HAW HAMBURG

BACHELORTHESIS
Anna Gennadinik

Text Mining und Relation Extraction in der Markergenforschung

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Computer Science and Engineering
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Anna Gennadinik

Text Mining und Relation Extraction in der Markergenforschung

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Wirtschaftsinformatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Zukunft
Zweitgutachter: Prof. Dr. Tropmann-Frick

Eingereicht am: 23. Februar 2022

Anna Gennadinik

Thema der Arbeit

Text Mining und Relation Extraction in der Markergenforschung

Stichworte

Markergen, SNP, NLP, Text Mining, Relation Extraction

Kurzzusammenfassung

Im Rahmen dieser Arbeit werden drei Modelle implementiert, die Relationen zwischen Erkrankungen und genetischen Markern in der biomedizinischen Literatur erkennen und extrahieren. Das Ziel dieser Arbeit ist es, drei Relation Extraction Verfahren zu vergleichen und einen Prototyp zu erstellen, der die Verarbeitung unstrukturierter biomedizinischer Texte und die Extrahierung wichtiger Zusammenhänge ermöglicht.

Anna Gennadinik

Title of Thesis

Text Mining and Relation Extraction in the Research of Genetic Markers

Keywords

Genetic markers, SNP, NLP, Text Mining, Relation Extraction

Abstract

The aim of this work is to compare three relation extraction methods and to create a prototype that enables the processing of biomedical texts and the extraction of relationships between diseases and genetic markers.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Problemstellung

Schon immer haben sich Menschen Gedanken um ihre Herkunft gemacht. Woher kommen sie? Wer waren ihre Vorfahren? Ein anderes brennendes Thema ist die menschliche Gesundheit – warum wird man krank und wie kann man Erkrankungen vorbeugen?

Beim Versuch diese Fragen zu beantworten, unterziehen sich Millionen Menschen einem Gentest. Schon seit über einem Jahrzehnt stehen Direct-to-Consumer (DTC)-Gentests den Verbrauchern ohne Arztbesuch direkt zur Verfügung.

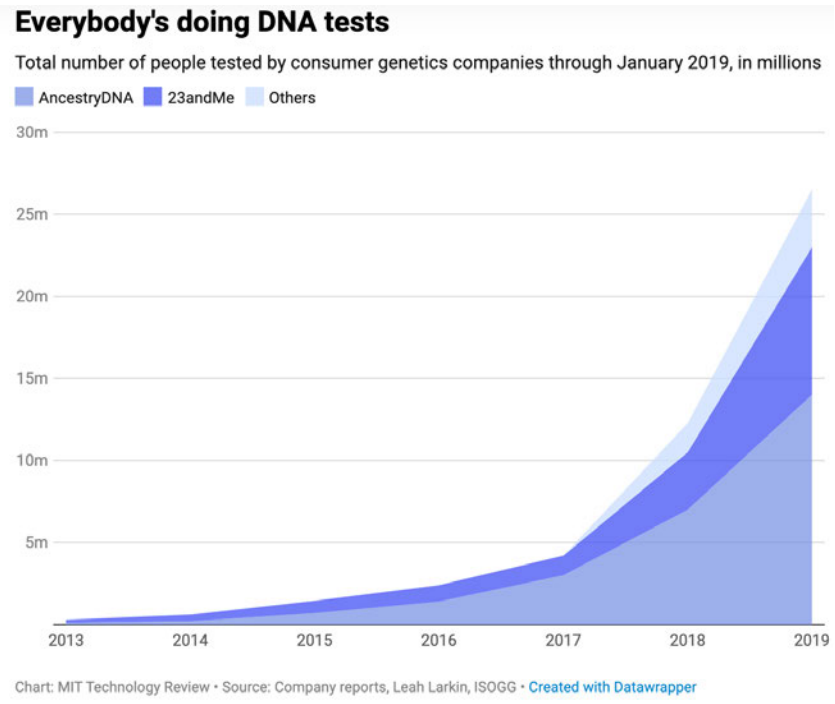


Abbildung 1 Verbreitung von Gentests 2013-2019 [1]

Laut einer MIT Studie haben im Jahr 2019 mehr als 26 Millionen Menschen einen Gentest erworben [1].

Die Forscher im Bereich Biomedizin kommen dieser Nachfrage nach – die Anzahl der Publikationen über genetische Disposition für verschiedene Krankheiten nimmt jedes Jahr zu.

Selbst für Fachleute stellt dieser riesige Informationsstrom eine Herausforderung dar. Viele Verbraucher der DTC-Gentests sind um so weniger bereit, sich in das Thema einzulesen.

Angesichts dessen wächst der Bedarf an domainspezifischen Information Retrieval Systemen, die in der Lage sind, das Wissen aus biomedizinischer Literatur zu extrahieren und systematisieren.

1.2 Zielsetzung

Im Rahmen dieser Arbeit werden drei Modelle implementiert, die Relationen zwischen Erkrankungen und genetischen Markern in der biomedizinischen Literatur erkennen und extrahieren. Als Datenquelle wird textbasierte Datenbank PubMed verwendet.

Das erste Modell extrahiert die Relationen mit Hilfe von Text Mining Regeln. Das zweite Modell nutzt dafür Abhängigkeitsbäume, die die Beziehungen zwischen den Satzgliedern im Satz darstellen. Drittes Modell basiert auf der Transformer Deep-Learning-Architektur. Drei Algorithmen werden bewertet und verglichen.

Das Ziel dieser Arbeit ist es, einen Proof of Concept durchzuführen und einen Prototyp zu erstellen, der die Verarbeitung unstrukturierter biomedizinischer Texte und die Extrahierung wichtiger Zusammenhänge ermöglicht.

1.3 Aufbau der Arbeit

Kapitel 2 und 3 geben einen theoretischen Überblick über zwei Bestandteile dieser Arbeit. Im Kapitel 2 handelt es sich um den Aufbau des Genoms, die Übertragung von genetischen Informationen und die Markergene. Kapitel 3 gibt einen Einblick in die Grundlagen des Natural Language Processing. Kapitel 4 „Related Works“ beschreibt den aktuellen Stand der Forschung in diesem Gebiet.

Kapitel 5, 6 und 7 beziehen sich auf den praktischen Teil. 5. Kapitel enthält Anforderungen und einen schematischen Überblick der Architektur der Applikation. Kapitel 6 beschreibt die Implementierung und im siebten Abschnitt werden die Ergebnisse der Arbeit zusammengefasst.

2 Einführung in die Genetik

2.1 Historischer Hintergrund

Genetik ist die Wissenschaft, die die Weitergabe von Merkmalen von einer zur nächsten Generation untersucht [2]. Als Begründer der klassischen Genetik gilt Gregor Johann Mendel, der im Jahr 1865 drei Grundregeln der Mendelschen Vererbung vorstellte. Diese Regeln, die als Ergebnis der Kreuzungsversuche mit Erbsenpflanzen entstanden sind, beschreiben das Verhalten von Genen bei ihrer Vererbung.

Vier Jahre später (1869) wurden die Nukleinsäuren entdeckt. 1888 fand man die Chromosomen in den menschlichen Zellen und stellte fest, dass der chemische Aufbau von Chromatin, aus dem die Chromosomen bestehen, und der chemische Aufbau von Nukleinsäuren identisch sind. Daraus ließ sich schließen, dass die Nukleinsäuren bei der Übertragung von der Erbinformation eine Rolle spielen. Erst im Jahr 1944 zeigte Oswald Theodore Avery mit Hilfe von Transformationsexperimenten an Pneumokokken, dass eben die Nukleinsäuren und nicht die Proteine in den Chromosomen der Träger der Erbinformation sind [3]. Schließlich wurde 1953 durch Watson, Crick und Wilkins die DNA-Doppelhelix-Struktur aufgeklärt.

In der Mitte der 1970er-Jahre wurden verschiedene Methoden entwickelt, um die Reihenfolge (Sequenz) der Nukleotide in den Nukleinsäuren zu ermitteln. Da die Abfolge der Nukleinbasen die eigentliche genetische Information beinhaltet, ermöglichte die Technik der DNA-Sequenzierung das Ablesen des so genannten „genetischen Codes“ [3].

1990 startete in den USA das internationale Humangenomprojekt. Sein Ziel war das vollständige Entschlüsseln des menschlichen Genoms (Erbgutes). Etwa 3 Milliarden Basenpaare der DNA wurden sequenziert und die Ergebnisse im Jahr 2004 publiziert [4].

Die vollständige Sequenzierung des Genoms bildet die Basis für die Erforschung vieler Erbkrankheiten und neuer Therapiemöglichkeiten sowie für die Prävention von Erbkrankheiten. Allerdings müssen die DNA-Sequenzen zuerst analysiert und interpretiert werden, um z.B. die Lage und die Funktion bestimmter Gene festzustellen. Diese Aufgabe hat sich im Laufe des HGP als äußerst schwierig erwiesen [4].

2.2 DNA Aufbau

Desoxyribonucleinsäure (DNS, DNA) ist in allen Organismen vorhanden und trägt die Erbinformation. Bei Eukaryoten, d.h. Organismen, die einen Zellkern besitzen, befindet sich die DNA im Zellkern.

Chemisch gesehen besteht ein DNA-Molekül aus drei Komponenten: einem Phosphatrest, dem Desoxyribose-Zucker und einer Nukleinbase. Diese drei Komponenten bilden ein Nukleotid. Tausende dieser Nukleotide bilden ein DNA-Molekül.

Die DNA kommt in Form einer Doppelhelix vor, die aus zwei entgegengesetzten Einzelsträngen besteht. Die beiden DNA-Stränge werden durch Wasserstoffbrücken zwischen den Basen verknüpft [5].

Es existieren vier Nukleinbasen: Adenin (A), Guanin (G), Thymin (T) und Cytosin (C). In der RNA wird Thymin durch Uracil (U) ersetzt. Die Basen bestehen aus Kohlenstoff- (C), Wasserstoff- (H), Stickstoff- (N) und Sauerstoffatomen (O):

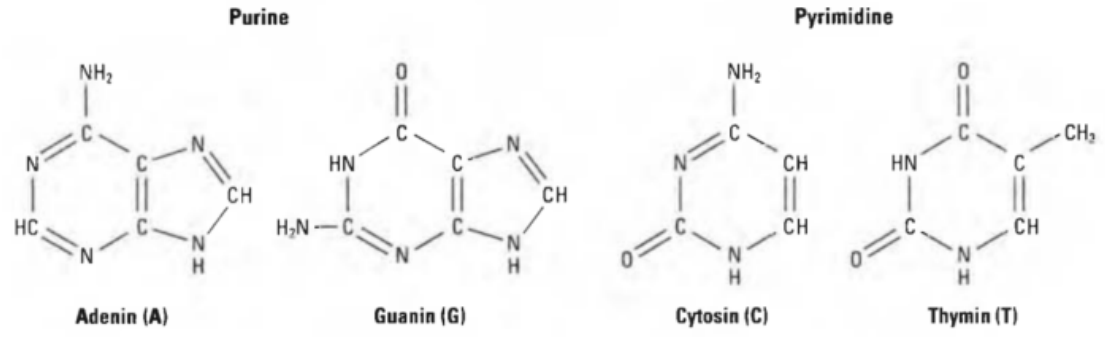


Abbildung 2 Chemischer Aufbau der Nukleinbasen [2]

Cytosin und Thymin sind Pyrimidine, d.h. bestehen aus einem sechskantigen Ring. Adenin und Guanin sind Purine – sie bestehen aus zwei sich überlappenden Ringsystemen mit 5 und 6 Kanten.

Einer Purinbase steht eine Pyrimidinbase gegenüber. In der Regel werden Guanin mit Cytosin und Adenin mit Thymin verbunden. Solche miteinander verbundenen Basen heißen Basenpaare und die durch Basenpaare verknüpften DNA-Stränge – komplementäre Stränge.

Die Abfolge der vier DNA-Basen eines Stranges wird als Basensequenz bezeichnet. Die Basensequenz beinhaltet die Erbinformation. Darüber hinaus sind die Nukleinbasen für die Replikation notwendig.

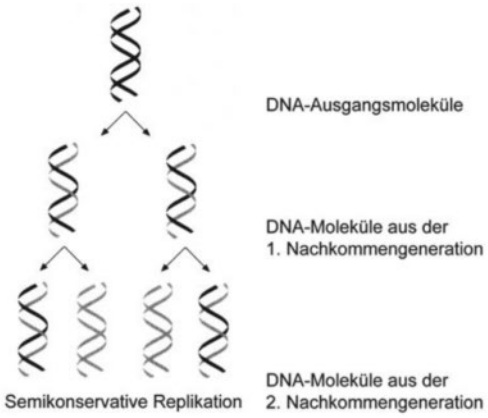


Abbildung 3 Semikonservative Replikation [3]

Unter Replikation wird die Verdoppelung der DNA bei der Zellteilung verstanden. Dabei trennen sich die beiden DNA-Hälften, und an jedem Elternstrang wird ein komplementärer neuer Strang synthetisiert. Es ist möglich, weil seine Struktur durch die Basenfolge in dem alten Strang vollständig festgelegt ist. Auf diese Weise entstehen aus einer Doppelhelix zwei identische neue Helices [3].

Diese Form der Replikation heißt semikonservativ. Die DNA-Replikation ist mit häufigem Fehleinbau von Nukleotiden (Mutationen) verbunden.

2.3 Informationsübertragung

Die DNA ist lediglich ein Informationsträger, sie kann auch mit einem Bauplan für die Proteinen verglichen werden. Mit Hilfe von der Ribonukleinsäure wird die genetische Information umgesetzt, indem nach ihrer Vorgabe Proteine aus Aminosäuren aufgebaut werden. Dieser Prozess heißt Genexpression [6].

Die RNA unterscheidet sich von der DNA dadurch, dass sie Uracil anstelle von Thymin und Ribose-Zucker anstelle von Desoxyribose-Zucker enthält. Durch 2 Hydroxylgruppen in der

Ribose ist RNA weniger stabil als DNA. Außerdem ist sie einzelsträngig. Es gibt drei Arten von RNA:

- **Messenger-RNA** (mRNA) überträgt die Information aus dem Zellkern ins Zytoplasma und legt die Reihenfolge fest, in dem die Polymerisierung der Aminosäuren zu Polypeptiden (Translation) an den Ribosomen erfolgt.
- **Transfer-RNA** (tRNA) liefert die Aminosäuren zu den Ribosomen.
- Aus der **ribosomalen RNA** (rRNA) und Proteinen werden Ribosomen aufgebaut.

Der erste Schritt der Genexpression heißt Transkription. Dieser Prozess findet mit Hilfe von Enzym RNA-Polymerase statt. RNA-Polymerase liest bestimmte Sequenzen eines einzelnen DNA-Strangs, die durch ein Startsignal (Promotor) und ein Endsignal (Terminationssignal) gekennzeichnet sind. Signalsequenzen legen außerdem fest, welcher von den beiden DNA-Strängen in RNA umgesetzt wird. Als Endprodukt der RNA-Polymerase-Aktivität liegt ein RNA-Einzelstrangmolekül vor [3].

Das entstandene mRNA-Molekül enthält oft unnötige Sequenzen, die nicht zu Proteinen übersetzt werden und folglich für keinen Phänotyp zuständig sind. Solche Sequenzen heißen Introns. Die codierenden Sequenzen nennt man Exons. Nach der Transkription müssen die Introns entfernt und die Exons ohne entstandene Lücken aneinander gefügt werden [2]. Dieser Prozess wird als Spleißen (Splicing, auch alternatives Spleißen) bezeichnet. Nach dem Spleißen verlässt das mRNA den Zellkern, um die Translation durchzulaufen.

Komplexe Phänotypen (z.B. Augenfarbe) sind das Ergebnis der Kombination verschiedener Proteine. Proteine bestehen aus langen Ketten von Aminosäuren (von 50 bis 1.000). Insgesamt gibt es 20 verschiedene Arten von Aminosäuren, was zu einer enormen Anzahl an Kombinationsmöglichkeiten führt.

Jede von 20 Aminosäuren wird mit drei Nukleinbasen (Triplet) in RNA kodiert. Da 4^3 bzw. 64 Kombinationen von drei Basen möglich sind, können einige Aminosäuren von mehreren Kombinationen kodiert werden. Dabei ist einer Kombination nur eine einzige Aminosäure zugewiesen. Einige der 64 Kombinationen bedeuten den Start- und den Endsignal für die Ribosomen, die die Translation durchführen.

Als die mRNA ins Zytoplasma gelangt, wird sie von Ribosomen „abgelesen“. Gleichzeitig liefert die tRNA die benötigten Aminosäuren zu den Ribosomen, so dass sie die Proteine aufbauen.

Der Begriff „Gen“ wurde 1909 erfunden. Seitdem haben mehrere Wissenschaftler versucht, diesen Begriff präzise zu definieren. Heutzutage gilt, dass ein Gen ein DNA-Abschnitt ist, der für ein Protein codiert. Da jede DNA-Sequenz durch alternatives Spleißen verschieden abgelesen werden kann, kann ein Gen für verschiedene Proteine codieren [7].

2.4 Markergene

Wie bereits im Abschnitt 2.1 erwähnt, wurde das menschliche Genom 2001 erfolgreich sequenziert. Es stellte sich fest, dass ein Mensch etwa 20000-25000 Genen besitzt. Die nächste große Herausforderung ist die Aufklärung der Funktionen dieser Gene. Hierfür spielen molekulare Marker eine große Rolle.

Als molekulare Marker werden Stellen im Genom bezeichnet, die in einer Population einen hohen Polymorphismus aufweisen [8]. Anders formuliert: Markergene sind eindeutig identifizierbare, kurze DNA-Abschnitte, deren Ort im Genom bekannt ist, und die mit bestimmten Merkmalen korrelieren können.

Im Rahmen des Humangenomprojektes wurden 3,7 Millionen Einzelbasen-Polymorphismen identifiziert (engl. single nucleotide polymorphisms, SNPs, ausgesprochen „Snips“). Ein SNP ist ein Einzelbasenaustausch, der in der DNA recht häufig vorkommt – ungefähr einmal pro 300 Nukleinbasen. D.h. dass im menschlichen Genom, der aus 3 Milliarden Basenpaaren besteht, 10 Millionen Positionen variabel sind [9]. Einige dieser Variationen können mit bestimmten Erbkrankheiten in Verbindung stehen.

Extrem hohe Markerdichte und die Möglichkeit der schnellen automatisierten Erkennung macht die SNP-Analyse sehr populär im Vergleich zu den anderen Markergenen wie z.B. Mikrosatelliten oder Restriktionslängenpolymorphismen (engl. restriction fragment length polymorphism, RFLP).

Im Rahmen dieser Arbeit werden nur die Einzelbasen-Polymorphismen betrachtet.

3 Methoden des Natural Language Processing

3.1 Einführung in Natural Language Processing

Maschinelle Verarbeitung der menschlichen Sprache wird auch als Natural Language Processing (NLP) bezeichnet. Durch NLP können Menschen mit den Computern auf natürliche Weise kommunizieren, sodass diese unsere Sprache verstehen. Das Ziel des NLP ist es, gesprochene und geschriebene Sprache zu erkennen, zu analysieren und deren Sinn zur weiteren Verarbeitung zu extrahieren [9]. Hierfür reicht nicht das Verständnis von einzelnen Wörtern und Sätzen. Komplette Textzusammenhänge müssen erkannt und Sachverhalte verstanden werden.

NLP ist – auch wie Machine Learning – ein Teilbereich der Künstlichen Intelligenz. NLP verwendet einige Techniken des maschinellen Lernens, beschränkt sich aber nicht darauf.

Eine Herausforderung für das Natural Language Processing stellt die Komplexität der menschlichen Sprache und deren Mehrdeutigkeit dar. Computer können nicht wie Menschen auf Erfahrungen zum besseren Verstehen von Sprache zurückgreifen, d.h. sie haben kein Weltwissen. Um Textbedeutungen ganzheitlich zu erkennen, ist es notwendig, im Vorfeld große Datenmengen zu erfassen und bereits erkannte Muster zu verwenden. Hierfür werden überwachtes und unüberwachtes Lernen sowie Big Data-Technologien verwendet.

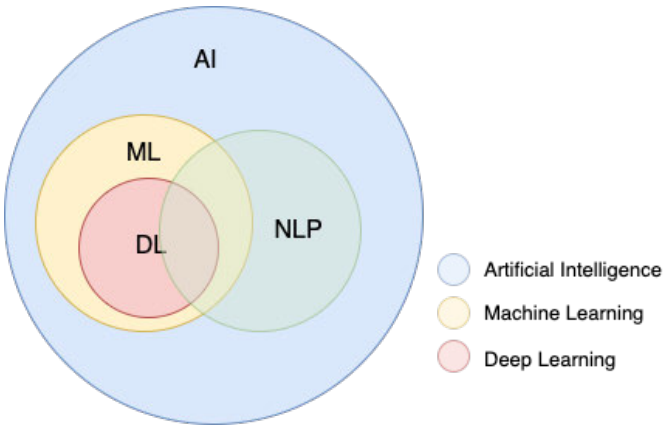


Abbildung 4 NLP als ein Teil der Künstlichen Intelligenz

Maschinelle Übersetzung war die erste und historisch wichtigste Aufgabe des NLP. Ihre Entwicklung wurde durch das militärische Interesse geprägt. Eines der frühesten Projekte war ein Russisch-Englisch-Übersetzungsprogramm für das US-Militär, das in den 50er Jahren des 20. Jahrhunderts stattfand [11]. Die Qualität der Übersetzungen war unbefriedigend, jedoch wurde durch diese Untersuchungen ein neues Forschungsgebiet eröffnet.

Auch heutzutage ist maschinelle Übersetzung eine der naheliegenden Anwendungen der NLP-Techniken. Zu den anderen aktuellen Aufgaben des NLP gehören:

- **Textklassifikation und Textzusammensetzung** werden eingesetzt, um einen Überblick über große Datenmengen zu bekommen. Die automatische Textzusammensetzung hat die Aufgabe, ohne menschliche Hilfe eine präzise und flüssige Zusammenfassung zu erstellen, ohne dabei die Bedeutung des Originaltextdokuments zu verlieren [12]. Textklassifikation strukturiert die Daten, indem die Texte in Gruppen (Topics) eingeteilt werden. Informationserschließung (engl. Information Retrieval) und darauffolgende Textklassifikation und Textzusammensetzung sind wichtige Instrumente des Text Minings. Durch Text Mining erschließt man Kerninformationen aus unstrukturierten Textdaten.
- **Fragebeantwortung- und Dialogsysteme** sind jetzt zum Teil des Alltags geworden. Sprach-Assistenten empfangen frei formulierte Fragen in natürlicher Sprache und extrahieren die entsprechenden Antworten aus sehr großen Beständen von unstrukturierten Dokumenten [12]. ML-basierte Sprach-Assistenten und Chatbots sind in der Lage, semi-automatisiert zu lernen. Regelbasierte Chatbots können nur auf die vorher definierten Fragen antworten.
- **Named Entity Recognition (NER) und Relationship Extraction**. Diese Aufgaben haben keinen Selbstzweck, sondern dienen als Hilfsmittel für das bessere Verständnis der Sprache auf semantischer Ebene.

  NER bezeichnet die Erkennung und Klassifikation von Eigennamen in Texten. Eigenname ist ein sprachlicher Ausdruck, der eine Entität beschreibt, z. B. Personen-, Firmen-, Produktnamen, ein Datum oder ein Maß [12]. NER ist besonders wichtig für die biomedizinischen Anwendungen, bei denen die Terminologie ein großes Problem

darstellt. Die biomedizinischen Korpora werden mit den zusätzlichen Entitätstypen wie „Protein“, „DNA“ und „RNA“ markiert [13].

Relationship Extraction (Relationsextraktion) ist die Fortsetzung der NER und hilft, die Zusammenhänge zwischen den durch NER erkannten Entitäten zu finden. In den Sätzen “Microsoft acquired Powerset” und “Powerset was acquired by Microsoft” sind schließlich nicht nur die als Unternehmensnamen erkannte „Microsoft“ und „Powerset“ wichtig, sondern die Beziehung zwischen den beiden Unternehmen [14].

Jede NLP-Aufgabe braucht eine maßgeschneiderte Lösung, allerdings gibt es ein Muster-Workflow, der einen Überblick über die möglichen Schritte gibt:

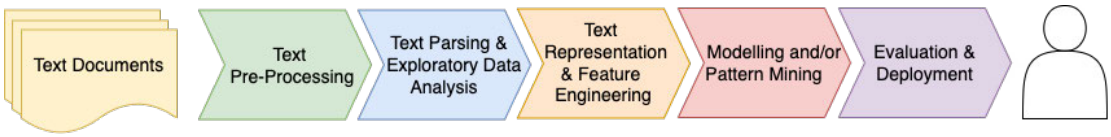


Abbildung 5 NLP Workflow [14]

Als Input bekommt man ein Dataset von Texten zu einem bestimmten Thema, d.h. einen Korpus. Zuerst werden diese Texte vorbereitet – bereinigt und normalisiert.

Als nächstes kann Exploratory Data Analysis durchgeführt werden. Das ist ein Analyseansatz, um mit Hilfe von visuellen und statistischen Methoden die Daten besser kennenzulernen, Zusammenhänge zu finden und Muster zu entdecken.

Dritter Schritt ist Text Representation und Feature Engineering. Algorithmen können mit den komplexen und unstrukturierten Texten nicht arbeiten, deswegen geht es bei der Text Representation um die Umwandlung der Wörter in Vektoren mit Hilfe von Feature Engineering. Features sind Input Parameter für ML-Algorithmen, also kategoriale oder numerische Größen, wie zum Beispiel:

- welche Wörter in einem Text auftauchen — die Menge dieser Wörter wird auch bag of words genannt,
- wie oft diese Wörter jeweils auftauchen,
- ihre relative Häufigkeit

Als nächstes wird ein Modell entworfen und erstellt. Es ist wichtig zu bemerken, dass nicht alle Aufgaben mit Machine Learning gelöst werden müssen. In einigen Fällen reichen nur statistische Methoden aus. Auch reguläre Ausdrücke dürfen nicht unterschätzt werden, z.B. bei Named Entity Recognition, um Datum- oder Email-Entitäten zu erkennen.

3.2 Natural Language Processing Pipeline

Der erste Schritt des Workflows – Text Preprocessing – verdient besondere Aufmerksamkeit, denn damit fängt jede NLP-Aufgabe an. Der Prozess der Bereinigung und Normalisierung der Daten heißt NLP-Pipeline.

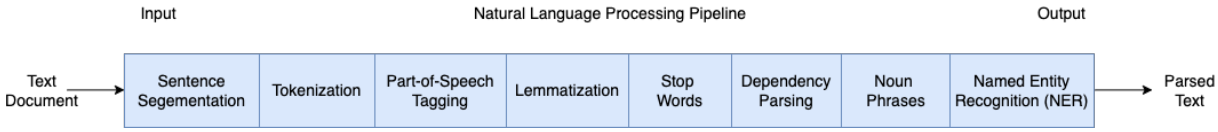


Abbildung 6 NLP Pipeline

Die ersten zwei Schritte sind Satzsegmentierung und Tokenisierung. Ein Input-Text wird zuerst in die Sätze und anschließend die Sätze in Tokens zerlegt. Ein Token ist die kleinste Texteinheit und kann ein Wort, eine Wortkombination, eine Zahl oder ein Satzzeichen sein. Als ein Token wird oft eine Zeichenfolge betrachtet, die am Anfang und am Ende mit jeweils einem Leerzeichen getrennt ist [9]. Dieser Ansatz berücksichtigt nicht die Satzzeichen, die am Ende eines Wortes stehen (z.B. Punkte, Kommata, Semikolone) und die als separate Tokens betrachtet werden sollen, weil sie einen semantischen Inhalt haben. Außerdem werden Wörter nicht in allen Sprachen durch Leerzeichen getrennt.

Auch Segmentierung stellt keine triviale Aufgabe dar. Die Trennung der Sätze an den Punkten mithilfe von regulären Ausdrücken liefert häufig inkorrekte Ergebnisse. Ein Punkt kann mehrere Bedeutungen in einem Satz haben – z.B. am Ende einer Abkürzung oder in einer Bruchzahl stehen. Außerdem werden Sätze nicht in allen Sprachen mit Punkten getrennt.

Der nächste Schritt der Datennormalisierung bringt eine noch größere Herausforderung mit sich. In diesem Schritt werden gleichartige Wörter auf die gleiche Form zurückgeführt, um die Dimension der Daten zu reduzieren [15]. Es wird zwischen zwei möglichen Verfahren entschieden – Lemmatisierung oder Stammformreduktion (engl. Stemming).

Durch das Stemming werden Wörter mit unterschiedlichen Suffixen auf denselben Stamm regelbasiert reduziert [12]. Dieser Stamm muss nicht unbedingt ein echtes Wort sein.

Bei der Lemmatisierung werden die Wörter auf ihre Grundformen reduziert, die in einer lexikalischen Datenbank nachgeschlagen werden [12]. Außerdem führt man vorher für jedes Wort ein Part-Of-Speech-Tagging durch. Das macht Lemmatisierung aufwändiger und langsamer als Stemming. Aus diesem Grund wird bei der Verarbeitung sehr großer Datenmengen Stemming bevorzugt.

Im nächsten Schritt wird der Rechenaufwand durch die Entfernung der Stoppwörter reduziert. Stoppwörter sind die häufigsten Wörter in einer Sprache bzw. in einem Dataset, die entfernt werden können, ohne die Bedeutung des Textes zu ändern [9]. Zu den Stoppwörtern gehören in der Regel Präpositionen, Konjunktionen und Artikel. Man muss aber ganz genau aufpassen, welche Wörter in die Stoppwörter-Liste einzutragen sind. Einige Wörter enthalten zwar wenig Information, zeigen aber die Beziehungen zwischen den anderen Wörtern im Satz, was für Dependency Parsing wichtig ist.

Erst nach den oben aufgeführten Schritten stehen die normalisierten Daten zur Analyse bereit.

3.3 Relation Extraction

3.3.1 Methoden der Relation Extraction

Für die Extrahierung der semantischen Informationen aus den natürlichen Texten reicht Named Entity Recognition nicht aus. In dem folgenden Beispiel wurden eine SNP-Entität und eine Krankheit-Entität identifiziert:

Abbildung 7 Beispiel einer Entitätenerkennung

Zwar wurden Entitäten SNP *rs2241766* und Krankheit *cancer* erkannt, bleibt immer noch unklar, in welcher Beziehung sie zueinanderstehen. Die Kerninformationen des Satzes wurden dementsprechend nicht offengelegt.

Die Aufgabe der Extrahierung semantischer Beziehungen zwischen Entitäten heißt Relation Extraction (RE). NER und RE sind zwei wichtige Unteraufgaben der Information Extraction (IE). Information Extraction liegt der maschinellen Übersetzung, den Fragebeantwortung-Systemen und der Event Extraction zugrunde [16].

Eine Relation ist definiert in der Form eines $n+1$-Tupels $t = (e_1, e_2, ..., e_n, r)$, wobei $e_1, e_2, \ldots, e_n$ n Entitäten in dem Dokument D sind und r eine Relation zwischen diesen n Entitäten ist. Derzeit stehen die binären Beziehungen im Fokus der Forschung, und in den meisten Fällen stehen die beiden Entitäten im selben Satz [17].

Ein Beispiel für solch eine Relation ist auf der Abbildung 7 aufgeführt. Der Satz enthält ein 3-Tupel [rs2241766, cancer, wird_assoziiert].

“At codons 12, the occurence of point mutations from G to T were observed” ist ein Beispiel für eine Beziehung höherer Ordnung. Diese Relation kann in der Form eines 5-Tupels [codon, 12, G, T, Punktmutation] dargestellt werden.

Es existieren folgende Relation Extraction Ansätze [16]:

- Rule-based RE
- Semi-Supervised RE
- Supervised RE
- Distantly Supervised RE
- Unsupervised RE

Regel-basierte RE Systeme beruhen auf manuell erstellten domainspezifischen pattern-matching Regeln. Die Regeln hängen von dem Kontext der Aufgabe ab und können deswegen nur auf eine Menge ähnlicher Texte mit einer begrenzten Anzahl Beziehungstypen angewendet werden [18].

Die Regeln können bespielweise die Form $[X, \alpha i, Y]$ haben, wobei X und Y Entitätstypen sind, $A = \{\alpha \mid \alpha$ ist ein Schlüsselwort$\}$ und $\alpha i \in A$.

So ein einfaches Pattern würde viele False Positives zurückgeben. Filtern nach Wortart oder Entitätstyp kann die Präzision der Regeln erhöhen. Durch Dependency Parsing können

syntaktische Abhängigkeiten zwischen den Entitäten festgestellt werden, was zu einem besseren Recall führt.

Regel-basierte RE Systeme sind schwer übertragbar auf andere Domänen und erfordern viel Handarbeit. Sie können jedoch effektiv eingesetzt werden, wenn die Aufgabe darin besteht, in klar definierten Domänen bzw. Dokumentensammlungen schnell zu suchen und grobe Ergebnisse zu erzielen.

Supervised RE betrachtet Relation Extraction als ein Klassifikationsproblem. Ein binärer Klassifikator bekommt bestimmte Text Features als Input und wird dadurch trainiert, die Beziehungen zwischen zwei Entitäten zu identifizieren. Für den Satz $S = w_1, w_2, .., e_1.., w_j, ..e_2, .., w_n$, wobei e_1 und e_2 Entitäten sind, sieht die Abbildung wie folgt aus [17]:

$$f_R(T(S)) = \begin{cases} 1, & wenn\ e_1\ und\ e_2\ in\ Beziehung\ stehen \\ 0, & wenn\ e_1\ und\ e_2\ nicht\ in\ Beziehung\ stehen \end{cases}$$

Funktion T extrahiert Features aus dem Satz. Als Features können am Beispiel des Satzes „researches found out that rs266729 was associated with type 2 diabetes in the obese group only" die in der folgenden Tabelle aufgeführten Merkmale betrachtet werden:

Table 1 Relation Extraction Features [19]

Entität-Features	
e_1 Entitätstyp	SNP
e_2 Entitätstyp	DISEASE
e_1 Entitätstyp Head	rs266729
e_2 Entitätstyp Head	diabetes (in der Phrase „type 2 diabetes")
konkatenierte Entitätstypen	SNPDISEASE
Wort-Features	
Part of Speech Tag	rs266729 (NOUN), associated (VERB), diabetes (NOUN)

Stamm	associated→associate
Wörter-Prädiktoren	associated
Distanz zwischen e_1 und e_2 in Tokens	3
Bag of Words zwischen e_1 und e_2	was associated with
Bag of Words vor e_1	researches found out that
Bag of Words vor e_2	in the obese group only
Syntaktische Features	
Chunk base-phrase Pfade	NP(rs266729) VP(was associated) PP(with) NP(type 2 diabetes)
Dependency-Tree Pfade	rs266729$\leftarrow_{nsubjpass}$associated$\rightarrow_{nmod}$diabetes

Neben dem Feature-basierten Ansatz existiert auch der Kernel-basierte Ansatz. Diese Methode braucht kein Feature Engineering und bemisst stattdessen die strukturelle Ähnlichkeit zwischen zwei Wort- oder Zeichenfolgen bzw. zwei Abhängigkeitsbäumen [20]. Wenn zwei Sätze s_{train} und s_{test} eine ähnliche syntaktische Struktur haben, gehören die Entitätspaare von s_{train} und s_{test} höchstwahrscheinlich zu demselben Beziehungstyp.

Semi-Supervised RE. Dieser Ansatz beginnt mit einem Set der handgefertigten Regeln oder mit den so genannten *seed tuples* – beispielhaften Relationen. Das Modell geht iterativ den unmarkierten Text durch und findet neue Patterns bzw. Relationen auf Basis bereits vorhandener Informationen.

Snowball ist ein klassischer Semi-Supervised Relation Extraction Algorithmus aus dem Jahr 2000.

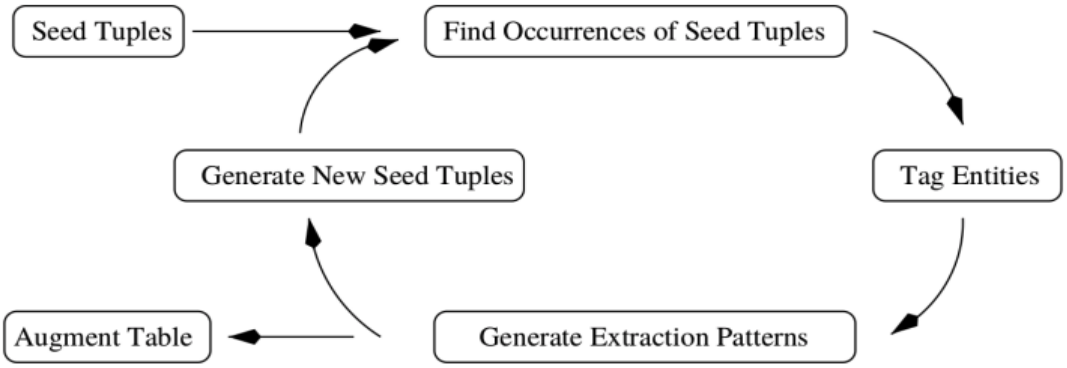


Abbildung 8 Snowball Relation Extraction Algorithmus [21]

Der Snowball setzt Named Entity Recognition ein, um neue Patterns zu erzeugen und seed tuples zu ergänzen. Ein falsches Muster kann aber zu neuen falschen Mustern führen, was das Pattern Set mit jeder Iteration immer fehleranfälliger macht.

Der andere Nachteil besteht darin, dass jeder Beziehungstyp manuell bereitgestellte Muster erfordert. Dennoch ist der menschliche Aufwand weniger als bei der überwachten Relation Extraction.

Distantly Supervised RE kombiniert den überwachten und den unüberwachten Ansatz. Distant Supervision benötigt keine markierten Daten, sondern nutzt eine oder mehrere Knowledge Bases (KB) als primäre Informationsquelle [22]. Die Annahme ist, dass wenn es eine Relation zwischen zwei bestimmten Entitäten in einer KB gibt, beinhalten alle Sätze im Korpus mit diesen zwei Entitäten genau diese Relation. Aus diesen Sätzen werden dann Features extrahiert, viele von denen irreführend sein können, denn nicht alle Sätze mit zwei Entitäten beinhalten ein und dieselbe Relation. Falsche Features werden anhand eines probabilistischen Klassifikators erkannt [23].

Seit den 2010er nimmt der Einsatz der neuronalen Netze für die Relation Extraction zu. CNN zeigen gute Ergebnisse bei der Lösung der IE-Probleme. Auch die zu RNN gehörenden LSTM (Long Short Term Memory) gewinnen an Bedeutung im NLP-Bereich [24]. Sie sind fähig, langfristige Kontextabhängigkeiten zu identifizieren, was für das tatsächliche Verstehen der Texte erforderlich ist.

Der nächste Evolutionsschritt sind Transformer Modelle. Im Gegensatz zu RNN verarbeiten Transformers die Daten nicht der Reihe nach. Um einen Satz zu verarbeiten, identifiziert der Transformer den Kontext für jeden Token, anstatt ihn vom ersten bis zum letzten Wort durchzugehen. Dies ermöglicht Parallelisierung und reduziert daher die Trainingszeiten [25]. Kürzere Trainingszeiten führten zur Entwicklung der vortrainierten Systeme wie BERT. Diese bereits vortrainierten Modelle können auf spezifische Aufgaben abgestimmt werden.

BERT-basierte Modelle zeigen mit einem F1-Score ≈90 ausgezeichnete Ergebnisse im Bereich Relation Extraction [26].

3.3.2 Evaluierung der Relation Extraction Modelle

Da Relation Extraction als ein Klassifikationsproblem in überwachten Verfahren gilt, werden Recall, Precision and F1-Score Metriken eigesetzt, um die Performanz der Modelle auszuwerten [17]. Sie lassen sich wie folgt berechnen:

$$\text{Precision } P = \frac{\text{Anzahl der korrekt identifizierten Relationen}}{\text{Anzahl aller identifizierten Relationen}}$$

$$\text{Recall } R = \frac{\text{Anzahl der korrekt identifizierten Relationen}}{\text{tatsächliche Anzahl der Relationen}}$$

$$F1 = \frac{2PR}{P + R}$$

Die Evaluierung der unüberwachten und semi-überwachten Verfahren ist komplizierter aufgrund fehlender markierter Testdaten. Aus dem Output wird eine Stichprobe gezogen, die manuell auf Relationen geprüft wird. Zur Auswertung dieser Stichprobe werden die oben aufgeführten Metriken herangezogen. Da die tatsächliche Anzahl der Relationen in großen Datasets schwer zu ermitteln ist, lässt sich der Recall nur annähernd berechnen [17].

4 Related Works

4.1 Relation Extraction Methoden in der Bioinformatik

Da die Anzahl der Literatur im Bereich Genetik und Biomedizin schnell wächst, wird die manuelle Datenpflege zu teuer und zeitaufwändig. Gleichzeitig ist es essenziell für die Biowissenschaften, den Überblick über die Beziehungen zwischen den zu untersuchenden Entitäten zu behalten. Als Beispiel lassen sich Protein-Protein Interaktionen oder Wirkstoff-Nebenwirkungen Assoziationen anführen. Auch die Anzahl der Forschungen, die den Einfluss von Markergenen auf Krankheitswahrscheinlichkeiten einschätzen, nimmt zu.

Aus diesem Grund ist das Interesse an der Entwicklung computergestützter Ansätze für die automatische Extraktion von Beziehungen gewachsen. Named Entity Recognition und Relation Extraction spielen dabei eine große Rolle. Gleichzeitig stellen sie eine große Herausforderung dar.

Wissenschaftliche Arbeit „Optimising biomedical relationship extraction with BioBERT [27] befasst sich mit den Protein-Protein Interaktionen (PPI). Protein-Protein-Interaktionen sind fast an allen biologischen Prozessen beteiligt, deshalb werden sie im Rahmen vieler Disziplinen wie Pharmazie und synthetische Biologie geforscht [27]. Obwohl die Publikation grundsätzlich den Protein-Protein Interaktionen gewidmet ist, sind die beschriebenen Methoden auf die Extraktion anderer Beziehungen übertragbar – z.B. der Relationen zwischen Markergenen und Merkmalen.

Im Fokus dieser Publikation steht die Datengenerierung – Erhebung der themenrelevanten Sätze aus den Texten mit Hilfe von dem traditionellen regelbasierten Text Mining. Daraufhin wird ein Deep Learning Modell für Relation Extraction anhand der generierten und annotierten Daten trainiert. Anschließend vergleichen die Autoren die Ergebnisse der regel-basierten

Methoden und des Deep Learning Modells – dafür werden Precision, Recall und F1-Score gemessen.

Um Datengenerierung zu beschleunigen, den Rechenaufwand zu sparen und den Fachexperten die Arbeit abzunehmen, wurde das ursprüngliche Dataset (bestehend aus MEDLINE Artikeln) mit Hilfe von Named Entity Recognition Software TERMite gelabelt. Nur 926 Sätze mit mindestens zwei als „Protein" gelabelten Entitäten wurden extrahiert und weiterhin betrachtet.

Drei Experten mussten die Sätze in die folgenden Kategorien einteilen:

Label	Class	Example
C	Coincidental mention	A and B were measured.
P	Positive	A binds to B.
N	Negative	A does not bind to B.
I	Incorrect entity recognition	Turn to PAGE1 to read about A.
?	Don't know/unclear	

Abbildung 9 Klassen der Protein-Protein Beziehungen [27]

Nur bei 451 von 925 Sätzen (48,8%) waren alle drei Experten einig. Bei 889 von 925 Sätzen (96,1%) hatten mindestens zwei Experten die gleiche Meinung. Dieser Sachverhalt weist auf die Komplexität der Texte hin – sogar für Fachleute.

Aus den von zwei Experten gleich klassifizierten Sätzen wurden Train- und Testsets gebildet.

Bei dem **regel-basierten Ansatz** entschieden sich die Autoren nur für eine Regel, laut der ein Satz zwei Genen/Proteinen neben einem Bioverb beinhalten sollte. Dafür wurde eine Liste von 41 solchen Verben zusammengestellt.

Dieser Ansatz wurde durch die Anwendung von Dependency Parsing verbessert. Um die Genauigkeit zu erhöhen, generierten die Autoren einen Abhängigkeitsbaum für jeden Satz und prüften damit, ob ein Zusammenhang zwischen den Genen und dem Bioverb besteht. Falls es keine syntaktische Verbindung gab, fiel der Satz unter die Kategorie „keine Korrelation".

Gemischter Ansatz kombinierte ein Deep Learning Modell von BioBERT und die durch den regel-basierten Ansatz ermittelten Datensätze. Zuerst lernte das Modell aus einfachen regel-basierten Daten. Daraufhin lernte es aus den mit Dependency Parsing präzisierten Daten.

Unter dem **Deep Learning Ansatz** meinten die Autoren ein BioBERT Modell, das anhand der manuell von Experten annotierten Daten trainiert wurde.

Die Methoden wurden getestet, ihre Ergebnisse gemessen und in eine Tabelle geschrieben:

Method	Precision	Recall	F1
Ruleset 1	0.666	0.357	0.465
Ruleset 2	0.759	0.133	0.227
Ruleset 1 + BioBERT	0.716	0.812	0.761
Ruleset 2 + BioBERT	0.762	0.708	0.734
Curated data + BioBERT	0.897	0.880	0.889

Abbildung 10 Vergleich der Methodenergebnisse [27]

Der Abbildung 7 lässt sich entnehmen, dass Deep-Learning deutlich bessere Ergebnisse zeigt als der Regel-basierte Ansatz. Eine zweite Beobachtung ist es, dass durch die Anwendung von Dependency Parsing die Precision höher und der Recall geringer wird. Durch strengere Regeln wird mehr Wert auf die Genauigkeit als auf die Vollständigkeit gelegt.

Der wichtigste Erfolgsfaktor ist aber „curated data“, d.h. die manuell von Experten vorbereiteten Daten. Das deutet darauf hin, dass die menschliche Überwachung für Machine Learning Modelle immer noch notwendig ist.

Im Zuge dieser Arbeit wurde versucht, die in der Publikation beschriebenen Methoden auf die Markergen-Merkmal Relation Extraction zu übertragen und vergleichbare Ergebnisse zu bekommen.

4.2 Negation in der biomedizinischen Literatur

Das Ziel von Text Mining ist es, relevante und zuverlässige Informationen zu extrahieren. Erkennung negativer Aussagen ist damit eine essenzielle Teilaufgabe von Text Mining.

Negation kommt in biomedizinischer Literatur häufig vor und führt zu geringerer Präzision bei automatisierten Text-Mining Systemen. Negative Aussagen werden oft übersehen, was zahlreiche False-Positives verursacht [28].

Der Satz in der Abbildung 8 zeigt, warum es wichtig ist, die Negation in Texten zu berücksichtigen. Es besteht keine Assoziation zwischen dem Snip rs429358 und dem CPP-Phänotyp;

wenn die Negation jedoch vernachlässigt wird, kann eine falsche Assoziation festgestellt werden.

Abbildung 11 Beispiel einer Negation [29]

Es gibt viele Forschungen im Bereich Bioinformatik, die die Negationserkennung untersuchen. Einige Methoden basieren auf einfachen RegEx Algorithmen [30], die anderen sind komplexer und verwenden Text Mining Regeln und Dependency Parsing.

Der in der Publikation "Inferring the Scope of Negation in Biomedical Documents [31] beschriebene Ansatz besteht aus drei Stufen. Zuerst definieren die Autoren eine Liste von Negationswörtern, daraufhin finden sie anhand von Abhängigkeitsbäumen die mit den Negationswörtern verbundenen Knoten. Anschließend wird ein so genannter „negation scope" identifiziert. Unter diesem Begriff versteht man die Wörter im Satz, die durch das Negationswort negiert werden.

In dem praktischen Teil dieser Arbeit wurden zwei Negationserkennungsmethoden angewandt – RegEx und Dependency Parsing mit Negation Scopes.

4.3 Überblick anderer verwandten Arbeiten

In den oberen Abschnitten wurden zwei für diese Bachelorarbeit wichtige Publikationen erwähnt aber die vollständige Liste der Literatur ist natürlich viel größer.

„GENETAG: a tagged corpus for gene/protein named entity recognition" [32] beschreibt eine NLP Pipeline im Kontext der biomedizinischen Texte. Neben Datenbereinigung und Tokenization gehen die Autoren besonders detailliert auf die Entitätenerkennung und auf das sie begleitende Mehrdeutigkeitsproblem ein. Die Gene haben neben einer einheitlichen Nomenklatur auch andere Namen, die oft unterschiedliche kontextabhängige Bedeutungen haben. Die Autoren lösen dieses Problem anhand semantischer Bedingungen und Listen von alternativen Namen.

Auch eine 15 Jahre spätere Arbeit befasst sich mit einer ähnlichen Fragestellung. Publikation "Named Entity Recognition and Relation Detection for Biomedical Information Extraction" [33] spricht das Mehrdeutigkeitsproblem an – Synonymen, Homonymen und mehrdeutige Abkürzungen lassen sich mit verschiedenen Methoden wie z.B. „name normalization" [34] und „noun head resolving" [35] behandeln.

In [33] und [29] befassen sich die Autoren mit der Modalität und den Negationen in Relation Extraction. Modalität gibt eine Aussage über die Stärke der Assoziation – ob es sich um eine Tatsache oder eine Vermutung handelt.

[33], [36] und [37] geben einen Überblick über die ML-basierten NER Methoden, wobei die Long Short-Term Memory (LSTM) neuronale Netze besonders ausgezeichnet werden. LSTM verfügen über Langzeitgedächtnis und können langfristige Abhängigkeiten lernen, was sie für Sprachmodelle geeignet macht. Außerdem lassen LSTM steuern, welche Informationen in Memory verbleiben bzw. vergessen werden.

Noch effizienter als LSTM haben sich Transformer Modelle erwiesen, die 2017 in der Publikation „Attention Is All You Need" [25] zum ersten Mal vorgestellt wurden. Transformers verwenden einen Attention Mechanismus, der jedem Input ein Gewicht zuweist, um die Wichtigkeit jedes Tokens für die anderen Tokens zu vermitteln.

Oben genannte Publikationen haben unterschiedliche Schwerpunkte und untersuchen verschiedene Methoden aber sie alle befassen sich mit der automatischen Extraktion der wichtigen Erkenntnisse aus biomedizinischen Texten – sowie auch diese Arbeit.

5 Anforderungen und Architektur

5.1 Anforderungen

Bevor die Phase der Implementierung startet, sollen zuerst die Anforderungen ermittelt werden. Es gibt drei typische Anforderungsquellen: Dokumente, Stakeholders und andere Systeme [38]. Nach der Analyse der Publikationen aus dem Abschnitt 4 „Related Works“ sind folgende funktionale Anforderungen entstanden:

1.1 Das System soll eine Verbindung zu der PubMed-Datenbank haben.
1.2 Das System kann die PubMed Abstracts in JSON- und Textformat importieren.
1.3 Das System kann eine Auflistung der relevanten menschlichen Einzelnukleotid-Polymorphismen dem User zurückgeben.
1.4 Das System soll in der Lage sein, wissenschaftliche Publikationen zu jedem Einzelnukleotid-Polymorphismus aus Punkt 1.3 zu liefern.
1.5 Das System kann die Einzelnukleotid-Polymorphismen nach ihrer Häufigkeit sortieren und die am besten erforschten Einzelnukleotid-Polymorphismen zurückliefern.
1.6 Das System soll die Relationen zwischen Einzelnukleotid-Polymorphismen und wahrscheinlichen Krankheiten in den Dokumenten finden.
1.7 Das System soll die Untersuchung unterschiedlicher Algorithmen für Markergenforschung unterstützen.
1.8 Das System soll die Relationen in zwei Gruppen klassifizieren: (1) positive Relationen oder (2) negative Relationen.
1.9 Das System soll die gefundenen Relationen und ihre Klasse in die Datenbank eintragen können.

Nicht-funktionale Anforderungen des Systems sind aus dem Qualitätsstandard für Software DIN 66272 entnommen [39]:

2.1 Funktionalität – alle funktionalen Anforderungen sind erfüllt.
2.2 Benutzbarkeit – das System soll dokumentiert und mit Kommentaren versehen sein, um das Verständnis für Entwickler zu erleichtern.
2.3 Änderbarkeit – das System soll horizontal erweiterbar sein, um künftig große Datenmengen verarbeiten zu können.
2.4 Zuverlässigkeit – die Korrektheit der Ergebnisse soll geprüft und bewertet werden.
2.5 Übertragbarkeit – das System muss nicht auf einer anderen Hardware und Software einsetzbar sein.
2.6 Konfigurierbarkeit – der User darf die Anzahl der SNPs, die Länge der Windows zwischen Entitäten und die Machine Learning Hyperparameter definieren sowie ein Transformer Modell auswählen.

Punkt 2.6 gehört nicht zu den Hauptqualitätsmerkmalen laut DIN 66272 aber kann als ein Aspekt der „Änderbarkeit“ angesehen werden. Diese Anforderung ist wünschenswert für ein experimentelles System, welches von den Benutzern für ihre Zwecke angepasst werden soll.

5.2 Architektur

In diesem Abschnitt wird das System aus zwei Architektursichten betrachtet – der Bausteinsicht und der Verteilungssicht. Die Verteilungssicht (Abbildung 12) beschreibt die Hardwarekomponenten, wobei die Bausteinsicht (Abbildung 13) das System auf die Softwarekomponenten zerlegt. Kapitel 6 „Design und Implementierung“ beschreibt ausführlich jede Komponente.

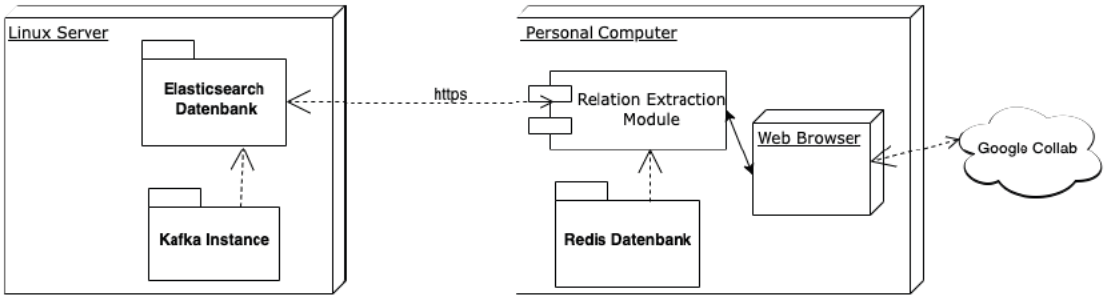


Abbildung 13 Verteilungssicht

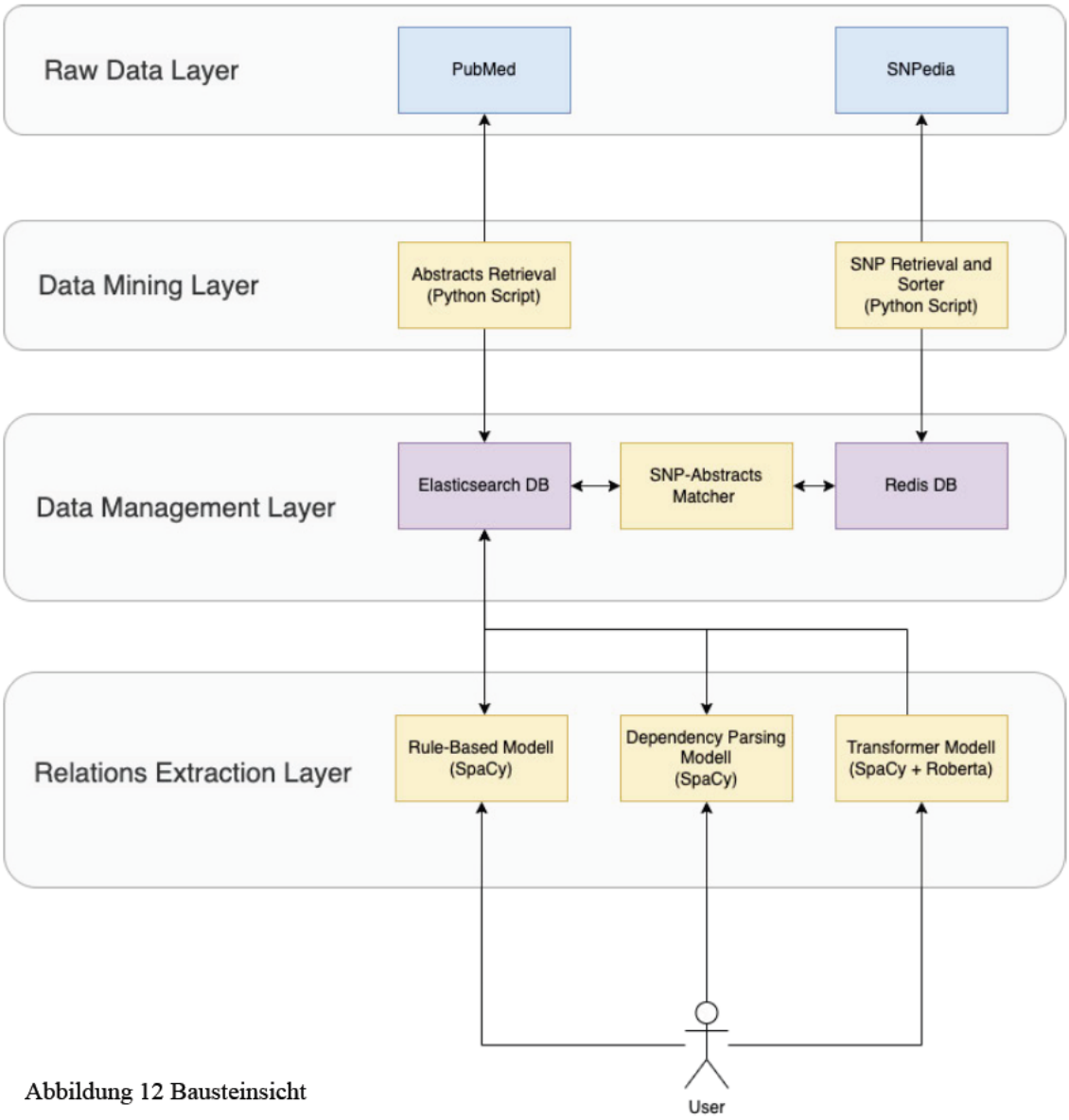


Abbildung 12 Bausteinsicht

6 Design und Implementierung

6.1 Datenaquisition

6.1.1 Datenquellen

Im Fokus dieser Bachelorarbeit liegen Einzelnukleotid-Polymorphismen und ihre potenzielle Auswirkung auf die menschliche Gesundheit. Da vollständige vertrauenswürdige Daten essenziell für eine wissenschaftliche Arbeit sind, wurde als Erstes eine Datenquelleanalyse im Bereich Biomedizin durchgeführt.

The National Center for Biotechnology Information (NCBI) ist ein zentrales amerikanisches Institut für Informationssysteme in der Molekularbiologie [40]. NCBI Webseite stellt einen Zugang zu der dbSNP-Datenbank. Diese Datenbank enthält über 400 Millionen nicht redundante Snips [40]. Jedem Snip wird eine Kennung zugewiesen, die mit dem zweibuchstabigen Code “rs“ beginnt und mit einer eindeutigen Nummer endet. Dieser Identifikator ist bei Mikroarrays und in der wissenschaftlichen Literatur weit verbreitet. Durch die Popularität der Genteste haben auch die Nicht-Wissenschaftler die rs-Identifikatoren kennengelernt.

DbSNP ist eine vollständige und aktuelle SNP-Datenbank aber die Verarbeitung und Analyse von 400 Millionen Polymorphismen ist im Rahmen einer Bachelorthesis nicht möglich. Außerdem sind in der Datenbank nicht nur menschliche Snips enthalten, sondern auch Snips von anderen Lebewesen, die kein Interesse für die gegebene Aufgabenstellung darstellen.

SNPedia ist eine seit 2006 existierende Wiki-basierte Website, die als Datenbank für Einzelnukleotid-Polymorphismen dient. SNPedia fasst medizinische, phänotypische, forensische und

genealogische Informationen zu verschiedenen Snips zusammen [41]. Für die Erkennung der Einzelnukleotid-Polymorphismen werden die rs-Identifikatoren von dbSNP verwendet.

SNPedia beschreibt über 100000 Polymorphismen. Im Vergleich zu dbSNP werden dort nur diejenige Snips erfasst, die eine historische, statistische oder – basierend auf einer veröffentlichten Studie mit mindestens 500 Patienten oder auf mindestens zwei unabhängigen Studien – medizinische Bedeutung haben [41].

Nicht alle Polymorphismen sind gleich gut beschrieben. Nur einige Einträge enthalten eine ausführliche Definition, Links zu wissenschaftlichen Publikationen sowie Microarray-Informationen über den entsprechenden Snip. Da die Artikel manuell geschrieben werden, sind diese Zusammenfassungen möglicherweise nicht vollständig. Aus diesem Grund darf SNPedia nicht als Single Point of Truth betrachtet werden aber kann einen Überblick über die potenziell interessanten Polymorphismen geben.

Außerdem verfügt SNPedia über eine benutzerfreundliche API-Schnitstelle. Im Rahmen dieser Arbeit wurde diese Datenbank als eine Datenquelle für die Extraktion der Snips gewählt.

PubMed Central (PMC) ist eine von NBCI entwickelte, frei zugängliche Meta-Datenbank, die Referenzen auf biomedizinische wissenschaftliche Artikel enthält. PubMed bietet einen Zugang zu den Datenbanken MEDLINE und PubMed Central mit mehr als 27 Millionen Artikeln [42].

Während eine PubMed-Abfrage nur den Abstract und den Link auf den Volltext zurückgibt, bitetet PubMed Central die Volltexte. Die erste Entscheingung war, mit den Volltexten zu arbeiten, weil sie einen tieferen Einblick in die Zusammenhänge zwischen Snips und Merkmalen gewähren könnten. Jedoch hat sich diese Idee als ineffizient erwiesen. Falls es Zusammenhänge zwischen einem Markergen und einer Krankheit gibt, werden sie in der Regel schon im Abstract erwähnt. Die Verarbeitung der vollen Artikeltexte würde einen vielfachen Aufwand erfordern ohne deutliche Verbesserung der Genauigkeit.

Tabelle 2 Zusammenfassende Tabelle der Datenquellen

Datenquelle	Daten

dbSNP-Datenbank	über 400 Millionen Snips von Menschen und anderen Organismen; ihre Lage im Genom
SNPedia	über 100000 relevante Snips; Beschreibung, Lage im Genom, Links zu wissenschaftlichen Publikationen
PubMed	Abstracts der biomedizinischen Artikel

6.1.2 Datenextraktion

SNPedia nutzt MediaWiki – eine frei verfügbare Verwaltungssoftware für das Erarbeiten, Organisieren und Publizieren von Wissen in Form eines Wiki-Systems. MediaWiki bietet eine API, die unter anderem die Suche und das Parsen der Wiki-Seiten ermöglicht. Dabei gibt es keine feste Beschränkung für Leseanfragen.

Als Python Client für MediaWiki API wird von SNPedia mwclient empfohlen. Mit Hilfe von mwclient wurden alle 111397 Snip-Identifikatoren geparst.

Zu der Mehrheit dieser Markergene findet PubMed aktuell keine Forschungen. Um ein genug großes Dataset zu bilden, welches in Trainings-, Tests- und Validierungssets aufgeteilt werden kann, mussten bereits gut erforschte Snips identifiziert werden.

Entrez Gene ist eine von NCBI betriebene Metasuchmaschine, die den gleichzeitigen Zugriff auf mehrere Biochemie- und Medizindatenbanken ermöglicht. Unter anderem gehört auch PubMed zu den vernetzten Datenbanken. Entrez fasst alle vernetzten Datenbanken unter einer grafischer Benutzeroberfläche zusammen. Zusätzlich bietet NCBI eine API, die Entrez Programming Utilities (eUtils) heißt. Auf die eUtils wird zugegriffen, indem die Abfragen in Form einer speziellen URL an den NCBI-Server gesendet und die XML-Antwort analysiert werden.

Die Biopython-Website (http://www.biopython.org) bietet eine Online-Ressource für Entwickler/-innen von Python-basierter Software. Grundsätzliches Ziel von Biopython ist es, Python für die Bioinformatik so einfach wie möglich zu gestalten [44]. Um auf die Entrez API zuzugreifen, wurde Biopython Package „Entrez“ verwendet.

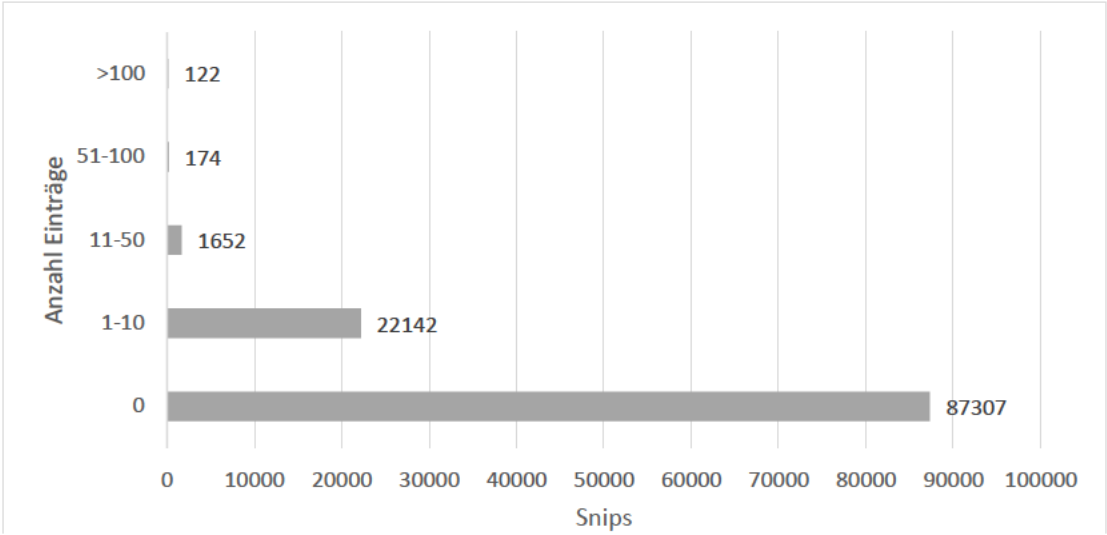


Abbildung 14 Anzahl Einträge zu jedem Markergen

Um die am besten erforschten Snips zu ermitteln, wurde die Anzahl der Einträge zu jedem der 111397 rs-Identifikatoren abgefragt. Aus der Abbildung 6 ist zu entnehmen, dass es zu 87307 von 111397 Snips keine Artikel in PubMed gibt. Nur 122 Polymorphismen figurieren in jeweils 100 und mehr Artikeln.

Im Rahmen dieser Bachelorarbeit werden nur die Top 50 Snips betrachtet. Eine volle Auflistung dieser Markergene befindet sich im Appendix A.

Zu 50 rs-Identifikatoren wurden 5970 Abstract-Texte gefunden und extrahiert.

6.1.3 Datenhaltung

Die Daten werden nur einmal extrahiert und anschließend gespeichert, um von den externen Datenquellen und APIs nicht anhängig zu sein.

Die Auswahl der Datenbanken erfolgte nach folgenden Kriterien:

- Performanz
- Niedrige Latenz
- Passendes Datenmodell
- Open-Source Software

Die rs-Identifikatoren wurden in die Redis Datenbank aufgenommen. Redis ist einer der verbreitetsten Schlüssel-Werte-Speicher. Da in diesem Fall nur die Namen gespeichert werden mussten, war das einfache Key-Value Datenmodell vorteilhaft.

Redis ist eine In-Memory Datenbank. Dadurch ist sie wesentlich performanter als die auf Festplatten zugreifenden Datenbankmanagementsystemen. Da Redis den Arbeitsspeicher eines Rechners nutzt, ist die Speicherkapazität der Datenbank begrenzt [45]. Für 111397 Key-Value-Paare reicht sie aber aus.

Als Python Client für Redis wurde redis-py Bibliothek verwendet und als GUI Client YetAnotherRedisClient eingesetzt.

Auch 5970 Abstract Texte mussten gespeichert werden, dafür wurden folgende Optionen betrachtet:

- **Apache Kafka** ist eine Open Source Software, die die Speicherung und Verarbeitung von Datenströmen ermöglicht. Das System basiert auf einer verteilten skalierbaren Streaming-Architektur. Dadurch eignet sich Kafka für große Datenmengen und Big Data-Anwendungen [46].
  Kafka besitzt die ACID-Eigenschaften, ist höchst performant sowohl für Producing als auch für Consuming und unterstützt alle Datenformate. Ein wesentlicher Nachteil dieser Option besteht darin, dass Kafka kein Datenbankmanagementsystem ist und normalerweise für andere Use Cases (Messaging, Stream Processing) verwendet wird.
- **MongoDB** ist ein dokumentenorientiertes Open-Source NoSQL-Datenbankmanagementsystem. MongoDB ist die meistverwendete dokumentenorientierte Datenbank laut DB-Engines Ranking (Stand: 14.11.2021). Sie unterstützt CRUD Operationen aber bietet eine langsame und ineffiziente Volltextsuche [47].

- **Elasticsearch** ist auch eine dokumentenorientierte NoSQL-Datenbank und gleichzeitig eine moderne Such- und Analyseplattform auf Basis von Apache Lucene. Lucene ist eine Volltextsuche-Bibliothek in Java, die alle Dokumente in Tokens aufteilt und daraufhin die Tokens indiziert.
  Im Vergleich zu den andren NoSQL-Datenbanken sind Write-Operationen in Elasticsearch langsam, außerdem unterstützt Elasticsearch keine ACID-Transaktionen – fehlende Atomarität kann zum Datenverlust führen. Zu den Vorteilen von Elasticsearch gehören Tokenization und schnelle Volltextsuche [48].

Als Speicher wurde Elasticsearch gewählt. Elasticsearch entspricht dem Anwendungsfall dieser Arbeit, der sowohl Tokenization als auch die Volltextsuche umfasst. Die Daten werden nur einmal geladen und kaum geändert, deswegen sind langsame Write-Operationen kein gravierender Nachteil.

Der Elasticsearch Cluster besteht aus 2 Knoten: auf einem liegt ein Primary- und auf dem anderen ein Replica-Shard. Der Replica-Shard ist eine redundante Kopie der Daten im Primary-Shard, die vor Hardwareausfällen schützt und die Geschwindigkeit der Volltextsuche erhöht.

Für die Suche und Visualisierung der Daten wird Kibana verwendet.

6.1.4 Datenfilterung

Bei näherer Betrachtung der Dokumente stellte sich heraus, dass Einzelnukleotid-Polymorphismen nicht nur aus medizinischer, sondern auch aus geografischer, historischer und statistischer Perspektive betrachtet werden. Irrelevante Texte, die keine Relationen zwischen Einzelnukleotid-Polymorphismen und Krankheiten beinhalteten, sollten das Dataset verlassen.

Anhand der Entitätserkennung wurden die Texte ohne Sätze mit mindestens einer SNP-Entität und einer Krankheit-Entität gefunden. Nach der Entfernung der irrelevanten Dokumente sind 2507 Einträge verblieben. Für das Baseline Experiment wurden zufällige 200 Abstracts ausgewählt, die 50 populären Polymorphismen beschreiben.

6.2 Anwendung der Pre-Processing NLP-Pipeline

SpaCy v3 Bibliothek bietet bequem konfigurierbare NLP-Pipelines für verschiedene Aufgaben. Drei in dieser Arbeit betrachtete Modelle benötigten unterschiedliche Pipelines:

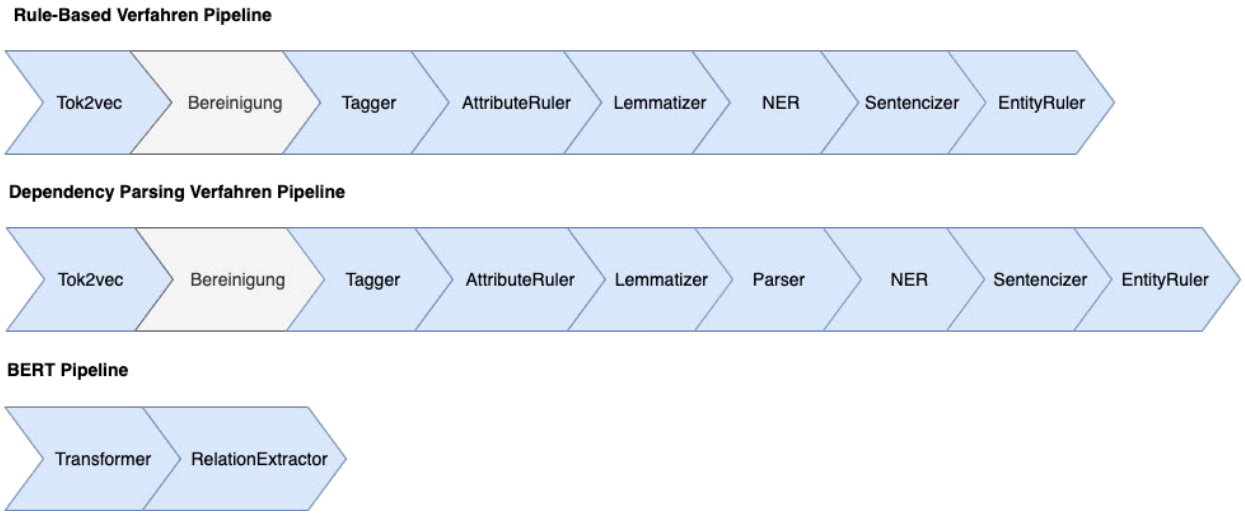


Abbildung 15 NLP Pipelines für drei Verfahren

Blau markierte Pfeile sind die von SpaCy angebotenen Pipeline Komponenten. Datenbereinigung wurde ohne Einsatz von SpaCy durchgeführt.

Tok2vec ist eine bereits vortrainierte Machine Learning Komponente, die lernt, dynamische Vektoren für Tokens zu erzeugen. Tok2vec wird üblicherweise nicht selbstständig verwendet, sondern ist ein Teil anderer SpaCy-Komponenten wie DependencyParser, POS-Tagger oder NER.

Während die ersten zwei Pipelines mit tok2vec starten, basiert die dritte Pipeline auf **Transformer**-Architektur. Diese Pipeline ist kürzer als die anderen und modifiziert die Daten nicht – für ein Transformer Modell ist der originale Kontext wichtig. Außerdem gibt es keine NER-Komponente, weil die Datasets im Voraus gelabelt werden.

Die **Datenbereinigung** sieht für das Rule-Based Verfahren und das Dependency Parsing Verfahren unterschiedlich aus. Im ersten Fall werden alle Satzzeichen außer Punkt gelöscht. Im

Gegensatz dazu, bleiben die Satzzeichen im zweiten Experiment bestehen, denn die Syntax und folglich die Interpunktion spielen eine große Rolle für Dependency Parsing.

Entfernung der Stoppwörter ist der nächste Schritt der Datenbereinigung. Im ersten Experiment legen die Regeln maximale Fenster zwischen, vor und nach den Entitäten fest. Nach der Entfernung irrelevanter, oft vorkommender Wörter erhöht sich die Anzahl der anhand Regeln entdeckten Relationen. Zu dieser Liste werden themenspezifische Begriffe wie „SNP“ und „Single nucleotid polymorphism“ hinzugefügt. Sie kommen häufig vor aber sagen nicht viel aus, weil die Regeln auf konkrete rs-Identifikatoren abgestimmt sind.

Präpositionen und Konjunktionen gehören aufgrund ihrer großen Verbreitung zu den Stoppwörtern. Da sie aber syntaktische Beziehungen zwischen den Satzgliedern im Satz ausdrücken, was für Dependency Parsing wichtig ist, werden solche Wörter aus der Liste der Stoppwörter im zweiten Modell entfernt.

In Spacy v3 laufen einige Komponenten nur in der Kopplung mit den anderen. Ohne **POS-Tagger** wird der Lemmatizer nicht funktionieren, da es wichtig ist, die Wortart zu bestimmen, um die richtige Grundform zu bilden. Auch ein **AttributeRuler** soll mit dem POS-Tagger in Verbindung stehen. Ein AttributeRuler behandelt Ausnahmen, indem er erlaubt, neue Regeln zu definieren und Attribute für Tokens zu setzen. Mithilfe von AttributeRuler wurde definiert, dass rs-Identifikatoren als Substantiven vom POS-Tagger markiert werden sollen.

Lemmatizer ist wichtig für Rule-Based Modelle, die auf bestimmten Wortlisten basieren. Die Wörter in diesen Listen werden in ihrer Grundform aufgelistet, deswegen erhöht Lemmatisierung die Vollständigkeit der regel-basierten Algorithmen. Das passiert aber nur in dem Fall, wenn die Lemmatisierung richtig durchgeführt wurde. Obwohl die SpaCy Komponente allgemein zuverlässige Ergebnisse liefert, funktioniert sie bei selten vorkommenden Wörtern wie Krankheitsnamen etwas schlechter (z.B „diabetes“ wird irrtümlich auf „diabete“ reduziert – wegen des Wortendes -s, welches oft als Marker der Pluralform dient). Da die falsch modifizierten Krankheitsnamen von den Regeln nicht mehr erkannt werden, werden solche Wörter aus dem Lemmatisierung-Prozess ausgeschlossen.

Dependency Parser nimmt nur in der zweiten Pipeline teil. Die anderen Verfahren berücksichtigen grammatische Beziehungen zwischen den Wörtern nicht, deswegen ist diese Komponente für sie irrelevant. Dependency Parser stellt einen Satz als einen Abhängigkeitsbaum dar. Dabei ist jeder Token ein Knoten des Baums und kann syntaktische „Eltern“ bzw. „Kinder“ haben.

Relation Extraction ist nur dann möglich, wenn ihr Named Entity Recognition voransteht. Da es in dieser Arbeit um die Relationen zwischen SNPs und Krankheiten geht, sollte die **NER** Komponente diese zwei Entitätstypen erkennen können.

en_ner_bc5cdr_md ist ein von scispaCy Bibliothek bereitgestelltes SpaCy NER Model, das mit BC5CDR Korpus trainiert wurde. BC5CDR Korpus besteht aus 1500 PubMed Artikeln mit 4409 annotierten Chemikalien, 5818 Krankheiten und 3116 Interaktionen zwischen Chemikalien und Krankheiten [49]. Dieses vortrainierte Model wird in den ersten zwei Modellen verwendet und erleichtert die Extraktion der Krankheit-Entitäten.

EntityRuler wird zusammen mit NER eingesetzt, um die Genauigkeit der Entitätenerkennung zu steigern. EntityRuler unterstützt RegEx. Rs-Identifikatoren sind gleichartig und lassen sich mit dem regulären Ausdruck "^rs\\d+" beschreiben. EntityRuler bekommt die Regel „Token ^rs\\d+ ist eine SNP-Entität" und erleichtert somit den NER-Prozess.

Abschließend unterteilt ein **Sentencizer** die Dokumente in die Sätze. Jeder Satz wird einzeln nach Relationen untersucht.

6.3 Vergleich der Relation Extraction Verfahren

6.3.1 Rule-Based

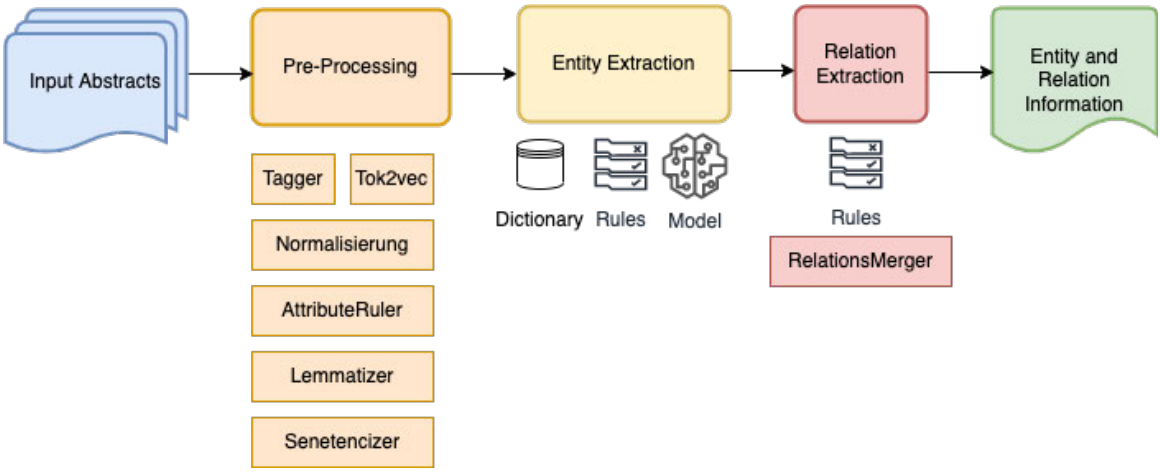


Abbildung 16 Workflow des regelbasierten Modells

Der erste Algorithmus basiert auf manuell erstellten domainspezifischen pattern-matching Regeln.

3 Objekte müssen in einer positiven Regel vorkommen:

- Eine SNP-Entität
- Eine Krankheit-Entität
- Ein oder mehrere Tokens, die diese Entitäten verbinden

Im Abschnitt 6.2 „Anwendung der NLP-Pipeline“ wurde die Erkennung der Entitätstypen beschrieben. Nachdem zwei Objekte erkannt sind, muss festgestellt werden, ob sie in einer Beziehung stehen und – falls ja – in welcher. Dafür wurde eine Nachschlageliste mit den verbindenden Wörtern erstellt.

Diese Liste enthält sowohl universelle Wörter wie „*association*“ und „*connection*“ als auch gebietsspezifische Begriffe wie „*susceptibility locus*“ (ein Chromosomenabschnitt, der für die Krankheitsdisposition zuständig ist [50]). Die Auflistung von 40 bindenden Wörtern befindet sich im Appendix B.

Als nächstes sollen die maximalen Abstände zwischen den Entitäten definiert werden, denn sie müssen nicht unbedingt direkt aufeinander folgen. Andererseits darf der Abstand nicht zu groß sein: weit voneinander entfernte Wörter sind mit höher Wahrscheinlichkeit nicht semantisch verbunden.

Da die Anzahl der Tokens nach der Normalisierung deutlich kleiner wird, reicht ein Window von 5 Tokens zwischen den Objekten in den meisten Fällen aus. Dies wurde experimentell ermittelt.

Die Anzahl der möglichen Anordnungen von 3 Objekten wird durch die Fakultät angegeben: 3! = 6 Permutationen ohne Wiederholdung bzw. 6 positive Regeln sind möglich. Sie sehen wie folgt aus:

1. SNP - 5 Tokens Window - Verbindung - 5 Tokens Window - Krankheit
2. Krankheit - 5 Tokens Window - Verbindung - 5 Tokens Window - SNP
3. Krankheit - 5 Tokens Window - SNP - 5 Tokens Window - Verbindung
4. SNP - 5 Tokens Window - Krankheit - 5 Tokens Window - Verbindung
5. Verbindung - 5 Tokens Window - Krankheit - 5 Tokens Window - SNP
6. Verbindung - 5 Tokens Window - SNP - 5 Tokens Window - Krankheit

Um eine negative Regel zu bilden, ist eine Verneinung notwendig. Tokens *no*, *not*, *non*, *n't*, *neither* und *nor* sind offenbare Zeichen einer Negation. Auch die semantischen Negationswörter sollen mitbetrachtet werden. Als Beispiel lässt sich folgender Satz anführen:

„Researchers failed to detect an association between the SNP rs17300539 and metabolic syndrome. "

Aus der syntaktischen Sicht ist es kein verneinender Satz. Überdies ist er mit der positiven Regel #6 konform. Dennoch bedeutet dieser Satz, dass es keine Relation zwischen rs17300539 und dem metabolischen Syndrom besteht. Aus diesem Grund wurde das Wort „fail" sowie einige andere Verben und Substantiven in die Liste der Negationswörter eingetragen.

Eine negierende Regel besteht also aus vier Objekten – einem Snip, einer Krankheit, einer Verbindung und einer Negation. Daraus ergeben sich 24 negative Regeln – so hoch ist die

Anzahl der möglichen Permutationen von vier Objekten. Fraglich ist, ob tatsächlich 24 Regeln notwendig sind, um alle Fälle abzudecken.

Der einfachste Weg, eine Assoziation zu verneinen, ist es, das verbindende Wort zu negieren, also eine Negationswort vor diesem Wort zu stellen. Das sind die ersten sechs Regeln. Eine Negation am Ende des Satzes kommt auch vor – das macht noch sechs Anordnungen aus. Eine Negation vor einem Snip tritt eher selten auf aber sollte auch berücksichtigt werden. In den letzten sechs Permutationen steht ein verneinendes Wort vor einer Krankheit. Folgendes Beispiel ist möglich:

„This study reveals a connection between rsXXX and the absence of Y disease. “

“Absence”, d.h. “Abwesenheit” ist in die Liste der verneinenden Wörter eingetragen. In diesem Satz wird aber nicht die ganze Assoziation, sondern nur ein Satzteil negiert. rsXXX übt in diesem Fall eine Schutzfunktion gegen Krankheit Y aus, d.h. es besteht eine *positive* Relation zwischen diesen zwei Entitäten.

Aus diesem Grund werden diese sechs Regeln nicht angenommen. Eine volle Auflistung der Regeln für den ersten regelbasierten Algorithmus befindet sich im Appendix C.

Der von spaCy angebotene RuleMatcher Algorithmus gibt einen Span zurück – einen Satzteil, der einer Regel entspricht. Eine Regel stellt eine 1:1 Relation dar, wobei viele Sätze eine oder mehrere 1:n Relationen beinhalten. Manchmal werden die Entitäten nebeneinander aufgezählt. Zum Beispiel für den Satz:

„Polymorphisms rs1800497 rs757110 rs1136287 association with obesity.”

gibt der RuleMatcher drei Spannen zurück:

*“**rs1136287** association with **obesity**.”*

„***rs757110*** *rs1136287 association with* ***obesity***.“

“***rs1800497*** *rs757110 rs1136287 association with* ***obesity***.”

Relation „rs757110 – obesity“ kommt dabei zweimal und ”rs1136287 – obesity” sogar dreimal vor, obwohl sie nur einmal im Text erwähnt werden. Dies geschieht, weil sich diese SNP-

Entitäten innerhalb eines 5-Tokens-Windows zwischen einer SNP-Entität und einer Krankheit-Entität befinden.

Um solche Missverständnisse zu vermeiden, wurde eine RelationsMegrer-Komponente geschrieben, die sich überlappende Satzglieder zu einem großen Satzteil zusammenfasst. Aus diesem Satzteil werden die Relationen nur einmal extrahiert.

6.3.2 Dependency Parsing Based

Das erste Modell basiert auf mehr als 20 strengen Regeln. Für den zweiten Algorithmus ist nur eine Regel notwendig: Sätze ohne mindestens einer SNP-Entität, einer Krankheit-Entität und einer Verbindung zwischen diesen Entitäten werden aussortiert. Der anschließende Prozessablauf ist auf der Abbildung 16 dargestellt:

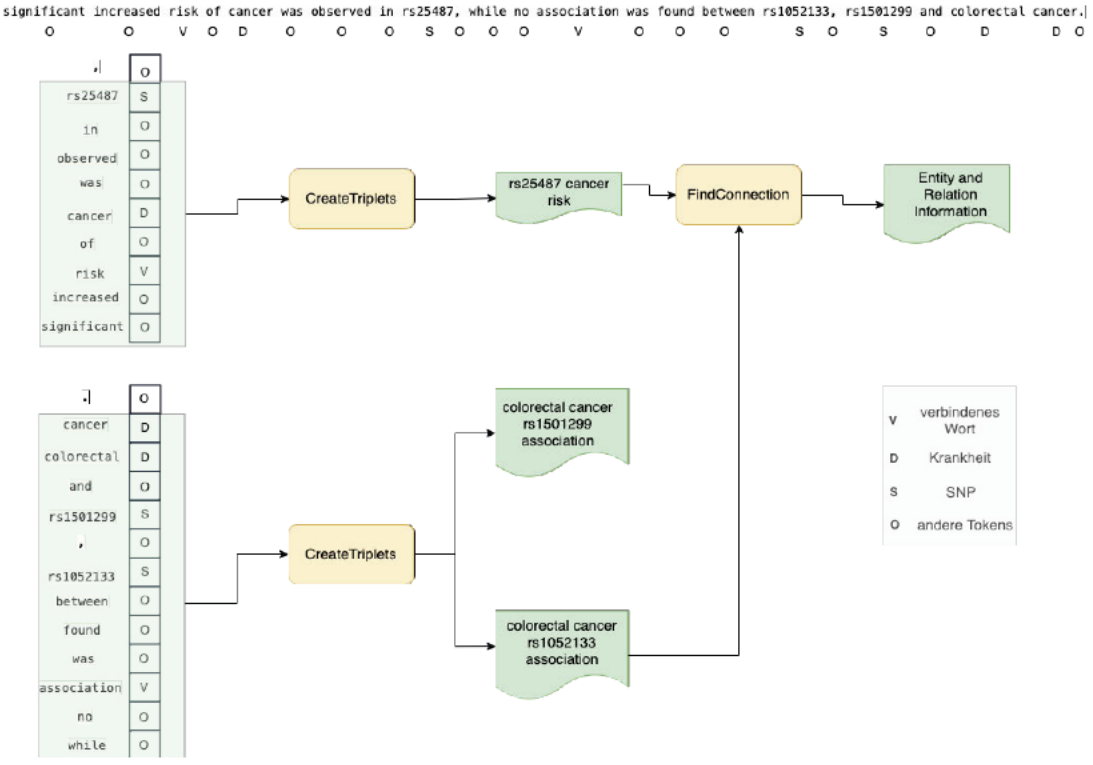


Abbildung 17 Prozessablauf des zweiten Modells

Nachdem nur die relevanten Sätze geblieben sind, muss festgestellt werden, inwieweit die Entitäten in den relevanten Sätzen syntaktisch verbunden sind. Jedes Triplet wird einzeln betrachtet. Zunächst müssen die Triplets extrahiert werden.

Ein Satz wird von Anfang bis Ende durchgegangen. Die Krankheit-, SNP- und Verbindung-Tokens werden auf einen Stack gelegt, bis die Tokens von allen drei Typen auf dem Stapel liegen. Falls sich Objekte von drei Typen auf dem Stapel befinden und das nächste daraufzulegende Token nicht zu einem dieser Typen gehört, werden die Tokens vom Stack heruntergenommen. Sie werden dann als Argument in die Funktion übergeben, die die Triplets erzeugt. Abbildung 17 zeigt den Pseudocode für diesen Algorithmus:

Algorithm 1 Find Relation Scope

```
1:  for Tokens in sentence do
2:      if token is SNP or Verbindung then
3:          Add token to stack
4:      end if
5:      if Token is Krankheit then
6:          if Token is inside a chunk then
7:              Remove previous token
8:          end if
9:          Add token to stack
10:     end if
11:     if Token is O and SNP, Verbindung, Krankheit are in stack then
12:         CreateTriplets(stack)
13:         Clear the stack
14:     end if
15: end for
```

Abbildung 18 Relation Scope Algorithmus

Zeilen 6 und 7 benötigen eine Erklärung: die Wortgruppen stellen eine Herausforderung für den Algorithmus dar. Medizinische Begriffe bestehen oft aus zwei oder mehr Wörtern – in der Regel handelt es sich um ein Substantiv und ein oder mehrere Adjektive, z.B. *colorectal* ***cancer*** oder *inflammatory bowel* ***disease***. Die NER-Komponente weist jedem Wort ein Disease-Tag

zu, was die Anzahl der möglichen Triplets deutlich erhöht. Syntaktisch gesehen reicht es, nur das Substantiv zu betrachten, weil die anderen Wörter ihm untergeordnet sind.

SpaCy erkennt solche Wortgruppen („Chunks“) und weist jedem Token neben eines Entität-Tags ein IOB-Tag zu:

I (Inside) – Token ist ein Teil der Entität

O (Outside) – Token gehört keiner Entität

B (Beginning) – Token steht am Anfang der Entität

Der Algorithmus prüft die Markierung jeder Krankheit-Entität. Falls eine Entität-Wortgruppe vorkommt, wird nur das letzte Wort der Entität auf den Stack gelegt. Dieses Wort ist normalerweise auch der Kern dieser Wortgruppe und repräsentiert die ganze Entität bei der Abhängigkeitssuche.

Jedes Triplet wird nach Abhängigkeiten im Satz untersucht. Ein Dependency Baum sieht wie folgt aus:

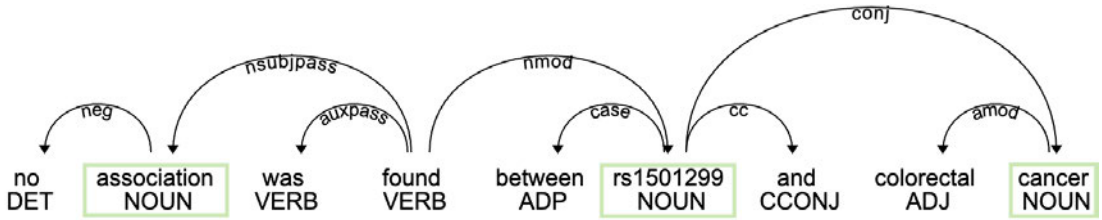


Abbildung 19 Beispiel eines Abhängigkeitsbaums

Die Wurzel eines Abhängigkeitsbaums ist immer das Hauptverb des Satzes. Die Satzglieder sind als Knoten dargestellt und die Kanten bedeuten die syntaktischen Beziehungen zwischen den Satzgliedern. In jeder Beziehung gibt es ein Elternteil („head“) und ein Kind („child“). Ein Knoten kann mehrere untergeordnete Kinder haben, dafür aber nur ein Elternteil.

Zuerst wird geprüft, ob eine negative Relation zwischen den Tokens im Triplet existiert. Falls es im Satz eine negierende Kante „neg“ gibt, wird der verweisende Knoten gefunden und ein

so genannter „Negation Scope“ definiert. Falls die Tokens aus dem Triplet in diesem Bereich vorkommen, gilt die Relation als negativ.

Algorithm 2 Find Negation Scope and Negative Relation

```
for Tokens in sentence do
    if Token's relation is "neg" then
        Find parent of this token
        Add parent to the negation scope
        Find all children of the parent
        Add all children of the parent to the negation scope
    end if
end for
if SNP or Verbindung or Krankheit in the negation scope then return
"Negative Relation: SNP-Krankheit"
```

Abbildung 20 Negation Scope Algorithmus

Z.B. der in der Abbildung 18 aufgeführte Satz hat einen Negation Scope mit zwei Elementen: [„no“, „association“]. „Association“ ist ein verbindendes Wort aus dem Triplet, deswegen wird eine negative Relation „*rs1501299 – colorectal cancer*“ erkannt.

Falls eine negative Relation nicht erkannt wird, wird das Triplet auf eine positive Relation geprüft.

Eine positive Relation gilt in den folgenden Fällen:

- Wenn es einen Pfad zwischen dem SNP-Knoten und Krankheit-Knoten gibt.
- Wenn es einen Pfad zwischen SNP-Knoten und Verbindung-Knoten und einen Pfad zwischen Krankheit-Knoten und Verbindung-Knoten gibt (siehe Abbildung 20 für ein Beispiel).

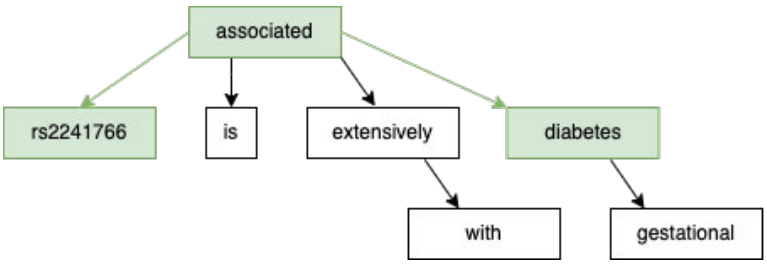


Abbildung 21 Positive Relation: rs2241766 - gestational diabetes

6.3.3 Deep Learning mit Transformers

Die ersten zwei Modelle verwenden kein maschinelles Lernen, sondern nur bestimmte Muster, um die Relationen zwischen den Entitäten zu erkennen. Diese Modelle sollen mit einer State-of-the-Art Technologie verglichen werden. Derzeit entsprechen Transformer Modelle dem aktuellen Stand der Technik im NLP-Bereich [51].

Transformers sind neuronale Netzwerkarchitekturen, die mit einem großen Korpus vortrainiert sind und jeden Token in dem Dokument kontextsensitiv in der Form eines dichten Vektors darstellen. Diese Kontextsensitivität ist ein großer Vorteil von Transformers.

Transformers erhöhen die Genauigkeit der Vorhersagen, nehmen dafür aber höhere Trainings- und Laufzeitkosten in Anspruch. SpaCy empfiehlt eine NVIDIA-GPU mit mindestens 10 GB Speicher, um mit Transformer-Modellen zu arbeiten. Aus diesem Grund wurde dieser Teil der Arbeit in einem Google Collab Notebook implementiert, der einen Zugriff auf die GPUs bietet.

SpaCy Transformers Plugin bietet eine vollständige Integration mit allen vortrainierten Modellen von der HuggingFace Library.

BERT ist eines der am häufigsten verwendeten Transformer Modelle. Google stellte BERT im Jahr 2018 vor [52]. Dieses Modell besteht aus mehreren Encoder-Schichten, die übereinandergestapelt sind. Die erste Schicht initialisiert einen Vektor für jedes Wort nach dem Zufallsprinzip und die nächste Encoder-Schicht transformiert dann die Ausgabe des vorherigen Encoders. Nach 12 Schichten in dem BERT Base und 24 Schichten in dem BERT Large wird ein Vektor für jedes Wort zurückgegeben.

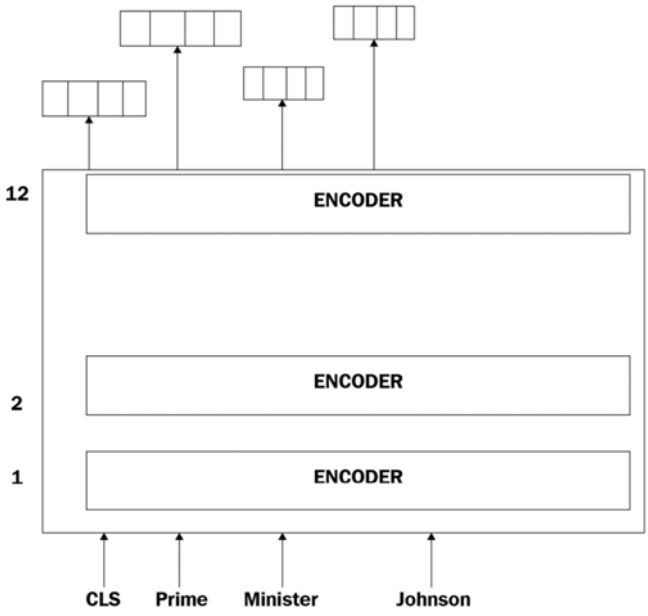


Abbildung 22 Inputs und Outputs im BERT Modell [53]

RoBERTa ist ein anderes Modell auf Basis von BERT. RoBERTa hat dieselbe Architektur aber wurde mit mehr Daten und mit längeren Sätzen vortrainiert (16G vs 160G in BERT) [54]. Im Rahmen dieser Arbeit wurden sowohl BERT als auch RoBERTa verglichen.

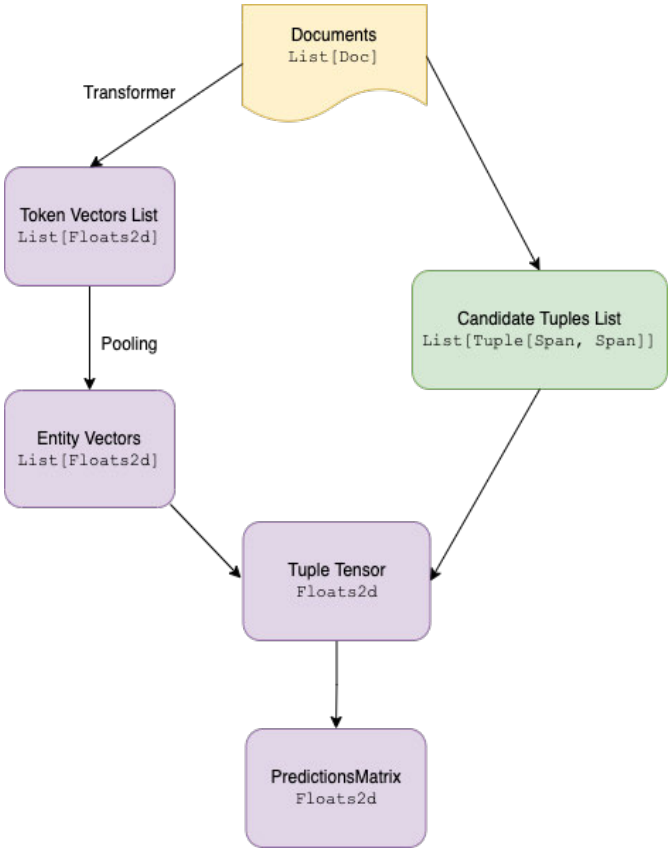


Abbildung 23 Prozessablauf des dritten Modells

Die Architektur des Relation Extraction Modells mit SpaCy und Transformers ist auf der Abbildung 22 aufgeführt.

200 Abstrakte wurden in Training-, Validation- und Test-Set aufgeteilt. Die empfohlene Proportion ist 60:20:20, d.h. das Modell wurde mit 120 annotierten Abstrakten trainiert, mit 40 validiert und noch mit 40 getestet.

Die Dokumente wurden mit dem Annotation Tool UBIAI annotiert: SNP- und DISEASE-Entitäten sowie positive und negative Relationen zwischen diesen Entitäten wurden markiert.

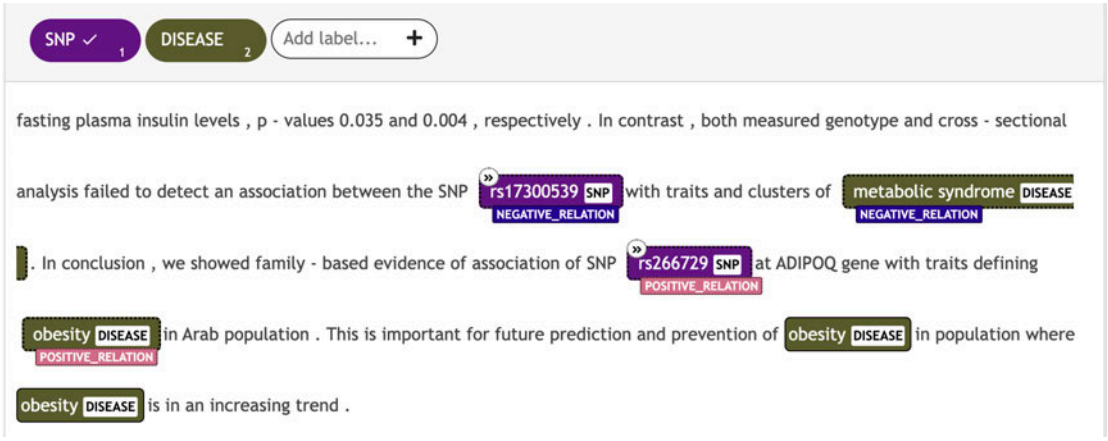


Abbildung 24 Markierung der Daten mit dem UBIAI Tool

Als nächstes wandelt ein Transformer (BERT bzw. RoBERTa) jeden Token von jedem Dokument in einen Vektor um (siehe Schritt 2 auf der Abbildung 24). Wie im Abschnitt 6.4.2 bereits erwähnt, stellen die Entitäten, die aus mehreren Wörtern bestehen, eine Herausforderung für die Algorithmen dar. Pooling reduziert die Anzahl der Dimensionen einer Entität, d.h. fasst mehrere Vektoren für jedes Wort zu einem Vektor zusammen (Schritt 3 auf der Abbildung 24).

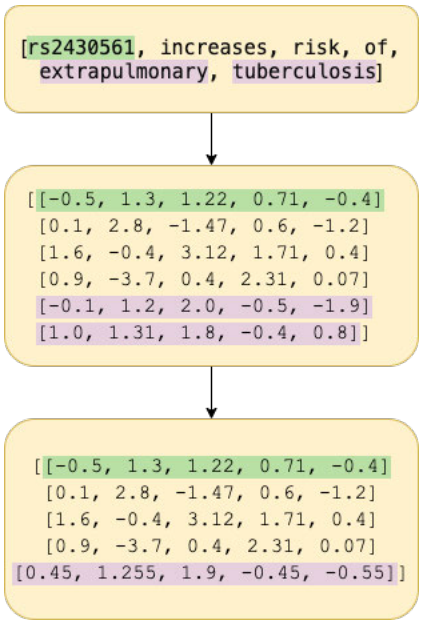


Abbildung 25 Transformer und Pooling

Parallel dazu werden die Kandidaten für die Entitätspaare gesucht. Dafür soll zuerst der maximale Abstand zwischen zwei Entitäten definiert werden, die potenziell in einer Beziehung stehen können.

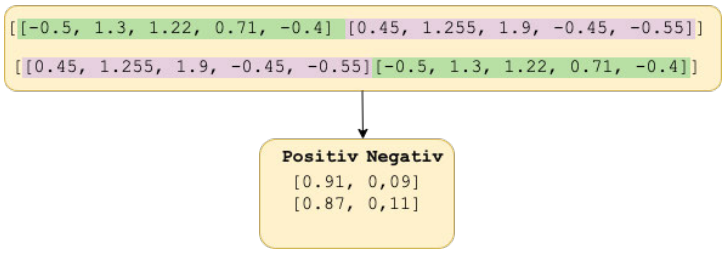


Abbildung 26 PredictionMatrix

Die Vektoren für zwei Kandidaten stellen einen Tensor bzw. eine Matrix dar. Diese Matrix dient als Input für die Klassifizierungsschicht (Schritt 4). Als Output erhält man eine PredictionMatrix, die besagt, wie hoch die Wahrscheinlichkeit einer positiven bzw. einer negativen Relation ist (Abbildung 25). Eine 50-prozentige Wahrscheinlichkeit gilt als Grenzschwelle für eine valide Relation.

7 Bewertung und Ergebnisse

7.1 Bewertungskriterien

Um die Algorithmen zu evaluieren und ihre Ergebnisse zu vergleichen, müssen einheitliche Bewertungskriterien erarbeitet werden. Die Aufgabe der Algorithmen besteht darin, möglichst viele korrekte Relationen zu finden. Dabei handelt es sich um eine binäre Klassifikation – die Relation entweder existiert oder nicht.

Vier Fälle können während der binären Klassifikation auftreten:

TP – eine Relation wurde richtig erkannt

FP – eine Relation wurde fälschlicherweise erkannt

TN – es wurde erkannt, dass es keine Relation gibt

FN – eine Relation wurde nicht erkannt

Um Precision, Recall und F-Score eines Klassifikatoren zu bemessen, wird gezählt, wie oft jeder von diesen vier Fällen auftritt.

Precision beschreibt das Verhältnis der korrekt erkannten Relationen zu allen erkannten Relationen. Recall misst das Verhältnis zwischen korrekt erkannten Relationen und tatsächlichen Relationen. F-Score ist das harmonische Mittel zwischen Recall und Precision und fasst die beiden Metriken zu einem Wert zusammen. Allerdings ist die Precision wichtiger für die gegebene Aufgabenstellung, denn die Nichterkennung einer Relation ist nicht so schlecht wie die Erkennung einer falschen Relation, die den Behandlungsprozess und folglich die Gesundheit von Menschen beeinflussen könnte.

Vor der Bewertung der Klassifikatoren soll zuerst definiert werden, was genau unter einer Relation zwischen einem SNP und einer Krankheit gemeint ist:

- Eine direkte Verbindung zwischen einem Snip und einer Krankheit gilt als eine positive Relation.

- Transitive Relationen Snip – Arzneistoff – Nebenwirkung gelten nicht als positive Relationen, weil die Krankheit nur als Reaktion auf ein Medikament entsteht.
- Relationen Snip – Krankheit – Arzeneistoff gegen Krankheit werden nicht berücksichtigt, denn es handelt sich lediglich um die genetisch bedingte Empfindlichkeit für Arzneimittel gegen bestimmte Erkrankungen.
- Die Voraussetzungen für die Entstehung einer Assoziation (Nationalität, Geschlecht, Lebensstil) spielen keine Rolle. Wenn die Krankheitsdisposition nur für Tataren/Frauen/Nichtraucherinnen gilt, dann ist es trotzdem eine positive Relation.
- Eine negative Relation gilt dann, wenn explizit steht, dass es keine Assoziation zwischen einem Snip und einer Krankheit besteht.

Die Bewertung von biomedizinischen Texten ist eine komplizierte Aufgabe und soll von mindestens zwei Experten durchgeführt werden. Da es im Rahmen dieser Arbeit nicht möglich war, können die Ergebnisse abweichen.

7.2 Ergebnisse

Verfahren	Precision	Recall	F-Score
Rule-Based	0.77	0.47	0.58
Dependency Parsing	0.75	0.51	0.6
Transformer BERT	0.64	0.63	0.64
Transformer RoBERTa	0.81	0.42	0.55

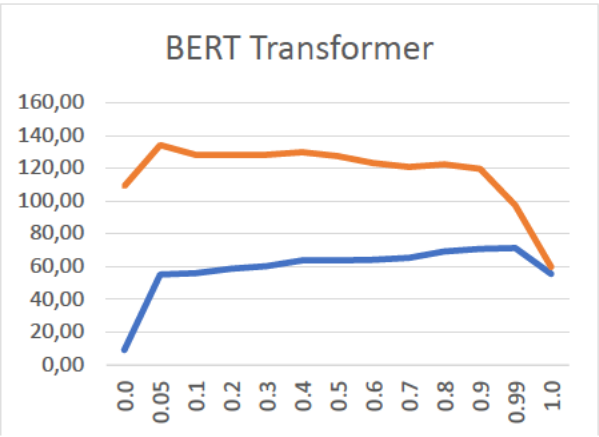
Tabelle 3 Ergebnisse von vier Modellen

Die Ergebnisse von den Modellen sind auf der Abbildung 26 dargestellt.

Das Rule-Based Modell hat eine gute Präzision von 77% und einen deutlich niedrigeren Recall von 47%. Dies lässt sich durch strenge Regeln erklären, die eine kleine Anzahl der Vorhersagen erzeugen. Eine kleine Anzahl der Vorhersagen bedeutet wenige falsche Aussagen, dafür aber übersieht der Algorithmus viele Relationen.

Die Dependency Parsing Modell hat einen höheren Recall und eine 0.02% niedrigere Precision. Dieser Algorithmus findet mehr Relationen als der strenge Rule-Based Algorithmus, was den höheren Recall erklärt. Auch bei den langen Aufzählungen übertrifft das Dependency Parsing Verfahren den Rule-Based Algorithmus. Da die Regeln nur ein Window von fünf Tokens haben, gehen die am Rande stehenden Entitäten verloren. Im Gegensatz dazu findet das zweite Modell alle Relationen, weil die aufgezählten Wörter ein gemeinsames syntaktisches Elternteil und einen ähnlichen Pfad haben.

Zu beachten ist, dass die Daten unbalanciert sind. Die Klassen stehen im Verhältnis 4:1 – die meisten Relationen sind positiv. Bei der Evaluierung der Ergebnisse ist zu beobachten, dass das Rule-Based Modell bei der Erkennung negativer Beziehungen besser als das Dependency Parsing Modell funktioniert. Das kann dadurch erklärt werden, dass die Regeln mit einer erweiterten Liste von negierenden Wörtern versehen sind, wobei das Dependecy Parsing nur *no*, *not*, *nor*, *n't*, *neitther* und *never* kennt. Aufgrund des Ungleichgewichts zwischen den Klassen fällt dies aber nicht auf. Als Schwellenwert für PredictionMatrix der beiden Transformer Modelle wurde 0,5 gewählt. Wenn die Wahrscheinlichkeit einer Relation höher als 50% ist, dann gilt diese Relation. Die Ergebnisse für verschiedene Schwellenwerte sehen wie folgt aus:



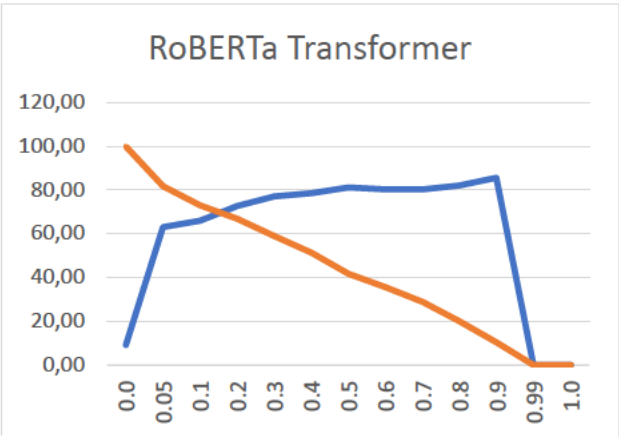


Tabelle 4 Precision und Recall für verschiedene Schwellenwerte

Ein BERT Transformer hat den besten Recall unter allen vier Modellen (63%) – und die schlechteste Precision. Aufgrund des hohen Recalls besitzt dieses Modell den höchsten F-Score. RoBERTa Transformer zeigt hingegen eine ausgezeichnete Precision und den niedrigsten Recall.

Mit dem besten Ergebnis von 63% haben alle Modelle einen mittelmäßigen Recall. Folgende Faktoren haben die Erkennung der Relationen erschwert:

Abkürzungen stellen eine Herausforderung für Tokenisierung und Named Entity Recognition dar. Häufig verwendete offizielle Fachabkürzungen werden von der NER-Komponente erkannt aber einige Autoren halten sich nicht an Standards und nutzen inoffizielle oder selbst ausgedachte Abkürzungen.

Ein anderes Problem ist die Nomenklatur der Marker. Im Rahmen dieser Arbeit wird der von dbSNP eingeführte rs### Standard betrachtet. The Human Genome Variation Society (HGVS) hat einen anderen Standard, der mehr Information über jeden SNP vermittelt: seine kodierende Region, das ersetzte Nukleotid und das ersetzende Nukleotid (z.B. 1377C > G und rs266729 bezeichnen denselben SNP). In einigen Publikationen kommen beide Formen gemischt vor. Da der HGVS Standard von den Modellen nicht berücksichtigt wird, gehen viele Relationen verloren.

Trotz dieser Umstände ist ein Relation Extraction Prototyp zustande gekommen, der allen Anforderungen aus dem Abschnitt 5.1 entspricht: das System hat einen Zugriff auf PubMed (Anforderungen 1.1, 1.2) und auf die Datenbank mit den SNPs (1.3), kann die Snips nach „Beliebtheit“ sortieren (1.5) und die relevanten Informationen zu jeden Snip zurückgeben (1.4). Das System unterstützt drei Algorithmen (1.7), die Relationen zwischen Einzelnukleotid-Polymorphismen und wahrscheinlichen Krankheiten finden (1.6) und in zwei Kategorien einteilen (1.8). Die mit ersten zwei Verfahren gefundenen Relation werden in die Datenbank eingetragen und können angeschaut werden (1.9).

8 Zusammenfassung und Ausblick

Relation Extraction ist eine der wichtigsten Aufgaben des Natural Language Processing. Relation Extraction wird in den Fragebeantwortung- und Dialogsystemen eingesetzt, auch für die Textklassifizierung und Textanalyse spielt diese Technologie eine große Rolle.

Außerdem ist sie eine der kompliziertesten Aufgaben des Natural Language Processing. Diese Arbeit zeigt, dass es mehrere Ansätze gibt, um diese Aufgabe einer Lösung näherzubringen.

Als Anwendungsgebiet wurde Biomedizin ausgewählt – ein gut strukturierter Bereich, der durch Ursache-Wirkungs-Zusammenhänge beschrieben werden kann. Drei Ansätze und vier Modelle wurden implementiert und verglichen.

Der erste Ansatz basiert auf 18 strengen domainspezifischen Regeln bzw. Patterns, die nach bestimmten Testabschnitten suchen und daraus die Relationen extrahieren. Mit einer Precision von 77% und einem Recall von 47% belegte dieses Modell den dritten Platz. Es findet zwar nur die Hälfte aller Relationen, aber diese Relationen sind mit hoher Wahrscheinlichkeit korrekt, was für Präventionsmedizin besonders wichtig ist.

Der zweite Ansatz beruht auf der Idee, dass die Satzsyntax die Beziehungen zwischen den Entitäten abbilden kann. Der Vorteil dieses Ansatzes besteht in seiner Universalität – er kann ohne große Änderungen auf einen anderen Bereich angewendet werden. Mit F-Score von 60% belegt dieses Modell den zweiten Platz.

Die ersten zwei Ansätze wurden implementiert, um eine Vorstellung von dem Thema zu bekommen und die historische Entwicklung von Text Mining Technologien zu verfolgen. Die nächsten zwei Modelle wurden mit der State-of-the-Art Technologie Transformers implementiert. Der einzige Unterschied zwischen den beiden Verfahren besteht in dem vortrainierten Modell (BERT vs. RoBERTa). BERT hat die besten Ergebnisse unter allen vier Modellen gezeigt (F-Score 64%). Dabei hatte RoBERTa die höchste Precision und BERT den höchsten Recall unter allen vier Modellen.

Natürlich reicht so ein kleines Dataset (120 wissenschaftliche Abstracts) für ein qualitatives Deep Learning Modell nicht aus.

Künftig sollen die Modelle mit größeren Datasets trainiert und getestet werden. Nicht nur die Anzahl der Daten soll vergrößert werden – neue Entitäts- und Relationstypen sind notwendig, um ein komplexes Graphensystem zu bilden. Die Relation Gen – SNP wäre besonders wichtig, weil die Auswirkung von Snips von ihrer Lage in Genom abhängt.

Außerdem sollen in den weiteren Arbeiten die im Abschnitt 7.2 erwähnten Probleme angesprochen werden. Da es mehrere Nomenklaturen für Einzelnukleotid-Polymorphismen gibt, soll für sie ein Mapping erstellt werden. Die Erkennung biomedizinischer Abkürzungen ist eine andere Aufgabe für die künftige Arbeit. Auch die Validierung der erkannten Relationen ist ein wichtiges Forschungsthema.

Literaturverzeichnis

[1] Regalado, A., 2019. More than 26 million people have taken an at-home ancestry test. In: https://www.technologyreview.com/2019/02/11/103446/more-than-26-million-people-have-taken-an-at-home-ancestry-test/, abgerufen am 10.02.2022.

[2] Robinson, T., 2006. Genetik für Dummies. Weinheim: WILEY-VCH Verlag

[3] Graw, J., 2010. Genetik. Berlin, Heidelberg: Springer Berlin Heidelberg

[4] Wenn die Welt an einem Strang zieht: Das Humangenomprojekt (HGP). In: ngfn.de. *Nationales Genomforschungsnetz*, abgerufen am 24.04.2021.

[5] Weiß, M. G., 2009. Die Auflösung der menschlichen Natur. Frankfurt am Main: Suhrkamp Verlag

[6] Gen. Kompaktlexikon der Biologie. In: https://www.spektrum.de/lexikon/biologie/gen/27194, abgerufen am 25.04.2021 Heidelberg: Spektrum Akademischer Verlag

[7] Genexpression. Kompaktlexikon der Biologie. In: https://www.spektrum.de/lexikon/biologie-kompakt/genexpression/4693, abgerufen am 25.04.2021. Heidelberg: Spektrum Akademischer Verlag

[8] Munk, K., Jahn, D., 2010. Genetik. Fit für den Bachelor. Stuttgart: Thieme Verlag

[9] Human Genome Project Results (2018, November 12). In: https://www.genome.gov/human-genome-project/results, abgerufen am 25.04.2021

[10] Hobson L., Hannes H., Cole H., 2019. Natural Language Processing in Action: Understanding, analyzing, and generating text with Python. New York: Manning Publications

[11] Hutchins, J. W., 2005. The history of machine translation in a nutshell, Technical Report. *University of East Anglia.*

[12] Carstensen, K.-U., Ebert Ch., 2010. Computerlinguistik und Sprachtechnologie. 3. Aufl. Heidelberg: Spektrum Akademischer Verlag

[13] Eisenstein, J., 2018. Natural Language Processing. *UCSD CSE*

[14] Indurkhya, N., Damerau, F.J., 2010. Handbook of Natural Language Processing, Second Edition. London: Chapman & Hall CRC Machine Learning & Pattern Recognition Series)

[15] Arkhipov, I., 2018. Extraktion und Stimmungsanalyse von Tweets bezüglich bestimmter Schlüsselwörter. Bachelorarbeit, *Hochschule für Angewandte Wissenschaften Hamburg*

[16] Nasar, Z., Jaffry, S.W., Malik M.M., 2021. Named Entity Recognition and Relation Extraction: State of the Art. *ACM Computing Surveys 54 (1)*

[17] Bach, N., Badaskar, S., 2017. A Review of Relation Extraction. In: http://www.cs.cmu.edu/, abgerufen am 10.12.2021

[18] Konstantinova, N. et al. (2014): Review of Relation Extraction Methods: What is New Out There? *AIST 2014: Analysis of Images, Social Networks and Texts pp 15-28*

[19] Jurafsky, D., Martin, J.H., 2009: "Information Extraction." Chapter 22 in Speech and Language Processing, Second Edition. New Jersey: Prentice-Hall, Inc.

[20] Zong, C., Xia, R., Zhang, J., 2021. Text Data Mining. Singapore: Springer

[21] Agichtein, E., Gravano, L., 2000. “Snowball: Extracting relations from large plain-text collections.” *Proceedings of the fifth ACM conference on Digital libraries. ACM, 2000.*

[22] Shi, Y., Xiao, Y., Niu, L., 2019. A Brief Survey of Relation Extraction based on Distant Supervision. ICCS 2019. Lecture Notes in Computer Science, vol 11538. Cham: Springer

[23] Pawar, S., Palshikar, G.K., Bhattacharyya, P., 2017. Relation Extraction: A Survey. arXiv:1712.05191 [cs.CL]. Abgerufen am 27.12.2021

[24] Yin, W., Kann, K., Yu M., Schutze, H., 2017. Comparative Study of CNN and RNN for Natural Language Processing. arXiv:1702.01923 [cs.CL]. Abgerufen am 02.01.2022

[25] Vaswani, A., et al., 2017. Attention Is All You Need. arXiv:1706.03762 [cs.CL] Abgerufen am 20.02.2022

[26] Ruder, S., 2020. Relationship Extraction. In: http://nlpprogress.com/english/relationship_extraction.html. Abgerufen am 02.01.2022

[27] Giles, O., et al., 2020. Optimising biomedical relationship extraction with BioBERT. In: https://www.biorxiv.org/content/10.1101/2020.09.01.277277v1.full. Abgerufen am 10.02.202

[28] Chapman, W.W. et al., 2002. Evaluation of negation phrases in narrative clinical reports. *AMIA Annual Symposium Proceedings 2001*

[29] Bokharaeian, et al., 2017. SNPPhenA: a corpus for extracting ranked associations of single-nucleotide polymorphisms and phenotypes from literature. *J Biomed Semantics*

[30] Chapman, W.W., et al., 2001. A simple algorithm for identifying negated findings and diseases in discharge summaries. *J Biomed Inform 2001*

[31] Ballesteros, M., et al., 2012. Inferring the Scope of Negation in Biomedical Documents. *International Conference on Intelligent Text Processing and Computational Linguistics*

[32] Tanabe, L., Xie, N., Thom, L.H., 2005. GENETAG: A tagged corpus for gene/protein named entity recognition. *BMC Bioinformatics*

[33] Perera, N., Dehmer, M., Emmert-Streib, F., 2020. Named Entity Recognition and Relation Detection for Biomedical Information Extraction. *Front. Cell Dev. Biol. 2020*

[34] D'Souza, J., Ng, V., 2012. Anaphora resolution in biomedical literature: a hybrid approach. *ACM Conference on Bioinformatics, Computational Biology and Biomedicine (ACM)*

[35] Li, H., Chen, Q., Tang, B., Wang, X., Xu, H., Wang, B., et al. 2017. CNN-based ranking for biomedical entity normalization. *BMC Bioinformatics*

[36] Saad, F., Aras, H., Hackl-Sommer, R., 2020. Improving Named Entity Recognition for Biomedical and Patent Data Using Bi-LSTM Deep Neural Network Models. Cham: Springer

[37] Luo, L., et al., 2018. A neural network approach to chemical and gene/protein entity recognition in patents. *Journal of Cheminformatics 10, Article number: 65*

[38] Scharbert, K., 2005. Requirements Analysis realisieren. Wiesbaden: Vieweg+Teubner Verlag

[39] DIN 66272:1994-10. In: beuth.de. Abgerufen am 21.02.2022.

[40] Sayers, E.W., et al. (2021): Database resources of the National Center for Biotechnology Information. *Nucleic Acids Research, Volume 49, Issue D1, 8 January 2021*

[41] Cariaso, M., Lennon, G., 2011. SNPedia: a wiki supporting personal genome annotation, interpretation and analysis. *Nucleic Acids Research, Volume 40, Issue D1, 1 January 2012, Pages D1308–D1312*

[42] SNPedia FAQ. In: https://snpedia.com/index.php/SNPedia:FAQ, abgerufen am 24.10.2021

[43] Fiorini, N., Lipman, D., Lu, Z., 2017. Towards PubMed 2.0. PMC5662282. In: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5662282/. Abgerufen am 04.10.2021

[44] Chapman, B., Chang, J., 2000. Biopython: Python tools for computational biology. *ACM SIGBIO Newsletter 20*

[45] Carlson, J., 2013. Redis in Action. New York: Manning Publications Co

[46] Narkhede, N., Sivaram, R., Palino, T., Shapira. G., 2021. Kafka: The Definitive Guide. 2. Auflage. Sebastopol: O'Reilly Media, Inc.

[47] Chodorow, K., 2013. MongoDB: The Definitive Guide, Sebastopol: O'Reilly Media, Inc.

[48] Hopf, F., 2016. Elasticsearch: Ein praktischer Einstieg. Heidelberg: dpunkt.verlag

[49] Jiao Li et al. (2016) : "BioCreative V CDR task corpus: a resource for chemical disease relation extraction", Database, Volume 2016

[50] Berrettini, W.H., 2001. The Human Genome: Susceptibility Loci. *Am J Psychiatry 158:6, June 2001.* In: https://ajp.psychiatryonline.org/doi/full/10.1176/appi.ajp.158.6.865. Abgerufen am 05.02.2022

[51] Chai, J., Li, A., 2019: Deep Learning in Natural Language Processing: A State-of-the-Art Survey. *2019 International Conference on Machine Learning and Cybernetics (ICMLC)*

[52] Delvin, J., et al., 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805v2 [cs.CL] In: https://arxiv.org/abs/1810.04805. Abgerufen am 10.02.2022

[53] Altinok, D., 2021. Mastering spaCy: An end-to-end practical guide to implementing NLP application using the Python ecosystem. Birmingham: Packt Publishing

[54] Liu, Y. (2019): RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692 [cs.CL] In: https://arxiv.org/abs/1907.11692. Abgerufen am 10.02.2022

A 50 meist untersuchte SNPs

'Rs2282679', 'Rs1570360', 'Rs12980275', 'Rs1344706', 'Rs28362491', 'Rs1421085', 'Rs1057910', 'Rs11200638', 'Rs1799964', 'Rs20541', 'Rs2254298', 'Rs662799', 'Rs2292832', 'Rs1799782', 'Rs2108622', 'Rs6313', 'Rs1799793', 'Rs2430561', 'Rs6311', 'Rs833061', 'Rs1143627', 'Rs1800587', 'Rs6295', 'Rs780094', 'Rs3212986', 'Rs861539', 'Rs1052133', 'Rs1137101', 'Rs2279744', 'Rs3087243', 'Rs1205', 'Rs13266634', 'Rs1051730', 'Rs1946518', 'Rs2243250', 'Rs3918242', 'Rs1333049', 'Rs8050136', 'Rs10830963', 'Rs11209026', 'Rs4986791', 'Rs662', 'Rs187238', 'Rs4073', 'Rs4149056', 'Rs3212227', 'Rs2070744', 'Rs2231142', 'Rs266729', 'Rs1360780'

B Liste verbindender Wörter

"associate", "indicate", "indication", "interact", "interaction", "link", "linkage", "confer", "connect", "connection", "correlate", "relate", "relationship", "affect", "contribute", "contribution", "attribute", "underlie", "cause", "determine", "role", "effect", "correlation", "factor", "association", "predictor", "explain", "susceptibility loci", "susceptibility locus", "relationship", "criterion", "marker", "predisposition", "susceptibility", "influence", "risk", "predispose", "affiliate", "result", "vulnerability"

C Negative Regeln

1. SNP - 5 Tokens Window - Negation - 2 Tokens - Verbindung - 5 Tokens Window - Krankheit
2. Krankheit - 5 Tokens Window - Negation - 2 Tokens - Verbindung - 5 Tokens Window - SNP
3. Krankheit - 5 Tokens Window - SNP - 5 Tokens Window - Negation - 2 Tokens - Verbindung
4. SNP - 5 Tokens Window - Krankheit - 5 Tokens Window - Negation - 2 Tokens - Verbindung
5. Negation - 2 Tokens - Verbindung - 5 Tokens Window - Krankheit - 5 Tokens Window - SNP
6. Negation - 2 Tokens - Verbindung - 5 Tokens Window - SNP - 5 Tokens Window – Krankheit
7. SNP - 5 Tokens Window - Verbindung - 5 Tokens Window – Krankheit - 5 Tokens Window – Negation
8. Krankheit - 5 Tokens Window - Verbindung - 5 Tokens Window – SNP - 5 Tokens Window - Negation
9. Krankheit - 5 Tokens Window - SNP - 5 Tokens Window – Verbindung - 5 Tokens Window - Negation
10. SNP - 5 Tokens Window - Krankheit - 5 Tokens Window – Verbindung - 5 Tokens Window - Negation
11. Verbindung - 5 Tokens Window - Krankheit - 5 Tokens Window – SNP - 5 Tokens Window - Negation
12. Verbindung - 5 Tokens Window - SNP - 5 Tokens Window – Krankheit - 5 Tokens Window - Negation
13. Negation - 2 Tokens - SNP - 5 Tokens Window - Verbindung - 5 Tokens Window - Krankheit
14. Krankheit - 5 Tokens Window - 5 Tokens Window - Negation - 2 Tokens - SNP

15. Krankheit - 5 Tokens Window - Negation - 2 Tokens - SNP - 5 Tokens Window - Verbindung
16. Negation - 2 Tokens - SNP - 5 Tokens Window - Krankheit - 5 Tokens Window - Verbindung
17. Verbindung - 5 Tokens Window - Krankheit - 5 Tokens Window - Negation - 2 Tokens - SNP
18. Verbindung - 5 Tokens Window - Negation - 2 Tokens - SNP - 5 Tokens Window - Krankheit

Erklärung zur selbstständigen Bearbeitung einer Abschlussarbeit

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original