THM | CAMPUS GIESSEN | MNI
TECHNISCHE HOCHSCHULE MITTELHESSEN
Mathematik, Naturwissenschaften und Informatik

Bachelorthesis

Konzeption und Realisierung eines Modells zur Multi-Label-Textklassifikation und Named Entity Recognition unter Verwendung von künstlicher Intelligenz

zur Erlangung des akademischen Grades

Bachelor of Science

eingereicht im Fachbereich Mathematik, Naturwissenschaften und Informatik an der Technischen Hochschule Mittelhessen

von

Paul David Hiller

27. November 2023

Referent: Dr. Dennis Priefer

Korreferent: Prof. Dr. Peter Kneisel

Erklärung der Selbstständigkeit

Hiermit versichere ich, die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie die Zitate deutlich kenntlich gemacht zu haben.

Gießen, den 27. November 2023 Paul David Hiller

Gender-Disclaimer

„In dieser Arbeit wird aus Gründen der besseren Lesbarkeit das generische Maskulinum verwendet. Weibliche und anderweitige Geschlechteridentitäten werden dabei ausdrücklich mitgemeint, soweit es für die Aussage erforderlich ist.“

The integration of artificial intelligence into business processes is a crucial step for companies to survive in a competitive environment. Text processing is one of the most important applications of artificial intelligence in a growing sector. This thesis focuses on text processing through multi-label text classification and named entity recognition. The aim is to investigate how multi-label text classification and named entity recognition can be applied, implemented and evaluated using artificial intelligence. To this end, the basics of neural networks in the context of multi-label text classification and named entity recognition as well as the associated metrics are first explained. With the help of a quantitative research approach and a structured literature review, the current state of research is identified. Based on this, a neural network consisting of a BERT and an ELMo encoder, a bidirectional long short-term memory and conditional random fields for named entity recognition as well as a neural network based on the universal sentence encoder with a bidirectional long short-term memory, a fully connected layer and individual heads for classifying the text into several labels are implemented, merged into one system and evaluated. The metrics and methods identified within the structured literature research are summarised in an evaluation concept. This is used to evaluate the realised models. On a Reuters 21578 dataset reduced to 20 labels, micro and macro F_1 scores of 73% and 56% respectively were achieved for the classification of texts with multiple labels and 94% and 93% respectively for the recognition of named entities on the CoNLL03 dataset.

Inhaltsverzeichnis

1 Einführung

> We have seen AI providing conversation and comfort to the lonely; we have also seen AI engaging in racial discrimination. Yet the biggest harm that AI is likely to do to individuals in the short term is job displacement, as the amount of work we can automate with AI is vastly larger than before. As leaders, it is incumbent on all of us to make sure we are building a world in which every individual has an opportunity to thrive [Ng16].

Im Rahmen, der derzeit stattfindenden vierten industriellen Revolution beginnt die künstliche Intelligenz (KI) einen Mehrwert zu bringen. Dabei hat die Nutzung dieser Technologie Einfluss auf Bereiche wie Bildung, Finanzen, Nachhaltigkeit und Armutsbekämpfung [Mhl21]. Begriffe wie künstliche Intelligenz, maschinelles Lernen und Deep Learning gewinnen in vielen Bereichen mit hohen Genauigkeitsanforderungen zunehmend an Bedeutung. Zu diesen Bereichen gehören unter anderem Medizin, Militär, Wirtschaft und die Industrie. Es wird erwartet, dass KI in Zukunft eine zentrale Rolle spielen und für viele Menschen zu einem unverzichtbaren Werkzeug im Alltag wird [Agg22].

Aktuelle wissenschaftliche Arbeiten beschäftigen sich mit innovativen Ansätzen zur Optimierung der Text- und Satzklassifikation sowie der Erkennung benannter Entitäten. Dazu gehören neuartige Architekturen, die Kombination verschiedener Methoden und fortgeschrittene Datenverarbeitungsstrategien [Zha23, Gan23, Ste22]. Techniken wie die Extreme-Multi-Label Text Klassifizierung können in vielen Anwendungsbereichen wie der Produktbeschriftung oder Kennzeichnung von Dokumenten gefunden werden [Xun20]. Darüber hinaus ist ein deutlicher Anstieg der wissenschaftlichen Publikationen zum Thema KI zu verzeichnen. Während im Jahr 2010 insgesamt 162.444 Arbeiten veröffentlicht wurden, stieg diese Zahl bis 2021 auf 333.497 Publikationen an, was mehr als einer Verdoppelung entspricht [Zha22a]. Die Aktualität und Vielzahl der wissenschaftlichen Publikationen auf diesem Gebiet zeigen ein wachsendes Interesse und einen Bedarf an verbesserten Techniken in diesem Bereich.

1.1 Motivation und Problembeschreibung

Die Integration von KI-Technologien in Geschäftsprozessen ist ein wesentlicher Anpassungsschritt für viele Unternehmen, um im aktuellen technologischen Wandel wettbewerbsfähig zu bleiben [Fer21]. Eine Studie mit 606 befragten Unternehmen in Deutschland zeigt eine erhebliche Diskrepanz zwischen den bisher getätigten und den geplanten Investitionen in KI. Während im Jahr 2021 nur 5% der Unternehmen in künstliche Intelligenz investierten, stieg dieser Anteil im Jahr 2022 auf 6%. Für das Jahr 2023 wird ein Anstieg auf 10% prognostiziert und in der Zukunftsplanung beabsichtigen 20% der befragten Unternehmen entsprechende Investitionen [Bit22]. Einer Prognose zufolge wird das durchschnittliche jährliche Wachstum des KI-Sektors in der US-Wirtschaft zwischen 2023 und 2028 bei 33,6% liegen. Basierend auf diesen Schätzungen wird erwartet, dass der globale Markt, der 2021 einen Wert von 142,2 Milliarden US-Dollar hatte, bis 2028 einen Wert von 1847,5 Milliarden US-Dollar erreichen wird [Con23]. Dies deutet auf eine erhebliche Zunahme der Anwendung und Integration von künstlicher Intelligenz hin. In einer weltweiten Studie wurde die Adaptionsrate verschiedener KI-Funktionalitäten in Unternehmen, die mindestens eine Form von KI-Technologie einsetzen, untersucht. Die Analyse ergab, dass die Fähigkeit, Texte in natürlicher Sprache zu interpretieren, mit einer Rate von 33% an dritter Stelle steht [McK23]. Dies unterstreicht die wachsende Bedeutung von Textverarbeitungstechnologien in der Wirtschaft.

Auch der an die mbi GmbH herangetragene Kundenwunsch, KI-Funktionalitäten in Softwarelösungen zu implementieren, unterstreicht den Bedarf und die Aktualität der Technologie. Dabei wurde vor allem der Wunsch geäußert, die automatisierte Verarbeitung von Dokumenten mit textuellem Inhalt zu ermöglichen. Daraus ergibt sich für die mbi GmbH die Notwendigkeit, Kompetenzen und Expertise in diesem Bereich aufzubauen. Ein zentrales Anliegen ist dabei die Identifikation und Extraktion spezifischer Text- und Dokumentbestandteile, insbesondere die von benannten Entitäten.

1.2 Ziele dieser Arbeit

Ziel der Arbeit ist die Konzeption und Implementierung eines künstlich-intelligenten Modells zur Multi-Label-Klassifikation von Texten, im englischen als „Multi-Label Text Classification“ (MLTC) bezeichnet, und zur Identifikation von benannten Entitäten, englisch als „Named Entity Recognition“ (NER) benannt, innerhalb dieser Texte. Darüber hinaus sollen mit Hilfe dieser Bachelorarbeit erste fundierte Fachkompetenzen und Expertisen innerhalb der mbi GmbH etabliert werden, um den Kundenanforderungen nach KI-Funktionalitäten adäquat begegnen zu können. Dieses Modell soll in der Lage sein, Texte auf der Grundlage spezifischer, vordefinierter Begriffe präzise zu klassifizieren.

Besonderer Wert wird auf die praktische Anwendbarkeit des Modells gelegt. Der entworfene Prototyp soll auf dem erarbeiteten Konzept und den darin definierten Funktionen basieren. Vor diesem Hintergrund ergeben sich die zentralen Forschungsfragen:

FF-1 Wie können Texte mittels künstlicher Intelligenz durch Multi-Label-Klassifikation kategorisiert und benannte Entitäten identifiziert werden?

FF-2 Wie kann eine auf künstlicher Intelligenz basierende Lösung zur Multi-Label-Klassifikation und Erkennung von benannten Entitäten in Texten implementiert werden?

Ein weiteres Ziel dieser Arbeit ist die Entwicklung eines modellspezifischen Evaluationskonzeptes. Es soll die Möglichkeit bieten, sowohl bereits erstellte als auch zukünftige Modelle mit dem im Rahmen dieser Arbeit entwickelten Modell anhand von hauptsächlich quantitativen, aber auch qualitativen Metriken zu vergleichen. Dabei soll das Evaluationskonzept insbesondere auf Metriken ausgerichtet sein, die die Genauigkeit im Zusammenhang mit MLTC und NER messen. Dies führt zur abschließenden Forschungsfrage:

FF-3 Welche Metriken und Methoden sind geeignet, um eine auf künstlicher Intelligenz basierende Lösung zur Multi-Label-Klassifikation und Erkennung von benannten Entitäten in Texten zu evaluieren?

1.3 Methodische Vorgehensweise

Für die Ausarbeitung wird ein quantitativer Forschungsansatz gewählt, der sich auf die Entwicklung und Implementierung eines KI-Modells für die MLTC und die NER in Texten konzentriert. Die Wahl dieses Ansatzes basiert auf der Erkenntnis, dass quantitative Methoden eine objektive und messbare Bewertung der Modellleistung ermöglichen. Die Arbeit beginnt mit einer umfassenden Einführung in die Grundlagen der KI, insbesondere im Kontext der MLTC und der NER. Diese Einführung dient dazu, den aktuellen Stand der Forschung darzustellen und eine klare Abgrenzung zu verwandten Technologien und Methoden zu schaffen.

Um eine umfassende und vertiefende Darstellung des aktuellen Zustands im Kontext dieser wissenschaftlichen Arbeit zu gewährleisten, wird eine strukturierte Literaturrecherche durchgeführt. Ziel dieser Recherche ist es, aktuelle wissenschaftliche Erkenntnisse und Technologien im Bereich der MLTC und der NER zu identifizieren und zu analysieren.

Die aus der Literaturrecherche gewonnenen Erkenntnisse bilden die Grundlage für die anschließende konzeptionelle Entwicklung des KI-Modells. Die Erkenntnisse stellen dabei

Erfolge aus diversen wissenschaftlichen Publikationen dar und bilden die Grundlage zur Auswahl einer adäquaten KI-Architektur. Die Erstellung des Konzeptes zur Implementierung stellt hierbei den Soll-Zustand dar und dient zur Beantwortung von *FF-1*. Des Weiteren werden die genutzten Metriken zur Evaluierung im Bereich von MLTC und NER genutzt, um ein Evaluationskonzept zur Beantwortung von *FF-3* aufzustellen. Das Konzept wird dabei so ausgearbeitet, dass es für einen aussagekräftigen Vergleich des entwickelten Modells mit anderen Ansätzen im Bereich der MLTC und NER genutzt werden kann.

Für die eigentliche Implementierung des Modells zur Beantwortung von *FF-2* wird zunächst ein für den Anwendungszweck geeigneter Datensatz identifiziert. Der Datensatz sollte dabei möglichst häufiger für Modelle mit ähnlichen oder gleichen Funktionalitäten verwendet worden sein, um eine erste Vergleichsgrundlage zu schaffen. Zudem sollte der Datensatz öffentlich zugänglich sein, um eine möglichst ideale Reproduzierbarkeit des Ergebnisses der Arbeit zu gewährleisten. Dieser wird im nächsten Schritt bereinigt, um eine möglichst optimale Trainingsgrundlage für das Modell zu schaffen. Das aufgestellte Evaluationskonzept wird an der realisierten Lösung weitestgehend angewandt um die Anwendbarkeit der Metriken im Sinne von *FF-3* zu demonstrieren.

In dieser Arbeit wird das KI-Modell (Abbildung 1.1) mittels eines etablierten Frameworks implementiert und mit dem aufbereiteten Datensatz trainiert. Der Fokus liegt dabei auf der Architektur der MLTC- und NER-Lösung, um möglichst präzise Ergebnisse zu erhalten. Dabei wird darauf geachtet, dass die im Rahmen dieser Arbeit entwickelte Lösung anpassungsfähig ist und effizient in ein produktionsrelevantes System integriert werden kann.

Abschließend werden die Ergebnisse in ihrer Gesamtheit betrachtet und diskutiert. Dabei wird nicht nur die Leistungsfähigkeit des entwickelten Modells analysiert, sondern auch kritisch reflektiert. Die Grenzen des gewählten Ansatzes werden ebenso aufgezeigt wie Möglichkeiten zur Weiterführung der vorliegenden Arbeit.

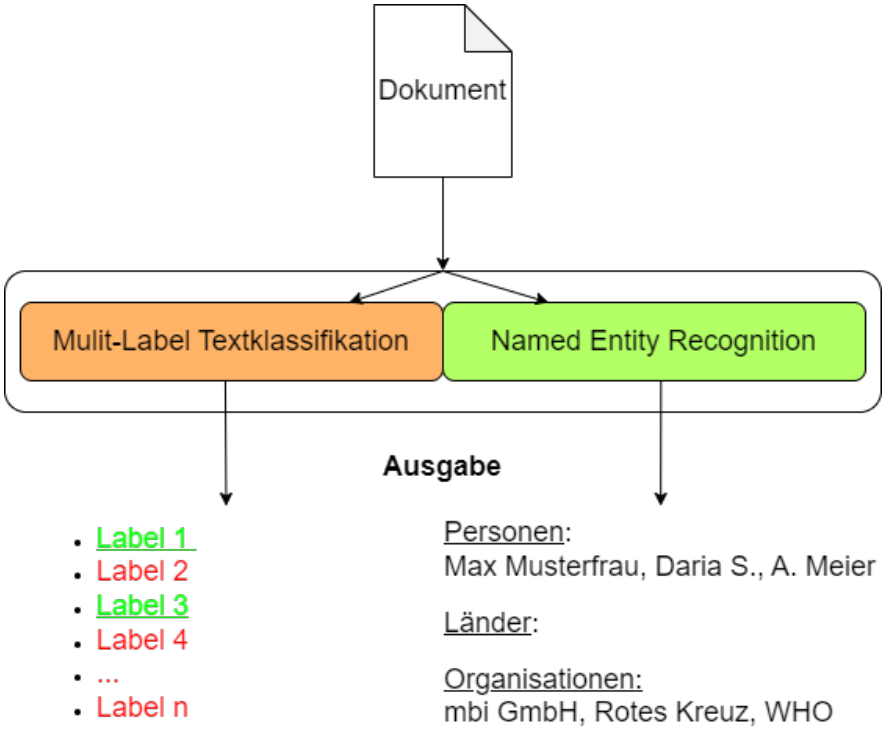

Abbildung 1.1: Vereinfachte Darstellung eines Systems mit einer NER- und einer MLTC-Komponente. Das Dokument wird von diesem System verarbeitet. Zugewiesene Labels sind grün dargestellt, nicht zugewiesene Labels sind rot markiert. Als Entitäten können Personen, Länder und Organisationen identifiziert werden. Es wurden drei Personen und drei Organisationen gefunden, jedoch kein Land.

1.4 Abgrenzung

Angesichts des gegebenen Umfangs der Arbeit sind für die strukturierte Literaturrecherche einige Anpassungen nötig, wodurch die Tiefe der Analyse beeinträchtigt wird. Der Schwerpunkt der Analyse soll dabei auf dem Veröffentlichungszeitraum und damit der Aktualität der Literatur liegen. Wissenschaftliche Arbeiten, die vor dem Jahr 2015 erschienen sind, werden nicht weiter berücksichtigt. Zudem werden lediglich drei Schlüsselwörter in verschiedenen Kombinationen für die Literatursuche verwendet. Die ausgewählten Schlüsselwörter sind dabei für das Thema zielführend, jedoch generisch genug, um dennoch ein möglichst weites Spektrum abdecken zu können. Falls die Anzahl an gefundenen Publikationen zu weitläufig ist und über 20.000 Ergebnisse gefunden werden, wird die Suche auf die Subdisziplin „künstliche Intelligenz“ eingegrenzt oder lediglich Ergebnisse berücksichtigt, in denen die Schlüsselwörter innerhalb der Titel gefunden werden. Andere Programmiersprachen oder Datenbank-Managementsysteme für die Erstellung des Graphen zur Darstellung relevanter Publikationen, werden aufgrund der fehlenden Relevanz für diese Ausarbeitung nicht in Betracht gezogen.

Für die Auswahl des Datensatzes ist der Inhalt oder das Thema der Texte selbst irrelevant. Wichtig für die Auswahl sind nur die bereits in dem Kapitel der methodischen Vorgehensweise beschriebenen Kriterien. Weitergehend soll im Rahmen dieser Arbeit nicht auf die Methoden oder andere Faktoren für die Erstellung oder Beschaffung eines Datensatzes eingegangen werden. Diese Abgrenzung schließt allerdings nicht die kritische Betrachtung der Eignung des Datensatzes für das zu implementierende Modell mit ein.

Strategien oder Maßnahmen zur Optimierung des Modells, die neben den identifizierten Maßnahmen und Strategien ebenfalls nützlich erscheinen, aber deren Anwendung den Umfang der Arbeit überschreiten, werden dem Maß der Literaturrecherche entsprechend erwähnt, jedoch nicht weiter diskutiert oder verglichen. Vergleiche oder Diskussionen werden nur dann geführt, wenn diese notwendig sind, um die Auswahl einer bestimmten Maßnahme oder Strategie zu begründen. Ein Vergleich oder eine Diskussion über die Wahl eines passenden Frameworks oder Programmiersprache wird nicht durchgeführt. Als Implementierung wird in diesem Kontext die reine Erstellung des Modells verstanden und nicht die Einbindung in ein bestehendes System oder die Erstellung eines Moduls zur Einbettung des Modells. Weiterführende mögliche Schritte, um das erstellte Modell in einer produktionsähnlichen Umgebung zu verwenden, werden im abschließenden Kapitel Ausblick grob benannt.

1.5 Struktur der Arbeit

Um den Lesern einen umfassenden Überblick über die Struktur der Arbeit geben zu können, ist die vorliegende Bachelorarbeit in folgende Kapitel untergliedert. Zunächst werden in Kapitel 2 Hintergrund alle nötigen Konzepte und Grundlagen für das Verstehen der Arbeit dargelegt. Der aktuelle Forschungsstand zu den Aspekten der Multi-Label-Klassifikation und der Identifizierung innerhalb eines Textes benannter Entitäten, die für die Arbeit aufgestellten Anforderungen und das daraus resultierende Konzept werden im Kapitel 3 ausführlich dargestellt und mit dem Aufstellen eines Evalutionskonzeptes abgeschlossen. Der darauf folgende Abschnitt, die Realisierung 4, umfasst die gesamte Umsetzung des zu entwickelnden Systems. Dies beinhaltet zum einen die Datenaufbereitung und Implementierung und zum anderen die gesamte Phase des Trainings und der Evaluierung der Modelle. Abschließend werden in diesem Kapitel die im vorherigen Abschnitt aufgestellten Anforderung verifiziert. In dem letzten Kapitel 5, der Zusammenfassung, wird ein abschließendes Fazit zu den Erkenntnissen dieser Arbeit gegeben und diese zudem kritisch reflektiert. Zuletzt werden weitere Ansätze zu der Beantwortung der aufgestellten Forschungsfragen vorgeschlagen und die nächsten Schritte für die Resultate der Arbeit vorgestellt.

2 Hintergrund

Ziel dieses Kapitels ist es, den Leser mit den grundlegenden Konzepten vertraut zu machen, die für das Verständnis der gesamten Arbeit notwendig sind. Als erstes wird in Abschnitt 2.1 das Unternehmen mbi GmbH vorgestellt. Für den Rahmen der Arbeit, wird zunächst ein komprimierter Überblick über das Gebiet der Künstlichen Intelligenz gegeben (siehe Abschnitt 2.2), wobei der Schwerpunkt auf den für diese Arbeit relevanten Aspekten liegt. Für alle weiteren Abschnitte kann KI und die beschriebenen Inhalte als übergeordneter Begriff angesehen werden. Im anschließenden Abschnitt 2.3 wird eine Einführung in neuronale Netze gegeben. Anzumerken ist, dass die Darstellung nicht den Anspruch erhebt, alle Bereiche neuronaler Netze oder die hier behandelten Aspekte in ihrer gesamten Komplexität abzudecken. Die Tiefe der Diskussion ist jedoch ausreichend, um ein Grundverständnis für die nachfolgenden Kapitel und Erläuterungen zu gewährleisten. Im Abschnitt 2.4 wird die Verarbeitung von Texten dargestellt. Die Abschnitte 2.5 und 2.6 erläutern Multi-Label Textklassifikation (MLTC) und Named Entity Recognition (NER) als Disziplin der Verarbeitung natürlicher Sprache. Abschließend werden im Abschnitt 2.7 verschiedene Metriken zur Beurteilung der Leistung eines MLTC- und NER-Modells vorgestellt.

2.1 mbi GmbH

Die mbi GmbH wurde in ihrer heutigen Form 1993 gegründet und beschäftigt derzeit über 50 Mitarbeiter. Das Unternehmen ist in den Bereichen Softwareentwicklung und IT-Beratung tätig. Die Geschäftsfelder der mbi GmbH bestehen zum einen aus der Entwicklung und dem Vertrieb der eigenen Software WINPACCS, die der Digitalisierung von Geschäftsprozessen in der Entwicklungszusammenarbeit dient. Zum anderen bietet der Geschäftsbereich Digitale Plattformen kundenorientierte individuelle Softwarelösungen an. Die Softwarelösungen umfassen in diesem Zusammenhang unter anderem Geschäfts-Webanwendungen, Backendlösungen und Desktop-basierte Client-Serversysteme [Gmb23]. Die Erstellung dieser Arbeit wird durch die mbi GmbH gefördert und hat zum Ziel, fundiertes Wissen im Bereich künstlicher Intelligenz innerhalb des Unternehmens aufzubauen und die Basis für den Einsatz von MLTC- und NER-Systemen in verschiedenen Anwendungsgebieten zu schaffen.

2.2 Künstliche Intelligenz (KI)

Innerhalb der Informatik wird die Erforschung der KI als Studium intelligenter Agenten interpretiert. Intelligente Agenten verändern ihre Handlung auf Grund ihrer Umwelt und maximieren somit ihren Erfolg im Rahmen einer Aufgabenstellung. Dies bildet den Kontext für die Interpretation von KI in der vorliegenden Arbeit. Umgangssprachlich wird KI als Nachahmung menschlicher kognitiver Fähigkeiten durch Maschinen verstanden. Eine tatsächlich generelle künstliche Intelligenz gibt es derzeit noch nicht. Der derzeitge Stand der KI umfasst das Verstehen menschlicher Sprache, das Spielen von Computer- und Brettspielen, selbstfahrende Autos und komplexe Simulationen [Ong17]. Von besonderer Relevanz sind dabei die Teildisziplinen **„Natural Language Processing“ (NLP)** und **Neuronales Rechnen**. Während sich NLP mit der maschinellen Interpretation und Generierung menschlicher Sprache beschäftigt, konzentriert sich das Neuronale Rechnen auf die Entwicklung und Anwendung künstlicher neuronaler Netze [Pan15]. Andere Bereiche der KI, wie z.B. die Robotik, sind für den Kontext dieser Arbeit nicht relevant. Spezifischer betrachtet lassen sich MLTC und **d!** (**d!**)er Informationsextraktion zuordnen, die wiederum eine Subdisziplin der Verarbeitung natürlicher Sprache (NLP) ist [Cho20]. Neuronale Netze bieten die algorithmische Grundlage, um komplexe Muster in großen Datenmengen zu erkennen, was für die Informationsextraktion aus Texten essentiell ist. Durch den Einsatz von spezialisierten Architekturen können neuronale Netze effizient Textdaten analysieren und relevante Informationen extrahieren [Mar05]. Diese Techniken ermöglichen es, Aufgaben wie MLTC und NER effizient und präzise durchzuführen [Che17, Lam16].

Für die Implementierung von KI-Systemen werden gegenwärtig zwei Hauptarten des Lernens verwendet: Maschinelles Lernen und Deep Learning. Maschinelles Lernen kann in vier Hauptkategorien unterteilt werden: überwachtes, unüberwachtes, teilüberwachtes und verstärktes Lernen. Deep Learning hingegen befasst sich mit neuronalen Netzen, die mehrere verborgene Schichten aufweisen, und den zugehörigen Lernalgorithmen [Ong17]. In dieser Arbeit wird der Fokus auf Deep Learning gelegt, während maschinelles Lernen eine untergeordnete Rolle spielt.

2.3 Einführung in Neuronale Netze

Künstliche neuronale Netze sind Rechenmodelle, die von der Struktur und Funktionsweise biologischer neuronaler Netze innerhalb des menschlichen Gehirns inspiriert sind. Im menschlichen Gehirn sind Milliarden von Neuronen miteinander verknüpft und verarbeiten Informationen in einer parallelen Struktur. Künstliche neuronale Netze versuchen, diese komplexe Architektur und Informationsverarbeitung theoretisch mathematisch

und praktisch in Form von Software nachzubilden [Wan03]. Im Rahmen dieser Arbeit werden künstliche neuronale Netze fortan als **Neuronales Netz (NN)** bezeichnet.

Feedforward neuronale Netze (FFNN) NN bestehen aus einer Eingabeschicht, einer oder mehreren versteckten Schichten und einer Ausgabeschicht. Die Schichten enthalten Neuronen (auch Knoten oder Einheiten genannt), die über Verbindungen miteinander verknüpft sind, wobei jeder Knoten mit allen Knoten der nächsten Schicht verbunden ist. Dieser grundlegende Aufbau wird als FFNN bezeichnet (siehe Abbildung 2.1) [Wan03].

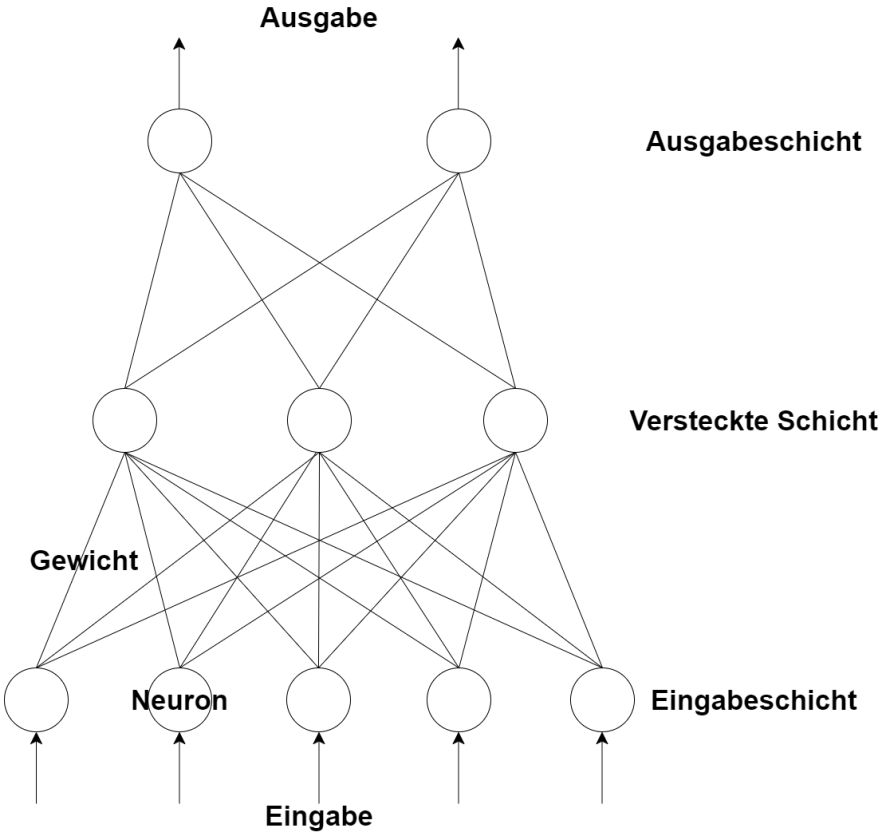

Abbildung 2.1: Feedforward neuronales Netz nach [Wan03]

Training Um korrekte und genaue Gewichtungen zu erzielen, wird das NN trainiert. Ein Datensatz oder ein Teil davon wird an die Eingabeschicht des NN angepasst und verarbeitet. Für jede Dateneinheit muss das korrekte Ergebnis bekannt sein, um es mit dem NN-Ergebnis zu vergleichen. Das Ziel ist es, die **Fehlerfunktion** (auch „Loss“-Funktion genannt) zu minimieren, die normalerweise aus der quadrierten Summe der Differenzen zwischen der Ausgabe und den bekannten Ergebnissen besteht. Die Gewichtsanpassungen erfolgen zyklisch [Wan03].

Rekurrente neuronale Netze (RNN) Die Verwendung von RNNs ist vorteilhaft für die Verarbeitung nichtlinearer Beziehungen in zeitlich veränderlichen Daten und

erweitern das traditionelle FFNN [Sus13, Che19]. RNNs ermöglichen komplexe zeitliche Abhängigkeiten und werden in Anwendungen wie Aktienprognosen oder Textverarbeitung eingesetzt. Im Vergleich zu FFNNs verfügen RNNs über Rückkopplungsschleifen, die Ergebnisse aus versteckten Schichten und der Ausgabeschicht in die Eingabeschicht zurückführen. Dies ermöglicht variable Ergebnisse je nach vorheriger Eingabe [Dem96]. Die Eingabevektoren $x_t \quad \{x_t, t = 1,2,...t_c\}$ werden einzeln in das RNN eingespeist, und das RNN kann Informationen aus allen vorherigen Zeitschritten bis zum aktuellen Zeitschritt t_c verwenden. Zukünftige Informationen können durch eine Verzögerung der Ausgabe um M Schritte einbezogen werden. y_{t_c} wird somit für die Eingabe x_{t+M} errechnet. Praktisch angewendet darf M nicht zu groß sein, da sonst die Qualität des Ergebnisses sinkt [Sch97].

RNNs haben längere Trainingszeiten aufgrund zahlreicher Durchläufe und sind anfällig für das **"Vanishing Gradient-Problem"(VGP)**, bei dem Gradienten der Fehlerfunktion in den unteren Schichten exponentiell abnehmen, wenn mehr Zeitschritte einbezogen werden. Dies ist auf die verwendeten Algorithmen zurückzuführen und führt zu minimalen Anpassungen der Gewichtungen und verlängert die Trainingszeit [Dem96, Hoc98, Hoc91].

Bidirektionale rekurrente neuronale Netze (BiRNN) BiRNNs nutzen jeweils getrennte Neuronengruppen in versteckten Schichten für positive und negative Zeitrichtungen, um sowohl vergangene als auch zukünftige Informationen für die Berechnung von y_{t_c} einzubeziehen ohne die Notwendigkeit von M Verzögerungsschritten. Dabei sind die Ausgänge der Neuronen nicht mit den Eingängen der Neuronen der anderen Zeitrichtung verbunden [Sch97].

Long Short-Term Memory (LSTM) LSTMs sind eine 1997 von Hochreiter und Schmidhuber vorgestellte erweiterte Form von RNNs, die dazu entwickelt wurde das VGP zu lösen und komplexe Abhängigkeiten in sequenziellen Daten zu erfassen. In einem LSTM werden Speicherzellen (siehe Abbildung 2.2 hinzugefügt, in denen relevante Abhängigkeiten gespeichert werden. Dabei werden ein „input gate“-Mechanismus und ein „output gate“-Mechanismus um ein lineares Neuron, welches als Speicher fungiert, konstruiert. Diese drei zusammengeführten Einheiten sind in der versteckten Schicht des LSTMs wiederzufinden. Der „input gate“-Mechanismus sorgt dabei dafür, dass keine irrelevanten Eingaben gespeichert werden. Gleichzeitig sorgt der „output gate“-Mechanismus dafür, dass irrelevant gewordene Informationen nicht an andere Einheiten weitergegeben werden. Da Eingaben aus mehreren Teilen oftmals schwer innerhalb eines einzigen Neurons zusammen zu fassen sind, werden Speicherzellenblöcke verwendet. Ein Speicherzellenblock der Größe S besteht aus S Speicherzellen und teilt sich mit diesen den selben „input gate“- und „output gate“-Mechanismus. Aktualisierung der Gewichtungen des RNNs besitzen im Vergleich zu anderen Algorithmen eine exzellente Laufzeitkomplexität von $\mathcal{O}(W)$, wobei W die Anzahl der Gewichte darstellt [Hoc97].

Zu der aktuellen Grundarchitektur eines LSTM gehört zusätzlich der von Gers et al. eingeführte „forget gate“-Mechanismus. Dieser sorgt über eine Überprüfung der in der Speicherzelle gespeicherten Werte mit Hilfe einer „sigmoid“-Funktion dafür, dass irrelevant gewordene Werte „vergessen“ werden. Vergessen bedeutet in diesem Kontext, dass der zu vergessene Wert mit 0 multipliziert wird und somit nicht mehr berücksichtigt wird. Der „input gate“- und „output gate“-Mechanismus bleiben dabei unverändert [Ger00].

Die Verwendung eines LSTMs als BiRNN wird als **„Bidirektionales Long Short-Term Memory“ (BiLSTM)** bezeichnet.

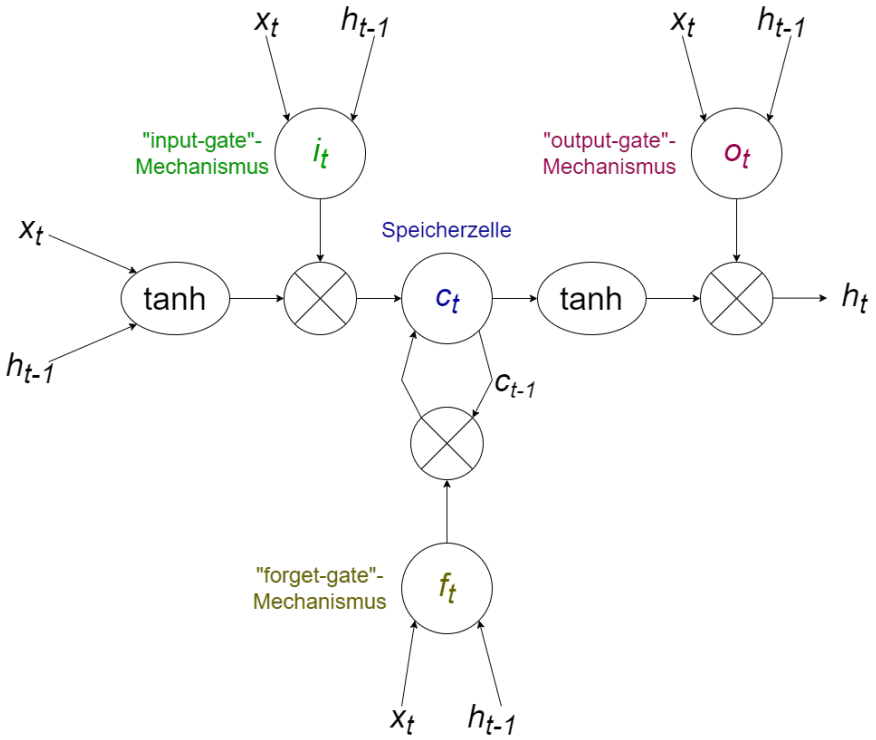

Abbildung 2.2: Der beispielhafte Aufbau einer LSTM-Zelle nach [Lia20]. t steht für den jeweiligen Zeitschritt und x_t für den aktuellen Eingabevektor. h_t steht für einen „versteckten Vektor“ und stellt die Ausgabe der Zelle dar. h_{t-1} steht für den Ausgabevektor des vorrangegangen Zeitschrittes. i_t, f_t, o_t sind die Funktionen und somit die Ausgaben von „input-gate“-, „forget-gate“- und „ output-gate“-Mechanismus.

2.4 Verarbeitung von Texten

Die Textverarbeitung ist ein zentraler Bereich der **NLP!** und somit auch der KI. Im Rahmen dieses Prozesses werden unstrukturierte Textdaten aufbereitet, sodass diese von Maschinen interpretiert und analysiert werden können. Die Vorverarbeitung des Textes bildet den ersten Schritt und die angewandten Methoden können je nach Anwendungsfall und gewählter Modellarchitektur variieren [Tab20]. Anschließend wird der

Text in einen Vektorraum transformiert, um eine neue Textrepräsentation zu schaffen. In der Regel stehen drei Hauptkategorien von Repräsentationsformen zur Verfügung: Häufigkeitsorientierte Ansätze, Einbettungsorientierte Ansätze und Transformerorientierte Ansätze. Basierend auf den Einbettungen können dann spezifische Berechnungen für die entsprechende Aufgabe durchgeführt werden [Lu20].

Vorverarbeitung Durch Textvorverarbeitung werden Texte und Dokumente in ein maschinenlesbares Format umgewandelt, das leicht verständlich und analysierbar ist. Eine Methode ist die **Satzsegmentierung**, bei der der Text in einzelne Sätze zerlegt wird. Für die **Veränderung zur Kleinschreibung** werden alle großgeschriebenen Buchstaben in kleingeschriebene umgewandelt. Eine weitere Technik ist die **Tokenisierung**, bei der der Text in kleinere Einheiten wie Wörter, Buchstaben und Satzzeichen zerlegt wird. **Part-of-Speech Tagging (POS)** ordnet jedem Wort seine grammatikalische Klasse zu, wie Verb, Adjektiv oder Substantiv. Stoppwörter, die in der Regel aus Hilfsverben, Konjunktionen und Artikeln bestehen, können entfernt werden, da sie für die Analyse oft irrelevant sind (**Stoppwörter-Entfernung)**. In einigen Fällen ist es auch sinnvoll, Satzzeichen zu entfernen (**Satzzeichen-Entfernung)**. Techniken wie **Stemming** („programmieren“ wird zu „programm“, aber „Schlieren“ wird zu „Schl“) und **Lemmatisierung** („gegangen“ wird zu „gehen“) reduzieren Wörter auf ihre Wurzel oder ihren Stamm. Eine logische Reihenfolge muss eingehalten werden um bei der Kombination der Techniken Fehler zu vermeiden [Tab20].

Techniken zur frequenzbasierten Repräsentation In der frequenzbasierten Textrepräsentation sind **„Bag-of-Words“ (BOW)**, **„Latent Dirichlet Allocation“ (LDA)** und **„Term Frequency - Inverse Document Frequenzy“ (TF-IDF)** drei gängige Methoden. BOW ist eine grundlegende Darstellungsform, bei der jede Spalte des Darstellungsvektors die Häufigkeit eines Wortes im Dokument angibt. Der individuelle Wortwert wird auch als Begriffshäufigkeit (TF) bezeichnet. Für TF-IDF wird die Begriffshäufigkeit mit einem Wert multipliziert, der die Gesamtzahl der Vorkommen des Wortes in Relation zur Anzahl der Dokumente (IDF) darstellt [Lu20]. LDA ist ein generatives probabilistisches Modell für Textkorpora, welches für die Einordnung von Dokumenten in mehrere Themen genutzt werden kann. Ein Thema wird dabei einem Dokument durch die Vorkommnisse von Wörtern in einer dem jeweiligen Thema zugehöriger Menge an Wörtern zugeordnet [Ble03].

Techniken zur einbettungsorientierten Repräsentation In der Vektorraumrepräsentation eines Wortes, auch Wort-Einbettung genannt, werden syntaktische und semantische Eigenschaften des Wortes durch die Distanzen zu anderen Wörtern im Vektorraum abgebildet. Die Technik **„word2vec“**[Mik13] gilt als erster Ansatz zur Erzeugung solcher Worteinbettungen. **„GloVe“**[Pen14] berücksichtigt zusätzlich die Wahrscheinlichkeit des gemeinsamen Auftretens von Wortpaaren, bei der Generierung der Wortvektoren. **„FastText“**[Boj16] erzeugt Wortvektoren, indem es Subwort-Informationen wie

Buchstaben oder Buchstabengruppen einbezieht. Diese Erweiterung ermöglicht eine präzisere Darstellung von unbekannten Wörtern im Vektorraum [Lu20]. Diese Wort-Einbettungstechniken spielen zwar eine zentrale Rolle im Bereich der NLP, stoßen jedoch an Grenzen, wenn es um die Repräsentation mehrdeutiger Wörter wie die „Bank“ als Sitzgelegenheit oder finanzielle Institution geht [Han21].

Techniken zur transformerorientierten Repräsentation Tiefere neuronale Netzwerke wie Transformer bieten im Vergleich zu bisherigen Möglichkeiten eine verbesserte Darstellung von mehrdeutigen Wörtern sowie komplexen lexikalischen und semantischen Strukturen [Han21]. Der Transformer-Modellansatz (Abbildung 2.3 wurde erstmals 2017 von Vaswani et al. präsentiert[Vas17]. Seitdem sind zahlreiche Varianten und Weiterentwicklungen entstanden, darunter BERT[Dev18], GPT[Rad18] und T5[Raf20]. Diese haben wiederum eigene Abwandlungen hervorgebracht, wie beispielsweise DistilBERT[San19], RoBERTa[Liu19], GPT-2/3/4[Rad19, Bro20, Ope23] und TS5X[Rob22]. Dieser Abschnitt fokussiert sich auf eine stark vereinfachte Darstellung des ursprünglichen Transformer-Modells von Vaswani et al., da somit die Grundlage für das Verständnis weiterer Transformer-Modelle geschaffen wird. Spezifische Varianten werden an den Stellen behandelt, an denen sie für den Kontext relevant sind.

Zunächst wird die Eingabesequenz in Worteinbettungen umgewandelt. Dabei werden keine der bereits beschriebenen Techniken verwendet, da das Transformer-Modell die Berechnung des Vektors eines Wortes während des Trainingprozesses lernt. Diesen Vektoren werden Positionsinformationen hinzugefügt. Diese sind im Gegensatz zu RNNs einfach zu integrieren und ermöglichen eine parallele und somit effizientere Verarbeitung. Das Modell besteht aus einem Encoder und einem Decoder. Die eingebetette Eingabesequenz wird durch den Encoder verarbeitet. Der Encoder besteht aus N identischen Schichten, wobei jede Schicht zwei Sub-Schichten besitzt. Diese bestehen aus einem „Self-Multi-Head“-Aufmerksamkeitsmechanimus und einem FFNN.
Ein **Aufmerksamkeitsmechanismus** kann an sich zunächst als Aufmerksamkeitsfunktion verstanden werden. Die Funktion selbst stellt eine Abfrage und eine Menge an Schlüssel-Wert-Paaren dar. Sowohl die Abfrage, als auch Schlüssel und Werte sind Vektoren. Für die Ausgabe der Funktion, also dem Ergebnis, wird die gewichtete Summe aller Werte berechnet. Für die Gewichtung wird eine Kompatibilitätsfunktion verwendet, welche eine Gewichtung anhand der Kompatibilität von Abfrage zum Schlüssel den Werten zuweist und somit einen Aufmerksamkeitswert ergibt [Vas17]. Die Schlüssel-Wert-Paare können im Allgemeinen als der gegebene Kontext verstanden werden. Ein **bidirektionaler Aufmerksamkeitsmechanismus** prüft nicht nur die Kompatibilität der Abfrage zum Schlüssel, sondern auch umgekehrt die Kompatibilität vom Schlüssel zur Abfrage. Diese bidirektionale Prüfung liefert gegensätzliche, jedoch sich ergänzende Informationen, wodurch die Genauigkeit des Gesamtergebnisses verbessert wird [Seo16]. Die Ableitung von Abfrage und Schlüssel-Wert Paar aus der selben Sequenz wird als

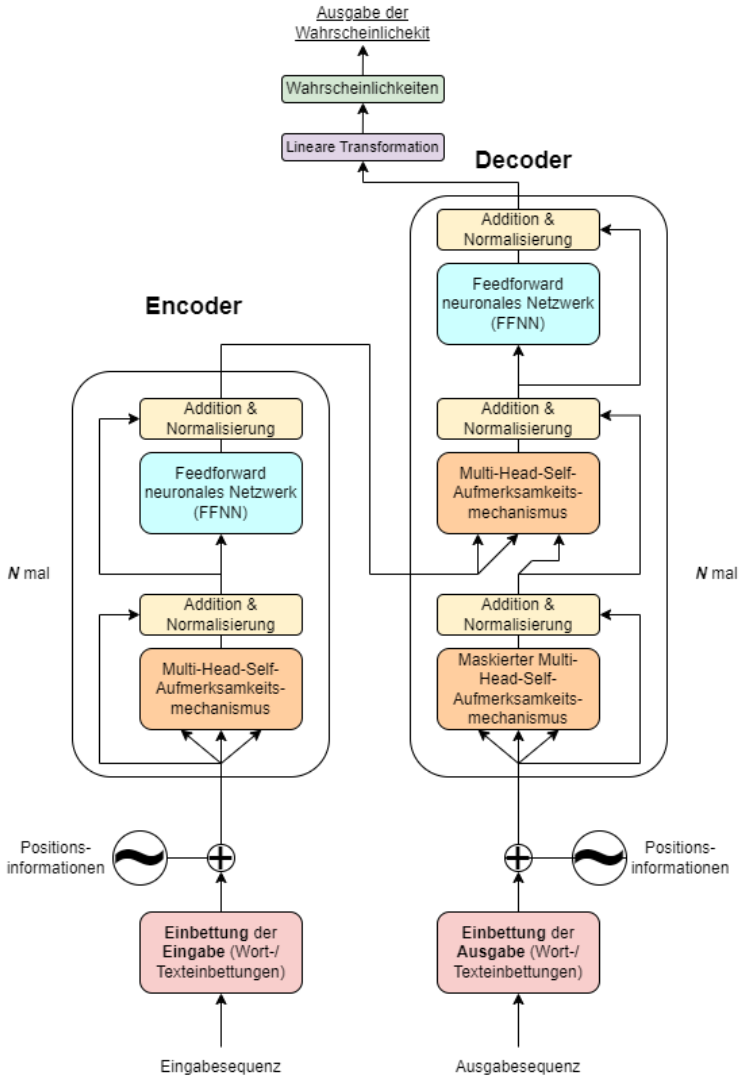

Abbildung 2.3: Transformer-Architektur nach [Vas17]. Es gibt N-Encoder und N-Decoder Schichten. Die Ausgabe einer Encoder-Schicht fließt in die nächste Encoder-Schicht. Die Ausgabe der letzten Encoder-Schicht, fließt in die zweiten Sub-Schichten aller Decoder-Schichten als Schlüssel-Wert-Paare ein. Die Ausgabe einer Decoder-Schicht fließt in die nächste Decoder-Schicht. Die letzte Decoder-Schicht gibt die Ausgabe an eine lineare Schicht und eine Funktion zur Berechnung der Wahrscheinlichkeiten an (im orginalen Transformer ist dies eine „softmax“-(Aktivierungs-)Funktion). Nähere Informationen zu dieser Grafik können in der Arbeit von Vaswani et al. gefunden werden.

„Self-Attention“, also **Selbstaufmerksamkeit** bezeichnet. Durch die Verwendung eines „Self-Multi-Head“-Aufmerksamkeitsmechanimus, also der Kombination von Selbstaufmerksamkeit und einem Zusammenspiel aus mehreren separaten Aufmerksamkeitsmechanismen, wird dem Modell ermöglicht, gleichzeitig auf unterschiedliche Repräsentationen der Daten zuzugreifen. Jeder „Kopf“ arbeitet unabhängig und kann sich daher auf unterschiedliche Aspekte der Eingabesequenz konzentrieren. Die Ausgaben werden konkateniert und mit Hilfe des FFNN wird eine Ausgabematrix in der gleichen Form wie die Eingabematrix erzeugt. Die Ausgabe kann daraufhin von der nächsten, gleich aufge-

bauten Schicht verarbeitet werden. Der auf den Encoder aufgebaute Decoder besitzt ebenfalls N identische Schichten, wobei eine Schicht der Architektur einer Encoder-Schicht gleicht und diese um einen „Self-Multi-Head“-Aufmerksamkeitsmechanimus für die Ausgabe des Encoders erweitert. Eine auf den Decoder aufbauende Ausgabeschicht entscheidet über die Form der Ausgabe [Vas17],

Diese Grundarchitektur kann individuell angepasst und dazu genutzt werden, um eine komplexere Repräsentation des Textes zu verwenden. Beispielsweise für NLP-Diziplinen wie Multi-Label Textklassifikation oder Named Entity Recognition.

2.5 Multi-Label Textklassifikation (MLTC)

Die Multi-Label-Textklassifikation ist eine wichtige Aufgabe im Bereich der NLP. Das Ziel der MLTC ist es, jedem gegebenen Text mehrere Labels zuzuweisen [Che20]. Die Anwendungsfälle sind dabei sehr vielfältig, und MLTC kann allgemein auch als Zuordnen von Kategorien oder deskriptiven Schlüsselwörtern für eine Sammlung von Texten oder Dokumenten verstanden werden (siehe Abbildung 2.4).

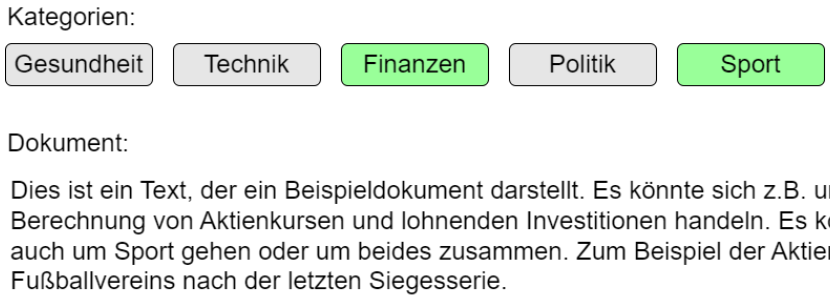

Abbildung 2.4: Ein Beispiel für eine MLTC-Aufgabe. Das Dokument stellt den zu prüfenden Text dar und die Kategorien die möglichen Kategorien. Die grün markierten Kategorien geben die Kategorien an, die dem Dokument zugeordnet werden sollen.

Um dies zu erreichen, wird ein automatisiertes System dazu trainiert, einem gegeben Text eine zugehörige Teilmenge an Kategorien aus einem Klassifikationssystem zuzuordnen. In der simpelsten Form kann die MLTC als eine Menge von mehreren binären Klassifizierungsaufgaben verstanden werden. Dies bedeutet, dass für jede mögliche Kategorie separat entschieden wird, ob diese dem gegebenen Text zugewiesen werden kann oder nicht [Nam14]. Hierbei wird MLTC als eine Problemtransformation behandelt. Weitere Ansätze behandeln MLTC als Algorithmusadaption. Hierbei werden Machine-Learning-Alogrithmen dazu verwendet um zu einem Ergebnis zu gelangen. Dabei wird die Zuweisung mehrerer Kategorien als eine Aufgabe in ihrer Gesamtheit verstanden und nicht als mehrere einzelne Aufgaben [Liu21]. Dies geschieht, da zwar die Reihenfolge

der zugewiesenen Kategorien grundsätzlich irrelevant ist, es jedoch Zusammenhänge zwischen den Vorkommnissen der Kategorien geben kann [Kem23].

Der Standardansatz für MLTC verwendet einen Dokumentkodierer in Kombination mit einer Reihe von Klassifikationsköpfen. Der Dokumentkodierer nimmt eine Sequenz von Tokens auf und erzeugt eine Dokumentendarstellung, die dann an die Klassifikationsköpfe weitergeleitet wird. Die Klassifikationsköpfe bestimmen dabei jeweils die Wahrscheinlichkeit der Zugehörigkeit eines Labels zu dem Dokument. Die Anzahl dieser Klassifikationsköpfe entspricht dabei der Anzahl der Label [Cha23].

2.6 Named Entity Recognition (NER)

Die Erkennung benannter Entitäten (NER) ist als Disziplin des NLP von zentraler Bedeutung für Anwendungen wie Frage-Antwort-Systeme, Informationsbeschaffung und Beziehungsextraktionen. In den letzten Jahren haben neuronale Netzwerkarchitekturen, insbesondere Deep-Learning-Modelle, den Stand der Technik in der NER erheblich verbessert, während frühere Ansätze auf von Menschen definierten Regeln und lexikalischen Merkmalen beruhen [Yad19].

Die Aufgabe der NER (siehe Abbildung 2.5) ist es, benannte Entitäten wie Personen, Orte, Organisationen und spezifische Fachbegriffe in Texten zu identifizieren. Die NER wird oft als Grundlage für die Bearbeitung weiterer Aufgabenstellungen innerhalb der NLP verwendet [Yad19]. Die konkrete Aufgabenstellung der NER wurde im Jahr 1996 auf der „Message Understanding Conference - 6“ vorgestellt [Gri96].

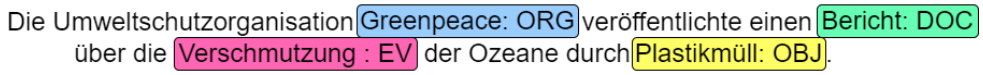

Abbildung 2.5: Ein Beispiel für eine NER-Aufgabe. Die Entitäten Organisation (ORG), Dokument (DOC), Ereignis (EV) und Objekt (OBJ) sind in dem Beispielsatz zu identifizieren.

In der modernen Forschung werden benannte Entitäten üblicherweise in zwei Kategorien unterteilt: allgemeine benannte Entitäten wie Personen und Orte, die üblicherweise durch Substantive repräsentiert werden, und domänenspezifische benannte Entitäten, die beispielsweise in der Biologie für Proteine und Enzyme relevant sind.

Durch den Einsatz von Deep Learning und kontinuierlichen vektorbasierten Repräsentationen haben neuere NER-Systeme den Stand der Technik erheblich verbessert, wobei eine umfangreiche Merkmals- oder Regelentwicklung nicht länger erforderlich ist [Li22b].

2.7 Metriken

Zur Evaluierung der Leistung von Modellen in den Bereichen NER und MLTC werden verschiedene Metriken herangezogen, die jeweils unterschiedliche Aspekte der Modellleistung abdecken. Insbesondere sind Präzision und Recall geeignete Metriken für Modelle, deren Hauptaufgabe die Auswahl bestimmter Elemente aus einer größeren Menge ist [Der16]. Der F_1-Wert, als Verknüpfung dieser Werte, kann in diesem Kontext als ebenso relevante Metrik betrachtet werden. Aufgrund seiner weit verbreiteten Anwendung in der Literatur wird der Genauigkeitswert ebenfalls aufgeführt.

Präzision Die Präzision P (Formel 2.1) gibt an, wie viele der als positiv klassifizierten Ausgabewerte tatsächlich positiv sind. Bei Labels oder Entitäten ist ein positiv klassifizierter Wert ein korrekt zugeordnetes Label oder eine korrekt zugeordnete Entität [Der16]. Zur Berechnung der **Präzision bei k** werden die ersten k Ergebnisse untersucht, wobei die Ergebnisse nach der Wahrscheinlichkeit der Zuordnung sortiert werden, vom höchsten zum niedrigsten Wert. Diese Wahrscheinlichkeit wird auch als Sicherheit bezeichnet. Für die Berechnung der **R-Präzision** wird k gleich der Anzahl der wahren Zuordnungen gesetzt [Man09].

$$P = \frac{|korrekt\ positiv|}{|korrekt\ positiv| + |inkorrekt\ positiv|} \tag{2.1}$$

Recall Der Recall R (Formel 2.2) gibt an, wie viele der zu findenden Werte tatsächlich gefunden wurden [Der16]. Der Recall kann als Sensitivität des Modells verstanden werden. Für die Berechnung des **Recall bei k** werden analog zur Präzision bei k nur die ersten k Ergebnisse, sortiert nach Wahrscheinlichkeit, in die Berechnung einbezogen.

$$R = \frac{|korrekt\ positiv|}{|korrekt\ positiv| + |inkorrekt\ negativ|} \tag{2.2}$$

F_1-Wert Der F_1-Wert ist eine Metrik, mit der die Leistung eines Klassifikationsmodells durch die Kombination von Präzision und Recall bewertet wird. Der Wertebereich für den F_1-Wert liegt zwischen 0 und 1, wobei ein Wert von 1 als optimal angesehen wird. Der F_1-Wert ist ein Spezialfall des allgemeineren F_β-Werts (Formel 2.3), bei dem durch die Einstellung von $\beta = 1$ Präzision und Recall gleich gewichtet werden [Gou05]. Der **Beispiel basierte F_1-Wert** betrachtet jedes Beispiel einzeln und berechnet den Durchschnitt dieser Werte [Nam17].

$$F_\beta = \frac{(1+\beta^2)|korrekt\ positiv|}{|korrekt\ positiv| + |inkorrekt\ positiv| + |korrekt\ positiv|\beta^2 + |inkorrekt\ negativ|\beta^2}$$
$$= (1+\beta^2)\frac{P \cdot R}{R + \beta^2 \cdot P} \tag{2.3}$$

Mikro und Makro Für Mikro- und Makrowerte wie -Precision, -Recall oder -F_1 werden die Ergebnisse nach einem anderen Schema berechnet. Bei den Makrowerten werden die Label-/Entitätsgruppen einzeln berechnet und daraus der Mittelwert gebildet [Ere12]. Für die Berechnung der Mikrowerte werden dagegen alle Labelvorhersagen bzw. alle erkannten Entitäten als ein Vektor betrachtet. Die Basisberechnung ist daher nur auf binäre Probleme anwendbar, während die Mikro- und Makro-Varianten für Multi-Label-Probleme geeignet sind und zwei unterschiedliche Sichtweisen eröffnen [Lip14].

Genauigkeit Die Genauigkeit A (Formel 2.4) gibt an wie viele korrekte Resultate ein Modell ausgibt, im Verhältnis zu allen Resultaten. Dabei neigt der Genauigkeitswert dazu die Leistung eines Modells zu überschätzen [Rib20].

$$A = \frac{|korrekt\ positiv| + |korrekt\ negativ|}{|korrekt\ positiv| + |inkorrekt\ positiv| + |korrekt\ negativ| + |inkorrekt\ negativ|} \tag{2.4}$$

Hamming Loss Der Hamming-Loss HL (Formel 2.5) gibt die Anzahl der falsch erkannten Labels oder Entitäten mit einem Wert zwischen 0 für keine Fehler und 1 für ausschließlich Fehler an [Dem10]. Dabei repräsentiert y die korrekten Labels und $h(x)$ die Vorhersagen der Labels. Durch Iteration von m möglichen Labels wird überprüft, ob das jeweilige Label y_i in Form von $h_i(x)$ korrekt zugeordnet wurde. Bei Gleichheit wird eine 0 zur Summe addiert, bei Ungleichheit eine 1. Diese Summe wird schließlich mit $\frac{1}{m}$ multipliziert, um einen Wert zwischen 0 und 1 zu erhalten.

$$HL(y, h(x)) = \frac{1}{m}\sum_{i=1}^{m}(y_i \neq h_i(x)) \tag{2.5}$$

3 Konzept

In diesem Kapitel werden zunächst die funktionalen und nichtfunktionalen Anforderungen definiert, die von der mbi GmbH gestellt wurden. Des Weiteren erfolgt eine strukturierte Literaturrecherche zur Identifikation aktueller Techniken mit dem weiteren Ziel, eine geeignete Modelllösung zu identifizieren, darauf aufbauend ein Architektur- und Implementierungskonzept für die gefundene Lösung zu entwickeln sowie ein Evaluationskonzept zu erstellen, das sowohl auf die gefundene Lösung als auch auf andere MLTC- und NER-Modelle anwendbar ist.

3.1 Anforderungen an das System

Für die Realisierung eines KI-Modells, welches durch die Verarbeitung von natürlicher Sprache in Textform in der Lage ist, mehrere Label zuzuweisen sowie Entitäten aus dem jeweiligen Text zu extrahieren, wurden von der mbi GmbH verschiedene Anforderungen gestellt. Die Anforderungen sollen den Einsatz in einem industriellen Anwendungsfeld, die vielseitige Einsetzbarkeit des Modells und die entsprechende Anpassbarkeit für unterschiedliche Anwendungen gewährleisten. Die Anforderungen werden in funktionale und nicht-funktionale Anforderungen aufgeteilt. Funktionale Anforderungen spezifizieren die Abläufe oder Funktionen, die ein System ausführen soll, und beschreiben das Systemverhalten in Bezug auf Eingaben, Ausgaben und deren Beziehungen. Nichtfunktionale Anforderungen beschreiben, unter welchen Bedingungen und mit welchen Anforderungen ein System seine Aufgabe erfüllen soll. Sie definieren die Qualität und die Rahmenbedingungen der Systemleistung [Gli07].

3.1.1 Funktionale Anforderungen

Die funktionalen Anforderungen (FA) sind für die Konzeption und Implementierung des Modells von entscheidender Bedeutung, um eine effiziente Lösung sicherzustellen.

FA-1 **MLTC:** Die Implementierung der KI-Lösung sollte so gestaltet sein, dass sie in der Lage ist, eingegebenen Texten von null bis zu einer beliebigen Anzahl n systembekannte Label zuzuordnen und die Ergebnisse im Textformat auszugeben.

FA-2 **NER:** Die KI-Lösung muss in der Lage sein, in einem eingegebenen Text Entitäten zu erkennen, die zu dem Modell bekannten Entitätsklassen gehören. Die erkannten Entitäten werden dann zusammen mit der erkannten Entitätsklasse im Textformat ausgegeben.

FA-3 **Textverarbeitung:** Das System muss in der Lage sein, Texte selbstständig zu verarbeiten. Genauer bedeutet dies, dass das System in der Lage sein muss, die Eingabe eines vollständigen Textes zu verarbeiten, einschließlich eventuell erforderlicher Tokenisierung oder anderer Vorverarbeitungsmethoden.

FA-4 **Training:** Das Modell muss die Fähigkeit besitzen anhand eines gegebenen Trainingsdatensatzes und einer zugehörigen Konfiguration in einem strukturierten Format (beispielsweise JSON[1], YAML[2] oder XML[3]) ein Training zu durchlaufen, welches dem Modell ermöglicht, die zugrundeliegenden Muster und Zusammenhänge in den Daten zu erkennen und zu erlernen.

FA-5 **Validierung:** Das System muss über Mechanismen zur Validierung der Leistung des KI-Modells verfügen. Darunter werden Tests mit einem Validierungsdatensätzen verstanden, bei denen die Leistung des Modells anhand reproduzierbarer und vergleichbarer Metriken gemessen wird.

3.1.2 Nicht-funktionale Anforderungen

Nicht-funktionale Anforderungen (NFA) konzentrieren sich auf die Rahmenbedingungen des Systems. Zur Überprüfung der Erfüllung der genannten Anforderungen werden entsprechende Akzeptanzkriterien festgelegt, sofern diese nicht bereits durch die Anforderung selbst definiert sind.

NFA-1 **Leistung:** Das System sollte nicht länger als zehn Sekunden für die Verarbeitung eines Dokumentes benötigen. Als Referenzwert wird ein Dokument als Text zwischen 5000 und 6000 Zeichen inklusive Leerzeichen definiert.

1. Die durchschnittliche Verarbeitungszeit eines Dokuments unter zehn Sekunden.

NFA-2 **Flexibilität:** Das System muss so konzipiert sein, dass auch bei Nutzung eines anderen Datensatzes, nichts an der Architektur des Systems verändert werden muss

1 https://www.json.org/json-en.html
2 https://yaml.org/
3 https://www.w3.org/XML/

NFA-3 **Rechenkapazität:** Rechenkapazitäten sollten effizient genutzt werden um einen alltäglichen Gebrauch zu ermöglichen.

1. Sowohl das Training, als auch die Ausführung der Lösung muss auf einem Computer mit 32 Gigabyte Arbeitsspeicher und einem Intel Core i7 Prozessor der zwölften Generation möglich sein.

NFA-4 **Datensatz:** Für das Training und die Validierung des Modells soll ein öffentlich zugänglicher Datensatz gefunden werden. Nach Möglichkeit soll ein einziger Datensatz für die MLTC und NER Aufgaben verwendet werden.

1. Die Texte des Datensatzes sind ausschließlich in deutscher oder englischer Sprache.

NFA-5 **Kategorienanzahl:** Zur Simulation eines realen Anwendungsszenarios im industriellen Umfeld, muss die Anzahl der möglichen Label und Entitätsklassen eingeschränkt werden.

1. Die Anzahl der Label, welche im Kontext der MLTC Aufgabe vergeben werden können, muss zwischen 10 und 20 Labeln liegen.

2. Im Rahmen der NER müssen mindestens drei und maximal fünf mögliche Entitätskategorien zur Verfügung stehen.

NFA-6 **Genauigkeit:** Um einen verlässlichen Einsatz in unterschiedlichen Szenarien zu gewährleisten, muss eine gewisse Erkennungsgenauigkeit erreicht werden. Dies schließt nicht nur eine möglichst hohe Anzahl an korrekt erkannten Labeln und Entitäten, sondern auch eine möglichst geringe Anzahl an inkorrekt erkannten Label und Entitäten mit ein.

1. Sowohl bei der Zuweisung von Labeln zu Texten im Rahmen der MLTC, als auch bei der Erkennung von benannten Entitäten im Rahmen der NER müssen 90% der korrekten Label und Entitäten korrekt identifiziert werden. Dies entspricht einem Mikro-Recall von 0,9.

2. In beiden Disziplinen dürfen nicht mehr als 10% der erkannten Entitäten und Label inkorrekt positiv sein. Dies entspricht einer Mikro-Präzision von 0,9.

3.2 Identifikation von aktuellen Techniken

Zur Beantwortung von *FF-1* und *FF-2* unter Erfüllung der funktionalen und nichtfunktionalen Anforderungen sowie zur Beantwortung von *FF-3*, wird der aktuelle Stand der Technik im Rahmen einer strukturierten Literaturrecherche (SLR) nach Konzepten von Kitchenham[Kit07] sowie Webster und Watson[Web02, Wat20] identifiziert. Der Fokus liegt dabei auf den Modellarchitekturen zur Bewältigung von MLTC- und NER-Aufgaben sowie den dabei verwendeten Metriken zur Bewertung. Ziel der anschließenden Diskussion der SLR-Ergebnisse ist es, mögliche Vor- und Nachteile der beschriebenen Architekturen herauszukristallisieren und einen Überblick über die Verwendung der identifizierten Metriken zu geben.

3.2.1 Motivation

Ziel der SLR ist es, einen fundierten Überblick über den aktuellen Stand der Forschung in den Bereichen Multi-Label Textklassifikation (MLTC) und Named Entity Recognition (NER) zu geben. Obwohl die Übersicht keinen Anspruch auf Vollständigkeit erhebt, stellt sie eine sorgfältige Auswertung der vorhandenen Literatur dar.

Die Erstellung eines Realisierungskonzeptes ist somit nachvollziehbar, reproduzierbar und basiert auf dem aktuellen Stand der Forschung. Unterschiedliche Ansätze können so identifiziert und im Rahmen einer Diskussion verglichen werden, wodurch sichergestellt wird, dass eine ideale Konzeption für die Realisierung des Systems entsprechend dem Umfang der SLR erfolgen kann. Darüber hinaus ergibt sich dadurch die Möglichkeit, unterschiedliche Metriken zu identifizieren, die im weiteren Vorgehen aufgrund ihrer Anwendung in verschiedenen Arbeiten für die Erstellung eines Bewertungskonzeptes evaluiert werden können.

3.2.2 Methode

Eine SLR ist nach Kitchenham[Kit07] in drei Phasen unterteilt (siehe Abbildung 3.1). Diese drei Phasen werden als Leitfaden für die Durchführung der vorliegenden SLR genutzt, wobei sich im Rahmen der Suchstrategie und dem Umgang mit den Ergebnissen zusätzlich an Konzepten von Webster und Watson orientiert wird[Web02, Wat20]. Innerhalb der ersten Phase der Planung, werden zunächst die übergeordneten Ziele der SLR definiert. Dies wurde bereits im Abschnitt 3.2.1 getan. Die Ziele werden, im Rahmen der SLR, in zu beantwortende SLR-Forschungsfragen formuliert. Weitergehend werden für die SLR benötigte Inklusions- und Exklusionskriterien definiert, welche den Rahmen für die folgende Identifikation von Studien ergeben. In der zweiten Phase werden nach den

definierten Kriterien relevante Studien identifiziert und validiert. Die Validierung der gefundenen Studien erfolgt in einem iterativen Prozess, in dem zunächst die gefundenen Titel beurteilt werden. Die daraus resultierenden Arbeiten werden über ihren Abstract einer weiter Überprüfung unterzogen. Zuletzt werden die Volltexte der Arbeiten gelesen, um die Relevanz für die Beantwortung der vorab aufgestellten Fragen sicherzustellen. Zuletzt wird auf Basis dieser gefundenen Literatur eine „forward-“ und „backward“-Suche nach Webster und Watson durchgeführt [Web02]. Innerhalb einer „backward-search“ werden die genutzten Quellen der als relevant identifizierten Publikationen betrachtet und die vorangegangenen Arbeiten auf Relevanz für die eigene Thematik geprüft. Das Prinzip der „forward-search“ funktioniert umgekehrt. Dabei werden die Arbeiten unter dem selben Gesichtspunkt betrachtet, in denen die als relevant befundenen Publikationen als Referenz angegeben wurden. Als letzter Schritt der Phase werden die benötigten Informationen und Daten aus den für relevant befundenen Arbeiten extrahiert. Die letzte Phase beinhaltet eine Auflistung und Diskussion der Ergebnisse, wodurch die aufgestellten Forschungsfragen der SLR beantwortet werden. Die als relevant identifizierten Publikationen wurden nach dem Vorschlag von Webster und Watson in einen Elementgraphen eingefügt. In dieser Arbeit wurde dafür eine Graph Datenbank[1] genutzt. Dies dient im Allgemeinen dazu, die Effizienz und Effektivität der Literaturrecherche zu verbessern. Ziel dabei ist es demnach auch, kausale und nicht kausale Zusammenhänge zwischen verschiedenen Arbeiten offensichtlicher darzustellen [Wat20]. Als Elemente wurde sich in dieser Ausarbeitung für die Entitäten Publikationen, Verlage, Zeitschriften, Jahr, Monat, sowie von den Publikationsautoren gewählte Schlüsselwörter und eigene, für sinnvoll befundene, Schlüsselwörter entschieden. Diese Entitäten stellen dabei jeweils eine Knotenkategorie dar. Die jeweiligen Knoten besitzen für die jeweilige Knotenkategorie einheitliche Eigenschaften. Für eine Publikation ist dies beispielsweise der Titel, der „Digital Object Identifier“ (DOI), falls dieser vorhanden ist, eine URL unter der die Ausarbeitung im Internet gefunden werden kann sowie eine Bibtex-Zitierung. Die Knoten sind über benannte und zielgerichtete Verbindungen miteinander verknüpft. Diese Verbindungen zeigen von welchem Verlag, in welcher Zeitschrift, in welchem Jahr und Monat eine Publikation veröffentlicht wurden. Des Weiteren zeigen sie auf, welche Schlüsselwörter sich die Publikationen teilen oder alleinig besitzen, welche Ausarbeitung welche Arbeit zitiert oder auf welcher Plattform die Publikation ursprünglich entdeckt wurde. Eine visualisierte Darstellung einer Publikation sowie ein Abbild der Datenbank können in Abbildung 3.2 und 3.3 gefunden werden.

1 Skript und Datenbank können unter `https://hbx.fhhrz.net/getlink/fiTZFehRAmFgEQjQqkQF9u/literatur_recherche.zip` gefunden werden.

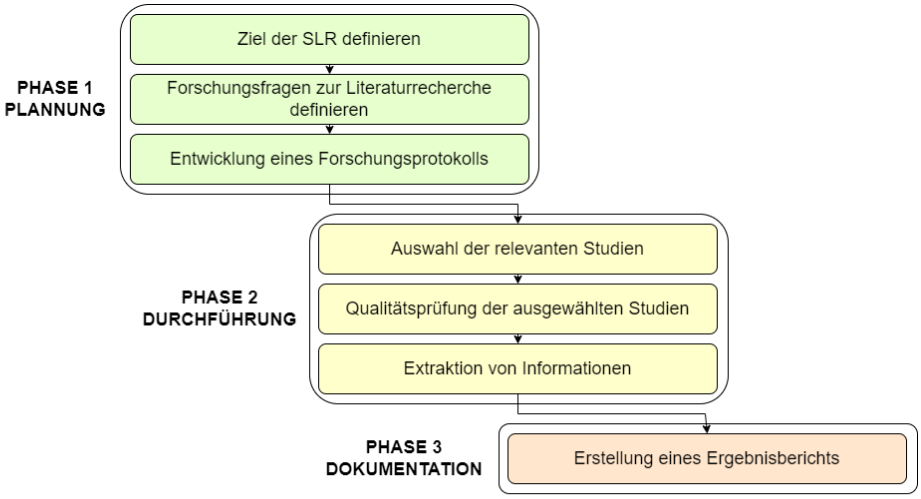

Abbildung 3.1: Der vereinfachte Aufbau einer strukturierten Literaturrecherche (SLR) nach Kitchenham [Kit07].

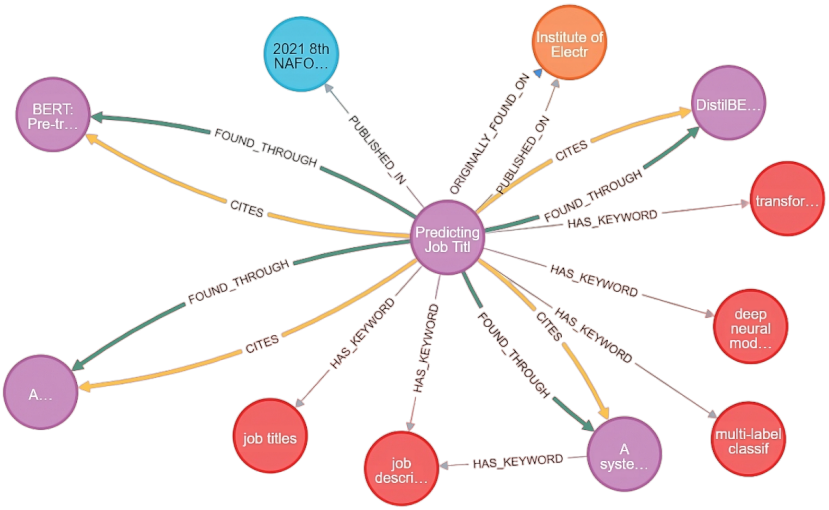

Abbildung 3.2: Eine Publikation (Mitte, lila) in der Graphdatenbank mit den zugehörigen Verknüpfungen zu Zitationen (gelb), anderen Publikationen (lila), zugeordneten Schlagwörtern (rot) und Publikationsinformationen (orange, blau).

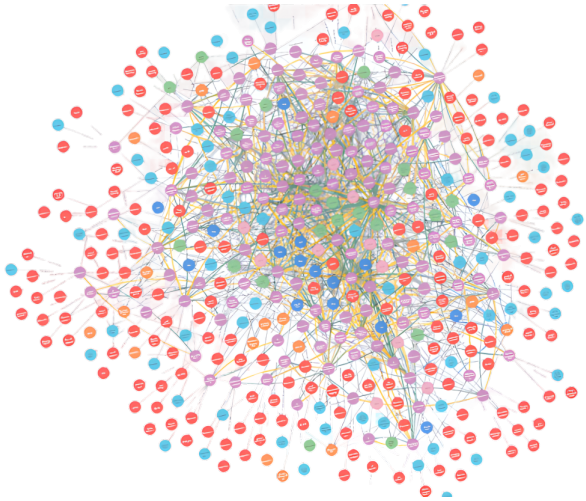

Abbildung 3.3: Eine visualisierte Darstellung über die Verbindungen zwischen den Knoten innerhalb der erstellten Graphdatenbank.

3.2.3 Forschungsfragen der SLR

Um das Ziel der SLR zu erreichen, wurden vier Forschungsfragen definiert:

SLRFF-1 Welche Methoden und Techniken erzielen derzeit überdurchschnittliche Leistungen im Bereich der Multi-Label Textklassifizierung?

SLRFF-2 Welche Methoden und Techniken erzielen derzeit überdurchschnittliche Leistungen im Bereich der Named Entity Recognition?

SLRFF-3 Welche integrierte Ansätze gibt es, die Multi-Label Textklassifizierung und Named Entity Recognition miteinander verbinden?

SLRFF-4 Welche Metriken kommen bei der Evaluierung von Multi-Label Textklassifikations und Named Entity Recognition System zum Einsatz?

3.2.4 Inklusions- und Exklusionskriterien

Die Festlegung von Ein- und Ausschlusskriterien ist ein entscheidender Schritt bei der systematischen Literaturrecherche, um die Relevanz und Qualität der ausgewählten Publikationen zu gewährleisten. Diese Kriterien dienen als methodische Leitlinien, die eine gezielte und reproduzierbare Auswahl der Literatur ermöglichen. Sie tragen dazu bei, den Umfang der Recherche einzugrenzen und gleichzeitig sicherzustellen, dass die ausgewählten Arbeiten einen direkten Beitrag zur Beantwortung der Forschungsfragen leisten.

Die für die SLR verwendeten Inklusions- und Exklusionskriterien werden wie folgt definiert:

Themenbezug Die Publikationen müssen einen expliziten Bezug zu MLTC und NER aufweisen. Dies bedeutet, dass der Schwerpunkt der betrachteten Arbeiten direkt auf diesen beiden Disziplinen im Rahmen der Verarbeitung natürlichsprachlicher Texte liegen muss. Varianten wie Extreme MLTC oder klassische Textklassifikation werden in diesem speziellen Kontext nicht als Teil von MLTC betrachtet.

Zeitraum Die Suche ist auf Publikationen zwischen dem 1. Januar 2015 und dem 1. August 2023 beschränkt.

Zugänglichkeit und Verfügbarkeit Alle Publikationen müssen über das Internet und ohne weitere Kosten zugänglich sein. Es wurden sieben wissenschaftliche, übers Internet zugängliche Datenbanken und Suchplattformen ausgewählt, um eine möglichste hohe Breite an Publikationen identifizieren zu könnnen.

Sprache Es werden Arbeiten welche die Aufgabenstellung der NER oder MLTC mit Fokus auf die deutsche oder englische Sprache bevorzugt. Der Grund dafür ist, dass die Sprachstruktur zu einer Variation in den Ergebnissen von Modellen bei der Ausführung von NLP-Aufgaben führen kann [Sid20, Rud21]. Arbeiten, die Modelle auf der Basis anderer Sprachfamilien verwenden, werden jedoch nicht grundsätzlich ausgeschlossen. Diese werden insbesondere dann berücksichtigt, wenn sie in anderen Aspekten eine hohe Relevanz für die Arbeit zeigten.

Anzahl an Ergebnissen Plattformen auf denen mehr als 10.000 Ergebnisse aus der Suchabfrage resultieren, müssen sinnvoll weitere Einschränkungen erhalten. Die Einschränkungen ergeben sich aus den Limitationen und Möglichkeiten der jeweiligen Plattform vor dem Hintergrund dennoch möglichst viele Treffer zu erzielen. Somit wird für Springer Link die Subdisziplin Ärtificial Intelligence", übersetzt "künstliche Intelligenz"gewählt und auf ACM lediglich die Titel der Publikationen nach Übereinstimmungen mit der Suchabfrage geprüft.

Plattformen Als Plattformen werden „ArXiv“, „Institute of Electrical and Electronics Engineers (IEEE)“, „Emerald Insight“, „Springer Link“, „Google Scholar“, „Association for Computing Machinery (ACM)“ und „ScienceDirect“ gewählt

Suchbegriffe Vor diesem Hintergrund werden die ausgewählten Suchbegriffe „multi-label text classification“, „named entity recognition“ und „transformation models“ mittels der booleschen Operatoren „AND“ und „OR“ so miteinander verknüpft, dass mindestens zwei der Schlüsselbegriffe in den aus der Suche resultierenden Publikationen

vorkommen müssen[1]. Die Auswahl des Suchbegriffes "transformer models"rührt neben dem Vorkommen in Schlüsselwörter und Titeln daher, dass MLTC und NER Disziplinen der Verarbeitung von natürlicher Sprache, im Englischen als „Natural Language Processing“ (NLP) bezeichnet, sind und die Verwendung von Transformer Modellen als akuteller Stand der Technik im Bereich der NLP gilt [Cha20, Fri19, Yan21].

Limitation der Plattformen Die Plattformen ACM und Google Scholar weisen eigene Limitationen bei der Anzeige der Ergebnisse auf. So werden auf der Plattform ACM lediglich die ersten 2000 Ergebnisse angezeigt und auf Google Scholar nur die ersten 1000.

3.2.5 Auswahl und Validierung der Literatur

In der ersten Phase wurden 17.672 Publikationen von sieben verschiedenen wissenschaftlichen Plattformen gesichtet. Diese wurden in mehreren Iterationen hinsichtlich ihrer Relevanz für die Forschungsfragen bewertet. In der ersten Iteration wurden 250 Arbeiten, respektive 244 abzüglich der sechs Dopplungen als potentiell von Belangen erkannt. Dabei konnten auf den Plattformen ACM maximal 2000 Einträge angezeigt werden und auf Google Scholar 1000. Auf Grund dessen wurde auf diesen Plattformen die Bestandteile der Suchabfrage welche durch das boolesche „OR“ getrennt sind, einzeln gesucht um möglichst alle Ergebnisse einsehen zu können. Ein Teil der Suchergebnisse konnte dennoch auf Grund von Dopplungen in den Ergebnissen und den Einschränkungen der Plattformen nicht eingesehen werden, welche somit ebenfalls in der Endauswertung fehlen.

Die ausgewählten Publikationen wurden in einer weiteren Iteration genauer untersucht und anhand der Abstracts bewertet. Von den 132 Publikationen, welche anhand des Abstracts als relevant identifiziert wurden, konnte auf 29 aufgrund deren kostenpflichtigen Erwerbung nicht zugegriffen werden. Diese 29 wurden dementsprechend im weiteren Verlauf nicht weiter berücksichtigt. Die anderen 103 Ausarbeitung wurden einer Volltext-Prüfung unterzogen, wodurch 52 Publikationen als direkt relevant für die Forschungsziele identifiziert werden konnten. Ergänzend wurden zwei Suchstrategien angewandt: Eine „backward“-Suche führte zur Identifikation von 85 weiteren Publikationen, die in den ursprünglich 52 ausgewählten Publikationen insgesamt 199 Mal zitiert wurden. Eine „forward-Suche“, führte zur Identifikation von 17 weiteren relevanten Publikationen,

1 Für die Suche auf Google Scholar müssen die Operatoren angepasst werden. Diese ergibt die folgende Abfrage: „multi-label text classification named entity recognition | multi-label text classification transformer models | named entity recognition transformer models“. Für alle weiteren Plattformen ergibt sich die folgende Abfrage: „(multi-label text classification AND named entity recognition) OR (multi-label text classification AND transformer models) OR (named entity recognition AND transformer models)“

wobei zwei dieser Publikationen mehrmals zitiert wurden. In beiden Schritten wurde der initial gewählte zeitliche Rahmen gewahrt, jedoch wurden vier Werke welche vor 2015 erschienen sind auf Grund ihrer hohen Relevanz der Literatursammlung hinzugefügt. Zusammengenommen ergibt dies eine Anzahl von 148 Publikationen (siehe Tabelle 3.1), die für die vorliegende Untersuchung als relevant erachtet werden.

Plattform	Suche	Titel	Abstract	Volltext	Kein Zugriff	Forward-S.	Backward-S.
IEEE	135	25	16	**13**	0	6	**47**
Emerald Insight	105	4	3	0	3	0	0
arXiv	250	21	13	10	0	**9**	40
ACM	2654	60	29	10	0	0	36
Springer Link	3431	27	14	3	**9**	1	14
Google Scholar	5070	**73**	**35**	11	**9**	2	44
ScienceDirect	**5967**	40	22	5	8	1	18
Insgesamt	17672	250	132	52	29	17	85

Tabelle 3.1: Anzahl der gefunden Literatur nach Iteration pro Suchplattform inklusive Dopplungen.

3.2.6 Informationsextraktion

Eine Analyse der Zitationen ergab, dass die am häufigsten zitierten Arbeiten grundlegende Arbeiten zur Transformer-Architektur sind. Besonders hervorzuheben sind die Publikationen „BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding“ mit 35 Zitaten und „Attention is All you Need“ mit 22 Zitaten. Weitere wichtige Beiträge sind „RoBERTa: A Robustly Optimized BERT Pretraining Approach“ und „DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter“ mit jeweils 9 Zitaten.

Die zeitliche Verteilung der relevanten Publikationen zeigt eine Häufung in den Jahren 2023, 2021 und 2022 mit 29, 28 und 27 Publikationen. Bei den Verlagen dominieren „ArXiv“ und das „Institute of Electrical and Electronics Engineers (IEEE)“ mit jeweils 50 und 30 Publikationen. Dabei ist zu beachten, dass Publikationen zum Teil bei unterschiedlichen Verlagen erscheinen, in dieser Analyse aber nur der jeweils zuerst gefundene Verlag berücksichtigt wurde.

Die von den Autoren am meisten verwendeten Schlagworte bestehen untere anderem aus Named Entity Recognition (18 Mal), Multi-Label Classification (17 Mal), Natural Language Processing (16 Mal) und Text Classification (12 Mal). Von den 148 relevanten Publikationen sind 47 der Named Entity Recognition und 62 der Multi-Label Text Classification zuzuordnen. Die restlichen Publikationen behandeln Disziplinen, die für die vorliegende Arbeit von untergeordneter, aber dennoch von Relevanz sind, wie z.B. Transformer-Modelle, Evaluationsansätze und die Implementierung verschiedener KI-Modelle.

3.2.7 Aktueller Stand der Forschung

Zur Beantwortung von *SLRFF-1* und *SLRFF-2* wurden die gefundenen Arbeiten auf ihre Leistungsfähigkeit untersucht, um einen „Stand der Technik“, im Englischen „State-of-the-art“ (SOTA) genannt, identifizieren zu können. Diese sind aufgrund der zum Teil unterschiedlichen getesteten Datensätze und Ansätze unterschiedlich zu bewerten. Aus diesem Grund werden zunächst einige Modelle vorgestellt, die im Rahmen der SLR derzeit die beste Performance erzielen. Diese Darstellung ist unterteilt in MLTC und NER. Zur Beantwortung der dritten und vierten Forschungsfrage der SLR werden zusätzlich die Unterteilungen „Kombination“ und „Metriken“ vorgestellt.

MLTC Die aktuelle Literatur zum Thema der Multi-Label Textklassifikation werden Transformer basierte Encoder in den meisten Fällen als Grundlage für die vorgeschlagenen Modelle verwendet. Dabei wird 18 Mal und damit am häufigsten das 2018 von Devlin et al. entworfenen Sprachrepräsentationsmodell „Bidirectional Encoder Representations from Transformers“ (BERT) [Dev18] verwendet. Oftmals werden auch Abwandlungen des BERT Modells verwendet, bei denen weniger Parameter verwendet oder das Modell auf einer spezifische Sprache oder Anwendungszweck angepasst wurde. Jeweils drei Mal und damit nach BERT am zweit häufigsten wurden DistilBERT und RoBERTa verwendet. DistilBERT ist ein Modell welches 40% kleiner ist als BERT, 97% der Sprachkompetenzen beibehält und währendessen 60% schneller ist [San19]. „Robustly optimized BERT approach“ (RoBERTa) ist ein Ansatz, bei dem BERT länger und mit mehr Daten trainiert wird. Dies führt zu einem Leistungsanstieg verglichen mit dem herkömmlichen BERT Modell [Liu19].

Javeed präsentiert einen Ansatz für die Multi-Label Klassifikation von Dokumenten, zum Einsatz in einem industriellen Umfeld. Dabei werden konkret Probleme angegangen, welche die essentielle Funktionsweise eines solchen Modells beeinträchtigen kann. Dazu zählen beispielsweise die Verarbeitung von Dokumenten mit unterschiedlicher Länge, das Problem des katastrophalen Vergessens oder auch die Modularität und Erklärbarkeit eines solchen Modells. Als Grundlage dient der „Universal Sentence Encode Model“(USE)[Cer18] in der „DAN Variante“. Diese zeichnet sich durch die

Einbettungen ganzer Sätze sowie der einfachen Verwendung aus [Jav23]. Aufbauend auf der „DAN Variante“ wird eine Abwandlung der sogenannten Hydranet Architektur nach Mullapudi et al.[Mul18] verwendet. Die Hydranet-Architektur besteht aus einem Rückgrat, in diesem Fall die „DAN Variante“, einem LSTM, mehreren Zweigen/Köpfen für spezifische Klassifikationsaufgaben und einem vorrangehenden Steuerungsmechanismus. In diesem Fall wird der Steuerungsmechanismus allerdings ausgelassen und die Zweige selbst haben die Erlaubnis die jeweilige Klassifikation auszuführen. Dies wird getan, um die Zweige an Klassifikatoren zu modularisieren und somit auswechselbar zu machen. Diese Variante der Hydranet-Architektur wird verwendet, um dem Problem des katastrophalen Vergessens entgegenzuwirken [Jav23]. Katastrophales Vergessen tritt auf, wenn ein maschinell lernendes System nach dem Training für eine zweite Aufgabe die Fähigkeit verliert, die erste Aufgabe auszuführen. Dies geschieht, weil das Wissen des trainierten Modells über die erste Aufgabe durch das Training mit der zweiten Aufgabe überschrieben wird. Eine erfolgreiche Klassifikation der ersten Aufgabe ist in diesem Fall nur noch auf zufällige Ähnlichkeiten zwischen den beiden Aufgaben zurückzuführen [Goo13]. Im Rahmen des Modells und der Hydranet-Architektur wird ein BiLSTM verwendet. Als Köpfe oder Zweige des Modell werden sowohl Transformer, als auch BiLSTM getestet. Dabei konnte der BiLSTM Ansatz auf allen getesteten Datensätzen bessere Ergebnisse erzielen. Die Genauigkeit für Web of Science (5763)[Kow17] liegt bei 94,45%, für Web of Science (11967)[Kow17] bei 92,81%, für BBC Sports[Gre06] bei 98,88% und für den BBC News[Gre06] Datensatz bei 99,06% [Jav23]. Anzumerken ist, dass im Falle des BBC Sports Datensatzes mit Hilfe des MPAD-path Models, entwickelt von Nikolentzos et al., eine höhere Genauigkeit von 99,59% erzielt wurde [Nik20].

Tang et al. präsentieren ein Transformer-LDA-MLTC-Modell für lange Texte. Dabei wird die Geschwindigkeit und Genauigkeit der Klassifikation von langen Texten verbessert, sowie die Fähigkeiten von traditionellen Multi-Label Textklassifikations Algorithmen übertroffen. Auf dem THUCNews[Sun16] Datensatz konnte dabei eine Genauigkeit von 96,23%, eine Präzision von 97,2%, sowie ein Recall und F_1-Wert von 93,2% und 95,16% erreicht werden. Das Modell nutzt die Transformer-XL Architektur als Grundlage [Tan23]. Transformer-XL selbst baut dabei auf herkömmlichen Transformern und RNNs auf und steht für „Transformer extra large“. „Extra large“, zu deutsch extra groß bedeutet in diesem Fall eine beliebige Länge von Texten, respektive Texte mit Tausenden von Token. Transformer-XL ermöglicht dabei eine bis 1800-Mal schnellere Verarbeitung von langen Texten als herkömmliche Transformer. Zudem können kontextuelle Abhängigkeiten besser und über weitere Distanzen innerhalb der Texte erkannt werden [Dai19]. Als Köpfe dieses Modells, bzw. als Klassifikatoren wird die sogenannte „LDA Topic Classification“ genutzt [Tan23].

NER

Im Bereich der Named Entity Recognition wird ähnlich zur MLTC mit 19 Mal meistens auf Transformer basierte Encoder gesetzt. Mit 17 Verwendungen wird auch im NER Bereich BERT präferiert. Ein weiterer oft genutzter Teil in der NER Model Architektur ist mit 15 Verwendungen der Klassifikator „Conditional Random Fields“ (CRFs). CRF sind statistische Modelle, die für eine Beobachtungssequenz die Wahrscheinlichkeiten für die Zuweisung einer Entität für jede Einheit der Sequenz berechnen. Dabei werden die einzelnen Einheiten der Sequenz nicht nur einzeln, sondern auch in Relation zu den anderen Einheiten der Sequenz bewertet. CRF nutzen eine exponentielle Vorgehensweise für diese Berechnung [Laf01]. Einen weiteren, oftmals verwendeten Teil der Architektur stellen bidirektionale, lange Kurzzeitgedächnisse, respektive BiLSTMs mit 14 Einsätzen dar. Ein zu erkennender Trend ist die Verwendung von Varianten der „Generative Pretrained Transformer“ (GPT). Ein GPT-Modell wurde, den Erkenntnissen dieser Literaturrecherche zu Folge, erstmals 2023 in einem wissenschaftlichen Kontext in den Ausarbeitungen von Wang et al.[Wan23c] und Yue et al.[Yue23] für die Aufgabenstellung der NER verwendet. Dabei konnten die Autoren sich dem derzeitigen SOTA Ergebnissen auf den von ihnen gewählten Testdatensätzen nähern, diese allerdings noch nicht erreichen oder übertreffen.

Das BINDER-Modell konnte auf den multilingualen Datensätzen ACE2004[Dod04] und ACE2005[Wal06] nach Angaben von Zhan et al. derzeitige SOTA-Ergebnisse erzielen. Auf dem englischsprachigen CoNLL03 erzielte das Modell einen F_1-Wert von 93,33%. Präzision und Recall liegen bei 93,08% und 93,57%. BINDER geht dabei einen neuen Ansatz im NER-Bereich und baut auf dem 2020 von Karpukhin et al. vorgestellten Bi-Encoder Modell auf [Zha22b]. Dabei werden zwei unabhängige Encoder, in diesem Fall BERT-Modelle für die Enkodierung verwendet. Ziel ist es dabei, die Fragen-Kontext Paare in einen Vektorraum zu transformieren, wobei ähnliche Fragen-Kontext Paare näher beieinander sind [Kar20]. In dem Anwendungsfall des BINDER-Modells werden für den ersten Encoder die Beschreibungen der jeweiligen Entitäten genutzt und für den zweiten die Texte in denen Entitäten erkannt werden sollen. Im Vektorraum sollen demnach Entitäten so nahe wie möglich an ihrem zugehörigen Entitätstypen sein und eine möglichst hohe Distanz zu nicht zugehörigen Entitätstypen aufweisen. Als Token-Einbettungen werden hier die finalen, verborgenen Zustände genutzt, welche von BERT dem jeweiligen Text zugewiesen wurden [Zha22b]. Die verborgenen Zustände beinhalten wichtige Informationen über die Beziehungen und den Kontext der Wörter des Textes und repräsentieren somit den gesamten Text. CLS- oder Klassifikations-Einbettungen werden vor jede Inputsequenz gestellt und beinhalten die für die weitere Verarbeitung notwendigen Informationen über die folgende Sequenz [Dev18].

Die besten Ergebnisse, die im Rahmen der SLR auf dem CoNLL03-Datensatz gefunden wurden, konnten mit dem 2021 vorgestellten BE-BLC-Modell von Affi et al. erzielt werden. Hier wurden Werte von 95,56% Präzision, 95,66% Recall und 95,57% auf dem F_1-Wert erreicht. Keine Höchstleistung, aber an SOTA-Ergebnisse angenäherte Werte konnten auf dem OntoNotes 5.0 Datensatz mit 88,54% Präzision, 89,91% Recall und 89,21% F_1 erzielt werden. Das Modell verwendet konkatinierte Worteinbettungen von BERT und ELMO, die von einem BiLSTM weiter verarbeitet werden. Letztendlich werden die Outputs durch CRF generiert [Aff21].

Kombination

Die Literaturrecherche ergibt, dass die Kombination von MLTC und NER bis jetzt nur einzeln erforscht wurde. Yan et al. haben das LANRTN-Modell entwickelt. Dieses besteht grundlegend aus den Architektur-Komponenten BERT, einem sogenannten R-Transformer, BiLSTM und CRF zur NER, sowie einem bidirektionalen Aufmerksamkeitsmechanismus. Bei diesem Ansatz wird NER genutzt, um die Ergebnisse der MLTC zu verbessern, d.h. dass NER keine direkte oder gemessene Disziplin dieses Modells ist. LANRTN konnte SOTA-Ergebnisse für die Metriken Recall und F_1-Wert auf den Datensätzen RCV1-V2[Lew04] und AAPD[Yan18] erzielen. Dabei konnten auf dem RCV1-V2 Datensatz 89% Recall und 89,3% F_1-Wert erreicht werden. Auf dem AAPD Datensatz konnten 68,9% Recall und 71,8% F_1-Wert erreicht werden. Die Präzision wurde auf beiden Datensätzen von einem anderem Modell, welches auf die Klassifizierung von Sätzen trainiert wurde, übertroffen [Yan22].

Metriken Die Ergebnisse von *SLRFF-4* wurden tabellarisch nach MLTC (Tabelle 3.2) und NER (Tabelle 3.3) kategorisiert. Erklärungen zu nicht bereits definierten Metriken können in den jeweiligen Ausarbeitungen gefunden werden. Diese wurden aufgrund ihrer Irrelevanz für diese Arbeit ausgelassen. Beispielsweise werden der erweiterte Recall, der erweiterte F_1-Wert und die erweiterte Präzision vorrangig im Bereich der verschachtelten NER-Aufgaben angewendet [Ju18].

Metriken	Referenzen
Micro-Präzision	[Fan23], [Li22c], [Sar20], [Kim22], [Ló20], [Ma21b], [Don22], [Cha22a], [Che20], [Len20], [Yan22], [Li20b], [Tan23], [Son22], [Jie20], [Zha21b]
Macro-Präzision	[Li22c]
Präzision bei k	[Xia19], [Mel22], [Ma21a], [Li22a]
Micro-Recal	[Fan23], [Li22c], [Abd21], [Sar20],[Rah23], [Kim22], [Ló20], [Ma21b], [Don22], [Cha22a], [Che20], [Len20], [Yan22], [Tan23], [Son22], [Jie20], [Zha21b]
Macro-Recall	[Li22c]
Micro-F_1-Wert	[Cha21], [Kem23], [Cha23], [Zha21a], [Fan23], [Li22c], [Liu21], [Yan22], [Pal20], [Lin23], [Din20], [Sar20], [Rah23], [Kim22], [Ló20], [Ma21b], [Don22], [Cha22a], [Che20], [Len20], [Tan23], [Son22], [Li22a], [Zha21b]
Macro-F_1-Wert	[Cha21], [Kem23], [Myl22], [Cha23], [Li22c], [Liu21], [Li20b], [Lin23], [Din20]
Beispiel basierter F_1-Wert	[Liu21]
Genauigkeit	[Rah23], [Ma21b], [Cha22a], [Yan22], [Tan23], [Jie20], [Sav23], [Din20]
HL	[Sar20], [Fan23], [Che20], [Liu21], [Pal20], [Son22], [Abd21]
Mean Reciprocal Rank	[Cha21], [Cha22b]
NDCG bei k	[Xia19], [Ma21a], [Li22a]
CO_2 Austoß des Trainings	[Sav23]
Jaccard-Koeffizient	[Lin23]

Tabelle 3.2: Die Verwendung verschiedener Metriken zur Evaluierung und Messung der Leistung verschiedener Modelle im Bereich der MLTC.

Metriken	Referenzen
Micro-Präzision	[Fu21], [Wan23b], [Men21], [Ji23], [Chi16], [Li21], [Zha22b], [Ju18], [Wan23c], [Jeh23], [Su23], [Li19], [Fre22], [Che21], [Wan23a], [Che22],[Zha21c], [Mou22]
Macro-Präzision	[Wos21], [Fre22], [Aff21]
Erweiterte Präzision	[Ju18]
Span-Präzision	[Mo23]
Micro-Recall	[Fu21], [Wan23b] , [Men21], [Ji23], [Chi16], [Li21], [Zha22b], [Ju18], [Wan23c], [Jeh23], [Su23], [Li19], [Fre22], [Che21], [Wan23a], [Che22], [Wos21], [Zha21c], [Nes22], [Mou22]
Recall bei k	[Yue23]
Macro-Recall	[Wos21], [Fre22], [Aff21]
Erweiterter Recall	[Ju18]
Micro-F_1-Wert	[Fu21], [Wan23b], [Li22b], [Men21], [Ji23], [Chi16], [ZG17], [Li21], [Zha22b], [Ju18], [Wan23c], [Jeh23], [Su23], [Li19], [Che21], [Wan23a], [Che22], [Wan21], [Zha21c], [Yan19], [ŽG18], [Pan18], [Zho20], [Mou22], [Sto22], [Tai20], [Nag22]
Span-Recall	[Mo23]
Macro-F_1-Wert	[Nag22], [Aff21], [Win21], [Vyc19], [Cui21], [Fre22], [Li20a]
Erweiterter-F_1-Wert	[Ju18]
Span-F_1-Wert	[Li23], [Mo23]
Genauigkeit	[Li20a], [Nag22]
Sätze pro Sekunde	[Zho20]

Tabelle 3.3: Die Verwendung verschiedener Metriken zur Evaluierung und Messung der Leistung verschiedener Modelle im Bereich der NER.

3.2.8 Diskussion

In Bezug auf *SLRFF-1*, haben sich das **HAWK-Modell** von Javeed [Jav23] und das **Transformer-LDA-MLTC-Modell** von Tang et al. [Tan23] hervorgehoben. Trotz der allgemein hohen Nutzung von BERT, wurde in beiden Modellen eine andere Grundkomponente für die Erstellung der Textrepräsentation gewählt. Das HAWK-Modell von Javeed wurde für den Einsatz im industriellen Kontext entwickelt und ist in der Lage, nach einem Training mit neuen Daten neue Label zu lernen, ohne bereits bekannte Label zu vergessen. Durch eine individuelle Verlustfunktion, die während des Trainings angewendet wird, ist das Modell auch bei Datensätzen, die nicht gut ausbalanciert sind, leistungsfähig. Darüber hinaus ist das Modell in der Lage, Texte unterschiedlicher Länge zu klassifizieren [Jav23]. Im Vergleich dazu ist das Transformer-LDA-MLTC-Modell von Tang et al. besonders gut beim Umgang mit langen Texten. Das Modell schneidet auf dem THUCNews-Datensatz in allen geprüften Bereichen sehr gut ab. Interessant ist die Kombination von älteren und neueren Techniken, da LDA mit modernen Transformer-Strukturen verknüpft wird [Tan23]. Ein direkter Vergleich der beiden Modelle ist schwierig, da sie auf unterschiedlichen Datensätzen getestet wurden und weitergehend auch nicht an den selben Metriken gemessen wurden. Trotzdem erscheint das HAWK-Modell durch seine Flexibilität und Anpassungsfähigkeit als gut geeignet für vielfältige Einsätze. Während das Transformer-LDA-MLTC-Modell besonders auf die Verarbeitung langer Texte ausgelegt ist, lässt die Fähigkeit des HAWK-Modells, variable Textlängen zu verarbeiten, darauf schließen, dass es in diesem Bereich ebenfalls effektiv sein sollte.

Im Kontext der *SLRFF-2* wurden das **BINDER-Modell** von Zhang et al.[Zha22b], sowie das **BE-BLC-Modell** von Affi et al[Aff21] vorgestellt. Im direkten Leistungsvergleich auf dem CoNLL03-Datensatz schneidet das BINDER-Modell besser ab. Durch die Kodierung des Textes über zwei BERT-Instanzen kann das Modell eine Fülle von Kontextinformationen in die Erkennung der benannten Entitäten einfließen lassen. Diese Eigenschaft deutet auf die Eignung des BINDER-Modells als robuste Lösung für industrielle Anwendungen hin. Das BINDER-Modell zeigt vor allem auf den mehrsprachigen Datensätzen gute Ergebnisse. Auf dem einsprachigen und weniger komplexen Datensatz CoNLL03 erzielt das BE-BLC-Modell noch bessere Ergebnisse. In diesem Modell wird ebenfalls viel Wert auf die bestmögliche Nutzung von Kontextinformationen durch die Verwendung einer Konkatenation von Einbettungen aus zwei verschiedenen Encodern gelegt.

SLRFF-3 kann teilweise durch das **LANRTN-Modells**[Yan22] beantwortet werden, jedoch konnte im Rahmen der SLR kein Ansatz gefunden werden, in dem ein MLTC und NER-Modell vorgestellt wurde, in dem der Fokus auf beiden Disziplinen lag. Dennoch stellt die Kombination der beiden Ansätze in Form des LANRTN-Modell einen interessanten Ansatz zur Bewältigung von MLTC dar. Es ist jedoch unklar, welche Ergebnisse und Leistungen für NER-Aufgaben erzielt werden könnten, wenn die Ausgabe entsprechend angepasst würde. Es besteht die Option, die NER-Komponente des Modells durch ein Modell zu ersetzen, das nachweislich gute Ergebnisse liefert, wobei jedoch nicht gewährleistet werden kann, dass das Modell weiterhin gleichwertige Ergebnisse für die MLTC liefert.

Die verwendeten Metriken sind im Bereich der MLTC wesentlich breiter gefächert als im Bereich der NER. Dennoch werden F_1, Präzision und Recall in beiden Fällen am häufigsten verwendet. Makro- und Mikro-F_1-Werte werden bei der MLTC häufiger verwendet, was auf die größere Relevanz bei nicht-binären Problemen zurückzuführen sein könnte. Für die MLTC wurde im Gegensatz zur NER auch die HL als Metrik zur Messung von Fehlern und die NDCG bei k zur Messung der Ergebnisqualität verwendet. Dabei ist zu beachten, dass der NDCG nicht notwendigerweise schlechtere Werte für fehlerhafte Vorhersagen liefert, da die kumulierten Werte als Ganzes bewertet werden und nicht im direkten Vergleich mit dem tatsächlichen Wert.

3.3 Ableitung des Konzeptes

Die zuvor definierten Anforderungen an das System bilden die Grundlage für die Auswahl der einzusetzenden Technologien, Algorithmen und Methoden. Die Auswahl erfolgt anhand der funktionalen und nicht-funktionalen Anforderungen. Da in den vorgestellten Arbeiten keine Angaben zu Rechenkapazität und Rechenzeit gemacht wurden, können *NFA-3* und *NFA-1* nicht als Entscheidungskriterien berücksichtigt werden. Da jedoch alle als neuronale Netze mit zugrundeliegendem Encoder eine grundsätzlich ähnliche Struktur aufweisen, werden diese Aspekte mangels Alternativen zunächst vernachlässigt. So kann erst im Rahmen der Implementierung der Ansätze darauf geachtet werden, die Modelle möglichst effizient zu gestalten.

Die Beantwortung von *SLRFF-3* befasst sich mit integrierten Ansätzen zwischen MLTC und NER und könnte somit die Grundlage für ein mögliches zusammenhängendes Modell für beide Aufgaben bilden. Allerdings wurde im Rahmen der SLR eine zu geringe Forschungsgrundlage festgestellt, so dass eine Weiterverfolgung dieses Ansatzes derzeit nicht ausreichend empirisch gestützt ist. Die Diskussion der Ergebnisse von *SLRFF-1* zeigt, dass das HAWK-Modell durch die Verarbeitung variabler Textlängen und die Modularität einerseits die Anforderungen bezüglich der Textlänge erfüllt und

andererseits weitere Möglichkeiten für die Anwendung im kommerziellen Umfeld eröffnet. In drei der vier getesteten Datensätze, die zwischen 5 und 33 Label enthielten, wurden im Vergleich zu anderen Modellen hohe Genauigkeiten erzielt. In zwei Fällen wurden neue Bestwerte erreicht [Jav23]. Wie bereits erwähnt, kann nach Ribeiro et al. die Genauigkeit den Anschein der Modellleistung verfälschen [Rib20]. Jedoch deuten die hohen Werte in der Genauigkeit auf eine relativ geringe Fehlerquote hin, die für die Erfüllung der Anforderungen der *NFA-6* notwendig ist. Ebenso wird die Erfüllung von *NFA-5.1* durch den Spielraum der möglichen Labelanzahlen und *NFA-2* durch die gegebene Flexibilität unterstützt. Diese Anforderung beruht auf der Prämisse, dass eine geringe Anzahl von Fehlern in den Modellvorhersagen ein Indikator für eine hohe Genauigkeit ist. Der Bereich der Label ist größer als in den Anforderungen beschrieben, deckt aber den geforderten Bereich von 10 bis 20 Labeln ab. Die NER-Komponente des Modells wird der Diskussion der *SLRFF-2* folgend dem Konzept des BE-BLC-Modells folgen. Das Modell überzeugt auf Grund der hohen Ergebnisse auf den CoNLL03 und OntoNotes 5.0 Datensätzen, welche beide mit vier möglichen Entitäten im Bereich von *NFA-5.2* liegen. Ebenso legen die hohen Werte in Präzision und Recall die Erfüllung von *NFA-6* nahe. Durch die fehlenden Angaben der Genauigkeit in der vorgestellten Arbeiten kann bezüglich dieser Metrik kein direkter Vergleich gezogen werden.

Es konnte kein Datensatz gefunden werden, der sowohl für das Training von NER- als auch von MLTC-Modellen geeignet ist, so dass *NFA-4* nicht erfüllt werden kann. Die Auswahl der separaten Datensätze wird durch die Bedingungen von *NFA-4* und *NFA-5* bestimmt. Um eine ähnliche Trainingsgrundlage für beide Modelle schaffen zu können, wird der bereits erwähnte CoNLL03 Datensatz für die Aufgabenstellung der NER genutzt und der Reuters 21578 Datensatz für die Aufgabenstellung der MLTC. Der CoNLL03 Datensatz wurde aus einem von Reuters erstellten Korpus geschaffen, zu dem ebenfalls der R21578 Datensatz gehört [TKS03]. Diese Anzahl der Label des R21578 Datensatzes muss auf eine Anzahl von 20 reduziert werden, um den Anforderungen gerecht zu werden. Die Entscheidung welche Label genutzt werden, wird anhand der höchsten Vorkommnisse der Label im Datensatz entschieden.

Da für die HAWK- und BE-BLC-Modelle kein Quellcode veröffentlicht wurde, werden Modelle anhand der Arbeiten implementiert. Die in den jeweiligen Arbeiten beschriebene Architektur dieser Modelle dient dabei als Leitfaden für die Realisierung der Modelle. Als Basis werden die online verfügbaren vortrainierten Modelle BERT[1], ELMO[2] und USE[3] verwendet. Die Entwicklung der Modelle erfolgt mit Hilfe des PyTorch-Frameworks und der Programmiersprache Python. Ein BiLSTM-Encoder wird verwendet, um die generierten Einbettungen zu verarbeiten. Das HAWK-Modell, das sowohl Transformer-

1 https://huggingface.co/bert-base-uncased
2 https://allenai.org/allennlp/software/elmo
3 https://tfhub.dev/google/universal-sentence-encoder/4

als auch BiLSTM-Klassifikatoren in seine Architektur integriert, zeigt eine bessere Performance mit den BiLSTM-Klassifikatoren, weshalb diese bevorzugt werden. Im Gegensatz dazu verwendet das BE-BLC Modell eine einzige CRF-Schicht für die Vorhersage der Ergebnisse. Schließlich werden beide Modelle zu einem umfassenden System zusammengeführt, welches die Eingaben verarbeitet und sowohl die im Text erkannten, benannten Entitäten als auch die vorhergesagten Label ausgibt.

Für das Training des Modells werden die angegeben Parameter verwendet, welche von Affi et al.[Aff21] zum CoNLL03 Datensatz angegeben wurden. Dazu zählt auch der frühere Abbruch des Trainings, falls über 25 Iteration hinweg keine Verbesserung des kumulierten Wertes der Fehlerfunktion gemessen werden kann. Das frühzeitige Abbrechen, auch „early stopping“ genannt, dient dem Zweck den Punkt abzupassen, an dem das Modell für unbekannte Daten das beste Resultat erzielt. Nach einer gewissen Zeit neigen NN dazu trotz einer steigenden Genauigkeit auf dem Trainingsdatensatz eine schlechtere Leistung auf unbekannten Daten zu erreichen. Dies wird auch als Überanpassung bezeichnet [Pre02]. Um dies zu überprüfen, wird neben dem Trainings- und dem Testdatensatz ein Validierungsdatensatz benötigt, auf den ebenfalls die Verlustfunktion angewendet wird, aber keine Anpassung der Gradienten aufgrund der Ergebnisse erfolgt. Anpassungen der Parameter des MLTC-Modell Trainings werden auf die Parameter beschränkt, welche in der Arbeit von Javeed [Jav23] zum Training anderer Datensätze verwendet wurden. Die Iterationen über den gesamten Trainingsdatensatz werden auf Grund des größeren Datensatzes dem NER-Modell folgend auf 170 angesetzt. Um Überpassung zu vermeiden, wird ebenfalls ein Abbruchmechanismus nach 25 Iterationen ohne Verbesserung eingesetzt. In beiden Ausarbeitung wird der Adam-Optimierer verwendet. Der Adam-Optimierer stellt ein Algorithmus dar, der die Parameter während des Trainings eines Modells automatisch anpasst, um bessere Ergebnisse zu erzielen [Kin14].

3.4 Evaluationskonzept

Um einen direkten Vergleich und eine Reproduzierbarkeit der Ergebnisse des Evaluationskonzeptes zu ermöglichen, muss zunächst ein Datensatz identifiziert werden, der in konsistenter Form vorliegt. Die Evaluierung sollte nach einem abgeschlossenen Trainingsprozess vorgenommen werden. Zur Ermöglichung des Vergleichs zwischen verschiedenen MLTC und NER Lösungen, müssen die Metriken in jedem Modell auf die gleiche Weise gemessen werden. Um eine Validität und die Generalisierung der erreichten Ergebnisse zu gewährleisten, dürfen keine Daten des für die Evaluierung verwendeten Testdatensatzes von dem zu evaluierenden Modells bereits „gesehen“ worden sein [Zhe15]. Dies bedeutet, dass es keine Überschneidung zwischen Test- und Trainingsdatensatz geben darf. Das Konzept deckt möglichst unterschiedliche Sichtweisen auf die erzielten Ergebnisse ab,

um unterschiedlichen Anforderungen gerecht zu werden. Es werden reproduzierbare und aussagekräftige Metriken verwendet, die in den untersuchten wissenschaftlichen Publikationen verwendet wurden, was auf eine allgemeine Akzeptanz hindeutet. Die Analyse der *SLRFF-4* Ergebnisse zeigt, dass verschiedene Metriken in MLTC und NER unterschiedlich häufig genutzt werden, was eine differenzierte Betrachtung in diesem Konzept erfordert. Trotz dieser Unterschiede existieren gemeinsame Metriken, die in beiden Disziplinen relevant sind.

In beiden Bereichen, MLTC und NER, wird der Mikro-F_1-Wert am häufigsten verwendet, was auf eine allgemeine Akzeptanz dieser Metrik hindeutet. Sie ermöglicht einen direkten Leistungsvergleich mit existierenden Modellen. Die Bedeutung von Präzision und Recall, insbesondere im Kontext der Vermeidung inkorrekt positiver oder negativer Werte, unterstreicht ihre Unverzichtbarkeit. Der F_1-Wert, als Kombination beider Metriken, dient dabei als Indikator für ein ausgewogenes Verhältnis zwischen Präzision und Recall. Eine Verschiebung des β-Wertes im F_β-Wert erscheint, aufgrund der Nicht-Toleranz inkorrekt positiver und negativer Werte, als nicht zielführend. Diese Metriken bieten den Vorteil, dass sie relativ einfach und auch ohne tiefgreifende Fachkenntnisse interpretierbar sind. Der Makro-F_1-Wert, obwohl weniger häufig verwendet, ist sowohl in MLTC als auch in NER relevant. Der Makro-F_1-Wert wurde 9-mal für die Bewertung von MLTC-Modellen und siebenmal für die Bewertung von NER-Modellen im Kontext der SLR verwendet. Die geringere Verwendung könnte darauf hindeuten, dass in einigen Arbeiten der Schwerpunkt der Bewertung eher auf der Gesamtleistung des Modells über alle Klassen hinweg als auf einer ausgewogenen Leistung in jeder einzelnen Klasse liegt. Ungeachtet dessen ist der Makro-F_1-Wert ein wichtiger Indikator für die Gesamtleistung und sollte bei der Bewertung berücksichtigt werden, um die Leistung aus verschiedenen Blickwinkeln zu betrachten. Das gleiche Argument kann auf die Makro-Präzision und den Makro-Recall angewendet werden, weshalb diese in das Konzept integriert werden.

Für die Evaluierung eines MLTC-Modells, sollte für eine zusätzliche Sichtweise auf die erbrachten Leistungen des Modells der siebenmal verwendete HL hinzugezogen werden. Der HL zeigt die Fehlerquote der inkorrekt klassifizierten Label in Relation zur Gesamtzahl der Label.

Das aufgestellte Konzept kann je nach benötigten Anforderungen durch weitere Metriken erweitert werden. So ist es für den Kontext dieser Arbeit für die Erfüllung von *NFA-1* von Bedeutung die durchschnittliche Verarbeitungszeit eines Dokumentes zu messen. In einem generellen Kontext könnte die durchschnittliche Zeit für den gesamten Datensatz berechnet werden, hier werden allerdings nur Texte zwischen 5000 und 6000 Zeichen in die Berechnung miteinbezogen. Ebenso hängt die Relevanz der gemessenen Metriken vom gegebenen Kontext ab, so sind im Bezug auf *NFA-6* vor allem Mikro-Präzision und -Recall von Bedeutung.

4 Realisierung

In diesem Kapitel wird die Umsetzung der im Abschnitt 3.3 beschriebenen Modelle vorgenommen. Dazu werden zunächst die identifizierten Datensätze aufbereitet. Anschließend werden die Modelle implementiert und mit den aufbereiteten Daten trainiert. Die trainierten Modelle werden anhand des zuvor erstellten Evaluationskonzepts ausgewertet. Abschließend wird auf die Erfüllung bzw. Nichterfüllung der funktionalen und nicht-funktionalen Anforderungen eingegangen.

4.1 Aufbereitung der Daten

Der „Reuters-21578, Distribution 1.0“[1]-Datensatz (R21578) besteht aus mehreren Spalten, wobei für diese Arbeit vor allem die Aufteilung in Test- bzw. Trainingsdaten, die vergebenen Label sowie die identifizierten Orte, Personen, Organisationen und Börsen von Interesse sind. Als Aufteilung wird der „Lewis-Split“ verwendet, der laut „readme.txt“-Datei[2] des Datensatzes für Klassifikationsaufgaben verwendet werden soll und somit einen Vergleich mit anderen Arbeiten ermöglicht. Die Verwendung des Datensatzes in dieser Arbeit beschränkt sich ausschließlich auf die Entwicklung und Leistungsbewertung des beschriebenen Modells in Form eines Prototyps, der für rein akademische Zwecke trainiert wird. Im Datensatz werden die zugewiesenen Label als Topics bezeichnet. Die Label decken ein breites Spektrum an wirtschaftlichen Themen in verschiedenen Bereichen ab. Zu den am häufigsten vorkommenden Labeln gehören Ertrag („earn“), Akquisitionen („acq“), Getreide („grain“), Rohöl („crude“) oder Zinsen („interest“). Für die Aufbereitung des MLTC Datensatzes werden zusätzlich lediglich die 20 meistgenutzten Label in Betracht gezogen. Somit wurden für das Training irrelevante Daten, welche diese Label nicht enthalten, entfernt. Die Spanne der Vorkommnisse reicht dabei von 3964 bis 105. Somit ergeben sich 7143 Texte für das Training und 2750 für das Testen des Modells. Nach dem Beispiel von Huang et al. werden 1000 zufällig ausgewählte Texte aus dem Trainingsdatensatz für die Validierung verwendet [Hua21]. Die Aufteilungen wurden in drei entsprechenden JSON-Dateien gespeichert.

1 https://www.daviddlewis.com/resources/testcollections/reuters21578/
2 http://www.daviddlewis.com/resources/testcollections/reuters21578/readme.txt

Der CoNLL03[1] Datensatz ist in der vorliegenden Form bereits in einzelne Wörter aufgeteilt, wobei jedem Wort eine oder keine Entität zugeordnet wurde. Ebenso ist in der vorliegenden Form bereits eine Aufteilung in Trainings-, Test- und Validierungsdatensätze vorgenommen worden. Der Datensatz wurde in ein JSON-Dateiformat konvertiert, um ein einheitliches Format mit dem vorverarbeiteten Datensatz R21578 zu erhalten. Der Datensatz enthält Sätze aus Nachrichtentexten, die in vier Kategorien von benannten Entitäten annotiert sind. Die Entitäten sind Personen („PER“), Organisationen („ORG“), Orte („LOC“) sowie sonstige Entitäten („MISC“). Die Annotation folgt dem „Inside Outside Beginning“ (IOB) Schema, wobei „Outside “ (O) für keine Entität steht, „Beginning “ (B) für den Anfang einer Entität und „Inside“ (I) für das Innere einer Entität. Ein Beispiel der Annotation kann in Abbildung 4.1 gefunden werden.

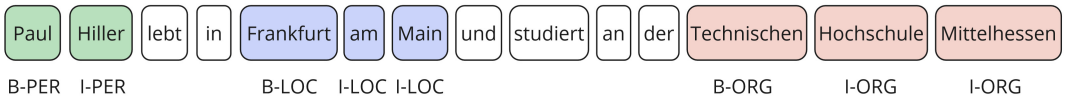

Abbildung 4.1: IOB-Annotation Beispielsatz

4.2 Implementierung

Die Implementierung der beiden Modelle[2] wurde in unterschiedlichen Umgebungen vorgenommen. Für die Handhabungen der Umgebungen und der damit verbundenen Abhängigkeiten wurde Conda[3] verwendet. Grundsätzlich werden beide Modelle mit Hilfe von PyTorch[Pas19] implementiert. Die einzelnen Komponenten der jeweiligen Modelle werden zunächst separat in entsprechenden Klassen umgesetzt und letztendlich in einer Modell-Klasse zusammengesetzt. Diese Klasse kann daraufhin in Skripten zum Training und Ausführen des jeweiligen Modells aufgerufen werden. Die Implementierung der Modelle orientiert sich an den vorgestellten Konzepten des HAWK-Modells[Jav23] und des BE-BLC-Modells[Aff21]. Trotz möglicher Abweichungen bei der Implementierung in dieser Arbeit werden die Modelle weiterhin als HAWK- und BE-BLC-Modell bezeichnet, da die Grundstruktur identisch ist.

1 https://huggingface.co/datasets/conll2003
2 Der Quellcode kann unter `https://hbx.fhhrz.net/getlink/fiKRGXCa9oZ7at3e7AaUNC/mltc_ner_bachelorthesis.zip` eingesehen werden.
3 https://docs.conda.io/en/latest/

4.2.1 HAWK-Modell

Das MLTC-Modell, welches den Beschreibungen des HAWK-Modells nach in dieser Arbeit implementiert wurde, besteht zum einen aus einer Einbettungsschicht, einem BiLSTM in Verbindung mit einer dichten, auch voll-verknüpften Schicht genannt und den abschließenden Klassifikationsköpfen in der BiLSTM-Variante. Innerhalb einer dichten Schicht ist jedes Eingabeneuron mit jedem Ausgabeneuron verbunden, wodurch jede Ausgabe durch jede Eingabe und die Gewichtungen der jeweiligen Verbindungen beeinflusst wird [Jos21]. In der Einbettungsschicht wird ein eingegebenes Dokument zunächst nach einer vorgegebenen Zeichenanzahl in mehrere Blöcke aufgeteilt. Die Blöcke werden dann mit Hilfe des „Universal Sentence Encoder“ (USE) in der „DAN“-Variante eingebettet. Der USE wandelt die Wörter in Vektoren um, berechnet den arithmetischen Mittelwert der Vektoren und verarbeitet diesen durch ein NN, um eine Repräsentation des Textes in Form von $|Wörtern| \times 512$ zu erzeugen [Cer18]. Da USE nicht in einer in PyTorch implementierten Version vorliegt, muss dieses Modell mit Tensorflow[1] importiert werden. Die Ausgabe wird in ein Format konvertiert, das in PyTorch weiterverarbeitet werden kann. Zuletzt werden den Einbettungen Leerwerte hinzugefügt um eine Matrix in Form der $Maximale - Blockanzahl \times 512$ zu erhalten. Dies geschieht, um zur Verarbeitung mehrerer Dokumente einen Tensor der Größe $Stapel - Größe \times Maximale - Blockanzahl \times 512$ bilden zu können. In einem weiteren Schritt wird zur Eliminierung überflüssiger Leerwerte vor der Eingabe in das BiLSTM die $MaximaleBlockanzahl$ innerhalb der gebildeten Matrix auf die höchste verwendete Blockanzahl des Stapels reduziert. Die Hydranet-Architektur[Mul18], die in der Arbeit von Javeed[Jav23] verwendet wurde, ermöglicht es außerdem, weitere Köpfe zu trainieren und parallel dazu die bereits erlernten Label nicht zu vergessen. Für den in dieser Arbeit realisierten Prototyp wurde keine konkrete Funktion dafür geschaffen, da diese Modularität nicht zu den Zielen der Arbeit gehört. Sie kann jedoch im Nachhinein ohne Änderung des Modells hinzugefügt werden. Die aus der Dense/BiLSTM-Schicht extrahierten Merkmale werden an die Klassifikationsköpfe übergeben, wobei jeder Klassifikationskopf die Überprüfung der Zuweisung eines Dokuments zu einer Klasse übernimmt. Die BiLSTM-Header bestehen aus zwei getrennten BiLSTMs und drei voll verknüpften Schichten, die die Dimension der Ausgabe schrittweise bis auf 1 reduzieren. Das Ergebnis R kann nur mit $R \geq 0.5$ interpretiert werden, wobei Werte über 0.5 als Zuweisung des Label angesehen werden. Je näher der Wert an 1 oder 0 liegt, desto wahrscheinlicher ist nach dem Modell die Vergabe oder Nicht-Vergabe des Label.

1 https://www.tensorflow.org/

4.2.2 BE-BLC-Modell

Die Abhängigkeiten der Umgebung für die Implementierung des BE-BLC-Modells unterscheiden sich hauptsächlich von der Umgebung des HAWK-Modells durch die Versionen von PyTorch und Python. Es wurde PyTorch Version 1.12.1 und Python 3.8.18 verwendet, anstatt der Versionen 2.1.0 bzw. 3.11.5. Diese älteren Versionen werden verwendet, um die AllenNLP[Gar18] Bibliothek nutzen zu können. Die aktuellste Version basiert auf dieser PyTorch Version und ist ohne weitere Komplikationen nicht mit höheren PyTorch und Python Versionen kompatibel. Die AllenNLP Bibliothek erlaubt die Nutzung des ELMo Modells[1], welches zusammen mit BERT für die Worteinbettungen des Modells verantwortlich ist. Die Konkatenation dieser Einbettungen bildet die erste Komponente des Modells. Für die Konkatenation der beiden Einbettungen, muss zunächst die Form der Einbettungen untersucht werden. Das ELMo-Modell zerlegt einen Satz oder Text in einzelne Wörter, die in diesem Fall die Token darstellen. Bei der Berechnung der Einbettung der Token wird der Kontext berücksichtigt [Pet18]. Dies führt zu einer Darstellung von $S = \{x_1, x_2, ..., x_n\}$ und $E_S = \{e_{x_1}, e_{x_2}, ..., e_{x_n}\}$, wobei S einen Satz oder Text mit den jeweiligen Wörtern x_n und E_S die Einbettung des Textes mit den zugehörigen Worteinbettungen e_{x_n}, einem Vektor bestehend aus 1024 Werten, darstellt. Praktisch wird also für jeden Text ein Tensor der Größe $|Worte| \times 1024$ erzeugt.
BERT verwendet WordPiece für die Tokenisation und fügt spezielle Token z.B. am Anfang („[CLS]“), am Ende eines Satzes („[SEP]“) oder für Leerwerte („[PAD]“) ein [Dev18]. WordPiece unterteilt Wörter in bestimmte Teilwörter, die aus einem Vokabular von Einzelwörtern und Wortfragmenten bestehen [Wu16]. So wird der Satz *„This is an example sentence for my bachelors thesis on MLTC and NER.“* zu *„[CLS], this, is, an, example, sentence, for, my, bachelor, ', s, thesis, on, ml, ##tc, and, ne, ##r, ., [SEP]“*. Die Eingabe in BERT sollte im Idealfall aus nicht mehr als einem oder zwei Sätzen bestehen. Die BERT-Texteinbettungen E_{T_S}, wobei die eingegebenen Text aus ein oder zwei Sätzen bestehen sollten, können als $E_{T_S} = \{e_{t_1}, e_{t_2}, ..., e_{t_m}\}$ zu einer Tokenfolge von $T_S = t_1, t_2, ..., t_m$ mit $m \geq n$ dargestellt werden. Diese Einbettungen haben die Form $|Token| \times 768$ [Dev18]. Um eine Konkatenation zu ermöglichen, müssen sowohl die BERT- als auch die ELMo-Einbettungen in der ersten Dimension die gleiche Menge haben, daher werden die BERT-Token-Einbettungen eines Wortes aggregiert, um eine Wortrepräsentation in Form eines Vektors der Größe 1×768 zu erhalten. Dies ermöglicht die Kombination der beiden Einbettungen entlang der ersten Dimension zu einer konkatenierten Einbettung in Form von $|Worte| \times 1792$. Die Verarbeitung eines Stapels von Sätzen erfordert die gleiche Anzahl von Wörtern oder Token in jedem Satz des Stapels. Daher werden sowohl bei den BERT- und ELMo-Einbettungen als auch bei der Weiterverarbeitung der konkatenierten Einbettungen die Sätze mit Leerwerten

1 https://allenai.org/allennlp/software/elmo

entsprechend der höchsten Anzahl von Wörtern oder Token eines Satzes des Stapels aufgefüllt. Diese Ergebnisse werden durch ein BiLSTM mit zwei versteckten Schichten pro Richtung verarbeitet. Jede verborgene Schicht hat 200 Neuronen. Dadurch wird eine bessere Repräsentation des Kontextes erreicht [Aff21]. Dem von Affi et al. präsentierten Modellaufbau folgend wird ein CRF zur Berechnung der Zugehörigkeit einer Entität zu einem Wort eingesetzt. Dabei kommt der Viterbi-Algorithmus[For73] zum Einsatz. Um die Effizienz während der Verarbeitung eines Stapels an Sätzen zu erhöhen wird PyTorch's Mutliprocessing[1] Bibliothek verwendet. Durch den Einsatz der Bibliothek kann die Berechnung der Entitäten parallel ausführen werden. Während des Trainings ist es nicht möglich diese Berechnungen auf diese Art und Weise zu parallelisieren, da die Informationen zu den Gradienten nicht über mehrere Prozesse hinweg geteilt werden können.

4.2.3 Zusammenführung der Modelle

Die Zusammenführung der beiden Modelle zu einem System wird über Docker[Mer14]-Container realisiert. Dazu werden Python-Programme erstellt, welche über eine REST[Fie00]-Schnittstelle Text im JSON-Format empfangen können und die Vorraussagen in Textform zurückgeben. Zusätzlich wird ein API-Gateway erstellt, welches die jeweiligen Schnittstellen der Modelle einzeln oder über eine gemeinsame Route aufruft und somit eine geschlossene Rückgabe erzeugen kann. Diese drei Schnittstellen werden in Container umgewandelt und in einer Multi-Container Anwendung zusammengefasst. Im Rahmen der Erstellung der Modell-Container können Konfigurationsdateien (siehe Listing 4.1 und **??**) als Umgebungsvariablen festgelegt werden. Diese sollte sowohl die Reihenfolge der gelernten Label und Entitäten als auch die Dateipfade zu den .pth[2]-Dateien enthalten, welche die zu verwendenden Zustände bzw. Gewichte der Verbindungen zwischen den Neuronen enthalten. Wird keine Konfigurationsdatei angegeben, so wird standardmäßig eine Konfiguration verwendet, die die Entitäten, Label und besten erreichten Zustände des im Abschnitt 4.3 durchgeführten Trainings lädt. Nach der Initialisierung der Container kann über die Route „/label-and-entities“ ein Text geschickt werden, wobei in dieser Route parallel Anfragen an die beiden Modellcontainer geschickt werden. Das HAWK-Modells erzeugt Vorraussagen mit Werteb zwischen 0 und 1 für jedes mögliche Label. Alle Werte größer als 0,5 werden als Zuweisung betrachtet und über die in der Konfigurationsdatei angegebenen Label in das entsprechende Label in natürlicher Sprache umgewandelt. Diese werden dann zurück an die Verteiler-Schnittstelle gegeben. Der an den Container des BE-BLC-Modells geschickte Text, wird durch Interpunktionszeichen in einzelne Sätze zerlegt. Anschließend wird jeder Satz nach Leerzeichen in

1 https://pytorch.org/docs/stable/multiprocessing.html
2 https://pytorch.org/docs/stable/notes/serialization.html

einzelne Wörter zerlegt, die gleichzeitig die Token des ELMo-Modells darstellen. Für jeden Satz gibt das Modell eine geordnete Menge numerischer Werte aus, die der Anzahl der Wörter entspricht. Eine Null repräsentiert keine Entität. Die Zahlen werden den entsprechenden Bezeichnungen zugeordnet, die über die Konfigurationsdatei in natürlicher Sprache eingegeben wurden. Die Position einer Entitätsbezeichnung innerhalb der Zahlenmenge entspricht der Position eines Wortes innerhalb des zugehörigen Satzes. Die Entitäten werden nach ihrer Entitätsklasse sortiert und zusammen mit den zuvor erkannten Label zurückgegeben. Nachdem beide Antworten der Modellcontainer in der Verteiler-Schnittstelle angelangt sind, werden diese gemeinsam zurückgegeben. Eine Darstellung des zusammengeführten Systems ist in Abbildung 4.2 zu finden.

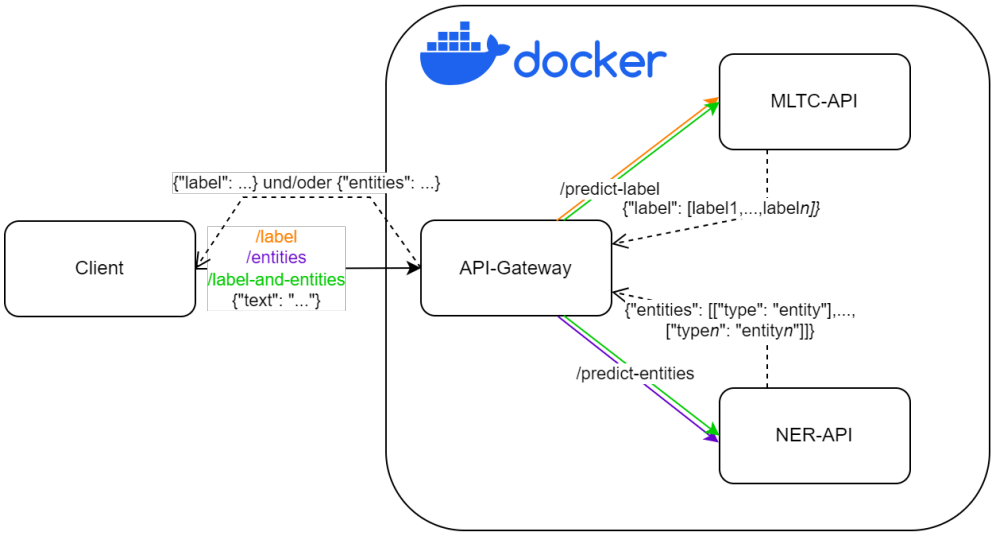

Abbildung 4.2: Darstellung der Zusammenführung der MLTC- und NER-Modelle in Form einer containerisierten API. [doc23]

```
{
  "name": "custom_r21578_mltc",
  "embedding_dimension": 512,
  "block_size_in_chars": 100,
  "max_document_length_chars": 13
    397,
  "batch": 16,
  "epochs": 170,
  "patience": 25,
  "master_address": "localhost",
  "master_port": 35464,
  "world_size": 8,
  "best_model": "
    saved_model_dictionaries/
    best_model.pth",
  "preprocessed_dataset": {
    "path": "data.reuters21578
    _custom_lewis_split.
    preprocess",
    "class": "R21578CustomDataset
    "
  },
  "label_map": {
    "earn": 0,
    "acq": 1,
    ...
     "nat-gas": 19
  }
```

Listing 4.1: JSON-Konfiguration des MLTC-Modells.

```
{
  "name": "Conll2003_ner",
  "dropout": 0.5,
  "batch_size": 64,
  "epochs": 170,
  "learning_rate": 1e-4,
  "patience": 25,
  "master_address": "localhost",
  "master_port": 35463,
  "world_size": 3,
  "best_model": "
    saved_model_dictionaries/
    be_blc_best.pth",
  "preprocessed_dataset": {
    "path": "data.
    ProcessedDataset",
    "class": "Connl2003Dataset"
  },
  "entity_ix": {
    "O": 0,
    "B-PER": 1,
    "I-PER": 2,
    ...
    "STOP_TAG": 10
  }
```

Listing 4.2: JSON-Konfiguration des NER-Modells.

4.3 Training

Für das Training des HAWK-Modells wurde von Javeed eine gewichtete Binary Cross-Entropy Verlustfunktion vorgestellt [Jav23]. Der klassische BCE betrachtet die Multi-Label-Klassifikation als binäre Probleme, die nach Klassen unterteilt sind. Dabei wird sowohl der Vergleich zwischen Vorhersage und tatsächlicher Zuordnung als auch zwischen Vorhersage und tatsächlicher Nicht-Zuordnung berücksichtigt. Jede Klasse wird in der klassischen BCE gleich gewichtet [Wu20]. Der gewichtete Binary Cross-Entropy Loss (siehe Listing 4.3) ist eine Variante der BCE-Verlustfunktion, bei der den Labeln unterschiedliche Gewichte zugewiesen werden, um Ungleichgewichte in der Häufigkeit der Label in den Trainingsdaten zu berücksichtigen. Diese Gewichte werden so berechnet, dass sie in umgekehrtem Verhältnis zur Häufigkeit der jeweiligen Klasse stehen, wodurch seltenere Klassen stärker gewichtet werden. Beim Training eines Modells mit dieser Gewichtung werden Vorhersagen für unterrepräsentierte Klassen stärker berücksichtigt, was zu einem ausgewogeneren Lernprozess führt. Um das Training auf mehrere CPUs zu parallelisieren wurde PyTorchs „Distributed Data Parallel“[1] verwendet. Dabei wurde das Training in acht parallelen Prozessen ausgeführt, wodurch die Zeit einer Iteration von rund 558 Sekunden auf rund 295 Sekunden reduziert werden konnte. Insgesamt konnte somit die Trainingszeit von etwa 26,3 Stunden auf etwa 13,9 Stunden verringert werden. Die für das Training verwendeten Parameter können in Tabelle 4.1 und die Verringerung der Fehlerfunktion über die Iterationen hinweg in Abbildung 4.3 eingesehen werden. Die genutzten Parameter werden in einer JSON-Datei gespeichert, wobei der Pfad zur Konfigurationsdatei als Argument dem Trainingsskript übergeben werden kann. Im Rahmen des Trainings wurde ≈ 0.324 als niedrigster Wert auf dem Validierungsdatensatz gemessen, bevor das Modell für 25 Iterationen keine weitere Verbesserungen mehr aufgewiesen hat.

Parameter	Wert
Einbettungsgröße	512
Blockgröße in Zeichen	100
Maximale Dokumentenlänge in Zeichen	13397
Größe Dokumentenstapel (Batch)	16
Iterationen über Datensatz (Epochen)	170
Geduld	25

Tabelle 4.1: Übersicht der Modellparameter

1 https://pytorch.org/docs/stable/generated/torch.nn.parallel.DistributedDataParallel.html

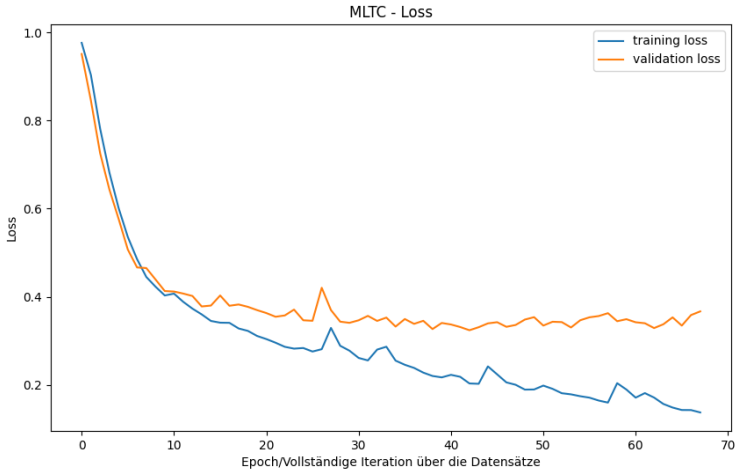

Abbildung 4.3: Werte der Verlustfunktion für die Trainings- und Validierungsdatensätze des MLTC-Modells. Die Werte wurden für eine vollständige Iteration des jeweiligen Datensatzes gemessen.

```
1 import torch
2 class WeightedBCELoss:
3     EPSILON: float = 1e-10  # Um Divisionen durch 0 zu verhindern
4     def __init__(self, number_of_categories: int):  # Initialisierung
5         self.K: int = number_of_categories  # K = Anzahl Kategorien
6     def __call__(self, predicted_label: torch.Tensor, true_label: torch.
      Tensor): # Aufruf der Klasse, Eingaben sind die Voraussagen von N-
      Dokumenten zu einem Label und die korrekten Zuweisungen des Label
      fuer die Dokumente
7         true_label_list = true_label.tolist()  # Umwandlung der wahren
      Label in eine Liste
8         Nk = true_label_list.count(1)  # Nk repraesentiert wie viele
      Label tatsaechlich wahr sind
9         if (N - Nk) == 0:
10            weight_k0 = N / (self.EPSILON * self.K)
11        else:
12            weight_k0 = N / ((N - Nk) * self.K)
13        if Nk == 0:
14            weight_k1 = N / (self.EPSILON * self.K)
15        else:
16            weight_k1 = N / (Nk * self.K)
17        weighted_bce = weight_k0 * (1 - true_label) * torch.log(1 -
      predicted_label) + weight_k1 * true_label * torch.log(predicted_label
      )  # Berechnung des gewichteten BCEs fuer jedes Dokument
18        E = -torch.sum(weighted_bce) / N  # Der Absolutbetrag des
      Durchschnitts der berechneten Werte
19        return E
```

Listing 4.3: Implementierung der gewichteten BCE-Verlustmethode. „weight_k0“ ist das Gewicht für die wahre negative Zuordnung und „weight_k1“ das Gewicht für die wahre positive Zuordnung.

Die für das Training des BE-BLC-Modells auf dem CoNLL03-Datensatz verwendeten Parameter sind in Tabelle 4.2 dargestellt. Auch in diesem Fall wird ein Pfad zu einer JSON-Konfigurationsdatei, welche die Modellparameter enthält, dem Skript vor Beginn des Trainings als Argument übergeben. Um das Training zu beschleunigen, wurde erneut PyTorchs „Distributed Data Parallel“ verwendet. Allerdings konnten nur drei Prozesse parallel ausgeführt werden, ohne den Computer zu überlasten. Ein Prozess benötigt dabei zwischen ca. 28% und ca. 33% der vorhandenen CPU-Kapazitäten, sowie ca. 2,2 Gigabyte bis ca. 4,8 Gigabyte Arbeitsspeicher. Trotzdem konnte durch die Parallelisierung die benötigte Zeit für eine Iteration mit einer Stapelgröße von 64 über die 14041 Einträge des Trainingsdatensatzes von ca. 78 Minuten auf ca. 47 Minuten reduziert werden. Als Verlustfunktion des BE-BLC-Modells wird Affi et al. folgend eine „negative Log-Likelihood“-Verlustfunktion verwendet. Dabei wird die Wahrscheinlichkeit gemessen, mit der einer Eingabesequenz im Vergleich zu allen anderen möglichen Entitätszuordnungen die richtigen Entitäten zugeordnet werden. Dazu wird die Punktzahl der korrekten Entitätszuordnung von der logarithmierten Summe der Punktzahlen aller möglichen Entitätszuordnungen subtrahiert [Aff21]. Die erreichten Ergebnisse können der Abbildung 4.4 entnommen werden.

Parameter	**Wert**
Dropout	0,5
Lernrate	0,0001
Größe Dokumentenstapel (Batch)	64
Iterationen über Datensatz (Epochen)	170
Geduld	25

Tabelle 4.2: Übersicht der Modellparameter des NER-Modells.

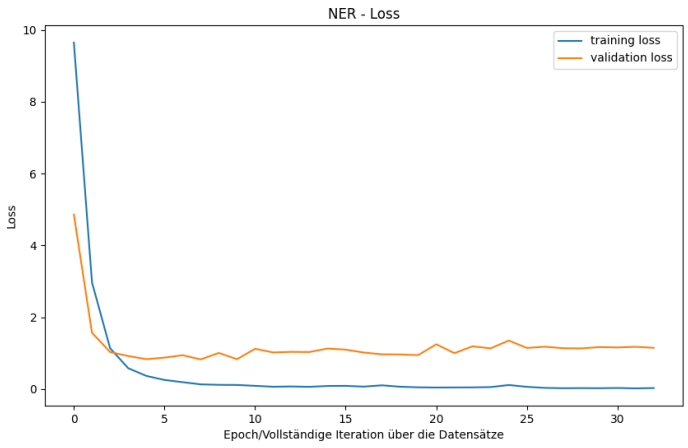

Abbildung 4.4: Verlustfunktion für Training und Validierung des NER-Modells.

4.4 Evaluierung

Für die Evaluation der implementierten Modelle wird das im Kapitel 3.4 vorgestellte Evaluationskonzept angewendet. Um eine ideale Grundlage zu schaffen, sollte für beide Aufgaben ein einheitlicher Datensatz verwendet werden. Dies ist nicht gegeben, was suboptimale Grundvoraussetzungen darstellt. Dennoch werden im Rahmen der gegebenen Möglichkeiten die Metriken für das NER-Modell an den gegebenen Testdaten des CoNNL03-Datensatzes und die Metriken für das MLTC-Modell an den in Abschnitt 4.1 getrennten Testdaten des R21578-Datensatzes gemessen.

4.4.1 Messung der Metriken

Zur Messung der Metriken wurden zunächst alle wahren Ergebnisse und alle vorhergesagten Ergebnisse für den kompletten Testdatensatz gespeichert. Diese wurden zusammengefasst und der Ergebnisse die Metriken errechnet. Für die Berechnung der Metriken des MLTC-Modells wurde die scikit-learn[1] Python-Bibliothek verwendet, welche Methoden für alle benötigten Metriken beinhaltet. Die Metriken zur Evaluierung des NER-Modells wurden mit Hilfe der seqeval[2] Python-Bibliothek gemessen. Die Bibliothek ermöglicht unter anderem die Messung von Metriken an Ausgaben, die im IOB-Format annotiert wurden. Alle gemessenen Metriken sind in der Tabelle 4.3 aufgelistet. Präzision, Recall und F_1-Werte der beiden Modelle einzeln berechnet für die jeweiligen Label und Entitäten können in den Abbildungen 4.5 für das MLTC-Modell und in Abbildung 4.6 für das NER-Modell gefunden werden.
Um die Zeit zu messen, die das MLTC- und das NER-System für die Verarbeitung eines Dokuments benötigen, wurde die in Abschnitt 4.2.3 erstellte Systemzusammenführung verwendet. Es wurden alle Texte zwischen 5000 und 6000 Zeichen des angepassten R21578 Datensatzes verwendet. Die Ressourcen sind hier begrenzt, da nur 21 Texte unter diese Bedingung identifiziert werden konnte. Diese haben eine durchschnittliche Länge von ca. 5263 Zeichen. Im Durchschnitt wurden für das Senden und Empfangen von Label und Entities ca. 8,5 Sekunden benötigt. Die längste Zeit von 10,9 Sekunden wurde bei einer Textlänge von 5165 Zeichen gemessen, die kürzeste Zeit für ein Dokument bei einer Zeichenanzahl von 5407 Zeichen mit ca. 7,4 Sekunden.

1 https://scikit-learn.org/stable/modules/model_evaluation.html
2 https://github.com/chakki-works/seqeval

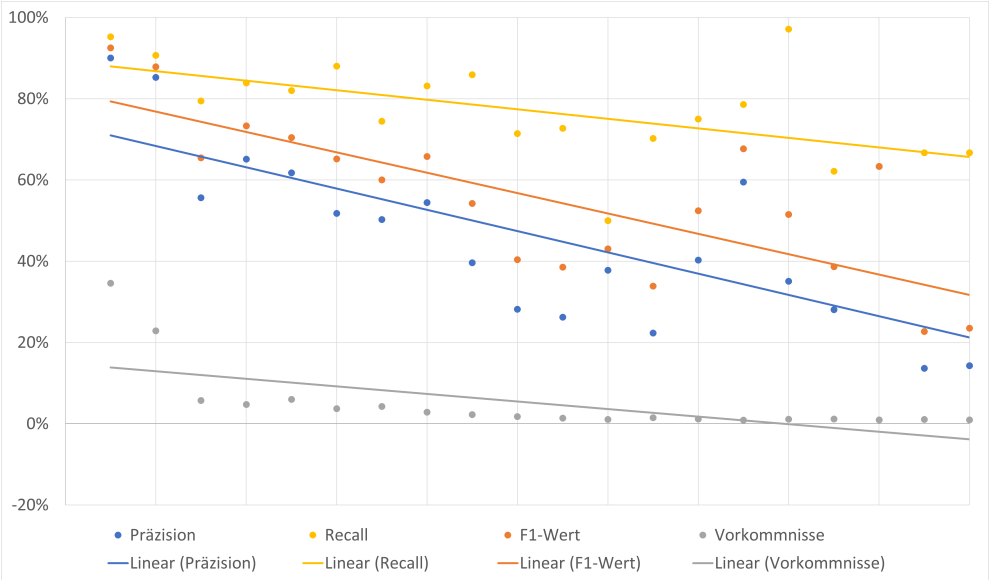

Abbildung 4.5: Präzision, Recall, F_1-Werte und Häufigkeit der einzelnen Label auf dem angepassten R21578-Testdatensatz sowie lineare Trendlinien. Testdatensatz und die entsprechenden linearen Trendlinien. Die Reihenfolge entspricht dem Auftreten der Label im Trainingsdatensatz (von oben nach unten).

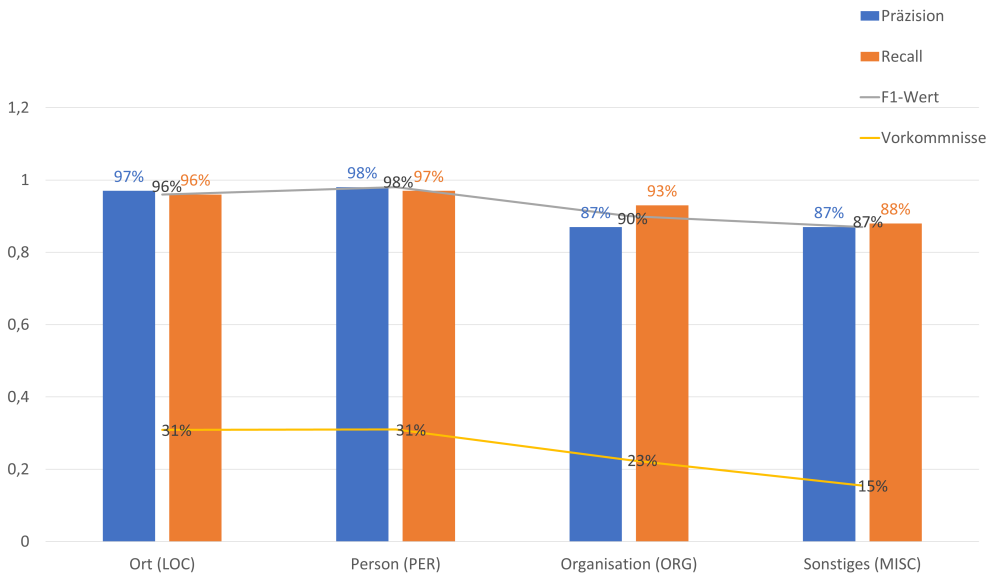

Abbildung 4.6: Präzision, Recall, F_1-Werte und Häufigkeit der einzelnen Entitäten für den CoNLL03 Testdatensatz.

Datensatz	angepasster R21578 Testdatensatz	CoNLL03 Testdatensatz
Metrik	MLTC	NER
Micro-Präzision	0,63	0,93
Micro-Recall	0,87	0,94
Micro-F_1-Wert	0,73	0,94
Makro-Präzision	0,46	0,92
Makro-Recall	0,77	0,94
Makro-F_1-Wert	0,56	0,93
Hamming Loss	0,04	/

Tabelle 4.3: Übersicht der gemessenen Metriken des HAWK- und BE-BLC-Modells an den jeweiligen Testdatensätzen,

4.4.2 Interpretation der Metriken

Das MLTC-Modell konnte einen Mikro-Recall von 87% erreichen, was im Umkehrschluss bedeutet, dass von allen tatsächlichen Zuordnungen 13% nicht erkannt werden konnten. Die niedrigere Mikro-Präzision von 62% zeigt, dass in 38% der Fälle die Zuordnung eines Label nicht der Wahrheit entsprach. Die Kombination der beiden Werte in Form des Mikro-F_1-Wertes zeigt, dass das Modell bei einem ausgewogenen Wert zwischen der Vermeidung falscher Vorhersagen und dem Versuch, möglichst viele tatsächlich zutreffende Labelzuordnungen vorherzusagen, eine ausgewogene Leistung in Bezug auf Genauigkeit und Vollständigkeit erreicht. Die Makrowerte stellen den Mittelwert aller entsprechenden Metriken dar, die für jedes Label einzeln berechnet wurden. Alle drei Werte sind im Vergleich zu den Mikrowerten niedriger, was auf eine unausgewogene Leistung bei der Labelzuordnung hindeutet. So werden nach dem Makro-Recall im Durchschnitt 77% richtig vorhergesagt und über alle Label betrachtet sind nach der Makro-Präzision 46% der Labelzuordnungen richtig. Dies zeigt, dass vor allem Label, die seltener vorkommen als andere, auch eine geringere Wahrscheinlichkeit haben, richtig vorhergesagt zu werden und eine höhere Wahrscheinlichkeit, falsch vorhergesagt zu werden. Der Hamming Loss von 4% zeigt, dass das Modell trotz der inkonsistenten Labelzuordnung die gesamte Ausgabe zu 96% korrekt ist. Das bedeutet, dass über alle Vorhersagen und Nicht-Vorhersagen eine Fehlerquote von 4% vorliegt.

Das NER-Modell erreichte auf dem Testdatensatz einen Mikro-Recall von 94% und eine Mikro-Präzision von 93%. Mit 7% falsch erkannten Entitäten und 6% nicht erkannten Entitäten spricht dies für eine insgesamt genaue Erkennung der Entitäten. Entsprechend der nahe beieinander liegenden Werte ergibt sich auch ein entsprechender Mikro-F_1-Wert von 94%. Der Makro-Recall entspricht mit 94% dem Mikro-Recall und zeigt damit eine konsistente Erkennung der tatsächlich richtigen Entitäten über alle Klassen hinweg. Die Makro-Präzision liegt mit 92% um 2% unter dem Mikro-Wert, was auf eine leichte Abschwächung der Vorhersagegenauigkeit bei seltener auftretenden Entitäten hindeutet. Dementsprechend ist auch das harmonische Mittel der beiden Werte in Form des Makro-

F_1-Wertes um 1% niedriger als der Mikro-F_1-Wert. Die starke Korrelation zwischen den Metriken der einzelnen Entitäten zu ihren prozentualen Vorkommnissen werden in Abbildung 4.6 verdeutlicht. Insgesamt liefert das Modell über alle Klassen hinweg überwiegend korrekte Ergebnisse.

Die Messung der Zeit für die Vorhersage von Label und Entitäten für ein Dokument hat gezeigt, dass die Textlänge nicht allein ausschlaggebend für die Verarbeitungszeit eines Dokuments ist. Andere Faktoren könnten die Komplexität des Textes oder die Anzahl der Entitäten oder Label sein. Dies kann jedoch nicht durch die Messung der Zeit allein bestimmt werden.

4.5 Verifikation der Anforderungen

In diesem Abschnitt wird die Erfüllung und Nicht-Erfüllung der in Abschnitt 3.1 aufgestellten Anforderungen eingegangen. Dafür werden zunächst die funktionalen und danach die nicht-funktionale Anforderungen behandelt. Einen Überblick über den Erfüllungsstatus kann in Tabelle 4.4 eingesehen werden.

Die Anforderungen *FA-1* und *FA-2* wurden erfüllt. Dies wird durch die in Abschnitt 4.4 durchgeführte Evaluierung der Modelle bestätigt. In dieser wird gezeigt, dass im zuvor durchlaufenen Training enthaltene Entitäts- und Labelklassen in bisher unbekannten Textaschnitten erkannt werden konnten. Die Umwandlung der numerischen Ausgabe in ein Textformat in natürlicher Sprache wird in Abschnitt 4.2.3 beschrieben.

Funktionale Anforderungen	
Anforderung	**Erfüllt**
FA-1	✓
FA-2	✓
FA-3	✓
FA-4	✓
FA-5	✓

Nicht-funktionale Anforderungen	
Anforderung	**Erfüllt**
NFA-1	✓
NFA-2	✓
NFA-3	✓
NFA-4	✗
NFA-5	
NFA-5.1	✓
NFA-5.2	✓
NFA-6	
NFA-6.1	✗
NFA-6.1	✗

Tabelle 4.4: Übersicht über die gestellten Anforderungen, unterteilt in funktionale und nicht-funktionale Anforderungen und deren jeweilige Erfüllung. Anforderungen mit nur einem Akzeptanzkriterium wurden ohne Unterpunkte aufgelistet.

In Abschnitt 4.2.3 wird die Aufteilung der Texte für die Eingabe in die jeweiligen Modelle beschrieben und in Abschnitt 4.2.1 und 4.2.2 die jeweilige vorgenommene Tokenisierung der Eingaben womit *FA-3* erfüllt wird.

FA-4 wird durch den in Abschnitt 4.3 nachweislich durchlaufenen Trainingsprozess bestätigt. Die Anpassung an den Datensatz wird durch eine Verringerung der Ausgabe der Verlustfunktionen bestätigt. In beiden Trainingsskripten werden JSON-Dateien zur Konfiguration der Parameter genutzt.

FA-5 Die in Abschnitt 4.4 durchgeführte Evaluierung kann in einem dem jeweiligen Modell zugehörigen Evaluationsskript gefunden werden. In diesem Fall wird über den jeweiligen Testdatensatz iteriert und die errechneten Metriken über die Python eigene „print“[1]-Methode ausgegeben. Die gemessenen Metriken basieren dabei auf dem in Kapitel 3.4 aufgestellten Evaluierungskonzept.

Die Zeit, die für die Verarbeitung eines Dokuments bzw. eines Textes mit 5000 bis 6000 Zeichen benötigt wird, wurde während der Evaluation im Abschnitt 4.4 gemessen. Demnach wurde für die durchschnittliche Verarbeitung von Texten dieser Länge 8,5 Sekunden benötigt, wonach *NFA-1* als erfüllt angesehen werden kann.

Die für das Training notwendige Angabe einer Konfigurationsdatei ermöglicht das Training des Modells auf anderen Datensätzen ohne die Struktur des Modells verändern zu müssen. Dazu muss ein neuer Datensatz in eine Subklasse eingebunden werden, die von der abstrakten Klasse Dataset[2] erbt. Somit wird die *NFA-2* erfüllt.

Sowohl das Training als auch die Validierung des Modells wurden auf einem Rechner durchgeführt, der die in *NFA-3* geforderten Anforderungen erfüllt.

Obwohl die gefundenen Datensätze in deutscher und englischer Sprache vorliegen, handelt es sich bei den verwendeten Datensätzen R21578 und CoNLL03 um zwei getrennte Datensätze, so dass *NFA-4*, wie in Abschnitt 3.3 beschrieben, nicht erfüllt werden kann.

Zur Erfüllung von *NFA-5* wurden im Rahmen des in Abschnitt 3.3 aufgestellten Konzepts Architekturen gewählt, die für die jeweils angegebene Anzahl von Label und Entitäten bereits verwendet werden. Darüber hinaus wurde der R21578-Datensatz angepasst, um die Anforderungen von *NFA-5.1* im Abschnitt 4.1 zu erfüllen. Der Datensatz CoNLL03 erfüllt die Anforderung aus *NFA-5.2* bereits. Da die Modelle auf diesen Datensätzen trainiert wurden, sind die Akzeptanzkriterien und damit die Anforderung erfüllt.

1 https://docs.python.org/3/library/functions.html?highlight=print#print
2 https://pytorch.org/docs/stable/data.html#torch.utils.data.Dataset

Im Hinblick auf *NFA-6* konnten die Akzeptanzkriterien nicht erfüllt werden. Zur Erreichung des ersten Akzeptanzkriteriums ist in beiden Modellen ein Mikro-Recall von 90% erforderlich, der nach 4.4 vom NER-Modell sogar übertroffen, vom MLTC-Modell jedoch nicht erreicht werden konnte. Dies gilt auch für *NFA-6.2*, wobei das BE-BLC-Modell 3% über der zur Erfüllung notwendigen 90%-Mikropräzision liegt, das HAWK-Modell jedoch 27% unter dem geforderten Wert.

5 Zusammenfassung

Diese Bachelorarbeit beschäftigt sich mit der Multi-Label Textklassifikation (MLTC) und Named Entity Recognition (NER) mit Hilfe von künstlicher Intelligenz (KI). Dazu wurden aktuelle Ansätze untersucht und miteinander verglichen, um eine geeignete Lösung für MLTC und NER zu identifizieren. Der identifizierte Lösungsansatz wurde mittels eines Prototyps implementiert. Zusätzlich wurde ein Evaluationskonzept entwickelt, um die Leistungsfähigkeit der realisierten Lösung zu messen und eine Vergleichsbasis zu anderen möglichen Lösungen zu schaffen.

Zu Beginn der Arbeit wurden die für das Verständnis der Arbeit notwendigen Begriffe erläutert. In diesem Abschnitt wurde eine Einführung in die für die Arbeit relevanten Bestandteile von Neuronalen Netzen sowie eine Erläuterung von MLTC und NER, Textverarbeitung mit KI und die Definitionen aller in der Arbeit verwendeten Metriken gegeben.

Im nächsten Schritt wurden sowohl funktionale als auch nicht-funktionale Anforderungen an das MLTC- und NER-System gestellt. Um den aktuellen Stand der Technik zu untersuchen, wurde eine strukturierte Literaturrecherche (SLR) durchgeführt, in der KI-Modelle, die in ihren jeweiligen Disziplinen Höchstleistungen erbracht haben, sowie mögliche Lösungsansätze, die MLTC und NER vereinen, identifiziert. Des Weiteren wurden die Metriken aus den als relevant identifizierten Arbeiten extrahiert, um auf dieser Basis im weiteren Verlauf ein Evaluationskonzept aufbauen zu können. Diese vorgestellten Modelle wurde als Basis der Diskussion zur Erstellung eines Realisierungskonzeptes verwendet.

Aufbauend auf der Konzeption wurde die Umsetzung des zuvor erstellten Realisierungskonzeptes behandelt. Dabei wurde sowohl die Implementierung des NER- als auch des MLTC-Modells sowie die anschließende Zusammenführung beschrieben. Im Anschluss an die Implementierung wurden die Datensätze sowie das Training der implementierten Modelle beschrieben, die für eine möglichst korrekte Ausführung der definierten Aufgaben notwendig sind. Abschließend wurde das zuvor aufgestellte Evaluationskonzept auf die trainierten Modelle angewandt und die Erfüllung bzw. Nichterfüllung der funktionalen und nicht-funktionalen Anforderungen beschrieben.

In diesem letzten Abschnitt wird ein abschließendes Fazit formuliert und die erzielten Ergebnisse bewertet. Ebenso werden verwandte Arbeiten, auf den Ergebnissen aufbauende notwendige weitere Schritte und ein Ausblick auf zukünftige Lösungen der Problemstellungen gegeben.

5.1 Fazit

In diesem Abschnitt werden die im Rahmen der gesamten Arbeit erreichten Ergebnisse zusammengefasst. Diese werden zur Beantwortung der in Abschnitt 1.2 aufgestellten Forschungsfragen verwendet.

In dieser Arbeit wurden mit Hilfe einer strukturierten Literaturrecherche und einer anschließende Diskussion mehrere aktuelle Modelle analysiert, um eine geeignete Lösung für die Aufgaben der Multi-Label Textklassifikation (MLTC) und Named Entity Recognition (NER) zu identifizieren. Insbesondere das Transformer-LDA Multi-Label Klassifikationsmodell von Tang et al.[Tan23] und das HAWK Modell von Javeed[Jav23] zeigten bemerkenswerte Leistungen in der MLTC. Für die Aufgabe der NER wurden das CLIM-Modell von Li et al.[Li23], das BINDER-Modell von Zhang et al.[Zha22b] und das BE-BLC-Modell von Afii et al.[Aff21] als besonders leistungsfähig identifiziert. Zusätzlich wurde die Möglichkeit untersucht, ein einziges Modell für beide Aufgaben zu verwenden, wobei das LANRTN-Modell von Yan et al.[Yan22] vorgestellt wurde, das eine NER-Komponente für MLTC verwendet. Die anschließende Diskussion und Konzeptentwicklung auf Basis dieser Modelle ergab unter Berücksichtigung der in Abschnitt [Anforderungen] definierten Anforderungen, dass das BE-BLC-Modell, das auf einer Kombination aus BERT- und ELMo-Encodern für Worteinbettungen sowie einem BiLSTM und einem darauf aufbauenden CRF basiert, und das HAWK-Modell, das USE in der DAN-Variante für Satzeinbettungen zusammen mit einem BiLSTM und BiLSTM-Klassifikationsköpfen verwendet, am besten geeignet sind. Diese Architekturen können als Antwort auf die *FF-1* betrachtet werden.

FF-2 kann durch die Realisierung der beiden Modelle beantworten werden. Demnach wurden vortrainierte Encoder verwendet und in den jeweiligen Modellen eingebettet. Auf dieser Grundlage wurden Methoden erstellt, welche Texte in eine für die Encoder passende Form bringen oder diese an eine weitere Schicht der neuronalen Netze weitergeben. Sowohl für das NER- als auch das MLTC-Modell wurden aufbauende BiLSTM Schichten verwendet. Diese sind an die Größe der Ausgabematrizen der Encoder angepasst. Das MTLC-Modell nutzt aufbauend auf das BiLSTM eine dichte Schicht bevor die Ergebnisse an die BiLSTM-Köpfe weitergegeben werden. Für jedes mögliche Label wird ein zugehöriger BiLSTM-Kopf bestehend aus zwei weiteren BiLSTMs, drei dichten Schichten und einer Sigmoid-Aktivierungsfunktion erstellt. Die Ausgabe jedes

Kopfes liegt zwischen 0 und 1 und steht für die Zuweisung eines Labels falls die jeweilige Ausgabe größer als 0,5 beträgt. Das NER-Modell nutzt auf dem BiLSTM aufbauend ein CRF welches für jedes Worttoken eine numerische Ausgabe produziert die einer Entitätsklasse zugewiesen werden kann. Die Gewichte der neuronalen Netze werden mit Hilfe von Iterationen über einen Trainingsdatensatz angepasst, wobei eine „negative Log-Likelihood“-Verlustfunktion für das NER-Modell und eine „gewichtete Binary Cross-Entropy“-Verlustfunktion für das MLTC-Modell genutzt werden.

Mittels der strukturierten Literaturrecherche wurde die Häufigkeit der Verwendung verschiedener Metriken für MLTC- und NER-Modelle ermittelt. Diese Metriken, die alle in einem wissenschaftlichen Kontext verwendet wurden, können als Antwort auf *FF-3* betrachtet werden. Konkreter wurde in Abschnitt 3.4 ein Evaluationskonzept entwickelt, das die Evaluation von MLTC- und NER-Modellen ermöglicht. Dabei wurden für beide Disziplinen die Metriken Mikro- und Makro-Präzision, -Recall und F_1-Werte definiert und der Hamming-Verlust als weitere Perspektive auf MLTC-Ergebnisse festgelegt. Um den gestellten Anforderungen gerecht zu werden, wird in beiden Disziplinen auch die durchschnittliche Bearbeitungszeit eines Dokumentes gemessen.

5.2 Auswertung

In diesem Abschnitt liegt der Schwerpunkt auf der detaillierten Analyse und Bewertung der ausgewählten Modelle und ihrer Ergebnisse. Dabei werden sowohl die Herausforderungen bei der Umsetzung als auch die erzielten Ergebnisse kritisch reflektiert.

Zunächst wird auf die Vorgehensweise bei der strukturierten Literaturrecherche (SLR) eingegangen. Um den aktuellen Stand der Forschung zu erfassen, wurde eine Vielzahl wissenschaftlicher Arbeiten gesichtet. Aufgrund des vorgegebenen zeitlichen Rahmens der Arbeit mussten Einschränkungen bei der Auswahl der Literatur vorgenommen werden. Eine vollständige Sichtung aller relevanten Literatur hätte zu einem anderen Ergebnis führen können, welches den weiteren Verlauf der Arbeit maßgeblich beeinflusst hätte. Des Weiteren wurden trotz vieler relevanter Arbeiten auch viele Arbeiten durch die Suchabfrage auf den verwendeten Plattformen gefunden, die keine vollständige Relevanz für diese Arbeit aufweisen konnten. Eine präzisere Suchabfrage oder die Verwendung anderer Schlagwörter hätte möglicherweise zum Ausschluss vieler irrelevanter Arbeiten und somit zu einem effizienteren Vorgehen führen können. Dennoch konnte ein weit gefächerter Einblick der vorhandenen und aktuellen Literatur im Bereich der NER und MLTC erreicht werden. Im Hinblick auf die Ermittlung des derzeitigen Technikstandes wurden die Modelle vorgestellt, die die jeweils höchste Leistungsfähigkeit erreicht haben. Dies ist zwar für die in der Arbeit vorgenommene Modellauswahl zielführend, jedoch hätte eine vollständige Auflistung der Ergebnisse, ähnlich der Darstellung der verwendeten

Metriken innerhalb der Arbeiten im späteren Teil der SLR, zu einer besseren Übersicht der Ergebnisse und einem zusätzlichen Informationsgewinn der Arbeit geführt. Die Auflistung der Metriken lieferte eine solide Grundlage für das Evaluationskonzept, aber eine genauere Analyse der jeweiligen Anwendungsszenarien und des Kontextes, in dem die Metriken interpretiert wurden, hätte tiefere Einblicke in die Verwendung der Metriken geben können.

Für die Implementierung konnte kein einheitlicher Datensatz identifiziert werden, was zur Folge hat, dass viele Schritte für beide Modelle getrennt betrachtet werden müssen. Somit kann eine Lösung, die MLTC und NER kombiniert, nicht direkt mit der in dieser Arbeit vorgestellten Lösung verglichen werden, da zwei getrennte Datensätze verwendet werden müssen. Da in den vorgestellten Arbeiten unterschiedliche Datensätze und Metriken verwendet wurden, war ein direkter Vergleich der Modelle nicht möglich. Ein direkter und damit aussagekräftiger Vergleich hätte jedoch die Implementierung der vorgestellten Modelle und umfangreiche Experimente erfordert, was nicht im Rahmen dieser Arbeit lag. Das Training der Modelle stellte lediglich im zeitlichen Umfang eine Herausforderung dar.

Die Ergebnisse der Evaluierung geben einen guten Überblick über die Leistungsfähigkeit des Modells für die jeweiligen Datensätze. Die Ergebnisse des NER-Modells konnten auf dem CoNLL03-Datensatz alle an das System gestellten Anforderungen erfüllen und gute Ergebnisse erzielen. Das MLTC-Modell konnte die Anforderungen auf dem individuell aufgeteilten Reuters 21578-Datensatz nicht erfüllen. So wurde die geforderte Quote für die Erkennung der wahren Labels (Mikro-Recall) um 3% und die Quote für die falsche Erkennung (Mikro-Präzision) um 27% unterschritten. Die Erkennungsraten der Modelle lassen sich durch die Unterscheidung der Aufgabenstellung nicht in Relation zueinander setzen, sondern die Differenz der Datensätze entfernt die Aussagekraft der Metriken weiter voneinander. Außerdem ist davon auszugehen, dass ein anderer Datensatz, welcher andere Texte, ein anderes Vorkommen von Labeln sowie Entitäten oder andere Bedingungen für die wahre Labelzuweisung aufweist, auch zu anderen Ergebnissen führen würde. Dies gilt für beide Modelle. Für eine zuverlässige Erfüllung der Anforderungen sowie für eine solide Vergleichsbasis der Ergebnisse der Evaluierung wäre daher die Festlegung eines Datensatzes erforderlich, da die pauschalen Anforderungen ohne umfangreiche Tests an vielen verschiedenen Datensätzen nicht voll umfassend erfüllt werden können.

5.3 Weitere Ansätze

In diesem Abschnitt werden alternative Ansätze zur Multi-Label Textklassifikation und Named Entity Recognition aufgezeigt, die potenzielle Alternativen oder Erweiterungen der in dieser Arbeit vorgestellten Methoden darstellen. Die Vorstellung existierender, relevanter Arbeiten wurde bereits innerhalb der SLR Resultate durchgeführt.

Innerhalb der SLR wurde festgestellt, dass die Kombination von MLTC und NER innerhalb eines einzigen Models nur spärlich erforscht ist. Die sinnvolle Kombination beider Komponenten konnte im Ansatz von Yan et al.[Yan22] bereits dargestellt werden. Eine Weiterführung und Betrachtung der Kombinationen aus anderen Blickwinkeln könnte für beide Disziplinen einzeln, aber vor allem in Kombination von Nutzen sein.

Die implementierten Modelle dieser Arbeit passen die eingegebenen Texte lediglich an die für die Encoder benötigte Form an. Eine Vorverarbeitung der Texte durch beispielsweise die in Abschnitt 2.4 aufgezeigten Methoden, könnte der Modellleistung zuträglich sein. Zur Bestätigung dieser Hypothese werden allerdings weitergehende Experimente benötigt. Weitergehend könnten nach dem Vorschlag von Javeed die Encoder Komponente des HAWK-Modells ausgetauscht werden, was zu einer Verbesserung der Modellleistung führen könnte [Jav23].

5.4 Nächste Schritte

Die nächsten Schritte, aufbauend auf den Ergebnissen der vorliegenden Arbeit, bestehen in der Anpassung der Modelle für weitere Anwendungsszenarien durch individuell angepasste Datensätze. Die neuen Gewichtungen der Verknüpfungen innerhalb der Modelle können mit dem vorhandenen Trainingsprogramm und der zugehörigen Konfigurationsdatei erstellt werden. Da derzeit keine verallgemeinerbaren Aussagen über die Leistungsfähigkeit der Modelle für verschiedene Szenarien getroffen werden können, sollte nach dem Training der Modelle das in der Arbeit beschriebene Evaluationskonzept angewendet werden, um die geforderten Erkennungs- und Fehlerraten durch die gemessenen Metriken verifizieren zu können. Anschließend kann das kombinierte System mit den individuellen Gewichten in verschiedene Anwendungen integriert werden.

5.5 Ausblick

Wie bereits im Abschnitt 3.2.7 im Rahmen der SLR aufgezeigt worden ist, lässt sich ein aktueller Trend zur Nutzung von „Generative Pretrained Transformer“-Modellen zur Bewältigung der NER feststellen. So wurden in den Arbeiten von Wang et al. und Yue et al. das GPT3-Modell verwendet, wobei bereits dem aktuellen Stand der Technik annähernde Resultate erzielt werden konnten. Die Erforschung der Aufgabenstellung durch GPT-Modelle der nächsten Generation, wie GPT-4[Ope23] oder anderen generativen Modellen könnte neue Möglichkeiten innerhalb der NER erreichen. Die Anwendung generativer Modelle könnte möglicherweise auch neue Erkenntnisse im Bereich der MLTC oder der Forschung in der Kombination beider Disziplinen liefern.

Literaturverzeichnis

[Abd21] Abdurrazzaq, Muhammad Adrinta; Saptawati, Gusti Ayu Putri und Rusmawati, Yanti: MAGNET Architecture Optimization on Multi-Label Text Classification, in: *2021 8th International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA)*, IEEE

[Aff21] Affi, Manel und Latiri, Chiraz: BE-BLC: BERT-ELMO-Based Deep Neural Network Architecture for English Named Entity Recognition Task. *Procedia Computer Science* (2021), Bd. 192: S. 168–181

[Agg22] Aggarwal, Karan; Mijwil, Maad M; Al-Mistarehi, Abdel-Hameed; Alomari, Safwan; Gök, Murat; Alaabdin, Anas M Zein; Abdulrhman, Safaa H et al.: Has the Future Started? The Current Growth of Artificial Intelligence, Machine Learning, and Deep Learning. *Iraqi Journal for Computer Science and Mathematics* (2022): S. 115–123

[Bit22] Bitkom: Which of the following statements about investment of your company into AI applies? (2022), URL `https://www.statista.com/statistics/1388586/ai-investment-companies-germany/`, in: Statista, aufgerufen am 01.09.2023

[Ble03] Blei, David M; Ng, Andrew Y und Jordan, Michael I: Latent dirichlet allocation. *Journal of machine Learning research* (2003), Bd. 3(Jan): S. 993–1022

[Boj16] Bojanowski, Piotr; Grave, Edouard; Joulin, Armand und Mikolov, Tomas: Enriching Word Vectors with Subword Information (2016)

[Bro20] Brown, Tom; Mann, Benjamin; Ryder, Nick; Subbiah, Melanie; Kaplan, Jared D; Dhariwal, Prafulla; Neelakantan, Arvind; Shyam, Pranav; Sastry, Girish; Askell, Amanda et al.: Language models are few-shot learners. *Advances in neural information processing systems* (2020), Bd. 33: S. 1877–1901

[Cer18] Cer, Daniel; Yang, Yinfei; Kong, Sheng-yi; Hua, Nan; Limtiaco, Nicole; John, Rhomni St.; Constant, Noah; Guajardo-Cespedes, Mario; Yuan, Steve; Tar, Chris; Sung, Yun-Hsuan; Strope, Brian und Kurzweil, Ray: Universal Sentence Encoder (2018)

[Cha20] Chang, Wei-Cheng; Yu, Hsiang-Fu; Zhong, Kai; Yang, Yiming und Dhillon, Inderjit S.: Taming Pretrained Transformers for Extreme Multi-label Text Classification, in: *Proceedings of the 26th ACM SIGKDD International*

Conference on Knowledge Discovery & Data Mining, ACM

[Cha21] Chalkidis, Ilias; Fergadiotis, Manos und Androutsopoulos, Ion: MultiEURLEX - A multi-lingual and multi-label legal document classification dataset for zero-shot cross-lingual transfer, in: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics

[Cha22a] Chaib, Rim; Azizi, Nabiha; Hammami, Nacer Eddine; Gasmi, Ibtissem; Schwab, Didier und Chaib, Amira: GL-LSTM Model For Multi-label Text Classification Of Cardiovascular Disease Reports, in: *2022 2nd International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, IEEE

[Cha22b] Chalkidis, Ilias und Søgaard, Anders: Improved Multi-label Classification under Temporal Concept Drift: Rethinking Group-Robust Algorithms in a Label-Wise Setting (2022)

[Cha23] Chalkidis, Ilias und Kementchedjhieva, Yova: Retrieval-augmented Multi-label Text Classification (2023)

[Che17] Chen, Guibin; Ye, Deheng; Xing, Zhenchang; Chen, Jieshan und Cambria, Erik: Ensemble application of convolutional and recurrent neural networks for multi-label text categorization, in: *2017 International Joint Conference on Neural Networks (IJCNN)*, IEEE

[Che19] Cheng, Feifan und Zhao, Jinsong: A novel process monitoring approach based on Feature Points Distance Dynamic Autoencoder, in: *Computer Aided Chemical Engineering*, Elsevier (2019), S. 757–762

[Che20] Chen, Ziheng und Ren, Jiangtao: Multi-label text classification with latent word-wise label information. *Applied Intelligence* (2020), Bd. 51(2): S. 966–979

[Che21] Chen, Xiang; Li, Lei; Deng, Shumin; Tan, Chuanqi; Xu, Changliang; Huang, Fei; Si, Luo; Chen, Huajun und Zhang, Ningyu: LightNER: A Lightweight Tuning Paradigm for Low-resource NER via Pluggable Prompting (2021)

[Che22] Cheng, Mingzhe; Li, Hongjun; Yang, Zelin; Fan, Wenjie und Gan, Yujin: Named Entity Recognition for Medical Dialogue Based on BERT and Adversarial Training, in: *2022 5th International Conference on Pattern Recognition and Artificial Intelligence (PRAI)*, IEEE

[Chi16] Chiu, Jason P.C. und Nichols, Eric: Named Entity Recognition with Bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics* (2016), Bd. 4: S. 357–370

[Cho20] Chowdhary, KR1442 und Chowdhary, KR: Natural language processing. *Fundamentals of artificial intelligence* (2020): S. 603–649

[Con23] Consulting, Next Move Strategy: Global Artificial Intelligence market in billion US$ (2023), URL `https://www.statista.com/study/50485/in-depth-report-artificial-intelligence/`, in: Statista, aufge-

rufen am 03.09.2023

[Cui21] Cui, Leyang; Wu, Yu; Liu, Jian; Yang, Sen und Zhang, Yue: Template-Based Named Entity Recognition Using BART, in: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, Association for Computational Linguistics

[Dai19] Dai, Zihang; Yang, Zhilin; Yang, Yiming; Carbonell, Jaime G.; Le, Quoc V. und Salakhutdinov, Ruslan: Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. *CoRR* (2019), Bd. abs/1901.02860, URL http://arxiv.org/abs/1901.02860

[Dem96] Dematos, Giovani; Boyd, Milton S.; Kermanshahi, Bahman; Kohzadi, Nowrouz und Kaastra, Iebeling: Feedforward versus recurrent neural networks for forecasting monthly japanese yen exchange rates. *Financial Engineering and the Japanese Markets* (1996), Bd. 3(1): S. 59–75

[Dem10] Dembczyński, Krzysztof; Waegeman, Willem; Cheng, Weiwei und Hüllermeier, Eyke: Regret Analysis for Performance Metrics in Multi-Label Classification: The Case of Hamming and Subset Zero-One Loss, in: *Machine Learning and Knowledge Discovery in Databases*, Springer Berlin Heidelberg (2010), S. 280–295

[Der16] Derczynski, Leon: Complementarity, F-score, and NLP Evaluation, in: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LRECÍ16)*, European Language Resources Association (ELRA), Portorož, Slovenia, S. 261–266, URL https://aclanthology.org/L16-1040

[Dev18] Devlin, Jacob; Chang, Ming-Wei; Lee, Kenton und Toutanova, Kristina: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (2018)

[Din20] Ding, Ming; Zhou, Chang; Yang, Hongxia und Tang, Jie: CogLTX: Applying BERT to Long Texts, in: H. Larochelle; M. Ranzato; R. Hadsell; M.F. Balcan und H. Lin (Herausgeber) *Advances in Neural Information Processing Systems*, Bd. 33, Curran Associates, Inc., S. 12792–12804, URL https://proceedings.neurips.cc/paper_files/paper/2020/file/96671501524948bc3937b4b30d0e57b9-Paper.pdf

[doc23] Docker, Inc. (2023), URL https://www.docker.com/company/newsroom/media-resources/, copyright 2013-2023 Docker, Inc. All rights reserved., aufgerufen am: 10.11.2023

[Dod04] Doddington, George; Mitchell, Alexis; Przybocki, Mark; Ramshaw, Lance; Strassel, Stephanie und Weischedel, Ralph: The Automatic Content Extraction (ACE) Program – Tasks, Data, and Evaluation, in: *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC{'}04)*, European Language Resources Association (ELRA), Lisbon, Portugal, URL http://www.lrec-conf.org/proceedings/

lrec2004/pdf/5.pdf

[Don22] Dong, Yonghao; Yang, Zhenmin und Cao, Hui: A Text Classification Model Based on GCN and BiGRU Fusion, in: *Proceedings of the 8th International Conference on Computing and Artificial Intelligence*, ACM

[Ere12] Erenel, Zafer und Altınçay, Hakan: Improving the precision-recall trade-off in undersampling-based binary text categorization using unanimity rule. *Neural Computing and Applications* (2012), Bd. 22(S1): S. 83–100

[Fan23] Fan, Caoyun; Chen, Wenqing; Tian, Jidong; Li, Yitian; He, Hao und Jin, Yaohui: Accurate use of label dependency in multi-label text classification through the lens of causality. *Applied Intelligence* (2023), Bd. 53(19): S. 21841–21857

[Fer21] Feroz, Abdul Karim; Zo, Hangjung und Chiravuri, Ananth: Digital Transformation and Environmental Sustainability: A Review and Research Agenda. *Sustainability* (2021), Bd. 13(3): S. 1530

[Fie00] Fielding, Roy Thomas: *Architectural styles and the design of network-based software architectures*, University of California, Irvine (2000)

[For73] Forney, G.D.: The viterbi algorithm. *Proceedings of the IEEE* (1973), Bd. 61(3): S. 268–278

[Fre22] Frei, Johann; Frei-Stuber, Ludwig und Kramer, Frank: GERNERMED++: Transfer Learning in German Medical NLP (2022)

[Fri19] Fritzler, Alexander; Logacheva, Varvara und Kretov, Maksim: Few-shot classification in named entity recognition task, in: *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, ACM

[Fu21] Fu, Jia; Liu, Jie und Shi, Wenxin: Exploiting Named Entity Recognition via Pre-trained Language Model and Adversarial Training, in: *2021 IEEE International Conference on Computer Science, Electronic Information Engineering and Intelligent Control Technology (CEI)*, IEEE

[Gan23] Gan, Chengguang; Zhang, Qinghao und Mori, Tatsunori: Sentence-to-Label Generation Framework for Multi-task Learning of Japanese Sentence Classification and Named Entity Recognition, in: *Natural Language Processing and Information Systems*, Springer Nature Switzerland (2023), S. 257–270

[Gar18] Gardner, Matt; Grus, Joel; Neumann, Mark; Tafjord, Oyvind; Dasigi, Pradeep; Liu, Nelson; Peters, Matthew; Schmitz, Michael und Zettlemoyer, Luke: AllenNLP: A Deep Semantic Natural Language Processing Platform (2018)

[Ger00] Gers, Felix A.; Schmidhuber, Jürgen und Cummins, Fred: Learning to Forget: Continual Prediction with LSTM. *Neural Computation* (2000), Bd. 12(10): S. 2451–2471

[Gli07] Glinz, M.: On Non-Functional Requirements, in: *15th IEEE International Requirements Engineering Conference (RE 2007)*, IEEE

[Gmb23] GmbH, mbi: Homepage mbi GmbH (2023), URL https://www.mbi.de/, aufgerufen am 08.10.2023

[Goo13] Goodfellow, Ian J.; Mirza, Mehdi; Xiao, Da; Courville, Aaron und Bengio, Yoshua: An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks (2013)

[Gou05] Goutte, Cyril und Gaussier, Eric: A probabilistic interpretation of precision, recall and F-score, with implication for evaluation, in: *European conference on information retrieval*, Springer, S. 345–359

[Gre06] Greene, Derek und Cunningham, Pádraig: Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering, in: *Proc. 23rd International Conference on Machine learning (ICML'06)*, ACM Press, S. 377–384

[Gri96] Grishman, Ralph und Sundheim, Beth: Message Understanding Conference-6, in: *Proceedings of the 16th conference on Computational linguistics* -, Association for Computational Linguistics

[Han21] Han, Xu; Zhang, Zhengyan; Ding, Ning; Gu, Yuxian; Liu, Xiao; Huo, Yuqi; Qiu, Jiezhong; Yao, Yuan; Zhang, Ao; Zhang, Liang; Han, Wentao; Huang, Minlie; Jin, Qin; Lan, Yanyan; Liu, Yang; Liu, Zhiyuan; Lu, Zhiwu; Qiu, Xipeng; Song, Ruihua; Tang, Jie; Wen, Ji-Rong; Yuan, Jinhui; Zhao, Wayne Xin und Zhu, Jun: Pre-trained models: Past, present and future. *AI Open* (2021), Bd. 2: S. 225–250

[Hoc91] Hochreiter, Sepp: Untersuchungen zu dynamischen neuronalen Netzen (1991)

[Hoc97] Hochreiter, Sepp und Schmidhuber, Jürgen: Long Short-Term Memory. *Neural Computation* (1997), Bd. 9(8): S. 1735–1780

[Hoc98] Hochreiter, Sepp: The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* (1998), Bd. 06(02): S. 107–116

[Hua21] Huang, Yi; Giledereli, Buse; Köksal, Abdullatif; Özgür, Arzucan und Ozkirimli, Elif: Balancing Methods for Multi-label Text Classification with Long-Tailed Class Distribution (2021)

[Jav23] Javeed, Arshad: Hawk: An Industrial-strength Multi-label Document Classifier (2023)

[Jeh23] Jehangir, Basra; Radhakrishnan, Saravanan und Agarwal, Rahul: A survey on Named Entity Recognition — datasets, tools, and methodologies. *Natural Language Processing Journal* (2023), Bd. 3: S. 100017

[Ji23] Ji, Weidong; Zhang, Yousheng; Zhou, Guohui und Wang, Xu: Research on Named Entity Recognition in Improved transformer with R-Drop structure (2023)

[Jie20] Jiechieu, Kameni Florentin Flambeau und Tsopze, Norbert: Skills prediction based on multi-label resume classification using CNN with model predictions

explanation. *Neural Computing and Applications* (2020), Bd. 33(10): S. 5069–5087

[Jos21] Josephine, V L Helen; Nirmala, A.P. und Alluri, Vijaya Lakshmi: Impact of Hidden Dense Layers in Convolutional Neural Network to enhance Performance of Classification Model. *IOP Conference Series: Materials Science and Engineering* (2021), Bd. 1131(1): S. 012007

[Ju18] Ju, Meizhi; Miwa, Makoto und Ananiadou, Sophia: A Neural Layered Model for Nested Named Entity Recognition, in: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Association for Computational Linguistics

[Kar20] Karpukhin, Vladimir; Oguz, Barlas; Min, Sewon; Lewis, Patrick; Wu, Ledell; Edunov, Sergey; Chen, Danqi und tau Yih, Wen: Dense Passage Retrieval for Open-Domain Question Answering, in: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics

[Kem23] Kementchedjhieva, Yova und Chalkidis, Ilias: An Exploration of Encoder-Decoder Approaches to Multi-Label Classification for Legal and Biomedical Text (2023)

[Kim22] Kim, Soojeong; Lee, Minhyeok und Seok, Junhee: Multi-label Text Classification of Economic Concepts from Economic News Articles using Natural Language Processing, in: *2022 Thirteenth International Conference on Ubiquitous and Future Networks (ICUFN)*, IEEE

[Kin14] Kingma, Diederik P. und Ba, Jimmy: Adam: A Method for Stochastic Optimization (2014)

[Kit07] Kitchenham, Barbara und Charters, Stuart: Guidelines for performing Systematic Literature Reviews in Software Engineering (2007), Bd. 2

[Kow17] Kowsari, Kamran; Brown, Donald E; Heidarysafa, Mojtaba; Jafari Meimandi, Kiana; ; Gerber, Matthew S und Barnes, Laura E: HDLTex: Hierarchical Deep Learning for Text Classification, in: *Machine Learning and Applications (ICMLA), 2017 16th IEEE International Conference on*, IEEE

[Laf01] Lafferty, John; McCallum, Andrew und Pereira, Fernando CN: Conditional random fields: Probabilistic models for segmenting and labeling sequence data (2001)

[Lam16] Lample, Guillaume; Ballesteros, Miguel; Subramanian, Sandeep; Kawakami, Kazuya und Dyer, Chris: Neural Architectures for Named Entity Recognition (2016)

[Len20] Lenivtceva, Iuliia; Slasten, Evgenia; Kashina, Mariya und Kopanitsa, Georgy: Applicability of Machine Learning Methods to Multi-label Medical Text Classification, in: *Lecture Notes in Computer Science*, Springer

International Publishing (2020), S. 509–522

[Lew04] Lewis, David D.; Yang, Yiming; Rose, Tony G. und Li, Fan: RCV1: A New Benchmark Collection for Text Categorization Research. *J. Mach. Learn. Res.* (2004), Bd. 5: S. 361–397

[Li19] Li, Xiaoya; Yin, Fan; Sun, Zijun; Li, Xiayu; Yuan, Arianna; Chai, Duo; Zhou, Mingxin und Li, Jiwei: Entity-Relation Extraction as Multi-Turn Question Answering (2019)

[Li20a] Li, Xiao; Liang, Wenteng; Li, Yifeng; Zhao, Yuxuan; Zhang, Zhizheng und Yang, Hengguang: RoBERTa Word Embedding Based Power Grid Dispatching Entity Recognition, in: *2020 12th IEEE PES Asia-Pacific Power and Energy Engineering Conference (APPEEC)*, IEEE

[Li20b] Li, Yachong und Yang, Youlong: Label Embedding for Multi-label Classification Via Dependence Maximization. *Neural Processing Letters* (2020), Bd. 52(2): S. 1651–1674

[Li21] Li, Xinyi; Qian, Ying und Dou, Liang: BERT Named Entity Recognition with Self-attention Mechanism, in: *2021 IEEE Conference on Telecommunications, Optics and Computer Science (TOCS)*, IEEE

[Li22a] Li, Irene; Feng, Aosong; Wu, Hao; Li, Tianxiao; Suzumura, Toyotaro und Dong, Ruihai: LiGCN: Label-interpretable Graph Convolutional Networks for Multi-label Text Classification, in: *Proceedings of the 2nd Workshop on Deep Learning on Graphs for Natural Language Processing (DLG4NLP 2022)*, Association for Computational Linguistics

[Li22b] Li, Jing; Sun, Aixin; Han, Jianglei und Li, Chenliang: A Survey on Deep Learning for Named Entity Recognition. *IEEE Transactions on Knowledge and Data Engineering* (2022), Bd. 34(1): S. 50–70

[Li22c] Li, Junzhe; Wang, Chenglong; Fang, Xiaohan; Yu, Kai; Zhao, Jinye; Wu, Xi und Gong, Jibing: Multi-label text classification via hierarchical Transformer-CNN, in: *2022 14th International Conference on Machine Learning and Computing (ICMLC)*, ACM

[Li23] Li, Qi; Xie, Tingyu; Peng, Peng; Wang, Hongwei und Wang, Gaoang: A Class-Rebalancing Self-Training Framework for Distantly-Supervised Named Entity Recognition. *Findings of the Association for Computational Linguistics: ACL 2023* (2023), URL `http://dx.doi.org/10.18653/v1/2023.findings-acl.703`

[Lia20] Liang, Ruiyu; Kong, Fanliu; Xie, Yue; Tang, Guichen und Cheng, Jiaming: Real-Time Speech Enhancement Algorithm Based on Attention LSTM. *IEEE Access* (2020), Bd. 8: S. 48464–48476

[Lin23] Lin, Nankai; Qin, Guanqiu; Wang, Gang; Zhou, Dong und Yang, Aimin: An Effective Deployment of Contrastive Learning in Multi-label Text Classification, in: *Findings of the Association for Computational Linguistics: ACL 2023*, Association for Computational Linguistics

[Lip14] Lipton, Zachary C.; Elkan, Charles und Naryanaswamy, Balakrishnan: Optimal Thresholding of Classifiers to Maximize F1 Measure, in: *Machine Learning and Knowledge Discovery in Databases*, Springer Berlin Heidelberg (2014), S. 225–239

[Liu19] Liu, Yinhan; Ott, Myle; Goyal, Naman; Du, Jingfei; Joshi, Mandar; Chen, Danqi; Levy, Omer; Lewis, Mike; Zettlemoyer, Luke und Stoyanov, Veselin: RoBERTa: A Robustly Optimized BERT Pretraining Approach (2019)

[Liu21] Liu, Naiyin; Wang, Qianlong und Ren, Jiangtao: Label-Embedding Bi-directional Attentive Model for Multi-label Text Classification. *Neural Processing Letters* (2021), Bd. 53(1): S. 375 – 389, URL `http://dx.doi.org/10.1007/s11063-020-10411-8`

[Lu20] Lu, Jinghui und MacNamee, Brian: Investigating the Effectiveness of Representations Based on Pretrained Transformer-based Language Models in Active Learning for Labelling Text Datasets (2020)

[Ló20] López, Federico und Strube, Michael: A Fully Hyperbolic Neural Model for Hierarchical Multi-Class Classification (2020)

[Ma21a] Ma, Qianwen; Yuan, Chunyuan; Zhou, Wei und Hu, Songlin: Label-Specific Dual Graph Neural Network for Multi-Label Text Classification, in: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Association for Computational Linguistics

[Ma21b] Ma, Xuchao; Zheng, Shuang und Wang, Quanmin: Dual-channel BERT-DBLCA Based on Attention Mechanism for News Category Label Classification Model, in: *2021 the 6th International Conference on Information Systems Engineering*, ACM

[Man09] Manning, Christopher D; Raghavan, Prabhakar und Schutze, Hinrich: *Introduction to Information Retrieval*, Cambridge University Press, Cambridge, England (2009)

[Mar05] Marinai, S.; Gori, M. und Soda, G.: Artificial neural networks for document analysis and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2005), Bd. 27(1): S. 23–35

[McK23] McKinsey: Leading AI capabilities adoption rate in business 2022 (2023), URL `https://www.statista.com/study/50485/in-depth-report-artificial-intelligence/`, in: Statista, aufgerufen am 03.09.2023

[Mel22] Melsbach, Johannes; Stahlmann, Sven; Hirschmeier, Stefan und Schoder, Detlef: Triplet transformer network for multi-label document classification, in: *Proceedings of the 22nd ACM Symposium on Document Engineering*, ACM

[Men21] Meng, Yu; Zhang, Yunyi; Huang, Jiaxin; Wang, Xuan; Zhang, Yu; Ji, Heng und Han, Jiawei: Distantly-Supervised Named Entity Recognition with Noise-Robust Learning and Language Model Augmented Self-Training (2021)

[Mer14] Merkel, Dirk: Docker: lightweight linux containers for consistent development and deployment. *Linux journal* (2014), Bd. 2014(239): S. 2

[Mhl21] Mhlanga, David: Artificial Intelligence in the Industry 4.0, and Its Impact on Poverty, Innovation, Infrastructure Development, and the Sustainable Development Goals: Lessons from Emerging Economies? *Sustainability* (2021), Bd. 13(11): S. 5788

[Mik13] Mikolov, Tomas; Chen, Kai; Corrado, Greg und Dean, Jeffrey: Efficient Estimation of Word Representations in Vector Space (2013)

[Mo23] Mo, Ying; Tang, Hongyin; Liu, Jiahao; Wang, Qifan; Xu, Zenglin; Wang, Jingang; Wu, Wei und Li, Zhoujun: Multi-Task Transformer with Relation-Attention and Type-Attention for Named Entity Recognition, in: *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE

[Mou22] Mou, Xiyv und Zhang, Xindong: Named Entity Recognition Based on Transformer Encoder in the Medical Field, in: *2022 3rd International Conference on Electronic Communication and Artificial Intelligence (IWECAI)*, IEEE

[Mul18] Mullapudi, Ravi Teja; Mark, William R; Shazeer, Noam und Fatahalian, Kayvon: Hydranets: Specialized dynamic architectures for efficient inference, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, S. 8080–8089

[Myl22] Mylonas, Nikolaos; Mollas, Ioannis und Tsoumakas, Grigorios: An Attention Matrix for Every Decision: Faithfulness-based Arbitration Among Multiple Attention-Based Interpretations of Transformers in Text Classification (2022)

[Nag22] Nagpal, Abhinav; Dasgupta, Riddhiman und Ganesan, Balaji: Fine Grained Classification of Personal Data Entities with Language Models, in: *5th Joint International Conference on Data Science & amp; Management of Data (9th ACM IKDD CODS and 27th COMAD)*, ACM

[Nam14] Nam, Jinseok; Kim, Jungi; Mencía, Eneldo Loza; Gurevych, Iryna und Fürnkranz, Johannes: Large-Scale Multi-label Text Classification — Revisiting Neural Networks, in: *Machine Learning and Knowledge Discovery in Databases*, Springer Berlin Heidelberg (2014), S. 437–452

[Nam17] Nam, Jinseok; Mencía, Eneldo Loza; Kim, Hyunwoo J. und Fürnkranz, Johannes: Maximizing Subset Accuracy with Recurrent Neural Networks in Multi-Label Classification, in: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, S. 5419–5429

[Nes22] Nesterov, Alexander und Umerenkov, Dmitry: Distantly supervised end-to-end medical entity extraction from electronic health records with human-level quality (2022)

[Ng16] Ng, Andrew: What Artificial Intelligence Can and Can't Do Right Now. *Harvard Business Review* (2016), URL https://hbr.org/2016/11/what-artificial-intelligence-can-and-cant-do-right-now

[Nik20] Nikolentzos, Giannis; Tixier, Antoine und Vazirgiannis, Michalis: Message Passing Attention Networks for Document Understanding. *Proceedings of the AAAI Conference on Artificial Intelligence* (2020), Bd. 34(05): S. 8544–8551

[Ong17] Ongsulee, Pariwat: Artificial intelligence, machine learning and deep learning, in: *2017 15th International Conference on ICT and Knowledge Engineering (ICT&KE)*, IEEE

[Ope23] OpenAI: GPT-4 Technical Report (2023), URL https://doi.org/10.48550/arXiv.2303.08774

[Pal20] Pal, Ankit; Selvakumar, Muru und Sankarasubbu, Malaikannan: MAGNET: Multi-Label Text Classification using Attention-based Graph Neural Network, in: *Proceedings of the 12th International Conference on Agents and Artificial Intelligence*, SCITEPRESS - Science and Technology Publications

[Pan15] Pannu, Avneet: Artificial intelligence and its application in different areas. *Artificial Intelligence* (2015), Bd. 4(10): S. 79–84

[Pan18] Panchendrarajan, Rrubaa und Amaresan, Aravindh: Bidirectional LSTM-CRF for named entity recognition, in: *Proceedings of the 32nd Pacific Asia conference on language, information and computation*

[Pas19] Paszke, Adam; Gross, Sam; Massa, Francisco; Lerer, Adam; Bradbury, James; Chanan, Gregory; Killeen, Trevor; Lin, Zeming; Gimelshein, Natalia; Antiga, Luca; Desmaison, Alban; Köpf, Andreas; Yang, Edward; DeVito, Zach; Raison, Martin; Tejani, Alykhan; Chilamkurthy, Sasank; Steiner, Benoit; Fang, Lu; Bai, Junjie und Chintala, Soumith: PyTorch: An Imperative Style, High-Performance Deep Learning Library (2019)

[Pen14] Pennington, Jeffrey; Socher, Richard und Manning, Christopher: Glove: Global Vectors for Word Representation, in: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics

[Pet18] Peters, Matthew E.; Neumann, Mark; Iyyer, Mohit; Gardner, Matt; Clark, Christopher; Lee, Kenton und Zettlemoyer, Luke: Deep contextualized word representations (2018)

[Pre02] Prechelt, Lutz: Early stopping-but when?, in: *Neural Networks: Tricks of the trade*, Springer (2002), S. 55–69

[Rad18] Radford, Alec; Narasimhan, Karthik; Salimans, Tim; Sutskever, Ilya et al.: Improving language understanding by generative pre-training (2018)

[Rad19] Radford, Alec; Wu, Jeffrey; Child, Rewon; Luan, David; Amodei, Dario; Sutskever, Ilya et al.: Language models are unsupervised multitask learners. *OpenAI blog* (2019), Bd. 1(8): S. 9

[Raf20] Raffel, Colin; Shazeer, Noam; Roberts, Adam; Lee, Katherine; Narang, Sharan; Matena, Michael; Zhou, Yanqi; Li, Wei und Liu, Peter J.: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.* (2020), Bd. 21(1)

[Rah23] Rahhal, Ibrahim; Carley, Kathleen M.; Kassou, Ismail und Ghogho, Mounir: Two Stage Job Title Identification System for Online Job Advertisements. *IEEE Access* (2023), Bd. 11: S. 19073–19092

[Rib20] Ribeiro, Marco Tulio; Wu, Tongshuang; Guestrin, Carlos und Singh, Sameer: Beyond accuracy: Behavioral testing of NLP models with CheckList. *arXiv preprint arXiv:2005.04118* (2020)

[Rob22] Roberts, Adam; Chung, Hyung Won; Levskaya, Anselm; Mishra, Gaurav; Bradbury, James; Andor, Daniel; Narang, Sharan; Lester, Brian; Gaffney, Colin; Mohiuddin, Afroz; Hawthorne, Curtis; Lewkowycz, Aitor; Salcianu, Alex; van Zee, Marc; Austin, Jacob; Goodman, Sebastian; Soares, Livio Baldini; Hu, Haitang; Tsvyashchenko, Sasha; Chowdhery, Aakanksha; Bastings, Jasmijn; Bulian, Jannis; Garcia, Xavier; Ni, Jianmo; Chen, Andrew; Kenealy, Kathleen; Clark, Jonathan H.; Lee, Stephan; Garrette, Dan; Lee-Thorp, James; Raffel, Colin; Shazeer, Noam; Ritter, Marvin; Bosma, Maarten; Passos, Alexandre; Maitin-Shepard, Jeremy; Fiedel, Noah; Omernick, Mark; Saeta, Brennan; Sepassi, Ryan; Spiridonov, Alexander; Newlan, Joshua und Gesmundo, Andrea: Scaling Up Models and Data with `t5x` and `seqio` (2022)

[Rud21] Ruder, Sebastian; Constant, Noah; Botha, Jan; Siddhant, Aditya; Firat, Orhan; Fu, Jinlan; Liu, Pengfei; Hu, Junjie; Garrette, Dan; Neubig, Graham und Johnson, Melvin: XTREME-R: Towards More Challenging and Nuanced Multilingual Evaluation (2021)

[San19] Sanh, Victor; Debut, Lysandre; Chaumond, Julien und Wolf, Thomas: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter (2019)

[Sar20] Sarwar, Muhammad Usman; Zafar, Sarim; Mkaouer, Mohamed Wiem; Walia, Gursimran Singh und Malik, Muhammad Zubair: Multi-label Classification of Commit Messages using Transfer Learning, in: *2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, IEEE

[Sav23] Savci, Pinar und Das, Bihter: Comparison of pre-trained language models in terms of carbon emissions, time and accuracy in multi-label text classification using AutoML. *Heliyon* (2023), Bd. 9(5): S. e15670

[Sch97] Schuster, M. und Paliwal, K.K.: Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* (1997), Bd. 45(11): S. 2673–2681

[Seo16] Seo, Minjoon; Kembhavi, Aniruddha; Farhadi, Ali und Hajishirzi, Hannaneh: Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603* (2016)

[Sid20] Siddhant, Aditya; Hu, Junjie; Johnson, Melvin; Firat, Orhan und Ruder, Sebastian: XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalization

[Son22] Song, Rui; Liu, Zelong; Chen, Xingbing; An, Haining; Zhang, Zhiqi; Wang, Xiaoguang und Xu, Hao: Label prompt for multi-label text classification. *Applied Intelligence* (2022), Bd. 53(8): S. 8761–8775

[Ste22] Stefanovič, Pavel und Kurasova, Olga: Approach for Multi-Label Text Data Class Verification and Adjustment Based on Self-Organizing Map and Latent Semantic Analysis. *Informatica* (2022): S. 109–130

[Sto22] Stollenwerk, Felix: Adaptive Fine-Tuning of Transformer-Based Language Models for Named Entity Recognition (2022)

[Su23] Su, Jindian und Yu, Hong: Unified Named Entity Recognition as Multi-Label Sequence Generation, in: *2023 International Joint Conference on Neural Networks (IJCNN)*, IEEE

[Sun16] Sun, Maosong; Li, Jingyang; Guo, Zhipeng; Zhao, Yu; Zheng, Yabin; Si, Xiance und Liu, Zhiyuan: THUCTC, An Efficient Chinese Text Classifier (2016), URL `http://thuctc.thunlp.org/`

[Sus13] Sussillo, David und Barak, Omri: Opening the Black Box: Low-Dimensional Dynamics in High-Dimensional Recurrent Neural Networks. *Neural Computation* (2013), Bd. 25(3): S. 626–649

[Tab20] Tabassum, Ayisha und Patil, Rajendra R: A survey on text pre-processing & feature extraction techniques in natural language processing. *International Research Journal of Engineering and Technology (IRJET)* (2020), Bd. 7(06): S. 4864–4867

[Tai20] Taillé, Bruno; Guigue, Vincent und Gallinari, Patrick: Contextualized Embeddings in Named-Entity Recognition: An Empirical Study on Generalization, in: *Lecture Notes in Computer Science*, Springer International Publishing (2020), S. 383–391

[Tan23] Tang, Mingjie; Yang, Weichun; Li, Yeli und Zeng, Qingtao: Research on multi-label long text classification algorithm based on transformer-LDA. *Fifth International Conference on Computer Information Science and Artificial Intelligence (CISAI 2022)* (2023), URL `http://dx.doi.org/10.1117/12.2667798`

[TKS03] Tjong Kim Sang, Erik F. und De Meulder, Fien: Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition, in: *Proceedings of the Seventh Conference on Natural Language Learning*

at HLT-NAACL 2003, S. 142–147, URL https://aclanthology.org/W03-0419

[Vas17] Vaswani, Ashish; Shazeer, Noam; Parmar, Niki; Uszkoreit, Jakob; Jones, Llion; Gomez, Aidan N; Kaiser, Lukasz und Polosukhin, Illia: Attention is All you Need, in: I. Guyon; U. Von Luxburg; S. Bengio; H. Wallach; R. Fergus; S. Vishwanathan und R. Garnett (Herausgeber) *Advances in Neural Information Processing Systems*, Bd. 30, Curran Associates, Inc., URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[Vyc19] Vychegzhanin, Sergey und Kotelnikov, Evgeny: Comparison of Named Entity Recognition Tools Applied to News Articles, in: *2019 Ivannikov Ispras Open Conference (ISPRAS)*, IEEE

[Wal06] Walker, Christopher; Strassel, Stephanie; Medero, Julie und Maeda, Kazuaki: ACE 2005 Multilingual Training Corpus (2006)

[Wan03] Wang, Sun-Chong: Artificial Neural Network, in: *Interdisciplinary Computing in Java Programming*, Springer US (2003), S. 81–100

[Wan21] Wang, Rui und Henao, Ricardo: Unsupervised Paraphrasing Consistency Training for Low Resource Named Entity Recognition, in: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics

[Wan23a] Wang, Chishe; Fan, Kangnan und Wang, Jie: Label Consistency Loss for Multi-label Decoding in Named Entity Recognition (2023)

[Wan23b] Wang, Peng; Wang, Zhe; Zhang, Xiaowang; Wang, Kewen und Feng, Zhiyong: Enhanced Named Entity Recognition through Joint Dependency Parsing, in: *2023 International Joint Conference on Neural Networks (IJCNN)*, IEEE

[Wan23c] Wang, Shuhe; Sun, Xiaofei; Li, Xiaoya; Ouyang, Rongbin; Wu, Fei; Zhang, Tianwei; Li, Jiwei und Wang, Guoyin: GPT-NER: Named Entity Recognition via Large Language Models (2023)

[Wat20] Watson, Richard T. und Webster, Jane: Analysing the past to prepare for the future: Writing a literature review a roadmap for release 2.0. *Journal of Decision Systems* (2020), Bd. 29(3): S. 129–147

[Web02] Webster, Jane und Watson, Richard T: Analyzing the past to prepare for the future: Writing a literature review. *MIS quarterly* (2002): S. xiii–xxiii

[Win21] Wings, Ivo; Nanda, Rohan und Adebayo, Kolawole John: A Context-Aware Approach for Extracting Hard and Soft Skills. *Procedia Computer Science* (2021), Bd. 193: S. 163–172

[Wos21] Wosiak, Agnieszka: Automated extraction of information from Polish resume documents in the IT recruitment process. *Procedia Computer Science* (2021), Bd. 192: S. 2432–2439

[Wu16] Wu, Yonghui; Schuster, Mike; Chen, Zhifeng; Le, Quoc V.; Norouzi, Mohammad; Macherey, Wolfgang; Krikun, Maxim; Cao, Yuan; Gao, Qin; Macherey, Klaus; Klingner, Jeff; Shah, Apurva; Johnson, Melvin; Liu, Xiaobing; Kaiser, Lukasz; Gouws, Stephan; Kato, Yoshikiyo; Kudo, Taku; Kazawa, Hideto; Stevens, Keith; Kurian, George; Patil, Nishant; Wang, Wei; Young, Cliff; Smith, Jason; Riesa, Jason; Rudnick, Alex; Vinyals, Oriol; Corrado, Greg; Hughes, Macduff und Dean, Jeffrey: Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation (2016)

[Wu20] Wu, Tong; Huang, Qingqiu; Liu, Ziwei; Wang, Yu und Lin, Dahua: Distribution-Balanced Loss for Multi-Label Classification in Long-Tailed Datasets (2020)

[Xia19] Xiao, Lin; Huang, Xin; Chen, Boli und Jing, Liping: Label-Specific Document Representation for Multi-Label Text Classification, in: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Association for Computational Linguistics

[Xun20] Xun, Guangxu; Jha, Kishlay; Sun, Jianhui und Zhang, Aidong: Correlation Networks for Extreme Multi-label Text Classification, in: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ACM

[Yad19] Yadav, Vikas und Bethard, Steven: A Survey on Recent Advances in Named Entity Recognition from Deep Learning models (2019)

[Yan18] Yang, Pengcheng; Sun, Xu; Li, Wei; Ma, Shuming; Wu, Wei und Wang, Houfeng: SGM: Sequence Generation Model for Multi-label Classification. *CoRR* (2018), Bd. abs/1806.04822, URL `http://arxiv.org/abs/1806.04822`

[Yan19] Yan, Hang; Deng, Bocao; Li, Xiaonan und Qiu, Xipeng: TENER: Adapting Transformer Encoder for Named Entity Recognition (2019)

[Yan21] Yang, Feihong; Wang, Xuwen; Ma, Hetong und Li, Jiao: Transformers-sklearn: a toolkit for medical language understanding with transformer-based models. *BMC Medical Informatics and Decision Making* (2021), Bd. 21(S2)

[Yan22] Yan, Yaoyao; Liu, Fangâai; Zhuang, Xuqiang und Ju, Jie: An R-Transformer_BiLSTM Model Based on Attention for Multi-label Text Classification. *Neural Processing Letters* (2022), Bd. 55(2): S. 1293...1316, URL `http://dx.doi.org/10.1007/s11063-022-10938-y`

[Yue23] Yue, Chongjian; Xu, Xinrun; Ma, Xiaojun; Du, Lun; Liu, Hengyu; Ding, Zhiming; Jiang, Yanbing; Han, Shi und Zhang, Dongmei: Leveraging LLMs for KPIs Retrieval from Hybrid Long-Document: A Comprehensive Framework and Dataset (2023)

[ZG17] Zukov-Gregoric, Andrej; Bachrach, Yoram; Minkovsky, Pasha; Coope, Sam und Maksak, Bogdan: Neural Named Entity Recognition Using a Self-Attention Mechanism, in: *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, IEEE

[ŽG18] Žukov-Gregorič, Andrej; Bachrach, Yoram und Coope, Sam: Named Entity Recognition With Parallel Recurrent Neural Networks, in: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Association for Computational Linguistics

[Zha21a] Zhang, Ximing; Zhang, Qian-Wen; Yan, Zhao; Liu, Ruifang und Cao, Yunbo: Enhancing Label Correlation Feedback in Multi-Label Text Classification via Multi-Task Learning (2021)

[Zha21b] Zhang, Xuelei; Song, Xinyu; Feng, Ao und Gao, Zhengjie: Multi-Self-Attention for Aspect Category Detection and Biomedical Multilabel Text Classification with BERT. *Mathematical Problems in Engineering* (2021), Bd. 2021: S. 1–6

[Zha21c] Zhao, Xintong; Greenberg, Jane; An, Yuan und Hu, Xiaohua Tony: Fine-Tuning BERT Model for Materials Named Entity Recognition, in: *2021 IEEE International Conference on Big Data (Big Data)*, IEEE

[Zha22a] Zhang, Daniel; Maslej, Nestor; Brynjolfsson, Erik; Etchemendy, John; Lyons, Terah; Manyika, James; Ngo, Helen; Niebles, Juan Carlos; Sellitto, Michael; Sakhaee, Ellie; Shoham, Yoav und Perrault, Jack Clark Raymond: The AI Index 2022 Annual Report (2022), URL `https://aiindex.stanford.edu/wp-content/uploads/2022/03/2022-AI-Index-Report_Master.pdf`, aI Index Steering Committee, Stanford Institute for Human-Centered AI, Stanford University, aufgerufen am 03.09.2023

[Zha22b] Zhang, Sheng; Cheng, Hao; Gao, Jianfeng und Poon, Hoifung: Optimizing Bi-Encoder for Named Entity Recognition via Contrastive Learning (2022)

[Zha23] Zhai, Hanming; Lv, Xiaojun; Hou, Zhiwen; Tong, Xin und Bu, Fanliang: MLNet: a multi-level multimodal named entity recognition architecture. *Frontiers in Neurorobotics* (2023), Bd. 17

[Zhe15] Zheng, Alice: *Evaluating machine learning models: A beginner's guide to key concepts and pitfalls* (2015)

[Zho20] Zhong, Zexuan und Chen, Danqi: A Frustratingly Easy Approach for Entity and Relation Extraction (2020)

Abkürzungsverzeichnis

MLTC Multi-Label-Textklassifikation

NER Named Entity Recognition

KI Künstliche Intelligenz

NN Neuronales Netz

FFNN Feedforward neuronales Netz

RNN Rekurrentes neuronales Netz

VGP Vanishing Gradient-Problem

LSTM Long Short-Term Memory

BiLSTM Bidirektionales Long Short-Term Memory

LDA Latent Dirichlet Allocation

BERT Bidirectional Encoder Representations from Transformers

GPT Generative Pretrained Transformer

RoBERTa Robustly Optimized BERT Pre-training Approach

USE Universal Sentence Encoder

ELMo Embeddings from Language Models

HL Hamming Loss

FA Funktionale Anforderungen

NFA Nicht-funktionale Anforderungen

SLR Strukturierte Literaturrecherche

SLRFF Forschungsfrage der strukturierten Literaturrecherche

CRF Conditional Random Fields

R21578 Reuters-21578, Distribution 1.0

IOB Inside Outside Beginning

BCE Binary Cross-Entropy

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

.