Recognition of Functional Relationships between Biomedical Concepts in the Scientific Literature using Text Mining and Machine Learning

DISSERTATION

submitted for the degree

Doctor rerum naturalium (Dr. rer. nat.)

at the

Faculty of Chemistry and Pharmacy

University of Freiburg

by

Ammar Qaseem

from Aden, Yemen

2023

Ammar Qaseem:

Recognition of functional relationships between biomedical concepts in the scientific literature using text mining and machine learning, 2023.

Chairman of the Doctoral Committee:	Prof. Dr. Stefan Weber
Dean of the Faculty:	Prof. Dr. Andreas Bechthold
1st Supervisor:	Prof. Dr. Stefan Günther
2nd Supervisor:	Prof. Dr. Rolf Backofen
Examiner:	Prof. Dr. Andreas Bechthold
Oral Examination:	7th June 2023

"When the human being dies, his deeds end except for three: ongoing charity, beneficial knowledge, or a righteous child who prays for him (for the deceased)."

- Prophet Muhammad (PBUH) (571-633)

Declaration

I hereby declare that the work presented in this thesis has not been submitted for any other degree or professional qualification and that it is the result of my own independent work.

Ammar Qaseem

Freiburg, January 2023

Acknowledgments

First and foremost, I would like to thank Allah the Almighty for blessing me and granting me countless knowledge and opportunities throughout my life and study.

A very special thank to my family in Yemen: my parents and my brothers, for supporting me spiritually throughout my life and study. Your support is the reason for being in this position. I want to express my gratitude to my wife **Amani** for her love, support, and patience during my study. Many thanks to my little daughters **Joud** and **Jamila** and apologies to them at the same time for being away from them most of the time and not spending much time with them during my studies.

Many thanks to **Prof. Dr. Stefan Günther**, who offered me the opportunity to accomplish my Ph.D. thesis. I would like to express my sincere gratitude to him for the continuous support of my thesis study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me a lot during my thesis work. I would like to thank **Prof. Dr. Rolf Backofen**, who was my supervisor during my Master's study and was my co-supervisor in my Ph.D. My thanks to **Prof. Dr. Andreas Bechthold** for being my second co-supervisor.

Also, I would like to thank all the group of Pharmaceutical Bioinformatics (PhaBi) for the great times which we spent together. I would like to thank all members of the group: **Dr. Aurélien F. A. Moumbock**, **Dr. Kersten Döring**, **Dr. Mehrosh Pervaiz**, **Dr. Kiran Telukunta**, **Jianyu Li**, **Mingjie Gao**, **Dr. Pankaj Mishra**, **Dr. Dennis Klementz**, **Dr. Paul Zierep**, **Dr. Martin Hügle**, **Pascal Kirchner**, **Marius Amann**, **Aly Abotaleb**, **Sinclair Rockwell-Kollmann**, **Simon Pfäffle**, and **Florian Sauter**. With them, I've had a lot of fantastic and enjoyable adventures and created a lot of unforgettable memories.

A very special thank to **Aurélien F. A. Moumbock** for helpful discussions and feedback

and collaboration in some publications.
I appreciate **Manuel Dorer** and **Zhongyi Kang**'s contributions to this thesis as master's students whom I have supervised throughout their practicum and master's thesis.

I would also like to take the chance to thank my friend **Khaled Alazzani** who represents more than a friend to me, but rather my older brother, the person who always supported me in my study and life. Besides this, I thank my friends and Colleagues at Spinner Data: **M. Ghadiry**, **M. Elbahidy**, **S. Mahmood**, and my manager **D. Spinner**, who supported me in my part-time job there.

Finally, my thanks to everyone who contributed to this work directly or indirectly.

Abstract

A tremendous amount of electronic research data is freely available as online open-source published literature, and which is rapidly growing. This huge, unstructured data contains a great wealth of valuable information which is hidden and difficult to access; e.g. it might be difficult for scientists to identify specific articles of interest. Artificial intelligence-based text mining and machine learning approaches are being exploited to process and analyze such huge amounts of data to identify and extract relevant information. Relevant information can be concepts as well as relationships between those concepts which answer questions of interest. Identifying biomedical concepts (e.g. compounds, proteins, diseases) and the functional relationships between them is one of the important domains in text mining and forms a key component in life science research. In the drug discovery field, knowledge of how small molecules associate with proteins plays a fundamental role in understanding how drugs or metabolites can affect cells, tissues, and human metabolism.

This dissertation focuses on the automated identification of functional compound-protein relationships in biomedical and life sciences literature using text mining and machine learning techniques. A new benchmark dataset of 2,613 sentences was created, consisting of 5,562 small molecule and protein pairs which had been previously annotated with the help of text mining tools. The pairs were subsequently classified manually as functional or non-functional. Three machine learning approaches named shallow linguistic kernel (SL), all-paths graph kernel (APG), and BioBERT were evaluated to classify these relationships between small molecules and proteins. Furthermore, the benefit of the presence of interaction verbs in sentences which include the functional related compound-protein pairs was evaluated.

On the benchmark dataset, the BioBERT machine learning approach achieved the best performance, with an F_1-score of 86.0%, precision of 85.2%, and recall of 86.8%. Moreover, the trained model was applied on all titles and abstracts of the articles stored in the

PubMed database. The results were processed and included in a new web server for literature research (CPRiL). The data allows novel query options, such as the calculation of the shortest relation path between any biomolecule. Currently, CPRiL contains $\sim$2.5 million unique functional related compound-protein pairs, with $\sim$460,000 unique names and synonyms of small molecules and $\sim$90,000 unique proteins.

Zusammenfassung

Eine enorme Menge elektronischer wissenschaftlicher Daten ist als online veröffentlichte Open-Source-Literatur frei verfügbar und wächst rasant an. Diese riesigen, unstrukturierte Daten enthalten einen großen Reichtum an wertvollen Informationen, welche jedoch versteckt und schwer verfügbar sind; für Wissenschaftler kann es z.B. schwierig sein, bestimmte Artikel von Interesse zu identifizieren. Auf künstlicher Intelligenz basierende Text Mining- und Machine Learning-Ansätze werden genutzt, um solche riesigen Datenmengen zu verarbeiten und zu analysieren, um relevante Informationen zu identifizieren und zu extrahieren. Bei den relevanten Informationen kann es sich sowohl um Konzepte als auch um Beziehungen zwischen diesen Konzepten handeln, die Antworten auf Fragen von Interesse geben. Die Identifizierung biomedizinischer Konzepte (z.B. chemische Verbindungen, Proteine, Krankheiten) und der funktionalen Beziehungen zwischen ihnen ist einer der wichtigsten Bereiche des Text Mining und bildet eine Schlüsselkomponente in der biowissenschaftlichen Forschung. Im Bereich der Arzneimittelforschung spielt das Wissen darüber, wie kleine Moleküle mit Proteinen assoziiert sind, eine grundlegende Rolle für das Verständnis, wie Arzneimittel oder Metaboliten Zellen, Gewebe und den menschlichen Stoffwechsel beeinflussen können.

Diese Dissertation befasst sich mit der automatisierten Identifizierung von funktionalen Wirkstoff-Protein-Beziehungen in der biomedizinischen und biowissenschaftlichen Literatur mithilfe von Text Mining und Machine Learning. Es wurde ein neuer Benchmark-Datensatz von 2.613 Sätzen erstellt, der aus 5.562 Paaren von kleinen Molekülen und Proteinen besteht, die zuvor mit Hilfe von Text Mining-Tools annotiert wurden. Die Paare wurden anschließend manuell als funktional oder nicht-funktional klassifiziert. Zur Klassifizierung dieser Beziehungen zwischen kleinen Molekülen und Proteinen wurden drei Machine Learning-Ansätze, ”shallow linguistic kernel” (SL), ”all-paths graph kernel” (APG) und BioBERT, bewertet. Darüber hinaus wurde der Nutzen des Vorhandenseins von Inter-

aktionsverben in Sätzen, die funktional verwandte Stoff-Protein-Paare enthalten, bewertet.

Für den Benchmark-Datensatz erzielte der BioBERT-Ansatz die beste Leistung mit einem F_1-Score von 86,0%, einer Präzision von 85,2% und einem Recall von 86,8%. Außerdem wurde das trainierte Modell auf alle Titel und Abstracts der in der PubMed-Datenbank gespeicherten Artikel angewendet. Die Ergebnisse wurden verarbeitet und in einen neuen Webserver für Literaturrecherchen (CPRiL) aufgenommen. Die Daten ermöglichen neuartige Abfragemöglichkeiten, wie z.B. die Berechnung des kürzesten Beziehungspfades zwischen beliebigen Biomolekülen. Derzeit enthält CPRiL $\sim$2,5 Millionen eindeutige, funktionell verwandte Substanz-Protein-Paare, mit $\sim$460.000 eindeutigen Namen und Synonymen kleiner Moleküle und $\sim$90.000 eindeutigen Proteinen.

TABLE OF CONTENTS

LIST OF PUBLICATIONS

The following peer-reviewed articles were published during the course of the doctoral studies. The first two listed publications are a result of the main project described in this dissertation.

1. **Qaseem A**, Günther S.
   CPRiL: Compound-Protein Relationships in Literature. *Bioinformatics*. **2022** Sep 15:4452-4453. doi: 10.1093/bioinformatics/btac539. PMID: 35920772.

2. Döring K*, **Qaseem A***, Becer M, Li J, Mishra P, Gao M, Kirchner P, Sauter F, Telukunta KK, Moumbock AFA, Thomas P, Günther S. Automated recognition of functional compound-protein relationships in literature. *PLoS One*. **2020**; 15(3):e0220925. PMID: 32126064.

3. Gao M*, Moumbock AFA*, **Qaseem A***, Qianqing Xu, Günther S. CovPDB: a high-resolution coverage of the covalent protein–ligand interactome. *Nucleic Acids Research*. **2022**; 50(D1):D445-D450. PMID:

34581813.

4. Li J*, Moumbock AFA*, **Qaseem A***, Xu Q, Feng Y, Wang D, Günther S. AroCageDB: A Web-Based Resource for Aromatic Cage Binding Sites and Their Intrinsic Ligands. *J Chem Inf Model*. **2021**; 61(11):5327-5330. PMID: 34738791.

5. Moumbock AFA*, Gao M*, **Qaseem A***, Li J, Kirchner PA, Ndingkokhar B, Bekono BD, Simoben CV, Babiaka SB, Malange YI, Sauter F, Zierep P, Ntie-Kang F, Günther S. StreptomeDB 3.0: an updated compendium of streptomycetes natural products. *Nucleic Acids Research*. **2021**; 49(D1):D600-D604. PMID: 33051671.

6. Simoben CV, **Qaseem A**, Moumbock AFA, Telukunta KK, Günther S, Sippl W, Ntie-Kang F. Pharmacoinformatic investigation of medicinal plants from East Africa. *Molecular Informatics*. **2020**; 39(11):e2000163. PMID: 32964659.

---

* **These authors contributed equally to this work.**

LIST OF ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
APG	All-paths Graph Kernel
API	Application Programming Interface
AUC	Area Under the Curve
BERN	Biomedical Named Entity Recognition and Normalization Tool
BERT	Bidirectional Encoder Representations from Transformers
BFS	Breadth-First Search
BioBERT	Bidirectional Encoder Representations from Transformers for Biomedical Text Mining
CIL	Compounds In Literature
COMP	Compound
CPI-DS	Compound-Protein Interaction Dataset

CPI-DS_IV	Compound-Protein Interaction Dataset with enclosed Interaction verb
CPI-DS_NIV	Compound-Protein Interaction Dataset without enclosed Interaction verb
CPRiL	Compound-Protein Relationship in Literature
CPU	Central Processing Unit
CTD	Comparative Toxicogenomics Database
CV	Cross-validation
DL	Deep Learning
FN	False Negatives
FP	False Positives
GPU	Graphics Processing Unit
HTML	HyperText Markup Language
ID	Identifier
InChI	International Chemical Identifier
IR	Information Retrieval
K	Thousand
M	Million
MeSH	Medical Subject Headings
ML	Machine Learning
MLM	Masked Language Modeling
NCBI	National Center for Biotechnology Information
NER	Named Entity Recognition
NLP	Natural Language Processing

NLTK	Natural Language Toolkit
NSP	Next Sentence Prediction
OS	Operating System
PMC	PubMed Central
PMID	PubMed ID
PoS	Part-of-Speech
PROT	Protein
PTC	PubTator Central web service
PubChem	Public Chemical Database
PubMed	Public/Publisher MEDLINE (NLM journal articles database)
RAM	Random Access Memory
RDBMS	Relational Database Management System
RE	Relation Extraction
RLS	Regularized Least Squares
ROC	Receiver Operating Characteristic
SL	Shallow Linguistic Kernel
SMILES	Simplified Molecular Input Line Entry Specification
SQL	Structured Query Language
SVM	Support Vector Machine
TN	True Negatives
TP	True Positives
TPU	Tensor Processing Units

UniProt	Universal Protein Resource

LIST OF FIGURES

LIST OF TABLES

CHAPTER

1 INTRODUCTION

1.1 Text Mining in Biomedical Research

A tremendous amount of data is freely available in the published literature and is rapidly growing in size. This huge, unstructured data contains a great wealth of information related to numerous and diverse topics. For instance, the MEDLINE database is one of the largest resources of unstructured data, in the form of several millions of publication reference strings and abstracts on life sciences and biomedical topics. However, the process of retrieving and extracting relevant information from text is increasingly difficult and time-consuming for human beings. Artificial intelligence (AI) has been extensively employed in a wide range of domains in the last decades, including pattern recognition and natural language processing (NLP). Artificial intelligence-based text mining is used to perform Information Retrieval (IR) tasks efficiently and intelligently. Machine learning approaches, on the other hand, enable machines to learn from data using features and then execute certain jobs intelligently.

1.2 Relation Extraction of Biomolecules from Literature

Relation extraction is one of the essential tasks of text mining; it concerns identifying the relationships between two entities in unstructured text, such as biomedical articles. Entity recognition is considered a cornerstone of relation extraction, where identifying the entities' quality and accuracy helps to recognize the relations between these entities accurately. Because the manual annotation of entities is a time-consuming process, artificial intelligence-based text mining and machine learning techniques are the most efficient alternative option to annotate entities automatically in the unstructured data.

Relationships can be extracted using a variety of methods, from the straightforward co-occurring method to the more complex automated machine learning methods. Simply explained, the concept behind the co-occurring approach is when biomolecules appear together in a text or a sentence they are more likely related; however, the machine learning approach involves creating a learned model which can automatically identify the newly described relationships in texts efficiently and intelligently. Functional relationships between biomolecules are essential for all processes in the cell, such as metabolism, signaling, regulation, and proliferation [1]. Small molecules (compounds) can serve as substrates by interacting with enzymes, as signal mediators by binding to receptor proteins, or as drugs by interacting with specific target proteins [2]. Extracting and studying such relationships is crucial to the fields of molecular biology, biochemistry, medicine, and pharmacy. This information, which is usually presented in the form of academic journals, offers a valuable resource for understanding signaling pathways, targeting of proteins, and efficacy and side effects of drugs. It is not easy to identify a precise functional relationship in these articles, because related material might be dispersed among a high number of articles.

1.3 Current Text Mining Applications and Methods for Evaluation

Very useful approaches were published for named entity recognition, e.g. chemical compounds, diseases, and proteins. PubTator Central (PTC) is a web-based service [3] for

automatic annotations of biomedical entities such as genes and chemical compounds in PubMed abstracts and PMC full-text articles using artificial intelligence-based text mining and machine learning approaches. Recently, a new neural biomedical named entity recognition and normalization tool (BERN) [4] was developed based on a pre-trained biomedical language representation model for biomedical text mining (BioBERT) [5]. BERN can annotate biomedical entities from plain text or PMID as input. The automatic and accurate named entity recognition process allows us to develop novel text mining methods and results - for example, the relationships between drugs and proteins.

The BioCreAtIvE (Critical Assessment of Information Extraction systems in Biology) challenge evaluation consists of a community effort for evaluating information extraction and text mining systems applied to the biological domain [6]. BioCreAtIvE is concerned with the extraction of biologically relevant information from the literature. Two main issues are addressed at BioCreAtIvE challenge: the first one is entity recognition such as chemical compounds and protein names; the second one is entity associations such as protein-functional associations. The BioCreAtIvE datasets have been created by biological experts and are useful resources for the development of relation extraction systems. Databases can provide a useful alternative to time-consuming manual literature research but mainly describe direct interactions, e.g. PDBbind offers experimentally measured binding affinity data for protein-ligand complexes [7]. ChEMBL is a manually curated bio-database which offers activity information of molecules with drug-like characteristics [8]. DrumPID is a database which offers information on drugs and associated protein networks including information on indications, protein targets, and off-targets [9]. DrugBank combines comprehensive drug target data, including sequence and structure, with in-depth drug data, covering chemistry and pharmacology [10].

The Comparative Toxicogenomics Database (CTD) provides manually collected relations of chemicals with genes/proteins which can affect human health [11]. Machine learning methodologies are already supported by some datasets which concentrate on molecular interactions, such as STITCH; this includes information of chemical-protein interactions which are collected from experimental data and other primary databases but also includes predicted data collected by text mining methods [2]. STRING is similar to STITCH, but

focuses mainly on protein-protein interactions [12]. OntoGene is a web-based service for biomedical entity recognition and their relationships based on text mining technologies [13].

However, no precise statistical measurements for predicting protein-compound interactions have been published. Furthermore, no gold standard corpus of annotated compound-protein interactions has been published for evaluation of text mining techniques for their identification. In the BioCreAtIvE challenges, rule-based and machine learning approaches are used for the automatic identification of drug/chemical and gene/protein interactions in the biological domain [14, 15]. The ChemProt benchmark dataset was utilized in the shared task for text mining chemical-protein interactions in BioCreAtIvE VI; it includes chemical-protein interactions extracted from PubMed abstracts and was annotated manually by domain experts [14]. However, ChemProt benchmark focuses on validated interactions and is therefore not suitable for the separation from functionally unrelated compound-protein pairs which are mentioned in texts.

1.4 Fundamentals of Artificial Intelligence and Graph Theory

This section describes general concepts which apply to the methods used to annotate biomedical entities and to identify functional relationships between them.

1.4.1 Artificial Intelligence (AI)

Artificial intelligence is a broad term which refers to techniques which simulate human behavior. It is a branch of computer science which focuses on the development of intelligent machines and technologies which have the ability to perform tasks which simulate those performed by human beings. Artificial intelligence is used in many applications, including robotics, aircraft, self-driving cars, and smartphones. All the methods of identifying the biomedical concepts (compounds and proteins) and compound-protein relationships (SL, APG, BioBERT) used in this dissertation are based on AI techniques.

1.4.2 Machine Learning (ML)

Machine learning is a field of artificial intelligence which attempts to make devices capable of learning but without the need to program them literally. It is a technology which allows machines to learn from data via computational methods to perform specific tasks intelligently. Thus, these machines can perform complex operations by learning from data using variables, also called 'features', rather than following pre-programmed rules. Machines are trained to think in a similar way to how humans do. An example would be statistical methods which train a system to identify patterns within data; these patterns can be applied to test data afterwards. The ML model can perform well if the features of the input data provide sufficient information to characterize the class of this data; it can also perform well if it can handle the complexity of the connections between the features of input data and its output class. There are many types of machine learning, but the most popular methods of machine learning are supervised learning and unsupervised learning [16, 17].

1.4.2.1 Supervised Learning

A supervised machine learning approach is trained on labeled training data. In supervised learning, each example consists of a pair of input and target/label. A supervised learning algorithm studies the training dataset and generates a model which can be applied to mapping a new observation on the basis of the learned information of the training dataset. Supervised learning can be divided into two types, classification and regression. The classification predicts a discrete target/label, while regression predicts a continuous quantity or real value [18]. The methods which are used in this dissertation (SL, APG, BioBERT) use supervised learning.

1.4.2.2 Unsupervised Learning

Unsupervised learning is a machine learning approach which learns from unlabeled training data. In this approach, the algorithms have the capability - and without any external assistance - to self-learn based on similarities and differences and to build a model which can classify and categorize a new observation into the closed category on the basis of

the similarities of the patterns. Clustering algorithms are good examples of unsupervised learning, where the algorithm is grouping and categorizing the objects which are similar to each other and different to the objects in the other clusters. Unlike supervised learning, the number of classes in this type of learning is unknown [19].

1.4.3 Kernels Method

The kernel in machine learning is a technique which allows for solving non-linear problems using linear classifiers by mapping non-linear features to a higher-dimensional space without explicitly creating those feature mapping, but rather by simply using a kernel trick which computes the inner products between all pairs of data in the feature space [20]. The kernel trick is computationally (time and space) cheaper than computing the coordinates explicitly. The kernel function implicitly maps data from its original space to a higher dimensional feature space. In the real world most likely the problems are not linearly separable, but mapping the data into the higher-dimensional makes the problem solvable. There are several types of kernels, including Support Vector Machine (SVM), polynomial, and Gaussian kernels [21, 22]. SL and APG, as used in this dissertation, are built based on the kernel technique.

1.4.4 Neural Networks

Neural networks are a subset of machine learning inspired by the human brain structure and form the core of deep learning algorithms. A neural network is a series of neurons connected to each other and where the neuron is the main block of the neural network which holds a number, precisely a number between 0 and 1. A neural network has three types of layer (Figure 1.1):

a) Input layer: this is the first layer in the neural network. It receives the input data and transmits them to the first hidden layer in the network. It does not perform any operation on the input data and does not have any weights or biased values associated.

b) Hidden layer: this is the one which performs mathematical computation on the inputs and can be imagined as a features extractor. A collection of neurons stacked vertically

represents one hidden layer. One of the challenges in creating neural networks is deciding the number of hidden layers and also the number of neurons for each layer. The network can have one or more hidden layers; the term deep in deep learning refers to having more than one hidden layer in it. The last hidden layer is connected to the output layer.

c) Output layer: this is the last layer in the neural network which returns the output data or prediction.

Each connection between neurons is associated with the weight (which represents the strength of the connection between neurons) and bias (which is used to adjust the output). The way the network operates activations in one layer determines the activations of the next layer, i.e. the pattern of activations in the input layer causes some very specific pattern in the next layer, which causes some pattern in the one after it, which finally gives some pattern in the output layer. The training's goal is to update this weight value in order to reduce the loss [23–26].

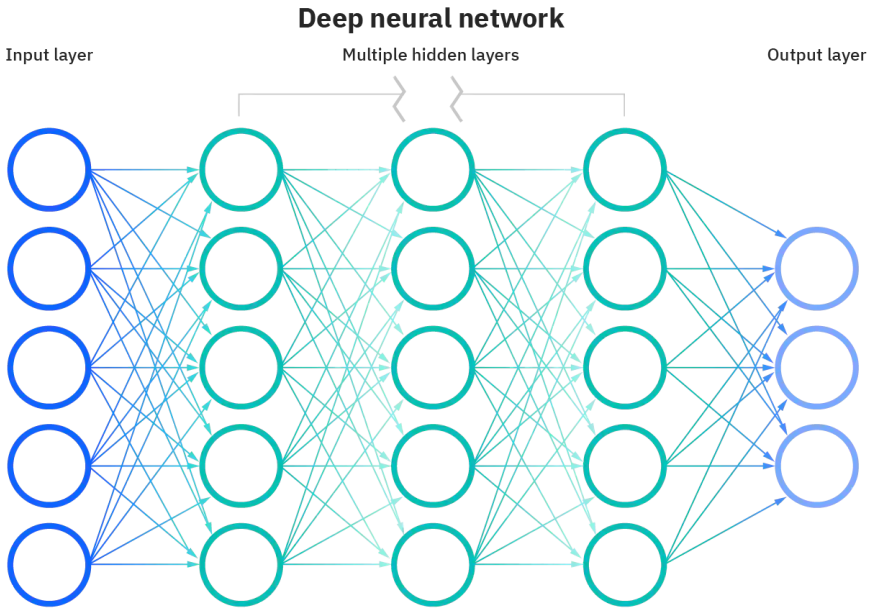


Figure 1.1: Neural networks structure. Taken from https://www.ibm.com/cloud/learn/neural-networks [27].

1.4.5 Deep Learning

Deep learning, or artificial neural network, is a subtype of machine learning which is inspired by the human brain's structure architecture. It is distinct from machine learning in which it learns without requiring human intervention. Deep learning algorithms use a logical structure to analyze data, in order to make comparable conclusions as a human would. They attempt to extract useful patterns from data in an automated way with as little human effort involved as possible and using a multi-layered structure of algorithms, called neural networks. Deep learning carries some limitations, including:

1. data as training a deep learning model requires huge chunks of dataset to make it decently accurate.

2. the training process in a deep learning system requires a high amount of computation; that's why it generally employs a graphical processing unit (GPU) which has more cores than a CPU. A deep learning system can take weeks or even months to process and train a neural network, the training time is usually dependent on the amount of data and the number of hidden layers in the network.

Deep learning is used widely in many fields including image recognition, speech recognition, self-driving cars, google search and translation, bioinformatics, drug design, medical image analysis, and much more [28, 29].

AI vs. Machine learning vs. Deep learning

Figure 1.2 shows the main difference between artificial intelligence (AI), machine learning (ML), and deep learning (DL). Artificial intelligence (AI) includes programs with the ability to learn and reason and mimic human behavior, however, machine learning (ML) includes algorithms with the ability to learn without being explicitly programmed like statistical methods which train a system to identify patterns within training data; afterward, these patterns

can apply to new data. Deep learning (DL) is a subset of machine learning (ML) in which artificial neural networks adapt and learn from huge amounts of data.

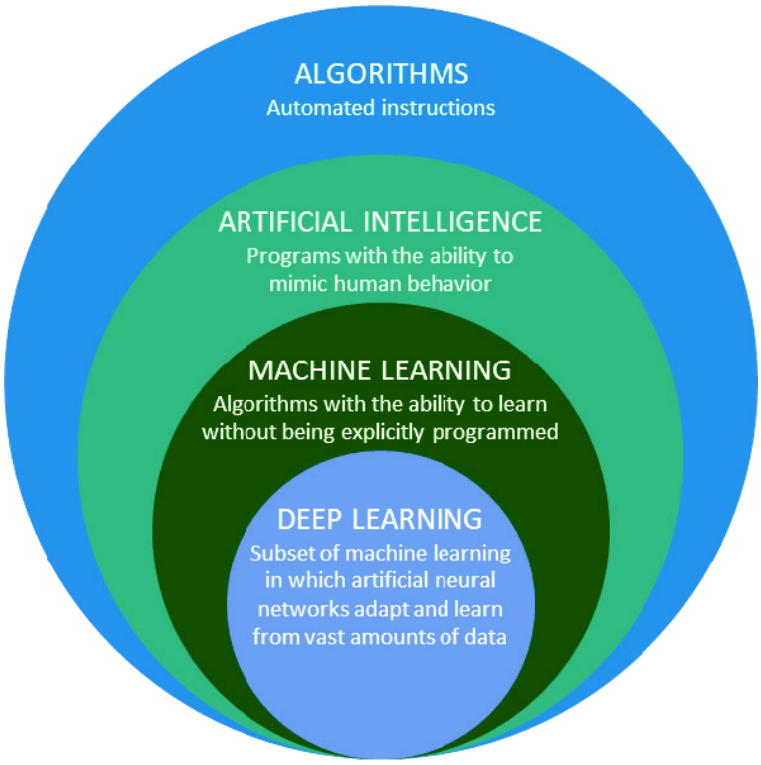


Figure 1.2: Difference between AI, ML, and deep learning (Author: Johannes Vrana, Vrana GmbH, Licenses: CC BY-ND 4.0 [30].

The big advantage of deep learning is its power to utilize huge data, as deep learning has a tendency to continue learning with receiving more data [31]. Classical typical machine learning methods tend to fairly quickly get to a point where more training data is not providing additional performance gains while deep learning methods tend to continue learning as long as you are willing to continue training them (Figure 1.3).

1.4.5.1 Bidirectional Encoder Representations from Transformers (BERT)

The BERT model, introduced by Jacob Devlin in Google in 2019, is pre-trained on plain, unlabeled text corpus including the entire English Wikipedia with 2500 million words and a BookCorpus with 800 million words [33]. These words were represented by a total of 30,000 token vocabularies including common words and parts of words. BERT is a word vector model, and its goal is to build a decent feature representation for words by

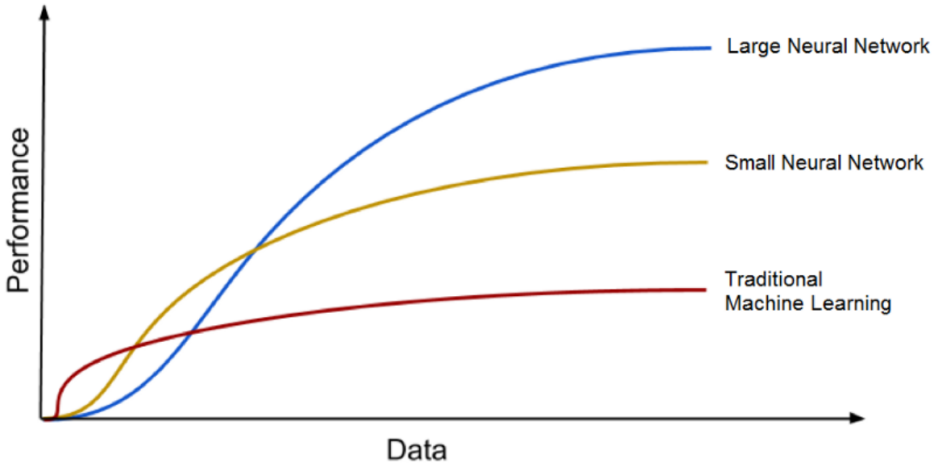


Figure 1.3: Performance of deep learning vs traditional machine learning methods with the scale of the amount of data. Deep learning models (artificial neural network) outperform machine learning models with increasing the training dataset, this tendency is quite significant [32].

performing a self-supervised learning approach on a massive corpus. Self-supervised learning is supervised learning which is performed on data which has not been manually labeled. No annotations of the texts by humans were required, so the training in the machine learning process is self-supervised.

BERT is bidirectionally trained i.e. every token can attend context to its left and right at the same time. This feature enables the model to learn the context of a word depending on the words around it, resulting in a better understanding and sense of the words. The bidirectional model has a better awareness of the language context than the single-direction model [33]. BERT is built on transformers, a deep learning model in which every output element is linked to every input element, and the weightings between them are determined dynamically based on their relationship (this is referred to as attention in NLP).

Pre-training of BERT

BERT is a pre-trained language model for natural language processing (NLP). The goal of pretraining BERT is to make it aware of the distinction between language and context. BERT is pre-trained on two independent but related NLP tasks:

1. **Masked Language Modeling (MLM):** The purpose of Masked Language Modeling training is to learn a representation for each token by understanding the bi-directional context of the tokens. In MLM training, 15% of the tokens of the input sequence are

masked at random, then the model is trained to predict the masked words based on the context of the masked words. BERT uses 80-10-10 strategy, from the chosen 15% tokens, randomly 80% of them are replaced by a [MASK] token, 10% by random tokens, and 10% are left unchanged. The latter is used to bias the representation towards the actual observed word. MLM helps to understand the relationship between words in the same sentence.

2. **Next Sentence Prediction (NSP):** The purpose of Next Sentence Prediction training is to have a model which predicts whether two given sentences are logically related or not. This helps BERT to understand context across different sentences and the relationship between them. A good example of NSP is the question-answering task which is the task of extracting the answer of a question in a given document.

Both Masked LM and NSP are used to train the BERT model. The goal is to get a good understanding of language and reduce the combined loss function of the two techniques.

BERT Architecture

The BERT model contains the following layers:

- Two inputs: One from word tokens, one from segment-layer. These get added and summed over to a third embedding: position embedding, followed by dropout and layer normalization.
- Followed by 12 Multi-head Self Attention layers.
- Following these 12 layers, there are two outputs — one for NSP (Next Sentence Prediction) and one for MLM (Masked Language Modeling).

BERT is the encoder of a transformer consisting of multiple layers, each layer applies self-attention and hands the results to the next layer [34].

BERT Tokenization and Encoding

BERT was designed to process input sequences of up to length 512. An input sequence

needs to be preprocessed before being fed into the BERT model. The following steps illustrate the preprocessing procedure for the input sequence:

- Tokenization: breaking down the input sequence into tokens using a method called WordPiece tokenization.
- Adding the [CLS] token at the beginning of the sentence, and the [SEP] at the end of the sentence. For the classification task, a special token [CLS] representing the class of the entire input sequence is added to the beginning of the input sequence. In the ”next sentence prediction” task, we need a special token to inform the model of the ending of each sentence in the sequence, for this purpose, [SEP] token is added to the end of each sentence in the input sequence.
- Padding the input sequence with [PAD] tokens so that all input sequences have the same maximum length.

Contextual Embeddings

Contextual embedding captures the semantics of the word such that the same word can have different meanings across varied contexts, unlike the word embedding in which the word has a global meaning regardless of the word's context in the sequence. BERT has three separate embedding layers:

- **Token Embeddings**: Because the model cannot directly recognize words, but only numbers, it is necessary to map each token to the corresponding unique vocabulary ID. Each token in the input sequence is transformed into a vector representation of a fixed 768-dimensional vector. Because the size of the dictionary of the vocabularies is fixed (around 30K), the words are split into their root to map them to the corresponding unique ID, the tokens not appearing in this dictionary are replaced by a special token [UNK].
- **Segment Embeddings**: A marker is added to each token to indicate the sentence (A or B) which this token belongs to.

- **Position Embeddings**: A positional embedding is added to each token to indicate the position of the token in the input sequence.

The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings (Figure 1.4). The input representation which is formed by summing the corresponding is passed to BERT's Encoder layer.

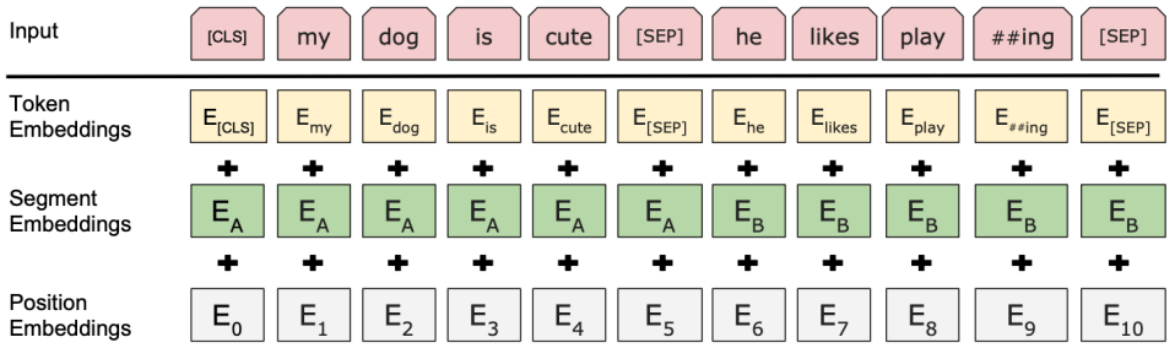


Figure 1.4: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings as well as the position embeddings [33].

Encoder Layers of BERT

BERT consists of multiple encoder layers, 12 for the BERT-Base model and 24 for the BERT-Large model. Each encoder has two layers: Self-attention and feed-forward network (Figure 1.5). The encoder receives a list of vectors as input, this list is processed by sending these vectors through a "self-attention" layer, then a feed-forward neural network, and finally the up-coming encoder. The transformer utilizes the self-attention method to understand the relevance of the other words to the one which is currently processing [35]. Each layer does the following:

- Each layer applies self-attention: The self-attention layer essentially learns a contextualized meaning for each word in the input.
- Passes its results through a feed-forward network: The purpose of this layer is to transform the output of the attention layer (the attention vector) into a format which can be processed by the next encoder block.

- Hands the result to the next encoder layer.
- The process is continued in this manner until the last transformer block is reached.

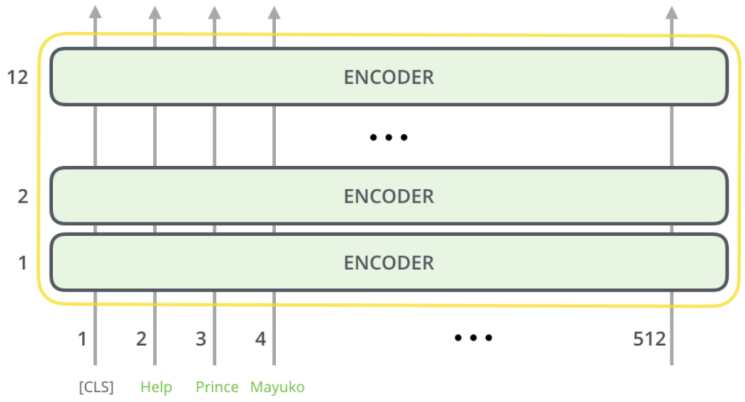


Figure 1.5: BERT Stack Encoder layers [36].

BERT Models

Google provides two main models of BERT:

1. **The BERT-Base model**: It has a total of 110 million parameters with 12 encoder layers, 768 hidden nodes, and 12 attention-heads.
2. **The BERT-Large model**: It has 340 million parameters with 24 encoder layers, 1024 hidden nodes, and 16 attention-heads.

Each model comes with BERT-uncased and BERT-cased formats, which correspond to whether to include case or not. BERT-uncased (only lowercase) is more commonly used because in most scenarios the case of a word does not have a big impact on the task. But in some specific scenarios, such as named entity recognition (NER), BERT-cased is more suitable. BERT-Base was trained for 4 days on 4 cloud TPUs, whereas BERT-Large was trained for 4 days on 16 cloud TPUs!

BERT Fine-Tuning

Fine-tuning is a process of taking a BERT model which was pre-trained on a large amount of generic text and adding more training with a specific application or domain-specific dataset to optimize the performance on a specific task. BERT can be used for a wide range of NLP tasks by adding only a single layer to the top of the core model.

1.4.6 Text Mining

Text mining is an artificial intelligence (AI) technique which is the ability to filter out a very large set of unstructured text in documents such as books and literature and convert them into structured data containing relevant information [37]. Relevant information can be concepts as well as relationships between those concepts which answer questions of interest. In the task of looking for associations, the standard keyword search introduces a lot of noise in the results, furthermore, it needs to go through the whole document of the results to extract the relevant information if it exists. However, text mining techniques are extracting those relevant information efficiently for further analysis or to drive machine learning algorithms [38, 39].

1.4.7 Natural Language Processing (NLP)

Natural language processing refers to the field of artificial intelligence which enables machines to read, analyze, understand, and interpret the meaning of human languages (text and speech) in a smart and efficient way [40–42]. NLP includes the field of linguistics in computer science which understands and learns the structure of the language and creates models which analyze text and speech to isolate and extract significant features. Data analysts and machine learning experts utilize data to enable machines to mimic human linguistic behavior. The applications of natural language processing include machine translation, spell-checking, keyword search, advertising matching, text filtering, and more. NLP is divided into two major components: natural language understanding and natural language generation. Natural language understanding is the process of analyzing the given input and extracting the significant data, whereas natural language generation is the process of generating meaningful sentences and phrases [43, 44].

1.4.7.1 Tokenization

Tokenization is a common task in Natural Language Processing (NLP), most NLP applications require tokenization as a pre-processing step. Tokenization of text into subword units, which typically maintain linguistic meaning, is a common solution in modern NLP ap-

proaches. In the tokenization process, the unstructured input string breaks into a series of discrete components appropriate for machine learning (ML) models called tokens; tokens can be either words, subwords, or even characters. Even if the model does not recognize a word, individual subword tokens may still include enough details for the model to derive the meaning of the word somehow. Words which aren't in the lexicon, on the other hand, are considered as "unknown" in this technique. WordPiece is one of the most popular subword tokenization techniques that is commonly applied to many NLP models [45, 46].

1.4.7.2 Information Retrieval (IR)

IR is a process to find relevant texts. The most common approach to do this is called ad hoc retrieval, which is in fact what we do every time when we go to PubMed and type a query. Pubmed has a very large document collection which has been indexed and which can be quickly searched for all papers which match a specific query. This is one approach for information retrieval. Another common approach is document similarity which is used by recommendation engines. The idea is to take each document and turn it into a term vector where each dimension in the vector corresponds to a different word in the document. In the next step, a weighting scheme is applied to place more emphasis on words which are more important. Finally, a vector similarity is calculated which can assess the similarity of the document and rank them in terms of which documents are most similar to the document of interest [47–49].

1.4.7.2.1 Named Entity Recognition (NER)

Named entity recognition is a field of natural language processing (NLP) of identifying and categorizing key information (entities) in text and classifying them into a pre-defined category, simply a process to find and classify entities in text. For NER a dictionary of official names and synonyms of the entity is needed which can be applied to recognize or identify for example genes/proteins or diseases. In addition, a black list which contains a list of names which are conflicting with the dictionary of entities and their synonyms are beneficial for training. This process can be very labor intensive and time-consuming. Another approach to identify and recognize entities utilizes machine learning techniques

which look into the context around the entity. In this way, the model can learn from the surroundings of the entities how the shape of the entities can be recognized more efficiently even if the specific name of this entity has not been seen before. The accuracy of the NER model highly depends on the training dataset which the model uses for learning including the context of each entity for recognition and for categorization [50, 51].

1.4.7.2.2 Relationships Extraction (RE)

Relationship extraction, also known as relation extraction, is the process of identifying the relationships between two entities in unstructured text, such as biomedical articles [52]. There are various techniques to perform relationships extraction, ranging from a simple approach so-called co-mentioning to a more sophisticated automated deep learning approaches. The idea of the co-mentioning approach is if A and B are mentioned together they might have something to do with each other. To improve this approach a higher number of co-occurrences increases the probability that A and B have some type of relationship. Deep learning approaches are used for relation extraction tasks that typically involve pre-trained language models like BERT that allow for recognizing statements like “A binds to B”. The advantage of using a deep learning approach is that it not only identifies the existence of the relationship but in addition, determines the type of relationship [52, 53].

1.4.8 Classification

Classification is the most common task of machine learning, it is a problem of assigning a given observation(s) or object(s) into a distinct class. Classification is a supervised machine learning technique, where the classifier is trained on a training dataset to understand how to identify the class of a new object on the basis of the information which is learned from the training dataset. Out of the training process, the classifier creates a model which can be used later to identify the unseen/new object to one of the available classes based on the similarity of the features. Classes can be called categories, targets, or labels [54–56]. There are two main types of Classifiers:

1.4.8.1 Binary Classification

Binary classification is the task to map an observation/object into one of two class targets/labels normally Yes/No or positive/negative. An application of binary classification is cancer detection (cancer or not), Email spam detection (spam or not), functional relationship prediction (functional relationship or not), and so on.

1.4.8.2 Multi-class Classification

In this type of classification, there are more than two class targets/labels. The classifier identifies one and only one target/label for each new observation/object, however, the available classes/labels are more than two. An application of this type of classification is Face recognition and Entity recognition.

Classifier: It is an algorithm which has the capability to map a new observation or object to a distinct class or category on the basis of the training dataset. The performance of a classifier model is measured using a so-called performance metric, it can be either a numeric value like precision or a score based metric like a receiver operating characteristic (ROC) curve.

1.4.9 Cross-validation (CV)

Cross-validation is a technique to validate how accurately a machine learning model will perform in practice and measure the performance of the model [57]. To perform cross-validation, the data is randomly split into two datasets, called the training dataset and the testing or validation dataset. Normally, the training dataset is bigger than the test dataset. The classifier is using the training dataset to learn and understand the dataset feature of each class and build a model based on this dataset which can predict and classify a new unseen dataset. The test dataset is a sample of data used to evaluate the performance of the classifier when it is applied to predict and classify the test dataset. In the training dataset, every given data has a specific label or class to allow the classifier to train and learn from this data, on the other hand in the test dataset the label or class of each data is

hidden to allow the model to make the prediction, then later actual labels or classes of the test dataset are used to evaluate the performance of the classifier when they compare with the predicted classes. The common cross-validation methods include holdout and k-fold cross-validation [58, 59].

1.4.9.1 Holdout Cross-validation

Holdout is the simplest method of cross-validation. In this method, the data is randomly split into two datasets, the training dataset, and the test dataset. The usual size of the training dataset is between 70% to 80% of the dataset and between 30% to 20% for the test dataset. The advantage of this method is that it is very simple and takes less computing time. However, its evaluation can have a high variance, because it does not consider the averaging of multiple runs over different splits [60, 61]. This method was used to evaluate the final performance of the evaluated methods used in this dissertation.

1.4.9.2 K-fold Cross-validation

This type of cross-validation is recommended when the size of the dataset is small. The dataset is randomly split into k-folds, the value of the parameter k can be arbitrary, but ideally, k is chosen between 5 to 10 based on the size of the dataset. The model is built using k-1 folds as a training dataset and then it validates using the kth fold as validating or testing dataset. Repeat this process k times by choosing every round a different fold as a validating or testing dataset. Every round the performance metric of the built model is calculated. Finally, calculate the average of the k-scores to get the performance metric of the entire model. Obviously, this method requires more computation time than a simple holdout cross-validation, because it considers all data in both the training and test processes [62].

1.4.10 Confusion Matrix

A confusion matrix is a critical tool for helping to evaluate the performance of the classification model or classifier on a set of test data for which the actual values are known. A

confusion matrix is only used for classification models meaning models which are predicting class labels and they are not used for regression models which are used to predict numeric values. For a binary classifier, there are two possible classes/labels which in this case is a two-by-two matrix, if there were three possible classes then it would be a three-by-three matrix, and so on. The two possible classes are Yes and No but they could be other things like positive and negative, or zero and one [62–64]. Figure 1.6 shows a confusion matrix for a binary classification problem, there are four terms:

- **True Positive (TP)**: when the model correctly predicts the positive class.
- **True Negative (TN)**: when the model correctly predicts the negative class.
- **False Positive (FP)**: when the model incorrectly predicts the positive class (sometimes referred to as type I error).
- **False Negative (FN)**: when the model incorrectly predicts the negative class (sometimes referred to as type II errors).

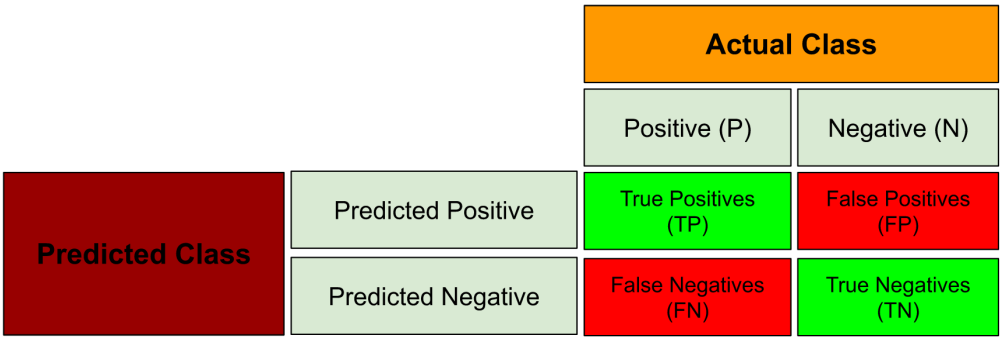


Figure 1.6: Confusion matrix for binary classification.

1.4.11 Performance Metrics

In the classification tasks, performance is the capacity of the model to identify the class of observation using test data. The performance metrics are metrics to evaluate the quality or the performance of the model or classifier to predict the classes of the observations. There are five main metrics: accuracy, specificity, recall, precision, and F-score. Next are the main performance metrics of the binary classifier [63–66].

- **Accuracy** is the portion of the correctly classified values. It tells us how often the classifier has been correct; it is calculated by the sum of all true values divided by the total values.

$$Accuracy = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1.1}$$

- **Specificity** is evaluating the model's ability to predict negative values and how often the model actually predicts the correct negative values; it is the true negative divided by the total number of actual negative values.

$$Specificity = \frac{TN}{N} = \frac{TN}{TN + FP} \tag{1.2}$$

- **Recall/Sensitivity** is used to measure the model's ability to predict positive values and how often the model actually predicts the correct positive values. it is calculated by dividing the true positives divided by the total number of actual positive values.

$$Recall = \frac{TP}{P} = \frac{TP}{TP + FN} \tag{1.3}$$

- **Precision** is measuring how well the predicted positive values are classified correctly. It answers the question "How often is the model correct when it predicts a positive value?". it is calculated by dividing true positives by the total number of predicted positive values.

$$Precision = \frac{TP}{TP + FP} \tag{1.4}$$

- **F-score or F-beta** is a metric to evaluate the model's performance; it is typically used for Imbalanced Classifications. It is useful when both precision and recall must be taken into account. Additional weights are applied, when either accuracy or recall is valued more highly than the other. Beta is a positive real value, commonly beta = 1, this is the so-called F_1-score when both recall and precision have the same importance (harmonic average), when the recall is more important than precision

then choose the beta value of more than 1, and choose a value less than 1 when precision is more important than recall.

$$F_\beta = (1 + \beta^2) \times \frac{Precision \times Recall}{\beta \times Precision + Recall} \tag{1.5}$$

When beta=1 (harmonic mean), then the score is called F_1-score and the formula becomes:

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = \frac{2 \times TP}{2 \times TP + FP + FN} \tag{1.6}$$

We used the above metrics to measure and evaluate the performance of the evaluated methods in this dissertation.

1.4.12 Graph Theory

Graph theory is the field of mathematics which deals with the study of graphs. The graph is a collection of points called vertices or nodes connected by lines which are so-called edges [67,68]. The nodes in the graph can represent any type of entity (e.g., cities, persons, companies, biomedical concepts, etc) and the edges represent the relationships between these entities. In graphs, it can be distinguished between directed graphs, when edges asymmetrically connect two vertices, and undirected graphs, when edges link two vertices symmetrically. The graph can be weighted by assigning a weight to the edges of the graph and unweighted when no weight is assigned. A graph can be represented by a so-called adjacency matrix [69–71].

1.4.12.1 Adjacency Matrix

Adjacency matrix is a two-dimensional array (V x V) where rows and columns are labeled by the name of the graph's vertices, i.e. the number of rows in the adjacency matrix is same as the number of columns equal to the number of vertices in the graph. In the unweighted graphs, the adjacency matrix stores 1 if the two vertices are connected (adjacent) otherwise 0, in the weighted graphs, the elements of the adjacency matrix indicate the weights of the

edges or 0 if the two vertices are not connected.

1.4.12.2 Sparse Matrix

Sparse matrix is a special type of matrix in which the proportion of zero entries to non-zero elements is significantly larger. For the large matrices in which most of the elements are zero, a sparse matrix offers efficient storage which can significantly reduce the amount of memory required for data storage by storing only the nonzero elements with their indices and ignoring the zero elements [72].

CHAPTER

MATERIALS AND METHODS 2

2.1 Tools and Programming packages

2.1.1 Python Programming Language

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics [73]. It is a very simple programming language and easy to learn. Python supports packages and modules in several domains. Python can be used for most kinds of programming tasks. All this makes python one of the most popular and used programming languages in several fields. Most of the scripts in this dissertation are programmed in python.

2.1.2 Django

Django is a free and open-source framework for building web applications with python. It is not the only web framework for python but it is the most popular one and helps to build a website in a short time with limited lines of code. Many companies such as youtube,

dropbox, Instagram, and Spotify use Django to build their web applications [74]. Django comes with a lot of features out of the box so they do not have to be coded from scratch. Additionally, it offers an admin interface for managing the data. It has an object-relational mapper which abstracts the database so queries can be sent without writing a lot of SQL code. It also offers an authentication package for identifying users and has a package for caching data. With all features which Django can offer, developers can focus on the application and its requirements without the need to code all these features from scratch [75]. The frontend and backend of the CPRiL web server were built using the Django framework.

2.1.3 PostgreSQL

PostgreSQL is a free and open-source object-relational database management system (RDBMS) in which the data is stored in the form of tables [76, 77]. It is the most advanced open-source database system, widely used in the development of backend systems. PostgreSQL includes capabilities like table inheritance and function overloading and offers a wealth of advanced features which can help in a more robust database for specific use cases. It includes support for popular programming languages like net javascript and python. Postgresql is used by several big technology companies such as Apple and Cisco where complex websites and applications require a highly customized database solution. The database of the CPRiL web server was built using PostgreSQL.

2.1.4 RDKit

RDKit is an open-source collection of cheminformatics and machine-learning software written in C++ and python. It's widely used for working with molecular data and analyzing the properties of chemical compounds [78]. RDKit has a molecular database cartridge for PostgreSQL and deals with several chemical properties such as SMILES and fingerprints. Furthermore, it generates 2D and 3D structures of the chemical compounds and calculates similarity searches [79]. The chemical compounds' properties and the 2D molecular structures stored in the CPRiL web server were generated using RDKit.

2.1.5 TensorFlow

TensorFlow is a complete open-source machine learning platform from Google. It was developed to deal with machine learning and deep learning applications on various data sets. TensorFlow integrates libraries and community resources which can be applied for creating and deploying powerful machine learning projects. TensorFlow applications can run on central processing units (CPUs), graphics processing units (GPUs), or tensor processing units (TPUs), which speed up TensorFlow jobs [80–86]. TensorFlow can be used in a variety of programming languages, including Python, C++, Java, and JavaScript. The deep learning method “BioBERT”, which was evaluated in this thesis and used to build the CPRiL web server, is using libraries of this platform.

2.1.6 NetworkX

NetworkX is a Python package for building, modifying, and researching the composition, dynamics, and purposes of complex networks. This module offers operations and functions for bipartite graphs. A bipartite graph or bigraph is a graph whose vertices can be decomposed into two distinct and independent sets U and V, every edge connects a vertex in U to one vertex in V, i.e. the connection between two vertices in the same set does not exist [87]. This package was used in this thesis for network visualization of the outputs and for calculating the shortest path between biomedical entities.

2.2 The Benchmark Dataset

2.2.1 Generation of the benchmark dataset for functional compound-protein relationships

Chemical compounds are referred to as small molecules up to a molecular weight of about 1,000 Dalton, for which a synonym and a related ID are contained in the PubChem database [88]. Similarly, genes and proteins must have UniProt synonyms and were assigned to related UniProt IDs [89]. PubChem synonyms were automatically annotated with

the approach described in the manuscripts about the web services Compounds In Literature (CIL) [90] and protein-literature investigation for interacting compounds (prolific) [91], by applying the rules described by Hettne et al. [92]. Proteins were annotated using the web service Whatizit [93]. The complete compound-protein interaction benchmark dataset (CPI-DS) was generated from the first 40,000 abstracts of all PubMed articles published in 2009.

All pairs of compounds and proteins co-occurring in a sentence are considered as potential functionally related or putative positive instances. Pairs with no functional relation were subsequently annotated as negative instances. If a named entity exists as a longform synonym and an abbreviated form in brackets, both terms are considered as individual entities. All sentences containing at least one compound-protein pair were transferred to an HTML form. An HTML annotation tool has been developed to help in the manual annotation and cross-check process of these annotations.

2.2.1.1 Pre-annotation

Annotation is a time-consuming and costly process. Without appropriate data preparation, an individual curator can read only a few papers per hour, which may or may not contain sentences of interest, i.e. sentences which don't contain relevant information. In the task of a functional compound-protein relationship, we looked for those sentences which have at least one compound-protein pair and excluded all other sentences. We used named entity recognition (NER) applications to pre-annotate a PubMed abstract with the entities under consideration, CIL and prolific to pre-annotate chemical compounds and Whatizit to pre-annotate proteins. This enabled us to extract only sentences containing relevant entities (chemical compounds and proteins). Filtering sentences in such a way reduces the total number of sentences and accelerates the process of manual annotation.

2.2.1.2 Manual Annotation Tool

A web-based annotation tool has been designed to assist the annotators in the manual annotation process and make the annotation task faster and easier (see Figure 2.1). The annotator can revise the highlighted entities and can mark the presence of a relationship

or not. The main components and features of the annotation tool are:

- The entities are pre-annotated automatically using named entity recognition (NER) tools. The entities are highlighted with different colors to easily distinguish between the type of each entity, green color for chemical compounds and purple for proteins/genes.
- Each entity is clickable and mapped to the corresponding database, compounds are mapped to the PubChem database, whereas proteins are mapped to the NCBI gene database. In the corresponding databases, it can be cross-checked by getting the full details about this entity.
- Each sentence is linked to the PubMed article where the full article can be observed to get a more detailed view onto the research topic to further support the annotation process.
- Each compound-protein pair is connected to a drop-down list on which the annotator can easily specify whether the pair is functionally related or not, or the pair is wrongly annotated. The drop-down list includes the following options: related, not related, wrong compound annotation, wrong protein annotation, both wrong, wrong segmentation (in case the sentence was not correctly split), and unclear (if the annotator can not make a decision about the existing of relationship or the correctness of the annotated entities).
- Each sentence which has been processed is highlighted as “checked”. This can help the annotator to distinguish between the sentences that are processed and those that are not.

2.2.1.3 Inter-annotation Agreement

The annotation process is performed based on specific rules. A compound-protein pair is functionally related if it appears in the same sentence and fulfills at least one of the following prerequisites:

4	17321121-703	The absorption spectrum of the hydrogenase enzyme showed an absorption peak at 425nm indicating that the enzyme had iron-sulfur clusters.	Interaction	Interaction	Push	Check
5	17403602-1067	When Vitreoscilla were grown in medium containing 60mM sodium nitrite under both normal and limited aeration conditions, the levels of Vitreoscilla hemoglobin (VHb) were decreased by greater than 90%, while the levels of the terminal respiratory oxidase, cytochrome bo, were increased 350% under normal aeration and 7-23% under limited aeration.	Interaction	Interaction	Push	Check
6	17590240-1233	Epigallocatechin gallate (EGCG) suppresses beta-amyloid-induced neurotoxicity through inhibiting c-Abl/FE65 nuclear translocation and GSK3 beta activation.	Interaction	Interaction	Push	Check
7	17590240-1233	Epigallocatechin gallate (EGCG) suppresses beta-amyloid-induced neurotoxicity through inhibiting c-Abl/FE65 nuclear translocation and GSK3 beta activation.	Interaction	Interaction	Push	Check
8	17590240-1233	Epigallocatechin gallate (EGCG) suppresses beta-amyloid-induced neurotoxicity through inhibiting c-Abl/FE65 nuclear translocation and GSK3 beta activation.	Interaction	Interaction	Push	Check
9	17590240-1234	Here, we used a human neuronal cell line MC65 conditional expression of an amyloid precursor protein fragment (APP-C99) to investigate the protection mechanism of epigallocatechin gallate (EGCG), the main constituent of green tea.	No interaction	No interaction	Push	Check
10	17590240-1234	Here, we used a human neuronal cell line MC65 conditional expression of an amyloid precursor protein fragment (APP-C99) to investigate the protection mechanism of epigallocatechin gallate (EGCG), the main constituent of green tea.	No interaction	No interaction	Push	Check
11	17616381-938	The protease was purified to homogeneity using ammonium sulfate precipitation, and ion exchange chromatography with a fold purification of 1.8 and a recovery of 49%.	No interaction	No interaction	Push	Check
12	17616381-938	The protease was purified to homogeneity using ammonium sulfate precipitation, and ion exchange chromatography with a fold purification of 1.8 and a recovery of 49%.	No interaction	No interaction	Push	Check
13	17629591-228	In addition, we demonstrated that Hsp20, HspB2 and HspB8 induced interleukin-6 production in cultured pericytes and astrocytes, which could be antagonized by dexamethasone, whereas other sHsps and A beta were inactive, suggesting that sHsps may be among the key mediators of the local inflammatory response associated with HCHWA-D and AD lesions.	Interaction	Interaction	Push	Check
14	17629591-228	In addition, we demonstrated that Hsp20, HspB2 and HspB8 induced interleukin-6 production in cultured pericytes and astrocytes, which could be antagonized by dexamethasone, whereas other sHsps and A beta were inactive, suggesting that sHsps may be among the key mediators of the local inflammatory response associated with HCHWA-D and AD lesions.	No interaction	No interaction	Push	Check
15	17629591-228	In addition, we demonstrated that Hsp20, HspB2 and HspB8 induced interleukin-6 production in cultured pericytes and astrocytes, which could be antagonized by dexamethasone, whereas other sHsps and A beta were inactive, suggesting that sHsps may be among the key mediators of the local inflammatory response associated with HCHWA-D and AD lesions.	No interaction	No interaction	Push	Check
16	17692997-193	Furthermore, dithiothreitol was found to be capable of significantly preventing the inhibitory effect of insulin on Abeta oligomer formation.	Interaction	Interaction	Push	Check
17	17719144-1171	The scavenger receptor, class B, type I (SR-BI) is critical in maintaining the homeostasis of cholesterol and alpha-tocopherol.	Wrong_annotation_protein	Wrong annotation protein	Push	Check
18	17719144-1171	The scavenger receptor, class B, type I (SR-BI) is critical in maintaining the homeostasis of cholesterol and alpha-tocopherol.	Interaction	Interaction	Push	Check
19	17719144-1171	The scavenger receptor, class B, type I (SR-BI) is critical in maintaining the homeostasis of cholesterol and alpha-tocopherol.	Interaction	Interaction	Push	Check
20	17719144-1171	The scavenger receptor, class B, type I (SR-BI) is critical in maintaining the homeostasis of cholesterol and alpha-tocopherol.	Wrong_annotation_protein	Wrong annotation protein	Push	Check
21	17719144-1172	SR-BI binds high-density lipoproteins (HDL) and mediates the selective transfer of cholesteryl esters and alpha-tocopherol from circulating HDL to cells.	No interaction	No interaction	Push	Check
22	17719144-1172	SR-BI binds high-density lipoproteins (HDL) and mediates the selective transfer of cholesteryl esters and alpha-tocopherol from circulating HDL to cells.	Interaction	Interaction	Push	Check
23	17719144-1173	Thus, SR-BI influences neural and cognitive processes, a finding that highlights the contribution of cholesterol and alpha-tocopherol homeostasis in proper cognitive function.	No interaction	No interaction	Push	Check
24	17719144-1173	Thus, SR-BI influences neural and cognitive processes, a finding that highlights the contribution of cholesterol and alpha-tocopherol homeostasis in proper cognitive function.	No interaction	No interaction	Push	Check
25	17768029-984	Alloxan is believed to confer its diabetogenic effect by inhibiting pancreatic glucokinase activity, leading to pancreatic beta-cell death.	Interaction	Interaction	Push	Check
26	17882129-602	Alpha-tocopherol supplementation prevents the exercise-induced reduction of serum paraoxonase 1/arylesterase activities in healthy	Interaction	Interaction	Push	Check

Figure 2.1: A Web-based annotation tool. The highlighted green entities represent the chemical compounds, and the purple represent the proteins. The annotation column shows the current status of the entities and the annotator's decision on the compound-protein pair.

- The entities (chemical compound and protein) of the candidate pair interact directly with each other.
- There is up or down-regulation of each other (directly or indirectly).
- The entities are part of each other.
- The small molecule is a cofactor of the protein.

All candidate pairs were annotated by eight different annotators. The entities' annotation and the relationship between them were at least proven by two different annotators. The inter-annotation agreement was performed in three stages:

1. In the first stage, an expert annotator performed the annotation for the whole corpus.
2. In the second stage, the corpus was distributed among six annotators to cross-check and go through all sentences. Unclear instances were left for the third stage, the unclear instances include unclear entities tagging or unclear relationships.

3. In the last stage, pairs which were either classified as “unclear” by one of the annotators or pairs which were classified differently by both annotators, the annotation instructor made the final decision.

2.2.2 Benchmark Dataset based on the Interaction Verb

To analyze how much specific interaction verbs, enclosed by compound and protein entities, affect the precision of functional relationships, we differentiated between sentences with or without this structure. According to this, the benchmark dataset “CPI-DS” has been split into two subsets. A dataset called “CPI-DS_IV” includes only compound-protein pairs which enclose an interaction verb, whereas the other dataset called “CPI-DS_NIV” includes those compound-protein pairs which don’t show this sentence structure. The interaction verbs which are enclosed by a compound-protein pair belong to a list of defined interaction verbs which were defined by Senger et al. [91] (see appendix C). Figure 2.2 shows detailed examples of the different types of functional compound-protein relationships based on interaction verbs.

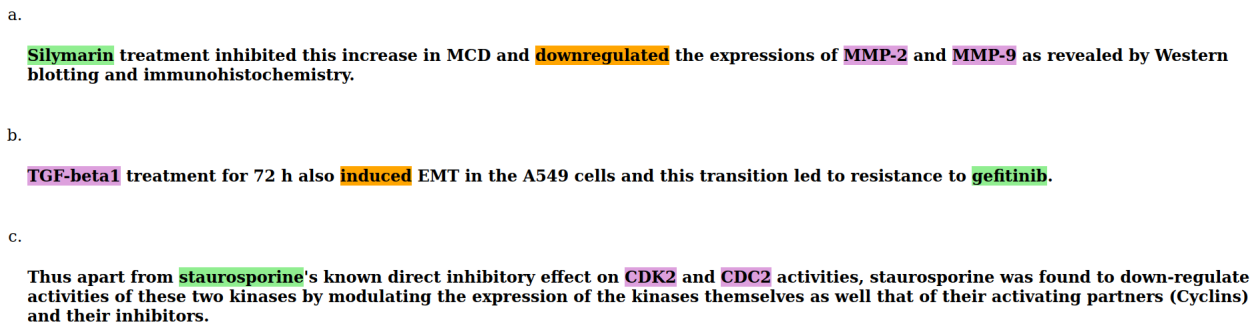


Figure 2.2: Types of functional compound-protein relationships based on interaction verbs. a) Direct functional relation with interaction verb. The orange colored verb is enclosed by the compound *“Silymarin”*, shown in green, and the proteins *“MMP-2”* and *“MMP-9”*, shown in purple [94]. The pair was annotated as functional. b) Indirect functional relation with interaction verb. *“TGF-beta1”* resistances *“gefitinib”* indirectly by inducing EMT in the A549 cells [95]. The pair was annotated as functional. c) Direct functional relation without interaction verb. The compound *“staurosporine”* has a direct inhibitory effect on the protein *“CDK2”*. This is indicated by the word *“inhibitory”* [96]. The pair was annotated as functional.

2.3 Functional Relationships Recognition Methods

Tikk et al. examined 13 kernel methods for protein-protein interaction extraction on different text corpora. Out of these methods, the all-paths graph kernel (APG) [97] and shallow linguistic kernel (SL) [98] consistently achieved very good results [99]. The APG kernel considers all weighted syntactic relationships in a sentence based on a dependency graph structure. In contrast, the SL kernel considers only surface tokens that come before, between, and after the potential interaction pair. Both kernels have been successfully applied in different domains, such as drug-drug interaction extraction [100]. A new deep learning pre-trained model (BioBERT) built on the basis of BERT has been introduced by Lee J et al. [5]. BioBERT is a pre-trained language representation model for the biomedical tasks; it achieved new state-of-the-art performances on most biomedical text mining tasks, including Named Entity Recognition (NER), Relation Extraction (RE), and Question Answering (QA).

We have evaluated the usability of the above three diverse machine learning methods (SL, APG, and BioBERT) which achieved good performance in the relation extraction (RE) domain for detecting functional and nonfunctional compound-protein relationships in texts.

2.3.1 Shallow Linguistic Kernel (SL)

Shallow Linguistic kernel, developed by Giuliano et al. [98], is a supervised machine learning approach for extracting relations between biomedical entities such as gene-protein and protein-protein from biomedical literature. SL is based on shallow linguistic information, such as tokenization, sentence splitting, Part-of-Speech (PoS) tagging, and lemmatization; these types of information can improve the performance of the relation extraction process. SL is a kernel-based approach which uses a Support Vector Machine (SVM) as a kernel algorithm. The main idea of kernel methods is instead of solving a complex non-linear problem, the input data is mapped into a higher feature space using a mapping function and then uses a linear algorithm to solve the problem linearly [98].

A shallow linguistic kernel uses two different information sources, global context, and local context. The shallow linguistic kernel is defined as the sum of a global and local context kernel. Each kernel is calculated as follows:

$$K(x_1, x_2) = \frac{\langle \phi(x_1), \phi(x_2) \rangle}{||\phi(x_1)||||\phi(x_2)||} \tag{2.1}$$

where ϕ is the embedding vector and || . || is the 2-norm. The kernel is normalized by the product of the norms of embedding vectors.

Global Context Kernel

The words which appear before, between, and after the candidate interacting entities are used to indicate the relationship between these two entities. There are three possible patterns for these tokens: Before-Between, Between, or Between-After. The global context kernel works on these patterns of words up to a length of n = 3 which is called n-gram. These n-grams are implemented using the bag-of-words approach. The method counts the number of occurrences of every word in a sentence including punctuation and stop words, but excludes the CANDIDATE and OTHER entities which includes the entities of interest (Figure 2.3). The patterns are computed regarding the phrase structures before-between, between, and between-after the considered entities. The global context kernel K_{GC} is defined as:

$$K_{GC} = K_{FB}(R_1, R_2) + K_B(R_1, R_2) + K_{BA}(R_1, R_2) \tag{2.2}$$

where K_{FB}, K_B, and K_{BA} are n-gram kernels which operate on the Fore-Between, Between and Between-After patterns respectively.

Local Context Kernel

The surrounding context of the candidate entities offers helpful information for determining the functions of the entities of the candidate pair within the relation. The local context kernel

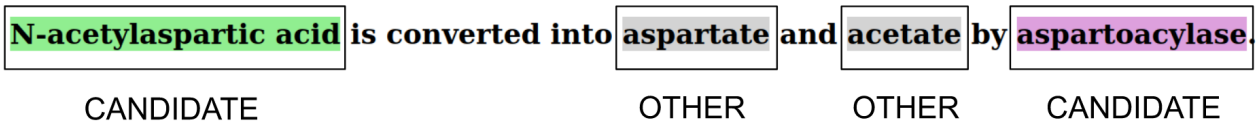


Figure 2.3: Representation of the sentence under shallow linguistic kernel. An example demonstrates four entities, “N-acetylaspartic acid” and “aspartoacylase” are the candidate related entities (CANDIDATE), and “aspartate” and “acetate” are not (OTHER).

considers tokens with their part-of-speech tags, lemmatization, capitalization, punctuation, and numerals [1, 98]. The left and right ordered word neighborhoods up to window size of w = 3 are considered in two separated kernels, which are summed up for each relationship instance. The local context kernel K_{LC} is defined as:

$$K_{LC} = K_{left}(R_1, R_2) + K_{right}(R_1, R_2) \tag{2.3}$$

Where K_{left} and K_{right} are left and right local kernels respectively.

Shallow Linguistic Kernel K_{SL} is defined by the combination of the global context kernel K_{GC} and local context kernel K_{LC} as follows:

$$K_{SL} = K_{GC}(R_1, R_2) + K_{LC}(R_1, R_2) \tag{2.4}$$

Shallow linguistic kernel uses a linear combination of kernels which has better performance than the individual ones.

2.3.2 All-paths Graph Kernel (APG)

All-paths graph kernel (APG) is a kernel-based machine learning method which employs graph data which uses the dependency graphs representing the sentence structure. APG kernel uses the parse regularized least squares (sparse RLS) kernel-based machine learning method [97]. The main idea of the APG method is to create a graph representation for the candidate compound-protein related pairs, then use the kernel function to measure the similarities of these graphs. A dependency parse of the sentence which includes a compound-protein pair as a candidate-related pair forms the input of the learning method of the APG kernel. The idea of APG is to create two unconnected, weighted, and direct

subgraphs; one represents the dependency structure of the sentence called dependency subgraph, and the other represents the linear order of the words in the sentence called linear order subgraph.

Dependency subgraph

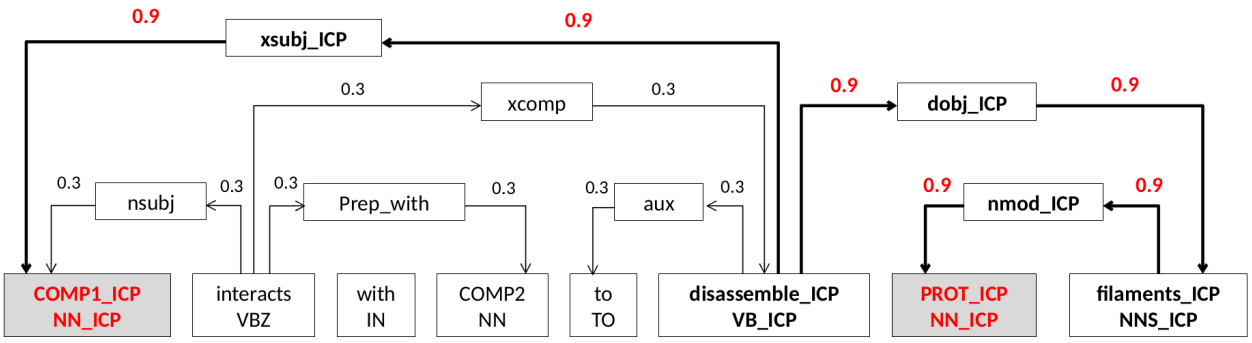
The dependency subgraph is built based on the dependency structure of the sentence. The vertices in the dependency graph represent the text tokens in the text (including the part-of-speech tag), and the edges represent the typed dependencies, showing the syntax of the sentence. For generalization, the candidate compound and protein entities are replaced with COMP and PROT respectively. The dependency's vertices are labeled with the type of dependency. The labels of the vertices on the shortest paths connecting the candidate entities are distinguished from the labels using a special tag. The highest emphasis is given to edges which are part of the shortest path connecting the candidate compound-protein pair by differentiating the labels of the vertices on the shortest paths using a special tag. In addition, a simple weighting scheme was chosen based on preliminary experiments, the edges on the shortest paths receive a weight of 0.9 and other edges receive a weight of 0.3 (Figure 2.4a).

Linear order subgraph

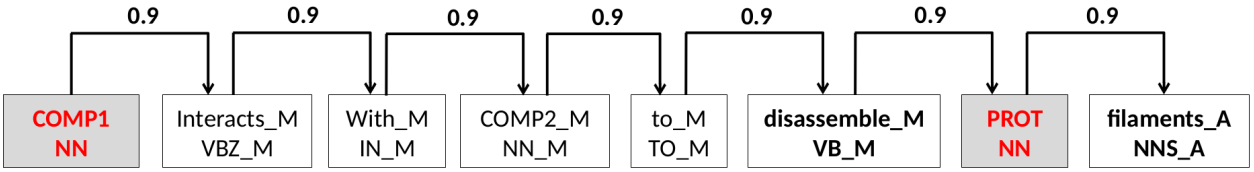
The linear order subgraph represents the linear structure of the sentence. Each token is represented by a vertex. The label of each vertex is derived from the texts, POS-tags, named entity tagging, and special tags representing the position of the token, before, in-between, or after the candidate compound-protein pair. Each vertex is connected to the next vertex by an edge which receives a weight of 0.9 (Figure 2.4b).

APG kernel implementation

Let V represents the set of vertices in the graph and L represents the set of labels. A graph



(a) The dependency subgraph.



(b) The linear order subgraph.

Figure 2.4: Graph representation of APG kernel.

can be represented in an adjacency matrix A $|V| \times |V|$. The entries in this matrix determine the weights of the connecting edges, the weight is zero if two vertices are not connected. Multiplication of the matrix with itself returns a new matrix with all summed weights of path length two.

All possible paths of all lengths can be calculated by computing the powers of the matrix. Matrix addition of all these matrices results in a final adjacency matrix, which consists of the summed weights of all possible paths [97].

$$(I - A)^{-1} = I + A + A^2 + ... = \sum_{k=0}^{\infty} A^k \tag{2.5}$$

Paths of length zero are removed by subtracting the identity matrix I.

$$W = (I - A)^{-1} - I \tag{2.6}$$

All labels are represented as a feature vector. The feature vector is encoded for every vertex, containing the value 1 for labels which are presented within this particular node. This

results in a label allocation matrix $L \in |l| \times |V|$, $L_i j = 1$ if the j-th vertex has the i-th label, otherwise $L_i j = 0$. A feature matrix as defined by Gärtner et al. sums up all weighted paths with all presented labels [101]. This calculation combines the strength of the connection between two nodes with the encoding of their labels. In general, it can be stated that the dependency weights are higher the shorter their distance to the shortest path between the candidate entities is [1]. The similarity of two feature matrix representations can be computed by summing up the products of all their entries [97]. In the implementation used here [1, 97], the regularized least squares classifier algorithm is applied to classify compound-protein relationships with the APG kernel. This classifier is similar to a standard support vector machine (SVM), but the underlying mathematical problem does not need to be solved with quadratic programming [97, 102].

$$f(x_*) = \sum_{i=1}^{b} a_i k(x_*, x_i) \tag{2.7}$$

x_* is the given text input, k is the kernel function, x_i are training data points, a_i are weights, and b is the size of $B \subset M$ (the training set), B is selected randomly in advance.

The Input format of the SL and APG

The input format for shallow linguistic and all-paths graph kernels is provided as XML format (Figure 2.5). This format includes the following main data:

- **Document (Article)**: This section includes a unique document ID and PubMed ID.
- **Sentence**: This section of the file includes a unique sentence ID and sentence text which includes the candidate compound-protein pair.
- **Entities**: This section of the file includes a unique entity ID, offset, entity type (compound or protein), and entity name
- **Pair**: This section includes a unique pair ID, Identifiers of the candidate compound-protein pair, and the type of the relation, the default is false (not functionally related).

```
<document id="DS.ds" origId="17719144">
 <sentence id="DS.ds.s0" origId="17719144-1" text="The scavenger receptor, class B, type I (SR-BI) is critical in maintaining the
  homeostasis of cholesterol and alpha-tocopherol.">
   <entity id="DS.ds.s0.e1" origId="O18824,P97943,Q60417,Q61009,Q8SQC1,Q8WTV0" charOffset="41-46" type="protein" text="SR-BI"/>
   <entity id="DS.ds.s0.e2" origId="304,5997,6432564,11025495" charOffset="94-105" type="compound" text="cholesterol"/>
   <entity id="DS.ds.s0.e3" origId="2116,14985" charOffset="110-126" type="compound" text="alpha-tocopherol"/>
   <pair e1="DS.ds.s0.e3" e2="DS.ds.s0.e1" id="DS.ds.s0.i1" interaction="True"/>
   <pair e1="DS.ds.s0.e2" e2="DS.ds.s0.e1" id="DS.ds.s0.i2" interaction="True"/>
  </sentence>
 <sentence id="DS.ds.s1" origId="17719144-2" text="SR-BI binds high-density lipoproteins (HDL) and mediates the selective transfer of
  cholesteryl esters and alpha-tocopherol from circulating HDL to cells.">
   <entity id="DS.ds.s1.e0" origId="O18824,P97943,Q60417,Q61009,Q8SQC1,Q8WTV0" charOffset="0-5" type="protein" text="SR-BI"/>
   <entity id="DS.ds.s1.e1" origId="P81182" charOffset="12-37" type="protein" text="high-density lipoproteins"/>
   <entity id="DS.ds.s1.e2" origId="2116,14985" charOffset="106-122" type="compound" text="alpha-tocopherol"/>
   <pair e1="DS.ds.s1.e2" e2="DS.ds.s1.e1" id="DS.ds.s1.i0" interaction="False"/>
   <pair e1="DS.ds.s1.e2" e2="DS.ds.s1.e0" id="DS.ds.s1.i1" interaction="True"/>
  </sentence>
 <sentence id="DS.ds.s2" origId="17719144-3" text="Thus, SR-BI influences neural and cognitive processes, a finding that highlights the
  contribution of cholesterol and alpha-tocopherol homeostasis in proper cognitive function.">
   <entity id="DS.ds.s2.e0" origId="O18824,P97943,Q60417,Q61009,Q8SQC1,Q8WTV0" charOffset="6-11" type="protein" text="SR-BI"/>
   <entity id="DS.ds.s2.e1" origId="304,5997,6432564,11025495" charOffset="101-112" type="compound" text="cholesterol"/>
   <entity id="DS.ds.s2.e2" origId="2116,14985" charOffset="117-133" type="compound" text="alpha-tocopherol"/>
   <pair e1="DS.ds.s2.e1" e2="DS.ds.s2.e0" id="DS.ds.s2.i0" interaction="False"/>
   <pair e1="DS.ds.s2.e2" e2="DS.ds.s2.e0" id="DS.ds.s2.i1" interaction="False"/>
  </sentence>
</document>
```

Figure 2.5: The input XML format of SL and APG kernel.

2.3.3 BioBERT

BioBERT is a deep learning and domain-specific model based on BERT. BioBERT is pre-trained on the BERT model corpora (English Wikipedia + CorpusBook) and large-scale biomedical corpora extracted from PubMed abstracts and PMC. The resulting model outperformed the BERT model in biomedical text-mining domains [5].

BioBERT approach

The BioBERT approach consists of two main phases: pre-trained and fine-tuning.

1. **Pre-trained**

   BioBERT is pre-trained on two different sets of data:

   (a) BioBERT is initialized from BERT which is pre-trained on 2,500 million English Wikipedia words and 800 million words extracted from BookCorpus which consists of more than 11,000 unpublished books from 16 different disciplines. Instead of random initialization of weights, BioBERT used the pre-trained weights from the BERT model.

   (b) Next, the BioBERT was pre-trained again but this time on domain-specific corpora. It was pre-trained on biomedical data extracted from PubMed abstracts

with 4.5 billion words and PMC full-text articles with 13.5 billion words.

Table 2.1, shows the different text corpora used for pre-trained BioBERT, however, Table 2.2, shows the pre-trained combinations of the models that BioBERT offered.

2. **Fine-tuning**

   In this phase, the BioBERT model was fine-tuned on biomedical domain-specific tasks such as the relation extraction task (RE) or name-entity recognition task (NER). The interesting part is that the pre-training is not solely on biomedical corpora, but rather on various combinations of general and biomedical corpora. The pre-training on a combination of different datasets gives the model a better performance when it is applied to biomedical tasks.

Table 2.1: List of text corpora used for BioBERT.

Corpus	Abbreviation	Number of words (billion)	Domain
English Wikipedia	Wiki	2.5	General
BookCorpus	Books	0.8	General
PubMed Abstracts	PubMed	4.5	Biomedical
PMC Full-text articles	PMC	13.5	Biomedical

Table 2.2: Corpus combination of the pre-trained BioBERT models.

Model	Corpus combination
BioBERT (+PubMed)	Wiki + Books + PubMed
BioBERT (+PMC)	Wiki + Books + PMC
BioBERT (+PubMed + PMC)	Wiki + Books + PubMed + PMC

As in the BERT model, BioBERT has two main pre-trained weights depending on the size of the trained corpus, BioBERT-Base, and BioBERT-Large.

The input format of BioBERT

In order to use BioBERT, the input data must be in a tsv format (tsv for tab-delimited values), the columns of the input file are given below:

- **Column 1**: A Sequential number representing a unique ID for the candidate pair.
- **Column 2**: The sentence's text which includes the candidate compound-protein pair. The compounds and protein names are masked with @COMPOUND$ and @PROTEIN$ respectively.
- **Column 3**: The label of the candidate pair, 1 if the candidate pair is functionally related and 0 if not functionally related.

2.4 Large-scale Dataset Analysis

On the related benchmark (CPI-DS), the predictive model was applied on all titles and abstracts of the biomedical articles stored in the PubMed database. Named Entity Recognition (NER) was applied to annotate small molecules and proteins. The sentences which did not include a compound-protein pair were excluded and the others were kept. Processing these annotations in combination with the Relation Extraction (RE) method allows for a complete automatic annotation of functional compound-protein relations in texts. The classification results as well as related data about the compounds and proteins were transferred into a relational database. Out of this database, we built a web server (CPRiL) for exploring functional compound-protein relationships which were extracted from PubMed literature.

2.4.1 CPRiL Web Server Implementation

A web-based service (CPRiL) for exploring the functional compound-protein relationships which were extracted automatically from the biomedical and life sciences literature (PubMed) was developed.

2.4.1.1 CPRiL Pipeline

The CPRiL pipeline is a fully automatic classification pipeline, it was developed based on a machine learning method trained on the above-mentioned benchmark (CPI-DS). The CPRiL Pipeline (Figure 2.6) has four main steps:

1. **Entities Annotation**

   Chemical compounds and proteins/genes in biomedical and life sciences articles (PubMed) were tagged by using the NCBI PubTator Central web service (PTC) [3]. In addition, each entity was provided with a unique identifier. Chemical compounds were mapped and linked to MeSH [103] and PubChem [88] if related IDs could be identified automatically. Proteins were mapped and linked to a GeneID [104] and UniProt IDs [89].

   PubTator Central Web service (PTC)

   PTC is a web-based service developed by NCBI which provides automatic named entity recognition of biomedical concepts such as chemicals, genes, diseases, and species in biomedical and life sciences articles [3]. PubTator applies machine learning techniques for the automatic recognition of biomedical entities. It provides the entities annotation for the entire PubMed articles and most of PubMed Central (PMC) full-text articles. It is available through both web and API access. Table 2.3 lists the taggers, training/evaluation corpus, and the performance of each tagger for chemical compounds and genes/proteins of PTC [3].

2. **Sentence Segmentation**

   Sentence segmentation is a task in natural language processing (NLP) of indicating the boundaries of the sentences in a text. It is a problem of dividing a written text into its meaningful sentences, so that downstream entities' relationships can happen at the sentence level. Generally, the languages use punctuation marks, particularly the full stop character is used to segment the text into sentences. In practice, sentence and word segmentation cannot be done properly independent from each other.

Table 2.3: Performance of PubTator Central (PTC). Evaluation results are reported by precision, recall, and F_1-score.

			Performance (%)		
Tagger	**Concept type**	**Training/ evaluation corpus**	**Precision**	**Recall**	**F_1**
TaggerOne [105]	Chemical	BioCreative V CDR [106]	88.8	90.3	89.5
GNormPlus [107]	Gene	BioCreative II GN [108]	87.1	86.4	86.7

Because a period may be used to signal an abbreviation as well as the end of a sentence in English, the distinction between the abbreviation and sentence boundary becomes an essential task. When an abbreviation appears at the end of a sentence and the period represents both the abbreviation and the sentence boundary, sentence segmentation becomes even more difficult.

In this step, the entire document was broken down, or "segmented", into constituent sentences. This "segmentation" was done throughout the article based on full stops. The Punkt sentence tokenizer of the Natural Language Toolkit (NLTK) data package which includes a pre-trained Punkt tokenizer for English was applied for sentence segmentation [109]. This tokenizer splits a text into a list of sentences by using an unsupervised algorithm. Next, sentences which have compound-protein pairs were kept with information such as PMID and position of the sentence in the text. All other sentences were excluded.

3. **Classification**

For the classification of functional/non-functional relations, the text mining model Bidirectional Encoder Representations from Transformers for Biomedical Text Mining (BioBERT, [5]) was applied. The pre-trained deep learning model is an adapted version of BERT which was trained on the English version of Wikipedia and CorpusBook to predict masked tokens (e.g. hidden words) from the context (e.g. related sentence or sequence of sentences) in texts [33]. BioBERT was further trained on the biomedical corpora PubMed abstracts and PMC full texts. Finally, BioBERT was trained on the benchmark dataset of functional compound-protein relationships described

above (CPI-DS) and applied for classification.

4. **Data Transition**

   Finally, all compound-protein functionally related pairs were transferred into a relational database (CPRiL database). In addition, the database was extended to include the related information (if available), namely:

   - **Article information**: PubMed ID (PMID), title, journal name, and publishing date.
   - **Chemical Compound information**: Mesh ID, PubChem ID, molecular structure, molecular formula, SMILES, InChI, compound synonyms, and more.
   - **Protein information**: Gene ID, Uniprot entry name, organism, protein synonyms, and more.

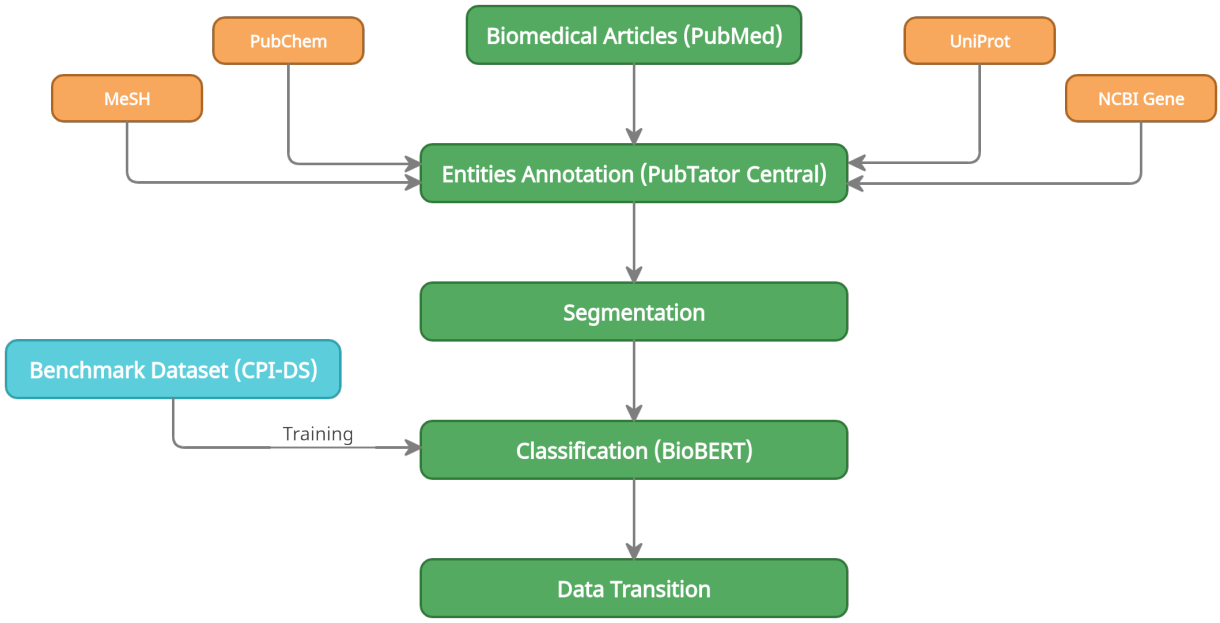


Figure 2.6: The CPRiL pipeline.

2.5 Shortest Path between Biomedical Entities

In graph theory, the shortest path is a problem of finding the shortest path between two vertices, source and destination, such that the sum of the weights of the edges is mini-

mum [110, 111]. There are different types of algorithms which can solve the shortest path problem. Breadth-First Search (BFS) algorithm [112] calculates the shortest path of the unweighted graph where the distances between any two vertices in the graph are the same, i.e. the graph has unweighted edges. For the weighted graph with negative edges, Bellman-Ford's algorithm [113, 114] is used to find the shortest path, whereas Dijkstra's algorithm [115] is the option when the graph is weighted with no negative edges. To find the shortest path between biomedical entities, a non-negative weighted graph was generated, and the best algorithm to calculate the shortest path for this type of graph was applied (Dijkstra's algorithm).

2.5.1 Dijkstra's Algorithm

Dijkstra's algorithm is used to calculate the shortest path from the source vertex to all other vertices in a non-negative weighted graph [115]. In the compound-protein relationship problem, each compound and protein is represented by a vertex, a compound vertex is connected to a protein vertex with an undirected weighted edge if they are functionally related, and the edge is weighted by the number of articles where this relationship is mentioned.

The algorithm

1. Set the distance of all vertices equal to infinity except zero for the source vertex.

2. Create a min-priority queue (Min Heap) of size V, where V is the number of vertices in the given graph. A min-priority queue is a queue in which priority is given to the element with minimum value.

3. Push all vertices into the Min Heap. The vertex in the Min Heap has the structure (vertex, distance), where vertex is the vertex's name and distance is the shortest distance from the source to this vertex.

4. While the Min Heap is not empty, do the following:

(a) gets the vertex with minimum distance value from the Min Heap. Let the extracted vertex be A.

(b) for every adjacent vertex B of A, update the distance value of B, if B is in Min Heap and its distance value is greater than the distance value of A plus the weight of the connected edge between A and B.

The time complexity of Dijkstra's algorithm is $O(|E| + |V|logV)$ using the Fibonacci heap min-priority queue, where V is the number of vertices, and E is the number of edges. The complexity of the space is $O(V)$ [116].

CHAPTER

3 RESULTS AND EVALUATION

3.1 Analysis of the Benchmark Datasets

The generation of the benchmark dataset (CPI-DS) resulted in a corpus of 2,613 sentences containing at least one compound-protein pair (CPI pair). Furthermore, this dataset was divided into two datasets based on the presence of an interaction verb: CPI-DS_IV includes 1,209 sentences which have candidate compound-protein pairs enclosed by an interaction verb, and CPI-DS_NIV has 1,404 sentences where the candidate compound-protein pairs don't have this structure. Table 3.1 shows the statistical information of each benchmark dataset: the unique number of compounds, the unique number of proteins, the number of positives (functionally related compound-protein pairs), the number of negatives (non-functionally related compound-protein pairs), and the total number of compound-protein pairs.

Table 3.1: Statistical information of CPI-DS, CPI-DS_IV, and CPI-DS_NIV.

Dataset	# Unique compound names	# Unique protein names	# Positive pairs	# Negative pairs	Total num. of pairs
CPI-DS	1,320	1,545	2,931	2,631	5,562
CPI-DS_IV	787	865	1,598	1,269	2,867
CPI-DS_NIV	775	990	1,333	1,362	2,695

3.1.1 Structure of the CPI-DS Benchmark Dataset

Within all sentences of the benchmark dataset (CPI-DS), a total number of 5,562 compound-protein pairs were curated with 2,931 functionally related compound-protein pairs (positive instances) and 2,631 non-functionally related (negative instances). For the evaluation process, the benchmark dataset (CPI-DS) was split into two datasets: 70% training dataset with 3,894 compound-protein pairs and 30% test dataset with 1,668 pairs (Table 3.2). All compound-protein pairs of one document or article were categorized into training dataset or test dataset.

Table 3.2: Number of positive and negative instances in the training and test datasets of benchmark dataset (CPI-DS).

Dataset		# Positives	# Negatives	Total
CPI-DS	**Training dataset**	2,023	1,871	3,894
	Test dataset	908	760	1,668

3.1.2 Relevance of Interaction Verbs

Subsequently, we analyzed the impact of interaction verbs on the classification. The independence of functional relationships and the presence of an interaction verb was tested with a chi-squared test. The chi-square statistic was 21.95, with a p-value < 0.00001. This test shows that both characteristic features are not independent of each other ($p<0.01$). The fraction of sentences containing an interaction verb is higher in the functionally related CPI-pairs (Figure 3.1).

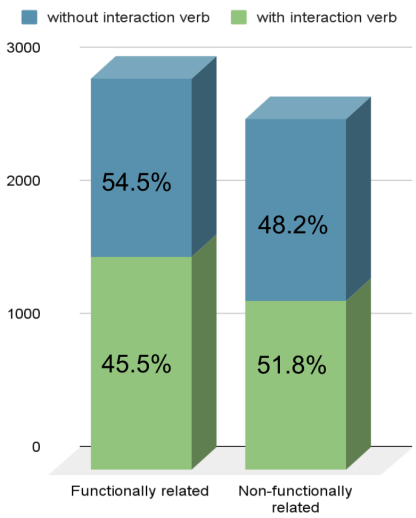


Figure 3.1: Ratio of functional and non-functional compound-protein related pairs in the benchmark dataset with and without interaction verbs.

To check whether and how the different classification methods make use of this correlation, we divided the CPI-DS into two subsets: CPI-DS_IV and CPI-DS_NIV. CPI-DS_IV includes compound-protein pairs which enclosed an interaction verb, whereas CPI-DS_NIV includes compound-protein pairs which do not show this structure, i.e. no interaction verb enclosed. Table 3.3 shows the number of functionally and non-functionally related pairs in each dataset.

Table 3.3: Number of functionally and non-functionally related instances of datasets CPI-DS_IV and CPI-DS_NIV.

Dataset	Functionally related	Non-functionally related	Total
CPI-DS_IV	1,598	1,269	2,867
CPI-DS_NIV	1,333	1,362	2,695
Total	2,931	2,631	5,562

For evaluation, each benchmark dataset was split into two datasets: 70% training dataset and 30% test dataset, Table 3.4 shows the number of functionally (positive) and non-functionally (negative) related compound-protein pairs. All compound-protein pairs of one document or article were distributed into training or test dataset but not both.

Figure 3.2 demonstrates the percentage of functionally (positive) and non-functionally (neg-

Table 3.4: Number of positive and negative instances in the training and test datasets CPI-DS_IV and CPI-DS_NIV.

Dataset		# Positives	# Negatives	Total
CPI-DS_IV	**Training dataset**	1,104	903	2,007
	Test dataset	494	366	860
CPI-DS_NIV	**Training dataset**	919	968	1,887
	Test dataset	414	394	808

ative) related instances of both training and test datasets of the benchmark datasets. The percentage of positive instances of the training dataset ranged from 49% to 55%, negatives of training ranged from 44% to 51%, positives of the test dataset ranged from 51% to 57%, and negatives of the test dataset ranged from 43% to 49%.

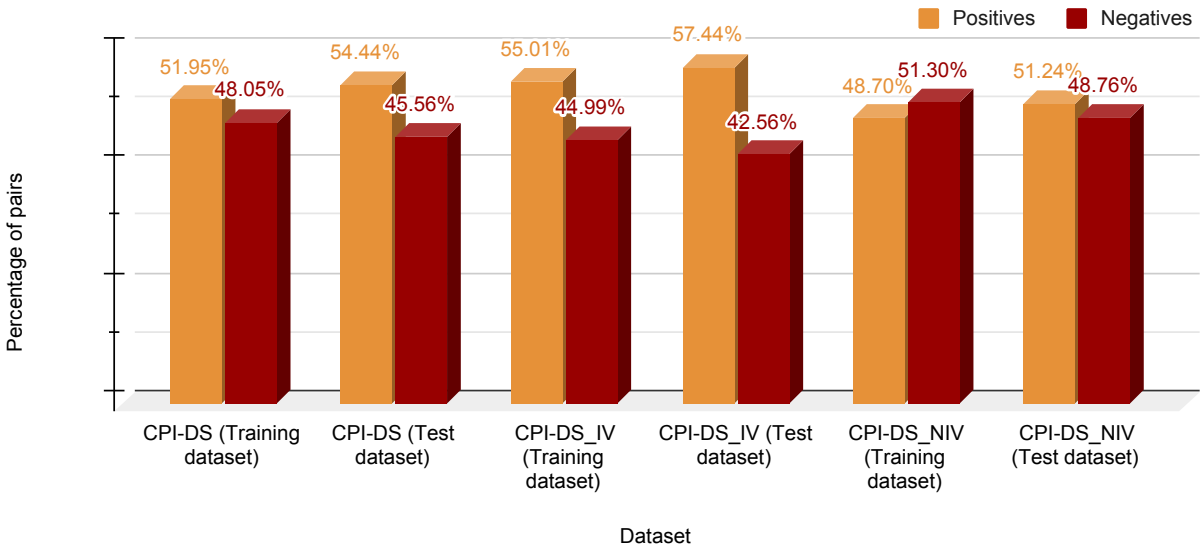


Figure 3.2: Percentage of functionally (positive) and non-functionally (negative) related instances of training and test datasets of the benchmark dataset.

3.2 Baseline Analysis

We considered co-occurrences as a simple approach to calculate the baseline in the way that every appearance of a compound and a protein in a sentence is classified as a functional relationship (recall 100%, specificity 0%), taking into account the number of all true

functional relationships. Figure 3.3 shows the confusion matrix of the prediction approach of co-occurrences for the test dataset of the combined benchmark dataset (CPI-DS) where all pairs are predicted as functionally related (positive). Table 3.5 demonstrates the performance of this approach, the sensitivity is 100% and the specificity is 0.0% by definition. It results in a precision (equal to accuracy because there are no true and false negative predictions) of 54.4% and an F_1-score of 70.5%.

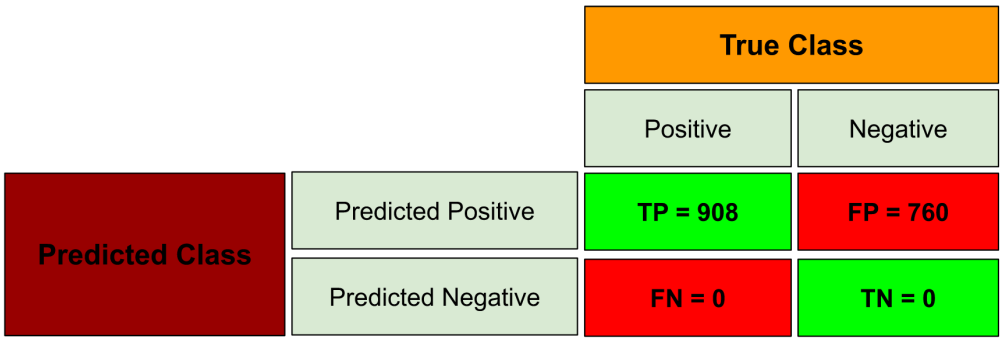


Figure 3.3: Confusion matrix of the prediction approach of co-occurrences.

Table 3.5: Analysis of the CPI-DS benchmark dataset using co-occurrences approach.

DS	#Sent.	#CPIs	#No-CPIs	#Pairs	Rec.	Spec.	Prec.	Acc.	F_1
CPI-DS	795	908	760	1,668	100.0	0.0	54.4	54.4	70.5

Baseline results for precision, recall, and F_1-score based on simple co-occurrences. Results are shown in percent (DS—dataset, Sent.—sentences, Rec.—recall, Spec.—specificity, Prec.—precision, F_1—F_1-score)

Table 3.6 shows the baseline results of CPI-DS_IV and CPI-DS_NIV datasets by using simple co-occurrences approach. In both datasets, the baseline achieves an F_1-score of 73.0% and 67.8% for CPI-DS_IV and CPI-DS_NIV, respectively.

Table 3.6: Analysis of the CPI-DS_IV and CPI-DS_NIV dataset using co-occurrences approach.

DS	#Sent.	#CPIs	#No-CPIs	#Pairs	Rec.	Spec.	Prec.	Acc.	F_1
CPI-DS_IV	346	494	366	860	100.0	0.0	57.4	57.4	73.0
CPI-DS_NIV	449	414	394	808	100.0	0.0	51.2	51.2	67.8

Baseline results for precision, recall, and F_1-score based on simple co-occurrences. Results are shown in percent (DS—dataset, Sent.—sentences, Rec.—recall, Spec.—specificity, Prec.—precision, F_1—F_1-score)

3.3 Evaluation of the Predictive Methods

The hyperparameter optimization process was performed for a range of hyperparameters on the training dataset to select the best hyperparameters of the predictive model using 10-fold cross-validation. After that, each predictive model (SL, APG, and BioBERT) has been evaluated individually using the selected hyperparameters with the same validation splits of training and test splits (described in sections 3.1.1 and 3.1.2) to have a fair evaluation for each method using holdout cross-validation. Subsequently, the evaluation analyzed the effect of the interaction verb's presence on the classification's performance; the predictive methods were applied individually to the dataset which includes candidate compound-protein pairs enclosed by interaction verb (CPI-DS_IV) and the dataset which does not have this structure (CPI-DS_NIV).

3.3.1 Shallow Linguistic Kernel (SL)

All parameter combinations in the range 1-4 for both n-gram and window size of the SL kernel were evaluated. The selection of n-gram=3 and window size=1 shows the best F_1-score, AUC value, and the highest precision in comparison to all other models with F_1-score of 77.4%, AUC of 80.5%, precision of 73.3%, and recall of 81.8% (Table 3.7). In general, a lower value of window size leads to higher precision, AUC, specificity, and accuracy and a lower recall (Figure 3.4); on the other hand, a higher value of n-gram leads to higher the overall performance (Figure 3.5). The results show the performance of the kernel does not change when the n-gram exceeds 3.

For both datasets (CPI-DS_IV and CPI-DS_NIV), the parameter selection n-gram 3 and window size 1 shows the highest area under the curve value (AUC). The results showed a lower value of window size leads to a higher precision and a lower recall. In general, the SL kernel performs slightly better in distinguishing between functional and non-functional relations on dataset CPI-DS_IV; just the recall performs differently, where the recall on dataset CPI-DS_NIV is better (Table 3.8 and Table 3.9). In general, the SL kernel performs better on sentences with an interaction verb, which clearly shows that the presence of the interaction verb in the sentence helps in the recognition of the compound-protein

Table 3.7: 10-fold CV performance of SL kernel on the dataset CPI-DS.

n	w	Recall	Specificity	Precision	Accuracy	F_1	AUC
1	1	75.6	66.2	71.5	71.0	73.0	78.7
1	2	86.3	54.8	68.0	71.3	75.7	78.7
1	3	88.0	50.1	66.1	69.8	75.2	78.2
1	4	88.2	46.7	64.7	68.3	74.3	77.5
2	1	76.5	67.3	72.5	72.0	74.0	79.7
2	2	84.7	57.6	69.0	71.7	75.6	79.1
2	3	85.2	57.0	68.8	71.7	75.7	79.0
2	4	86.8	52.5	67.0	70.4	75.3	78.6
3	**1**	**81.8**	**68.2**	**73.4**	**73.3**	**77.4**	**80.5**
3	2	86.1	57.6	69.4	72.5	76.5	80.0
3	3	85.4	57.2	69.0	72.0	76.0	79.8
3	4	87.1	54.0	67.8	71.3	75.9	79.4
4	1	81.8	68.2	73.4	73.3	77.4	80.5
4	2	86.1	57.6	69.4	72.5	76.5	80.0
4	3	85.4	57.2	69.0	72.0	76.0	79.8
4	4	87.1	54.0	67.8	71.3	75.9	79.4

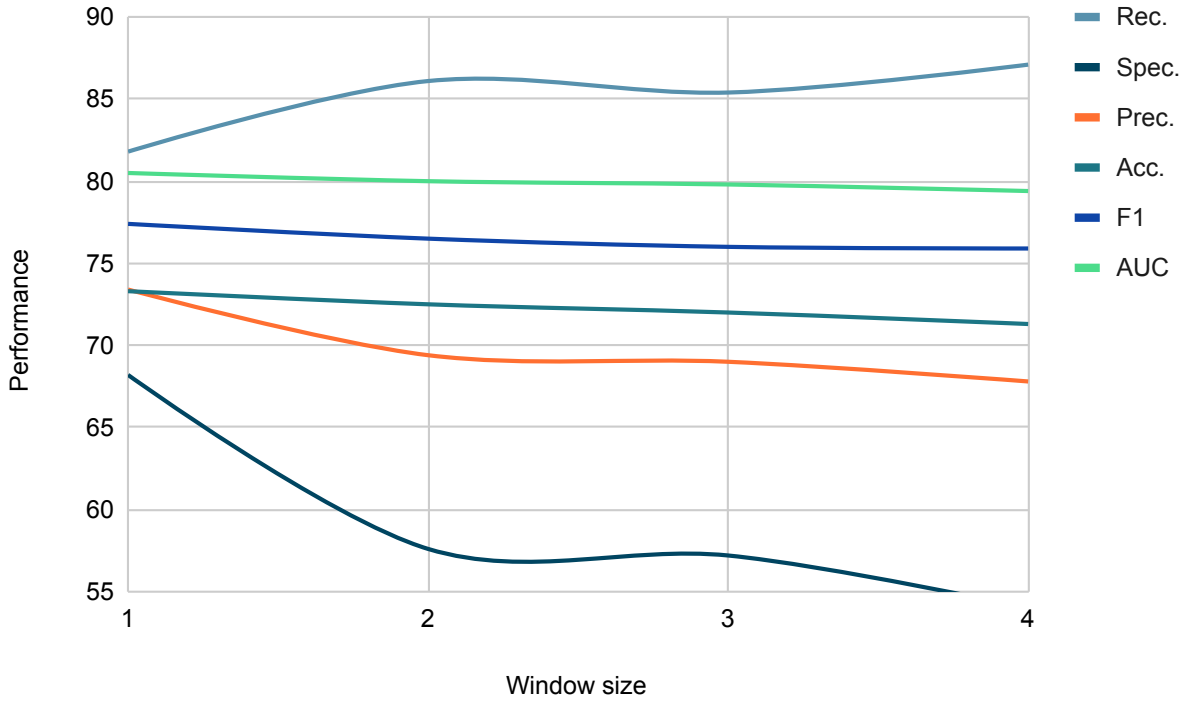


Figure 3.4: Effect of the window size parameter (w) on the performance of the model using shallow linguistic kernel (SL).

relationship.

Table 3.10 shows the holdout cross-validation performance of the predictive model of SL kernel on the unseen test datasets (described in sections 3.1.1 and 3.1.2). SL cross-validation achieved performance with F_1-score of 79.3, precision of 75.8, and AUC of 83.1. The evaluation process shows slight superiority of the prediction model on the dataset that includes candidate compound-protein pairs surrounded by interaction verb (CPI-DS_IV) over the dataset which does not have this structure (CPI-DS_NIV).

3.3.2 All-paths Graph Kernel (APG)

We evaluated the APG kernel using the same validation splits as for the SL kernel. The results shown in Table 3.11 indicate that models achieve almost similar performance with F_1-score of $\sim$77.7%, AUC of $\sim$83.8%, recall of $\sim$79.0%, and precision of $\sim$76.7% independent of the hyperplane optimization parameter c which had values of 0.25, 0.5, 1.0, and 2.0. Mathematically, a larger generalization parameter c represents a lower risk of

Table 3.8: 10-fold CV performance of SL kernel on the dataset CPI-DS_IV.

n	w	Recall	Specificity	Precision	Accuracy	F_1	AUC
1	1	75.5	67.0	73.9	72.2	74.5	77.9
1	2	80.5	61.0	71.6	72.1	75.6	78.0
1	3	82.2	59.9	71.7	72.6	76.4	77.8
1	4	84.6	55.5	70.2	72.0	76.6	77.4
2	1	74.6	71.2	76.3	73.5	75.2	80.2
2	2	79.2	65.5	73.9	73.4	76.2	79.6
2	3	78.7	64.3	73.1	72.7	75.7	79.2
2	4	80.7	62.0	72.4	72.8	76.2	79.2
3	1	74.8	71.9	76.8	73.9	75.7	80.7
3	2	78.6	66.6	74.4	73.6	76.3	80.2
3	3	78.7	65.6	74.0	73.3	76.1	79.7
3	4	80.6	63.6	73.3	73.4	76.6	79.7
4	1	74.8	71.9	76.8	73.9	75.7	80.7
4	2	78.6	66.6	74.4	73.6	76.3	80.2
4	3	78.7	65.6	74.0	73.3	76.1	79.7
4	4	80.6	63.6	73.3	73.4	76.6	79.7

Table 3.9: 10-fold CV performance of SL kernel on the dataset CPI-DS_NIV.

n	w	Recall	Specificity	Precision	Accuracy	F_1	AUC
1	1	74.8	69.9	70.5	72.2	72.1	78.8
1	2	82.1	63.0	68.1	72.3	73.9	78.6
1	3	82.0	59.9	66.2	70.6	72.7	77.7
1	4	80.4	60.8	66.4	70.3	72.2	76.8
2	1	77.1	69.2	70.7	72.8	73.2	80.2
2	2	83.0	62.7	68.1	72.3	74.2	79.5
2	3	84.1	61.5	67.8	72.3	74.5	79.3
2	4	82.2	61.4	67.4	71.4	73.5	78.7
3	1	77.8	69.4	71.1	73.3	73.8	80.8
3	2	84.3	62.3	68.3	72.8	74.9	80.2
3	3	84.6	60.7	67.7	72.1	74.5	80.1
3	4	83.8	59.3	66.7	71.0	73.6	79.3
4	1	77.8	69.4	71.1	73.3	73.8	80.8
4	2	84.3	62.3	68.3	72.8	74.9	80.2
4	3	84.6	60.7	67.7	72.1	74.5	80.1
4	4	83.8	59.3	66.7	71.0	73.6	79.3

Table 3.10: Holdout CV performance of SL kernel on the benchmark dataset.

	Recall	Specificity	Precision	Accuracy	F_1	AUC
CPI-DS	83.3	68.2	75.8	76.4	79.3	83.1
CPI-DS_IV	77.1	77.3	82.1	77.2	79.5	84.4
CPI-DS_NIV	81.4	67.5	72.5	74.6	76.7	82.4

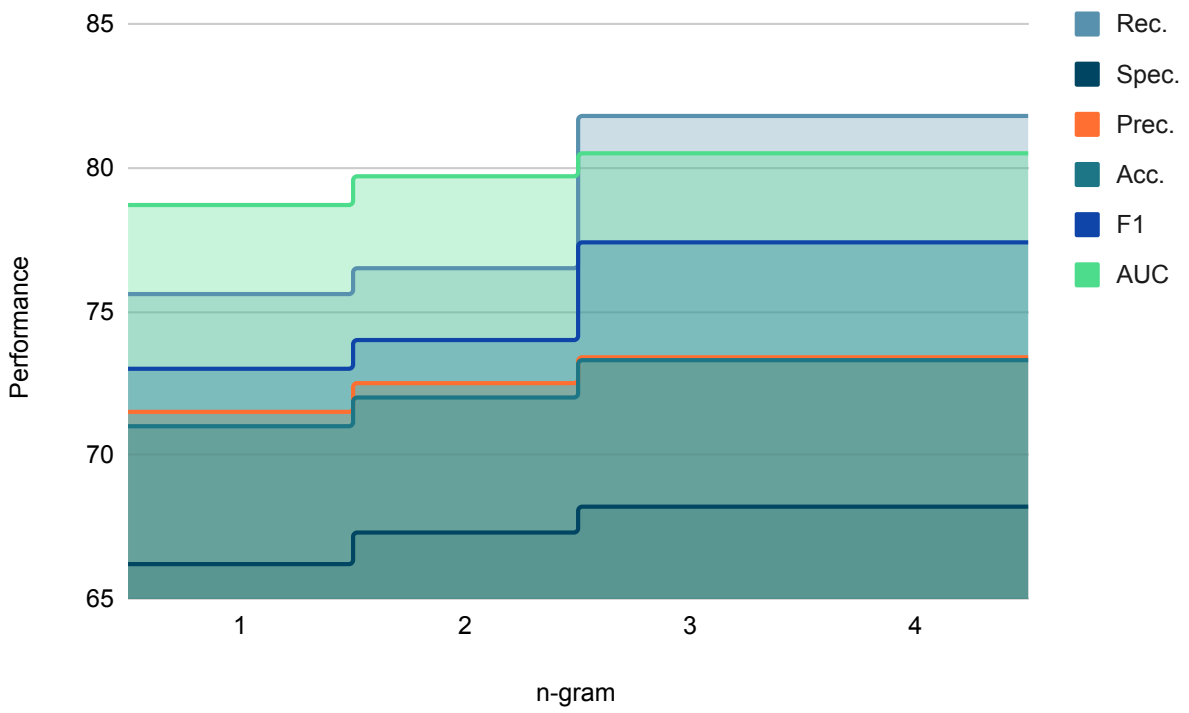


Figure 3.5: Effect of the n-gram parameter (n) on the performance of the model using shallow linguistic kernel (SL).

overfitting [97, 102].

Table 3.11: 10-fold CV performance of APG kernel on the dataset CPI-DS.

c	Recall	Specificity	Precision	Accuracy	F_1	AUC
0.25	79.0	73.3	76.7	76.4	77.7	83.8
0.5	77.6	73.8	76.8	76.0	77.0	83.8
1	77.6	73.8	77.0	76.1	77.1	83.6
2	76.7	74.5	77.2	75.8	76.8	83.1

Evaluating the APG kernel on datasets CPI-DS_IV and CPI-DS_NIV showed that experiments within the same dataset achieve similar performances, independent of the hyperplane optimization parameter c. For both datasets, the AUC values do not differ by more than 2%, indicating high robustness of the classifier. Furthermore, the F_1-score and AUC values on dataset CPI-DS_IV are about 2-5% better than on dataset CPI-DS_NIV, due to clearly higher recall and precision values (Table 3.12 and Table 3.13). Therefore, the APG

kernel performs better in distinguishing between functional and non-functional relations on dataset CPI-DS_IV, i.e. APG is doing well with the presence of the interaction verb.

Table 3.12: 10-fold CV performance of APG kernel on the dataset CPI-DS_IV.

c	Recall	Specificity	Precision	Accuracy	F_1	AUC
0.25	83.8	68.2	77.2	77.4	80.2	84.0
0.5	79.2	73.9	79.6	77.3	79.3	83.9
1	78.4	73.5	79.5	77.0	78.7	83.6
2	76.5	74.0	79.2	75.8	77.6	83.0

Table 3.13: 10-fold CV performance of APG kernel on the dataset CPI-DS_NIV.

c	Recall	Specificity	Precision	Accuracy	F_1	AUC
0.25	79.6	68.1	70.6	73.9	74.6	82.0
0.5	76.3	71.7	72.1	74.1	73.9	82.0
1	75.3	72.5	72.4	74.0	73.6	82.0
2	73.5	74.2	73.3	73.9	73.0	81.9

Table 3.14 shows the holdout cross-validation performance of the predictive model of APG kernel on the unseen test datasets (described in sections 3.1.1 and 3.1.2). The cross-validation on CPI-DS dataset achieved performance with F_1-score of 79.8, precision of 77.1, and AUC of 84.4. Similarly to SL kernel, the evaluation process of APG kernel shows superiority of the prediction model on the dataset that includes candidate compound-protein pairs surrounded by interaction verb (CPI-DS_IV) over the dataset which does not have this structure (CPI-DS_NIV).

Table 3.14: Holdout CV performance of APG kernel on the benchmark dataset.

	Recall	Specificity	Precision	Accuracy	F_1	AUC
CPI-DS	82.8	70.5	77.1	77.2	79.8	84.4
CPI-DS_IV	87.3	66.1	77.7	78.3	82.2	85.2
CPI-DS_NIV	87.0	65.5	72.6	76.5	79.1	82.8

3.3.3 BioBERT

In BioBERT, we have evaluated a set of internal model parameters which perform well against the performance of the model, which is called hyperparameter optimization. The set combination of four well-performing hyperparameters which can make a concrete influence on the performance of the model were selected to evaluate the performance of the BioBERT method on the benchmark dataset (CPI-DS). The selected hyperparameters are:

1. **max_seq_length**: maximum number of tokens of the input sequence after WordPiece tokenization. Input sequences which are longer than this length will be truncated, whereas the shorter ones will be padded, i.e. add a special padding token "[PAD]" to ensure shorter sequences will have the maximum length accepted by the model (max_seq_length). The most popular sequence lengths are 2^n tokens.

2. **train_batch_size**: number of training samples which must be processed in the training before the internal parameters of the model are updated. Because one batch is too big to feed to the memory at once, we divided it into several smaller batches. The possible value of train_batch_size is greater or equal to 1 and less than or equal to the number of input instances in the training dataset. In general, smaller batch sizes train more slowly but can converge more quickly, whereas bigger batch sizes progress in training more quickly but don't always converge quickly [117]. The most popular batch sizes include 32, 64, and 128 samples.

3. **learning_rate**: the learning rate (or step size) is a hyperparameter that controls how much to update the model weights in response to the estimated error during the training process. The learning rate is usually set to a positive value < 1.0. Choosing the learning rate is challenging; with a large learning rate, the system learns fast with large weight updates, which might cause undesirable divergent behavior in the loss function. However, when the learning rate is very small, the training progresses slowly with a very small update to the weights of the network. The optimal learning rate is lying in between, typical learning rate is ranging [1e-1, 1e-5].

4. **num_train_epochs**: number of times which the learning algorithm will run through the whole training dataset during the training process. It allows the learning algorithm to run until the loss is sufficiently minimized. As the number of epochs increases, the more times the weights are updated in the neural network and the higher the running time. When a neural network model is trained using more epochs than necessary, the training model learns patterns that are specific to the sample data. This prevents the model performing well on a new dataset (overfitting) [118]. In general, as the training and validation loss continue to decrease, the training should continue. The number of epochs can be set to a positive integer value greater than one.

All parameter combinations values of the BioBERT were evaluated: 64 and 128 for *max_sequence_length*; 8, 16, and 32 for *train_batch_size*; 2e-5, 3e-5, and 5e-5 for *learning_rate*; and 5 and 10 for *num_train_epochs*.

The model has the best performance with the combination of the hyperparameters values as follows: *max_seq_length* = 128, *train_batch_size* = 16, *learning_rate* = 2e-5 and *num_train_epochs* = 10 with F_1-score of 83.8%, AUC of 89.6%, precision of 81.6%, and recall of 86.3% (Table 3.15).

Table 3.15: 10-fold CV performance of BioBERT on the dataset CPI-DS.

max_ seq_ length	train_ batch_ size	learning_ rate	num_ train_ epochs	Rec.	Spec.	Prec.	Acc.	F_1	AUC
64	8	2e-5	5	85.1	76.0	80.0	81.1	82.3	87.7
64	8	2e-5	10	83.5	77.4	80.4	80.8	81.8	87.5
64	8	3e-5	5	83.9	75.3	79.0	80.1	81.3	87.7
64	8	3e-5	10	83.1	79.5	81.7	81.6	82.3	87.5
64	8	5e-5	5	81.8	73.0	77.0	77.6	79.1	84.3
64	8	5e-5	10	81.3	78.1	80.4	79.9	80.7	86.1

64	16	2e-5	5	86.9	72.7	78.0	80.4	82.1	87.8
64	16	2e-5	10	84.1	78.3	81.2	81.5	82.5	88.2
64	16	3e-5	5	86.3	74.7	79.2	81.1	82.5	88.0
64	16	3e-5	10	83.3	77.7	80.3	80.6	81.6	87.5
64	16	5e-5	5	83.2	72.2	76.9	78.1	79.7	85.4
64	16	5e-5	10	81.9	76.4	79.3	79.4	80.4	87.1
64	32	2e-5	5	85.6	74.3	78.6	80.4	81.8	87.9
64	32	2e-5	10	83.8	77.4	80.4	80.9	82.0	87.5
64	32	3e-5	5	85.0	74.9	78.9	80.4	81.8	88.3
64	32	3e-5	10	82.1	79.2	81.3	80.9	81.6	87.5
64	32	5e-5	5	84.6	73.7	78.0	79.5	81.0	86.6
64	32	5e-5	10	82.6	77.5	80.1	80.2	81.1	86.5
128	**8**	**2e-5**	**5**	**87.2**	**77.3**	**81.0**	**82.8**	**83.9**	**89.4**
128	8	2e-5	10	83.7	77.8	80.9	81.3	82.2	88.6
128	8	3e-5	5	84.9	76.1	79.7	81.0	82.1	88.3
128	8	3e-5	10	84.1	78.5	81.3	81.8	82.5	88.4
128	8	5e-5	5	84.1	73.9	78.1	79.3	80.7	87.6
128	8	5e-5	10	82.2	79.0	81.2	80.7	81.5	88.5
128	16	2e-5	5	87.5	75.4	79.6	81.8	83.3	89.2
128	**16**	**2e-5**	**10**	**86.3**	**78.2**	**81.6**	**82.8**	**83.8**	**89.6**
128	16	3e-5	5	87.0	74.3	79.2	81.3	82.8	88.9
128	16	3e-5	10	84.4	79.2	82.2	82.2	83.1	89.0

128	16	5e-5	5	84.2	75.8	79.3	80.5	81.6	87.9
128	16	5e-5	10	82.7	78.5	81.2	80.9	81.8	87.5

Evaluating BioBERT on datasets CPI-DS_IV showed that the model has the best performance with the combination of the values of the following hyperparameters: *max_seq_length* = 128, *train_batch_size* = 16, *learning_rate* = 3e-5 and *num_train_epochs* = 10 (Table 3.16). For dataset CPI-DS_NIV, the model has the best performance when the combination of the values of the hyperparameters was: *max_seq_length* = 128, *train_batch_size* = 8, *learning_rate* = 2e-5 and *num_train_epochs* = 5 (Table 3.17).

BioBERT achieves significantly better performance on CPI-DS_IV than on CPI-DS_NIV with F_1-score higher in $\sim$5%. This shows that BioBERT performs better in distinguishing between functional and non-functional relations with the presence of the interaction verb. Furthermore, the F_1-score and AUC values on dataset CPI-DS_IV are about 2-5% better than on dataset CPI-DS_NIV, due to clearly higher recall and precision values.

Table 3.16: 10-fold CV performance of BioBERT on the dataset CPI-DS_IV.

max_ seq_ length	**train_ batch_ size**	**learning_ rate**	**num_ train_ epochs**	**Rec.**	**Spec.**	**Prec.**	**Acc.**	**F_1**	**AUC**
64	8	2e-5	5	83.3	80.4	84.6	82.3	83.8	88.4
64	8	2e-5	10	87.2	78.3	83.8	83.6	85.4	87.9
64	8	3e-5	5	83.2	78.7	83.7	81.7	83.3	86.8
64	8	3e-5	10	83.0	77.2	82.1	80.6	82.5	87.0
64	8	5e-5	5	81.1	74.8	80.2	78.7	80.6	84.4
64	8	5e-5	10	84.1	76.5	82.3	81.2	83.0	84.9
64	16	2e-5	5	83.1	81.3	85.1	82.6	84.0	88.9

64	16	2e-5	10	84.8	80.1	84.6	83.1	84.6	87.8
64	16	3e-5	5	83.6	77.2	82.4	81.1	82.8	86.6
64	16	3e-5	10	84.1	78.9	83.6	82.1	83.7	87.9
64	16	5e-5	5	77.5	77.8	82.0	78.2	79.2	85.0
64	16	5e-5	10	82.9	75.3	81.0	79.8	81.7	85.3
64	32	2e-5	5	82.6	79.7	84.1	81.7	83.1	88.1
64	32	2e-5	10	84.4	77.7	82.9	81.8	83.6	88.1
64	32	3e-5	5	83.9	77.6	82.7	81.3	83.1	87.2
64	32	3e-5	10	86.4	78.1	83.5	82.9	84.9	88.0
64	32	5e-5	5	81.5	78.9	83.4	81.1	82.4	86.7
64	32	5e-5	10	83.6	78.3	83.1	81.6	83.3	86.7
128	8	2e-5	5	84.3	79.6	84.3	82.6	84.1	88.9
128	8	2e-5	10	86.9	77.6	83.5	83.3	85.0	89.8
128	8	3e-5	5	81.7	77.1	82.4	80.3	81.9	86.4
128	8	3e-5	10	86.4	78.5	83.9	83.4	85.0	89.2
128	8	5e-5	5	81.4	73.8	79.9	78.6	80.4	84.2
128	8	5e-5	10	86.5	77.5	83.0	82.9	84.6	87.4
128	16	2e-5	5	82.0	78.7	83.2	81.1	82.5	88.8
128	**16**	**2e-5**	**10**	**86.4**	**78.9**	**83.8**	**83.4**	**85.0**	**88.4**
128	16	3e-5e-5	5	84.8	75.2	81.4	81.0	83.0	87.4
128	**16**	**3e-5**	**10**	**87.4**	**79.2**	**84.5**	**84.3**	**85.8**	**89.7**
128	16	5e-5	5	82.2	78.1	82.5	81.0	82.1	87.5

128	16	5e-5	10	84.2	77.1	82.7	81.4	83.3	86.7

Table 3.17: 10-fold CV performance of BioBERT on the dataset CPI-DS_NIV.

max_ seq_ length	train_ batch_ size	learning_ rate	num_ train_ epochs	Rec.	Spec.	Prec.	Acc.	F_1	AUC
64	8	2e-5	5	79.3	78.2	77.2	78.5	77.9	85.5
64	8	2e-5	10	83.1	75.9	77.3	79.5	79.8	86.9
64	8	3e-5	5	81.2	77.4	77.7	79.2	79.2	86.9
64	8	3e-5	10	82.3	77.1	77.6	79.7	79.7	86.7
64	8	5e-5	5	74.1	79.9	77.9	77.0	75.6	84.2
64	8	5e-5	10	72.6	76.3	67.3	74.4	69.5	80.3
64	16	2e-5	5	77.4	78.0	77.4	77.9	77.2	86.2
64	16	2e-5	10	81.5	74.1	75.2	77.7	78.0	84.7
64	16	3e-5	5	76.8	78.9	78.5	78.0	77.3	86.8
64	16	3e-5	10	80.7	78.6	78.3	79.5	79.2	86.6
64	16	5e-5	5	75.1	78.8	77.0	77.0	75.8	85.2
64	16	5e-5	10	79.6	78.0	77.5	78.8	78.4	86.1
64	32	2e-5	5	80.0	77.7	77.6	78.8	78.5	86.5
64	32	2e-5	10	82.7	75.7	76.9	79.1	79.5	86.6
64	32	3e-5	5	75.1	79.7	78.1	77.7	76.5	85.9
64	32	3e-5	10	81.1	74.0	75.7	77.9	78.2	86.2
64	32	5e-5	5	74.6	76.5	75.3	75.6	74.7	85.1

64	32	5e-5	10	79.4	76.1	76.3	77.8	77.6	84.4
128	**8**	**2e-5**	**5**	**83.6**	**79.5**	**79.4**	**81.4**	**81.2**	**88.8**
128	8	2e-5	10	84.3	77.6	78.5	80.7	80.9	87.7
128	8	3e-5	5	85.0	76.2	77.4	80.5	80.7	87.7
128	8	3e-5	10	85.5	75.7	76.8	80.5	80.7	87.8
128	8	5e-5	5	76.9	74.6	74.1	75.5	75.1	83.9
128	8	5e-5	10	81.6	77.6	77.9	79.6	79.5	86.0
128	16	2e-5	5	79.0	77.3	77.0	78.0	77.7	87.2
128	**16**	**2e-5**	**10**	**84.7**	**77.5**	**78.3**	**80.9**	**81.2**	**88.2**
128	16	3e-5	5	81.2	75.9	76.4	78.5	78.6	87.0
128	16	3e-5	10	83.3	76.3	77.4	79.7	79.9	87.6
128	16	5e-5	5	79.4	79.2	78.5	79.5	78.8	86.6
128	16	5e-5	10	84.2	76.0	76.9	80.1	80.2	86.6

Table 3.18 shows the holdout cross-validation performance of the predictive model of BioBERT on unseen test datasets (described in sections 3.1.1 and 3.1.2). It used in the evaluation process the following set of hyperparameters values: *max_seq_length* = 128, *train_batch_size* = 16, *learning_rate* = 2e-5 and *num_train_epochs* = 10, which achieved the best performance of the model in the validation process. The cross-validation of BioBERT on CPI-DS dataset achieved performance with F_1-score of 83.8, precision of 81.6, and AUC of 89.6. The cross-evaluation of BioBERT shows superiority of the prediction model on the dataset that includes candidate compound-protein pairs surrounded by interaction verb (CPI-DS_IV) over the dataset which does not have this structure (CPI-DS_NIV).

Table 3.18: Holdout CV performance of BioBERT on the benchmark dataset.

	Recall	Specificity	Precision	Accuracy	F_1	AUC
CPI-DS	86.8	82.0	85.2	84.6	86.0	91.2
CPI-DS_IV	82.2	87.7	90.0	84.4	85.8	89.0
CPI-DS_NIV	87.9	78.2	80.9	83.2	84.3	89.5

3.4 Comparison and Combination of the Predictive Methods

In addition to evaluating the performance of each individual method (shallow linguistic kernel (SL), all-paths graph kernel (APG), and BioBERT), we evaluated combinations of the models to test the confidence and precision. We have analyzed whether the combination of the evaluated methods (SL, APG, BioBERT) yields higher confidence and precision than the individual method. We combined them by applying:

- **majority voting**: the candidate compound-protein pair is classified as functionally related if at least two of the three evaluated methods have predicted this pair as functionally related; otherwise the pair is classified as not functionally related.
- **jury decision**: the candidate compound-protein pair is classified as functionally related if and only if all the evaluated methods predicted this pair as functionally related; otherwise the pair is classified as not functionally related.

Table 3.19 shows that BioBERT performs better than APG and SL; however, APG performs slightly better than SL. The jury decision of the combination of the three evaluated methods has the highest precision, with 88.1%; on the other hand, the recall is decreased, a significant fraction (17%) of the functionally related (positive) pairs is not identified (lost). Similarly, the combinations of BioBERT AND SL, BioBERT AND APG, BioBERT AND (SL OR APG), and the majority vote all have a slightly better precision but low recall, i.e. more functionally related (positive) pairs were not identified (Figure 3.6). In general, combining more than one method can improve the precision, while on the other hand, it reduces the

recall, i.e. losing a significant fraction of the functionally related (positive) pairs. BioBERT has the best overall performance, with an F_1-score of 86.0%, precision of 84.6%, and recall of 86.8%.

Table 3.19: The performance of the ML model of the evaluated methods (SL, APG, BioBERT) and their combinations.

Method	Recall	Specificity	Precision	Accuracy	F_1	AUC
SL kernel	83.3	68.2	75.8	76.4	79.3	83.1
APG kernel	82.8	70.5	77.1	77.2	79.8	84.4
BioBERT	**86.8**	**82.0**	**85.2**	**84.6**	**86.0**	**91.2**
SL AND APG	74.7	79.1	81.0	76.7	77.7	-
BioBERT AND SL	76.4	86.2	86.9	80.9	81.3	-
BioBERT AND APG	76.2	86.5	87.0	80.9	81.3	-
BioBERT AND (SL OR APG)	82.6	84.0	86.0	83.2	84.3	-
Majority vote*	87.2	74.3	80.2	81.6	83.6	-
Jury Decision**	70.0	88.7	88.1	78.5	78.0	-

* Majority vote: The pair is predicted as functional if at least two of the three evaluated methods (BioBERT, SL, APG) predict the pair as functional, otherwise non-functional. **Jury Decision: The pair is predicted as functional if all of the methods (BioBERT, SL, APG) predict the pair as functional, otherwise non-functional.

3.4.1 Runtime of the Evaluated Methods

The validation procedure for the optimal hyperparameters of the evaluated methods done on a machine has the specifications as shown in Table 3.20.

Table 3.20: The specifications of the machine which was used for the evaluation process.

	Specification
CPU	Intel core i5-9600k (6x 3.70GHz)
GPU	Geforce RTX 2070 SUPER, 8 GB GDDR6, 2560 CUDA cores, 1.77 GHz
RAM	55 GB DDR4
OS	Ubuntu 18.04.6 LTS (64 bits)

Table 3.21 shows the runtime of each method and the type of the processing unit used, central processing unit (CPU) or graphics processing unit (GPU). The BioBERT model

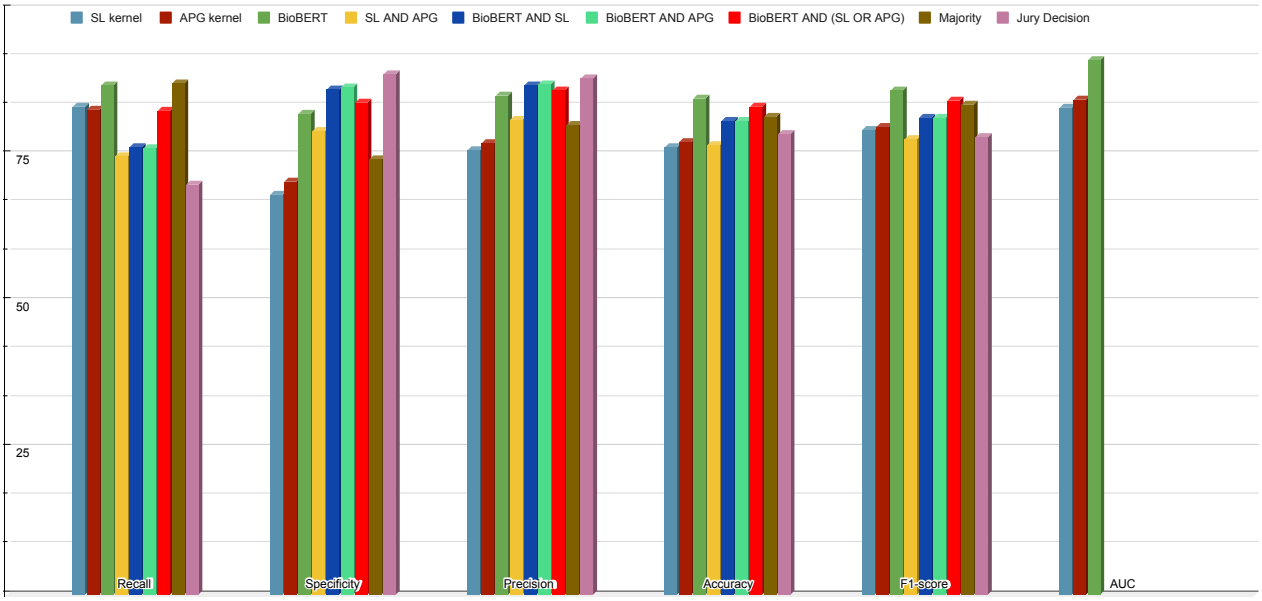


Figure 3.6: The performance comparison of the predictive methods and their combinations.

takes significantly less runtime compared to SL and APG. Using GPU clearly shows the significant advantage of this processing unit, which has many more cores than CPU - it can be up to 1000x. This aspect has to be considered within the scenario of applying a selected model to all PubMed articles, where we can see the significant advantage of using GPU over CPU, and the superiority of BioBERT over other models in terms of runtime (see section Large scale dataset application).

Table 3.21: Runtime of the validation process of SL, APG, and BioBERT on benchmark dataset.

Method	Type of processing unit	Runtime
SL	CPU	9 minutes
APG	CPU	28 minutes
BioBERT	CPU	117 minutes
BioBERT	**GPU**	**3.4 minutes**

3.5 Large Scale Dataset Application

The evaluated methods (SL, APG, and BioBERT) have been successfully applied to the whole MEDLINE database of references and abstracts on life sciences and biomedical top-

ics which were published before July 2022, comprising about 33M references to biomedical and life sciences articles. The dataset consists of more than 140M sentences, with around 6M of them containing at least one compound-protein candidate pair (Table 3.22). The three evaluated methods classified 55-59% of the candidate pairs as functionally related and 41-45% of them classified as non-functionally related (Figure 3.7). 62% of the candidate pairs are classified identically by the three evaluated methods, with around 2.5M unique functionally related compound-protein pairs. The total elapsed time of the BioBERT shows the superiority of the parallel processing using a graphics processing unit (GPU) over the central processing unit (CPU). BioBERT took around 13 hours using GPU to apply the model on the whole MEDLINE dataset, whereas shallow linguistic kernel (SL) and all-paths graph kernel (APG) did the same job in around 15 and 24 days, respectively (Figure 3.8).

Table 3.22: Statistical information of application of the predictive model of SL, APG, and BioBERT on the whole MEDLINE database.

	SL	**APG**	**BioBERT**
PubMed articles	33M		
Number of sentences	140M		
Number of articles with candidate pairs	2.8M		
Number of sentences with candidate pairs	6.0M		
Number of candidate pairs	16.3M		
Functional relations	9.6M = 58.9%	9.5M = 58.3%	9.0M = 55.2%
Non-functional relations	6.7M = 41.1%	6.8M = 41.7%	7.3M = 44.8%
Number of identical predictions of the three evaluated methods	10.1M = 62% (61.0% functional, 39.0% non-functional)		
Number of the distinct functional relation pairs	2.5M		
Total elapsed time	15 days	24 days	13 hours

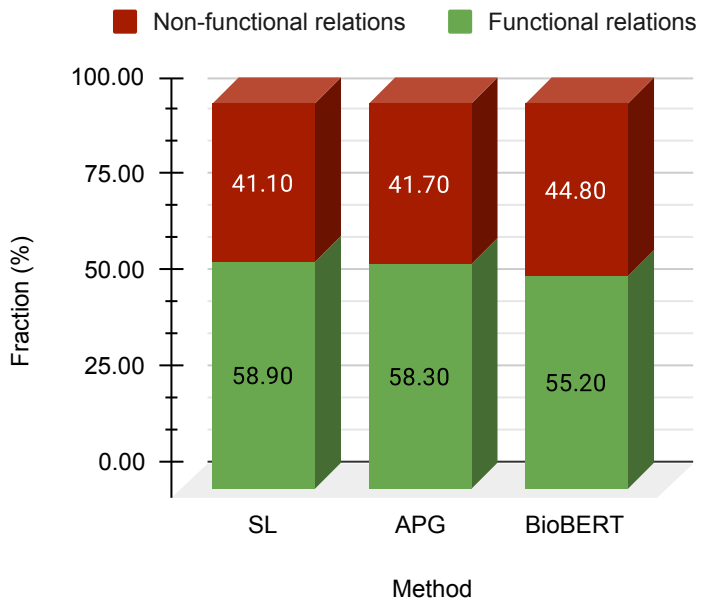


Figure 3.7: Percentage distribution of functional and non-functional compound-protein relationship pairs of the whole MEDLINE database when applying the predictive models (SL, APG, BioBERT).

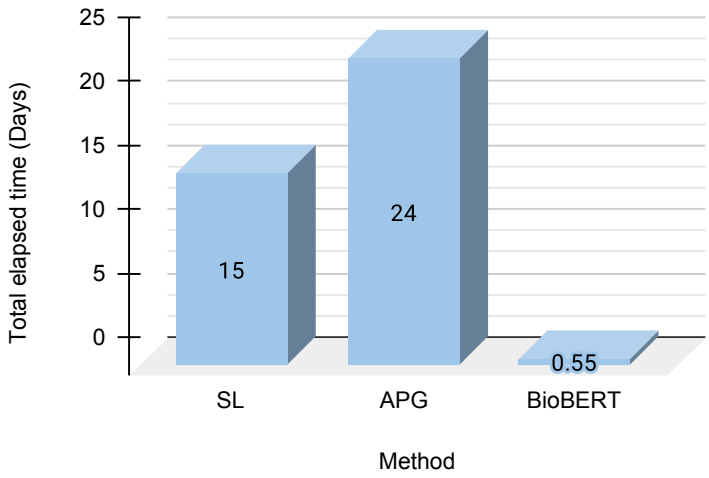


Figure 3.8: Runtime of the predictive models (SL, APG, BioBERT) when applied on the whole MEDLINE database. SL and APG used CPU, whereas BioBERT used GPU.

3.6 Web Server: Compound-Protein Relationships in Literature (CPRiL)

Compound-Protein Relationship in Literature (CPRiL) is a new, user-friendly, freely available web-based service for functional compound-protein relationships in biomedical and life sciences literature. CPRiL is built by applying the CPRiL pipeline on the full MEDLINE database, which comprises more than 33 million references to biomedical and life sciences articles. CPRiL is built using Django as the web framework for developing web resources.

3.6.1 CPRiL Database Schema

CPRiL used PostgreSQL relational database as a backend database. Figure 3.9 shows the schema of CPRiL database. The main tables in CPRiL database are:

- **tbl_articles**: This table includes information about the biomedical articles such as PMID, title, journal, and published year.
- **tbl_sentences**: This table includes the text of sentences which have the functionally related pairs.
- **tbl_compounds**: This table includes information about the interacted compounds such as compound name, MeSH ID, PubChem ID, inChi, SMILES, and molecular formula.
- **tbl_proteins**: This table includes information about the interacted proteins such as protein name, NCBI gene ID, gene symbol, organism ID, and gene summary.
- **tbl_comp_synonyms** and **tbl_prot_synonyms**: These two tables include the most general synonyms of compounds and proteins which are collected from MeSH, gene, and PubChem databases.
- **tbl_organisms**: This table includes the organisms of the interacted proteins.

- **tbl_cpi_prediction**: This table includes information about the functional related compound-protein pairs such as PMID, sentence ID, compound tagged names, protein tagged names, the prediction of the evaluated machine learning methods (SL, APG, BioBERT), and the date of collecting this information.

3.6.2 CPRiL Features

This section describes the main features of CPRiL. These features include searching types, network layout visualization of the output, and the shortest path between two entities.

3.6.2.1 Searching Types

CPRiL as a search engine offers two main types of search: standard search and advanced search.

Standard Search: This type of search includes:

- Searching by name or synonym of a compound for functional relations to proteins. Figure 3.10 shows an example of the result of this type of search, where *"Remdesivir"* is the searched compound name.
- Searching by name, synonym, or UniProt entry name of a protein for functional relations to small molecules. Figure 3.11 and Figure 3.12 show the output result of this type of searching, where *"P53; Homo sapiens"*, and *"SPIKE_SARS2"* are the searched protein name and UniProt entry name, respectively.
- Searching for all functional compound-protein relationships in a specific article by a unique identifier number (PubMed ID) of this article. Figure 3.13 shows all functional compound-protein relationships which appear in the article with *PMID = 32521159*.

Figure 3.14 shows an example of the output result of the functional relationship between *"Remdesivir"* and *"ORF1a polyprotein"* which are described in the article *"RNA-dependent RNA polymerase: Structure, mechanism, and drug discovery for COVID-19"* [119].

Advanced Search: This type of search offers to query for the functional compound-protein relationship within a specific time period using all the combinations of compound

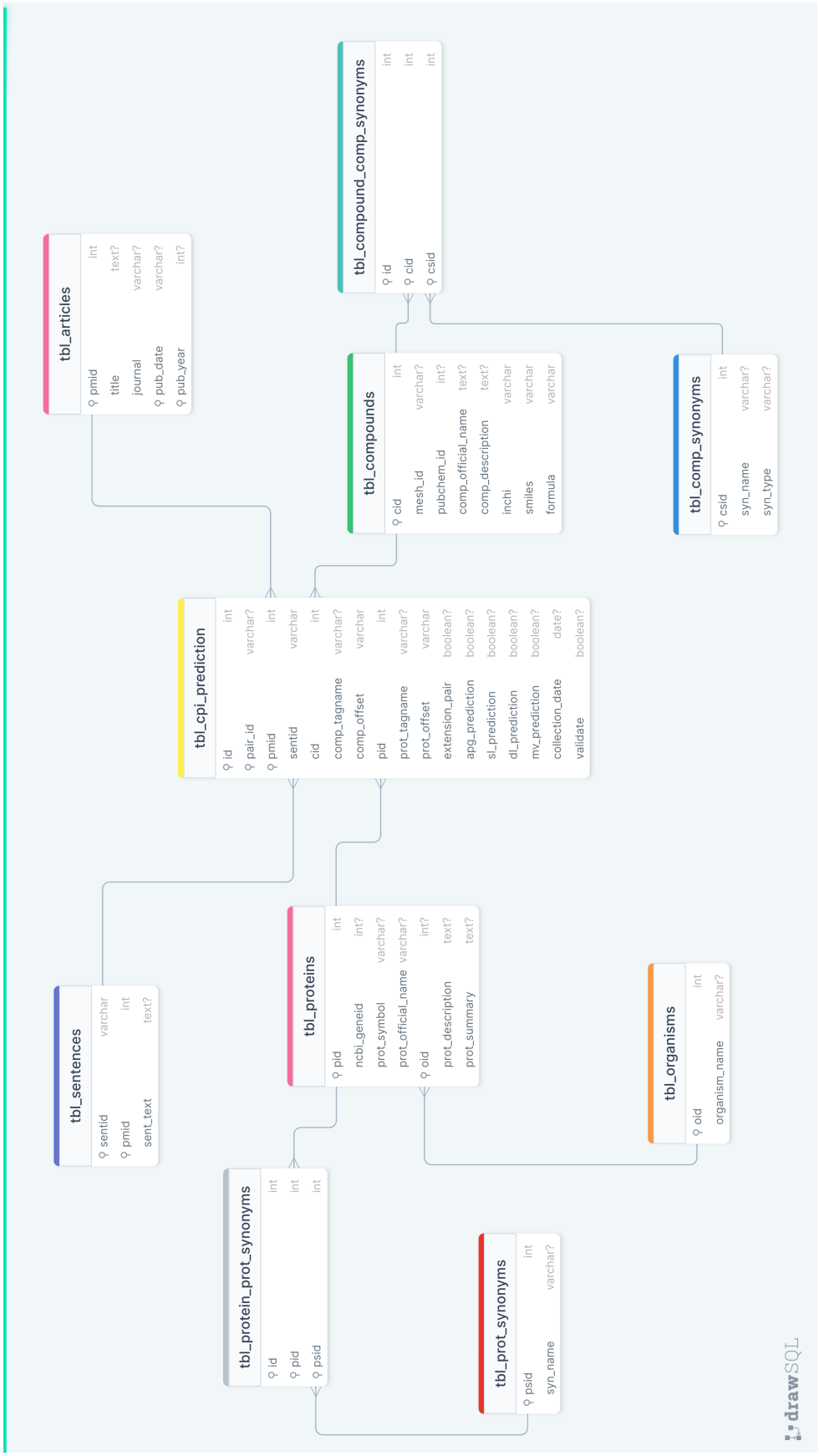


Figure 3.9: The schema of CPRiL database.

remdesivir

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z 1 2 3 4 5 6 7 8 9 All

1 2 ▶ 3

106 Functionally related protein(s)

Download

	Protein Name	UniProt Entry	Gene Symbol	Organism	NCBI Gene ID	# All related compound(s)	#Article(s)
1	ORF1a polyprotein;ORF1ab polyprotein	R1AB_SARS2	ORF1ab	Severe acute respiratory syndrome coronavirus 2	43740578	489	94
2	NEWENTRY	-	NEWENTRY	Severe acute respiratory syndrome-related coronavirus	8673700	634	12
3	angiotensin converting enzyme 2	ACE2_HUMAN	ACE2	Homo sapiens	59272	899	11
4	surface glycoprotein	SPIKE_SARS2	SPIKE_SARS2	Severe acute respiratory syndrome coronavirus 2	43740568	642	9
5	interleukin 6	IL6_HUMAN	IL6	Homo sapiens	3569	5445	6
6	carboxylesterase 1	EST1_HUMAN	CES1	Homo sapiens	1066	615	5
7	cytochrome P450 family 3 subfamily A member 4	CP3A4_HUMAN	CYP3A4	Homo sapiens	1576	3476	5
8	C-reactive protein	CRP_HUMAN	CRP	Homo sapiens	1401	2286	4
9	solute carrier family 29 member 1 (Augustine blood group)	S29A1_HUMAN	SLC29A1	Homo sapiens	2030	349	4

Figure 3.10: An example of searching for functionally related proteins to specific compound using compound name. The output result of searching by compound name *"Remdesivir"* for functionally related proteins.

P53 ; *Homo sapiens*

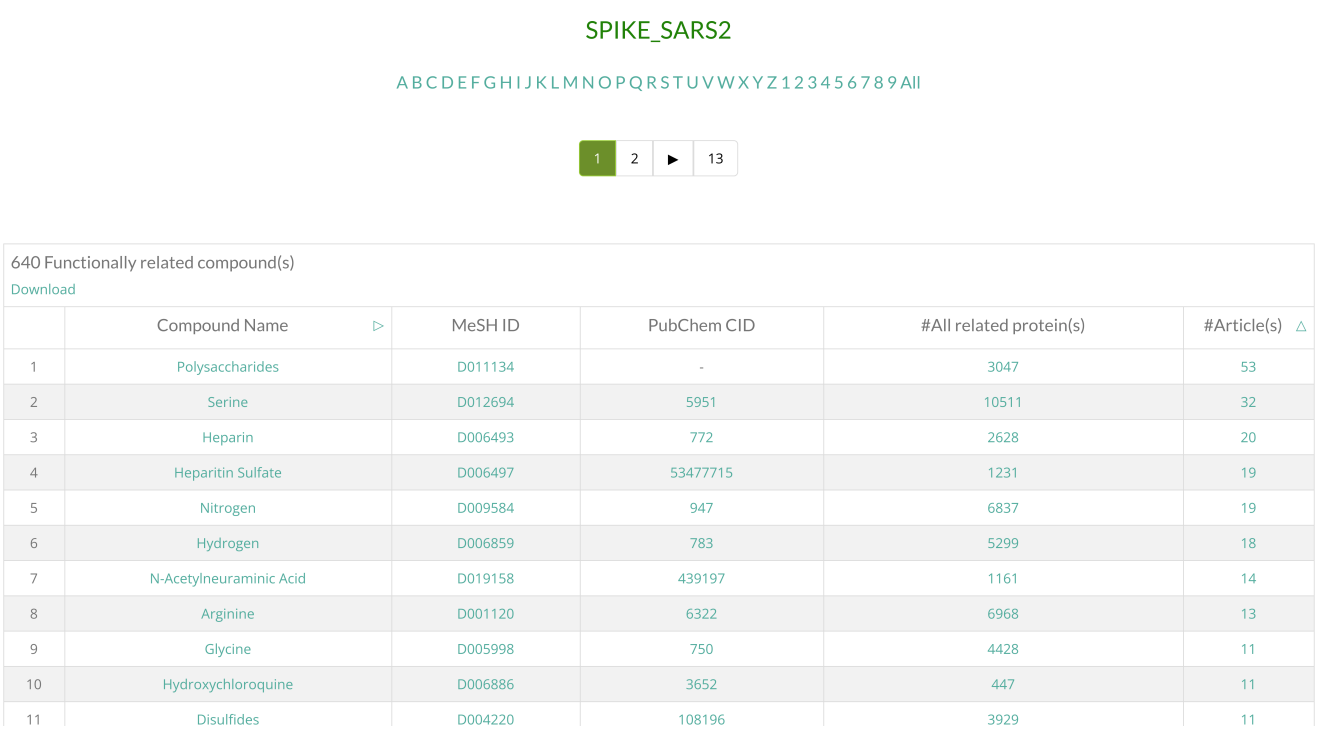
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z 1 2 3 4 5 6 7 8 9 All

1 2 ▶ 112

5585 Functionally related compound(s)
Download

	Compound Name ▷	MeSH ID	PubChem CID	#All related protein(s)	#Article(s) △
1	Cisplatin	D002945	-	5829	886
2	Serine	D012694	5951	10511	798
3	Doxorubicin	D004317	31703	4938	715
4	Arginine	D001120	6322	6968	561
5	Reactive Oxygen Species	D017382	-	8447	511
6	Proline	D011392	-	4879	383
7	Paraffin	D010232	-	2141	365
8	Fluorouracil	D005472	3385	2783	354
9	Etoposide	D005047	36462	1669	267
10	Lysine	D008239	5962	7811	249
11	pifithrin	C121565	9929138	206	240
12	Paclitaxel	D017239	36314	3312	240
13	Hydrogen Peroxide	D006861	784	6836	195

Figure 3.11: An example of searching for functionally related compounds to a specific protein using protein name and organism name. The output result of searching by protein name *"P53"* and organism *"Homo sapien"* for functionally related small molecules.



	Compound Name	MeSH ID	PubChem CID	#All related protein(s)	#Article(s)
1	Polysaccharides	D011134	-	3047	53
2	Serine	D012694	5951	10511	32
3	Heparin	D006493	772	2628	20
4	Heparitin Sulfate	D006497	53477715	1231	19
5	Nitrogen	D009584	947	6837	19
6	Hydrogen	D006859	783	5299	18
7	N-Acetylneuraminic Acid	D019158	439197	1161	14
8	Arginine	D001120	6322	6968	13
9	Glycine	D005998	750	4428	11
10	Hydroxychloroquine	D006886	3652	447	11
11	Disulfides	D004220	108196	3929	11

Figure 3.12: An example of searching for functionally related compounds to a specific protein using UniProt entry name. The output result of the functionally related small molecules to protein with UniProt entry name *"SPIKE_SARS2"*.

synonym, protein synonym, Uniprot entry name, and publishing time of the article. Figure 3.15 shows an example of searching for the functional relationship between small molecule *"Apixaban"* and protein *"coagulation factor X; Homo sapiens"* which are mentioned in the biomedical articles published in the period between 2010 and 2022.

The search process is supported by features such as autocomplete and suggestions in case the searched entry does not have an exact match or if there is a typo. Additionally, all the output results of all types of searches can be sorted and provided for download as a csv file (comma-separated values).

3.6.2.2 Network Visualization of the Output

The network view is an efficient way to summarize the output results of the search because it offers a visual representation of the output. The network layout gives a summary overview of the relationship between the searched compound/protein and another entity (compound/protein). In the network layout, each entity is represented by a vertex; the

Title : Structural Basis of the Potential Binding Mechanism of Remdesivir to SARS-CoV-2 RNA-Dependent RNA Polymerase.

Pub. Date : 2020 Aug 13

PMID : 32521159

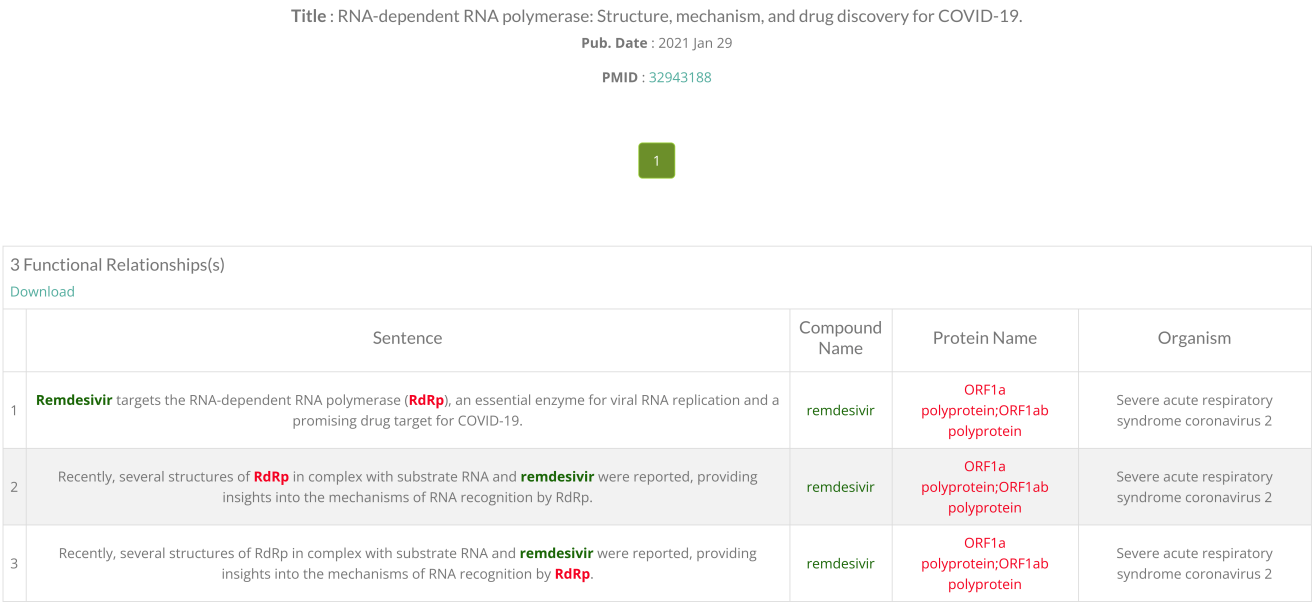
1

11 Functional Relationships(s)

Download

	Sentence	Compound Name	Protein Name	Organism
1	Here, we used molecular dynamics simulations and free energy perturbation methods to study the inhibition mechanism of **remdesivir** to its target SARS-CoV-2 virus RNA-dependent RNA polymerase (**RdRp**).	remdesivir	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
2	We then built a putative preinsertion binding structure by aligning the **remdesivir** + **RdRp** complex to the ATP bound poliovirus RdRp without the RNA template.	remdesivir	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
3	We then built a putative preinsertion binding structure by aligning the **remdesivir** + RdRp complex to the ATP bound poliovirus **RdRp** without the RNA template.	remdesivir	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
4	We then built a putative preinsertion binding structure by aligning the remdesivir + RdRp complex to the **ATP** bound poliovirus **RdRp** without the RNA template.	Adenosine Triphosphate	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
5	The resulting stable preinsertion state of **remdesivir** appeared to form hydrogen bonds with the RNA template when aligned with the newly solved cryo-EM structure of SARS-CoV-2 **RdRp**.	remdesivir	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
6	The relative binding free energy between **remdesivir** and ATP was calculated to be -2.80 +- 0.84 kcal/mol, where remdesivir bound much stronger to SARS-CoV-2 **RdRp** than the natural substrate ATP.	remdesivir	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
7	The relative binding free energy between remdesivir and **ATP** was calculated to be -2.80 +- 0.84 kcal/mol, where remdesivir bound much stronger to SARS-CoV-2 **RdRp** than the natural substrate ATP.	Adenosine Triphosphate	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
8	The relative binding free energy between remdesivir and ATP was calculated to be -2.80 +- 0.84 kcal/mol, where **remdesivir** bound much stronger to SARS-CoV-2 **RdRp** than the natural substrate ATP.	remdesivir	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
9	The ~100-fold improvement in the Kd from **remdesivir** over ATP indicates an effective replacement of ATP in blocking of the **RdRp** preinsertion site.	remdesivir	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
10	The ~100-fold improvement in the Kd from remdesivir over **ATP** indicates an effective replacement of ATP in blocking of the **RdRp** preinsertion site.	Adenosine Triphosphate	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
11	The ~100-fold improvement in the Kd from remdesivir over ATP indicates an effective replacement of **ATP** in blocking of the **RdRp** preinsertion site.	Adenosine Triphosphate	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2

Figure 3.13: An example of the functional compound-protein relations in CPRiL by searching using PMID. All functional compound-protein relations which appear in the article with PMID *"32521159"*.



Title : RNA-dependent RNA polymerase: Structure, mechanism, and drug discovery for COVID-19.

Pub. Date : 2021 Jan 29

PMID : 32943188

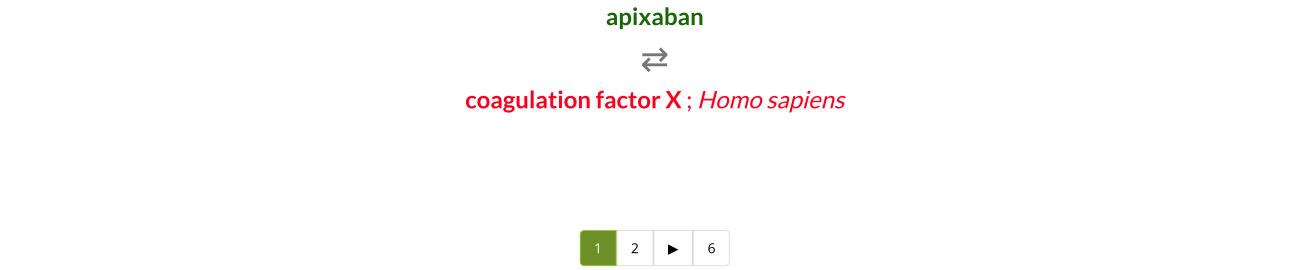
1

3 Functional Relationships(s)

Download

	Sentence	Compound Name	Protein Name	Organism
1	**Remdesivir** targets the RNA-dependent RNA polymerase (**RdRp**), an essential enzyme for viral RNA replication and a promising drug target for COVID-19.	remdesivir	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
2	Recently, several structures of **RdRp** in complex with substrate RNA and **remdesivir** were reported, providing insights into the mechanisms of RNA recognition by RdRp.	remdesivir	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2
3	Recently, several structures of RdRp in complex with substrate RNA and **remdesivir** were reported, providing insights into the mechanisms of RNA recognition by **RdRp**.	remdesivir	ORF1a polyprotein;ORF1ab polyprotein	Severe acute respiratory syndrome coronavirus 2

Figure 3.14: An example of the functional relationship between a specific compound and protein appears in an article. The output shows the sentences which appear the relationship between *Remdesivir* and *ORF1a polyprotein* in the article with PMID *"32943188"*.



apixaban

⇄

coagulation factor X ; *Homo sapiens*

1 2 ▶ 6

291 Article(s)

Download

	PMID	Title	Pub. Year	#Total Relationships
1	34521333	An Update of the Efficacy and Comparative Characteristics of Direct (New) Oral Anticoagulants (DOACs).	2022	1
2	34864841	Utilization of apixaban anti-Xa levels in transition from apixaban to warfarin in a patient with chronic renal dysfunction.	2022 Apr 19	1
3	35135308	MRI-Detected Brain Lesions and Cognitive Function in Patients With Atrial Fibrillation Undergoing Left Atrial Catheter Ablation in the Randomized AXAFA-AFNET 5 Trial.	2022 Mar 22	1
4	35455642	Impact of the Genotype and Phenotype of CYP3A and P-gp on the Apixaban and Rivaroxaban Exposure in a Real-World Setting.	2022 Mar 24	1

Figure 3.15: An example of the advanced search of CPRiL. The functional relationship between small molecule *"Apixaban"* and protein *"coagulation factor X; Homo sapiens"* in the biomedical articles published between 2010 and 2022.

entity types are discriminated with different colors. The edges represent the relationship between the entities; stronger relationships are represented by thicker edges with a weight representing the number of biomedical articles where this relationship appears. Furthermore, the shortest paths between compound and protein are calculated even if there is no direct connection between them; this can give an overview when a compound-protein pair is related via other compounds and proteins.

In CPRiL, the functional compound-protein relationships can be displayed as a network. This network displays the top n functional compound-protein relationships of the searched entity (compound/protein) which have the highest number of occurrences in the biomedical and life sciences articles. The node in the middle represents the searched compound/protein; the other nodes represent the proteins which are functionally related to the searched compound, or compounds in case the searched entity is a protein. Moreover, the layout shows the number of occurrences of the compound-protein relationship as a number on the edge. All the nodes and edges in the network are clickable: the nodes link to the web card of the compound or protein, the edges link to the articles where this relationship appears. Figure 3.16 shows an example of the network layout of a compound-protein relationship; it shows the top 10 proteins which have functional relationships to the compound *“Remdesivir”*; furthermore, it shows that *“ORF1a polyprotein; Severe acute respiratory syndrome coronavirus 2”* the most frequently related protein to *“remdesivir”* with 94 articles. Similarly, Figure 3.17 shows the top 10 it shows the top 10 compounds which have functional relationships to the protein *“SPIKE_SARS2”*, *“polysaccharides”* is the most frequently related compound to *“SPIKE_SARS2”* with 53 articles.

3.6.2.3 Shortest Path between Entities

In case there is no direct relation between a compound and the protein or if you are looking for indirect relation between two proteins, then the shortest path feature of CPRiL will be very useful and gives an idea of how the two entities can be related to each other. Figure 3.18 shows the indirect relationship between the compound *“otamixaban”* and the protein *“SPIKE_SARS2”* using the shortest path. It shows the top N shortest paths between the compound *“otamixaban”* and the protein *“SPIKE_SARS2”*; the weight of the edges repre-

Figure 3.16: Network Visualization of the functional compound-protein relation for compound searching. Top 10 proteins which have functional relationships to the compound *"Remdesivir"*.

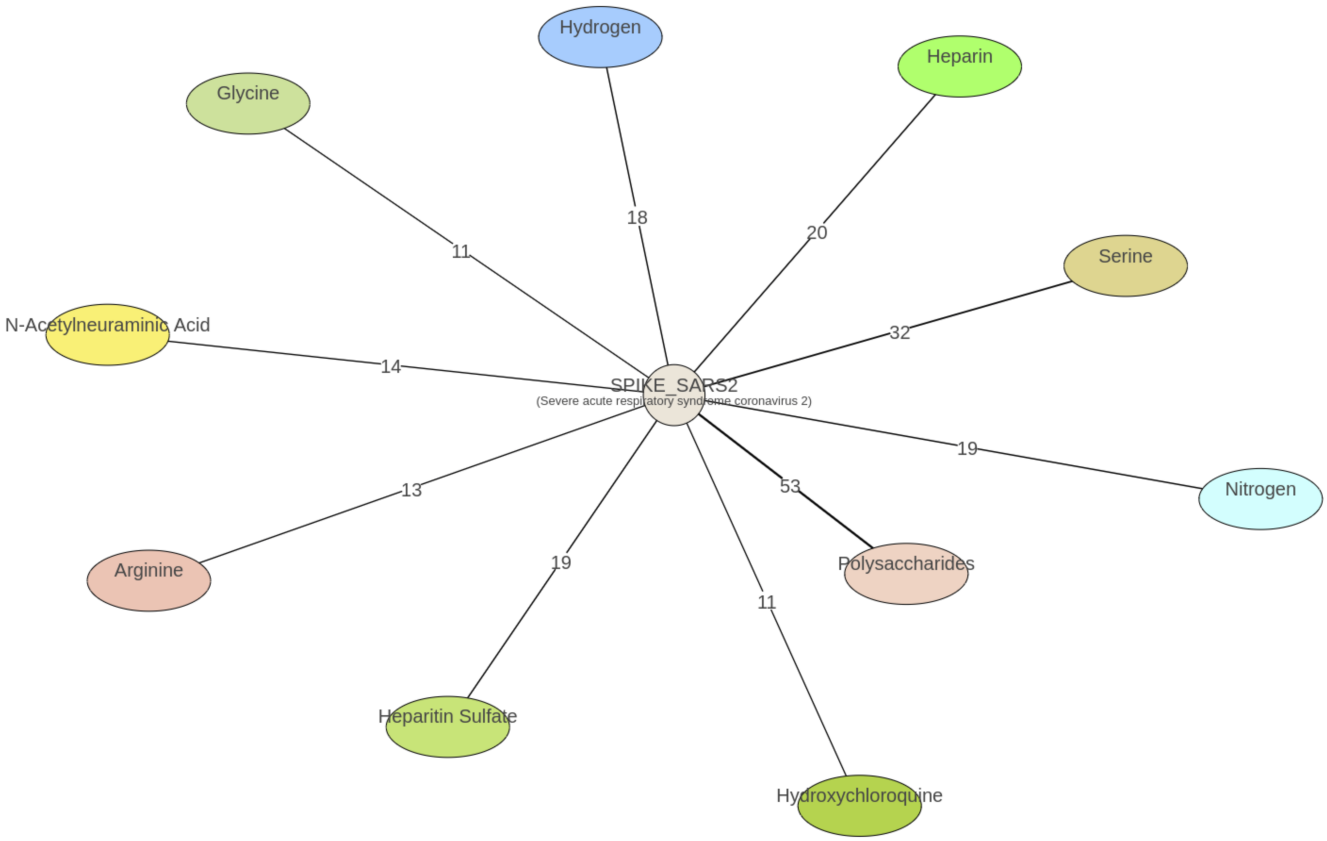


Figure 3.17: Network Visualization of the functional compound-protein relation for protein searching. Top 10 compounds which have functional relationships to the protein *"SPIKE_SARS2"*.

sents the number of biomedical articles where this relationship appears. In this example, the shortest path is *otamixaban —> coagulation factor II, thrombin (Home Sapiens) —> Heparin —> SPIKE_SARS2*.

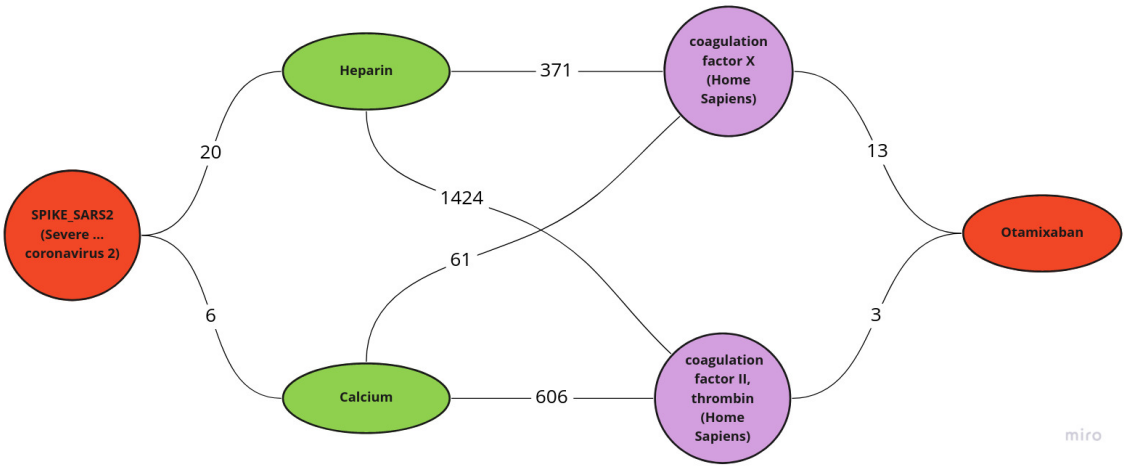


Figure 3.18: Shortest path between compound and protein. The top n shortest paths between the compound *"otamixaban"* and the protein *"SPIKE_SARS2"*. The source and destination are shown in red, the compounds in the shortest path are shown in green, and proteins are shown in purple.

Figure 3.19 shows the indirect relationship between the two proteins *"SPIKE_SARS2"* and *"TMPS2_HUMAN"* using the shortest path. It shows that *SPIKE_SARS2* can be indirectly related to *TMPS2_HUMAN* by the interaction with Bromhexine, Hydroxychloroquine, or Nafamostat. All the nodes and edges in the network are clickable: the nodes link to the card of the compound or protein, and the edges link to the articles where this relationship appears.

3.6.3 Statistical Data of CPRiL

Table 3.23 shows some statistical data of CPRiL. Figure 3.20 displays the distribution of the annual number of articles over the last 15 years. Figure 3.21 shows the changes in the annual number of functional compound-protein relationships over the last 15 years.

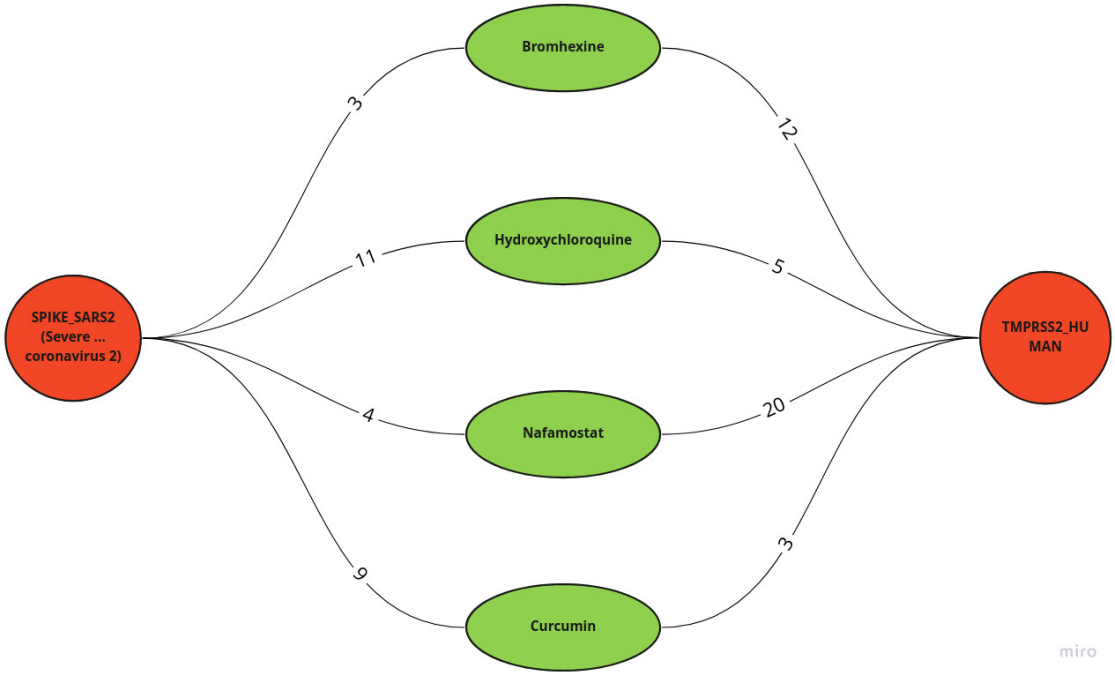


Figure 3.19: Shortest path between two proteins. The top n shortest paths between two proteins *"SPIKE_SARS2"* and *"TMPS2_HUMAN"*. The source and destination are shown in red, the compounds in the shortest path are shown in green, and proteins are shown in purple.

Table 3.23: Statistical data of CPRiL.

Attribute	Count
Number of PubMed articles	∼33 M
Number of articles which have at least one functionally related compound-protein pair	∼2.1 M
Number of unique sentences which have at least one functionally related compound-protein pair	∼4.3 M
Number of functionally related compound-protein pairs	∼8.9 M
Number of unique functionally related compound-protein pairs	∼2.5 M
Number of unique names and synonyms of chemical compounds	∼459 K
Number of unique Molecules with Mesh IDs	∼42.7 K
Number of unique Molecules with PubChem IDs	∼50.7 K
Number of unique proteins	∼90.7 K
Number of unique organisms	1129

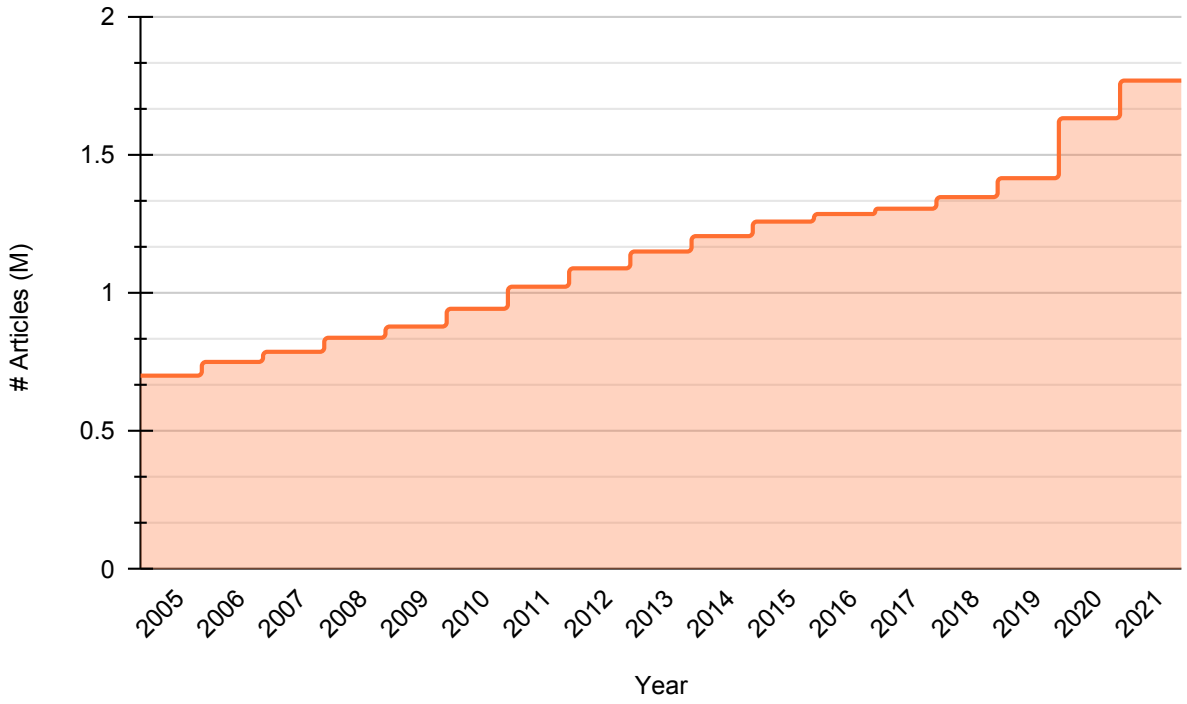


Figure 3.20: The distribution of biomedical articles over the last 15 years.

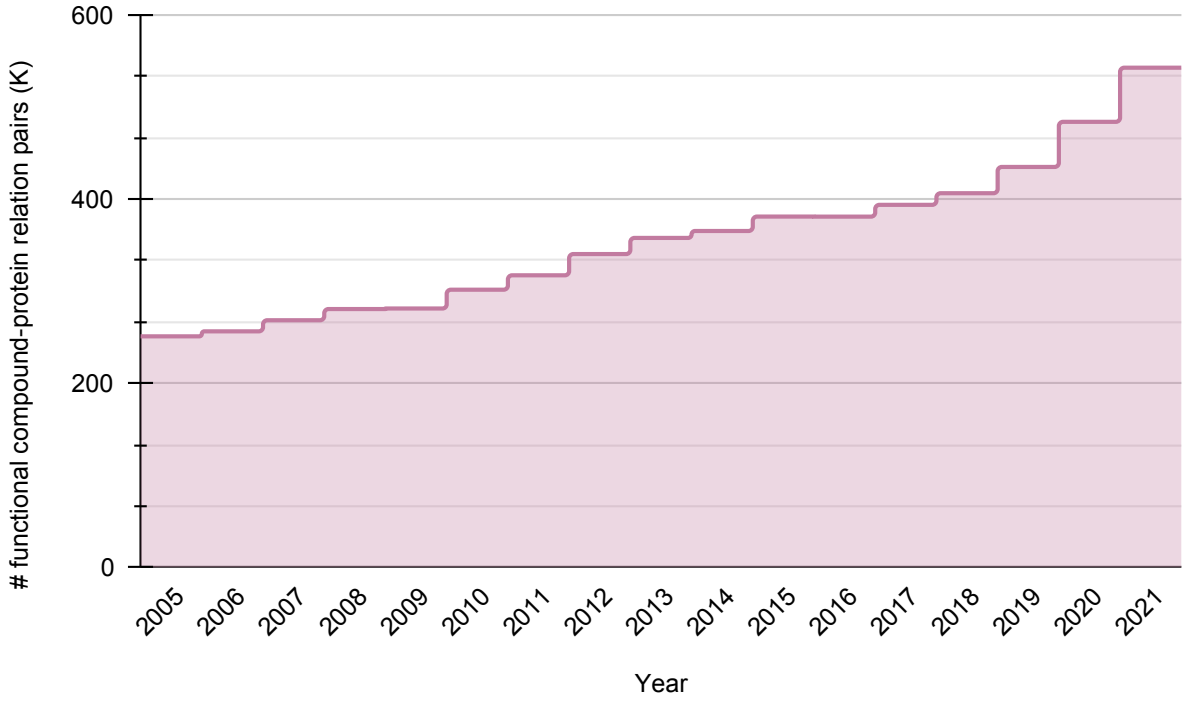


Figure 3.21: The annual number of functionally related compound-protein pairs over the last 15 years.

Table 3.24 shows the top 10 functional compound-protein related pairs which have the highest number of articles where these relationships are described.

Table 3.24: Top ten functionally related compound-protein pairs.

	Compound name	PubChem ID	Protein name	Organism	UniProt ID	#Articles
1	Glucose	5793	insulin	Homo sapiens	P01308 (INS_HUMAN)	21,875
2	Iron	23925	transferrin	Homo sapiens	P02787 (TRFE_HUMAN)	3,127
3	Gefitinib	123631	epidermal growth factor receptor	Homo sapiens	P00533 (EGFR_HUMAN)	2,974
4	Lipids	-	insulin	Homo sapiens	P01308 (INS_HUMAN)	2,606
5	Sirolimus	5284616	mechanistic target of rapamycin kinase	Homo sapiens	P42345 (MTOR_HUMAN)	2,470
6	Erlotinib	176871	Hydrochloride epidermal growth factor receptor	Homo sapiens	P00533 (EGFR_HUMAN)	2,428
7	Iron	23925	hepcidin antimicrobial peptide	Homo sapiens	P81172 (HEPC_HUMAN)	1,956
8	Imatinib	123596	Mesylate ABL proto-oncogene 1, non-receptor tyrosine kinase	Homo sapiens	P00519 (ABL1_HUMAN)	1,908
9	2-(4-morpholinyl)-8-phenyl-4H-1-benzopyran-4-one	3973	AKT serine/threonine kinase 1	Homo sapiens	P31749 (AKT1_HUMAN)	1,802
10	Calcium	5460341	parathyroid hormone	Homo sapiens	P01270 (PTHY_HUMAN)	1,797

CHAPTER

4 DISCUSSION

This work aimed at extracting functional relationships between molecules from biomedical literature using artificial intelligence-based text mining and machine learning techniques. A new benchmark dataset (CPI-DS) containing annotations of proteins, compounds, and their functional relationships was manually curated for evaluation purposes. This dataset serves as a great resource in the field of relation extraction and can be used for benchmarking predictive models in this field. A related benchmark applied for the BioCreAtIvE challenge includes a dataset for chemical-protein interactions from PubMed abstracts which were annotated manually by domain experts. While the ChemProt-benchmark for the training of the classification of functional compound-protein interactions into different groups (e.g. upregulator, antagonist, etc.), focuses on validated interactions, it is therefore not suitable for the separation from functionally unrelated compound-protein pairs that are mentioned in texts. It is planned to share the CPI-DS dataset with BioCreAtIvE organizers to also address this text mining task. This will expand the size of the BioCreative dataset and will make it more comprehensive.

In deep learning, the model normally tends to continue learning as it is given more

data, thus improving the quality of the model [31]. Though the effects of increasing the benchmark dataset were not studied in this dissertation, I expect that increasing the dataset size would improve the performance of the predictive model used here (BioBERT). Increasing the dataset manually requires a lot of work and is thus time-consuming. However, semi-manual annotation might accelerate the process of increasing the size of the training dataset. Such a process might consist of two steps: in the first, artificial intelligence-based text mining models such as BioBERT can be used to annotate the entities and to identify the relationships between them; in the second step, all entities and relationships can be manually crosschecked by domain experts. The first step will help the curators to reduce the number of sentences which need to be checked, instead of spending a lot of effort and time going through thousands of articles - many of which do not contain any relevant information.

The evaluation procedure studied the effect of the presence of interaction verbs on the relation extraction. Unsurprisingly, the results showed that the presence of the interaction verb in the sentence improved the model's ability to predict functional relationships. Although the presence of the interaction verb in a sentence can improve the performance of the model, the real data includes sentences with and without interaction verbs; thus the predictive model has been built using the CPI-DS dataset, which includes sentences with and without interaction verbs to reflect the real dataset.

Although PubTator Central (PTC) provides a useful service in the field of identifying biomedical entities, some random checks gave the impression that the model is overfitting, i.e. the model cannot perform accurately against unseen data. According to a simple examination of the outputs of entity recognition for chemical compounds and proteins when PubTator tested on a real dataset (unseen dataset), it showed that the quality is not as good as the tool's performance on the benchmark dataset. Another named entity recognition tool called BERN2 (Advanced Biomedical Entity Recognition and Normalization) was presented recently. BERN2 is a biomedical text mining tool based on BioBERT for biomedical named entity recognition (NER) and named entity normalization (NEN). It

combines rule-based and neural network-based NEN models to improve the quality of entity normalization. BERN2 shows a better performance than Pubtator. A replacement of Pubtator with BERN2 in the CPRIL pipeline might further improve the prediction quality. The disadvantages of BERN are: it does not offer a bulk download facility to download the complete dataset; and the API is limited to 3 requests per second (for processing 33M articles it will take more than 4 months). Having an accurate entity recognition tool will improve the overall performance of the relation extraction process because the entity recognition process is considered a cornerstone of the relation extraction process.

Furthermore, the full text contains a significant amount of information which is not available in the abstract. Thus, an extension of CPRiL that includes not only the titles and abstracts but also the full text of biomedical and life sciences journal literature (PMC) will enable the inclusion of more relationships which may not appear in the abstracts. The availability of full texts of all articles would have a great impact on the community and would help to extend the available data for training novel models dramatically.

In the task extraction of compound-protein functional relationships, the focus is more on the accurate identification of positive relationships rather than negatives (no relationship), i.e. precision and recall are more significant than specificity. Although combining more than one productive model can decrease the false positives (or increase precision), a lot of positive pairs are lost (worse recall). In general, the overall performance of the model was not improved, i.e. the individual approach of ”BioBERT” performs better overall than the combination of more than one method. The combination of more than one model depends strongly on the task which needs to be solved and the experimenter must balance between precision and recall. Combining more than one method can be helpful if high precision is more important than recall (losing positive pairs), but this is not the case in the task of extracting compound-protein functional relationships.

The same procedure of extracting functional compound-protein relationships can be used to extract the functional relationships among other biomedical entities such as relationships

between drug and diseases, between drugs, between gene and diseases, and between proteins. Getting this information and linking all the entities' relationships in one network might give a comprehensive view on the inter- and intra-relationships of all entities; this might be of immense help for the development of novel data mining approaches.

CHAPTER

5 CONCLUSION AND OUTLOOK

Different artificial intelligence-based text mining models were tested and compared. The BioBERT model performed best and was applied to develop a fully automated web server (CPRiL) which identifies functional compound–protein relationships described in biomedical articles (PubMed) containing $\sim$ 33 million titles and abstracts. Cross-validated and tested results with a recall of 86.8%, precision of 85.2%, and an F_1-score of 86.0% represent a remarkable performance within the research area of relation extraction.

CPRiL presents the outputs of the search as a network view, which provides a simple overview and summary of the output results. Moreover, CPRiL finds the shortest path between two entities; this offers particularly helpful information, especially when the entities are indirectly related via relationships with other compound/protein entities. The application not only provides scientists and students with a good and quick overview of the functional relationships of individual compounds or proteins, but also enables experts to identify articles relevant to their specific field of research more comprehensively. Currently, CPRiL contains $\sim$ 2.5 million unique functional related compound-protein pairs, and all identified pairs are available for download.

In future versions, we will increase the size of the benchmark dataset and will examine how this can affect the performance of the model. In addition, in the future versions of CPRiL, data collection will be extended to include the full text of the biomedical and life sciences literature available in PubMed Central (PMC), which constitutes around 8.5M articles archived in PMC. Furthermore, in the next version, the functional relationships can be categorized, i.e., offer the type of the relationship (up-regulation, down-regulation, agonist, inhibitor, etc). One of the extensions in the next version will be adding more search options, such as searching by SMILES and structure of molecules. One of the future prospects of CPRiL is to be independent of PubTator to annotate the biomedical entities using deep learning approaches such as BioBERT; this can improve the performance of entity recognition. As a consequence, the overall performance of the relationship extraction might further increase. Moreover, the next version of CPRiL will include other functional relationships such as protein-protein, gene-disease, and drug-disease relationships.

APPENDICES

APPENDIX

BENCHMARK DATASET A

The full benchmark dataset in an XML format can be downloaded from this link:

http://histone.pharmazie.uni-freiburg.de/ftp/CPI/

APPENDIX

HOW TO USE THE EVALUATED METHODS

B

B.1 How to use the Shallow Linguistic Kernel (SL) and All-paths Graph Kernel (APG)

All the scripts and the documentation are freely available in the following repository of GitHub.

https://github.com/KerstenDoering/CPI-Pipeline [120].

In this repository, it is described how to run and use the predictive methods (SL and APG) with the combined benchmark dataset (CPI-DS), CPI-DS_IV, and CPI-DS_NIV. The productive methods come with three different modes:

- Cross-validation (CV): 10-fold cross-validation. In this mode, you can run a 10-fold cross-validation and generate the performance of each model.

- Prediction mode (PR): This mode uses the model which was trained on CPI-DS to predict user-specific dataset as test dataset.

- Cross-corpus (XX): This mode uses the predictive methods on user-specific datasets (training and test dataset).

B.2 How to use BioBERT

All the scripts and the documentation of BioBERT are available in the following GitHub repository of DMIS Laboratory - Korea University which is based on the original BERT code provided by Google in the following GitHub:
https://github.com/dmis-lab/biobert [121].
In this repository, it is described how to install and use the BioBERT.

1. Download the repository and the pre-trained weights BioBERT-Base v1.1 (+ PubMed 1M) from the GitHub.

2. Download the benchmark dataset from
   http://histone.pharmazie.uni-freiburg.de/ftp/CPI/

3. The following modes can be done:

   - For evaluation: split the benchmark dataset randomly into 70% training dataset and 30% test dataset.

   - To use the predictive method on user-specific datasets, use the benchmark dataset as training dataset and user-specific dataset as test dataset.

4. Finally, follow the instruction of Relation Extraction (RE) described in this repository to run the script.

B.3 Values of the other parameters that are used to evaluate BioBERT

Table B.1: The default value of the other main parameters that are used to evaluate BioBERT model.

Parameter	Description	Value
eval_batch_size	Total batch size for evaluation.	8
predict_batch_size	Total batch size for prediction.	8
optimizer	Optimization Algorithm	Adam
warmup_proportion	Proportion of training to perform linear learning rate warmup for E.g., 0.1 = 10% of training.	0.1
do_lower_case	Whether to lowercase the input text. Should be True for uncased models and False for cased models.	False

APPENDIX

WHITELIST VERBS (INTERACTION VERBS)

C

These verbs have been defined in the publication of the web service prolific (Senger and Grüning et al., 2012. Mining and evaluation of molecular relationships in literature. Bioinformatics).

Table C.1: Whitelist verbs (interaction verbs).

accelerate	degrading	expressed	methylate	remove
accelerates	degraded	extend	methylates	removes
accelerating	dehydrate	extends	methylating	removing
accelerated	dehydrates	extending	methylated	removed
acetylate	dehydrating	extended	migrate	reoxidize
acetylates	dehydrated	extinguish	migrates	reoxidizes
acetylating	dehydrogenate	extinguishes	migrating	reoxidizing
acetylated	dehydrogenates	extinguishing	migrated	reoxidized

acidify	dehydrogenating	extinguished	mimic	reoxidise
acidifes	dehydrogenated	farnesylate	mimics	reoxidises
acidifying	delay	farnesylates	mimicking	reoxidising
acidified	delays	farnesylating	mimicked	reoxidised
acquire	delaying	farnesylated	mineralize	reoxygenate
acquires	delayed	fill	mineralizes	reoxygenates
acquiring	delineate	fills	mineralizing	reoxygenating
acquired	delineates	filling	mineralized	reoxygenated
act	delineating	filled	mineralise	repair
acts	delineated	fix	mineralises	repairs
acting	demarcate	fixes	mineralising	repairing
acted	demarcates	fixing	mineralised	repaired
activate	demarcating	fixed	minimize	replicate
activates	demarcated	fixt	minimizes	replicates
activating	demethylate	generate	minimizing	replicating
activated	demethylates	generates	minimized	replicated
acylate	demethylating	generating	minimise	repolarize
acylates	demethylated	generated	minimises	repolarizes
acylating	demineralize	geranylate	minimising	repolarizing
acylated	demineralizes	geranylates	minimised	repolarized
add	demineralizing	geranylating	miss	repolarise
adds	demineralized	geranylated	misses	repolarises
adding	demineralise	glycate	missing	repolarising

added	demineralises	glycates	missed	repolarised
address	demineralising	glycating	mitigate	repress
addresses	demineralised	glycated	mitigates	represses
addressing	denature	graft	mitigating	repressing
addressed	denatures	grafts	mitigated	repressed
adsorb	denaturing	grafting	mobilize	resist
adsorbs	denatured	grafted	mobilizes	resists
adsorbing	deoxygenate	halogenate	mobilizing	resisting
adsorbed	deoxygenates	halogenates	mobilized	resisted
affect	deoxygenating	halogenating	mobilise	resolve
affects	deoxygenated	halogenated	mobilises	resolves
affecting	dephosphorylate	hamper	mobilising	resolving
affected	dephosphorylates	hamperes	mobilised	resolved
aggregate	dephosphorylating	hampering	moderate	resorb
aggregates	dephosphorylated	hampered	moderates	resorbs
aggregating	deplete	haptenize	moderating	resorbing
aggregated	depletes	haptenizes	moderated	resorbed
alleviate	depleting	haptenizing	modify	respond
alleviates	depleted	haptenized	modifies	responds
alleviating	depress	harbor	modifying	responding
alleviated	depresses	harbors	modified	responded
alter	depressing	harboring	modulate	restimulate
alters	depressed	harbored	modulates	restimulates

altering	deprive	harbour	modulating	restimulating
altered	deprives	harbours	modulated	restimulated
aminate	depriving	harbouring	monomerize	restore
aminates	deprived	harboured	monomerizes	restores
aminating	deprotonate	herniate	monomerizing	restoring
aminated	deprotonates	herniates	monomerized	restored
amplify	deprotonating	herniating	monoubiquitinate	restrain
amplifies	deprotonated	herniated	monoubiquitinates	restrains
amplifying	deregulate	heterodimerize	monoubiquitinating	restraining
amplified	deregulates	heterodimerizes	monoubiquitinated	restrained
antagonise	deregulating	heterodimerizing	move	retain
antagonises	deregulated	heterodimerized	moves	retains
antagonising	derepress	hinder	moving	retaining
antagonised	derepresses	hinders	moved	retained
antagonize	derepressing	hindering	mutagenize	retarget
antagonizes	derepressed	hindered	mutagenizes	retargets
antagonizing	derive	hydrate	mutagenizing	retargeting
antagonized	derives	hydrates	mutagenized	retargeted
arise	deriving	hydrating	need	reuse
arises	derived	hydrated	needs	reuses
arising	desalt	hydrogenate	needing	reusing
arose	desalts	hydrogenates	needed	reused
arisen	desalting	hydrogenating	neutralize	reverse

arize	desalted	hydrogenated	neutralizes	reverses
arizes	desensitize	hydrolyse	neutralizing	reversing
arizing	desensitizes	hydrolyses	neutralized	reversed
aroze	desensitizing	hydrolysing	neutralise	revert
arozen	desensitized	hydrolysed	neutralises	reverts
aromatize	desensitise	hydrolyze	neutralising	reverting
aromatizes	desensitises	hydrolyzes	neutralised	reverts
aromatizing	desensitising	hydrolyzing	nitrosylate	rise
aromatized	desensitised	hydrolyzed	nitrosylates	rises
aromatise	designate	hydroxylate	nitrosylating	rising
aromatises	designates	hydroxylates	nitrosylated	rose
aromatising	designating	hydroxylating	obviate	risen
aromatised	designated	hydroxylated	obviates	saturate
ascend	desorb	immobilize	obviating	saturates
ascends	desorbs	immobilizes	obviated	saturating
ascending	desorbing	immobilizing	occlude	saturated
ascended	desorbed	immobilized	occludes	seal
assemble	destabilze	immobilise	occluding	seals
assembles	destabilizes	immobilises	occluded	sealing
assembling	destabilizing	immobilising	occupy	sealed
assembled	destabilized	immobilised	occupies	secret
assign	destabilise	immortalize	occupying	secrete
assigns	destabilises	immortalizes	occupied	secreting

assigning	destabilising	immortalizing	open	secretting
assigned	destabilised	immortalized	opens	secreted
assimilate	destroy	immortilise	opening	secretted
assimilates	destroys	immortilises	opened	segregate
assimilating	destroying	immortilising	oppose	segregates
assimilated	destroyed	immortilised	opposes	segregating
associate	detach	immunize	opposing	segregated
associates	detaches	immunizes	opposed	sensitize
associating	detaching	immunizing	optimise	sensitizes
associated	detached	immunized	optimises	sensitizing
attack	detoxify	immunise	optimisiung	sensitized
attacks	detoxifies	immunises	optimised	sensitise
attacking	detoxifying	immunising	optimize	sensitises
attacked	detoxified	immunised	optimizes	sensitising
attract	deuterate	impact	optimizing	sensitised
attracts	deuterates	impacts	optimized	shift
attracting	deuterating	impacting	originate	shifts
attracted	deuterated	impacted	originates	shifting
augment	develop	impaire	originating	shifted
augments	developes	impairs	originated	shorten
augmenting	developing	impairing	osmoregulate	shortens
augmented	developed	impaired	osmoregulates	shortening
autophosphorylate	developt	impart	osmoregulating	shortened

autophosphorylates	differentiate	imparts	osmoregulated	simulate
autophosphorylating	differentiates	imparting	overload	simulates
autophosphorylated	differentiating	imparted	overloads	simulating
autoregulate	differentiated	impede	overloading	simulated
autoregulates	diffuse	impedes	overloaded	slow
autoregulating	diffuses	impeding	oxidize	slows
autoregulated	diffusing	impeded	oxidizes	slowing
bind	diffused	improve	oxidizing	slowed
binds	digest	improves	oxidized	solubilize
binding	digests	improving	oxidise	solubilizes
bound	digesting	improved	oxidises	solubilizing
bioactivate	digested	inactivate	oxidising	solubilized
bioactivates	dilute	inactivates	oxidised	solubilise
bioactivating	dilutes	inactivating	oxygenate	solubilises
bioactivated	diluting	inactivated	oxygenates	solubilising
biodegrade	diluted	include	oxygenating	solubilised
biodegrades	dimerize	includes	oxygenated	solve
biodegrading	dimerizes	including	palmitoylate	solves
biodegraded	dimerizing	included	palmitoylates	solving
biosynthesise	dimerized	incorporate	palmitoylating	solved
biosynthesises	dimerise	incorporates	palmitoylated	stabilize
biosynthesising	dimerises	incorporating	paralyze	stabilized
biosynthesised	dimerising	incorporated	paralyzes	stabilizes

biosynthesize	dimerised	increase	paralyzing	stabilizing
biosynthesizes	diminish	increases	paralyzed	stabilise
biosynthesizing	diminishes	increasing	paralyse	stabilises
biosynthesized	diminishing	increased	paralyses	stabilising
block	diminished	indicate	paralysing	stabilised
blocks	disable	indicates	paralysed	stain
blocking	disables	indicating	passage	staines
blocked	disabling	indicated	passages	staining
bring	disabled	induce	passaging	stained
brings	disaggregate	induces	passaged	start
bringing	disaggregates	inducing	penetrate	starts
brought	disaggregating	induced	penetrates	starting
brominate	disaggregated	infect	penetrating	started
brominates	displace	infects	penetrated	stimulate
brominating	displaces	infecting	perfuse	stimulates
brominated	displacing	infected	perfuses	stimulating
bury	displaced	infer	perfusing	stimulated
buries	disrupt	infers	perfused	stop
burying	disrupts	inferring	permeate	stops
buried	disrupting	interred	permeates	stopping
butylate	disrupted	influence	permeating	stopped
butylates	dissociate	influences	permeated	stretch
butylating	dissociates	influencing	permutate	stretches

butylated	dissociating	influenced	permutate	stretching
bypass	dissociated	inhabit	permutating	stretched
bypasses	dissolve	inhabits	permutated	substitute
bypassing	dissolves	inhabiting	perturb	substitutes
bypassed	dissolving	inhabited	perturbs	substituting
calcify	dissolved	inhibit	perturbing	substituted
calcifies	disturb	inhibits	perturbed	sulfonate
calcifying	disturbs	inhibiting	phosphorylate	sulfonates
calcified	disturbing	inhibited	phosphorylates	sulfonating
carbonate	disturbed	initiate	phosphorylating	sulfonated
carbonates	dock	initiates	phosphorylated	sulphate
carbonating	docks	initiating	photodissociate	sulphates
carbonated	docking	initiated	photodissociates	sulphating
carboxylate	docked	innervate	photodissociating	sulphated
carboxylates	down-regulate	innervates	photodissociated	sulfate
carboxylating	down-regulates	innervating	polarize	sulfates
carboxylated	down-regulating	innervated	polarizes	sulfating
carry	down-regulated	intensify	polarizing	sulfated
carries	downregulate	intensify	polarized	suppress
carrying	downregulates	intensifying	polarise	suppresses
carried	downregulating	intensified	polarises	suppressing
catabolize	downregulated	interact	polarising	suppressed
catabolizes	dye	interacts	polarised	sustain

catabolyzing	dyes	interacting	polymerize	sustains
catabolized	dyeing	interacted	polymerizes	sustaining
catalyse	dyed	intercalate	polymerizing	sustained
catalyses	dysregulate	intercalates	polymerized	synthesize
catalysing	dysregulates	intercalating	polymerise	synthesizes
catalysed	dysregulating	intercalated	polymerises	synthesizing
catalyze	dysregulated	interconnect	polymerising	synthesized
catalyzes	effect	interconnects	polymerised	synthesise
catalyzing	effects	interconnecting	polyubiquitinate	synthesises
catalyzed	effecting	interconnected	polyubiquitinates	synthesising
change	effected	interfere	polyubiquitinating	synthesised
changes	elevate	interferes	polyubiquitinated	tag
changing	elevates	interfering	potentiate	tags
changed	elevating	interfered	potentiates	tagging
charge	elevated	interlink	potentiating	tagged
charges	elicit	interlinks	potentiated	take
charging	elicits	interlinking	precondition	takes
charged	eliciting	interlinked	preconditiones	taking
chelate	elicited	interpenetrate	preconditioning	took
chelates	eliminate	interpenetrates	preconditioned	taken
chelating	eliminates	interpenetrating	prevent	target
chelated	eliminating	interpenetrated	prevented	targets
chlorinate	eliminated	interrupt	preventing	targeting

chlorinates	elongate	interrupts	prevented	targetting
chlorinating	elongates	interrupting	proceed	targeted
chlorinated	elongating	interrupted	proceeds	targetted
cleave	elongated	intersperse	proceeding	terminate
cleaves	elucidate	intersperses	proceeded	terminates
cleaving	elucidates	interspersing	process	terminating
cleaved	elucidating	interspersed	processes	terminated
cleft	elucidated	introgress	processing	transactivate
cloven	elute	introgresses	processed	transactivates
color	elutes	introgressing	produce	transactivating
colors	eluting	introgressed	produces	transactivated
coloring	eluted	invade	producing	transdifferentiate
colored	embed	invades	produced	transdifferentiates
colour	embeds	invading	progress	transdifferentiating
colours	embedding	invaded	progresses	transdifferentiated
colouring	embedded	investigate	progressing	transect
coloured	emit	investigates	progressed	transects
compete	emits	investigating	prohibit	transecting
competes	emitting	investigated	prohibits	transected
competing	emitted	invoke	prohibiting	transfect
competed	employ	invokes	prohibited	transfects
complement	employs	invoking	proliferat	transfecting
complements	employing	invoked	proliferats	transfected

complementing	employed	iodinate	proliferating	transfer
complemented	enable	iodinates	proliferated	transfers
compose	enables	iodinating	prolong	transferring
composes	enabling	iodinated	prolongs	transferred
composing	enabled	iodize	prolonging	transform
composed	enantioenriche	iodizes	prolonged	transforms
conjugate	enantioenriches	iodizing	promote	transforming
conjugates	enantioenriching	iodized	promotes	transformed
conjugating	enantioenriched	iodise	promoting	transition
conjugated	encapsulate	iodises	promoted	transits
connect	encapsulates	iodising	prompt	transiting
connects	encapsulating	iodised	prompts	transited
connecting	encapsulated	join	prompting	transition
connected	enclose	joins	prompted	transitions
consist	encloses	joining	protect	transitioning
consists	enclosing	joined	protects	transitioned
consisting	enclosed	keep	protecting	transmigrate
consisted	enforce	keeps	protected	transmigrates
contain	enforces	keeping	proteolyze	transmigrating
contains	enforcing	kept	proteolyzes	transmigrated
containing	enforced	kill	proteolyzing	transmit
contained	engage	kills	proteolyzed	transmits
contaminate	engages	killing	protonate	transmitting

contaminates	engaging	killed	protonates	transmitted
contaminating	engaged	label	protonating	transport
contaminated	engulf	labels	protonated	transporting
convert	engulfs	labelling	protract	transports
converts	engulfing	labeling	protracts	transported
converting	engulfed	labelled	protracting	transpose
converted	enhance	labeled	protracted	transposes
counteract	enhances	lessen	provide	transposing
counteracts	enhancing	lessens	provides	transposed
counteracting	enhanced	lessening	providing	trigger
counteracted	enlarge	lessened	provided	triggers
cross-react	enlarges	level	radiolabel	triggering
cross-reacts	enlarging	levels	radiolabels	triggered
cross-reacting	enlarged	levelling	radiolabeling	trimethylate
cross-reacted	enrich	leveling	radiolabelling	trimethylates
crossreact	enriches	leveled	radiolabeled	trimethylating
crossreacts	enriching	levelled	radiolabelled	trimethylated
crossreacting	enriched	liberate	raise	ubiquinate
crossreacted	enter	liberates	raises	ubiquinates
curtail	enters	liberating	raising	ubiquinating
curtails	entering	liberated	raised	ubiquinated
curtailing	entered	ligate	reabsorb	ubiquitinate
curtailed	entrap	ligates	reabsorbs	ubiquitinates

damage	entraps	ligating	reabsorbing	ubiquitinating
damages	entrapping	ligated	reabsorbed	ubiquitinated
damaging	entrapped	limit	react	ubiquitinylate
damaged	envelope	limits	reacts	ubiquitinylates
deacetylate	envelopes	limiting	reacting	ubiquitinylating
deacetylates	enveloping	limited	reacted	ubiquitinylated
deacetylating	enveloped	link	reactivate	ubiquitylate
deacetylated	eradicate	links	reactivates	ubiquitylates
deactivate	eradicates	linking	reactivating	ubiquitylating
deactivates	eradicating	linked	reactivated	ubiquitylated
deactivating	eradicated	lock	reassemble	uncovere
deactivated	escalate	locks	reassembles	uncoveres
deafferent	escaltes	locking	reassembling	uncovering
deafferents	escalating	locked	reassembled	uncovered
deafferenting	escalated	loose	rebind	upmodulate
deafferented	escape	looses	rebinds	upmodulates
deamidate	escapes	loosing	rebinding	upmodulating
deamidates	escaping	loosed	rebound	upmodulated
deamidating	escaped	lower	receive	up-regulate
deamidated	establishe	lowers	receives	up-regulates
deaminate	establishes	lowering	receiving	up-regulating
deaminates	establishing	lowered	received	up-regulated
deaminating	established	lyse	reconstitute	upregulate

deaminated	esterify	lyses	reconstitutes	upregulates
dearomatize	esterifies	lysing	reconstituting	upregulating
dearomatizes	esterifying	lysed	reconstituted	upregulated
deamoratizing	esterified	maintain	recruit	use
dearomatized	ethoxylate	maintains	recruits	uses
dearomatise	ethoxylate	maintaining	recruiting	using
dearomatises	ethoxylating	maintained	recruited	used
dearomatising	ethoxylated	manipulate	reduce	utilize
dearomatised	evade	manipulates	reduces	utilizes
decaffeinate	evades	manipulating	reducing	utilizing
decaffeinates	evading	manipulated	reduced	utilized
decaffeinating	evaded	mark	regenerate	utilise
decaffeinated	evoke	marks	regenerates	utilises
decarboxylate	evokes	marking	regenerating	utilising
decarboxylates	evoking	marked	regenerated	utilised
decarboxylating	evoked	match	regress	vary
decarboxylated	evolve	matches	regresses	varies
decelerate	evolves	matching	regressing	varying
decelerates	evolving	matched	regressed	varied
decelerating	evolved	maximize	regulate	weaken
decelerated	exacerbate	maximizes	regulates	weakens
dechlorinate	exacerbates	maximizing	regulating	weakening
dechlorinates	exacerbating	maximized	regulated	weakened

dechlorinating	exacerbated	maximise	reinforce	widen
dechlorinated	exaggerate	maximises	reinforces	widens
decline	exaggerates	maximising	reinforcing	widening
declines	exaggerating	maximised	reinforced	widened
declining	exaggerated	mediate	reinstate	wrap
declined	excrete	mediates	reinstates	wraps
decrease	excreted	mediating	reinstating	wrapping
decreases	excreting	mediated	reinstated	wrapped
decreasing	excreted	metabolize	reinvestigate	yield
decreased	expand	metabolizes	reinvestigates	yields
defat	expands	metabolizing	reinvestigating	yielding
defats	expanding	metabolized	reinvestigated	yielded
defatting	expanded	metabolise	release	
defatted	expose	metabolises	releases	
degenerate	exposes	metabolising	releasing	
degenerates	exposing	metabolised	released	
degenerating	exposed	metallate	relieve	
degenerated	express	metallates	relieves	
degrade	expresses	metallating	relieving	
degrades	expressing	metallated	relieved	

BIBLIOGRAPHY

[1] Domonkos Tikk, Philippe Thomas, Peter Palaga, Jörg Hakenberg, and Ulf Leser. A comprehensive benchmark of kernel methods to extract protein–protein interactions from literature. *PLoS computational biology*, 6(7):e1000837, 2010.

[2] Damian Szklarczyk, Alberto Santos, Christian Von Mering, Lars Juhl Jensen, Peer Bork, and Michael Kuhn. STITCH 5: augmenting protein–chemical interaction networks with tissue and affinity data. *Nucleic acids research*, 44(D1):D380–D384, 2016.

[3] Chih-Hsuan Wei, Alexis Allot, Robert Leaman, and Zhiyong Lu. PubTator central: automated concept annotation for biomedical full text articles. *Nucleic acids research*, 47(W1):W587–W593, 2019.

[4] Mujeen Sung, Minbyul Jeong, Yonghwa Choi, Donghyeon Kim, Jinhyuk Lee, and Jaewoo Kang. BERN2: an advanced neural biomedical named entity recognition and normalization tool. *Bioinformatics*, 38(20):4837–4839, 09 2022.

[5] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

[6] Lynette Hirschman, Alexander Yeh, Christian Blaschke, and Alfonso Valencia. Overview of BioCreAtIvE: critical assessment of information extraction for biology. *BMC Bioinformatics*, 6(S1), May 2005.

[7] Renxiao Wang, Xueliang Fang, Yipin Lu, Chao-Yie Yang, and Shaomeng Wang. The PDBbind database: methodologies and updates. *Journal of Medicinal Chemistry*, 48(12):4111–4119, June 2005.

[8] Anna Gaulton, Anne Hersey, Michał Nowotka, A. Patrícia Bento, Jon Chambers, David Mendez, Prudence Mutowo, Francis Atkinson, Louisa J. Bellis, Elena Cibrián-Uhalte, Mark Davies, Nathan Dedman, Anneli Karlsson, María Paula Magariños, John P. Overington, George Papadatos, Ines Smit, and Andrew R. Leach. The ChEMBL database in 2017. *Nucleic Acids Research*, 45(D1):D945–D954, January 2017.

[9] Meik Kunz, Chunguang Liang, Santosh Nilla, Alexander Cecil, and Thomas Dandekar. The drug-minded protein interaction database (DrumPID) for efficient target analysis and drug development. *Database: The Journal of Biological Databases and Curation*, 2016:baw041, 2016.

[10] David S. Wishart, Yannick D. Feunang, An C. Guo, Elvis J. Lo, Ana Marcu, Jason R. Grant, Tanvir Sajed, Daniel Johnson, Carin Li, Zinat Sayeeda, Nazanin Assempour, Ithayavani Iynkkaran, Yifeng Liu, Adam Maciejewski, Nicola Gale, Alex Wilson, Lucy Chin, Ryan Cummings, Diana Le, Allison Pon, Craig Knox, and Michael Wilson. DrugBank 5.0: a major update to the DrugBank database for 2018. *Nucleic Acids Research*, 46(D1):D1074–D1082, January 2018.

[11] Allan Peter Davis, Cynthia J Grondin, Robin J Johnson, Daniela Sciaky, Jolene Wiegers, Thomas C Wiegers, and Carolyn J Mattingly. Comparative toxicogenomics database (CTD): update 2021. *Nucleic acids research*, 49(D1):D1138–D1143, 2021.

[12] Damian Szklarczyk, John H. Morris, Helen Cook, Michael Kuhn, Stefan Wyder, Milan Simonovic, Alberto Santos, Nadezhda T. Doncheva, Alexander Roth, Peer Bork,

Lars J. Jensen, and Christian von Mering. The STRING database in 2017: quality-controlled protein-protein association networks, made broadly accessible. *Nucleic Acids Research*, 45(D1):D362–D368, January 2017.

[13] Fabio Rinaldi, Simon Clematide, Hernani Marques, Tilia Ellendorff, Martin Romacker, and Raul Rodriguez-Esteban. OntoGene web services for biomedical text mining. *BMC bioinformatics*, 15 Suppl 14:S6, 2014.

[14] Martin Krallinger, Obdulia Rabal, Saber A Akhondi, Martın Pérez Pérez, Jesús Santamaría, Gael Pérez Rodríguez, Georgios Tsatsaronis, Ander Intxaurrondo, José Antonio López, Umesh Nandal, and others. Overview of the BioCreative VI chemical-protein interaction Track. In *Proceedings of the sixth BioCreative challenge evaluation workshop*, volume 1, pages 141–146, 2017.

[15] Antonio Miranda, Farrokh Mehryary, Jouni Luoma, Sampo Pyysalo, Alfonso Valencia, and Martin Krallinger. Overview of DrugProt BioCreative VII track: quality evaluation and large scale text mining of drug-gene/protein relations. In *Proceedings of the seventh BioCreative challenge evaluation workshop*, 2021.

[16] Tom M Mitchell and Tom M Mitchell. *Machine learning*, volume 1. McGraw-hill New York, 1997.

[17] Zhi-Hua Zhou. *Machine learning*. Springer Nature, 2021.

[18] Pádraig Cunningham, Matthieu Cord, and Sarah Jane Delany. Supervised learning. In *Machine learning techniques for multimedia*, pages 21–49. Springer, 2008.

[19] Zoubin Ghahramani. Unsupervised learning. In *Summer school on machine learning*, pages 72–112. Springer, 2003.

[20] Thomas Hofmann, Bernhard Schölkopf, and Alexander J Smola. Kernel methods in machine learning. *The annals of statistics*, 36(3):1171–1220, 2008.

[21] Thomas Hofmann, Bernhard Schölkopf, and Alexander J Smola. A review of kernel methods in machine learning. *Max-Planck-Institute Technical Report*, 156, 2006.

[22] Martin Sewell. Kernel methods. *Department of Computer Science, University College London*, 2009.

[23] Chris M Bishop. Neural networks and their applications. *Review of scientific instruments*, 65(6):1803–1832, 1994.

[24] James A Anderson. *An introduction to neural networks*. MIT press, 1995.

[25] Berndt Müller, Joachim Reinhardt, and Michael T Strickland. *Neural networks: an introduction*. Springer Science & Business Media, 1995.

[26] Jeannette Lawrence. *Introduction to neural networks*. California Scientific Software, 1993.

[27] What are Neural Networks?, August 2021. URL: https://www.ibm.com/cloud/learn/neural-networks.

[28] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[29] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[30] Johannes Ludwig Vrana and Ripudaman Singh. Nde 4.0 from design thinking to strategy. *arXiv preprint arXiv:2003.07773*, 2020.

[31] Md Zahangir Alom, Tarek M Taha, Chris Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Mahmudul Hasan, Brian C Van Essen, Abdul AS Awwal, and Vijayan K Asari. A state-of-the-art survey on deep learning theory and architectures. *Electronics*, 8(3):292, 2019.

[32] AI vs. Machine Learning vs. Deep Learning: What's the Difference? | Built In. URL: https://builtin.com/artificial-intelligence/ai-vs-machine-learning.

[33] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association*

for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[34] Jay Alammar. The Illustrated Transformer. URL: https://jalammar.github.io/illustrated-transformer/.

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, \Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[36] Jay Alammar. The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning). URL: https://jalammar.github.io/illustrated-bert/.

[37] Ah-Hwee Tan. Text mining: The state of the art and the challenges. In *Proceedings of the Pacific Asia Conf on Knowledge Discovery and Data Mining PAKDD'99 workshop on Knowledge Discovery from Advanced Databases (KDAD'99)*, volume 8, pages 65–70. Citeseer, 1999.

[38] Andreas Hotho, Andreas Nürnberger, and Gerhard Paaß. A brief survey of text mining. *LDV Forum - GLDV Journal for Computational Linguistics and Language Technology*, 20(1):19–62, May 2005.

[39] Marti Hearst. What is text mining. *SIMS, UC Berkeley*, 5, 2003.

[40] Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman. Natural language processing: an introduction. *Journal of the American Medical Informatics Association*, 18(5):544–551, 2011.

[41] KR1442 Chowdhary. Natural language processing. *Fundamentals of artificial intelligence*, pages 603–649, 2020.

[42] Hang Li. Deep learning for natural language processing: advantages and challenges. *National Science Review*, 5(1):24–26, 2018.

[43] Elizabeth D Liddy. Natural language processing. In *Encyclopedia of Library and Information Science, 2nd Ed*. Marcel Decker, Inc, 2001.

[44] Yue Kang, Zhao Cai, Chee-Wee Tan, Qian Huang, and Hefu Liu. Natural language processing (NLP) in management research: A literature review. *Journal of Management Analytics*, 7(2):139–172, 2020.

[45] Gregory Grefenstette. Tokenization. In *Syntactic Wordclass Tagging*, pages 117–133. Springer, 1999.

[46] Jonathan J Webster and Chunyu Kit. Tokenization as the initial phase in NLP. In *COLING 1992 volume 4: The 14th international conference on computational linguistics*, 1992.

[47] Christopher D Manning. *Introduction to information retrieval*. Syngress Publishing„ 2008.

[48] Amit Singhal and others. Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, 24(4):35–43, 2001.

[49] Gobinda G Chowdhury. *Introduction to modern information retrieval*. Facet publishing, 2010.

[50] Behrang Mohit. Named entity recognition. In *Natural language processing of semitic languages*, pages 221–245. Springer, 2014.

[51] Alireza Mansouri, Lilly Suriani Affendey, and Ali Mamat. Named entity recognition approaches. *International Journal of Computer Science and Network Security*, 8(2):339–344, 2008.

[52] Jalaj Thanaki. *Python natural language processing*. Packt Publishing Ltd, 2017.

[53] Sonit Singh. Natural language processing for information extraction. *arXiv preprint arXiv:1807.02383*, 2018.

[54] Sotiris B Kotsiantis, Ioannis D Zaharakis, and Panayiotis E Pintelas. Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review*, 26(3):159–190, 2006.

[55] M Ikonomakis, Sotiris Kotsiantis, and V Tampakas. Text classification using machine learning techniques. *WSEAS transactions on computers*, 4(8):966–974, 2005.

[56] Rob Schapire. Machine learning algorithms for classification. *Princeton University*, 10, 2015.

[57] Michael W Browne. Cross-validation methods. *Journal of mathematical psychology*, 44(1):108–132, 2000.

[58] Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-validation. *Encyclopedia of database systems*, 5:532–538, 2009.

[59] Mervyn Stone. Cross-validation: A review. *Statistics: A Journal of Theoretical and Applied Statistics*, 9(1):127–139, 1978.

[60] Ron Kohavi and others. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995. Issue: 2.

[61] Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4(none):40–79, January 2010.

[62] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[63] Tom Fawcett. An introduction to ROC analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[64] David MW Powers. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*, 2020.

[65] Claude Sammut and Geoffrey I Webb. *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

[66] S Madeh Piryonesi and Tamer E El-Diraby. Data analytics in asset management: Cost-effective prediction of the pavement condition index. *Journal of Infrastructure Systems*, 26(1):04019036, 2020.

[67] Douglas Brent West and others. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001.

[68] Norman Biggs, E Keith Lloyd, and Robin J Wilson. *Graph Theory, 1736-1936*. Oxford University Press, 1986.

[69] William Thomas Tutte and William Thomas Tutte. *Graph theory*, volume 21. Cambridge university press, 2001.

[70] Jonathan L Gross and Jay Yellen. *Handbook of graph theory*. CRC press, 2003.

[71] Ronald Gould. *Graph theory*. Courier Corporation, 2012.

[72] Di Yan, Tao Wu, Ying Liu, and Yang Gao. An efficient sparse-dense matrix multiplication on a multicore system. In *2017 IEEE 17th International Conference on Communication Technology (ICCT)*, pages 1880–1883. IEEE, 2017.

[73] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[74] Jeff Forcier, Paul Bissex, and Wesley J Chun. *Python web development with Django*. Addison-Wesley Professional, 2008.

[75] Daniel Rubio. *Beginning Django*. Springer, 2017.

[76] Bruce Momjian. *PostgreSQL: introduction and concepts*, volume 192. Addison-Wesley New York, 2001.

[77] Behandelt PostgreSQL. PostgreSQL. *Web resource: http://www. PostgreSQL. org/about*, 1996.

[78] Greg Landrum. Rdkit documentation. *Release*, 1(1-79):4, 2013.

[79] Greg Landrum and others. RDKit: A software suite for cheminformatics, computational chemistry, and predictive modeling. *Greg Landrum*, 2013.

[80] Martín Abadi. TensorFlow: learning functions at scale. In *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*, pages 1–1, 2016.

[81] Joshua V Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matt Hoffman, and Rif A Saurous. Tensorflow distributions. *arXiv preprint arXiv:1711.10604*, 2017.

[82] Peter Goldsborough. A tour of tensorflow. *arXiv preprint arXiv:1610.01178*, 2016.

[83] Nishant Shukla and Kenneth Fricklas. *Machine learning with TensorFlow*. Manning Greenwich, 2018.

[84] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, and others. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[85] Bo Pang, Erik Nijkamp, and Ying Nian Wu. Deep learning with tensorflow: A review. *Journal of Educational and Behavioral Statistics*, 45(2):227–248, 2020.

[86] Giancarlo Zaccone. *Getting started with TensorFlow*. Packt Publishing Birmingham, 2016.

[87] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using networkx. In Gaël Varoquaux, Travis Vaught, and Jarrod Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.

[88] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, and others. PubChem in 2021: new data content and improved web interfaces. *Nucleic acids research*, 49(D1):D1388–D1395, 2021.

[89] UniProt Consortium. UniProt: the universal protein knowledgebase in 2021. *Nucleic Acids Research*, 49(D1):D480–D489, January 2021.

[90] Björn A Grüning, Christian Senger, Anika Erxleben, Stephan Flemming, and Stefan Günther. Compounds In Literature (CIL): screening for compounds and relatives in PubMed. *Bioinformatics*, 27(9):1341–1342, 2011.

[91] Christian Senger, Björn A Grüning, Anika Erxleben, Kersten Döring, Hitesh Patel, Stephan Flemming, Irmgard Merfort, and Stefan Günther. Mining and evaluation of molecular relationships in literature. *Bioinformatics*, 28(5):709–714, 2012.

[92] Kristina M Hettne, Rob H Stierum, Martijn J Schuemie, Peter JM Hendriksen, Bob JA Schijvenaars, Erik M van Mulligen, Jos Kleinjans, and Jan A Kors. A dictionary to identify small molecules and drugs in free text. *Bioinformatics*, 25(22):2983–2991, 2009.

[93] Dietrich Rebholz-Schuhmann, Miguel Arregui, Sylvain Gaudan, Harald Kirsch, and Antonio Jimeno. Text processing through Web services: calling Whatizit. *Bioinformatics*, 24(2):296–298, 2008.

[94] Gopalakrishnan Ramakrishnan, Sundaram Jagan, Sattu Kamaraj, Pandi Anandakumar, and Thiruvengadam Devaki. Silymarin attenuated mast cell recruitment thereby decreased the expressions of matrix metalloproteinases-2 and 9 in rat liver carcinogenesis. *Investigational New Drugs*, 27(3):233–240, June 2009.

[95] Jin Kyung Rho, Yun Jung Choi, Jin Kyung Lee, Baek-Yeol Ryoo, Im Il Na, Sung Hyun Yang, Cheol Hyeon Kim, and Jae Cheol Lee. Epithelial to mesenchymal transition derived from repeated exposure to gefitinib determines the sensitivity to EGFR in-

hibitors in A549, a non-small cell lung cancer cell line. *Lung Cancer (Amsterdam, Netherlands)*, 63(2):219–226, February 2009.

[96] Mugdha N Harmalkar and Neelam V Shirsat. Staurosporine-induced growth inhibition of glioma cells is accompanied by altered expression of cyclins, CDKs and CDK inhibitors. *Neurochemical research*, 31(5):685–692, 2006.

[97] Antti Airola, Sampo Pyysalo, Jari Björne, Tapio Pahikkala, Filip Ginter, and Tapio Salakoski. All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. *BMC bioinformatics*, 9(11):1–12, 2008.

[98] Claudio Giuliano, Alberto Lavelli, and Lorenza Romano. Exploiting shallow linguistic information for relation extraction from biomedical literature. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 401–408, 2006.

[99] Domonkos Tikk, Illés Solt, Philippe Thomas, and Ulf Leser. A detailed error analysis of 13 kernel methods for protein-protein interaction extraction. *BMC bioinformatics*, 14:12, January 2013.

[100] Renata Kabiljo, Andrew B. Clegg, and Adrian J. Shepherd. A realistic assessment of methods for extracting gene/protein interactions from free text. *BMC bioinformatics*, 10:233, July 2009.

[101] Thomas Gärtner, Peter Flach, and Stefan Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Learning theory and kernel machines*, pages 129–143. Springer, 2003.

[102] Ryan Rifkin, Gene Yeo, Tomaso Poggio, and others. Regularized least-squares classification. *Nato Science Series Sub Series III Computer and Systems Sciences*, 190:131–154, 2003.

[103] Carolyn E Lipscomb. Medical subject headings (MeSH). *Bulletin of the Medical Library Association*, 88(3):265, 2000.

[104] Garth R Brown, Vichet Hem, Kenneth S Katz, Michael Ovetsky, Craig Wallin, Olga Ermolaeva, Igor Tolstoy, Tatiana Tatusova, Kim D Pruitt, Donna R Maglott, and others. Gene: a gene-centered information resource at NCBI. *Nucleic acids research*, 43(D1):D36–D42, 2015.

[105] Robert Leaman and Zhiyong Lu. TaggerOne: joint named entity recognition and normalization with semi-Markov Models. *Bioinformatics*, 32(18):2839–2846, 2016.

[106] Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. BioCreative V CDR task corpus: a resource for chemical disease relation extraction. *Database*, 2016, 2016.

[107] Chih-Hsuan Wei, Hung-Yu Kao, and Zhiyong Lu. GNormPlus: an integrative approach for tagging genes, gene families, and protein domains. *BioMed research international*, 2015, 2015.

[108] Alexander A Morgan, Zhiyong Lu, Xinglong Wang, Aaron M Cohen, Juliane Fluck, Patrick Ruch, Anna Divoli, Katrin Fundel, Robert Leaman, Jörg Hakenberg, and others. Overview of BioCreative II gene normalization. *Genome biology*, 9(2):1–19, 2008.

[109] Edward Loper and Steven Bird. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*, 2002.

[110] Kairanbay Magzhan and Hajar Mat Jani. A review and evaluations of shortest path algorithms. *International journal of scientific & technology research*, 2(6):99–104, 2013.

[111] Giorgio Gallo and Stefano Pallottino. Shortest path algorithms. *Annals of operations research*, 13(1):1–79, 1988.

[112] Alan Bundy and Lincoln Wallen. Breadth-first search. In *Catalogue of artificial intelligence tools*, pages 13–13. Springer, 1984.

[113] Richard Bellman. On a routing problem. *Quarterly of applied mathematics*, 16(1):87–90, 1958.

[114] Lester R Ford Jr. Network flow theory. Technical report, Rand Corp Santa Monica Ca, 1956.

[115] Donald B Johnson. A note on Dijkstra's shortest path algorithm. *Journal of the ACM (JACM)*, 20(3):385–388, 1973.

[116] Michael L Fredman and Robert Endre Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3):596–615, 1987.

[117] D Randall Wilson and Tony R Martinez. The general inefficiency of batch training for gradient descent learning. *Neural networks*, 16(10):1429–1451, 2003.

[118] Choose optimal number of epochs to train a neural network in Keras, June 2020. Section: Machine Learning. URL: https://www.geeksforgeeks.org/choose-optimal-number-of-epochs-to-train-a-neural-network-in-keras/.

[119] Yi Jiang, Wanchao Yin, and H Eric Xu. RNA-dependent RNA polymerase: Structure, mechanism, and drug discovery for COVID-19. *Biochemical and biophysical research communications*, 538:47–53, 2021.

[120] Kersten Döring. Compound-Protein Interaction Pipeline, September 2022. original-date: 2016-02-11T09:36:34Z. URL: https://github.com/KerstenDoering/CPI-Pipeline.

[121] DMIS Laboratory Korea University. BioBERT, October 2022. original-date: 2019-01-24T18:27:35Z. URL: https://github.com/dmis-lab/biobert.