Albert-Ludwigs-Universität Freiburg

**Automatisierte Identifizierung von Wirkstoffen in wissenschaftlicher Literatur mithilfe maschineller Lernverfahren**

Bachelorarbeit

*Pharmazeutische Wissenschaften*

Manuel Dorer

Fakultät für Chemie und Pharmazie,

Institut für Pharmazeutische Wissenschaften,

Pharmazeutische Bioinformatik

vorgelegt von

Manuel Dorer

am XX .XX 2021, Freiburg

Erstgutachter: Prof. Dr. Stefan Günther

Pharmazeutische Bioinformatik,

Albert-Ludwigs-Universität Freiburg

Zweitgutachter: Jun.-Prof. Dr. Jennifer Andexer,

Pharmazeutische und Medizinische Chemie,

Albert-Ludwigs-Universität Freiburg

Betreuer: Ammar Qaseem,

Pharmazeutische Bioinformatik,

Albert-Ludwigs-Universität Freiburg



Eigenständigkeitserklärung

Name des Studenten: Manuel Dorer

Matrikelnummer: 4531053

*ERKLÄRUNG*

*zur Abgabe der Bachelor-Arbeit   
im Studiengang B.Sc. Pharmazeutische Wissenschaften*

Hiermit versichere ich, dass ich die Bachelorarbeit selbstständig verfasst und keine andere als die von mir angegebenen Quellen und Hilfsmittel benutzt habe. Alle Zitate sind gekennzeichnet und alle Abbildungen enthalten nur die originalen Daten und sind in keinem Fall inhaltsverändernder Bildbearbeitung unterzogen worden. Die abgegebene schriftliche und elektronische Fassung sind identisch. Weiterhin versichere ich, dass die Arbeit noch nicht anderweitig als Bachelorarbeit eingereicht wurde.

............................................ ........................................................

Ort/Datum Unterschrift des Studenten

# Abstract

In recent years, deep learning has garnered tremendous success in a variety of application

domains. This new field of machine learning has been growing rapidly and has been applied to

most traditional application domains, as well as some new areas that present more opportunities.

Different methods have been proposed based on different categories of learning, including

supervised, semi-supervised, and un-supervised learning. Experimental results show state-of-theart

performance using deep learning when compared to traditional machine learning approaches in

the fields of image processing, computer vision, speech recognition, machine translation, art,

medical imaging, medical information processing, robotics and control, bioinformatics, natural

language processing, cybersecurity, and many others. This survey presents a brief survey on the

advances that have occurred in the area of Deep Learning (DL), starting with the Deep Neural

Network (DNN). The survey goes on to cover Convolutional Neural Network (CNN), Recurrent

Neural Network (RNN), including Long Short-Term Memory (LSTM) and Gated Recurrent Units

(GRU), Auto-Encoder (AE), Deep Belief Network (DBN), Generative Adversarial Network (GAN),

and Deep Reinforcement Learning (DRL). Additionally, we have discussed recent developments,

such as advanced variant DL techniques based on these DL approaches. This work considers most

of the papers published after 2012 from when the history of deep learning began. Furthermore, DL

approaches that have been explored and evaluated in different application domains are also

included in this survey. We also included recently developed frameworks, SDKs, and benchmark

datasets that are used for implementing and evaluating deep learning approaches. There are some

surveys that have been published on DL using neural networks and a survey on Reinforcement

Learning (RL). However, those papers have not discussed individual advanced techniques for

training large-scale deep learning models and the recently developed method of generative models.

🡺 Heutzutage sind noch viele Datensätze von Hand zusammengestellt

# **Inhaltsverzeichnis**

[Abstract i](#_Toc66478120)

[Inhaltsverzeichnis ii](#_Toc66478121)

[Abbildungsverzeichnis iii](#_Toc66478122)

[Tabellenverzeichnis iv](#_Toc66478123)

[Abkürzungsverzeichnis v](#_Toc66478124)

[1 Einleitung 1](#_Toc66478125)

[1.1 Begriffe aus dem Feld der KI 1](#_Toc66478126)

[1.1.1 KI 1](#_Toc66478127)

[1.1.2 maschinelles Lernen 2](#_Toc66478128)

[1.2 ~~Finden neuer Wirkstoffe im Zeitalter von Big Data KI in der Pharmazeutischen Bioinformatik~~ bessere überschrift vlt datamining 12](#_Toc66478129)

[1.2.1 klassischer Weg 12](#_Toc66478130)

[1.2.2 virtuelles Screening 12](#_Toc66478131)

[1.3 CPI Text Mining 13](#_Toc66478132)

[1.3.1 Historie von RE-Text Mining mit Blick auf CPI-Text Mining 14](#_Toc66478133)

[2 Zielsetzung der Arbeit 17](#_Toc66478134)

[3 Motivation der Arbeit 17](#_Toc66478135)

[4 Material und Methoden 17](#_Toc66478136)

[4.1 Allgemeines 17](#_Toc66478137)

[4.1.1 Systemspecs 17](#_Toc66478138)

[4.2 Installation 17](#_Toc66478139)

[4.3 Datensätze 17](#_Toc66478140)

[4.4 Statistische Parameter zur Bewertung 18](#_Toc66478141)

[4.5 10-fache Kreuzvalidierung 18](#_Toc66478142)

[5 Ergebnisse 18](#_Toc66478143)

[6 Diskussion und Ausblick 18](#_Toc66478144)

[7 Zusammenfassung 19](#_Toc66478145)

[8 Literaturverzeichnis 20](#_Toc66478146)

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Abkürzungsverzeichnis

Im Abkürzungsverzeichnis sind alle im Dokument verwendeten Abkürzungen mit Ihren Erklärungen10 aufzulisten. Auf die im Duden als allgemeinverständlich angeführten Abkürzungen (wie "z. B.", "usw.", "vgl." und Ähnliches) und Akronyme (zum Beispiel "USA") sollte im Abkürzungsverzeichnis verzichtet werden.

# Einleitung

Am 15 März 2016 stand es fest: AlphaGo besiegt den zu der Zeit als stärksten Spieler der Welt geltenden Lee Sedol in einer Partie Go mit 4:1 [1]. Doch was hat dieses Ereignis mit autonom fahrenden Autos, der Wettervorhersage oder Spracherkennung zu tun?

Die Antwort auf diese Frage ist: In allen genannten Bereichen wird zunehmend mit künstlicher Intelligenz (KI) gearbeitet, bzw. gibt es immer mehr Ansätze, welche auf einer KI basieren [2] [3] [4] [5]. KI’s kommen häufig dort zum Einsatz, wo große Datenmengen anfallen und zeitnah bewertet werden müssen. Dabei erkennen KI’s Muster in den Daten und können Vorhersagen über die weitere Entwicklung treffen, was für das Fällen von Entscheidungen von großer Bedeutung ist. Nicht zuletzt haben KI’s auch auf vielfältige Art und Weise ihren Weg in die naturwissenschaftliche Forschung gefunden. Anwendungsgebiete sind z.B.: Das Stellen von Hautkrebsdiagnosen in der Medizin [6], Optimierung von Molekülsimulationen für die Werkstoffforschung in der Physik; ~~dem Finden von neuen potentiellen Wirkstoffen in der Pharmazie;~~ oder generell das Extrahieren von Wissen aus Daten. Besonders letzteres gewinnt im Zeitalter von Big Data immer mehr an Relevanz und soll in dieser Bachelorarbeit im Mittelpunkt stehen.

## Begriffe aus dem Feld der KI

In diesem Kapitel werden für die Arbeit wichtige Begriffe aus dem großen Themengebiet der KI erläutert. Dies stellt einerseits eine Basis für ein Grundverständnis dar und soll andererseits dazu dienen diese Arbeit, mit der verwendeten Methode, in das sehr weitläufige Gebiet der KI besser einzuordnen.

### KI

Eine einfache universelle Definition von KI gibt es nicht. Das liegt daran, dass der Begriff Intelligenz selbst nur schwer zu definieren ist. Die Fraunhofer-Gesellschaft beschreibt KI in einer Studie als „ein Teilgebiet der Informatik mit dem Ziel, Maschinen zu befähigen, Aufgaben »intelligent« auszuführen“ [7, s.8]. Dabei wird versucht die menschliche Intelligenz bzw. die Entscheidungsstrukturen des Menschen mithilfe von Algorithmen und mathematischen Modellen in Computersystemen nachzubilden und diese auf eigene Problemstellungen anzuwenden. Im Zusammenhang mit KI wird zwischen schwacher und starker KI differenziert. Schwache KI’s können nur in einem Bereich ein konkretes Anwendungsproblem bewältigen. D.h. sie sind nicht in der Lage Erlerntes auf andere Bereiche zu übertragen. Typische Anwendungsgebiete sind: Schach/- GO-Computer, Bilderkennung, Spracherkennung, personalisierte Werbung und automatisierte Übersetzung.

Starke KI’s hingegen sind zurzeit nicht existent. Eine starke KI besitzt intellektuelle Fähigkeiten auf Augenhöhe eines Menschen. Diese beruhen darauf, dass die KI in der Lage ist erlerntes Wissen in andere Bereiche zu übertragen und auf neue Probleme anzuwenden. Ob und wann es eine solche allgemeine Intelligenz geben wird, ist in der Wissenschaft umstritten [8]. Der Streitpunkt ist dabei, inwiefern sich typisch menschliche Eigenschaften, wie das Vorhandensein eines Bewusstseins, mit einer KI vereinbaren lassen.

Die Anfänge in der Forschung und Entwicklung KI lassen sich auf das Jahr 1956 datieren, als am Dartmouth College eine Konferenz namens Dartmouth Conference stattfand. Einen ersten Boom verzeichneten KI’s in den 1980er Jahren über die sogenannten Expertensysteme. Die Entscheidungen, welche von den Expertensystemen getroffen werden, basieren auf einer langen Kette von Wenn-Dann-Verknüpfungen. Im zweiten KI-Winter sank das öffentliche Interesse an KI und damit auch die Gelder für die Forschung. Durch die Verwendung von maschinellem Lernen, einer weiteren, bedeutsameren Untergruppe der KI, erlebt diese in dem 21. Jahrhundert eine Blütezeit (siehe Abbildung 1).



Abbildung 1: Entwicklung der Publikationen im Bereich maschinelles Lernen für Deutschland (DE), Frankreich (FR), Großbritannien (UK), USA (US) und China (CN) sowie die restlichen EU-Staaten und übrige Länder  
Quelle: Frauenhofer…. S.

### maschinelles Lernen

Maschinelles Lernen stellt ein wichtiger Zweig von KI dar. Abbildung 2 veranschaulicht das grundlegende Prinzip des maschinellen Lernens.



Abbildung 2: Grundlegendes Prinzip des maschinellen Lernens; icon quelle

Ein IT-System, bestehend aus Lernalgorithmen und mathematischen Modellen, bekommt als Input einen Datensatz und ist in der Lage in diesem Muster und Zusammenhänge zu erkennen. Das erlernte Wissen wird von dem System genutzt, um selbständig ein Modell zu trainieren, welches eine mathematische Beschreibung der im Datensatz gefundenen Gesetzmäßigkeiten ist. Dieses Modell ist der Output des Systems und kann anschließend, auf für das System unbekannte Daten, angewendet werden, um neue Erkenntnisse zu gewinnen. Zusammenfassend kann man sagen, dass der große Unterschied von maschinellem Lernen zu herkömmlichem Programmieren ist, dass der Mensch dem IT‑System keine direkten Regeln vorgibt, nach denen ein bestimmter Output auszugeben ist, sondern, dass das IT‑System diese selbständig finden muss.

Mit der Technik des maschinellen Lernens ist es möglich riesige Datenmengen effizient zu bearbeiten, was dem Menschen schlicht aufgrund der großen Menge nicht möglich wäre. Maschinelles Lernen lässt sich heutzutage in Bereichen wie der Bilderkennung oder der Textanalyse finden.

Ein viel zitiertes Beispiel ist der Einsatz in Antispamfiltern für E-Mails [9]. Hierfür wird ein Modell mit einem Trainingsdatensatz, bestehend aus vielen E-Mails und dem zugehörigen Label *Spam‑E‑Mail* oder *keine Spam‑E‑Mail*, trainiert. Das trainierte Modell ist dann in der Lage bei einer unbekannten E‑Mail zu erkennen, ob es sich um eine *Spam‑E‑Mail* handelt, oder nicht.

Die klassischen Schritte des maschinellen Lernens sind:

**Datenerfassung:** Überführen der Rohdaten, inklusive der Label, in eine maschinenlesbare Form.

**Datenaufbereitung:** Vorbereitung der Daten (z.B. Normalisieren und Durchmischen der Reihenfolge); Am Ende wird der fertige Datensatz auf einen getrennten Test- und Trainingsdatensatz aufgeteilt. Das Verhältnis in dem zufällig gesplittet wird ist standardmäßig 70 % Trainingsdatensatz zu 30 % Testdatensatz, kann aber auch variiert werden. Es ist wichtig sicherzustellen, dass die Verteilung der Daten zwischen den beiden erzeugten Datensätzen ähnlich ist. Eine Möglichkeit die Ähnlichkeit zu bewerten, ist der Vergleich der Anteile der identischen Label in beiden Datensätzen. Bei dem oben beschriebenen Antispamfilter, stellt man so sicher, dass der Anteil der Spam-E-Mails in beiden Datensätzen ähnlich groß ist. Es wird vermieden, dass das Modell einseitig trainiert wird und dadurch eine spätere Bewertung des Models weniger aussagekräftig werden würde.

**Auswahl des passenden Modells:** Je nach Anforderung, wird aus mehrenden Modellen ein passendes ausgewählt. In Abbildung 3 sind einige Modelle hierarchisch dargestellt. Grundsätzlich wird, wenn keine Labelung der Daten vorhanden ist, von unüberwachtem Lernen gesprochen. Der Antispamfilter ist ein Beispiel für überwachtes Lernen, da wie oben beschrieben, eine Labelung vorliegt. Ein überwachtes Lernen Modell wird mit gelabelten Daten trainiert, dass es zukünftig zu ungelabelten Daten ein Label als Output ausgeben kann. Innerhalb des überwachten Lernens wird zwischen den beiden Aufgabentypen Regression und Klassifizierung differenziert. Das Entscheidende einer Klassifizierung, zu der auch der Antispamfilter gehört, ist, dass die Anzahl möglicher Label eine endliche Zahl ist. Bei einer Regression hingegen weist das Modell dem Datenpunkt einen stetigen Wert zu, d.h. das Label hat unendlich viele Ausprägungen. Die für diese Arbeit wichtigen Modelle sind Kern‑Methodenund neuronale Netze. Bei beiden handelt es sich um überwachtes Lernen in Verbindung mit einer Klassifizierungsaufgabe.



Abbildung 3: Übersicht über verschiedene Modelle des maschinellen Lernens. Die für diese Arbeit relevanten Modelle sind in orangener Farbe dargestellt. Quelle(selber gemacht, vorlage aus dem Paper)

**Trainieren des Modells:** Beim Modelltraining geht wird das Modell mit dem Testdatensatz dahingehend optimiert, dass es später möglichst gut mit den unbekannten Daten zurechtkommt. Es kann beim Trainieren des Modells zu den in Abbildung 4 gezeigten Szenarien kommen.



Abbildung 4: Unteranpassung *(underfit),* gute Anpassung *(good fit)* und Überanpassung *(overfit)* beim Modelltraining einer Regression. Die blauen Punkte sind die genauen Werte des Datensatzes und die rote Linie gibt die Näherung durch das trainierte Model an. Quelle: XXXXXXXXXXXXXXX

Von einer Überanpassung ist die Rede, wenn das Modell anfängt sich die Beispiele zu merken, anstelle sich die Werte über gefundene Muster und Zusammenhänge selbst zu erschließen. Bei der Vorhersage mit dem Testdatensatz ist das Modell nicht mehr flexibel genug, um eine gute Vorhersage zu treffen [10]. Das bedeutet, dass die Testgenauigkeit sinkt und dadurch die Differenz zwischen Trainings- und Testgenauigkeit signifikant größer wird, was ein Indiz für eine Überanpassung sein kann. Hingegen liegt eine Unteranpassung vor, wenn das Modell unfähig ist, ein Muster in den Trainingsdaten zu erkennen, um daraus einen Trend ableiten zu können. Zu Unteranpassung kommt es, wenn zu wenige Trainingsdaten vorhanden sind, oder die Komplexität der zu erledigenden Aufgabe, die des Modells wesentlich überschreitet. Das Ziel des Trainings ist es, eine Anpassung zu erreichen, die genau zwischen der einer Überanpassung und der einer Unteranpassung liegt.

**Bewertung des Modells:** Zur Bewertung wird das Modell mit gelabelten Daten des Testdatensatzes getestet, welche bei der Datenerfassung von dem gesamten Datensatz abgetrennt wurden. Als Ergebnis erhält man, mehrere Maßzahlen, welche die generelle Leistung des Modells beschreiben (siehe. Statistische parameter). Zusätzlich können mithilfe einer Validierung die übergeordneten Parameter zur Steuerung des Models *(Hyperparameter)* überprüft und gegebenenfalls angepasst werden. Die Validierung kann mit einem Datensatz gemacht werden, welcher wie der Testdatensatz, dem Trainingsdatensatz vorenthalten wurde (Houldout-Methode). Das Aufteilen auf diese Teildatensätze bringt oftmals das Problem mit sich, dass der Trainingsdatensatz zu klein wird. Um eine Unteranpassung zu vermeiden, kann die Validierung auch mit einer Kreuzvalidierung erfolgen (siehe. 10-fache Kreuzvalidierung).

**Vorhersage:**  Das Modell wird an unbekannten Daten angewandt.

Im Folgenden wird die Funktionsweise der beiden Modelle Kern-Methoden und neuronale Netze näher erläutert.

#### Kern-Methoden

Mithilfe von Kern‑Methoden (engl. kernel‑methods) lassen sich nicht linear klassifizierbare Daten klassifizieren. Für das Verständnis von Kern‑Methoden ist es sinnvoll erst die lineare Klassifizierung zu betrachten.

Bei einer **linearen Klassifizierung** ist das Ziel eine Hyperebene zu finden, welche die Datenpunkte eines Datensatzes in zwei Klassen einteilt (binäre Klassifizierung). Im zweidimensionalen Vektorraum ist diese Hyperebene eine Gerade (siehe Abbildung 5). Allgemein können Hyperebenen mit der Funktion beschrieben werden. Später kann anhand des Vorzeichens der Funktion bei Einsetzung eines zu klassifizierenden Datenpunktes , die Klasse bestimmt werden. Um die optimale Hyperebene zu finden, muss der Rand (Margin) um die Klassengrenzen möglichst breit werden (engl. *large‑margin‑classification*) [11] , bzw. , was dem Margin entspricht, maximiert werden. Mithilfe des Sattelpunktes der Lagrange-Funktion [12] lassen sich die beiden Stützvektoren definieren, die sich bei einer Margin von befinden und somit die Hyperebene umgeben (siehe Abbildung 5).



Abbildung 5: Lineare Klassifizierung eines Datensatzes mit zwei Eigenschaften pro Datenpunkt. Variablen beschreiben noch

Die *large‑margin‑classification* erreicht ihre Grenzen, wenn bereits wenige Ausreißer im Datensatz vorhanden sind. Der Margin würde sehr klein werden und es würde sich eine große Ungenauigkeit für die Klassifizierung ergeben. Die Einführung der Schlupfvariable , welche einerseits Fehleinordnungen erlaubt, diese aber andererseits auch bestraft, stellt eine Möglichkeit dar, diesen potenziellen Fehler zu vermindern. ist Bestandteil der *soft‑margin‑classification* (engl.), einem flexibleren Konzept der Klassifizierung.

Das Modell kann auch in höhere Dimensionen angewandt werden. Die Größe der Dimension des verwendeten Vektorraums wird festgelegt durch die Anzahl der Eigenschaften, die für jeden Datenpunkt gegeben sind (siehe Achsenbeschriftung in Abbildung 5). Die Dimension der Hyperebene ist dabei immer um eins geringer als die Dimension des Modells. Im dreidimensionalen Vektorraum hat die Hyperebene so die Gestalt einer Ebene, oder allgemein ausgedrückt: In der ‑ten Dimension hat die Hyperebene genau Dimensionen.

Oftmals sind die Datenpunkte eines Datensatzes aber nicht linear trennbar (siehe Abbildung 6). Bei der **nichtlinearen Klassifizierung** kommt der sogenannte Kernel-Trick zum Einsatz. Der Trick besteht darin, die Daten in einen höher dimensionalen Raum zu transformieren, in dem eine lineare Trennung (siehe lineare Klassifizierung) möglich ist. Die Problematik ist, dass die Berechnung der genauen Datenpunkte in der höheren Dimension sehr rechenlastig ist [11]. Für das Bestimmen der Hyperebene in der höheren Dimension werden diese genauen Datenpunkte aber gar nicht benötigt, vielmehr genügt es die Skalarprodukte der Datenpunkte zu kennen. Die Kernel-Funktion ist eine Funktion, welche die jeweiligen Skalarprodukte bereits aus den untransformierten Datenpunkte berechnen kann [13].

Es gibt verschiedene Arten von Kernel-Funktionen, wie z.B. lineare Kernel, polynomiale Kernel, Sigmoid Kernel oder die Gaußsche Radiale Basis-Funktion (RBF).



Abbildung 6: Nicht linear trennbare Daten eines Datensatzes mit zwei Eigenschaften pro Datenpunkt.

#### Neuronale Netze

Neuronale Netze, besser *künstliche neuronale Netze (KNN)*, bezeichnen eine Art von Modellen des maschinellen Lernens. KNN wurden ihrem Vorbild, dem menschlichen Gehirn, nachempfunden. Die Lernfähigkeit des menschlichen Gehirns resultiert daraus, dass in dem riesigen Netzwerk aus Neuronen und Synapsen, die Synapsen die Effektivität der Signalübertragung zwischen den Neuronen variieren können [14]. So werden neue synaptische Verbindungen geschaffen und andere getrennt oder vermindert. Ein einzelnes Neuron kann über Dendriten Signale anderer Neuronen empfangen, diese in dem Zellkörper verarbeiten und über die Axone ein resultierendes Signal an andere Neurone weiterleiten. Im Meschen existiert so ein riesiges Netz aus Neuronen und Synapsen.

Das einfachste Modell eines künstlichen Neurons wurde bereits 1943 von Warren McCulloch und Walter Pitts beschrieben [15]. Ihr Neuronenmodell folgt dem Alles-oder-nichts-Prinzip: Nur, wenn die Summe der Eingaben in das Neuron einen festen reellen Schwellenwert überschreitet, feuert das Neuron, d.h. es wird eine 1 weitergegeben. Bleibt der Schwellenwert unterschritten, wird eine 0 ausgegeben. Die Eingabe und Ausgabe sind stets binär. Mit Hilfe eines kleinen Netzes dieser Neuronen lassen sich so bereits einfache logische Verknüpfungen wie UND oder ODER darstellen.

1958 veröffentlichte Frank Rosenblatt das Modell des Perzeptrons [16]. Es beschreibt ein künstliches Neuron, bei dem nicht nur binäre Ein- und Ausgabewerte vorkommen. Außerdem kommt es zu einer Gewichtung der Eingabewerte, wodurch jeweils die Stärke des Einflusses der einzelnen Eingangssignale festgelegt ist. Ein einzelnes Neuron dieses Modells *(**einlagiges Perzeptron)* stellt bereits ein linearer Klassifikator dar. Es ist in Abbildung 7 dargestellt.



Abbildung 7: Einlagiges Perzeptron. (selbst gemacht)

Jedem Eingangssignal des Perzeptrons wird ein Gewicht mit zugeordnet. In der Eingangsfunktion wird die gewichtete Summe über die gesamte Eingabe gebildet. Zusätzlich wird auf diese Summe der Bias addiert. Der Bias gibt den Schwellenwert an, ab dem das Perzeptron feuern soll.

|  |  |  |
| --- | --- | --- |
|  |  | (1.1) |

Mithilfe einer Aktivierungsfunktion wird die Ausgabe des Perzeptrons bestimmt:

|  |  |  |
| --- | --- | --- |
|  |  | (1.2) |

Als Aktivierungsfunktion können verschiedene Funktionen Verwendung finden. Die Sigmoidfunktion findet häufig in modernen KNN Anwendung.

|  |  |  |
| --- | --- | --- |
|  |  | (1.3) |

Bei der Sigmoidfunktion handelt es sich um eine differenzierbare, nicht lineare Funktion. Sie führt, verglichen mit einer einfachen Schwellenwertsfunktion, zu einer weicheren Aktivierung des Neurons [17]. Das Modell des Perzeptrons bildet die Grundlage für KNN.

Mit einem einfachen einlagigen Perzeptron lässt sich aber noch keine XOR-Verknüpfung (exklusiv-ODER-Verknüpfung) darstellen. Um diese darstellen zu können wird ein dreilagiges Netz aus an Perzeptren benötigt. Ein solches ist in Abbildung 8 dargestellt.



Abbildung 8: Mehrlagiges XOR-Netz. Achtung, hier stellt jeder Kreis ein komplettes Perzeptron dar (wichtiger Unterschied zu Abbildung 7). X und y stehen für die Eingabeoperanden 1 (wahr) und 0 (falsch). Auf den Pfeilen ist je das Gewicht angegeben. Die Neuronen der verdeckten Schicht besitzen zudem einen Schwellenwert, welcher in dem jeweiligen Kreis angegeben ist. Allgemein ist der Schwellenwert nichts anderes als der negative Wert des oben erwähnten Bias. Z ist die Variable für den Ausgabewert.

Wenn die beiden Eingabewerte identisch sind, ergibt sich für die gewichtete Summe der Ausgabewerte der verdeckten Schicht (für ) oder (für ). In diesem Fall wird als z = 0 (falsch) ausgegeben. Nur wenn die beiden Eingabewerte sich unterscheiden, ist z = 1 (wahr) [18].

**Aufbau eines KNN:** Das in Abbildung 8 gezeigte mehrlagige Perzeptron ist ein KNN und besteht aus drei Schichten (eine Eingabeschicht, eine verdeckte Schicht und eine Ausgabeschicht). In der Eingabeschicht befinden sich Perzeptren, die von keinem anderem Perzeptron einen Input erhalten. Sie dienen der Netzeingabe. Die, sich an die Eingabeschicht anschließenden verdeckten Schichten eines KNNs (bei dem XOR-Netz ist das eine einzige), entsprechen den bereits oben beschriebenen einlagigen Perzeptren. Allgemein erhalten die Neurone der -ten verdeckten Schicht die Ausgabe der Neurone der -ten verdeckten Schicht als Eingabe. Für jede verdeckte Schicht wird nun mit (1.1) die gewichtete Summe aller verschiedenen Eingaben berechnet und anschließend mit der Aktivierungsfunktion (1.2) die Ausgabe bestimmt. Die Neurone der Ausgabeschicht geben den erhaltenen Wert aus.

Bei KNNs wird allgemein zwischen zwei verschiedenen Klassen unterschieden: Es gibt rekurrente neuronale Netze (engl. recurrent neural network, RNN) und vorwärtsgerichtete neuronale Netze (engl. feedforward neuronal network, FNN). Sie unterscheiden sich in der Verknüpfung der einzelnen Neurone. Bei dem RNN fließen die Signale stets nur in eine Richtung, während in einem FNN der Signalfluss in Zyklen vorliegt. D.h. es sind auch Verbindungen zu Neuronen derselben oder einer vorangegangenen Schicht möglich. Die heutzutage häufig vorkommende Bezeichnung tiefes Lernen (engl. deep learning), beschreibt ein KNN, das über viele verdeckte Schichten verfügt. Je größer die Anzahl der verdeckten Schichten und der Neuronen ist, desto besser können Muster erkennt werden, oder Objekte klassifiziert werden. So basiert z.B. auch der zu Beginn der Einleitung beschriebene GO-Computer auf tiefen neuronalen Netzen.

**Trainieren von KNN:** Generell ist es wichtig, eine genügend große Menge an Trainingsdaten zu haben, um ein Modell für das tiefe Lernen erfolgreich zu trainieren [19]. Das Training kann auch als Optimierungsproblem aufgefasst werden, bei dem die Gewichte der einzelnen Neurone so verändert werden sollen, dass der Fehler bei der Klassifizierung möglichst gering wird. Es erfolgt häufig mit dem sogenannten Gradientenabstiegsverfahren (engl. backpropagation). Gegeben ist ein Datensatz mit wobei für die Eingabewerte und für die gewünschten Ausgabewerte stehen. Die tatsächliche Ausgabe ergibt sich mit (1.2). Der quadratische Fehler berechnet dann sich folgendermaßen:

|  |  |  |
| --- | --- | --- |
|  |  | (1.4) |

Mit Hilfe der partiellen Ableitungen von lässt sich die Gewichtsänderung für jedes…..

Die Lernrate gibt an wie stark die Gewichte verändert werden sollen.

Unterscheiden zu den hidden und ein satz dass es backwards ist also von outout nach input (schritte auf das perzeptron oder neuronalke netze ulm)

AM Besten MIT DAS PERZEPTRON ARBEIOTEN UND WIKI !!!!!!!!!!!!

Back Propagation und Delta Regel Training nichts anderes als bias und gewichte zu finden (gute) ; am anfang zufällig mithilfe von average cost (je größer desto schlechter)

zur

**DL bild**

<https://www5.in.tum.de/lehre/vorlesungen/mod_sim/SS06/ModSim_024>

<https://www.informatik.uni-kiel.de/inf/Sommer/doc/Downloads/Skripte/neuroskript.pdf>

<http://ml.informatik.uni-freiburg.de/former/_media/teaching/ss13/ml/sperschneider/08perzeptron.pdf>

<http://ml.informatik.uni-freiburg.de/former/_media/teaching/ss13/ml/sperschneider/08perzeptron.pdf>

<http://www.mathematik.uni-ulm.de/stochastik/lehre/ss07/seminar_sl/ausarbeitung_wallner.pdf>

http://ml.informatik.uni-freiburg.de/former/\_media/teaching/ss10/05\_mlps.printer.pdf

https://data-science-blog.com/blog/2017/12/20/maschinelles-lernen-klassifikation-vs-regression/

https://developers.google.com/machine-learning/crash-course/introduction-to-neural-networks/anatomy

## Finden neuer Wirkstoffe im Zeitalter von Big Data ~~KI in der Pharmazeutischen Bioinformatik~~

### klassischer Weg

Das Finden und Entwickeln von neuen Wirkstoffen ist in der heutigen Zeit ein Prozess, welcher Jahre dauert und eine große Menge Geld benötigt. Ein Artikel beziffert die durchschnittlichen Kosten für die Entwicklung und Forschung eines neuen Wirkstoffes auf 985 Millionen US-$ [20]. Wenn ein Zielmolekül (meist ein Protein) im Körper identifiziert wurde, geht es darum eine passende Leitstruktur zu finden, welcher mit dem Zielmolekül interagieren kann. Dafür kommt seit Mitte der 90er Jahre das Hochdurchsatz-Screening zum Einsatz, bei dem große Molekülbibliotheken mittels automatisierten in vitro-Testsystemen durchmustert werden. Um den fertigen Wirkstoff zu erhalten muss die gefundene Leitstruktur hinsichtlich ihrer pharmakologischen und physikochemischen Eigenschaften optimiert werden. Bei dem fertigen Wirkstoff selbst handelt es sich meistens eine niedermolekulare Verbindung (engl. small molecule), d.h. eine chemische Verbindung mit einem geringen Molekulargewicht. Die Verbindungen greifen an Zielmolekülen wie Enzymen, Rezeptoren oder Ionenkanälen an und erzielen so eine Wirkung auf den Organismus.

### virtuelles Screening

<https://pubmed.ncbi.nlm.nih.gov/19881490/>

paper dass zeigt das CPI pairs wichtig für discovery and development of safe and effective drugs

Auf dem Weg zum fertigen Wirkstoff stellt das Hochdurchsatz-Screening einen zeitaufwendigen und zeitintensiven Prozess dar [21]. Angesichts der riesigen Größe der Molekülbibliotheken, ist es von Interesse der Pharmabranche, die für eine im Hochdurchsatz‑Screening getestete Substanz anfallenden Kosten von 2 bis 5 US-$, zu minimieren. Daher findet das virtuelle Screening, basierend auf computergestützten Techniken, immer mehr an Bedeutung. Dockingprogramme können die Moleküle der Datenbank als 3D-Modelle in die Bindetasche des Proteins legen und mithilfe einer Scoring-Funktion deren zu erwartende Bindungsaffinität bestimmen. So wird die Zahl der möglichen Wirkstoffkandidaten erheblich reduziert und kann mit kostenintensiveren biologischen Tests weiter reduziert werden.

#### Datensätze als Grundlage

Die Grundlage des virtuellen Screenings sind riesige Datensätze mit Informationen zu der Interaktion von kleinen chemischen Verbindungen mit Proteinen (engl. compound-protein interaction; CPI). Solche Informationen sind bereits in diversen Moleküldatenbänken, wie ChEMBL, DrugBank oder PDBbind gesammelt [22] [23] [24]. Diese Daten haben ihren Ursprung in der riesigen Anzahl an wissenschaftlichen Arbeiten der letzten Jahre. 2018 wurden auf der gesamten Welt insgesamt ca. 2,5 Millionen neue wissenschaftliche Arbeiten publiziert [25]. Angesichts dieser riesigen und immer größer werdenden Zahl ist es mit einem sehr großen Aufwand verbunden, immer die aktuellen Daten aus den wissenschaftlichen Arbeiten in die Datenbänke zu integrieren.

#### Einsatz von Text Mining und NLP

Der Begriff Text Mining beschreibt die Menge an Techniken zum automatischen Extrahieren von neuen, zuvor unbekannten Informationen aus unstrukturierten Texten [26]. Diese Techniken verwenden häufig auf maschinellem Lernen basierende Algorithmen. In diesem Zusammenhang fällt auch der Begriff der Computerlinguistik (engl. natural language processing; NLP). Er bezeichnet die Verarbeitung der natürlichen Sprache mit einem Computer. Bei NLP wird die Sprache zuerst in kleine sprachlich relevante Einheiten (Token) zerlegt, bevor mit Analysen der Morphologie, Syntax und Semantik die Sprache für die Maschine zunehmend erfassbarer wird [26].

**Die Kombination von Text Mining mit NLP stellt eine gute Möglichkeit dar, die für das virtuelle Screening benötigten Daten der CPI direkt aus wissenschaftlichen Arbeiten zu extrahieren.**

In dieser Arbeit werden zwei auf maschinellem Lernen basierende Methoden zum Text Mining von CPI verglichen. In Kapitel 1.3 befindet sich ein Überblick über die Entwicklung von CPI Text Mining in der Wissenschaft.

## CPI Text Mining

Im CPI Text Mining geht es, wie in 1.2 beschrieben, um die Extraktion der Informationen zu einer möglichen Interaktion zwischen einer kleinen chemischen Verbindung und einem Protein (CPI-Paar) aus wissenschaftlichen Arbeiten. Die interessante Information ist dabei, ob in dem Satz eine funktionale Beziehung des CPI-Paars beschrieben wird oder nicht (siehe Abbildung 9).



Abbildung 9: Beispielsätze mit CPI-Paaren aus wissenschaftlichen Arbeiten. Die vorhandenen Proteine sind mit blau und die kleinen chemischen Verbindungen mit rot hinterlegt. (1) Beispielsatz: Hier besteht eine funktionale Beziehung (d.h. Interaktion) zwischen dem Protein „insulin“ und der Verbindung „dithiothritol“. (2) Beispielsatz mit zwei CPI-Paaren. In beiden besteht keine funktionale Beziehung zwischen dem Protein „BDNF“ und den beiden Verbindungen „MHPG“ und „HVA“.

Bei dem CPI Text Mining handelt es sich um eine konkrete Anwendung von allgemeinem Beziehungs‑Text Mining (engl. relation extraction; RE). Weitere mögliche extrahierte Beziehungen sind z.B. Protein-Protein Interaktionen (eng. protein-protein interactions), Gen-Krankheit Zusammenhänge (engl. gene-disease associations) und chemische Verbindungen‑Gen Interaktionen (engl. compound-gene interaction). Allgemein werden die einzelnen Objekte (Protein, Gen, Krankheit und chemische Verbindungen) auch als Entitäten bezeichnet. Wenn in einer wissenschaftlichen Arbeit ein Modell zu einer konkreten RE Aufgabe publiziert wird, lässt sich dieses mit kleinen Adaptionen auf weitere Beziehungen anwenden. Bei den auf maschinelles Lernen basierenden Modellen muss aber auf jeden Fall ein neuer passender Trainingsdatensatz für das Modell vorhanden sein.

### Historie von RE-Text Mining mit Blick auf CPI-Text Mining

Zum RE aus der Literatur gibt es grundlegend verschiedene Ansätze. In XXXXXXX sind ausgewählte Arbeiten zum RE-Text Mining grafisch dargestellt.

**Dann hier die timeline** [C:\Users\Manuel Dorer\Desktop\BA\_win\schreiben\Timeline.pptx](file:///C:\Users\Manuel%20Dorer\Desktop\BA_win\schreiben\Timeline.pptx)

Die Basis für die ersten Arbeiten ist die Kookkurrenz, bei der sobald zwei Entitäten gemeinsam in einem Abschnitt des Textes (z.B. ein Satz), vorkommen, eine Interaktion zwischen diesen angenommen wird [27]. Die Präzision kann, durch die zusätzliche Anwesenheit eines Interaktionswortes zwischen dem Entitäten-Paar [28], oder den Abgleich mit lexikalisch‑syntaktischen Mustern [29], verbessert werden.

Ein weiterer Ansatz ist, unter Verwendung von NLP die Entitäten‑Paare anhand von selbst definierten Regeln in den Sätzen zu identifizieren. Anhand von typischen Mustern an Entitäten-Paaren in der Literatur, werden diese Regeln abgeleitet. Die Muster können hierfür entweder manuell [30] oder automatisiert [31] gefunden werden. Mit den Muster‑basierten‑Ansätzen zum RE-Text Mining, lassen sich relativ gut einfache Interaktionen in Sätzen erfassen. Aber bei in der Wissenschaft üblichen komplexeren Sätzen, gerät dieser Ansatz schnell an seine Grenzen [32].

Aus diesem Grund werden Klassifikatoren, die an gelabelten Daten trainierte Modelle des überwachten maschinellen Lernens verwenden, immer interessanter. In den ersten Arbeiten mit diesem Ansatz werden z.B. Bayes’sche Netze [33] oder maximale‑Entropie‑basierte‑Methoden [34] verwendet. Außerdem wird der Einsatz von Kern-Methoden-basierte Modellen immer beliebter. *Tikk et al.* vergleicht 13 publizierte Kern‑Methoden zur Extraktion von Protein-Protein Interaktionen [35]. Zwei dieser Methoden, welche je zu guten Ergebnissen geführt hatten, werden in der Arbeit von *Döring K, Qaseem A, Becer M, Li J, Mishra P, Gao M, et al.* zur Extraktion von CPI‑Paaren aus der Literatur verwendet [36] (siehe. *Automated recognition of functional compound-protein relationships in literature (2020)*). Erst seit den letzten Jahren werden als Klassifikatoren verschiedene Arten von Modellen des tiefen Lernens verwendet, z.B. RNN [37]. Sehr richtungsweisend für NLP ist die Entwicklung eines vortrainierbaren allgemeinen Sprachmodels wie BERT [38] oder ELMO [39], welche unter anderem zu Text Mining verwendet werden können. Das erfolgreiche Spezialisieren von BERT auf biomedizinische Texte gelingt 2019 [40] (siehe. *BioBERT: a pre-trained biomedical language representation model for biomedical text mining 2019)*).

Im Folgenden werden die beiden für diese Bachelorarbeit wichtigen wissenschaftlichen Arbeiten [36], [40] näher beschrieben.

#### Automated recognition of functional compound-protein relationships in literature (2020)

In dieser Arbeit von *Döring K, Qaseem A, Becer M, Li J, Mishra P, Gao M, et al.* aus dem Jahr 2020 [36] wurden die beiden Kern-Methoden *all-paths graph* und *shallow linguistic* mittels eines eigen erstellten Datensatzes kreuzvalidiert und anschließend zum CPI‑Text Mining aus den Titeln und Abstracts aller vor Juli 2019 veröffentlichen PubMed Artikel verwendet. Die Hauptidee dieser Kern‑Methoden ist, die Ähnlichkeit der beiden Entitäten anhand der Ähnlichkeiten in der Unterstruktur des Satzes zu bestimmen. Bei der *all‑paths graph* Kernmethode wird ein Abhängigkeits-Graph der Satzstruktur verwendet, um diese Ähnlichkeiten mit Hilfe von gewichteten Pfaden zu bestimmen. Der Kern der *shallow linguistic* Kernmethode setzt sich als Summe aus zwei Kernen zusammen, welche aber nur einen kleinen Bereich an Wörtern um die Entitäten berücksichtigen. Die resultierende Ähnlichkeit des Entitätenpaars ergibt sich als Skalarprodukt der Ergebnisvektoren beider Kerne. Die Ergebnisse der Kreuzvalidierung sind in Tabelle 1 dargestellt. Die Kombination der beiden Kerne führt zu einem leicht besseren Ergebnis.

Tabelle 1: Ergebnisse der Kreuzvalidierung [36]. Alle Angaben sind in Prozent. Es sind nur die Ergebnisse mit den jeweils besten Parametern angegeben.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Kernmethode** | **Sensitivität** | **Spezifität** | **Genauigkeit** | **F-Maß** | **AUC** |
| *all-paths graph* | 81.7 | 71.8 | 76.6 | 79.0 | 84.6 |
| *shallow linguistic* | 79.5 | 70.2 | 75.0 | 77.2 | 82.5 |
| kombiniert | 68.5 | 81.6 | **80.5** | 74.0 | - |

Der verwendete Datensatz wurde aus den Abstracts der 40000 ersten im Jahr 2009 veröffentlichten PubMed Artikel gebildet und anschließend von Hand mit den Labeln „True“ für eine bestehende funktionale Interaktion oder „False“ für keine funktionale Interaktion versehen. **Die binäre Labelung ist, mit bereits existierenden CPI-Datensätzen verglichen, einzigartig.** So gibt es z.B. in dem häufig verwendeten BioCreativ VI Datensatz [41] 10 verschiedene Label, welche die Art der CPI-Interaktion näher beschreiben.

In dieser Bachelorarbeit soll, wie in der „Conclusion“ [36] beschrieben, ein Vergleich der Kern-Methoden mit auf tiefen Lernen basierenden Ansätzen, unter Verwendung desselben Datensatzes erfolgen.

#### BioBERT: a pre-trained biomedical language representation model for biomedical text mining (2019)

BioBERT ist speziell trainierte Version von BERT für das NLP von biomedizinischen Texten. Daher entsprechen sich die Aufbauten der beiden Modelle. In BERT werden Transformer ausgiebig genutzt, was, verglichen mit den in ELMO verwendeten *long short-term memories* (bestimmte Art von RNN), zu einer besseren Effizienz führt. Reine Transformer bestehen aus einem Kodierer, zum Lesen des Inputtextes und einem Dekodierer für das Bearbeiten der bestimmten NLP‑Aufgabe. Das Training von Transformern, ist auf die beiden Prozesse Vortraining *(engl. pre‑training)* und Feintuning *(engl. finetuning)* aufgeteilt. Für das Vortraining von BERT (siehe Abbildung 10) werden zwei Trainingsverfahren gleichzeitig durchgeführt:

1. Bei dem **MLM *(masked language modeling)*** maskiert das Modell eine Zeichenfolge innerhalb eines als Input erhaltenen Satzes und lernt dieses vorherzusagen.
2. Bei dem **NSP *(next sentence prediction)*** bekommt das Modell zwei Sätze als Input und soll lernen zu prognostizieren, ob der zweite Satz auf den ersten folgt.

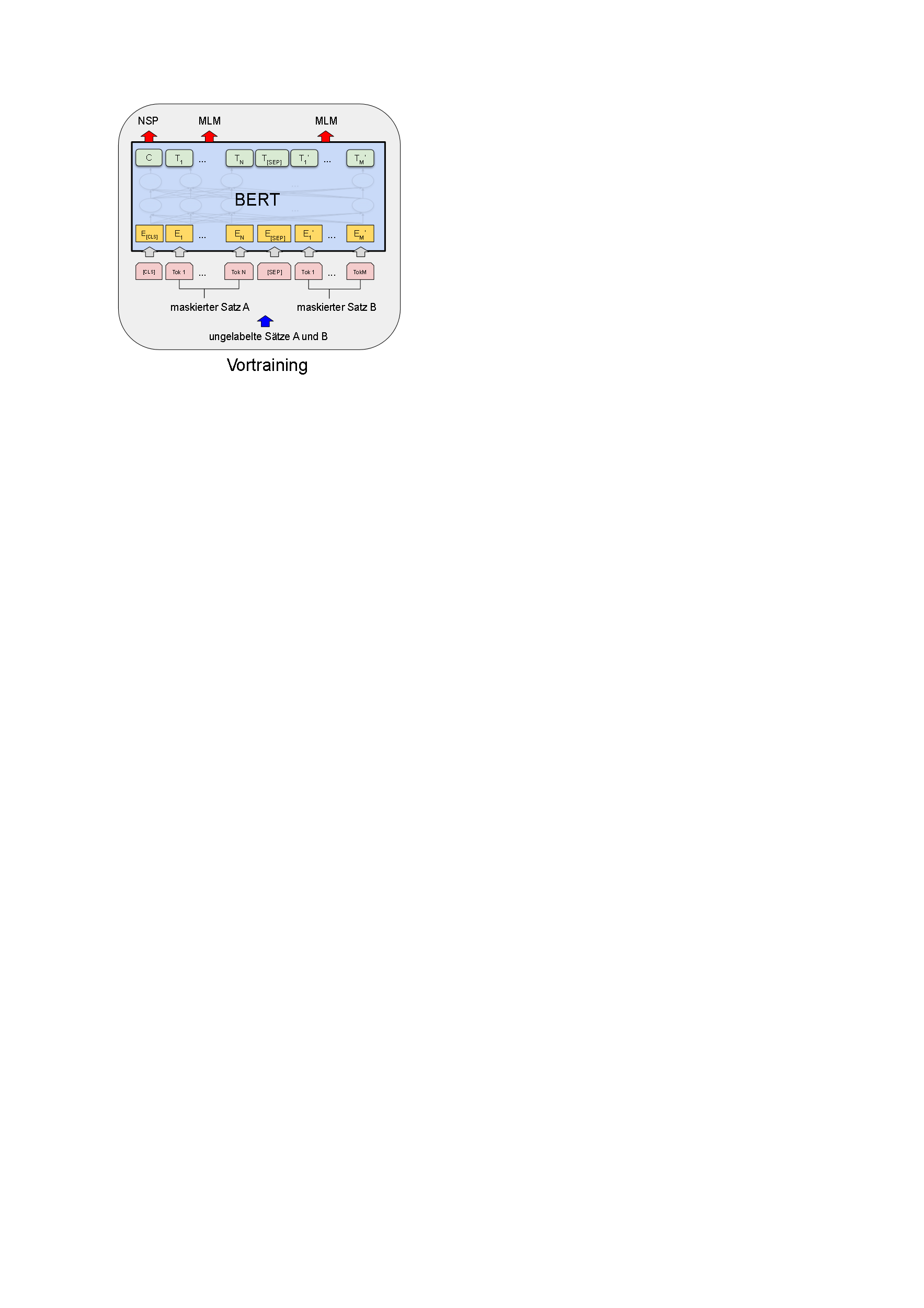


Abbildung 10: Schemata zum Vortraining von BERT. Die beiden Sätze A und B in die Token aufgeteilt und mache Token maskiert. Die Token werden anschließend mittels Worteinbettung in eine Vektorform überführt und diese dem Kodierer für NSP und MLM zugeführt. Als Output gibt das neuronale Netz die binärere Variable als Ergebnis des NSP und als Vorhersage des maskierten Tokens an.

Für dieses Vortraining werden riesige Datenmengen ungelabelter Sätze benötigt. Bei BERT wurde hierfür ein Datensatz aus über 11000 Büchern (800 Millionen Wörter), sowie das englische Wikipedia (2,5 Milliarden Wörter) verwendet. Bei BioBERT wurden zusätzlich, je nach Ausführung, noch die PubMed Abtracts (4,5 Milliarden Wörter) oder die gesamten Artikel der PMC‑Datenbank (13,5 Milliarden Wörter) verwendet. Als Ergebnis werden in beiden Arbeiten je die vortrainierten Sprachmodelle, welche gelernt haben, den Kontext eines Wortes in seiner Umgebung zu verstehen, veröffentlicht.

Beim Finetuning wird das Sprachmodell mit einem gelabelten Datensatz auf eine konkrete Aufgabe, wie z.B. dem CPI­‑Text Mining, trainiert. Dies wird in dem Kapitel Material und Methoden beschrieben.

# Zielsetzung der Arbeit

Diese Bachelorarbeit soll, wie in der „Conclusion“ [36] beschrieben, als Vergleich von Kern-Methoden mit auf tiefes Lernen basierende Ansätze dienen. Das verwendete Modell des tiefen Lernens ist dabei das vortrainierte BioBERT-Modell [40]. Um eine Vergleichbarkeit zu erreichen, wird der identische Datensatz verwendet. Es ergeben sich folgende Zwischenziele:

1) Überführen des Datensatzes in die notwendige Form

2) Evaluieren des BioBERT-Modells über eine 10-fache Kreuzvalidierung mit dem fertigen Datensatz

3) Bestimmen des Einflusses der Größe des verwendeten Trainingsdatensatzes auf die Klassifikation

# Material und Methoden

Beschreibung was in welchem teil hier gemacht

In 3.1 Ausgangsbedingungen dargelegt…………….

## Grundlagen

### Systemeigenschaften

Betriebssystem: Ubuntu 18.04.5 LTS (Bionic Beaver)[[1]](#footnote-1)

Prozessor: Intel® Core™ i5-6600K CPU @ 4.20GHz

Grafikkarte: NVIDIA® GeForce® GTX 1060 6GB

Programmiersprache: Python 3.6.9

### Verwendete Software

Bei dem verwendeten Modell handelt es sich um BioBERT-Base v1.1 (+ PubMed 1M). Dieses stammt zusammen mit der Software für das Finetuning von dem offiziellen GitHub BioBERT Repository[[2]](#footnote-2). Für diese Arbeit wurden die Skripte an wenigen Stellen ergänzt, bzw. neue Skripte hinzugefügt. Zudem wurde die Ordnerstruktur leicht verändert. Jede Änderung wird an der entsprechenden Stelle in dieser Arbeit angegeben. Die Software, wie sie Letzen Endes zum Einsatz kam, ist auf GitHub in einem eigenen Repository verfügbar.[[3]](#footnote-3)

Als Framework der Arbeit dient die TensorFlow Bibliothek, welche von der Google Brain Abteilung stammt, aber als Open Source Projekt weiterentwickelt wird. TensorFlow ist in Python und C++ geschrieben. Die Verwendung von BioBERT mit Pytorch, einer weiteren bekannten Open Source Bibliothek für maschinelles Lernen, ist ebenfalls möglich[[4]](#footnote-4).

Grundsätzlich wird für die Arbeit mit BioBert eine virtuelle Umgebung verwendet. Das bringt den Vorteil mit sich, einen Überblick über die installierten Python-Pakete zu bewahren und diese besser organisieren zu können. Alle benötigten Pakte, mit den genauen in dieser Arbeit verwendeten Versionen, können dem File requirements.txt entnommen werden und nach dem Klonen des GitHub Repositorys wie folgt installiert werden:

$ git clone https://github.com/mmmddd98/ba\_git

$ cd ba\_git; pip install -r requirements.txt

### Verwendeter Datensatz

CPI‑DS.xml

XML-format

Ein bild von Auschnitt DS himachen und kurz erkären was was ist

[Haupteil\CPI-DS\_ausschnitt.txt](Haupteil/CPI-DS_ausschnitt.txt)

Dann noch zu Größe tabelle aus freiburg

Tabelle 2: Inhalt des CPI-DS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Datensatz** | **Anz. Artikel** | **Anz. Sätze** | **Anz. funk. CPIs** | **Anz. keine funk. CPI** | **Ges. Anz. CPI** |
| CPI‑DS | 1808 | 2613 | 2931 | 2631 | 5562 |

### Statistische Parameter zur Bewertung von Modellen

Erst tabelle mit TP TN….

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Kernmethode** | **Sensitivität** | **Spezifität** | **Genauigkeit** | **F-Maß** | **AUC** |
| *all-paths graph* | 81.7 | 71.8 | 76.6 | 79.0 | 84.6 |
| *shallow linguistic* | 79.5 | 70.2 | 75.0 | 77.2 | 82.5 |
| kombiniert | 68.5 | 81.6 | **80.5** | 74.0 |  |

https://www.jeremyjordan.me/evaluating-a-machine-learning-model/

Siehe [..\ProceedingsBCVI\_v2\_seite 140.pdf](file:///C:\Users\Manuel%20Dorer\Desktop\BA_win\ProceedingsBCVI_v2_seite%20140.pdf)

Oder doktorarbeit von freidok s 39

Oder ba ml

#### Sensitivität = Recall

|  |  |  |
| --- | --- | --- |
|  |  | (3.1) |

#### Spezifität = Specifity

|  |  |  |
| --- | --- | --- |
|  |  | (3.2) |

#### Genauigkeit = precision

|  |  |  |
| --- | --- | --- |
|  |  | (3.3) |

#### F-Maß = F1 score

|  |  |  |
| --- | --- | --- |
|  |  | (3.4) |

#### AUC = AUC (Area under Curve)

## Datensatzpräparation

Allgemein mit pipelin wie in git

Scripte teilweise erkären

## Vergleich (10-fache Kreuzvalidierung)

### 10-fache Kreuzvalidierung … was ist das

<https://link.springer.com/chapter/10.1007/978-3-319-47217-1_9>

### splitten des DS

### Durchführen der 10cv val

## Untersuchung des Einflusses der Trainingsdatensatzgröße

### splitten des DS

### Durchführung

# Ergebnisse

* Gleicher split wie in fr paper zum dir vergleich
* Validation zum Sagen das Hyperparameter okay waren aber auch zum vergleich nutzen
* Veränderung der DS größe um

# Diskussion und Ausblick

Overfitting underfitting?

Ml paper generell fig 1b performance kernek´l methiden schlechter als dl

[einleitung\einleitung\_paper\ML\_paper\_generell.pdf](file:///C:\Users\Manuel%20Dorer\Desktop\BA_win\schreiben\einleitung\einleitung_paper\ML_paper_generell.pdf)

hyperparamter optimierung noch was rausholen

pytoech vs tensorflow (sollte eig nix machen aber vlt überprüfen)

ML andere vs dl <https://machinelearningmastery.com/what-is-deep-learning/>

<https://arxiv.org/pdf/2101.01907.pdf> dl ganz neues paper überblick

cuda

Datensätze nicht identisch wie in Freiburg 🡺 paar rausgeworfen

Vlt risiko mit tensorflow kostenpflichtigkeit ansprechen

# Zusammenfassung

# Literaturverzeichnis

[1] “Google-Software besiegt Go-Genie auch im letzten Match.” https://www.faz.net/aktuell/gesellschaft/menschen/google-computer-alphago-besiegt-go-weltmeister-14125664.html (accessed Feb. 15, 2021).

[2] D. Silver *et al.*, “Mastering the game of Go with deep neural networks and tree search,” *Nature*, vol. 529, no. 7587, pp. 484–489, 2016, doi: 10.1038/nature16961.

[3] S. Rasp, P. D. Dueben, S. Scher, J. A. Weyn, S. Mouatadid, and N. Thuerey, “WeatherBench: A Benchmark Data Set for Data-Driven Weather Forecasting,” *J. Adv. Model. Earth Syst.*, vol. 12, no. 11, 2020, doi: 10.1029/2020MS002203.

[4] A. El Sallab, M. Abdou, E. Perot, and S. Yogamani, “Deep reinforcement learning framework for autonomous driving,” *arXiv*, pp. 70–76, 2017.

[5] E. V. Polyakov, M. S. Mazhanov, A. Y. Rolich, L. S. Voskov, M. V. Kachalova, and S. V. Polyakov, “Investigation and development of the intelligent voice assistant for the Internet of Things using machine learning,” *Moscow Work. Electron. Netw. Technol. MWENT 2018 - Proc.*, vol. 2018-March, pp. 1–5, 2018, doi: 10.1109/MWENT.2018.8337236.

[6] A. Esteva *et al.*, “Dermatologist-level classification of skin cancer with deep neural networks,” *Nature*, vol. 542, no. 7639, pp. 115–118, 2017, doi: 10.1038/nature21056.

[7] I. Döbel *et al.*, “Maschinelles Lernen: Kompetenzen, Forschung, Anwendung,” *Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e.V.*, 2018. https://www.bigdata-ai.fraunhofer.de/content/dam/bigdata/de/documents/Publikationen/Fraunhofer\_Studie\_ML\_201809.pdf.

[8] R. Fjelland, “Why general artificial intelligence will not be realized,” *Humanit. Soc. Sci. Commun.*, vol. 7, no. 1, pp. 1–9, 2020, doi: 10.1057/s41599-020-0494-4.

[9] “How To Design A Spam Filtering System with Machine Learning Algorithm.” https://towardsdatascience.com/email-spam-detection-1-2-b0e06a5c0472 (accessed Feb. 20, 2021).

[10] “Verhindern von Überanpassung und unausgeglichenen Daten durch automatisiertes maschinelles Lernen.” https://docs.microsoft.com/de-de/azure/machine-learning/concept-manage-ml-pitfalls (accessed Feb. 22, 2021).

[11] “Support Vector Machines (SVM); Universität Ulm.” http://www.mathematik.uni-ulm.de/stochastik/lehre/ss07/seminar\_sl/fischer.pdf (accessed Feb. 24, 2021).

[12] “Support Vector Machines (SVM) Ausarbeitung; Universität Ulm.” http://www.mathematik.uni-ulm.de/stochastik/lehre/ss07/seminar\_sl/ausarbeitung\_fischer.pdf (accessed Feb. 24, 2021).

[13] “The Kernel Trick in Support Vector Classification.” https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f (accessed Feb. 24, 2021).

[14] “Das Gehirn,” *MAX-PLANCK-GESELLSCHAFT*. https://www.mpg.de/gehirn (accessed Feb. 26, 2021).

[15] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity (reprinted from bulletin of mathematical biophysics, vol 5, pg 115-133, 1943),” *Bull. Math. Biol.*, vol. 52, no. 1--2, pp. 99–115, 1990, [Online]. Available: http://journals2.scholarsportal.info/pdf/00928240/v52i1-2/99\_alcotiiina.xml.

[16] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain,” *Psychol. Rev.*, vol. 65, no. 6, pp. 386–408, 1958, doi: 10.1037/h0042519.

[17] A. Scherer and A. Scherer, “Das Perzeptron,” *Neuronale Netze*, pp. 65–70, 1997, doi: 10.1007/978-3-322-86830-5\_5.

[18] “Neuronale Netze; Univerität Ulm.” .

[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2014.

[20] O. J. Wouters, M. McKee, and J. Luyten, “Estimated Research and Development Investment Needed to Bring a New Medicine to Market, 2009-2018,” *JAMA - J. Am. Med. Assoc.*, vol. 323, no. 9, pp. 844–853, 2020, doi: 10.1001/jama.2020.1166.

[21] G. Klebe, *Wirkstoffdesign*, 2.Auflage. Spektrum Akademischer Verlag, 2009.

[22] D. Mendez *et al.*, “ChEMBL: Towards direct deposition of bioassay data,” *Nucleic Acids Res.*, vol. 47, no. D1, pp. D930–D940, 2019, doi: 10.1093/nar/gky1075.

[23] D. S. Wishart *et al.*, “DrugBank 5.0: A major update to the DrugBank database for 2018,” *Nucleic Acids Res.*, vol. 46, no. D1, pp. D1074–D1082, 2018, doi: 10.1093/nar/gkx1037.

[24] R. Wang, X. Fang, Y. Lu, C. Y. Yang, and S. Wang, “The PDBbind database: Methodologies and updates,” *J. Med. Chem.*, vol. 48, no. 12, pp. 4111–4119, 2005, doi: 10.1021/jm048957q.

[25] “Scientific and technical journal articles.” https://data.worldbank.org/indicator/IP.JRN.ARTC.SC?year\_low\_desc=true (accessed Mar. 04, 2021).

[26] D. R. Witte and J. Mülle, “Text Mining: Wissensgewinnung aus natürlichsprachigen Dokumenten,” 2005.

[27] T. Jenssen, J. Komorowski, and E. Hovig, “A literature network of human genes for high-throughput analysis of gene,” vol. 28, no. may, pp. 21–28, 2001.

[28] R. Kabiljo, A. B. Clegg, and A. J. Shepherd, “A realistic assessment of methods for extracting gene / protein interactions from free text,” vol. 12, pp. 1–12, 2009, doi: 10.1186/1471-2105-10-233.

[29] F. Rinaldi *et al.*, “Open Access OntoGene in BioCreative II,” vol. 9, no. Suppl 2, pp. 1–11, 2008, doi: 10.1186/gb-2008-9-S2-S13.

[30] J. M. Temkin and M. R. Gilder, “Extraction of protein interaction information from unstructured text using a context-free grammar,” *Bioinformatics*, vol. 19, no. 16, pp. 2046–2053, 2003, doi: 10.1093/bioinformatics/btg279.

[31] Y. Hao, X. Zhu, M. Huang, and M. Li, “Discovering patterns to extract protein-protein interactions from the literature: Part II,” *Bioinformatics*, vol. 21, no. 15, pp. 3294–3300, 2005, doi: 10.1093/bioinformatics/bti493.

[32] I. Segura-Bedmar, P. Martínez, and C. de Pablo-Sánchez, “Using a shallow linguistic kernel for drug-drug interaction extraction,” *J. Biomed. Inform.*, vol. 44, no. 5, pp. 789–804, 2011, doi: 10.1016/j.jbi.2011.04.005.

[33] R. Chowdhary, J. Zhang, and J. S. Liu, “Bayesian inference of protein-protein interactions from biological literature,” *Bioinformatics*, vol. 25, no. 12, pp. 1536–1542, 2009, doi: 10.1093/bioinformatics/btp245.

[34] C. Sun, L. Lin, X. Wang, and Y. Guan, “Using maximum entropy model to extract protein-protein interaction information from biomedical literature,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4681 LNCS, no. August, pp. 730–737, 2007, doi: 10.1007/978-3-540-74171-8\_72.

[35] D. Tikk, P. Thomas, P. Palaga, J. Hakenberg, and U. Leser, “A comprehensive benchmark of kernel methods to extract protein-protein interactions from literature,” *PLoS Comput. Biol.*, vol. 6, no. 7, p. 32, 2010, doi: 10.1371/journal.pcbi.1000837.

[36] K. Döring *et al.*, “Automated recognition of functional compound-protein relationships in literature,” *PLoS One*, vol. 15, no. 3, pp. 1–14, 2020, doi: 10.1371/journal.pone.0220925.

[37] D. Zhang and D. Wang, “Relation Classification via Recurrent Neural Network,” 2015, [Online]. Available: http://arxiv.org/abs/1508.01006.

[38] C. Sun *et al.*, “Chemical-protein interaction extraction via Gaussian probability distribution and external biomedical knowledge,” *Bioinformatics*, vol. 36, no. 15, pp. 4323–4330, 2020, doi: 10.1093/bioinformatics/btaa491.

[39] M. E. Peters *et al.*, “Deep contextualized word representations,” *NAACL HLT 2018 - 2018 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.*, vol. 1, pp. 2227–2237, 2018, doi: 10.18653/v1/n18-1202.

[40] J. Lee *et al.*, “BioBERT: A pre-trained biomedical language representation model for biomedical text mining,” *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 2020, doi: 10.1093/bioinformatics/btz682.

[41] “ChemProt corpus: BioCreative VI.” https://biocreative.bioinformatics.udel.edu/media/store/files/2017/ChemProt\_Corpus.zip (accessed Mar. 08, 2021).

1. Zuerst wurde in einer VirtualBox unter Windows 10 begonnen, dann mittels einer neuen SSD ein Dual‑Boot‑System mit Ubuntu eingerichtet. [↑](#footnote-ref-1)
2. Link: https://github.com/dmis-lab/biobert [↑](#footnote-ref-2)
3. Link: https://github.com/mmmddd98/ba\_git [↑](#footnote-ref-3)
4. Link: https://github.com/dmis-lab/biobert-pytorch [↑](#footnote-ref-4)