 https://nutch.apache.org/
9 https://www.octoparse.com/
10 https://github.com/BruceDone/awesome-crawler
11 https://scrapy.org/
12 https://cheerio.js.org/

HTML-Code der Webseite anschließend und bietet Schnittstellen für den Zugriff auf den virtuellen DOM-Baum der Webseite.

Für die Entwicklung eines Crawler-Skripts müssen die HTML-Elemente der Links zu den Detailseiten und deren Texte im DOM-Baum manuell identifiziert werden. Dabei helfen die Webentwickler-Werkzeuge der gängigen Webbrowser, wie Firefox oder Google Chrome. Die HTML-Elemente können aufgrund ihrer Position im DOM-Baum und ihrer Attribute mithilfe von CSS-Selektoren adressiert werden.

Das Framework Cheerio bietet Zugriff auf die Knoten des DOM-Baumes mittels der Syntax ähnlich der CSS-Selektoren. Im DOM-Baum der Übersichtsseite wird mit den entsprechenden CSS-Selektoren nach den Links zu den Detailseiten gesucht. Aus den Knoten werden die URLs der Detailseiten extrahiert und in einem Array abgespeichert.

Alle Links zu den Detailseiten werden anschließend ebenfalls mittels Puppeteer aufgerufen. Anschließend wird mit dem gleichen Verfahren nach den HTML-Elementen, die den gewünschten Text der Detailseite enthalten, gesucht. Die Cheerio Methode .text() bietet die Möglichkeit, den Text eines Knotens des DOM-Baumes als String zu extrahieren. Dabei ist es irrelevant, ob ein Knoten weiter HTML-Knoten oder ausschließlich Text-Knoten enthält. Wenn ein Knoten allerdings weitere HTML-Knoten enthält, dann wird deren Funktion ignoriert und nur der Text dieser Knoten extrahiert. Das führt dazu, dass die Anwendung der .text() Methode auf Aufzählungslisten (HTML-Element <ul> oder <ol>) möglicherweise unvollständige Sätze zurückgibt, da die Struktur der einzelnen Aufzählungspunkte (HTML-Element <li>) nicht beachtet wird und verloren geht. Anderenfalls müssten für Listen eigene Selektoren bestimmt werden. Zudem verwenden die meisten Quellen Aufzählungslisten innerhalb ihrer Texte nicht für grammatikalisch vollständiger Sätze. Da im Rahmen der Arbeit mit angemessenem Aufwand keine Ausnahmebehandlung umgesetzt werden konnte, wurden ausschließlich Texte aus den HTML-Elementen für Fließtexte wie <p>, <article> usw. extrahiert. Damit konnte in der Praxis der Aufwand in der Textnachbearbeitung erheblich reduziert werden.

Texte in leichter oder einfacher Sprache zeichnen sich insbesondere durch ihren Satzbau aus. Die durchschnittliche Satzlänge und die Verwendung von Nebensätzen sind starke Indikatoren der beiden Sprachkategorien (Maaß, 2015; Bock, 2019; Maaß, 2019; Spring, Gonzales und Ebling, 2021). Aufgrund dessen ist es von besonderer Bedeutung für den Aufbau des Korpus, nur vollständige Sätze zu verwenden, um den Trainingsprozess eines ML-Modells nicht nachteilig zu beeinflussen.

Im letzten Schritt werden die extrahierten Texte in einer Textdatei je Unterseite gespeichert. Absätze im Text werden auf den Webseiten der Quellen in der Regel in eigenen HTML-Elemente und auf gleicher Ebene im DOM-Baum strukturiert. Meistens weisen sie auch die gleichen Attribute auf und können so mit einem gemeinsamen CSS-Selektor identifiziert werden. Wenn

ein CSS-Selektor zu mehreren HTML-Elementen passt, dann gibt Cheerio ein Array von Texten zurück. So bleibt die Absatzstruktur der Texte für den nächsten Schritt, der Textaufbereitung, erhalten.

Im Vergleich zu Webseiten lassen sich Texte aus PDF-Dateien nur mit erheblichem Aufwand strukturiert extrahieren. Bekannte Frameworks, wie pdf-to-text[13], OCRmyPDF[14], pdf2json[15] oder PyPDF2[16], erlauben zwar die Extraktion von Texten aus PDF-Dateien, Informationen zur Textstruktur gehen allerdings verloren. Das führt dazu, dass Seitenzahlen, Titel und zusätzliche Texte neben dem Haupttext nur gemeinsam extrahiert werden können. Eine Einschränkung auf die relevanten Textteile ist nur mit erheblichem Aufwand zu realisieren und war nicht ihm Rahmen dieser Arbeit möglich. Zudem extrahieren die genannten Frameworks die Texte nicht als Fließtext, sondern mit den Textumbrüchen aus der PDF-Datei. Das Entfernen der Textumbrüche in der Nachbearbeitung ist mit erheblichem Aufwand verbunden und war im Rahmen der Arbeit ebenfalls nicht möglich. Aufgrund dessen wurde auf PDF-Dateien als Textquellen zum Aufbau des Textkorpus verzichtet.

4.1.2 *Quellenauswahl*

Als Vorarbeit zum Aufbau des Korpus wurde eine Liste von Textquellen in den vier Sprachkategorien leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache zusammengestellt. Die Quellen setzen sich aus diversen Webseiten und bereits bestehenden Korpora zusammen und sind frei über das Internet verfügbar, unterliegen aber dem Urheberrecht der jeweiligen Autoren. Nachfolgend werden die Quellen und ihr Anteil am Korpus in der entsprechenden Sprachkategorie aufgeführt.

Leichte Sprache

Die folgenden Quellen wurden für den Aufbau des Textkorpus in der Sprachkategorie leichte Sprache verwendet. Die Abbildung 4.1 zeigt den Anteil der einzelnen Quellen an der Gesamtsumme aller Texte des Korpus in leichter Sprache.

- *Arbeit & Gesundheit – Das Portal für Sicherheitsbeauftragte*[17]
  Das Portal der gesetzlichen Unfallversicherung richtet sich an die Sicherheitsbeauftragten von Unternehmen und stellt Informationen zur Arbeitssicherheit in leichter Sprache zur Verfügung.
- *EINFACHSTARS | Aktuelles über Stars in leichter Sprache*[18]
  Der Boulevardblog EINFACHSTARS publiziert Artikel in leichter Sprache über Stars, Mode und ähnliche Themen.

13 https://github.com/spatie/pdf-to-text
14 https://github.com/ocrmypdf/OCRmyPDF
15 https://github.com/modesty/pdf2json
16 https://github.com/py-pdf/PyPDF2
17 https://aug.dguv.de/leichte-sprache/
18 https://einfachstars.info/

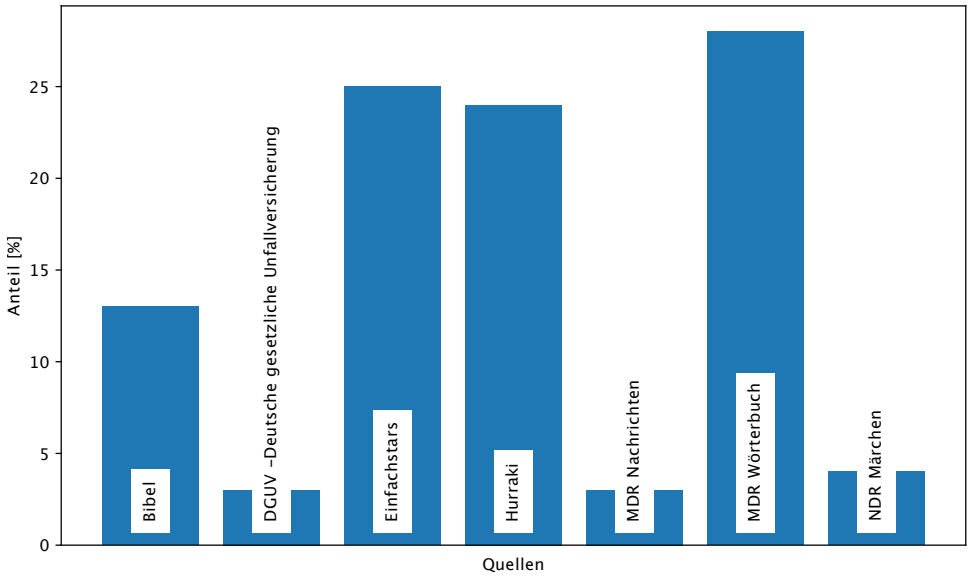

Abbildung 4.1: Anteile der Quellen am Korpus in leichter Sprache

- *Hurraki*[19]
  Das Wörterbuch Hurraki ist ein öffentliches Projekt auf Basis eines Wikis in leichter Sprache. Da die Mitarbeit an dem Wörterbuch grundsätzlich jedem offen steht und keine durchgängige Qualitätskontrolle durch Fachpersonal erfolgt, ist eine fortwährend hohe Textqualität nicht garantiert.
- *Nachrichten in leichter Sprache*[20]
  Das Nachrichtenportal des Mitteldeutschen Rundfunks veröffentlicht auch Artikel in leichter Sprache zum Weltgeschehen und dazu ein korrespondierendes Wörterbuch ebenfalls in leichter Sprache.
- *Märchen in leichter Sprache*[21]
  Der Norddeutsche Rundfunk hat eine Sammlung an Märchen in leichte Sprache übersetzt und veröffentlicht.
- *Evangelium in leichter Sprache*[22]
  Einige Teile der Bibel wurden von der Katholischen Kirche in leichte Sprache übersetzt und ausschnittweise veröffentlicht.

Einfache Sprache

Die folgenden Quellen wurden für den Aufbau des Textkorpus in der Sprachkategorie einfache Sprache verwendet. Die Abbildung 4.2 zeigt den Anteil

19 https://hurraki.de/wiki/Hauptseite/
20 https://www.mdr.de/nachrichten-leicht/rueckblick/leichte-sprache-rueckblick-verteilseite-100.html
21 https://www.ndr.de/suche10.html?query=m%C3%A4rchen+leichte+sprache
22 https://www.evangelium-in-leichter-sprache.de/

der einzelnen Quellen an der Gesamtsumme aller Texte des Korpus in einfacher Sprache.

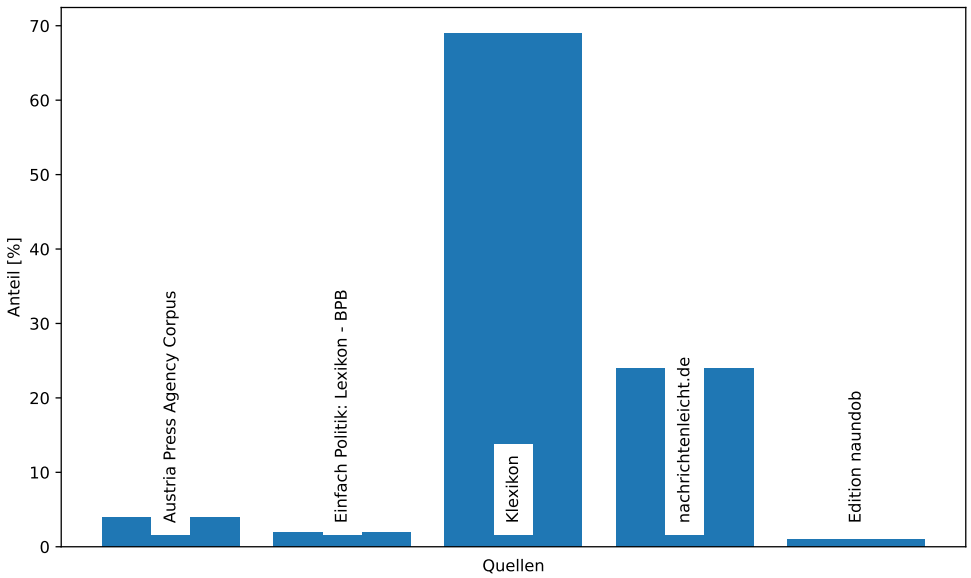

Abbildung 4.2: Anteile der Quellen am Korpus in einfacher Sprache

- *Nachrichtenleicht*[23]
  Das Nachrichtenportal des Deutschlandfunk veröffentlicht regelmäßig Artikel zum aktuellen Weltgeschehen auch in einfacher Sprache.
- *Einfach Politik: Lexikon*[24]
  Das Lexikon der Bundeszentrale für politische Bildung enthält Erklärungen aus dem politischen Themenfeld in einfacher Sprache.
- *Klexikon*[25]
  Allgemeines Kinderlexikon mit Erklärungen in kindgerechter Sprache. Die Texte des Lexikons werden über den bestehenden Korpus (Aumiller und Gertz, 2022) bezogen.
- *Leseproben für Literatur in einfacher Sprache*[26]
  Der Edition naundob Verlag hat sich auf Literatur in einfacher Sprache spezialisiert und stellt Leseproben seiner Bücher zur Verfügung.
- *Austria Press Agency Corpus*
  Ein bestehender Korpus (Spring, Rios und Ebling, 2021) auf Basis von Newsartikel in einfacher Sprache, veröffentlicht von der österreichischen Presseagentur. Die Texte wurden nach den Stufen des gemeinsamen europäischen Referenzrahmens für Sprachen gruppiert. Für die Verwendung in dem aufzubauenden Textkorpus wurde ausschließlich auf die Texte der Stufe A2 zurückgegriffen.

23 https://www.nachrichtenleicht.de/
24 https://www.bpb.de/kurz-knapp/lexika/lexikon-in-einfacher-sprache/
25 https://klexikon.zum.de/wiki/Klexikon/
26 https://www.naundob.de/b%C3%BCcher-1/

Alltagssprache

Die folgenden Quellen wurden für den Aufbau des Textkorpus in der Sprachkategorie Alltagssprache verwendet. Die Abbildung 4.3 zeigt den Anteil der einzelnen Quellen an der Gesamtsumme aller Texte des Korpus in Alltagssprache.

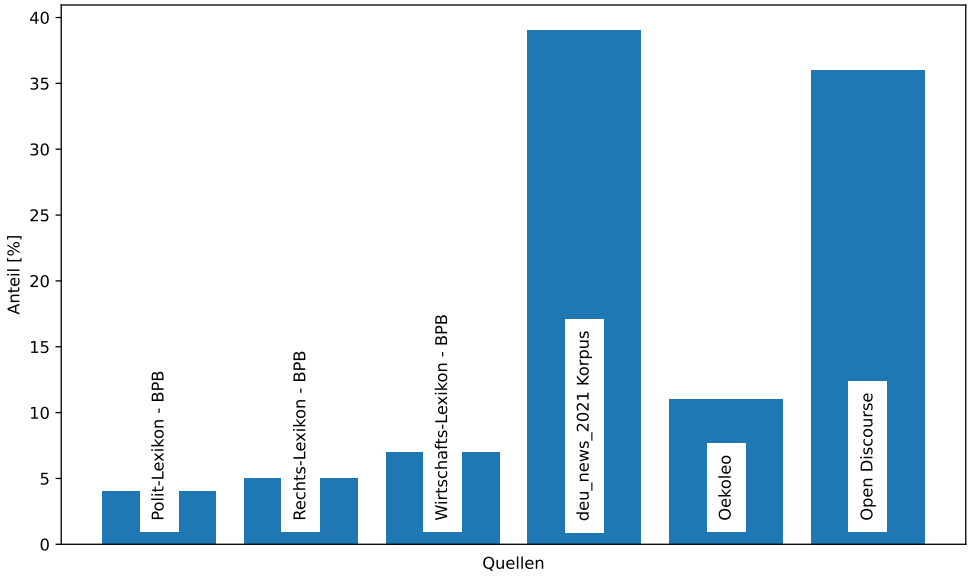

Abbildung 4.3: Anteile der Quellen am Korpus in Alltagssprache

- *Rechtslexikon*[27]
  Das „Rechtslexikon“ ist ein Angebot der Bundeszentrale für politische Bildung. Es stellt Erklärungen in Alltagssprache zu Themen mit Rechtsbezug zur Verfügung.
- *Politlexikon*[28]
  Das „Politlexikon“ ist ein Angebot der Bundeszentrale für politische Bildung. Es enthält Erklärungen in Alltagssprache zu Themen mit politischem Bezug.
- *Lexikon der Wirtschaft*[29]
  Das „Lexikon der Wirtschaft“ ist ein Angebot der Bundeszentrale für politische Bildung. Es enthält Erklärungen in Alltagssprache mit Bezug zu Wirtschaftsthemen.
- *Wortschatz Leipzig*
  Die Universität Leipzig hat mit dem Projekt „Leipzig Corpora Collection“ (Goldhahn, Eckart und Quasthoff, 2018) eine umfangreiche Sammlung an Textkorpora aufgebaut. Für den beschriebenen Textkorpus wird aus der Sammlung das Korpus „deu_news_2021_1M“ ver-

27 https://www.bpb.de/kurz-knapp/lexika/recht-a-z/
28 https://www.bpb.de/kurz-knapp/lexika/politiklexikon/
29 https://www.bpb.de/kurz-knapp/lexika/lexikon-der-wirtschaft/

wendet. Er enthält Newsartikel in Alltagssprache von verschiedenen deutschsprachigen Zeitungen und deren Onlineangeboten.

- *Open Discourse*
  Das Projekt „Open Discourse“ (Richter u. a., 2021) hat einen Korpus aus den Plenarsaal-Protokollen des Deutschen Bundestags generiert. Er enthält alle protokollierten Reden seit 1949. Da auch die verwendete Schriftsprache fortlaufender Veränderung unterliegt, ist es wichtig, aktuelle Texte zu verwenden. Um ein möglichst aktuelles Sprachbild zu erhalten, wurden für den aufzubauenden Textkorpus nur Reden ab dem Jahr 2012 verwendet.

Fachsprache

Die folgenden Quellen wurden für den Aufbau des Textkorpus in der Sprachkategorie Fachsprache verwendet. Die Abbildung 4.4 zeigt den Anteil der einzelnen Quellen an der Gesamtsumme aller Texte des Korpus in Fachsprache.

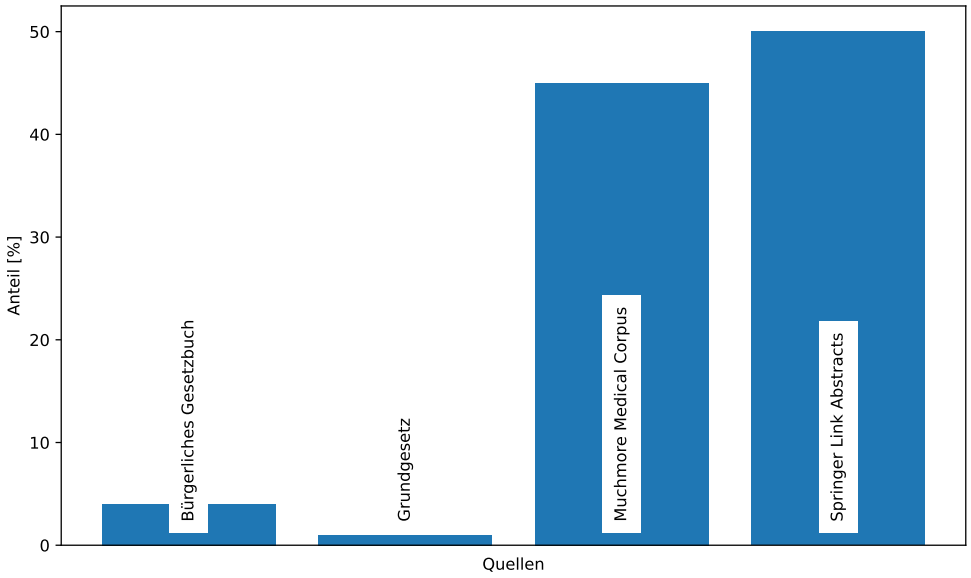

Abbildung 4.4: Anteile der Quellen am Korpus in Fachsprache

- *Deutsches Grundgesetz*[30]
  Das Grundgesetz der Bundesrepublik Deutschland wurde im Volltext verwendet.
- *Bürgerliches Gesetzbuch*[31]
  Das Bürgerliche Gesetzbuch der Bundesrepublik Deutschland wurde ebenfalls im Volltext verwendet.

30 http://www.gesetze-im-internet.de/gg/BJNR000010949.html
31 http://www.gesetze-im-internet.de/bgb/BJNR001950896.html

- *MuchMore Korpus*[32]
  Der parallele „MuchMore“ Korpus des Deutschen Forschungszentrum für Künstliche Intelligenz beinhaltet deutsche und englische Zusammenfassungen (Abstracts) wissenschaftlicher Veröffentlichungen aus dem medizinischen Bereich. Die Zusammenfassungen des Korpusses wurden von der Plattform Springer Link bezogen.
- *Zusammenfassungen von wissenschaftlichen Veröffentlichungen*[33]
  Über eine Schnittstelle der Plattform Springer Link können Zusammenfassungen (Abstracts) von wissenschaftlichen Veröffentlichungen bezogen werden. Für den aufzubauenden Textkorpus wurden aus diversen Fachbereichen Zusammenfassungen verwendet.

4.1.3 *Textaufbereitung*

Für das Training eines ML-Modells sind qualitativ hochwertige Daten essenziell. Für den Aufbau eines Korpus werden Texte der vier Sprachkategorien leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache aus verschiedenen Quellen (vgl. Abschnitt 4.1.2) gewonnen. Neben Webseiten werden auch Texte aus bereits bestehenden Korpora übernommen, sowie frei verfügbaren APIs verwendet, um auf Quellen zuzugreifen.

Die Zusammenstellung des Korpus aus den extrahierten Texten geschieht in zwei Schritten. Im ersten Schritt werden die originalen Texte bereinigt und satzweise segmentiert. Im zweiten Schritt werden anschließend aus der Menge der Sätze neue Texte zusammengesetzt und diese in den Korpus übernommen.

Die unverarbeiteten Texte der Quellen stehen als einfache Textdateien zur Verfügung. Zur Verarbeitung der hohen Anzahl von mehr als 100.000 Dateien wurde eine Multiprozess-Parallelverarbeitung nach dem Producer-Consumer Modell in Python implementiert. Der Producer-Prozess ermittelt die Pfade aller zu verarbeitenden Textdateien und stellt sie in einer FIFO-Queue den Consumer-Prozessen zur Verfügung. Anschließend wartet er auf die Ergebnisse der Consumer-Prozesse. Diese Prozesse konsumieren die Elemente der FIFO-Queue solange, bis keine neuen Elemente nachkommen. Jedes Element entspricht dabei einem Dateipfad und respektive einer zu verarbeitenden Datei. Die Consumer-Prozesse öffnen die entsprechenden Dateien selbst, verarbeiten den Text und geben diesen bereinigt und in einzelne Sätze segmentiert anschließend an den ursprünglichen Producer-Prozess zurück. Dieser speichert die Sätze zusammen mit ihrer Kategorie, ihrer Quelle und einem SHA-256 Hash des Satzes in einer SQLite Datenbank. Um doppelte Sätze auszuschließen, wurde die Spalte der Hashes in der Datenbank als „Unique“ deklariert.

32 https://muchmore.dfki.de/resources1.htm
33 https://dev.springernature.com/

Zur Segmentierung der Texte in einzelne Sätze wird das Framework Spacy (vgl. Kapitel 2.2.3) verwendet. Um eine hohe Genauigkeit in der Satzerkennung zu erreichen, ist das Framework auf bereinigte Texte angewiesen. Überschriften, Aufzählungen ohne korrekte Satzstruktur und Steuerzeichen werden mithilfe selbst entwickelter Filter aus den Texten entfernt bevor diese an das Framework übergeben werden. Bei der Satzerkennung mit Spacy konnte die höchste Genauigkeit auf Basis des transformerbasierten deutschen Sprachmodells[34] erzielt werden (f1 Score von 0.98). Eine Versuchsreihe zum Test anderer Sprachmodelle und Frameworks wie Stanza[35] erzielte schlechtere Ergebnisse in der Satzsegmentierung. Anschließend werden Sätze, die nur zwei Worte enthalten, entfernt, da sich im Testbetrieb gezeigt hat, dass sie in der Regel das Produkt einer fehlerhaften Erkennung waren. Zwei Quellen in der Kategorie Fachsprache, die Zusammenfassungen von wissenschaftlichen Veröffentlichungen, enthalten teilweise englische Texte. Diese wurden mithilfe einer Spacy-Erweiterung[36] erkannt und ebenfalls entfernt.

Im zweiten Schritt des Aufbaus des Korpus werden die Sätze aus der SQLite Datenbank zufällig zu neuen Texten zusammen gesetzt und in den Korpus übernommen. Damit sollen zwei Ziele erreicht werden: 1. Reduktion eines Bias aufgrund der Textthemen der Quellen und 2. die Steuerung der Länge der einzelnen Texte.

Durch das zufällige Zusammenstellen der Sätze zu neuen Texten soll verhindert werden, dass das Korpus irrelevante Muster in Bezug auf die Themen der zugrundeliegenden Texte abbildet. Der Fokus des Korpus soll auf der Struktur der Sätze liegen. Insbesondere bei vergleichsweise kleinen Datensätzen, wie dem vorliegenden, besteht zudem die Gefahr des Overfitting (Shah, Schwartz und Hovy, 2020). Zudem können die verfügbaren Quellen nicht als repräsentativ in ihrer Themenauswahl betrachtet werden. Beispielsweise soll so verhindert werden, dass ein Text als „einfach“ klassifiziert wird, weil er berühmte Persönlichkeiten thematisch behandelt (vgl. die Quelle EinfachStars Abschnitt 4.1.2). Bei der Auswahl der Quellen lag der primäre Fokus auf der Verfügbarkeit der Texte und nachrangig auf der Themenvielfalt innerhalb einer Sprachkategorie.

Für möglichst authentische Daten und somit einem hochqualitativen Korpus ist auch die Länge eines einzelnen Textes wichtig. Jeder Text repräsentiert eine Chatbot-Antwort und sollte dementsprechend auch eine typische Länge aufweisen. Somit kann die Anzahl der Sätze je Text beispielsweise ein Unterscheidungskriterium der Sprachkategorien darstellen. Neben der Authentizität der Daten tritt ein weiteres Problem auf. Auf Basis des Korpus wird im Rahmen dieser Arbeit ein Fine-Tuning von Transformer-Modellen betrieben. Die verwendeten Transformer-Modelle sehen eine Maximallänge von 512 Token eines Eingabetextes vor. Eine Anpassung der ursprünglichen Textlänge wäre somit in jedem Fall notwendig geworden.

34 https://spacy.io/models/dede_dep_news_trf
35 https://stanfordnlp.github.io/stanza/
36 https://pypi.org/project/spacy-language-detection/

Die typische Länge einer Chatbot-Antwort lässt sich nicht allgemeingültig bestimmen. Verschiedene Faktoren wie die Sprache und das Thema beeinflussen die Länge. Während Chatnachricht zwischen Menschen zu kurzen Längen zwischen durchschnittlich 5 und 6 Wörtern tendieren (Rosenfeld u. a., 2018), sind Chatbot-Anworten durchaus länger.

Zwei wissenschaftliche Veröffentlichungen (Jahanshahi, Kazmi und Cevik, 2022; Cañizares u. a., 2022) ermitteln eine durchschnittliche Antwortlänge zwischen 10 und 14 Wörtern. In den Extremen treten auch Längen bis zu 100 Wörtern respektive 54 Wörter auf. Das Ergebnis beider Veröffentlichung ist allerdings nur eingeschränkt aussagefähig, da sie als Datengrundlage entweder ausschließlich medizinische Antworten eines Chatbots verwenden oder nur eine geringe Anzahl an Antworten analysieren.

Aufgrund der ungenauen Datenlage wurden 2.131 Antworten verschiedener und themenübergreifender Chatbots der Firma MakeIT untersucht. Demnach sind ihre Antworten im Durchschnitt 28 Worte lang. Für den Aufbau des Korpus wird diese Länge als weiche Maximallänge und nicht als Mittelwert übernommen (vgl. Abbildung 4.5). Das Korpus repräsentiert somit mehr kürzere Nachrichtenlängen, um den voran gegangenen Untersuchungen (Rosenfeld u. a., 2018; Jahanshahi, Kazmi und Cevik, 2022; Cañizares u. a., 2022) zumindest Anteilig Rechnung zu tragen. Das Ergebnis der Analyse der Chatbots wird als aussagekräftiger bewertet, weil im Vergleich zu den wissenschaftlichen Veröffentlichungen mehrere und themenübergreifende Chatbots verwendet wurden.

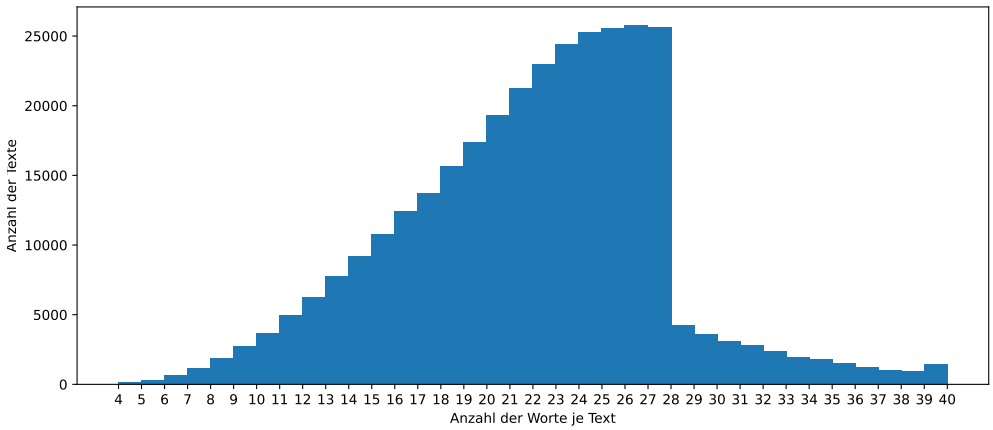

Abbildung 4.5: Verteilung der Textlänge des Korpus

Die Abbildung 4.5 zeigt die Verteilung der Textlängen des Korpus. Die Sätze einer Sprachkategorie werden bei der Generierung der Texte so lange aneinander gehängt, bis die Anzahl der Worte addiert mit der Anzahl der Worte des nächsten Satzes die maximale Länge von 28 Wörtern überschreiten würde. In diesem Fall wird der Satz nicht mehr zum aktuellen Text hinzugefügt. Der Text wird in den Korpus übernommen und der aktuelle Satz bildet den Start des nächsten Textes. Sobald die Länge des nächsten Satzes die maximale Länge überschreiten würde, gilt der aktuelle Text als abgeschlossen, auch

wenn er aus weniger als 28 Wörtern besteht. Aufgrund dieses Vorgehens variieren die Textlängen auf natürliche Weise. Die Textlängen über 28 Worten kommen durch sehr lange Sätze zustande, die bereits alleine die Maximallänge überschreiten.

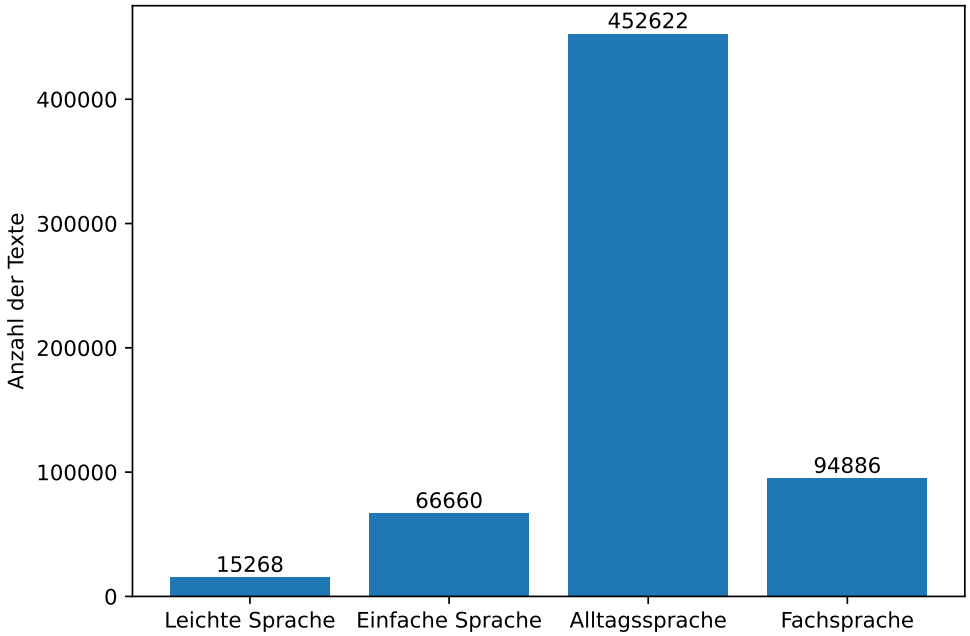

Abbildung 4.6: Unbalancierte Anzahl der Texte je Sprachkategorie

Die absolute Anzahl der Texte in den Sprachkategorien des Korpus unterscheidet sich, wie in der Abbildung 4.6 zu sehen, erheblich. Aufgrund der Verfügbarkeit von Quellen, stehen deutlich weniger Texte in den Sprachkategorien leichte Sprache und einfache Sprache zur Verfügung, als es sie für Alltags- und Fachsprache gibt. Bei der Verwendung eines Korpus für das Training von ML-Modellen ist mit einer deutlich schlechteren Vorhersagegenauigkeit in stark unterrepräsentierten Kategorien (Minderheitsklasse) zu rechnen (Shaikh u. a., 2021; Demirkıran u. a., 2022; ValizadehAslani u. a., 2022). Die folgenden Methoden haben sich bewährt (Krawczyk, 2016; Rocca, 2019; Kaur, Pannu und Malhi, 2019) im Umgang mit unausgeglichenen Datensätzen bei ML-Trainingsprozessen:

- Reduktion der Anzahl der Texte in den überrepräsentierten Kategorien durch zufälliges Entfernen (*Undersampling*)
- Generierung von Textvariationen aus bereits verwendeten Sätzen (*Oversampling*)
- Gewichtung der Kategorien im Trainingsprozess

Für die Balancierung der Anzahl der Texte werden alle drei Methoden angewendet. Die Quellen „Wortschatz Leipzig“ und „Open Discourse“ der Kategorie Alltagssprache stellen 59% der Texte des gesamten Korpus. Dementsprechend wird die Kategorie Alltagssprache stark überrepräsentiert und

eine Reduktion der Texte mittels *Undersampling* bietet sich an. Dafür wird die Anzahl der Sätze aus den beiden überrepräsentierten Quellen, die zur Generierung der Texte verwendet werden, um 90% bzw. 55% reduziert. Die Auswahl der zu entfernenden Sätze erfolgt zufällig.

Auch nach der Reduktion der Texte in den überrepräsentierten Sprachkategorien beträgt der Anteil der Texte in leichter Sprache nur 12% und ist somit unverhältnismäßig unterrepräsentiert. Um dem entgegenzuwirken, werden künstlich zusätzliche Texte in leichter Sprache generiert (*Oversampling*). Dafür werden die vorhandenen Sätze in leichter Sprache nicht, wie in den anderen Sprachkategorien nur einmal, sondern 2,5-mal bei der Generierung von unterschiedlichen Texten verwendet. Mit dieser Maßnahme kann der Anteil der Texte in leichter Sprache auf 30% der Anzahl der Texte in der am stärksten repräsentierten Kategorie bzw. der Mehrheitsklasse (Alltagssprache) gesteigert werden. Bei einem zu aggressiven *Oversampling* und einem Verhältnis von über 30% besteht die Gefahr des Overfitting in Bezug auf die verwendeten Texte der unterrepräsentierten Kategorie bzw. der Minderheitsklasse (Fernández u. a., 2018, S. 83). Die Anzahl der Texte der Sprachkategorien einfache Sprache und Fachsprache werden nicht mittels *Oversampling* erhöht, da das Ungleichgewicht im Verhältnis zur Mehrheitsklasse deutlich geringer ist und mittels Gewichtung im Trainings- bzw. Fine-Tuning Prozess ausgeglichen werden kann. Somit wird die Wahrscheinlichkeit der Overfitting zusätzlich reduziert.

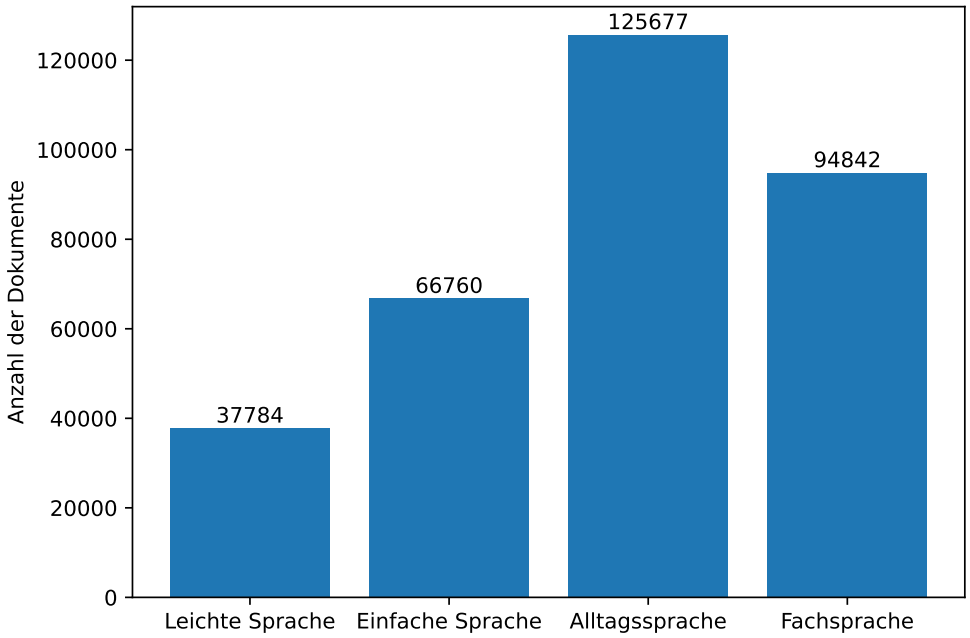

Abbildung 4.7: Balancierte Anzahl der Texte je Sprachkategorie

Nach der Anwendung des *Undersampling* und *Oversampling* werden die Sprachkategorien deutlich balancierter im Korpus repräsentiert (vgl. Abbildung 4.7) und können nachfolgend für das Training und Fine-Tuning von ML-Modellen verwendet werden. Die geringe Abweichung der Anzahl der Tex-

te in den Sprachkategorien einfache Sprache und Fachsprache zwischen der Abbildung 4.6 und der Abbildung 4.7 kommt aufgrund der erneuten Generierung der Texte zustande. Diese war für die Anwendung der Methoden zur Balancierung notwendig. Da bei jedem Generierungsvorgang die Sätze zufällig zu neuen Texten zusammengesetzt werden, ist aufgrund der unterschiedlichen Satzlängen und der Maximallänge keine exakt reproduzierbare Anzahl an Texten möglich.

4.1.4 *Fazit*

Für das Training und das Fine-Tuning von ML-Modellen stellt das neue Korpus im Rahmen dieser Arbeit einen hochqualitativen und umfassenden Datensatz dar. Dafür wurden Texte aus bestehenden Korpora und von verschiedenen Webseiten extrahiert, bereinigt, in Sätze segmentiert und zu neuen Texten in den typischen Längen einer Chatbot-Antwort zusammengesetzt. Aufgrund der sehr unterschiedlichen Verfügbarkeit von Texten in einigen Sprachkategorien wurden mittels *Undersampling* und *Oversampling* das Korpus balanciert. Das Korpus enthält 325.063 Texte in den Sprachkategorien leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache.

4.2 TEXTKOMPLEXITÄT KLASSIFIZIEREN MITTELS TRANSFORMER

Die Erfassung der Komplexität von Texten wurde lange Zeit von Formeln, wie dem Flesch-Index, dominiert. Diese Formeln berechnen einen Index auf der Basis verschiedener Merkmale eines Textes. Zu dieser gehören beispielsweise die durchschnittliche Wortlänge oder die Anzahl der Worte je Satz. In der jüngeren Vergangenheit konnten immer mehr Merkmale aus einem Text gewonnen werden. Der Einsatz verschiedener Methoden der Computer Linguistik bzw. NLP hilft dabei auch komplexere Merkmale zu extrahieren. In der aktuellen Forschung werden zunehmend ML-Algorithmen eingesetzt, um mithilfe sehr vieler Merkmale bessere Aussagen zur Textkomplexität treffen zu können. Die Extraktion tiefgehender Merkmale ist, wie in Kapitel 3 dargestellt, aufwendig und teilweise fehleranfällig. Die Genauigkeit der Komplexitätsklassifikation bisheriger ist Methoden teilweise umstritten und Gegenstand der Forschung.

Im Kontrast dazu ist es Ziel dieser Arbeit, Methoden zur Textkomplexitätseinschätzung abseits der merkmalbasierten, klassischen Komplexitätsklassifizierung zu untersuchen. Die Forschung klassifiziert Textkomplexität bisher vorzugsweise in Jahrgangsstufen oder in abstrakten Indexen. Im Rahmen der Forschungsfrage dieser Arbeit soll beantwortet werden, ob beliebige Texte den Komplexitätskategorien leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache automatisiert mittels Transformer-Modellen zugeordnet werden können.

Zu diesem Zweck wird die Leistungsfähigkeit zur Textklassifizierung Transformer basierter neuraler Netzwerke untersucht. Transformer haben seit ih-

rer Erfindung im Jahr 2017 (Vaswani u. a., 2017) und mit der Veröffentlichung eines ersten Sprachmodells BERT (Devlin u. a., 2019) im Bereich des Natural Language Processing für große Fortschritte gesorgt. Sie ermöglichen neben viele anderen NLP-Aufgaben auch die Klassifizierung von Texten, ohne auf explizites Feature Engineering (Extraktion von Merkmalen) angewiesen zu sein. Dabei übertreffen sie traditionelle Verfahren zur Textklassifizierung, wie CNNs (Convolutional Neural Network), RNNs (Recurrent Neural Network), GRUs (Gated Recurrent Unit) oder LSTM (Long Short-Term Memory) bei Benchmark-Klassifizierungstests in der Genauigkeit deutlich (Lu, Ehwerhemuepha und Rakovski, 2022).

Transformer setzen mit dem Konzept des Transfer-Learning zudem einen neuartigen Ansatz im Vergleich zu den traditionellen Verfahren des NLP um. Dabei wird ein Modell unüberwacht mithilfe eines großen Textkorpus trainiert. Das Modell repräsentiert zunächst erst einmal Sprache abstrakt auf statistische Weise. Diese Modelle werden auch als Sprachmodelle bezeichnet. Für dieses Vortraining kommen unterschiedliche Transformer-Implementierungen, auch Architekturen genannt, wie BERT oder ELECTRA, zum Einsatz. Erst durch das Fine-Tuning (überwachtes Lernen) eines Sprachmodells wird dieses auf eine bestimmte Aufgabe wie die Klassifizierung von Texten trainiert. Der Ansatz wird als Transfer-Learning bezeichnet. Er bietet den Vorteil, die Ressourcen für das Vortraining eines Sprachmodells, nur einmalig aufbringen zu müssen. Mit deutlich geringerem Ressourceneinsatz können dann auf Basis eines Sprachmodells konkrete Aufgaben nachtrainiert (Fine-Tuning) werden. Die daraus entstehenden Transformer-Modelle sind dann für diese Aufgabe spezialisiert. Das Sprachmodell muss dabei nicht immer wieder neu aufwendig trainiert werden. In der Literatur wird das Vortrainieren und das Fine-Tuning eines Sprachmodells häufig synonym als „trainieren“ bezeichnet. In dieser Thesis werden beide Prozesse aus sprachlichen Gründen teilweise ebenfalls synonym als Training bezeichnet.

4.2.1 *Vorgehen*

Ziel ist das Fine-Tuning von Sprachmodellen zur Klassifizierung eines Textes in einer der Komplexitätskategorien. Die Komplexitätsklassen eines Textes, leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache, werden dabei als Textkategorien betrachtet. Da jeder Text nur einer Kategorie bzw. einem Label zugeordnet werden soll, wird diese Transformer-Aufgabe als „multi-class text-classification“ bezeichnet. Um einem Sprachmodell die Zuordnung zu einer der Komplexitätsklassen beizubringen, findet ein Fine-Tuning verschiedener Sprachmodelle auf Basis des Korpus (vgl. Kapitel 4.1) statt. Das Korpus enthält eine große Anzahl an Texten aller Kategorien mit entsprechender Zuordnung und kann dementsprechend für das Training der Modelle verwendet werden.

Für das Fine-Tuning und die Evaluation der Modelle wird das Korpus in zwei Datensätze unterteilt. 80% aller Texte werden für das Fine-Tuning und

20% für die Evaluation verwendet. Die Auswahl der Texte geschieht, über alle Kategorien gleich verteilt, zufällig. Es findet eine schrittweise Evaluation der Checkpoints beim Fine-Tuning statt, um den besten Checkpoint zu ermitteln. Anschließend wird jeweils der beste Checkpoint als Transformer-Modell gespeichert und steht zur Evaluation aller untersuchten Modell zur Verfügung.

Die Leistungsfähigkeit eines Transformer-Modells hängt maßgeblich von der Wahl des zugrundeliegenden Sprachmodells und der gewählten Hyperparameter ab. Aus diesem Grund wird das Fine-Tuning auf mehrere Sprachmodelle mit mehreren Hyperparameter-Konfigurationen angewendet, mit dem Ziel, ein möglichst leistungsstarkes Modell zu trainieren. Ein leistungsstarkes Modell definiert sich in diesem Fall über eine möglichst präzise Vorhersage einer Komplexitätsklasse zu einem beliebigen Text in der Länge einer Chatbot-Antwort. Die Auswahl der Sprachmodelle und die Konfiguration der Hyperparameter wird in den nachfolgenden Abschnitten weiter erläutert.

Die Implementierung der Trainings-Skripte erfolgt in Python mit dem Simple Transformer Framework (vgl. Kapitel **??**). Es baut auf dem Hugging Face Transformers Framework (vgl. Kapitel 2.4) auf und bietet abstrahierte und vereinfachte Schnittstellen für das Fine-Tuning von Transformer-Modellen. Verarbeitungsschritte, wie die Auswahl und Anwendung des zur Architektur des Sprachmodell passenden Tokenizer, übernimmt dabei das Framework.

Das Fine-Tuning benötigt deutlich weniger Rechenleistung als das initiale Training der Sprachmodelle. Allerdings ist eine leistungsstarke Grafikkarte auch für das Fine-Tuning notwendig. Aufgrund dessen wurden die Trainings-Skripte auf der Infrastruktur von Google Colab Pro+[37] auf NVIDIA Tesla T4 GPUs ausgeführt.

4.2.2 *Hyperparameter*

Das Sprachmodell und die gewählten Hyperparameter beeinflussen maßgeblich die Leistungsfähigkeit des trainierten Transformer Modells. Aufgrund dessen werden mehrere Sprachmodelle mit jeweils zwei Hyperparameter-Sets trainiert. Je nachdem, welche Hyperparameter für das Fine-Tuning gewählt werden, kann die benötigte Rechenleistung stark ansteigen. Idealerweise wird mit Hyperparameter Tuning-Methoden, wie Gridsearch, durch Ausprobieren verschiedener Parameterkombinationen nach der optimalen Konfiguration gesucht. Aufgrund der dafür notwendigen besonders umfangreichen Rechenleistung und der verfügbaren Kapazitäten musste allerdings auf die Optimierung der Hyperparameter verzichtet werden.

Um die Anzahl der notwendigen Fine-Tuning-Durchläufe zu reduzieren und Parameterkombinationen auszuschließen, die unverhältnismäßig viele Re-

37 https://colab.research.google.com/signup

chenkapazitäten benötigen, werden stattdessen zwei Parameterkonfigurationen festgelegt. Beide Parametersets setzen sich aus den Standardparametern des Hugging Face Transformers Frameworks und des Simple Transformers Frameworks zusammen. Die Parameter wurden so gewählt, dass sie für viele Sprachmodelle und Fine-Tuning-Aufgaben verwendet werden können. Sie erzielen gute Ergebnisse, aber entsprechen nicht der optimalen Konfiguration in Bezug auf die gewählten Sprachmodelle und die Klassifizierungsaufgabe. Die Tabelle 4.1 zeigt die einzelnen Parameter der beiden Konfigurationen. Die Parameter „Train Batch Size“ und „Eval Batch Size“ wurden abweichend von den Standardwerten an den über Google Colab Pro+ verfügbaren CPU-Hauptspeicher und GPU-Hauptspeicher angepasst.

Konfiguration	**Learning Rate**	**Train Epochs**	**Weight Decay**	**Train Batch Size**	**Eval Batch Size**
Hugging Face	2e-5	5	0.01	24	16
Simple Transformers	4e-5	3	0	24	16

Tabelle 4.1: Hyperparameter-Konfiguration für das Fine-Tuning der Sprachmodelle

4.2.3 *Sprachmodelle*

Seit der Veröffentlichung des ersten Transformer-Sprachmodells BERT (Devlin u. a., 2019) wurden Fortschritte in der Entwicklung erzielt und neue leistungsfähigere Sprachmodelle veröffentlicht. Zuerst lag der Fokus der Leistungssteigerung auf der Vergrößerung des Korpus, der für das Vortraining des Sprachmodells verwendet wurde. Durch die gesteigerte Anzahl der Parameter konnten bessere Ergebnisse in den Standardbenchmarks, wie GLUE, erzielt werden. Das wurde allerdings nur auf Kosten der notwendigen Rechenleistung möglich.

Im Gegensatz dazu wurden in der jüngeren Vergangenheit verbesserte Architekturen, wie ELECTRA, entwickelt. Diese ermöglichen es mit einem kleineren Korpus und weniger eingesetzter Rechenleistung eine vergleichbare Leistungsfähigkeit bzw. Genauigkeit des trainierten Sprachmodells zu erreichen. Wenn hingegen die gleiche Rechenleistung eingesetzt wird, kann teilweise die Leistungsfähigkeit der urspünglichen Architekturen übertroffen werden.

Für die Experimente wurden verschiedene, zum Teil besonders ressourceneffiziente, Sprachmodelle ausgewählt. Diese Sprachmodelle eigenen sich besonders für Klassifizierungsaufgaben aufgrund ihrer Encoder-Architektur und haben sich zudem für „multi-class text-classification“ Aufgaben besonders etabliert (Mráz, o. D.; Yu, o. D.; Shaheen, Wohlgenannt und Filtz, 2020; Gasparetto u. a., 2022):

- *BERT*
  Die BERT-Architektur (Devlin u. a., 2019) wurde zum Standard in NLP. Es wurde ein großes Transformer basiertes neurales Netz mithilfe eines ebenfalls sehr großen Korpus trainiert. Für das Fine-Tuning wurde das deutsche BERT-Sprachmodell *gbert-base* (Chan, Schweter und Möller, 2020) verwendet.

- *ELECTRA*
  Für die Einführung der ELECTRA-Architektur (Clark, Luong und Le, 2020) wurde der Ansatz des Vortrainings im Vergleich zur BERT-Architektur grundlegend geändert. Dadurch benötigt ein ELECTRA basiertes Sprachmodell weniger als 25% der Rechenleistung, die für das Training von RoBERTa oder XLNet notwendig ist. Die Leistungsfähigkeit bleibt dabei vergleichbar. Für das Fine-Tuning wurde das deutsche ELECTRA-Sprachmodell *gelectra-base* (Chan, Schweter und Möller, 2020) verwendet.

- *XLNet*
  Im Gegensatz zu den anderen Architekturen ist XLNet eine autoregressive, Decoder basierte Weiterentwicklung der Transformer-XL Architektur. Obwohl die Klasse der Decoder-Architekturen typischerweise nicht für die Textklassifizierung verwendet wird, werden damit immer wieder sehr gute Ergebnisse erzielt. Es kam für das Vortraining ein sehr umfangreicher Korpus zum Einsatz. Für das Fine-Tunining wurde das XLNet-Sprachmodell *xlnet-based-cased* verwendet.

- *RoBERTa*
  Die RoBERTa-Architektur (Liu u. a., 2019) modifiziert die BERT-Architektur in der Wahl der Parameter und einiger interner Methoden. Im Vergleich zu BERT wurden für das Vortraining allerdings ein erheblich größerer Korpus und mehr Trainingsepochen verwendet. Für das Fine-Tuning wurde das RoBERTa-Sprachmodell *roberta-base* verwendet.

- *DistilBERT*
  Die DistilBERT-Architektur (Sanh u. a., 2020) baut vollständig auf der BERT-Architektur auf. Nach dem Vortraining eines klassischen BERT-Sprachmodells wird das Modell um 40% durch die Anwendung verschiedener Methoden komprimiert. In den Standardbenchmarks GLUE kann DistilBERT eine 95% Genauigkeit des BERT-Sprachmodells erreichen und ist dabei kleiner und effizienter. Für das Fine-Tuning wurde das DistilBERT-Sprachmodell *distilbert-base-cased* verwendet.

4.3 TEXTKOMPLEXITÄT KLASSIFIZIEREN MITTELS SVMS

Die aktuelle Forschung zur Messung der Lesbarkeit von Texten bedient sich zunehmend Methoden des Machine Learning für die Kombination klassischer Lesbarkeitsindizes als auch in der Auswertung vieler Textmerkmale (vgl. Kapitel 3.2). Support Vector Machines (SVM) sind eine beliebte ML-

Methode zur Klassifizierung der Lesbarkeit eines Textes auf Basis zahlreicher extrahierter Merkmale. Im Vergleich zu Transformer wird für den Einsatz von SVMs deutlich weniger Rechenleistung benötigt. Für die Extraktion der Merkmale aus Texten werden hingegen häufig ressourcenintensive NLP-Methoden verwendet.

Im Rahmen der Forschungsfrage dieser Arbeit soll ein transformersbasierter Ansatz zur Klassifizierung der Textkomplexität in die Kategorie leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache untersucht werden. Zur Einordnung der Leistungsfähigkeit dieses Ansatzes im Vergleich zu einem klassischen, merkmalbasierten Ansatzes wird ein SVM Modell trainiert und zur Evaluation bereitgestellt.

4.3.1 *Vorgehen*

Das Training eines SVM Modells zur Textklassifizierung erfordert wie das Training von Transformer-Modellen einen umfangreichen Korpus. Im Vergleich zu Transformern sind Support Vectors Machines allerdings auf explizit extrahierte Merkmale angewiesen. Merkmale, wie die Anzahl der Passivsätze oder die Anzahl der verwendeten Abkürzungen in einem Text, ermöglichen einen Rückschluss auf die Komplexitätskategorien. Dies lässt sich an einem Beispiel verdeutlichen. Die Regeln der leichten Sprache[38] sehen einen Verzicht auf Abkürzungen vor. Für die einfache Sprache gibt es bisher keine einheitliche Regel. Trotzdem wird von den meisten Regelwerken geraten, auf Abkürzungen zu verzichten. In der Alltagssprache sind Abkürzungen fester Bestandteil, während die Anzahl der Verwendungen in der Fachsprache nochmals deutlich höher liegt. Diese Kennzahl bzw. dieses Merkmal und weitere Merkmale eines Textes in Kombination mit der Zuordnung zu einer Komplexitätsklasse werden als Input für die Support Vector Machine verwendet.

Da das Korpus bisher nur aus den Texten und den Zuordnungen zu einer der Komplexitätsklassen besteht, muss er um Merkmale zu den Texten erweitert werden. Der Abschnitt 4.3.2 beschreibt die Gewinnung der Merkmale aus den Texten des Korpus.

Nach der Erweiterung des Korpus muss der Datensatz vorverarbeitet werden, um ihn als Input der Support Vector Machnine verwenden zu können. Merkmale, die nicht relevant oder hinderlich für die Unterscheidung der Komplexitätsklassen sein könnten, werden entfernt. Der Abschnitt 4.3.3 beschreibt die Vorverarbeitung der Daten. Wie für das Fine-Tuning der Transformer-Modelle, wird auch für das Training der Support Vector Machine der Datensatz zu 80% in einen Trainingsdatensatz und zu 20% in einen Testdatensatz aufgeteilt.

38 https://www.leichte-sprache.org/wp-content/uploads/2017/11/Regeln_Leichte_Sprache.pdf

Anschließend wird mithilfe der Gridsearch Methode des Scikit-learn Frameworks nach guten Hyperparametern für das Training des SVM-Modells gesucht. Dafür werden strukturiert unterschiedliche Werte für die Parameter C, Gamma und Kernel für das Training des Modells eingesetzt und im Anschluss evaluiert. Auf den Einsatz des Sigmoid Kernels wird verzicht, da seine Leistungsfähigkeit bei Multi-class Klassifizierungen im Vergleich zu den anderen Kernelfunktionen RBF, Poly und Linear schlechter abschneidet (Cho, Gantulga und Choi, 2017). Das leistungsstärkste Modell wird mit der Kernelfunktion RBF und den Werten 0,01 für Gamma und 1 für C durchgeführt.

Das Framework verwendet den „One-vs-rest" Ansatz. Dabei wird die Klassifizierung in eine von mehreren Kategorien heruntergebrochen. Dafür wird jede Klassifizierungsentscheidung einer Sprachkategorie als binäre Klassifizierung betrachtet. Die Datenpunkte der Merkmale werden jeweils in Komplexitätskategorie X und Rest eingeteilt. Dementsprechend wird nacheinander eine bestimmte Komplexitätsklasse von allen anderen unterschieden.

Für die Suche nach geeigneten Hyperparametern und für die Implementierung der eigentlichen Support Vector Machine kommt das Scikit-learn Frameworks[39] zum Einsatz. Das Modell wird nach dem Training für die anschließende Evaluation gesichert.

4.3.2 *Extraktion der Merkmale*

Die Support Vector Machine ist auf linguistische Merkmale zur Unterscheidung der Komplexitätskategorien angewiesen. Die Extraktion solcher Merkmale ist aufwendig und ist Gegenstand der Forschung. Die Software LanguageTool[40] bietet die Möglichkeit, Texte auf Grammatik, Stil und Rechtschreibung zu überprüfen, Fehler zu markieren und Verbesserungen vorzuschlagen. Sie wird Open Source entwickelt und bietet zudem die Möglichkeit eigene Regelwerke zu entwickeln.

Ein Teil der Regeln[41] der leichten Sprache wird mit dem Regelwerk „de-DE-x-simple-language" abgebildet. Es wurde ursprünglich von Annika Nietzio (Nietzio, Naber und Bühler, 2014) veröffentlicht und wird zur Zeit von ihr und anderen Freiwilligen gepflegt. Mit dem Regelwerk und der leistungsstarken Sprachverarbeitung des LanguageTools, ist es in der Lage beispielsweise Passivsätze, die in der leichten Sprache vermieden werden sollen, zu markieren.

Zur Extraktion von Merkmalen aus den Texten des Korpus wird ein Cluster aus mehreren Instanzen des LanguageTool-Servers betrieben. Ein Python-Skript übergibt die Texte des Korpus nacheinander an die API der Server und führt eine Überprüfung der Texte nach dem Regelwerk der leichten

39 https://scikit-learn.org/
40 https://languagetool.org/de/
41 https://multisprech.org/einfache-sprache/einfach-schreiben/languagetool-leichte-sprache/

Sprache aus. Die normale Überprüfung auf Grammatik, Stil und Rechtschreibung ist dabei deaktiviert. Die Server markieren Fehler in den Texten und geben Informationen zu diesen zurück. Das Skript zählt die Anzahl der gefundenen Fehler in jeder der Kategorien, die das LanguageTool mit dem Regelwerk erfassen kann. Somit wird unter anderem die Anzahl der Passivsätze oder die Verwendung des Genitivs erfasst. So werden relevante Indikatoren zur Unterscheidung der Komplexitätskategorien über die Zähler der Fehlerkategorien als numerische Merkmale abgebildet. Neben den linguistischen Merkmalen des LanguageTools werden außerdem vier Lesbarkeitsindizes, die sich für die deutsche Sprache etabliert haben, berechnet: LIX-Index, Flesch-Index, gSmog-Index und die 4. Wiener Sachtextformel. Diese Merkmale können nach der Vorverarbeitung als Input für das Training der Support Vector Machine verwendet werden.

4.3.3 *Vorverarbeitung der Daten*

Nachdem verschiedene Merkmale aus den Texten des Korpus extrahiert wurden, müssen die Daten aufbereitet werden. Bei der Analyse der extrahierten Merkmale fällt auf, dass die Software LanguageTool nicht alle Fehlerkategorien zuverlässig erkennt. Die Verwendung des Präteritums beispielsweise sollte entsprechend des Regelwerks eigentlich markiert werden. Das hat allerdings in keinem der Texte funktioniert.

Andere Fehlerkategorien des Regelwerks bieten kein aussagekräftiges Unterscheidungskriterium zwischen den Sprachkategorien leichte Sprache, einfache Sprache, Alltagssprache oder Fachsprache. Beispielsweise konnten nach der Analyse der Merkmale keine signifikant unterschiedlichen Wahrscheinlichkeiten für die Verwendung von Datumsangaben ermittelt werden. Wie in Tabelle 4.2 zusehen, werden Datumsangaben von allen Sprachkategorien ungefähr gleich wahrscheinlich verwendet.

Anzahl der Verwendung von Datumsangaben	**0**	**1**	**2**	**3**
leichte Sprache	99,4%	0,5%	0,1%	0%
einfache Sprache	99,2%	0,8%	0%	0%
Alltagssprache	98,5%	1,4%	0,1%	0,0%
Fachsprache	99%	0%	0%	0%

Tabelle 4.2: Wahrscheinlichkeit der Verwendung von Datumsangaben in den Sprachkategorien.

Es wurden alle Merkmale entfernt, die nicht erkannt wurden oder sich nicht für die Unterscheidung zwischen den Sprachkategorien eignen.

Nach der Bereinigung nicht detektierter oder für die Klassifizierung irrelevanter Merkmale, bleiben folgende linguistische Merkmale für das Training eines SVM-Modells übrig: Anzahl der Worte, Anzahl der Sätze, Anzahl der

Abkürzungen, Anzahl der abstrakten Worte, Anzahl der Anglizismen, Anzahl der komplizierten Worte, Flesch-Index, Anzahl der Verwendung des Genitivs, gSmog-Index, LIX-Index, Anzahl der langen Worte, Anzahl der Verneinungen, Anzahl der Zahlworte, Anzahl der Zahlen, Anzahl der Passivsätze, Anzahl der Relativsätze, Anzahl der römischen Zahlen, Anzahl der Sonderzeichen, Anzahl der Fachausdrücke, Anzahl der Nebensätze, Anzahl der Sätze im Konjunktiv, Anzahl der Sätze mit mehr als einer Informationseinheit und die 4. Wiener Sachtextformel.

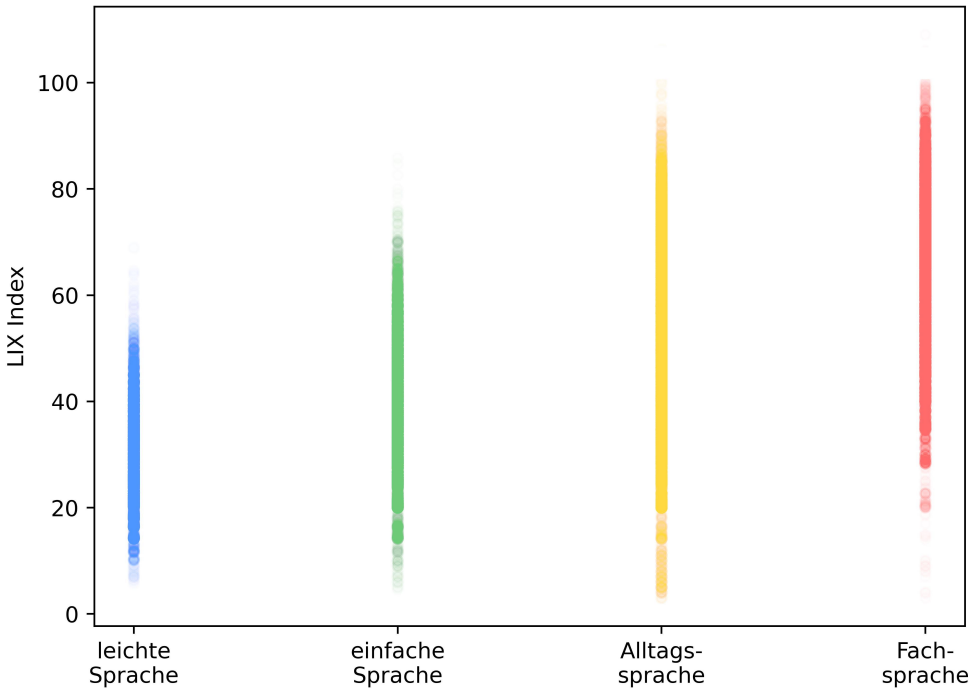

Abbildung 4.8: Die Wertebereiche des LIX-Index in den Sprachkategorien. Je kräftiger die Farbe, desto häufiger wird der spezifische LIX-Index für die Texte der Sprachkategorie berechnet.

Nicht alle Merkmale bieten ein ausgeprägtes Unterscheidungskriterium. In der Abbildung 4.8 sind exemplarisch die Wertebereiche der Sprachkategorien des LIX-Index abgebildet. Hierbei sind die Überlappungen der Wertebereiche hervorzuheben. Im Vergleich dazu eigenen sich andere Merkmale besser als Unterscheidungskriterium. In der Abbildung 4.9 wird die Wahrscheinlichkeit der Verwendung des Genitivs in den Texten der Sprachkategorien abgebildet. Die Abbildung zeigt, dass zu einer Wahrscheinlichkeit von mehr als 90% Texte in der Sprachkategorie leichte Sprache den Genitiv nicht verwenden, während etwa 85% der Texte in einfacher Sprache und 66% der Texte in Alltagssprache sowie 55% der Texte in Fachsprache auf die Anwendung des Genitivs verzichten.

Nach der Auswahl aussagekräftigen Merkmale, werden deren Zahlenwerte standardisiert. Bei der Standardisierung werden alle Zahlenwerte zentriert. Dafür wird von jedem Zahlenwert der Mittelwert abgezogen und dieser anschließend skaliert, indem er durch dessen Standardabweichung geteilt wird.

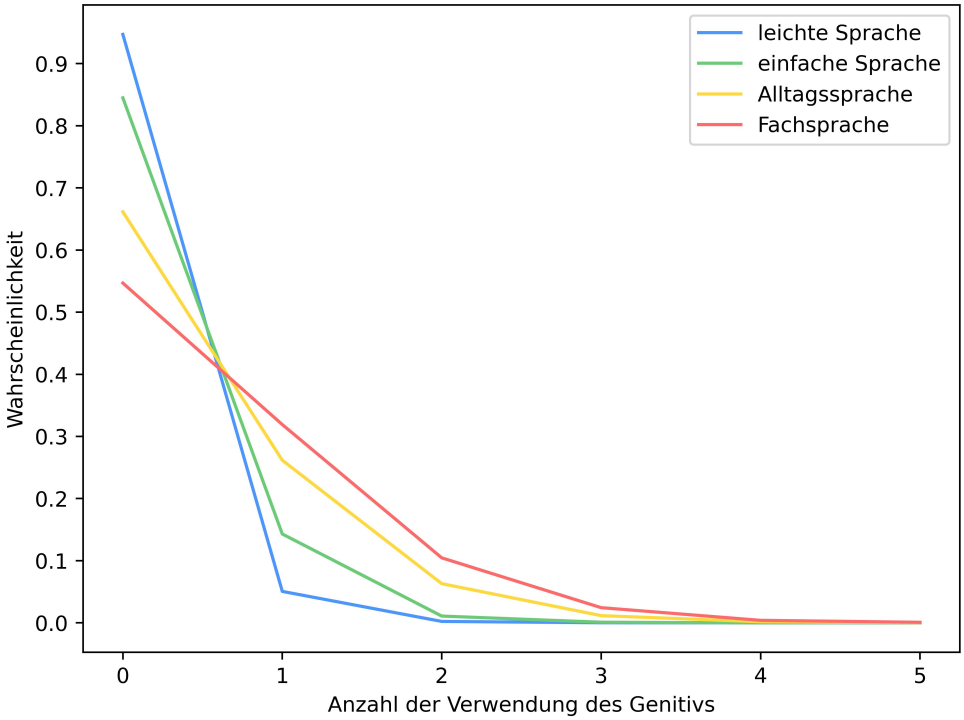

Abbildung 4.9: Wahrscheinlichkeiten, zu denen der Genitiv in den Texten der Sprachkategorien verwendet werden.

Nach der Standardisierung liegt der Mittelwert der Daten bei 0 und die Standardabweichung liegt bei 1. Durch die Standardisierung wird die Leistungsfähigkeit eines SVM-Modells stark verbessert.

Nachdem das Korpus um aussagekräftige linguistische Merkmale erweitert und deren Zahlenwerte für die Verwendung in der Support Vector Machine vorbereitet wurden, können die Texte aus den Inputdaten entfernt werden. Sie werden für die Klassifizierung nicht weiter benötigt.

5 EVALUATION

Zur Beantwortung der Forschungsfrage dieser Arbeit werden die zuvor trainierten Transformer-Modelle und das SVM-Modell bezüglich ihrer Leistungsfähigkeit evaluiert. Die Leistungsfähigkeit der Modelle bezieht sich darauf, wie präzise die Sprachklassen leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache vorhergesagt werden können. Je mehr Texte korrekt einer der Sprachklassen zugeordnet werden können, desto leistungsfähiger ist ein Modell.

Die Ermittlung der Leistungsfähigkeit soll dabei helfen, die Eignung von Transformer-Modellen zur Klassifizierung von Textkomplexität im Allgemeinen zu bewerten und eine geeignete Kombination aus Transformer-Sprachmodell und Hyperparameter-Konfiguration zu ermitteln. Außerdem soll sie den Vergleich und die Einordnung zwischen dem neuen Ansatz der Transformer-Modelle mit einem traditionellen Ansatz ermöglichen. Für das Fine-Tuning der Transformer-Modelle werden ausschließlich Texte des zuvor aufgebauten Korpus mit der Zuordnung zu einer der Sprachklassen verwendet. Im Gegensatz dazu wird für den traditionellen Ansatz eine Reihe linguistischer Merkmale aus den Texten extrahiert und mit diesen eine Support Vector Machine trainiert.

Die Klassifizierung der Texte wird im Machine Learning Kontext als „Multi-Class Classification“ Aufgabe bezeichnet. Es soll einer Datenmenge, in diesem Fall einem Text, nur eine Klasse aus einer Menge von Klassen zugewiesen werden. Dies ist nicht zu verwechseln mit einer „Multi-Label Classification“ Aufgabe. Bei dieser werden die Wahrscheinlichkeiten zur Zugehörigkeit einer Klasse für alle Klassen parallel berechnet.

Für die Evaluation der Transformer-Modelle und des SVM-Modells werden 20% der Texte des Korpus verwendet. Das Korpus wurde bereits vor dem Training der Modelle in Trainings- und Evaluationsdaten aufgeteilt, um die Mehrfachverwendung der Texte für beide Anwendungen auszuschließen. Die Evaluation wird, wie das Training der Modelle, auf der Infrastruktur von Google Colab Pro+ ausgeführt.

5.1 METRIKEN

Zur Bewertung der Leistungsfähigkeit von ML-Modellen stehen verschiedene Metriken, wie Accuracy oder F1 zur Verfügung. Die Accuracy-Metrik gibt den Anteil der korrekten Klassifizierungen im Verhältnis zu allen Klassifizierungen an. Aufgrund dessen eignet sie sich nicht für die Verwendung in der

Evaluation der Modelle mit dem zur Verfügung stehenden Korpus. Da die Anzahl der Texte im Korpus je Sprachklasse nicht gleichmäßig verteilt ist, stellt die Metrik nur einen verzerrten Indikator zur Leistungsfähigkeit dar. Das Korpus enthält mehr Texte, die zur Sprachklasse „Alltagssprache “ gehören, als es jeweils Texte für die anderen Sprachklassen gibt (vgl. Abbildung 4.7). Sollten beispielsweise die Texte der Sprachkategorie „Alltagssprache “ besonders häufig korrekt klassifiziert werden, nimmt die Accuracy-Metrik einen höheren Wert an, weil es anteilig mehr korrekte Klassifizierungen gibt. Wenn die weniger stark repräsentierten Sprachklassen tendenziell eher falsch klassifiziert werden sollten, würde deren Anteil aber aufgrund der absoluten Anzahl der Klassifizierungen nur geringer in die Metrik einfließen. Somit wird die Metrik von der Leistungsfähigkeit der stärksten Sprachklassen dominiert. Die Vorhersage oder Zuordnung eines Textes zu einer Sprachklasse durch ein ML-Modell wird nachfolgend auch als Klassifizierung bezeichnet.

Aus diesem Grund hat sich die F1-Metrik in der wissenschaftlichen Community zur Bewertung von „Multi-Class Classification“ Aufgaben etabliert. Sie setzt sich aus zwei weiteren Metriken, Precision und Recall zusammen und beschreibt das harmonische Mittel beider Werte.

Die Metrik Precision (Präzision) gibt den Anteil der korrekten Klassifizierungen an der Summe aller Klassifizierungen einer Sprachklasse an. Sie beantwortet die Frage: Wie viel der Klassifizierungen sind tatsächlich korrekt? Die Metrik wird aus der Anzahl der korrekten Klassifizierungen (TP: True Positives) und der Summe aus korrekten Klassifizierungen und falschen Klassifizierungen (FP: False Positives) einer Sprachklasse berechnet:

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

Die Metrik Recall (Abdeckung) gibt das Verhältnis zwischen den korrekten Klassifizierungen und der Anzahl aller Texten einer Sprachklasse an. Sie beantwortet die Frage: Wie viele der Texte einer Sprachklasse wurden tatsächlich korrekt klassifiziert? Die Metrik wird aus der Anzahl der korrekten Klassifizierungen (TP: True Positivs) und der Summe aus korrekten Klassifizierungen einer Sprachklasse und der Anzahl an Texten in der gleichen Sprachklasse, die falsch klassifiziert wurden (FN: False Negatives):

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

Die F1-Metrik kombiniert die Metriken Precision und Recall miteinander. Sie beschreibt das harmonische Mittel zwischen Recall und Precision. Je mehr Klassifizierungen einer Sprachklasse tatsächlich korrekt sind und je mehr Texte der gleichen Sprachklasse korrekt klassifiziert werden, desto höher ist der Wert der Metrik. Nur wenn beide Teilmetriken einen hohen Wert einnehmen, dann steigt auch der Wert der F1-Metrik. Die F1-Metrik berechnet sich

aus dem Verhältnis des Produktes und der Summe der Metriken Precision und Recall:

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

Die Berechnung der F1-Metrik kann auf unterschiedliche Weise durchgeführt werden. Bei der Micro-F1-Metrik werden die Werte der True Positives, False Positives und False Negativs über alle Sprachklassen gemeinsam berechnet. Aufgrund dessen nehmen die Metriken Precision und Recall den gleichen Wert an. Dementsprechend ergibt die Berechnung der F1-Metrik ebenfalls den gleichen Wert. Dieser entspricht zugleich der Accuracy-Metrik. Die Metriken Precision, Recall, F1 und Accuracy können dementsprechend in der Berechnungsmethode „micro " gleichgesetzt werden. Der Begründung des Ausschlusses der Accuracy-Metrik folgend, wird die Micro-F1-Metrik für die Evaluation der Leistungsfähigkeit ebenfalls ausgeschlossen.

Die Weighted-F1-Metrik berechnet die Metriken Precision, Recall und F1 im ersten Schritt für jede Sprachklasse getrennt. Anschließend wird das gewichtete Mittel der Werte gebildet. Für die Gewichtung wird das Verhältnis der Anzahl der Texte einer Sprachkategorie zu allen Texten verwendet. Dementsprechend eignet sich diese Version der F1-Metrik ebenfalls nicht für den unbalancierten Korpus, da die Leistungsfähigkeit der Klassen nicht gleich berücksichtigt wird.

Für die Evaluation ist es von besonderem Interesse die Leistungsfähigkeit aller Klassen über eine Metrik abzubilden, um Schwächen bei der Klassifizierung einer oder mehrerer Sprachklassen, für die weniger Texte im Korpus zur Verfügung stehen, identifizieren zu können. Dafür eignet sich die Macro-F1-Metrik. Die Metriken Precision, Recall und F1 werden dabei für jede Sprachklasse getrennt berechnet. Anschließend wird der Durchschnitt der Werte gebildet. Dabei findet keine Gewichtung statt und alle Sprachklassen werden auch bei einem unbalancierten Korpus gleich berücksichtigt. Nachfolgend wird die Macro-F1-Metrik verkürzt als F1-Metrik bezeichnet.

5.2 ERGEBNISSE

Die Tabelle 5.1 zeigt die Ergebnisse der Evaluation der Transformer-Modelle. Die Abkürzungen HF und ST in der Spalte Hyperparameter stehen für die verwendeten Hyperparameter-Konfigurationen von Hugging Face und Simple Transformers. Das beste Ergebnis konnte mit dem *gbert-base* Modell bei einem F1-Wert von 0.982 und der Hyperparameter-Konfiguration von Hugging Face erzielt werden. Fast gleichauf liegt das *gelectra-base* Modell mit einem F1-Wert von 0.981 unter Verwendung der Hyperparameter-Konfiguration von Hugging Face. Mit einem F1-Wert von 0.940 werden die Modelle *xlnet-base-cased* und *distilbert-base-cased* erzielen die Modelle die schlech-

testen Ergebnisse mit der Hyperparameter-Konfiguration von Simple Transformer.

Im Vergleich sind die Sprachmodelle *gbert-base* und *gelectra-base*, die explizit für die deutsche Sprache trainiert wurden, leistungsfähiger als die anderen Sprachmodelle. Auch schneiden alle Modelle, die mit der Hyperparameter-Konfiguration von Simple Transformers trainiert wurden, schlechter ab als ihre Pendants, für dessen Fine-Tuning die Hyperparameter-Konfiguration von Hugging Face verwendet wurde. Beide Konfigurationen unterscheiden sich in den Parametern Learning Rate, Train Epochs, und Weight Decay (vgl. Tabelle 4.1). Hugging Face verwendet eine kleinere Learning Rate (2e-5) im Vergleich zu Simple Transformer (4e-5). Auch die Anzahl der Train Epochs unterscheidet sich. Hugging Face setzt auf 5 Epochen und Simple Transformer auf 3. Der Parameter Weight Decay wird von Hugging Face auf 0.01 gesetzt und von Simple Transformer auf 0. Im Vergleich erfordert die Konfiguration von Hugging Face mehr Rechenleistung beim Fine-Tuning der Sprachmodelle, erzielt dafür aber auch besser Ergebnisse.

Auffällig ist ebenfalls, dass das Transformer-Modell auf Basis der BERT-Architektur leistungsfähiger ist, als das Transformer-Modell auf Basis der ElECTRA-Architektur. Die ELECTRA-Architektur ist eine Weiterentwicklung der BERT-Architektur und eigentlich dafür bekannt, bei dem Einsatz der gleichen Rechenleistung gleiche oder bessere Ergebnisse zu erzielen.

Modell	**Precision**	**Recall**	**F1**	**Hyperparameter**
gbert-base	**0.981**	**0.983**	**0.982**	**HF**
gbert-base	0.967	0.966	0.966	ST
gelectra-base	0.980	0.982	0.981	HF
gelectra-base	0.967	0.967	0.967	ST
xlnet-base-cased	0.953	0.963	0.958	HF
xlnet-base-cased	0.936	0.944	0.940	ST
roberta-base	0.957	0.965	0.961	HF
roberta-base	0.941	0.947	0.943	ST
distilbert-base-cased	0.958	0.965	0.961	HF
distilbert-base-cased	0.939	0.942	0.940	ST

Tabelle 5.1: Ergebnisse der Evaluation der Transformer-Modelle
HF: Hugging Face Hyperparameter-Konfiguration
ST: Simple Transformer Hyperparameter-Konfiguration

Das Ergebnis der Evaluation der Leistungsfähigkeit des SVM-Modells wird in der Tabelle 5.2 dargestellt. Im Vergleich zu den Transformer-Modellen konnte das beste SVM-Modell nur einen deutlich geringeren F1-Wert von 0.728 erreichen.

Neben der Evaluation der Leistungsfähigkeit der Modelle wird zusätzlich für jedes Modell eine Konfusionsmatrix generiert. Sie gibt eine detaillierte

Modell	Precision	Recall	F1	Hyperparameter
SVM	0.732	0.726	0.728	Kernel: RBF, C: 1 und Gamma 0.01

Tabelle 5.2: Ergebnisse der Evaluation des SVM-Modells

Übersicht über die Genauigkeit der Klassifizierung in den jeweiligen Sprachklassen eines Modells. Auf der x-Achse werden die vorhergesagten Sprachklassen und auf der y-Achse die korrekten Sprachklassen aufgetragen. In den Zellen der Konfusionsmatrix wird die normalisierte Anzahl der Texte abgebildet, auf die die entsprechende Zuordnung zwischen Vorhersage des Modells und korrekter Sprachklasse zutrifft. In der Diagonalen von oben links bis unten rechts wird die Anzahl der korrekt klassifizierten Texte dargestellt.

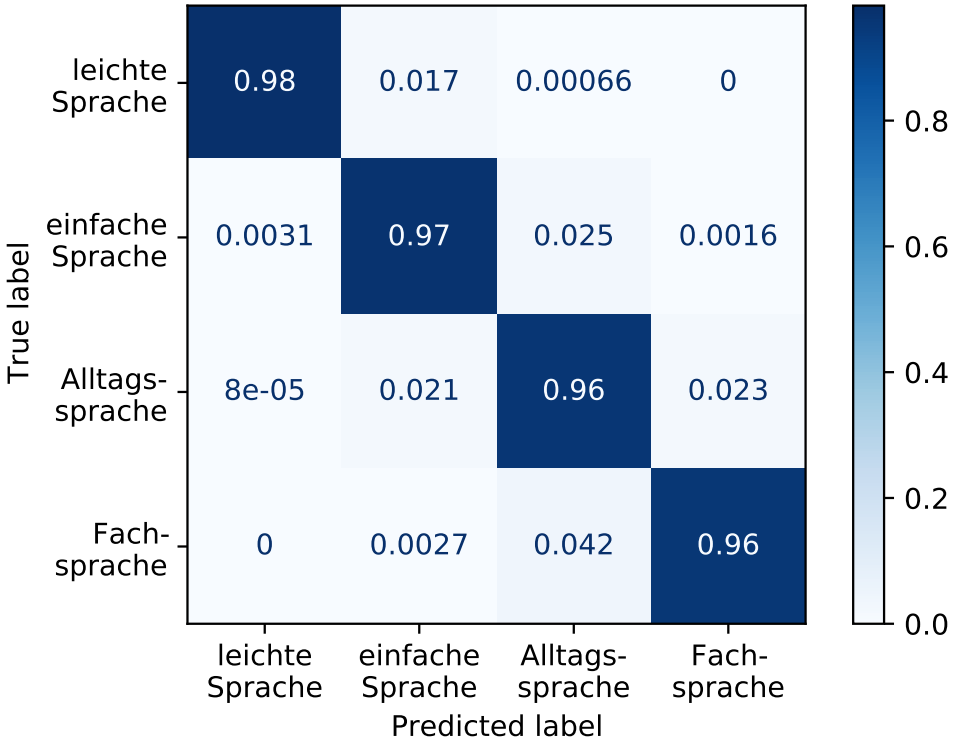

Abbildung 5.1: Normalisierte Konfusionsmatrix des Modells gbert-base HF

In der Konfusionsmatrix der Abbildung 5.1 wird die Genauigkeit der Klassifizierung in den einzelnen Sprachkategorien des besten Transformer-Modells *gbert-base* gezeigt. Es werden 98% der Texte in leichter Sprache auch als leichte Sprache klassifiziert. Im Vergleich dazu werden 97% der Texte in einfacher Sprache und 96% der Texte in Alltagssprache und Fachsprache als solche klassifiziert. Das belegt eine sprachklassenübergreifend hohe Genauigkeit in der Klassifizierung. Besonders interessant ist die geringere Klassifizierungsgenauigkeit in den Sprachklassen Alltagssprache und Fachsprache. Für beide Klassen existieren jeweils mehr Texte im Korpus, als für die

Sprachklassen leichte Sprache und einfache Sprache. Demnach wäre mit einer höheren Klassifizierungsgenauigkeit zu rechnen gewesen. Da hingegen weniger Texte korrekt klassifiziert wurden, kann das als möglichen Hinweis auf ein Overfitting des Modells betrachtet werden. Die Texte beider Sprachkategorien stammen jeweils maßgeblich aus Quellen weniger Texttypen, wie News-Artikel oder wissenschaftlichen Zusammenfassungen. Das Modell ist möglicherweise stärker auf diese Texttypen trainiert und erkennt beispielsweise Texte in Fachsprache innerhalb eines anderen Texttyps weniger präzise. Für eine definitive Aussage ist allerdings eine weiterführende Analyse notwendig.

Das SVM-Modell weist erwartungsgemäß eine deutlich schlechtere Klassifizierungsgenauigkeit in den Sprachkategorien auf. Der Konfusionsmatrix aus der Abbildung 5.2 ist zu entnehmen, dass Texte in leichter Sprache mit einer überraschenden Genauigkeit von 90% erkannt werden, während die Genauigkeit der anderen Sprachkategorien bedeutend niedriger liegt. Eine mögliche Erklärung liegt in der Auswahl der linguistischen Merkmale. Alle Merkmale beziehen sich auf die Fehlerklassen der leichten Sprache und scheinen sich nur eingeschränkt für die Klassifizierung der anderen Sprachkategorien zu eignen. Hierfür scheinen weitere Merkmale, deren Werte sich zwischen den Sprachkategorien stärker unterscheiden, notwendig zu sein.

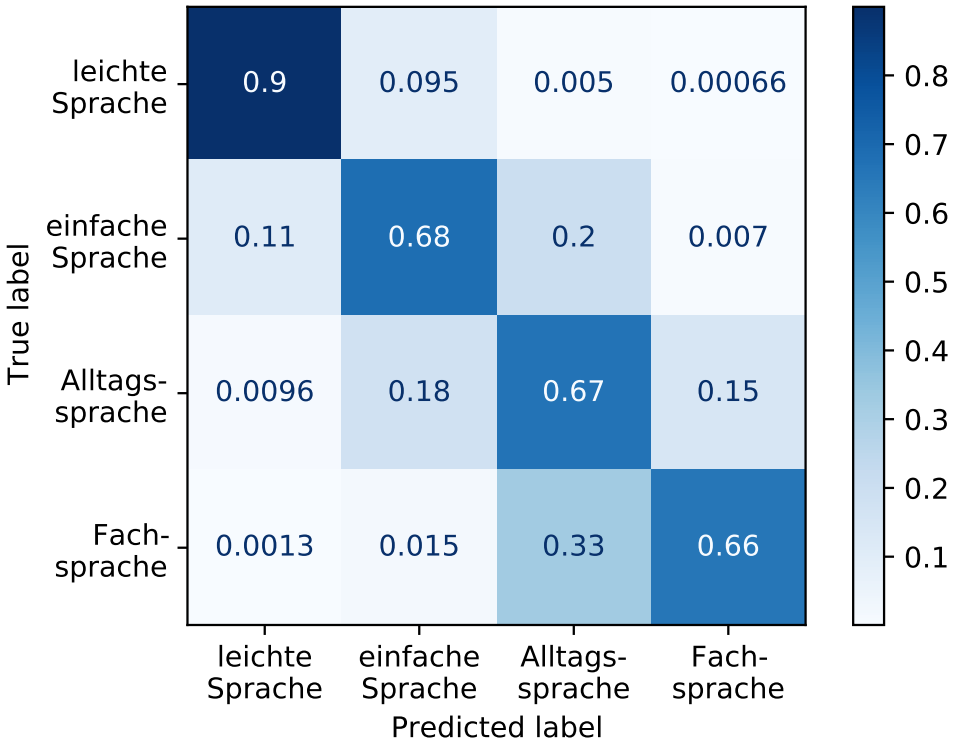

Abbildung 5.2: Normalisierte Konfusionsmatrix des SVM-Modells

Die Genauigkeit der Klassifizierung der einzelnen Sprachkategorien ist bei allen Transformer-Modellen trotz der unterschiedlichen F1-Werte gleichblei-

bend hoch. Selbst bei dem am schlechtesten abschneidenden Transformer-Modell *distilbert-base-cased* liegt die Klassifizierungsgenauigkeit bei mehr als 90%, wie in Abbildung 5.3 zu sehen. Ähnlich zum SVM-Modell ist allerdings auch bei diesem Modell eine, im Vergleich zu den anderen Kategorien, deutlich höhere Genauigkeit in der Klassifizierung der leichten Sprache festzustellen. Der Grund für diesen Unterschied konnte nicht im Rahmen dieser Arbeit ermittelt werden und sollte in weiterführenden Experimenten untersucht werden.

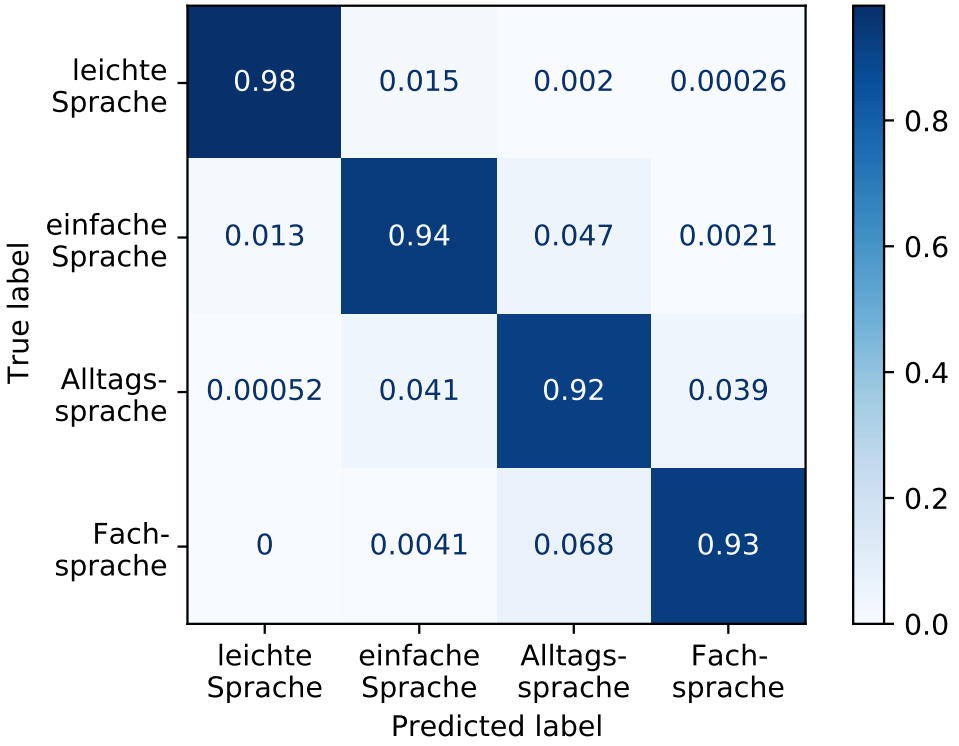

Abbildung 5.3: Normalisierte Konfusionsmatrix des Modells distilbert-base-cased ST

Zusammenfassend lassen sich folgende Schlüsse aus den Ergebnissen ziehen: Transformer-Modelle stellen eine leistungsstarke Alternative zu ML-Klassifikation-verfahren auf Basis von linguistischen Merkmalen dar. Die Hyperparameter-Konfiguration des Frameworks Hugging Face führt aus dem Stand beim Fine-Tuning von Transformer-Modellen zur Textklassifizierung zu leistungsstarken Ergebnissen. Die Auswahl der Transformer-Sprachmodelle hat sich bewährt. Allerdings führen aktuelle Architekturen, wie ELECTRA, nicht automatisch zu besseren Ergebnissen. Im Vergleich zu einer einfachen Textklassifizierung auf Basis einer Support Vector Machine und linguistischer Merkmale lassen sich die Sprachkategorien leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache deutlich besser mithilfe eines Transformer-Modells klassifizieren.

6

ZUSAMMENFASSUNG

Im Rahmen der vorliegenden Thesis wurde untersucht, ob sich Transformer-Modelle für die Klassifizierung von Chatbot-Antworten in die Sprachklassen leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache eignen. Die Sprachklassen geben Aufschluss über die Lesbarkeit einer Chatbot-Antwort. Mit der Klassifizierung eines solchen Textes ist der Autor in der Lage Anpassungen vorzunehmen, bis eine Chatbot-Antwort den Grad der gewünschten Lesbarkeit erreicht hat.

Da es bisher noch keinen frei zugänglichen Korpus gab, der zum einen Texte aller Sprachklassen enthält und zum anderen einen ausreichenden Umfang aufweist, wurde im Rahmen dieser Arbeit ein entsprechendes Korpus angelegt. Dafür wurden mehr als 100.000 Texte mittels selbst entwickelter Crawler von themenübergreifenden Webseiten extrahiert oder aus bereits bestehenden Korpora bezogen. Alle Texte wurden anschließend bereinigt, in einzelne Sätze zerlegt und innerhalb einer Sprachkategorie zufällig zu neuen Texten in den typischen Längen einer Chatbot-Antwort zusammengesetzt. Das Korpus umfasst ca. 325.000 Sätze und deren Zuordnungen zu einer der Sprachklassen. Für die Verwendung als Trainingsdaten eines SVM-Modells wurde das Korpus zusätzlich um eine Reihe linguistischer Merkmale der Texte erweitert.

Zur Untersuchung der Eignung von Transformer-Modellen zur Klassifizierung von Sprachklassen wurden mehrere Transformer-Sprachmodelle mit den Texten und Sprachklassen des Korpus trainiert, sowie deren Leistungsfähigkeit miteinander verglichen. Um die Einordnung der Ergebnisse im Vergleich zu traditionellen Ansätzen der Lesbarkeitsforschung zu ermöglichen, wurde außerdem ein klassisches SVM-Modell trainiert. Es ermöglicht ebenfalls die Klassifizierung eines Textes in die Sprachklassen, verwendet dafür allerdings die zuvor extrahierten linguistischen Merkmale des Korpus.

Die Evaluation der Modelle zeigt eine deutliche Überlegenheit des Transformer basierten Ansatz gegenüber dem SVM-Modell. Die Leistungsfähigkeit der untersuchten Transformer-Modelle in Bezug auf die Klassifizierung liegt bei allen Modellen bei einem F1-Wert von mindestens 0.94. Damit sind alle untersuchten Transformer-Modelle deutlich leistungsstärker als das SVM-Modell mit einem F1-Wert von 0.728. Innerhalb der Transformer-Modelle konnte das *gbert-base* Modell den höchsten F1-Wert von 0.982 erreichen und ist damit das leistungsstärkste Modell. Neben der Überlegenheit gegenüber dem SVM-Modell zeigt die Evaluation auch die sprachklassenübergreifend hohe Genauigkeit in der Klassifizierung.

6.0.1 *Fazit*

Die Auswertung der Evaluation führt zu dem Schluss, dass Transformer-Modelle eine vielversprechende Möglichkeit darstellen, um Texte in die Sprachkategorien leichte Sprache, einfache Sprache, Alltagssprache oder Fachsprache einzuordnen. Das ist aus mehreren Gründen bemerkenswert. Zum einen sind Transformer-Modelle offentsichlich in der Lage Texte aufgrund ihrer Struktur und nicht nur aufgrund ihres Themas zu klassifizieren. Bisher wurden Transformer-Modelle im Bereich NLP häufig auf inhaltlicher Ebene verwendet, um beispielsweise Hassrede herauszufiltern. Im Rahmen dieser Thesis wurden hingegen Texte verwendet, die keinem Thema eindeutig zuzuordnen sind und trotzdem konnte eine hohe Genauigkeit in der Zuordnung zu einer der Sprachklassen erzielt werden.

Zum anderen wurden bisher Klassifizierungen von Texten in ein Lesbarkeitsniveau mithilfe linguistischen Merkmalen und Formeln oder ML-Algorithmen, die diese Merkmale verwenden, vorgenommen. Im Gegensatz dazu sind Transformer-Modelle in der Lage relevante Merkmale zur Unterscheidung der Sprachkategorien intern selbst herauszufinden. Damit kann der mitunter sehr aufwendige Prozess zur Ermittlung der Merkmale ausgelassen werden. Dies gilt für die Aufbereitung der Daten für ein Training, als auch für den Klassifizierungsprozess selbst. Beispielsweise kann ein Text so deutlich schneller im Backend einer Chatbot-Plattform klassifiert werden. Dadurch kann der Autor direkte Rückmeldung erhalten.

Obwohl keine vollständige Hyperparameter-Optimierung vorgenommen werden konnte, erkannten die trainierten Transformer-Modelle mit beeindruckender Genauigkeit die Sprachkategorie von Texten in der Länge einer Chatbot-Antwort. Eine weiterführende Forschung zu diesem Thema erscheint vielversprechend.

Ein Vergleich der Leistungsfähigkeit mit State-of-the-Art ML-Modellen und auf Basis einer größeren Anzahl an linguistischen Merkmalen zur Klassifizierung in die Sprachkategorien waren im Rahmen dieser Arbeit nicht möglich. Vergleichbare Arbeiten anderer Autoren sind mir nicht bekannt. Auch ist die Forschung der Transformer-basierten Klassifikation in leichte Sprache, einfache Sprache, Alltagssprache und Fachsprache noch am Anfang, sodass ein Vergleich mit den Ergebnissen anderer Autoren nicht durchgeführt werden konnte.

In Bezug auf die Klassifikation eines Textes in leichte Sprache können die vorgestellten Modelle nur ungefähre Vorhersagen treffen. Für die korrekte Anwendung aller Regeln der leichten Sprache müssen weitere Eigenschaften eines Textes und dessen Darstellung betrachtet werden. Der Textaufbau in Form von Absätzen und die Kontrastverhältnisse der Texte zum Hintergrund werden von den Modellen nicht beachtet. Zudem ist ein Grundsatz bei der Verwendung von leichter Sprache eng mit der Zielgruppe zusammenzuarbeiten und Rückmeldungen zur Verständlichkeit einzuholen. Allein

aus diesen Gründen kann die Klassifikation eines Textes in leichte Sprache nur als Hinweis verstanden werden, nicht jedoch als Zertifizierung nach den vollständigen Regeln.

6.0.2 *Ausblick*

Die Untersuchungen dieser Arbeit lassen sich auf vielfältige Weise fortsetzen. Mit dem entsprechenden Einsatz von Ressourcen wäre die individuelle Optimierung der Hyperparameter und das Training weiterer Transformer-Sprachmodelle am interessantesten. Somit kann die klassenübergreifende Genauigkeit bei der Klassifizierung möglicherweise noch weiter gesteigert oder ein ressourcen-effizienteres Sprachmodell ermittelt werden.

Für einen tiefergehenden Vergleich zwischen dem traditionellen Merkmal basierten Ansatz und den Transformer-Modellen ist ein erneutes Training eines ML-Modells notwendig. Abseits der Support Vector Machines gibt es viele weitere Algorithmen, die möglicherweise bessere Ergebnisse in der Klassifikation erzielen. Neben der Wahl eines Algorithmus ist auch die Anzahl und die Auswahl der zur Unterscheidung der Sprachkategorien relevanten Merkmale bedeutend. In beiden Punkten kann die Arbeit der vorliegenden Thesis umfangreich fortgeführt werden.

Zusätzlich ist eine tiefergehende Analyse der Klassifikationsgenauigkeit in Korrelation mit der Länge der Texte von Interesse. Auch kurze Texte können beispielsweise besonders lange Wörter oder Fachwörter verwenden und von der Zielgruppe der leichten und einfachen Sprache schwer verstanden werden. Die Wahrscheinlichkeit einer fehlerhaften Klassifikation steigt womöglich bei besonders kurzen Texten, da sich die Strukturen der Texte, die Wortwahl, Zeichensetzungen, usw. in den Sprachkategorien eher überlappen. Eine tiefergehende Analyse könnte hierzu Aufschluss geben.

Alle Plots, die Skripte zum Training und zur Evaluation der Modelle sind in einem Github-Repository[1] des Autors zu finden.

1 https://github.com/krupper/transformer-text-readability-classification

LITERATUR

Allison, Paul D. (7. Dez. 2001). *Logistic Regression Using the SAS: Theory and Application*. 1. Aufl. Cary, NC: Wiley-SAS. 307 S. ISBN: 978-0-471-22175-3.

Amstad, Toni (1978). *Wie verständlich sind unsere Zeitungen?* Studenten-Schreib-Service. 196 S. Google Books: kiI7vwEACAAJ.

Aumiller, Dennis und Michael Gertz (2022). "Klexikon: A German Dataset for Joint Summarization and Simplification". arXiv: 2201.07198. URL: https://arxiv.org/abs/2201.07198.

Bai, Shaojie, J. Zico Kolter und Vladlen Koltun (19. Apr. 2018). *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling*. DOI: 10.48550/arXiv.1803.01271. arXiv: 1803.01271 [cs]. URL: http://arxiv.org/abs/1803.01271 (besucht am 01. 09. 2022).

Balakrishna, Sowmya (1. Jan. 2015). "Analyzing Text Complexity and Text Simplification: Connecting Linguistics, Processing and Educational Applications". DOI: 10.15496/publikation-5781.

Bamberger, Richard (2006). *Erfolgreiche Leseerziehung. Theorie Und Praxis.* München: Domino.

Bamberger, Richard und Erich Vanecek (1984). *Lesen-Verstehen-Lernen-Schreiben: die Schwierigkeitsstufen von Texten in deutscher Sprache*. Jugend und Volk. 198 S. ISBN: 978-3-224-15250-2. Google Books: TElTAAAACAAJ.

Bjornsson, C. H, Stockholm und Pedagogiskt centrum (1968). *Lesbarkeit durch Lix*. Stockholm: Pedagogiskt Centrum.

Bock, Bettina M. (2019). *„Leichte Sprache" – kein Regelwerk: sprachwissenschaftliche Ergebnisse und Praxisempfehlungen aus dem LeiSA-Projekt*. Kommunikation – Partizipation – Inklusion Band 5. Berlin: Frank & Timme, Verlag für wissenschaftliche Literatur. 99 S. ISBN: 978-3-7329-0534-8.

Breiman, Leo (1. Okt. 2001). "Random Forests". In: *Machine Learning* 45.1, S. 5–32. ISSN: 1573-0565. DOI: 10.1023/A:1010933404324. URL: https://doi.org/10.1023/A:1010933404324 (besucht am 23. 08. 2022).

Brügelmann, Hans und Erika Brinkmann (2021). *Wie kann man erfassen, was Texte für echte Leseanfänger*innen leicht oder schwierig macht? Zur Begründung des "Bremer Erstlese-Index"(BRELIX)*. pedocs. 26 S. URL: http://nbn-resolving.de/urn:nbn:de:0111-pedocs-216680 (besucht am 18. 08. 2022).

Bundesamt, Statistisches (23. Apr. 2021). *Lebenslagen der behinderten Menschen - Ergebnis des Mikrozensus 2019*. Statistisches Bundesamt. URL: https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Gesundheit/Behinderte-Menschen/Publikationen/Downloads-Behinderte-Menschen/lebenslagen-behinderter-menschen-5122123199004.html (besucht am 23. 10. 2022).

Cañizares, Pablo C., Sara Pérez-Soler, Esther Guerra und Juan de Lara (25. Apr. 2022). "Automating the Measurement of Heterogeneous Chatbot Designs". In: *Proceedings of the 37th ACM/SIGAPP Symposium on Applied*

Computing. SAC '22. New York, NY, USA: Association for Computing Machinery, S. 1491–1498. ISBN: 978-1-4503-8713-2. DOI: 10.1145/3477314.3507255. URL: https://doi.org/10.1145/3477314.3507255 (besucht am 02. 09. 2022).

Chan, Branden, Stefan Schweter und Timo Möller (3. Dez. 2020). *German's Next Language Model*. arXiv: 2010.10906 [cs]. URL: http://arxiv.org/abs/2010.10906 (besucht am 19. 10. 2022).

Chang, Chih-Chung und Chih-Jen Lin (1. Aug. 2002). "Training V-Support Vector Regression: Theory and Algorithms". In: *Neural Computation* 14.8, S. 1959–1977. ISSN: 0899-7667. DOI: 10.1162/089976602760128081. URL: https://doi.org/10.1162/089976602760128081 (besucht am 23. 08. 2022).

Cho, Gi-Sung, Narangerel Gantulga und Yun-Woong Choi (Juli 2017). "A Comparative Study on Multi-Class SVM & Kernel Function for Land Cover Classification in a KOMPSAT-2 Image". In: *KSCE Journal of Civil Engineering* 21.5, S. 1894–1904. ISSN: 1226-7988, 1976-3808. DOI: 10.1007/s12205-016-1739-z. URL: http://link.springer.com/10.1007/s12205-016-1739-z (besucht am 20. 10. 2022).

Clark, Kevin, Minh-Thang Luong und Quoc V Le (2020). "ELECTRA: PRE-TRAINING TEXT ENCODERS AS DISCRIMINATORS RATHER THAN GENERATORS". In: S. 18.

Coleman, Meri und T. L. Liau (1975). "A Computer Readability Formula Designed for Machine Scoring". In: *Journal of Applied Psychology* 60.2, S. 283–284. ISSN: 1939-1854. DOI: 10.1037/h0076540.

Collins-Thompson, Kevyn (2014). "Computational Assessment of Text Readability: A Survey of Current and Future Research". In: S. 38.

Collins-Thompson, Kevyn und Jamie Callan (1. Jan. 2004). *Information Retrieval for Language Tutoring: An Overview of the REAP Project*, S. 545. 544 S. DOI: 10.1145/1008992.1009112.

Conneau, Alexis, Douwe Kiela, Holger Schwenk, Loïc Barrault und Antoine Bordes (Sep. 2017). "Supervised Learning of Universal Sentence Representations from Natural Language Inference Data". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2017. Copenhagen, Denmark: Association for Computational Linguistics, S. 670–680. DOI: 10.18653/v1/D17-1070. URL: https://aclanthology.org/D17-1070 (besucht am 01. 09. 2022).

Cortes, Corinna und Vladimir Vapnik (1. Sep. 1995). "Support-Vector Networks". In: *Machine Learning* 20.3, S. 273–297. ISSN: 1573-0565. DOI: 10.1007/BF00994018. URL: https://doi.org/10.1007/BF00994018 (besucht am 23. 08. 2022).

Dale, Edgar und Jeanne S. Chall (1948). "A Formula for Predicting Readability: Instructions". In: *Educational Research Bulletin* 27.2, S. 37–54. ISSN: 1555-4023. JSTOR: 1473669.

Demirkıran, Ferhat, Aykut Çayır, Uğur Ünal und Hasan Dağ (22. Juni 2022). *An Ensemble of Pre-trained Transformer Models For Imbalanced Multiclass Malware Classification*. arXiv: 2112.13236 [cs, stat]. URL: http://arxiv.org/abs/2112.13236 (besucht am 10. 10. 2022).

Deutsch, Tovly, Masoud Jasbi und Stuart Shieber (Juli 2020). "Linguistic Features for Readability Assessment". In: *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*. Seattle, WA, USA → Online: Association for Computational Linguistics, S. 1–17. DOI: 10.18653/v1/2020.bea-1.1. URL: https://aclanthology.org/2020.bea-1.1 (besucht am 11. 08. 2022).

Devlin, Jacob, Ming-Wei Chang, Kenton Lee und Kristina Toutanova (24. Mai 2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. DOI: 10.48550/arXiv.1810.04805. arXiv: 1810.04805 [cs]. URL: http://arxiv.org/abs/1810.04805 (besucht am 01. 09. 2022).

Dubay, William (1. Jan. 2004). "The Principles of Readability". In: *CA* 92627949, S. 631–3309.

Feng, Lijun (1. Jan. 2010). "Automatic Readability Assessment". In: *Dissertations, Theses, and Capstone Projects*. URL: https://academicworks.cuny.edu/gc_etds/1934.

Feng, Lijun, Noémie Elhadad und Matt Huenerfauth (30. März 2009). "Cognitively Motivated Features for Readability Assessment". In: *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. EACL '09. USA: Association for Computational Linguistics, S. 229–237.

Fernández, Alberto, Salvador García, Mikel Galar, Ronaldo C. Prati, Bartosz Krawczyk und Francisco Herrera (2018). *Learning from Imbalanced Data Sets*. New York, NY: Springer Science+Business Media. ISBN: 978-3-319-98073-7.

Flesch, Rudolph (1948). "A New Readability Yardstick". In: *Journal of Applied Psychology* 32.3, S. 221–233. ISSN: 1939-1854. DOI: 10.1037/h0057532.

Friederici, Angela D. (31. Juli 2015). "Sprache und Gehirn". In: *Sprache - Kognition - Kultur*. De Gruyter, S. 51–72. ISBN: 978-3-11-097055-5. DOI: 10.1515/9783110970555-005. URL: https://www.degruyter.com/document/doi/10.1515/9783110970555-005/html (besucht am 19. 08. 2022).

Friedman, Jerome (1. Feb. 2002). "Stochastic Gradient Boosting". In: *Computational Statistics & Data Analysis* 38, S. 367–378. DOI: 10.1016/S0167-9473(01)00065-2.

Friedrich, Marcus (Feb. 2020). "Buchbesprechung: Base-1 Method: A Structural-Functional Approach to Word, Sentence and Discourse Readability." In: *Zeitschrift für Pädagogische Psychologie* 34.1, S. 61–63. ISSN: 1010-0652. DOI: 10.1024/1010-0652/a000264. URL: https://econtent.hogrefe.com/doi/10.1024/1010-0652/a000264 (besucht am 04. 08. 2022).

Gasparetto, Andrea, Matteo Marcuzzo, Alessandro Zangari und Andrea Albarelli (Feb. 2022). "A Survey on Text Classification Algorithms: From Text to Predictions". In: *Information* 13.2 (2), S. 83. ISSN: 2078-2489. DOI: 10.3390/info13020083. URL: https://www.mdpi.com/2078-2489/13/2/83 (besucht am 19. 10. 2022).

Goldhahn, Dirk, Thomas Eckart und Uwe Quasthoff (2018). "Building Large Monolingual Dictionaries at the Leipzig Corpora Collection: From 100 to 200 Languages". In: S. 7.

Grotlüschen, Anke und Klaus Buddeberg, Hrsg. (2020). *LEO 2018: Leben mit geringer Literalität*. Bielefeld: wbv. 378 S. ISBN: 978-3-7639-6072-9 978-3-7639-6071-2.

Gunning, Robert (1952). *The Technique of Clear Writing*. McGraw-Hill. 316 S. ISBN: 978-7-00-001419-0. Google Books: ofI0AAAAMAAJ.

Hall, Mark, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann und Ian H. Witten (16. Nov. 2009). "The WEKA Data Mining Software: An Update". In: *ACM SIGKDD Explorations Newsletter* 11.1, S. 10–18. ISSN: 1931-0145. DOI: 10.1145/1656274.1656278. URL: https://doi.org/10.1145/1656274.1656278 (besucht am 10. 08. 2022).

Hancke, Julia, Sowmya Vajjala und Detmar Meurers (Dez. 2012). "Readability Classification for German Using Lexical, Syntactic, and Morphological Features". In: *Proceedings of COLING 2012*. COLING 2012. Mumbai, India: The COLING 2012 Organizing Committee, S. 1063–1080. URL: https://aclanthology.org/C12-1065 (besucht am 19. 08. 2022).

Hansen-Schirra, Silvia, Jean Nitzke und Silke Gutermuth (2021). "An Intralingual Parallel Corpus of Translations into German Easy Language (Geasy Corpus): What Sentence Alignments Can Tell Us About Translation Strategies in Intralingual Translation". In: *New Perspectives on Corpus Translation Studies*. Hrsg. von Vincent X. Wang, Lily Lim und Defeng Li. New Frontiers in Translation Studies. Singapore: Springer, S. 281–298. ISBN: 9789811649189. DOI: 10.1007/978-981-16-4918-9_11. URL: https://doi.org/10.1007/978-981-16-4918-9_11 (besucht am 14. 09. 2022).

Hartrumpf, Sven (2003). *Hybrid Disambiguation in Natural Language Analysis*. Osnabrück: Der Andere Verl. 205 S. ISBN: 978-3-89959-080-7.

Herzberg, Andy (12. Feb. 2019). "Analyse der oberflächlichen Merkmale von Qualitätsjournalismus-Texten". Thesis. Hochschule für angewandte Wissenschaften Hamburg. URL: https://reposit.haw-hamburg.de/handle/20.500.12738/8606 (besucht am 11. 08. 2022).

Hohenheim, Universität (o. D.). *Hohenheimer Verständlichkeitsindex: Klartext-Initiative*. URL: https://klartext.uni-hohenheim.de/hix (besucht am 24. 03. 2022).

Jablotschkin, Sarah und Heike Zinsmeister (30. Juni 2020). *LeiKo: A corpus of easy-to-read German*. DOI: 10.5281/zenodo.3923917. URL: https://zenodo.org/record/3923917 (besucht am 14. 09. 2022).

Jach, Daniel (13. Nov. 2020). *Korpus Einfaches Deutsch (KED)*. URL: https://daniel-jach.github.io/simple-german/simple-german.html.

Jahanshahi, Hadi, Syed Kazmi und Mucahit Cevik (29. Juni 2022). *Auto Response Generation in Online Medical Chat Services*. arXiv: 2104.12755 [cs]. URL: http://arxiv.org/abs/2104.12755 (besucht am 25. 07. 2022).

James, Gareth, Daniela Witten, Trevor Hastie und Robert Tibshirani, Hrsg. (2013). *An Introduction to Statistical Learning: With Applications in R*. Springer Texts in Statistics 103. New York: Springer. 426 S. ISBN: 978-1-4614-7137-0.

Kate, Rohit J, Xiaoqiang Luo, Siddharth Patwardhan, Martin Franz, Radu Florian, Raymond J Mooney, Salim Roukos und Chris Welty (Jan. 2010).

"Learning to Predict Readability Using Diverse Linguistic Features". In: S. 9.

Kaur, Harsurinder, Husanbir Singh Pannu und Avleen Kaur Malhi (30. Aug. 2019). "A Systematic Review on Imbalanced Data Challenges in Machine Learning: Applications and Solutions". In: *ACM Computing Surveys* 52.4, 79:1–79:36. ISSN: 0360-0300. DOI: 10.1145/3343440. URL: https://doi.org/10.1145/3343440 (besucht am 10. 10. 2022).

Kidwell, Paul, Guy Lebanon und Kevyn Collins-Thompson (1. Jan. 2009). *Statistical Estimation of Word Acquisition With Application to Readability Prediction*. Bd. 106, S. 909. 900 S. DOI: 10.1198/jasa.2010.ap09318.

Kim, Yoon (Okt. 2014). "Convolutional Neural Networks for Sentence Classification". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. EMNLP 2014. Doha, Qatar: Association for Computational Linguistics, S. 1746–1751. DOI: 10.3115/v1/D14-1181. URL: https://aclanthology.org/D14-1181 (besucht am 24. 08. 2022).

Kim, Yoon, Yacine Jernite, David Sontag und Alexander M. Rush (1. Dez. 2015). *Character-Aware Neural Language Models*. DOI: 10.48550/arXiv.1508.06615. arXiv: 1508.06615 [cs, stat]. URL: http://arxiv.org/abs/1508.06615 (besucht am 01. 09. 2022).

Kincaid, J., Robert Fishburne, Richard Rogers und Brad Chissom (1. Jan. 1975). "Derivation Of New Readability Formulas (Automated Readability Index, Fog Count And Flesch Reading Ease Formula) For Navy Enlisted Personnel". In: *Institute for Simulation and Training*. URL: https://stars.library.ucf.edu/istlibrary/56.

Krawczyk, Bartosz (1. Nov. 2016). "Learning from Imbalanced Data: Open Challenges and Future Directions". In: *Progress in Artificial Intelligence* 5.4, S. 221–232. ISSN: 2192-6360. DOI: 10.1007/s13748-016-0094-0. URL: https://doi.org/10.1007/s13748-016-0094-0 (besucht am 10. 10. 2022).

Langer, Inghard, Friedemann Schulz von Thun und Reinhard Tausch (2019). *Sich verständlich ausdrücken*. 11. Auflage. München: Ernst Reinhardt Verlag. 222 S. ISBN: 978-3-497-02532-9.

Likert, R. (1932). "A Technique for the Measurement of Attitudes". In: *Archives of Psychology* 22 140, S. 55–55.

Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer und Veselin Stoyanov (26. Juli 2019). *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. DOI: 10.48550/arXiv.1907.11692. arXiv: 1907.11692 [cs]. URL: http://arxiv.org/abs/1907.11692 (besucht am 19. 10. 2022).

Lu, Hongxia, Louis Ehwerhemuepha und Cyril Rakovski (2. Juli 2022). "A Comparative Study on Deep Learning Models for Text Classification of Unstructured Medical Notes with Various Levels of Class Imbalance". In: *BMC Medical Research Methodology* 22.1, S. 181. ISSN: 1471-2288. DOI: 10.1186/s12874-022-01665-y. URL: https://doi.org/10.1186/s12874-022-01665-y (besucht am 19. 10. 2022).

M. A.K. Halliday, Ruqaiya Hasan (1976). *Cohesion in English (English Language Series; No. 9)*. London, Uk: Longman. ISBN: 978-0-582-55041-4. URL: https:

//www.biblio.com/book/cohesion-english-english-language-series-9/d/1489055050 (besucht am 19. 08. 2022).

Maaß, Christiane (2015). *Leichte Sprache: das Regelbuch*. Barrierefreie Kommunikation 1. Münster: Lit. 184 S. ISBN: 978-3-643-12907-9.

– (2019). "Übersetzen in Leichte Sprache". Version 1. In: DOI: 10.25528/021. URL: https://hildok.bsz-bw.de/frontdoor/index/index/docId/988 (besucht am 23. 03. 2022).

Manning, Christopher D., Prabhakar Raghavan und Hinrich Schütze (7. Juli 2008). *Introduction to Information Retrieval*. Higher Education from Cambridge University Press. DOI: 10.1017/CBO9780511809071. URL: https://www.cambridge.org/highereducation/books/introduction-to-information-retrieval/669D108D20F556C5C30957D63B5AB65C (besucht am 10. 08. 2022).

Martinc, Matej, Senja Pollak und Marko Robnik-Šikonja (21. Apr. 2021). "Supervised and Unsupervised Neural Approaches to Text Readability". In: *Computational Linguistics* 47.1, S. 141–179. ISSN: 0891-2017. DOI: 10.1162/coli_a_00398. URL: https://doi.org/10.1162/coli_a_00398 (besucht am 11. 08. 2022).

McLaughlin, G. H. (1969). "SMOG Grading: A New Readability Formula". In: *Journal of Reading* 12.8, S. 639–646.

Mráz, David (o. D.). *Text Classification with Transformers in Tensorflow 2: BERT, XLNet*. Atheros.ai. URL: https://atheros.ai/blog/text-classification-with-transformers-in-tensorflow-2 (besucht am 20. 10. 2022).

Mühlenbock, Katarina Heimann (2013). *I See What You Mean: Assessing Readability for Specific Target Groups*. Göteborgs Universitet, Humanistika Fakulteten. 217 S. ISBN: 978-91-87850-50-9. Google Books: NdkrnQEACAAJ.

Naderi, Babak, Salar Mohtaj, Karan Karan und Sebastian Möller (Juni 2019). "Automated Text Readability Assessment for German Language: A Quality of Experience Approach". In: *2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX)*. 2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX), S. 1–3. DOI: 10.1109/QoMEX.2019.8743194.

Nguyen, Viet (2018). *[PYTORCH] Hierarchical Attention Networks for Document Classification*. URL: https://github.com/uvipen/Hierarchical-attention-networks-pytorch (besucht am 24. 08. 2022).

Nietzio, Annika, Daniel Naber und Christian Bühler (31. Dez. 2014). "Towards Techniques for Easy-to-Read Web Content". In: *Procedia Computer Science* 27, S. 343–349. DOI: 10.1016/j.procs.2014.02.038.

Pearson, P. David und Elfrieda H. Hiebert (Dez. 2014). "The State of the Field: Qualitative Analyses of Text Complexity". In: *The Elementary School Journal* 115.2, S. 161–183. ISSN: 0013-5984. DOI: 10.1086/678297. URL: https://www.journals.uchicago.edu/doi/abs/10.1086/678297 (besucht am 19. 08. 2022).

Radford, Alec, Karthik Narasimhan, Tim Salimans und Ilya Sutskever (2018). "Improving Language Understanding by Generative Pre-Training". In:

S. 12. URL: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

Richter, Florian, Philipp Koch, Oliver Franke, Jakob Kraus, Fabrizio Kuruc, Anja Thiem, Judith Högerl, Stella Heine und Konstantin Schöps (25. Mai 2021). *Open Discourse*. DOI: 10.7910/DVN/FIKIBO. URL: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/FIKIBO (besucht am 20. 07. 2022).

Rocca, Baptiste (27. Jan. 2019). *Handling Imbalanced Datasets in Machine Learning*. Medium. URL: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28 (besucht am 10. 10. 2022).

Rosenfeld, Avi, Sigal Sina, David Sarne, Or Avidov und Sarit Kraus (9. Feb. 2018). *A Study of WhatsApp Usage Patterns and Prediction Models without Message Content*. arXiv: 1802.03393 [cs]. URL: http://arxiv.org/abs/1802.03393 (besucht am 29. 07. 2022).

Sanh, Victor, Lysandre Debut, Julien Chaumond und Thomas Wolf (29. Feb. 2020). *DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter*. DOI: 10.48550/arXiv.1910.01108. arXiv: 1910.01108 [cs]. URL: http://arxiv.org/abs/1910.01108 (besucht am 19. 10. 2022).

Shah, Deven, H. Andrew Schwartz und Dirk Hovy (2020). "Predictive Biases in Natural Language Processing Models: A Conceptual Framework and Overview". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, S. 5248–5264. DOI: 10.18653/v1/2020.acl-main.468. arXiv: 1912.11078 [cs]. URL: http://arxiv.org/abs/1912.11078 (besucht am 19. 09. 2022).

Shaheen, Zein, Gerhard Wohlgenannt und Erwin Filtz (24. Okt. 2020). *Large Scale Legal Text Classification Using Transformer Models*. arXiv: 2010.12871 [cs]. URL: http://arxiv.org/abs/2010.12871 (besucht am 20. 10. 2022).

Shaikh, Sarang, Sher Muhammad Daudpota, Ali Shariq Imran und Zenun Kastrati (Jan. 2021). "Towards Improved Classification Accuracy on Highly Imbalanced Text Dataset Using Deep Neural Language Models". In: *Applied Sciences* 11.2 (2), S. 869. ISSN: 2076-3417. DOI: 10.3390/app11020869. URL: https://www.mdpi.com/2076-3417/11/2/869 (besucht am 06. 09. 2022).

Si, Luo und Jamie Callan (5. Okt. 2001). "A Statistical Model for Scientific Readability". In: *Proceedings of the Tenth International Conference on Information and Knowledge Management*. CIKM '01. New York, NY, USA: Association for Computing Machinery, S. 574–576. ISBN: 978-1-58113-436-0. DOI: 10.1145/502585.502695. URL: https://doi.org/10.1145/502585.502695 (besucht am 10. 08. 2022).

Spring, Nicolas, Annette Rios Gonzales und Sarah Ebling (Sep. 2021). "Exploring German Multi-Level Text Simplification". In: Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021), S. 1339–1349. URL: https://aclanthology.org/2021.ranlp-1.150 (besucht am 23. 03. 2022).

Spring, Nicolas, Annette Rios und Sarah Ebling (1. Sep. 2021). *LHA Sentence Alignments Extracted From the Austria Press Agency Corpus*. DOI: 10.5281/zenodo.5148163. URL: https://zenodo.org/record/5148163 (besucht am 20. 07. 2022).

Stephan, Alea (2014). "Leichte Sprache und der Übersetzungsaspekt - Lassen sich Fachtexte in Leichte Sprache übersetzen? Ist ein Leichte-Sprache-Text überhaupt eine Übersetzung?" Stiftung Universität Hildesheim. URL: https://hildok.bsz-bw.de/frontdoor/index/index/docId/243 (besucht am 14. 09. 2022).

Vajjala, Sowmya und Detmar Meurers (Juni 2012). "On Improving the Accuracy of Readability Classification Using Insights from Second Language Acquisition". In: *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*. Montréal, Canada: Association for Computational Linguistics, S. 163–173. URL: https://aclanthology.org/W12-2019 (besucht am 24. 08. 2022).

ValizadehAslani, Taha, Yiwen Shi, Jing Wang, Ping Ren, Yi Zhang, Meng Hu, Liang Zhao und Hualou Liang (21. Juli 2022). *Two-Stage Fine-Tuning: A Novel Strategy for Learning Class-Imbalanced Data*. DOI: 10.48550/arXiv.2207.10858. arXiv: 2207.10858 [cs]. URL: http://arxiv.org/abs/2207.10858 (besucht am 10. 10. 2022).

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser und Illia Polosukhin (5. Dez. 2017). *Attention Is All You Need*. DOI: 10.48550/arXiv.1706.03762. arXiv: 1706.03762 [cs]. URL: http://arxiv.org/abs/1706.03762 (besucht am 19. 10. 2022).

Vor der Brück, Tim, Sven Hartrumpf und Hermann Helbig (1. Jan. 2008). *A Readability Checker with Supervised Learning Using Deep Syntactic and Semantic Indicators*.

Vor der Brück, Tim und Johannes Leveling (1. Jan. 2007). *Parameter Learning for a Readability Checking Tool*. S. 153. 149 S.

Weiß, Zarah und Detmar Meurers (Aug. 2018). "Modeling the Readability of German Targeting Adults and Children: An Empirically Broad Analysis and Its Cross-Corpus Validation". In: *Proceedings of the 27th International Conference on Computational Linguistics*. COLING 2018. Santa Fe, New Mexico, USA: Association for Computational Linguistics, S. 303–317. URL: https://aclanthology.org/C18-1026 (besucht am 10. 08. 2022).

Wild, Johannes und Johannes Pissarek (2020). *Ratte. Regensburger Analysetool Für Texte. Version 1.6.1*. URL: https://www.uni-regensburg.de/sprache-literatur-kultur/germanistik-did/downloads/ratte/index.html (besucht am 17. 08. 2022).

Xia, Menglin, Ekaterina Kochmar und Ted Briscoe (Juni 2016). "Text Readability Assessment for Second Language Learners". In: *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*. San Diego, CA: Association for Computational Linguistics, S. 12–22. DOI: 10.18653/v1/W16-0502. URL: https://aclanthology.org/W16-0502 (besucht am 24. 08. 2022).

Yu, Hsiang-Fu (o. D.). *Taming Transformers for Text Classification with Millions of Classes*. Amazon Science. URL: https://www.amazon.science/blog/natural-language-processing-techniques-text-classification-with-Transformers-at-scale (besucht am 20. 10. 2022).