A HYBRID BERT-GAN SYSTEM FOR PROTEIN-PROTEIN INTERACTION EXTRACTION FROM BIOMEDICAL TEXT

by

Mert Basmacı

B.S., Computer Engineering, Middle East Technical University, 2016

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2021

ACKNOWLEDGEMENTS

First, and most of all, I would like to express my deepest gratitude to my supervisor, Assoc. Prof. Arzucan Özgür for her continuous support, guidance, and patience throughout my Master's education. The completion of this thesis would not have been possible without her expertise and deep knowledge.

I would like to thank my committee members, Prof. Tunga Güngör and Assoc. Prof. Gülşen Cebiroğlu Eryiğit, for kindly accepting to be in my thesis committee and for their valuable comments.

I would also like to thank Assoc. Prof. Junguk Hur and Assoc. Prof. Yongqun "Oliver" He for their contributions to our case study and their excessive efforts during COVID-19 dataset curation, annotation, and evaluation.

The most special thanks go to my best friends Gizem and Zeki for being influential in my decision to pursue my Master's education. They have always been very helpful in the most important decisions of my life since the day I met them. I would also like to extend my special thanks to my roommate Emre for being such a motivating friend.

Last but definitely not least, I would like to thank my family, my parents, İlknur and Tayfun, and my brother, Bora, for their endless love, encouragement, and support throughout my life. I would like to dedicate this thesis to my beloved grandparents Selahattin and Müzeyyen Ayrancıoğlu, my first and lifelong teachers. They are my heroes who have always believed in and encouraged me throughout my education and every day of my life.

ABSTRACT

A HYBRID BERT-GAN SYSTEM FOR PROTEIN-PROTEIN INTERACTION EXTRACTION FROM BIOMEDICAL TEXT

Considering the rapid increase in the biomedical literature, manual extraction of information regarding Protein-Protein Interactions (PPIs) becomes an exhausting task. Therefore, there is a strong need for the development of automatic relation extraction techniques from scientific publications. In this study, we introduce a novel two-stage system to extract PPIs from biomedical text. Our approach contains two cascaded stages. In the first stage, we utilize a transformer-based model, BioBERT, to determine whether pairs of proteins appearing in a sentence interact with each other; therefore, we perform a binary relation extraction task. In the second stage, we adopt a Generative Adversarial Network (GAN) model that consists of two contesting neural networks to eliminate false-positive predictions of the first stage. We evaluate the performance of both stages separately on five benchmark PPI corpora: AIMed, BioInfer, HPRD50, IEPA, and LLL. Later on, we combine the five corpora into a single source to examine the system performance on a general PPI corpus. Finally, we apply our system to a case study for Host-Pathogen Interaction extraction from the COVID-19 literature. The experimental results show that our first stage achieves the state-of-the-art F1-score of 79.0% on the AIMed corpus and obtains comparable results to previous studies on the other four corpora. Moreover, our second stage results reveal that the GAN model improves the first stage results when our BioBERT model is trained on the combined corpus. Our case study results demonstrate that the proposed system can be useful as a real-world application.

ÖZET

PROTEIN-PROTEIN ETKİLEŞİMİ ÇIKARIMI İÇİN HİBRİT BERT-GAN SİSTEMİ

Biyomedikal literatürdeki hızlı artış göz önünde bulundurulduğunda, Protein-Protein Etkileşimleri ile bilgilerin el ile çıkarılması zorlu bir iştir. Bu sebeple bilimsel yayınlardan otomatik ilişki çıkarma yöntemlerinin geliştirilmesine ihtiyaç vardır. Bu çalışmada, biyomedikal metinlerden Protein-Protein Etkileşimlerini çıkarmak için iki aşamalı yeni bir sistem sunulmaktadır. İlk aşamada, cümlelerde geçen protein çiftlerinin birbirleriyle etkileşime girip girmediklerini belirlemek için BioBERT adlı transformatör tabanlı bir model kullanılmaktadır ve dolayısıyla ikili ilişki çıkarma işlemi uygulanmaktadır. İkinci aşamada ise, ilk aşamadan gelen yanlış-pozitif tahminleri ayıklamak için birbiriyle yarışan iki sinir ağından oluşan Çekişmeli Üretici Ağ modeli kullanılmaktadır. Her iki aşamanın performansı AIMed, BioInfer, HPRD50, IEPA ve LLL adlı beş Protein-Protein etkileşimi veri kümesinde ayrı ayrı değerlendirilmektedir. Ardından, sistemin başarısı bu beş veri kümesi birleştirilerek elde edilen genel bir Protein-Protein Etkileşimi veri kümesinde incelenmektedir. Son olarak sistemimiz, COVID-19 yayınlarından Konak-Patojen Etkileşimlerini çıkardığımız örnek çalışmada denenmiştir. Deneysel sonuçlar, ilk aşamamızın AIMed veri kümesinde %79.0 F1 puanıyla önceki çalışmaları geçtiğini, diğer veri kümelerinde ise önceki çalışmalarla benzer sonuçlar elde ettiğini göstermektedir. İkinci aşama sonuçlarımız ise birleştirilmiş veri kümesi üzerinde Çekişmeli Üretici Ağ modelinin, ilk aşama sonuçlarını iyileştirdiğini göstermektedir. Örnek çalışmadan elde ettiğimiz sonuçlar ise, önerilen sistemin gerçek dünya uygulaması olarak faydalı olabileceğini ortaya koymaktadır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

B_i	ith bag
$D(G(z))$	The probability of the discriminator estimating a fake instance as real
$D(x)$	The discriminator's estimate of the probability when given sample x
E_x	Expected value over all real data instances
E_z	Expected value over all random instances
$G(z)$	The generator's estimate of the probability when given noise from latent space
L_D	The loss function of the discriminator
L_G	The loss function of the generator
$log(n)$	The logarithmic function of the value n
N	Number of bags
ND	Negative dataset to pretrain discriminator
NG	Negative dataset to pretrain generator
P	Positive dataset
s_j	jth sentence
z	Latent space
$*$	The convolution operator
$\sum$	The summation operator

LIST OF ACRONYMS/ABBREVIATIONS

ACE2	angiotensin-converting enzyme 2
APG	All-paths graph
BERT	Bidirectional Encoder Representations from Transformers
BioBERT	Bidirectional Encoder Representations from Transformers for Biomedical Text Mining
BioInfer	Bio Information Extraction Resource
BioNLP	Biomedical Natural Language Processing
BOW	Bag-of-words
CBOW	Continuous Bag-of-Words Model
CNN	Convolutional Neural Network
CSV	Comma Separated Values
CTK	Convolution Tree Kernel
DCNN	Deep Convolutional Neural Network
DDI	Drug-Drug Interaction
DIP	Database of Interacting Proteins
DNN	Deep Neural Network
DPP4	dipeptidyl peptidase 4
DPT	Dependency Parse Tree
DSTK	Distributed smoothed tree kernel
EDG	Extended Dependency Graph
GAN	Generative Adversarial Networks
HCoV	Human Coronavirus
HPI	Host-Pathogen Interaction
HPRD	Human Protein Reference Database
IEPA	Interaction Extraction Performance Assessment
IPT	Interaction Pattern Tree
kBSPS	k-Band Shortest Path Spectrum Kernel
KNN	K-Nearest Neighbor

LLL	Learning Language in Logic
LLL05	Learning Language in Logic 2005
LSTM	Long Short-Term Memory
MCCNN	Multichannel Convolutional Neural Network
McDepCNN	Multichannel Dependency-based Convolutional Neural Network
MIL	Multi-Instance Learning
MIML	Multi-Instance Multi-Label
MLM	Masked Language Model
MLP	Multi Layer Perceptron
NLP	Natural Language Processing
NNLM	Neural Net Language Model
OOV	Out-of-Vocabulary
PCNN	Piecewise Convolutional Neural Networks
PIPE	PPI Pattern Extraction
PPI	Protein-Protein Interaction
RELU	Rectified Linear Activation FunctioN
RNN	Recurrent Neural Network
SARS-CoV-2	Severe Acute Respiratory Syndrome Coronavirus 2
ST	Subset tree
SVM	Support Vector Machine

1. INTRODUCTION

Protein-Protein Interaction (PPI) is one of the key studies in biology. They are vital in biological processes such as cell metabolism, signal pathways, DNA replication and transcription, and conversion of genotypes to phenotypes [1]. In addition, the disruption of these processes often causes diseases and has a significant impact on the initiation or progression of pathology [2]. Therefore, the study of PPIs not only provides a better understanding of biological processes but also helps in the identification of several diseases and the development of potential treatments and therapies that are targeting interactions.

PPIs with their central role in biological processes examined in many biomedical studies and those studies are mostly published and stored in literature databases. PubMed, one of the main archives of biomedical and life sciences journal literature, as of today, comprises more than 32 million citations for the biomedical literature [3]. With the rapid advancements in technology and ease of access to the vast amount of research knowledge provided by literature databases, studies have gained momentum over the years. Figure 1.1 depicts, the number of indexed citations added to MEDLINE during each fiscal year since 1995 [4]. As seen in Figure 1.1, the number of citations over the years increase linearly. Hence, much information regarding protein interactions remains in those papers.

Furthermore, several databases and protein interaction networks such as STRING, Search Tool for the Retrieval of Interacting Genes/Proteins [5], BioGRID, The Biological General Repository for Interaction Datasets [6], IntAct, Molecular Interaction Database [7], BIND, Biomolecular Interaction Network Database [8], APID, Agile Protein Interactomes DataServer [9], and DIP, Database of Interacting Proteins [10] that empower the current knowledge on protein cascades are created. These databases store protein interactions in a structural format and provide easy access to previous work on proteins. They also shed light on scientists to explore possible protein targets and

interactions. However, most of them are created manually, and they store a small portion of the information available while the biomedical domain increases rapidly. Since it is an onerous and time-consuming task for database curators to detect these interactions from long papers, much information regarding protein interactions may remain hidden in those papers. Keeping these databases and interaction networks updated also requires manual effort; therefore, the development of information extraction and text mining techniques for automatic extraction and identification of PPIs from free texts is inevitable [11].

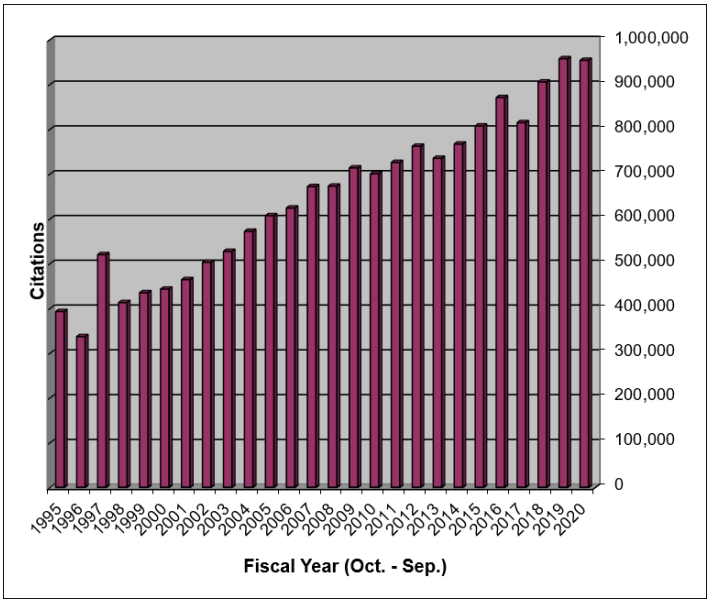


Figure 1.1. Number of indexed citations added to MEDLINE during each fiscal year since 1995.

This thesis focuses on extracting Protein-Protein Interactions from textual sentences applying deep learning techniques. BioBERT (Bidirectional Encoder Representations from Transformers for Biomedical Text Mining) [12] and GAN (Generative Adversarial Networks) [13] based hybrid model is introduced to accomplish this task. A two-stage ensemble learning model is applied such that those two models work cascaded way, or simply put one after another. In the first stage, the BioBERT model, a state-of-art method in the biomedical domain, predicts whether a sentence contains a protein interaction or not. Then, the predicted outputs of the BioBERT model are

given into the GAN model to eliminate the false positives. In other words, these two stages have different purposes. While the first stage predicts interactions, the second stage aims to increase performance. Our models are evaluated on various experiments by using five benchmark PPI datasets. Obtained results are compared with all related studies. In order to justify the performance improvement achieved by our second stage, several significance tests are conducted.

1.1. Contributions of the Thesis

The main contributions of this thesis are summarized as follows:

- We introduce a hybrid system for the Protein-Protein Interaction Extraction task.
- To the best of our knowledge, BioBERT and GAN models together are first examined for the Protein-Protein Interaction Extraction task.
- We train and test our models on Five Unified Benchmarking Corpora: AIMed, BioInfer, HPRD50, IEPA, and LLL. BioBERT and GAN models are first examined on Five Unified Benchmarking PPI Corpora to the best of our knowledge.
- Our first stage achieves state-of-the-art result on AIMed corpus. On other corpora, we obtain comparable results with previous state-of-the-art approaches.
- We obtain performance improvements with the GAN model on a generalized PPI corpus, and we provide significance test results to justify the improvement.
- We applied our model to a case study: Host-Pathogen Interaction Extraction on COVID-19 dataset. The results reveal that our system can be applied for a wide range of biomedical relation extraction tasks.

1.2. Organization of the Thesis

The rest of the thesis is organized into seven chapters. Chapter 2 gives background information about the terms that are used in the thesis. The protein interaction databases that are the primary resources for our dataset and have been used in several related works are described in this chapter. Moreover, background information about

word embeddings and deep learning models are also introduced in the chapter.

Chapter 3 gives a summary of the previous studies in the Protein-Protein Interaction Extraction domain. In addition, early kernel-based approaches and recent deep learning models applied in this domain are explained, and their performances on different datasets are provided in the chapter.

Chapter 4 provides detailed information about the Five Benchmarking PPI Corpora used in our study. The characterization of the corpora is given with their statistics. Unified Dataset Format is explained in this chapter as well.

Chapter 5 describes our proposed hybrid BERT-GAN system. The overall system architecture is given in this chapter. The technical aspects of our preprocessing method, word representation, hyperparameter settings, and model training details are explained.

Chapter 6 provides experiments and the results of our proposed model. First, we explain the evaluation metrics and applied statistical significance tests in this chapter. Then, we compare our model performance with other related studies experimented on the same dataset. Since our first stage itself is capable of extracting PPIs, we also report its performance separately. We provide cross-corpus and combined corpus evaluation results and significance test reports in the chapter.

Chapter 7 describes a case study for Host-Pathogen Interaction Extraction on COVID-19 dataset. Dataset curation details and performance evaluation of our system are reported in this chapter.

Chapter 8 provides final remarks, conclusion and future work.

2. BACKGROUND

2.1. Protein Interaction Databases

2.1.1. STRING

STRING, Search Tool for the Retrieval of Interacting Genes/Proteins, is a database that stores Protein-Protein Interactions, including both physical interactions as well as functional interactions [14]. It aims to integrate all known information regarding protein interactions so that it uses several resources like public text collection, automated text mining and computational prediction methods and experiments of interaction databases. Thus, it maps all interaction evidence in a single place. It also aims for comprehensive coverage, so it contains 52857362 protein interactions with high confidence(score $\geq$ 0.9) from 14000 organisms [15]. In addition, STRING provides a customizable user interface to demonstrate protein interaction networks, as seen in Figure 2.1.

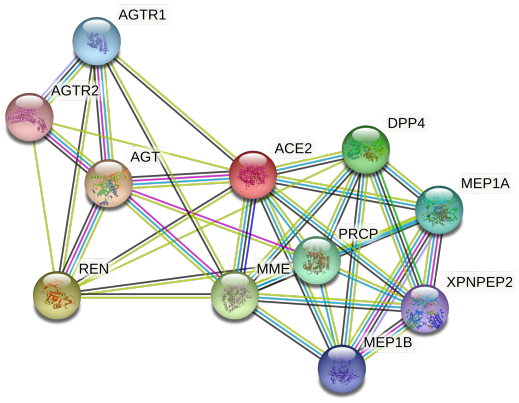


Figure 2.1. Interaction network of ACE2 protein generated by STRING.

2.1.2. BioGRID

BioGRID, The Biological General Repository for Interaction Datasets, a biomedical interaction database, consists of Protein-Protein Interactions, genetic interactions and chemical interactions. It is generated by comprehensive curation efforts containing compilation interaction records through structured evidence codes, phenotype ontologies, and gene annotations. As of today, it contains 2,045,743 protein and genetic interactions, 29,093 chemical interactions, and 1,070,825 posttranslational modifications from 76687 publications [16]. BioGRID data are freely distributed and available for download in many standardized formats. Thus, it is one of the primary literature for protein-protein interactions.

2.1.3. IntAct

IntAct is a freely available open-source database for molecular interaction data curated from literature or direct data resources. While IntAct is designed for representing any biomolecular interactions, their main focus is always Protein-Protein Interactions which are experimentally verified. Since all data is being annotated manually, it is a trustworthy source for Protein-Protein Interactions. The number of proteins and interactions of the IntAct database has been updated over the years, and it enlarged rapidly, as seen in Figure 2.2. As of today, IntAct contains 122,338 interactors and 1,139,018 interactions obtained through 22,368 biomedical publications and 72,393 experiments [17].

2.1.4. BIND

BIND, The Biomolecular Interaction Network Database stores interactions, molecular complexes, and pathways. According to BIND, two or more objects and a publication reference are required to describe a basic interaction. Those objects can be DNA, RNA, genes, proteins, or molecular complexes. BIND is manually curated, and each entry requires a PubMed publication reference and entry references to another database

such as GenBank [18]. Each entry within the database provides references/authors for the data. BIND database grows continually. As of 2004, it contains 28,266 Protein-Protein Interactions, 4,225 genetic interactions, and also 874 protein-small molecule interactions from 11,649 unique biomedical publications [19].

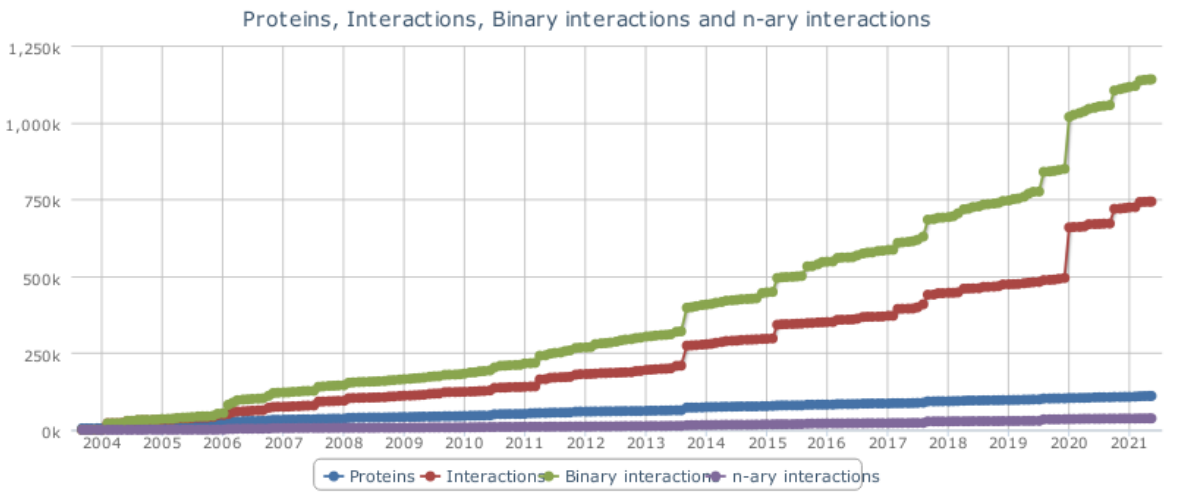
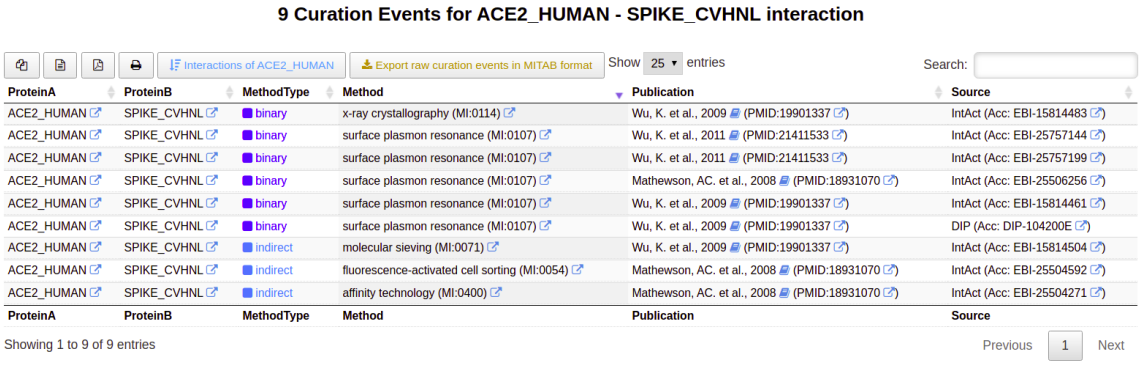


Figure 2.2. IntAct: Number of Proteins, Interactions, Binary interactions and n-ary interactions since 2004.

2.1.5. APID

APID, Agile Protein Interactomes DataServer is a protein interaction database of 400 organisms. It is manually curated, and consists of experimentally validated Protein-Protein Interactions. APID is a collection of several interaction databases and networks such as BIND, BioGRID, DIP, IntAct and provides a unified format for PPIs. APID is a comprehensive collection of 90,379 distinct proteins and 678,441 singular interactions [20]. It also provides a data visualization tool that allows the construction of protein interaction tables as seen in Figure 2.3 and visual exploration of the corresponding networks.



9 Curation Events for ACE2_HUMAN - SPIKE_CVHNL interaction

Interactions of ACE2_HUMAN | Export raw curation events in MITAB format | Show 25 entries | Search:

ProteinA	ProteinB	MethodType	Method	Publication	Source
ACE2_HUMAN	SPIKE_CVHNL	binary	x-ray crystallography (MI:0114)	Wu, K. et al., 2009 (PMID:19901337)	IntAct (Acc: EBI-15814483)
ACE2_HUMAN	SPIKE_CVHNL	binary	surface plasmon resonance (MI:0107)	Wu, K. et al., 2011 (PMID:21411533)	IntAct (Acc: EBI-25757144)
ACE2_HUMAN	SPIKE_CVHNL	binary	surface plasmon resonance (MI:0107)	Wu, K. et al., 2011 (PMID:21411533)	IntAct (Acc: EBI-25757199)
ACE2_HUMAN	SPIKE_CVHNL	binary	surface plasmon resonance (MI:0107)	Mathewson, AC. et al., 2008 (PMID:18931070)	IntAct (Acc: EBI-25506256)
ACE2_HUMAN	SPIKE_CVHNL	binary	surface plasmon resonance (MI:0107)	Wu, K. et al., 2009 (PMID:19901337)	IntAct (Acc: EBI-15814461)
ACE2_HUMAN	SPIKE_CVHNL	binary	surface plasmon resonance (MI:0107)	Wu, K. et al., 2009 (PMID:19901337)	DIP (Acc: DIP-104200E)
ACE2_HUMAN	SPIKE_CVHNL	indirect	molecular sieving (MI:0071)	Wu, K. et al., 2009 (PMID:19901337)	IntAct (Acc: EBI-15814504)
ACE2_HUMAN	SPIKE_CVHNL	indirect	fluorescence-activated cell sorting (MI:0054)	Mathewson, AC. et al., 2008 (PMID:18931070)	IntAct (Acc: EBI-25504592)
ACE2_HUMAN	SPIKE_CVHNL	indirect	affinity technology (MI:0400)	Mathewson, AC. et al., 2008 (PMID:18931070)	IntAct (Acc: EBI-25504271)
ProteinA	ProteinB	MethodType	Method	Publication	Source

Showing 1 to 9 of 9 entries | Previous | 1 | Next

Figure 2.3. APID: The interaction between two proteins (ACE2 and SPIKE) with nine curation events and four distinct pieces of experimental evidence.

2.1.6. DIP

DIP, The Database of Interacting Proteins, is a Protein-Protein Interaction network consisting of experimentally proven interactions from many different scientific journals and archives such as MEDLINE. It provides a comprehensive tool for browsing and extracting protein interactions. DIP contains valuable information regarding protein functions, protein-protein relationships, and properties of interacting protein networks. All protein entities have cross-references to at least one of the major protein databases such as PIR [21], SWISSPROT [22], and GenBank [18]. As of 2003, it contains 18,500 protein interactions from 2,500 research articles.

2.2. Word Embeddings

In Natural Language Processing (NLP), words are represented by vectors so that several vector calculus can be applied for different statistical analyses. Word embedding is a term used for the representation of words in the form of real-valued vectors. Those vectors are obtained using a set of language modelling and feature learning techniques, where words with similar meaning have a similar representation; in other words, they are closer to each other in the vector space [23].

There are different approaches for word embedding representations, and they are mainly classified into two categories as follows:

(i) Frequency-based methods such as count vector, TF-IDF vector, and Co-Occurance vector,
(ii) Prediction-based methods such as Word2Vec, GloVe, and fastText.

Frequency-based approaches work based on the count of a word in each document. They vectorize the words depending on their frequency of occurrences in documents, assuming that words in the same contexts share similar or related semantic meanings. Unlike the frequency-based approaches, prediction-based approaches build predictive models that directly target predicting a word given its neighbours. The quality of the resulting vector representations in prediction-based approaches are measured by considering not only similar words tend to be close to each other, but words can also have multiple degrees of similarities [24]. Typically, in predictive approaches, word vectors are initialized with random weights. These weights are updated through back-propagation to maximize the data log-likelihood under a neural network architecture that will produce word representations that hold both syntactic and semantic properties [25].

2.2.1. Word2Vec

Word2Vec is a prediction-based word representation approach. It is a two-layer feed-forward neural net language model (NNLM) [26] in which the hidden layers replaced by a projection layer. This projection layer depends heavily on the efficiency of the softmax normalization, which makes the NNLM model simpler but more efficient with the large amount of data [23]. Word2Vec is trained in two steps:

(i) First, Continuous Bag-of-Words Model (CBOW) is trained using simple NNLM architecture,
(ii) and then, the Skip-Gram model is trained using N-gram NNLM architecture.

The CBOW model has a simple NNLM architecture in which a single projection layer replaces the hidden layers. The CBOW model predicts the current word given surrounding context words. It averages all words given in the window so that the order of the surrounding context words does not affect the prediction; therefore, the term bag-of-words is used [23].

The Skip-Gram model has a similar NNLM architecture, but instead, it predicts the surrounding context words given a current word. Since distant words which are less relevant to the current word are sampled less, they have less weight on the obtained word vector [23]. The overall Word2Vec model architecture is shown in Figure 2.4.

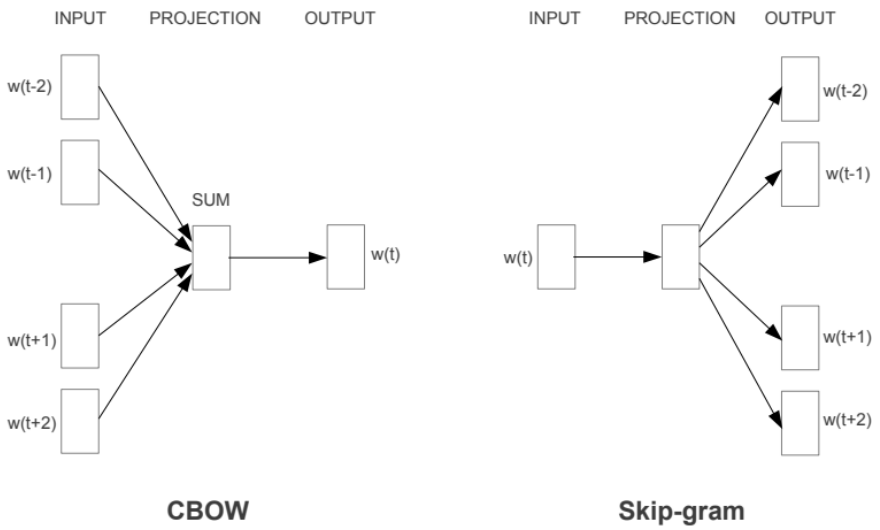


Figure 2.4. Word2Vec model architectures (taken from [23]). The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

2.3. Convolutional Neural Network

Convolutional Neural Networks (CNN) [27] is a kind of neural network architecture that has been widely used in several domains such as Signal Processing, Computer Vision, and Natural Language Processing, and it is tremendously successful in several

tasks. The name comes from a mathematical operation called convolution, which is a kind of linear operation. In essence, it consists of two real-valued functions given in Equation 2.1 where the first argument of the convolution x is called as the input, and the second argument w is called as the kernel in the deep learning terminology [28].

$$(x * w)(t) = \int x(a)w(t-a)da \tag{2.1}$$

In CNN architecture, having a smaller kernel size compared to the input size allows the kernel to travel the input array at different points systematically, and at these points, matrix multiplications are performed between the corresponding part of the input array and the kernel. Figure 2.5 illustrates a sample convolution process on 2-D vectors. The output is sometimes referred to as the feature map, which is the sparse representation of the input array. Since feature maps are smaller than input arrays in size, they reduce memory requirements and improves statistical efficiency [28].

In general, convolutional neural networks consists of three stages which are the convolution stage, detector stage and pooling stage [28]. After convolution is applied and the feature map is obtained, each value in the feature map are passed through a nonlinear function such as the rectified linear activation function (RELU). This stage is called as the detector stage. The purpose of the detector stage is to introduce nonlinearity into the network since the inputs contain various properties that are nonlinear to each other. Other nonlinear functions can also be used in this stage to produce nonlinearity, such as the logistic sigmoid function or tanh function. However, RELU outperforms others in practice; therefore, it is frequently used [29]. RELU is a very basic nonlinear function that replaces all negative elements in the feature map with zero. Figure 2.6 illustrates the RELU function.

After the detector stage, a pooling function is used to reduce the dimensionality of the feature map. Pooling function has different variants such as max pooling, average pooling and sum pooling etc. For example, in max-pooling [30] operation yields the

largest value within a portion of the feature map with a certain size, and in average pooling, it takes the average of values in that portion of the feature map [31]. Figure 2.7 illustrates the max pooling and average pooling functions.

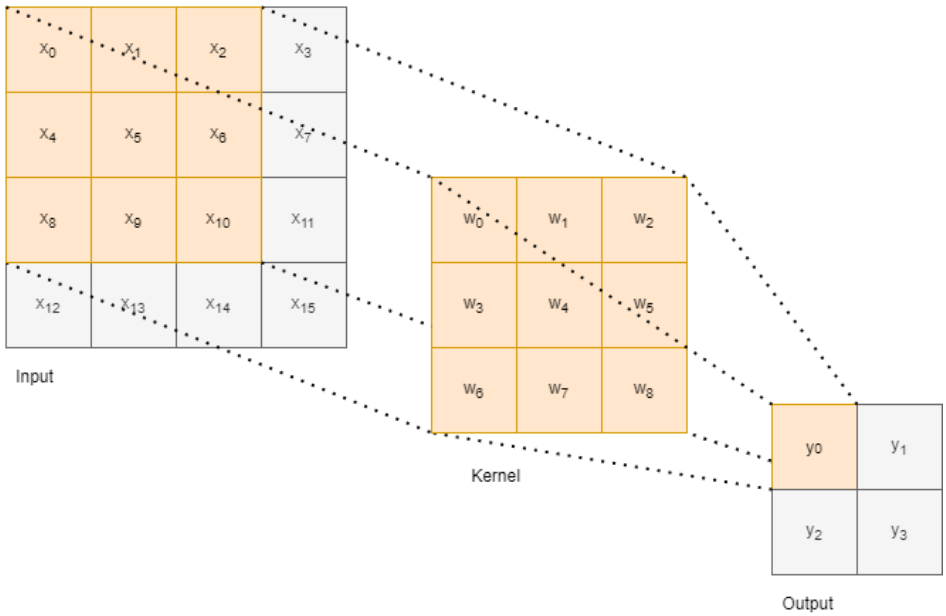


Figure 2.5. An example of 2-D convolution. Boxes with orange color are used to indicate how the upper-left element of the output tensor is formed by applying the kernel to the corresponding upper-left region of the input tensor.

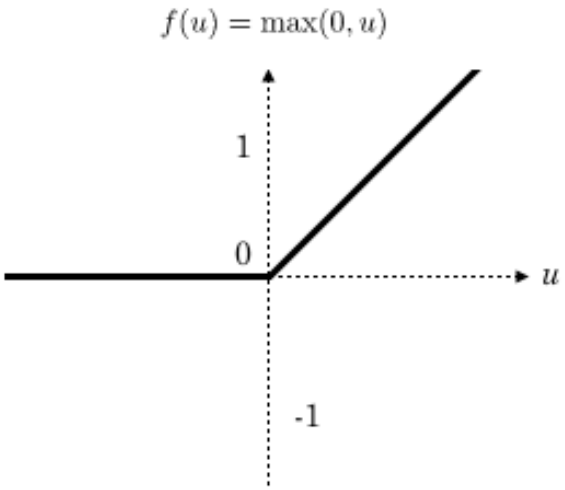


Figure 2.6. The RELU function.

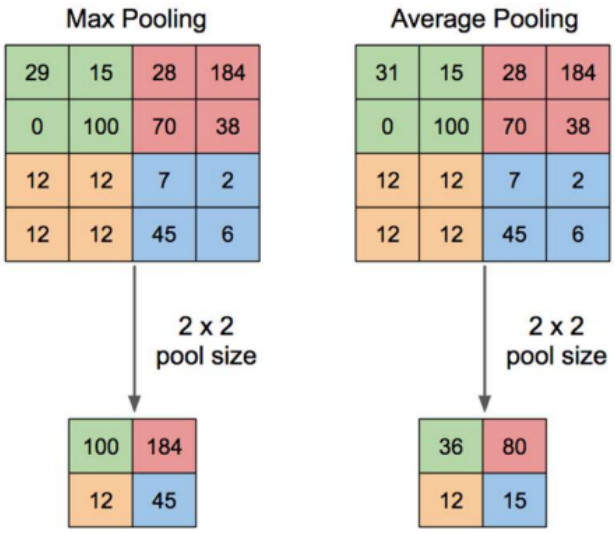


Figure 2.7. Illustration of Max Pooling and Average Pooling (taken from [31]).

2.4. Generative Adversarial Networks

Ian Goodfellow [13] introduced Generative adversarial networks (GANs), which are deep neural network architectures that consist of two neural networks, a generator G and a Discriminator D. GANs perform an adversarial learning process where two neural networks contesting with each other to generate new samples of data that can preserve similar patters to the original data. They are widely used in Computer Vision tasks such as image and video generation and super-resolution.

During the adversarial process, two neural networks have different purposes:

(i) The generative model G captures the data distribution and generates new samples that preserve the same statistics with the training samples.
(ii) The discriminator model D learns to distinguish the samples generated by the generative model from training samples.

Typically, both neural networks are multilayer perceptrons that optimize opposing objective functions. The generator starts with generating noisy samples $G(z)$ from a latent space z. Since each variable in the initial sample comes from a Gaussian distribution, they have no meaning at first. Hence, when the training begins, the discriminator assigns a low probability to fake samples and easily distinguishes them from the real ones. Through the training phase, the generator aims to learn the fooling discriminator; therefore the objective function of the generator is defined as to minimize the following Equation 2.2:

$$log(1 - D(G(z))) \tag{2.2}$$

where $D(G(z))$ is the probability of the discriminator estimating a fake instance as real. Since a strong generator can be obtained by the contest with a strong discriminator, the discriminator also aims to be better distinguishing real and fake samples. Therefore,

the objective function of the discriminator is to maximize the following Equation 2.3:

$$E_x[log(D(x))] + E_z[log(1 - D(G(z)))] \tag{2.3}$$

where $D(x)$ is the probability of the discriminator accurately estimating a real instance.

The training process continues until the optimal generator is obtained. At this point, the probability of the discriminator making a mistake is maximum; in other words, it can no longer distinguish real samples from fakes. Figure 2.8 illustrates the overall GAN architecture.

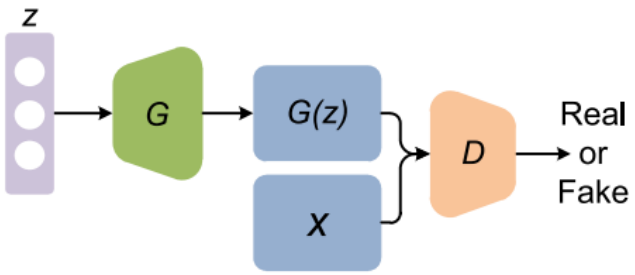


Figure 2.8. The architecture of generative adversarial networks (taken from [31]).

2.5. BERT

BERT, Bidirectional Encoder Representations from Transformers, is a recent representation model that has achieved state-of-the-art performance on several NLP tasks. It is a pretraining language representation where a language model is trained on a large corpus, and later on, it can be used for several tasks by applying a fine-tuning process. Unlike the previous language representation models (e.g., Word2Vec, GloVe), BERT focuses on using deep bidirectional representations of unlabeled text by jointly conditioning on both the left and the right context [32]. Being trained on unlabeled text data brings an opportunity to use an enormous amount of plain text data publicly available on the web.

Pretrained language models can be mainly classified into two categories which are context-free language models and contextual language models. While context-free language models such as Word2Vec and FastText generates a single word embedding for each word regardless of the context, contextual language models generate different word representations depending on the other words in the sentence. BERT is a contextual language model; therefore, word representations contain enriched information depending on the context.

Furthermore, contextual language models can be divided into two categories which are unidirectional and bidirectional models. In unidirectional language models, each word is contextualized using the words standing to its left or right. On the other hand, BERT is a bidirectional language model where words are contextualized, combining both the left and right context. Furthermore, unlike the previous bidirectional language models, BERT uses a masked language model (MLM) where some of the input tokens are randomly masked and predicted based on the context.

3. RELATED WORK

There have been many approaches to extract protein-protein interactions from text sentences. Distant Supervision [33] is one of the standard technique for relation extraction and uses the existing knowledge stored in the knowledge base. The distant supervision assumes that if two entities (proteins in our case) have a relationship in a given database, then all sentences containing these two entities will express that relationship in some way. However, since any individual sentence may carry a wrong indication, the resulting databases are often noisy; in other words, they contain many false positives [34].

Several denoising approaches are introduced to solve the false-positive problem of distant supervision. Multi-instance single-label learning [35] and multi-instance multi-label learning [36] were introduced to tackle the wrong label problem in Distant Supervision. Multiple sentences for the same entity pair are allowed in multi-instance single label learning, but each should have a single label. Multi-instance multi-label learning (MIML), on the other hand, is an extension of multi-instance single-label learning and assumes the same pair of entities may have multiple labels. MIML was later on tested against single-instance learning for the Protein-Protein Interaction task. [37]. According to their experiments on the Salmonella and Human Protein dataset, multi-instance learning greatly outperforms single-instance learning, achieving an overall MCC score of 0.6306 versus 0.5260 for single-instance learning.

Piecewise Convolutional Neural Networks (PCNNs) was introduced with multi-instance learning (MIL) for distantly supervised relation extraction [38]. Distantly supervised samples are treated as multi-instances where some noisy samples are taken into account. In addition, they came up with a piecewise max-pooling layer in CNN architecture to capture structural information between entities to address the wrong label problem of distant supervision. Compared to MIML [36], they improved the recall score by approximately 34% without any loss of precision on the Freebase dataset.

Dependency Parse Trees (DPTs) of the sentences were introduced to analyze paths between two protein entities [11]. They have defined separate cosine and edit distance-based similarity functions to compare the similarity between two DPT of sentences containing protein names. They investigated semi-supervised machine learning methods on top of the dependency features. The performance of supervised learning algorithms, i.e., Support Vector Machines (SVMs) and K-Nearest Neighbor (KNN), were compared with the semi-supervised counterparts of these algorithms, transductive SVMs and harmonic functions, respectively. It was stated that edit distance kernels with Semi-supervised algorithms perform better than supervised alternatives for the PPI extraction task.

All-paths graph (APG) kernel [39] was proposed for Protein-Protein Interaction Extraction. APG considers all possible dependency paths connecting any two vertices and sums the weights of these paths. They assign higher path weights for the shortest paths and lower weights for other paths. As a result, their representation highlights the shortest path without discarding potentially relevant words outside the path. The model was evaluated on five PPI corpora, AIMed, BioInfer, HPRD50, IEPA and LLL. It was stated that APG achieves 56.7% F1 score and 84.8% AUC on the AIMed corpus, which was better compared to the previous kernel studies.

A hybrid kernel-based approach [40] was proposed for the same task. By combining different kernels such as Bag-of-words (BOW) kernel, Subset tree (ST) kernel, and Graph kernel, based on several syntactic parsers such as Dependency parser and Deep parser, they utilized a wide range of information from sentences. Furthermore, they have applied an SVM classifier on top of the resulting kernels and achieved better results than the previous kernel-based approaches on four out of the five PPI corpora, AIMed, BioInfer, HPRD50, IEPA and LLL.

The k-Band Shortest Path Spectrum Kernel (kBSPS) [41] was proposed for extracting relations from biomedical text. It is claimed that the shortest path is not enough for relation extraction, so they have included dependency graph nodes and

dependencies within a distance of k from the shortest path. They have used the SVM model as a classifier and achieved comparable results with the APG kernel.

The walk-weighted subsequence kernel was proposed for the Protein-Protein Interaction extraction task [42]. They have assigned different weights to common substrings of two shortest path strings. They have handled lexical subgraphs, syntactic subgraphs, e-walk and v-walk, and dependencies with varying weights. They compared the results with other systems, and their model achieved a higher score than the previous stories on the small datasets. It is reported that their model has obtained an 82.1% F1-score on the LLL corpus.

Neighborhood hash graph kernel was introduced for the same task [43]. Unlike previous graph kernel-based approaches, the neighborhood hash graph kernel makes use of complete dependency graphs to represent the sentence structures. A mapping function was implemented to map each node label of the dependency graph into a fixed-length binary array. Then, bit labels of nodes were updated by order-independent logical operations on the bit labels of the neighbor dependency nodes. They investigated the performance of the neighborhood hash graph kernel on five benchmarking PPI corpora and obtained comparable performance to the previous state-of-the-art PPI extraction systems.

Extended Dependency Graph (EDG) was proposed for biomedical relation extraction [44]. The EDG vertices were constructed with text, part-of-speech tags and word lemmas. Entities and phrases that span multiple words in a sentence were stored into a single vertex. Additional syntactic relations such as coreference, appositive, is-A, member-collection, and part-whole relations were also included in EDGs. They evaluated the performance of EDG with two different kernels, which are edit distance kernel and all-paths graph kernel, by applying them on five different PPI Corpora, AIMed, BioInfer, HPRD50, IEPA and LLL. They showed that EDG with edit distance kernels achieved up to 10% F1 score improvement compared to dependency graphs with the same kernel on PPI corpora.

PIPE [45], the PPI Pattern Extraction module was proposed for BioCreativve challenge [46]. They proposed an interaction pattern tree (IPT) kernel that integrates convolution tree kernel (CTK) to extract PPIs. The IPT kernel captures the syntactic and semantic information of sentences. CTK integrates IPT with SVMs. They also evaluated their method on Five Benchmarking PPI Corpora. Their approach outperformed several kernel methods and achieved comparable results with multi-kernel-based methods.

Distributed smoothed tree kernel (DSTK) was introduced for Protein-Protein Interaction extraction from biomedical literature [47]. DSTK utilizes both syntactic and semantic vectors to overcome the shortcomings of information loss from single kernel approaches. They used several word, and word distance features such as protein names, interaction keywords, surrounding words, words between two protein entities for their feature-based kernel. For their distributed smooth tree kernel, they generated distributed trees with distributional semantic vectors that capture the semantic information of sentences. They evaluated their method on Five Benchmarking PPI corpora and achieved better F1-scores (71.01% on AIMed, 76.29% on BioInfer, 80.00% on HPRD50, 80.23% on IEPA, and 89.20% on LLL corpus) compared to other previous state-of-the-art systems.

Since deep learning techniques have achieved great success in many domains through deep hierarchical feature construction and capturing long-range dependencies in data, they are also being examined for a wide range of NLP tasks, including Protein-Protein Interaction Extraction. A feed-forward neural network architecture that can learn complex and abstract features of data was proposed for PPI extraction [48]. First, the parameters of the Deep Neural Network (DNN) were initialized by training auto-encoders. Then, the gradient descent algorithm with backpropagation was applied to train the model. Finally, the performance of the system was evaluated on two PPI corpora, AIMed and BioInfer, and obtained 0.8% and 1.3% F1-score improvement compared to the APG kernel method.

A deep convolutional neural network (DCNN) was introduced with various feature embeddings to extract PPIs from sentences [49]. First, they concatenated existing word embeddings with position embeddings which are relative distances from target protein entities. Then, the obtained features were passed into a CNN model for training. Finally, they evaluated their work on AIMed corpus. Unlike previous studies, they used macro-averaged F1 scores as an evaluation metric, and they obtained an 85.2% score.

A Shortest Dependency Path-Based Convolutional Neural Network (sdpCNN) model was proposed for Protein-Protein Relation Extraction [50]. They used only the shortest dependency paths (SDPs) as a feature. It was claimed that manual feature selection and feature combination approaches bring bias to representations. CNN, on the other hand, can automatically learn these features from SDPs. They evaluated their model on AIMed and BioInfer corpora and achieved 66.0% and 75.2% F1-score, respectively.

A multichannel Convolutional Neural Network (MCCNN) model was proposed for biological relation extraction [51]. The model has CNN architecture and relies on multichannel word embeddings that enable the fusion of multiple word embeddings. They applied their model for Drug-Drug Interaction (DDI) and Protein-Protein Interaction (PPI) tasks. The model was tested on the DDIExtraction dataset for the DDI task and outperformed SVM based model score by 3.2%. For the PPI task, AIMed [52], and BioInfer [53] datasets, which have been extracted from the DIP database, were used, and the model outperformed SVM based model score by 2.7% and 5.6%, respectively on two datasets.

A multichannel dependency-based convolutional neural network model (McDepCNN) was proposed for extracting protein-protein interactions from biomedical literature [54]. They applied two channels; one for embedding vectors, the second for the embedding vector of the head of the corresponding word. They claimed the different channels provide richer information to the model. The model was evaluated on

two benchmarking corpora, AIMed [52], and BioInfer [53], and it outperformed previous deep learning models and single feature-based or kernel-based models significantly. Moreover, they showed that the McDepCNN model is also successful over different corpora and can learn long-distance features in sentences with the help of multichannel embeddings.

A recurrent neural network (RNN) was proposed to identify PPIs [55]. They have implemented a bi-directional RNN with Long Short-Term Memory (LSTM) model to extract protein interactions. Their system consists of three layers: (i) an embedding layer that transforms words into embeddings, (ii) a recurrent layer that is constructed using LSTM cells, and (iii) a fully connected layer that performs the classification task. They evaluated their work on two PPI corpora: AIMed and BioInfer. Their model outperformed the existing models on both corpora by achieving 76.9% and 87.2% F1-scores, respectively.

A hybrid model that combined RNN and CNN models was introduced for this task [56]. They generated the shortest dependency paths (SDPs) based on the dependency graph of sentences. While their RNN model captures sentence features from sentence sequences, the CNN model utilizes SDPs. On top of these, they implemented a multilayer perceptron (MLP) with a softmax output layer for classification. Finally, they evaluated their proposed model on a (Drug-Drug Interaction) DDI corpus and Five Benchmark PPI Corpora; AIMed, BioInfer, HPRD50, IEPA, and LLL. They obtained the best F1-score on AIMed and IEPA corpora compared to kernel-based approaches.

Tree LSTM (tLSTM) model with structured attention architecture was proposed for identifying PPIs [57]. The dependency parse trees were traversed through tree LSTM model in order to learn structural information of sentences. They have also combined tLSTM results with a structured attention-based model results. The model has been evaluated on Five Benchmark PPI corpora; AIMed, BioInfer, HPRD50, IEPA, and LLL. The obtained macro-averaged F1-scores are 81.6% 89.1% 78.5% 81.3%, and 84.2%, respectively.

A deep residual convolutional neural network was introduced for PPI extraction [58]. They have added a residual connection between convolutional modules to make the model strong for classification. The model consists of three layers: (i) an embedding layer for word representation, (ii) the convolutional layer that consists of several convolutional modules with a residual connection, and (iii) a classifier layer that gives prediction results. They achieved comparable results with previous RNN based approaches on Five Benchmark PPI Corpora; AIMed, BioInfer, HPRD50, IEPA, and LLL.

4. DATA SET

In this thesis, we used five benchmark corpora, AIMed [52], BioInfer [53], IEPA [59], HPRD50 [60] and LLL [61] that have been converted into a unified format by Pyysalo [62]. These five corpora have been widely used in different PPI extraction research and have become common benchmarking datasets. It also provides predefined train and test splits to accomplish the standardization.

4.1. AIMed

AIMed is a corpus extracted from 200 PubMed abstracts identified by the Database of Interacting Proteins (DIP). For the negative samples, different 30 abstracts that contain no PPIs are selected. It contains manually annotated protein-protein interactions and widely used for PPI extraction method comparison.

4.2. BioInfer

BioInfer is a corpus consisting of sentences from PubMed abstracts that contain interaction protein pairs identified according to the Database of Interacting Proteins. The sentences are manually annotated by the corpus creators identifying entity pairs, protein types, and interaction words containing complex and negative interactions and static relations. It contains nearly 7500 sentences, with 30percentage indicating positive interactions.

4.3. The HPRD50

The HPRD50 is a protein-protein interaction corpus intentionally created as an evaluation dataset for RelEx systems. The sentences are extracted from 50 abstracts identified by the Human Protein Reference Database (HPRD). First, the protein and gene names are automatically annotated by software. Then, the interactions of these

entities are manually annotated. Each entity pair also contains the interaction certainties.

4.4. The IEPA

The IEPA is a corpus created from the sentences of PubMed abstracts, containing ten pairs of co-occurring chemicals that were mainly proteins. The corpus is intentionally created for biomedical research topics. The entity pairs and interactions are manually annotated concerning the direct and indirect effects of the entities on each other.

4.5. The LLL

The LLL corpus is a shared dataset that contains gene interactions of Bacillus subtilis, created for the Learning Language in Logic 2005 (LLL05) challenge. It contains manually annotated gene-protein interactions stating types of interactions such as explicit interaction, binding protein-gene interaction, or membership in a regulon family.

4.6. Characterization of the Corpora

All five corpora consist of sentences extracted from biomedical literature, and these sentences contain several protein and gene entities. In addition to entity annotations, interacting entity pairs are also specified in each sentence. Even though some corpora contains additional properties such as interaction direction, interaction certainty, binding of words that specify the interaction and interaction complex where an annotation includes more than two entities, these do not exist in each corpus as seen in Table 4.1 [63]. Therefore, the unified corpora only reveal the shared properties where entity pairs contain simple interactions, i.e., undirected, untyped interactions with no complex structure and no interaction certainty.

Table 4.1. The characteristics of the five corpora.

	AIMed	**BioInfer**	**HPRD50**	**IEPA**	**LLL**
Corpus Size	1955	1100	145	486	77
Entity Types	Human Protein/Gene	Protein/ Gene/RNA	Human Protein/Gene	Chemicals	Protein /Gene
PPI direction	no	yes	no	yes	yes
PPI certainty	no	no	yes	no	no
PPI binding	no	yes	no	yes	no
PPI complex	no	yes	no	no	no

Each corpus contains a different number of sentences. AIMed and BioInfer are the largest corpora among them having 1955 and 1100 sentences, respectively. LLL, on the other hand, contains 77 sentences. The average sentence length is nearly the same in each corpus, except HPRD50 has a slightly smaller sentence length.

Furthermore, each sentence consists of a different number of entities. BioInfer is the most complex corpus with four entities for each sentence on average and a larger ratio of multiple entities to two entities. It is the wealthiest source that contains the largest number of entity pairs, 9666, while LLL is the smallest, containing 300 entity pairs. Although LLL is the smallest corpora, it is one of the most complexes after BioInfer, with 3.1 entities for each sentence on average. IEPA is one of the most simple corpora with the lowest ratio of multiple entities to two entities for each sentence. It means that the sentences contain single Protein-Protein interactions in general rather than having interaction complexes.

Moreover, not each entity pair in a sentence reflects Protein-Protein Interaction. The entity pairs that contain PPI are annotated as positive samples, and others annotated as negative samples. Hence, each corpus yields a different positive/negative sample distribution as seen in Table 4.2.

Table 4.2. The statistics of the five corpora.

Corpus	AIMed	BioInfer	HPRD50	IEPA	LLL
Sentence Count	1955	1100	145	486	77
Min. Sent. Length	4	6	6	6	9
Max. Sent. Length	127	95	57	90	72
Avg. Sent. Length	31	35	27	32	34
Avg. Entity Count	2.2	4	2.8	2.3	3.1
Max. Entity Count	18	21	7	6	9
Sent. w/ Two Entities	477	273	69	375	35
Sent. w/ Mult. Entities	685	817	76	111	42
Sent. w/ Single PPI	316	268	49	281	25
Sent. w/ Mult. PPIs	254	516	41	23	42
Max. PPI count	12	38	6	4	7
Pos. Pair	1000	2534	335	163	164
Neg. Pair	4834	7132	482	270	166
Total Pair	5834	9666	817	433	300

4.7. Unified Dataset Format

The five unified PPI corpora are provided in a simple XML format. Data from each corpus is divided into two separate files, one for training and the other for testing. The correct labels for the test data are stored in another file. The directory contains three separate folders named train, test, and answers. Train and test folders consist of five XML files corresponding to AIMed, BioInfer, HPRD50, IEPA, and LLL datasets. In the test XML files, there is no interaction information exists. This information can be found in five TXT files located in the answers folder. Two samples from train and test datasets can be seen in Figure 4.1 and Figure 4.2 respectively. These figures also illustrate the XML format of the five unified datasets. A part of an answer key that is in TXT format can be seen in Figure 4.3.

```
<corpus cvsVersion="1.2" source="AIMed">
    <document id="AIMed.d39" origId="10508860" set="train">
        <sentence charOffset="0-87" id="AIMed.d39.s334" seqId="s334" text="Presenilin 1 suppresses
        the function of c-Jun homodimers via interaction with QM/Jif-1.">
          <entity charOffset="0-12" id="AIMed.d39.s334.e0" seqId="e711" text="Presenilin 1" type="protein" />
          <entity charOffset="40-45" id="AIMed.d39.s334.e1" seqId="e714" text="c-Jun" type="protein" />
          <entity charOffset="78-80" id="AIMed.d39.s334.e2" seqId="e717" text="QM" type="protein" />
          <entity charOffset="81-86" id="AIMed.d39.s334.e3" seqId="e719" text="Jif-1" type="protein" />
          <interaction e1="AIMed.d39.s334.e0" e2="AIMed.d39.s334.e2" id="AIMed.d39.s334.i0" type="PPI" />
          <interaction e1="AIMed.d39.s334.e0" e2="AIMed.d39.s334.e3" id="AIMed.d39.s334.i1" type="PPI" />
        </sentence>
        ...
    </document>
    ...
</corpus>
```

Figure 4.1. A train sample having two Protein-Protein Interactions from Five Unified PPI Corpora.

```
<corpus cvsVersion="1.2" source="AIMed">
    <document id="AIMed.d3" origId="11781834" set="test">
        <sentence charOffset="727-877" id="AIMed.d3.s30" seqId="s30" text="Temporally following this growth arrest,
        the cells develop a senescence morphology and express senescence-associated beta-galactosidase (SA-beta-gal).">
          <entity charOffset="117-135" id="AIMed.d3.s30.e0" seqId="e46" text="beta-galactosidase" type="protein" />
          <entity charOffset="95-135" id="AIMed.d3.s30.e1" seqId="e47" text="senescence-associated beta-galactosidase" type="protein" />
          <entity charOffset="137-148" id="AIMed.d3.s30.e2" seqId="e48" text="SA-beta-gal" type="protein" />
          <pair e1="AIMed.d3.s30.e0" e2="AIMed.d3.s30.e1" id="AIMed.d3.s30.p0"/>
          <pair e1="AIMed.d3.s30.e0" e2="AIMed.d3.s30.e2" id="AIMed.d3.s30.p1"/>
          <pair e1="AIMed.d3.s30.e1" e2="AIMed.d3.s30.e2" id="AIMed.d3.s30.p2"/>
        </sentence>
        ...
    </document>
    ...
</corpus>
```

Figure 4.2. A test sample from Five Unified PPI Corpora.

```
AIMed.d3.s30.p0 False
AIMed.d3.s30.p1 False
AIMed.d3.s30.p2 False
AIMed.d3.s31.p0 False
AIMed.d3.s31.p1 False
AIMed.d3.s31.p2 False
AIMed.d3.s31.p3 False
AIMed.d3.s31.p4 False
AIMed.d3.s31.p5 False
AIMed.d3.s31.p6 False
AIMed.d3.s31.p7 False
AIMed.d3.s31.p8 False
AIMed.d3.s31.p9 False
```

Figure 4.3. A part of an answer key, in TXT format.

The format contains the following properties:

- The *sentence* element contains a raw sentence.
- Entities are identified through the *entity* element, and it contains character offset attribute that locates entity position.
- The *interaction* element is only available in the training dataset. It specifies interaction details where $e1$ and $e2$ attributes are entity pairs and *type* attribute describes interaction type.
- The *pair* element is only available in the test dataset. It specifies entity pairs with $e1$ and $e2$ attributes. The information regarding whether the entity pairs indicate Protein-Protein Interaction can not be seen in the test file; instead, they can be found in the answer key.
- The answer key consists of each entity pair id given in the test dataset and the corresponding interaction information, which indicates whether the entity pair contains PPI or not.

For instance, the training sample *"Presenilin 1 suppresses the function of c-Jun homodimers via interaction with QM/Jif-1."* seen in Figure 4.1 contains four protein entities which are "Presenilin 1", "c-Jun", "QM" and "Jif-1", and two interacting protein pairs which are "Presenilin 1" - "QM" and "Presenilin 1" - "Jif-1".

5. METHODOLOGY

5.1. Preprocessing

The dataset described in Chapter 4 is in XML format, and it should go through some preprocessing steps before we can move on to the further process. Since XML data format contains plentiful information such as XML tags, elements, attributes, characters, and we simply require plain sentences, the XML format is quite noisy for our study. Furthermore, our dataset contains many biomedical terms, such as protein, gene and chemical names. However, protein names are just an entity for us, and we are only interested in whether the two protein entities interact with each other rather than which exact protein names interact. Fortunately, each biomedical entity is annotated in Five Unified PPI Corpora with their character offsets, and we replace their exact names with common indicators: PROTEIN1 and PROTEIN2 for protein pairs that are investigated for protein interaction and PROTEIN keywords for other proteins.

In short, our aim in the preprocessing step is to eliminate irrelevant and redundant information and noisy and unreliable data. We have applied the following data preprocessing steps to obtain higher quality data:

- Firstly, we parse XML files so that each XML element and attribute is accessible as an XML element tree.
- At the top of the XML element tree, we have document trees, consisting of sentence trees. We process each document tree one by one to process sentence trees.
- Training and test sets have different sentence tree structures; therefore, they are processed differently.
  (i) In the training set, a sentence tree contains a raw sentence, a list of all entities in this sentence, and a list of entity pairs that indicate protein-protein interactions in the same sentence. The training XML format does not con-

tain non-interacting entity pairs; however, this information is also important for our model since they are considered as negative samples. Therefore we generate all entity pair combinations for each sentence, and then we label the interacting pairs in the given sentence tree as positive samples and the non-interacting pairs as negative samples. In a sentence with n protein entities ($n \geq 2$), $\binom{n}{2}$ entity pairs are generated. Moreover, with the help of entity character offsets, we replace entity pair names with PROTEIN1 and PROTEIN2 keywords and entity names other than the entity pair with PROTEIN keyword.

(ii) In the test set, sentence trees contain the raw sentence, all entities, and all entity pairs. Same as the training set, we replace the entity pair names with PROTEIN1 and PROTEIN2 keywords and entity names other than the entity pair with PROTEIN keyword. This time we fetch the interaction label from the answer key file.

- After XML parsing, we obtain clean sentences for each entity pair and the corresponding interaction annotation. Then, we apply further basic preprocessing steps to obtained final sentences:
  (i) Punctuation marks are removed.
  (ii) Digit only strings are removed.
  (iii) Blank spaces are removed.
- Finally, we export the resulting sentences and annotations in a Comma Separated Values (CSV) file.

As an example, consider the sample training sentence given in Figure 4.1. The sentence *"Presenilin 1 suppresses the function of c-Jun homodimers via interaction with QM/Jif-1."* contains four protein entities which are *"Presenilin 1"*, *"c-Jun"*, *"QM"* and *"Jif-1"*, and two interacting protein pairs which are "Presenilin 1" - "QM" and "Presenilin 1" - "Jif-1". The sentence contains $\binom{n=4}{2} = 6$ entity pair combinations, therefore we evaluate the same sentence for 6 times for each entity pairs. First, we replace corresponding protein names with PROTEIN1 and PROTEN2 keywords and other proteins as PROTEIN. Then we annotate interacting protein pairs that have

been given in training data as True PPI samples and other protein pairs as False PPI samples. Table 5.1 summarizes the resulting preprocessed sentences for this example.

Table 5.1. A sample preprocessed sentence.

Protein Pair	**PPI**	**Preprocessed Sentence**
Presenilin 1 / c-Jun	False	PROTEIN1 suppresses the function of PROTEIN2 homodimers via interaction with PROTEIN PROTEIN
Presenilin 1 / QM	True	PROTEIN1 suppresses the function of PROTEIN homodimers via interaction with PROTEIN2 PROTEIN
Presenilin 1 / Jif-1	True	PROTEIN1 suppresses the function of PROTEIN homodimers via interaction with PROTEIN PROTEIN2
c-Jun / QM	False	PROTEIN suppresses the function of PROTEIN1 homodimers via interaction with PROTEIN2 PROTEIN
c-Jun / Jif-1	False	PROTEIN suppresses the function of PROTEIN1 homodimers via interaction with PROTEIN PROTEIN2
QM / Jif-1	False	PROTEIN suppresses the function of PROTEIN homodimers via interaction with PROTEIN1 PROTEIN2

5.2. Overall System Architecture

In this chapter, we introduce the overall system architecture. Figure 5.1 depicts the system overview to provide a better understanding for the following chapters. We implement a two-staged hybrid BERT-GAN system for the PPI extraction task. The system works in a cascaded way, or simply put one after another. Our two stages have different purposes:

- The first stage is implemented to perform predictions. It predicts whether a given sentence contains PPIs or not. Therefore it perform a binary relation extraction.
- The second stage is implemented to increase the accuracy of the first stage. The prediction outputs of the first stage pass through the second stage. By an adver-

sarial process, false positive predictions are eliminated to increase the performance in this stage.

To accomplish these two tasks, we implemented two separate learning models. First, we fine-tune BioBERT, a pretrained language model for the biomedical domain, to learn PPI information for the first stage. Then, for the second stage, we implemented a Generative Adversarial Network that consists of two Convolutional Neural Network models to eliminate false positives with the help of the adversarial process. Both stages use the Five Benchmarking corpora explained in Chapter 4. Furthermore, we apply the same preprocessing steps explained in Section 5.1.

5.3. BioBERT Model

Bidirectional Encoder Representations from Transformers for Biomedical Text Mining (BioBERT) is a biomedical language representation model designed for Biomedical Natural Language Processing (BioNLP) tasks. It has the same structure as BERT that explained in section 2.5 in details. However, since BERT is pre-trained on generic corpora like Wikipedia and BooksCorpus and is not developed for the biomedical domain particularly, estimating its performance on biomedical text is tricky. BioBERT, on the other hand, is pre-trained on large biological corpora obtained from PubMed and PMC articles, alongside general corpora (Wikipedia and BooksCorpus). As a consequence, BioBERT also covers terms that are not found in biomedical corpora. The vocabulary used for pre-training of BioBERT is listed in Table 5.2. It remarkably outperforms BERT on three main BioNLP tasks, which are biomedical named entity recognition (0.62% F1 score improvement), biomedical relation extraction (2.80% F1 score improvement), and biomedical question answering (12.24% MRR improvement) [12]. Since its tremendous success, we decided to adopt the BioBERT model to perform the PPI extraction task. We accomplish this by fine-tuning BioBERT on Five Benchmarking PPI corpora. Fine-tuned BioBERT model is designed for our first stage to predict whether given sentences contain PPIs. The detailed information regarding the fine-tuning process and the implementation is given in the following section.

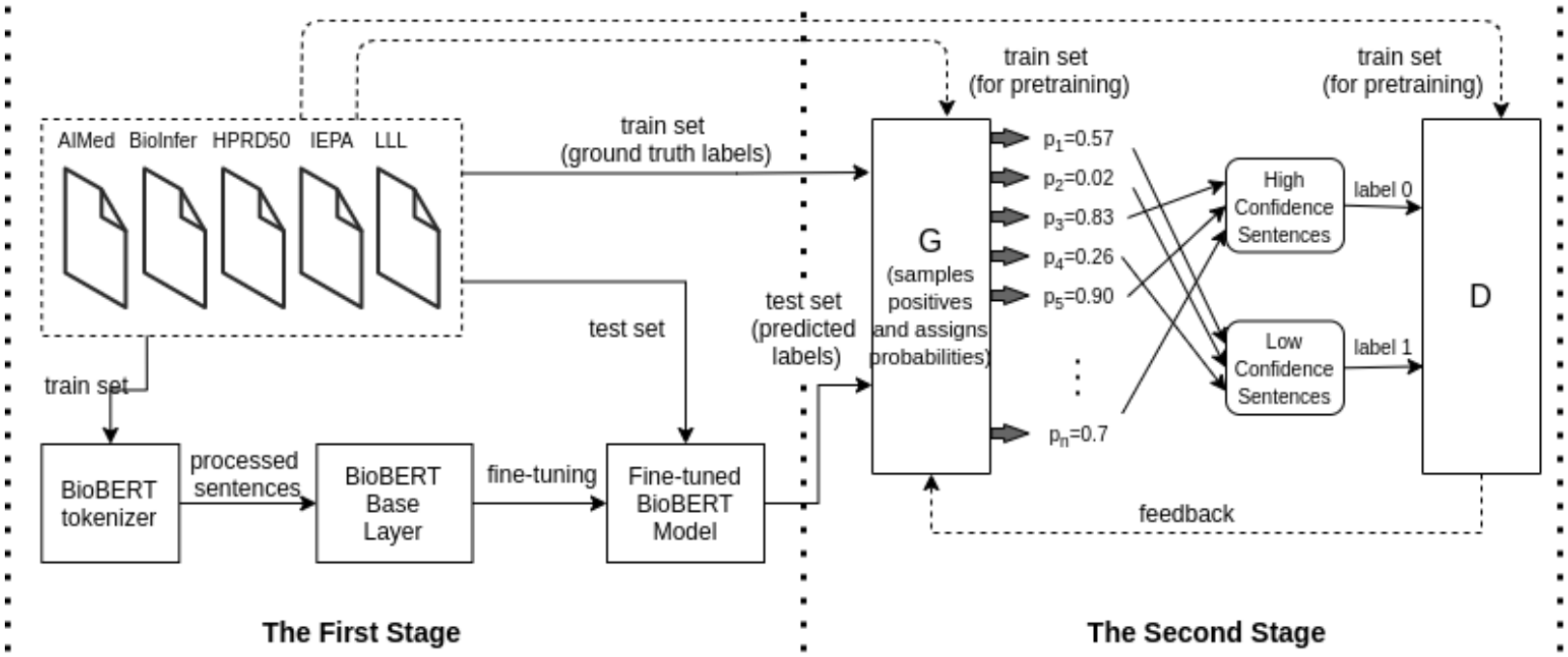


Figure 5.1. An overview of our two-staged hybrid BERT - GAN pipeline.

Table 5.2. List of corpora and number of their words used BioBERT.

Corpus	**Number of words**	**Domain**
English Wikipedia	2.5B	General
BooksCorpus	0.8B	General
PubMed Abstracts	4.5B	Biomedical
PMC Full-text articles	13.5B	Biomedical

5.3.1. Fine-Tuning the BioBERT

We fine-tune the pre-trained BioBERT(v1.1+PubMed) model using the training sets of Five Benchmarking PPI Corpora after applying preprocessing steps explained Section 5.1 and we adapt it for our PPI extraction task. We utilized the sentence classifier of the BERT, which uses a [CLS] token for the classification of relations and a [SEP] token for the fixed-length sentences. Our preprocessed sentences are encoded using the pre-trained BioBERT tokenizer that applies WordPiece tokenization.

Huggingface Transformers library [64] provides many general-purpose architectures and pre-trained NLP models including BERT and BioBERT models. Both pre-trained BioBERT model and BioBERT tokenizer are obtained from Transformers library which is implemented with Python programming language. BioBERT is implemented with PyTorch deep learning library, therefore our fine-tuning implementation also utilizes Pytorch.

As an initial step, we encode our preprocessed sentences with the help of pre-trained BioBERT tokenizer. BioBERT tokenizer adds the [CLS] token to the beginning of each sentence and [SEP] token to the end of each sentence and then map each token to their ids. Since sentences in our dataset are of varying length, we will use padding to make all sentences have the same length. We define a maximum sequence length to pad the sentences. The maximum sequence length is defined by looking at the distribution of the sentence length in our dataset as seen in Figure 5.2.

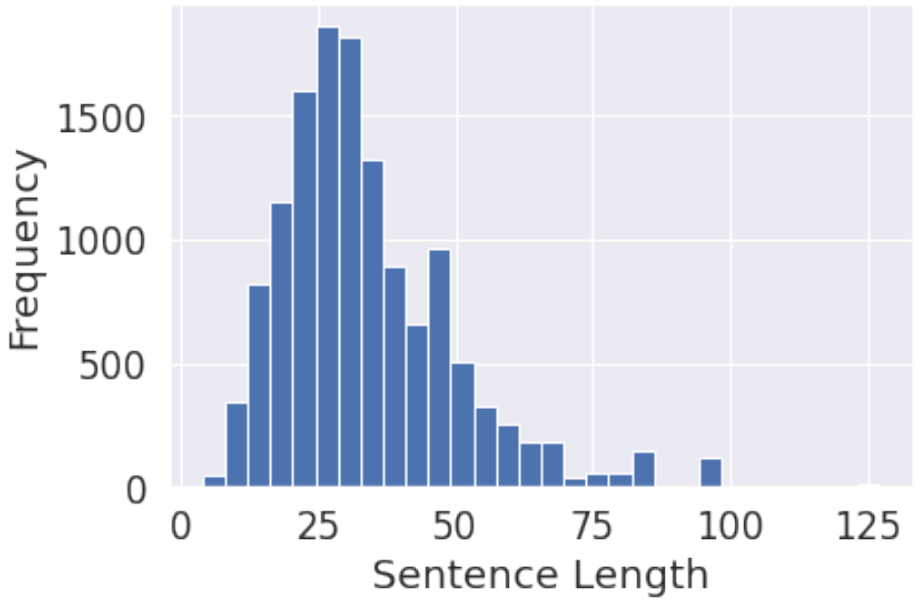


Figure 5.2. The distribution of sentence length in our dataset.

As seen in Figure 5.2 most of the sentences have a length of 64 words or less, whereas the maximum sentence length is 127. The reason for defining a shorter maximum sentence length is that padding operation adds padding tokens to the end of each sentence until it reaches the desired sentence length, therefore the resulting sentences may contain redundant information and further it decreases the training performance.

Moreover, an attention mask is defined for each sentence. The attention mask is a filter that indicates which tokens are padding and which tokens are sentence words. This tells the BioBERT to not process PAD tokens for its interpretation of the sentence. Later on, both encoded sentences and attention masks pass into the pre-trained BioBERT model for fine-tuning task. HuggingFace's BertForSequenceClassification model is used with our dataset to obtain a classifier which predicts whether sentences contain protein interactions. A single output layer is added on top of the BERT Model to obtain BertForSequenceClassification Model. As we feed our PPI data during fine-tuning, the whole pre-trained BioBERT model with an additional untrained output layer will be trained on Protein-Protein Interaction task.

We select AdamW optimizer [65] as an optimization algorithm for fine-tuning, which is one of the most commonly used optimizers for deep learning models. The AdamW optimizer has several hyperparameters such as learning rate, epsilon and weight decay. Epsilon is a very small number to prevent any division by zero in the training phase. Learning rate and weight decay control how weights will be updated in each epoch. The other hyperparameters for our model are epoch number and batch size. We fine-tune our BioBERT model with the given hyperparameter settings in Table 5.3. These values are obtained by several experiments and will be explained in Chapter 6.

Table 5.3. Hyperparameter selection for fine-tuning BioBERT model.

HyperParameter	**Value**
Max Sentence Length	64
Optimizer	AdamW
Batch Size	16
Learning rate	2e-5
Epsilon	1e-8
Epochs	4
Weight decay	0.01

5.3.2. The first-stage / BioBERT PPI Prediction

In the first stage, we predict PPI sentences using fine-tuned BioBERT model. After we fine-tune the BioBERT model with our Five Unified PPI corpora, we obtain a classifier designed for a PPI extraction task. For correct evaluation, we obtain predictions with various experimental settings:

- We train five different BioBERT models by using training sets of five PPI corpora. Then we obtain predictions on the corresponding test sets.
- We obtain cross-corpus predictions by using these models on cross test sets.

- We combine the training sets of five corpora and train a single BioBERT model on this single data source. Then we obtain predictions on each test set.

All these predictions then will be used in our second stage. Since our first stage itself is capable of predicting PPIs, we also evaluate its performance separately. All experiments and their results will be explained in Chapter 6.

5.4. GAN Model

Inspired by the work by [66], we adopt a GAN model to eliminate False-Positive predictions obtained from our first stage. We use training sentences of five PPI corpora with ground truth labels and test sentences of five PPI corpora with predicted labels annotated by our BioBERT model together during the adversarial learning. By combining these two, we obtain a new dataset that contains some possibly false-positive labels.

Unlike the generator used in the Computer Vision field that generates fake images from a random input space, our generator samples positive sentences from our dataset and assigns some probabilities to each sample. On the other hand, our discriminator has the same responsibility as the discriminator in the Computer Vision field; it is simply a binary classifier and tries to distinguish whether given sentences contain protein interactions or not. The generator updates its network with the feed-backs of the discriminator gradually. Since our dataset contains some false-positive samples, we believe that the generator will assign lower probabilities to false positives at the end of the adversarial training by capturing the correct data distribution with the help of our discriminator.

We label high-confidence positive samples determined by G as negatives and low-confidence positive samples determined by G as positives in order to fool the discriminator. During the adversarial learning phase, the generator's objective is to maximize the probability of being positive for true samples, while the objective of the discrim-

inator is to minimize this probability. In the end, we define a probability threshold and change the label of sentences below this threshold but annotated as positive by BioBERT to negative.

5.4.1. PreTraining GAN

In order to obtain better initial parameters, both the generator and discriminator are pre-trained. As a positive pre-training dataset, the positive instances of our training dataset coming from the Five Benchmarking Corpora (P) are selected for our both networks. As a negative pre-training dataset, on the contrary, we split our negative instances into two separate negative datasets, ND for the discriminator, and NG for the generator, for correct evaluation. At the end of the adversarial learning process, we wish for a strong generator that fools the discriminator. Since the strong generator can be obtained by competition with a robust discriminator, we pre-train the discriminator until it reaches above 90% accuracy. The pre-training strategy of the generator is similar to the discriminator except that it uses a separate negative dataset NG, and it overfits on the positive dataset, P. The reason for overfitting in the pre-training phase is that we want the generator to assign high probabilities for false-positive instances at the beginning wrongly. Then, by applying adversarial learning, the generator will learn false-positive samples with the help of feedback from the discriminator.

5.4.2. Training Generative Network

Convolutional Neural Networks have recently achieved great success in NLP-related tasks, especially for sentence-level classification. For the protein-protein interaction extraction, we have sentences and entity pairs as input. Both our generative and discriminator networks perform a similar task with sentence-level classification; therefore, CNN architectures are very well fitted for our research. We have used the typical setting as [67], proposed for their CNN architecture. We combine both word embeddings and position embeddings to represent our sentences with vectors.

5.4.2.1. Word Embeddings. Different from this work [67], we use another pre-trained word embedding, BioWordVec [68], which is designed for the biomedical domain. BioWordVec has similar architecture to Word2Vec. However, it is trained on 27,599,238 PubMed articles and 28,436 MeSH term graphs; therefore, favourable for our PPI dataset.

5.4.2.2. Position Embeddings. We also use position embeddings along with word embeddings. Relative distances from target protein entities are mapped into a vector of dimension d_{pos}. Since we have two target proteins, we obtain two different position vectors. Relative distances of each word to target proteins are represented in these two vectors, and the resulting position embedding is obtained by concatenation. While the words staying on the other target protein direction are represented with positive distances, the words staying on the opposite direction are represented with negative distances.

5.4.2.3. CNN Architecture. Our CNN architecture consists of three basic components, (i) Embedding layers, for word and position representations, (ii) a convolutional layer that performs convolution operation and learns features, and (iii) an output layer, applies a linear transformation and returns a two-dimensional vector representing the probability of being positive and negative interaction. Our CNN architecture has some hyperparameters such as window size and kernel size for convolutional layer, d_{pos} for position embedding dimension, and learning rate for Adam optimizer. We use the same hyperparameter settings except the learning rate for both generator and discriminator networks. The optimal hyperparameter values are obtained by several experiments and given in Table 5.4.

In Computer Vision-related tasks, the generator produces new images from low dimensional input noise. However, generating meaningful sentences from a generator is a more complicated task in NLP. Therefore, we adopt the generator not to generate new sentences instead to pick up random positive sentences that indicate protein inter-

Table 5.4. Hyperparameter settings of the generator and the discriminator.

HyperParameter	**Value**
Optimizer	Adam
d_{pos}	64
Window Size	3
Kernel Size	100
Learning rate, G	1e-6
Learning rate, D	1e-5
Epochs	10

actions from the training dataset and assign probabilities to each of them. For an input sentence s_j , the generator assigns a probability of being positive protein interaction $pG(s_j)$. After, the same sentence will be evaluated by the discriminator. In this time, the discriminator will assign some probability of being a positive interaction $pD(s_j)$ to the same sentence. Since the selected sentences are positive interactions from the dataset, they consist of high-confidence sentences and regarded as true positives by the generator. Therefore, the generator aims to maximize the following equation:

$$L_G = \sum_{s_j \in T} \log p_D(s_j)\,. \tag{5.1}$$

5.4.3. Training Discriminator Network

The neural network architecture selection of the discriminator is the same as the generator. We have used the CNN model for discriminator with the same configurations as the generator. The hyperparameter selection of the discriminator is given in Table 5.4. The discriminator gets a sample subset evaluated by the generator, and then it assigns some probabilities to each sample. Even the generator regards these sentences as true positives; the discriminator treats them as negative samples. The

objective of the discriminator is to minimize the cross-entropy loss function:

$$L_D = -(\sum_{s_j \in (B_i - T)} \log p_D(s_j) + \sum_{s_j \in T} \log(1 - p_D(s_j))) \,. \tag{5.2}$$

5.4.4. The Second Stage / Adversarial Training

In the training phase, we use positive samples (P) from our training dataset. We split $P = s_1, s_2, ..., s_j, ...$ into the N bags $B = B_1, B_2, ..., B_N$ so that we can utilize our training process by gathering more feedback from the discriminator. In each epoch, we traverse all positive dataset P once. The generator network assigns a probability to each sentence in a bag B_i, $p_G(s_j)_{j=1,...,|B_j|}$. Then we select the high-confidence sentences among them, and they are regarded as negatives by the discriminator. The low-confidence sentences are regarded as true samples by the discriminator. While the generator tries to maximize the probabilities of being positive, the discriminator tries to minimize the cross-entropy loss function.

6. EXPERIMENTS AND RESULTS

In this chapter, we provide different experimental settings and evaluate the performance of our system by comparing it with other previous studies. The chapter starts with the definition of the evaluation metrics and then provides different experimental results. Since our first stage is capable of extracting Protein-Protein Interactions by itself, we report its performance separately as well. Moreover, in order to claim the success of the second stage, we also report Significance Test results for this stage.

6.1. Evaluation Metrics

The F1-score is the most commonly used performance evaluation metric for the Protein-Protein Interaction Extraction task. It is the combination of precision and recall metrics; therefore, it evaluates the system's performance more precisely. The precision, recall and F1-score metrics are defined as follows:

$$Precision = \frac{TP}{TP + FP} \tag{6.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{6.2}$$

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{6.3}$$

where,

- TP is True-Positive, which represents the number of sentences that are correctly predicted as positive interactions by our model.
- FP is False-Positive, which represents the number of sentences that are wrongly predicted as positive interactions by our model.

- TN is True-Negative, which represents the number of sentences that are correctly predicted as non-interacting by our model.
- FN is False-Negative, which represents the number of sentences that are wrongly predicted as non-interacting by our model.

Then, more specifically,

- The precision score attempts to answer the question of what proportion of positive predictions were actually correct.
- The recall score attempts to answer the question of what proportion of actual positives were predicted correctly.

Moreover, since some related studies provide macro-averaged F1 scores for correct evaluation on imbalanced datasets, we also calculate macro-averaged F1 scores in each experiment for the correct comparison. Macro-averaged F1-score is the average F1-score on positive instances and negative instances, and it is widely used for imbalanced data cases. It assigns a higher weight to the minority class, interacting proteins in our case, and hence serves as a proper evaluation metric for our task. The macro-averaged F1-score is calculated as follows:

$$\text{macro-averaged-F1-score } = \frac{1}{N} \sum_{i=0}^{N} \text{F1-score }_i \,. \tag{6.4}$$

6.2. McNemar's Test

McNemar's test is a non-parametric statistical hypothesis test for paired comparisons [69]. It is also referred to as the "within-subjects chi-squared test". McNemar's test operates upon a contingency table which is a version of a 2x2 confusion matrix. The contingency table compares the outcome of two different tests on the same table. It is commonly used in medicine to determine whether a drug or treatment affects a population. The McNemar test can also be applied to compare the performance of two

different classifiers. The contingency table of two different classifiers is represented as seen in Table 6.1.

Table 6.1. Contingency table for two classifiers.

	Classifier B correct	**Classifier B incorrect**
Classifier A correct	(X), number of samples correctly classified by both A and B	(Y), number of samples misclassified by B but not by A
Classifier A incorrect	(Z), number of samples misclassified by A but not by B	(T), number of samples misclassified by both A and B

The null hypothesis, H_0, states that the performance of the two classifiers are similar, i.e., $p(X) + p(Y) = p(X) + p(Z)$ and $p(Z) + p(T) = p(Y) + p(T)$. Then, the null and alternative hypotheses are defined as follows:

$$H_0 : p(Y) = p(Z)\,, \tag{6.5}$$

$$H_1 : p(Y) \neq p(Z)\,. \tag{6.6}$$

Then, McNemar test statistic, chi-squared, is calculated as follows:

$$\text{chi-squared} = \frac{(Y - Z)^2}{Y + Z}\,. \tag{6.7}$$

For a predefined significance threshold, i.e., $\alpha = 0.05$, the p-value is calculated assuming the null hypothesis is true. If the p-value is lower than the chosen significance threshold, α, then the null hypothesis is rejected. This means that the performance of the two systems differs.

We apply McNemar's test to compare our two stage's performance with each other. We use the alternative hypothesis that the performance of our two stages is

different. We want to prove that our second stage significantly improves the predictions of the first stage. If we obtain a better F1-score with $alpha < 0.05$ on the second stage, we can conclude that the performance improvement is statistically significant.

6.3. Results of the First Stage

Since Five Benchmarking PPI Corpora consists of separate training and test datasets, we fine-tuned the pre-trained BioBERT(v1.1+PubMed) model on the training datasets of five corpora and obtained five different trained models for the PPI Extraction task in our first stage. Then, as our first experiment, we evaluated the performance of our first stage on the test datasets of five corpora. Finally, we reported our F1 results by comparing all previous related work that has been constructed on the same five PPI corpora in Table 6.2.

Our first stage outperforms all previous studies on AIMed corpus, one of the most enormous and complex corpus, and we obtain a state-of-the-art F1-score with a relative improvement of 2.1%. Furthermore, we obtain the second-best result on BioInfer corpus and the third-best result on HPRD50 and IEPA corpora. Our first stage performance on the LLL corpus, on the other hand, lags below earlier studies, with a relative 11.6 percent F1-score. The reason may be the size and the characteristics of the LLL corpus. It is the smallest corpus among five corpora, which has 300 sentences, and other kernel-based approaches and sentence representations handle the characteristics of the LLL corpus better than BioBERT.

As previously given in Table 4.2, all five corpora except the LLL corpus are highly imbalanced, i.e., they have a substantially higher proportion of negative samples than positives. Therefore, some studies provide macro-averaged F1 scores, which is the average F1-score of positive and negative instances. Therefore, we have also calculated macro-averaged F1 scores in our experiments, and we compare these results in Table 6.3, separately.

Lastly, we evaluated the first stage performance without applying protein name replacement in preprocessing phase since we utilize a transformer-based BioBERT model in this stage. However, since some protein names are rare and, they bring some biases to sentences, they negatively affected the performance. The obtained F1-scores are 31.4%, 41.0%, 45.9%, 62.8%, and 68.6% on five corpora. The results reveal that the performance loss is more significant on larger corpora and, therefore, protein name replacement has a positive effect on our system.

The compared models are briefly described as follows:

(i) Edit kernel. The similarity between two DPT of sentences is calculated by edit distance similarity [11].

(ii) APG kernel. The weighted sum of all dependency paths between target proteins is used. They assign higher weights for the shortest paths, and lower weights to other paths [39].

(iii) hybrid kernel. BOW, ST and Graph kernels with syntactic, dependency and deep parsers are combined as a hybrid kernel [40].

(iv) kBSPS kernel. Nodes within a distance k from the shortest path are included in the representation [41].

(v) wws kernel. Different weights are assigned to common substrings of two shortest path strings [42].

(vi) NHGK. Each node label of dependency graphs is map into a fixed-length binary array, and values are updated according to neighbor dependency nodes [43].

(vii) EDG kernel. The extended dependency graph vertices are constructed with text, part-of-speech tags, and word lemmas [44].

(viii) CTK. Convolution tree kernel integrates interaction pattern trees for sentence representation [45].

(ix) DSTK. Distributed smoothed tree kernel utilizes both syntactic and semantic vectors to overcome the shortcomings of information loss from single kernel approaches [47].

(x) DNN. A feed-forward neural network architecture for PPI extraction [48].

(xi) DCNN. The deep convolutional neural network implementation that combines word and position embeddings [49].

(xii) sdpCNN. The sdpCNN model learns features from the shortest dependency paths with a CNN architecture [50].

(xiii) MCCNN. A CNN model with a multichannel word embedding input layer [51].

(xiv) McDepCNN. A CNN model with multichannel dependency-based word embeddings [54].

(xv) LSTM. The bi-directional RNN architecture with Long Short-Term Memory [55].

(xvi) hybrid CNN-RNN. A hybrid CNN and RNN based model. While the RNN model captures sentence features from sentence sequences, the CNN model utilizes SDPs [56].

(xvii) t-LSTM. The dependency parse trees are traversed through a tree-LSTM structure in order to learn the structural information of sentences. t-LSTM model is combined with a structured attention-based model [57].

(xviii) DRCNN. A deep residual CNN model architecture. A residual connection layer is added between the convolutional layer to make the model stronger [58].

Table 6.2. Performance comparison of our first stage based on Five Benchmarking PPI corpora.

Model	**Year**	**Corpus**	**Precision**	**Recall**	**F1-Score**
Edit kernel	2007	AIMed	77.5	43.5	55.6
APG kernel	2008	AIMed	52.9	61.8	56.4
Hybrid kernel	2009	AIMed	55.0	68.8	60.8
kBSPS kernel	2009	AIMed	49.4	44.7	46.1
wws kernel	2010	AIMed	61.4	53.3	56.6
NHGK	2011	AIMed	54.9	68.5	60.2
EDG kernel	2015	AIMed	57.3	65.3	61.1
CTK	2016	AIMed	57.2	64.5	60.6
DSTK	2017	AIMed	68.9	73.2	71.0
DNN	2016	AIMed	51.5	63.4	56.1

Table 6.2. Performance comparison of our first stage (cont.)

Model	**Year**	**Corpus**	**Precision**	**Recall**	**F1-Score**
sdpCNN	2016	AIMed	64.8	67.8	66.0
MCCNN	2016	AIMed	76.4	69.0	72.4
McDepCNN	2017	AIMed	67.3	60.1	63.5
LSTM	2017	AIMed	**78.8**	75.2	76.9
hybrid CNN-RNN	2018	AIMed	59.9	63.5	61.7
Our First Stage/ BioBERT Model	2021	AIMed	78.4	**79.6**	**79.0**
Edit kernel	2007	BioInfer	-	-	-
APG kernel	2008	BioInfer	56.7	67.2	61.3
Hybrid kernel	2009	BioInfer	65.7	71.1	68.1
kBSPS kernel	2009	BioInfer	-	-	-
wws kernel	2010	BioInfer	61.8	54.2	57.6
NHGK	2011	BioInfer	59.3	68.1	63.4
EDG kernel	2015	BioInfer	57.6	59.9	58.7
CTK	2016	BioInfer	68.6	70.3	69.4
DSTK	2017	BioInfer	75.7	76.9	76.3
DNN	2016	BioInfer	53.9	72.9	61.6
sdpCNN	2016	BioInfer	73.4	77.0	75.2
MCCNN	2016	BioInfer	81.3	78.1	79.6
McDepCNN	2017	BioInfer	62.7	68.2	65.3
LSTM	2017	BioInfer	**87.0**	**87.4**	**87.2**
hybrid CNN-RNN	2018	BioInfer	62.7	67.3	64.8
Our First Stage/ BioBERT Model	2021	BioInfer	81.2	78.9	79.9
Edit kernel	2007	HPRD50	-	-	-
APG kernel	2008	HPRD50	64.3	65.8	63.4
Hybrid kernel	2009	HPRD50	68.5	76.1	70.9
kBSPS kernel	2009	HPRD50	66.7	80.2	70.9

Table 6.2. Performance comparison of our first stage (cont.)

Model	**Year**	**Corpus**	**Precision**	**Recall**	**F1-Score**
wws kernel	2010	HPRD50	66.7	69.2	67.8
NHGK	2011	HPRD50	67.8	85.3	74.6
EDG kernel	2015	HPRD50	76.7	83.3	79.9
CTK	2016	HPRD50	63.8	81.2	71.5
DSTK	2017	HPRD50	76.3	84.2	**80.0**
DNN	2016	HPRD50	58.7	**92.4**	71.3
sdpCNN	2016	HPRD50	-	-	-
MCCNN	2016	HPRD50	-	-	-
McDepCNN	2017	HPRD50	-	-	-
LSTM	2017	HPRD50	-	-	-
hybrid CNN-RNN	2018	HPRD50	75.1	76.4	75.6
Our First Stage/ BioBERT Model	2021	HPRD50	**82.6**	73.1	77.6
Edit kernel	2007	IEPA	-	-	-
APG kernel	2008	IEPA	69.6	82.7	75.1
Hybrid kernel	2009	IEPA	67.5	78.6	71.7
kBSPS kernel	2009	IEPA	70.4	73.0	70.8
wws kernel	2010	IEPA	73.8	71.8	72.9
NHGK	2011	IEPA	72.4	79.8	75.3
EDG kernel	2015	IEPA	69.9	76.2	72.9
CTK	2016	IEPA	62.5	83.3	71.4
DSTK	2017	IEPA	**75.9**	**85.2**	**80.2**
DNN	2016	IEPA	71.8	79.4	74.22
sdpCNN	2016	IEPA	-	-	-
MCCNN	2016	IEPA	-	-	-
McDepCNN	2017	IEPA	-	-	-
LSTM	2017	IEPA	-	-	-
hybrid CNN-RNN	2018	IEPA	73.2	84.4	78.2

Table 6.2. Performance comparison of our first stage (cont.)

Model	**Year**	**Corpus**	**Precision**	**Recall**	**F1-Score**
Our First Stage/ BioBERT Model	2021	IEPA	74.2	82.1	78.0
Edit kernel	2007	LLL	-	-	-
APG kernel	2008	LLL	72.5	87.2	76.8
Hybrid kernel	2009	LLL	77.6	86.0	80.1
kBSPS kernel	2009	LLL	76.8	91.8	82.2
wws kernel	2010	LLL	76.9	91.2	82.4
NHGK	2011	LLL	86.2	92.1	89.1
EDG kernel	2015	LLL	**92.1**	78.2	84.6
CTK	2016	LLL	73.2	89.6	80.6
DSTK	2017	LLL	87.3	91.2	**89.2**
DNN	2016	LLL	76.0	91.0	81.4
sdpCNN	2016	LLL	-	-	-
MCCNN	2016	LLL	-	-	-
McDepCNN	2017	LLL	-	-	-
LSTM	2017	LLL	-	-	-
hybrid CNN-RNN	2018	LLL	76.6	**96.1**	85.2
Our First Stage/ BioBERT Model	2021	LLL	70.3	86.7	77.6

Table 6.3. Performance comparison of our first stage by macro precision, macro recall and macro F1-score.

Model	**Year**	**Corpus**	**Precision**	**Recall**	**macro-averaged F1-Score**
DCNN	2016	AIMed	**88.6**	81.7	85.0
t-LSTM	2019	AIMed	81.4	81.9	81.6
DRCNN	2019	AIMed	87.0	85.9	86.3
Our First Stage/ BioBERT Model	2021	AIMed	87.0	**87.5**	**87.2**
DCNN	2016	BioInfer	72.1	77.5	74.7
t-LSTM	2019	BioInfer	88.9	89.3	89.1
DRCNN	2019	BioInfer	**91.5**	**90.8**	**91.1**
Our First Stage/ BioBERT Model	2021	BioInfer	84.5	84.9	84.6
DCNN	2016	HPRD50	-	-	-
t-LSTM	2019	HPRD50	81.7	**82.3**	81.3
DRCNN	2019	HPRD50	82.6	82.2	81.7
Our First Stage/ BioBERT Model	2021	HPRD50	**83.9**	82.0	**82.7**
DCNN	2016	IEPA	-	-	-
t-LSTM	2019	IEPA	78.6	78.7	78.5
DRCNN	2019	IEPA	78.4	78.5	78.3
Our First Stage/ BioBERT Model	2021	IEPA	**80.3**	**81.1**	**80.5**
DCNN	2016	LLL	-	-	-
t-LSTM	2019	LLL	**84.8**	**84.3**	**84.2**
DRCNN	2019	LLL	83.2	82.7	82.6
Our First Stage/ BioBERT Model	2021	LLL	76.8	75.6	77.6

6.3.1. Cross-Corpus Results

Secondly, we performed a cross-corpus evaluation. This time, we trained our model on one corpus and tested the performance on the others. Cross-corpus evaluation allows us to determine if our BioBERT model can perform well on other corpora so that it can be generalized for different PPI datasets. We can also discover similarities among corpora and create larger datasets that may reduce the data sparseness for deep learning models. Table 6.4 depicts the cross-corpus results.

Table 6.4. Cross-Corpus evaluation results on Five Benchmarking PPI corpora.

Training Corpus	**Test Corpus**	**Precision**	**Recall**	**F1-Score**
AIMed	AIMed	78.4	79.6	79.0
AIMed	BioInfer	84.0	26.9	40.7
AIMed	HPRD50	75.0	69.2	72.0
AIMed	IEPA	80.0	7.1	13.1
AIMed	LLL	100	3.3	6.5
BioInfer	AIMed	39.0	89.5	54.3
BioInfer	BioInfer	81.2	78.9	79.9
BioInfer	HPRD50	60.0	80.8	68.9
BioInfer	IEPA	67.9	67.9	67.9
BioInfer	LLL	78.3	60.0	67.9
HPRD50	AIMed	52.6	63.9	57.7
HPRD50	BioInfer	72.8	28.9	41.3
HPRD50	HPRD50	82.6	73.1	77.6
HPRD50	IEPA	69.6	57.1	62.7
HPRD50	LLL	70.0	23.3	35.0
IEPA	AIMed	34.3	64.9	44.8
IEPA	BioInfer	59.6	40.1	48.0
IEPA	HPRD50	57.9	84.6	68.8
IEPA	IEPA	74.2	82.1	78.0

Table 6.4. Cross-Corpus evaluation results (cont.)

Training Corpus	**Test Corpus**	**Precision**	**Recall**	**F1-Score**
IEPA	LLL	80.0	53.3	64.0
LLL	AIMed	24.0	78.5	36.7
LLL	BioInfer	38.1	69.2	49.1
LLL	HPRD50	46.8	84.6	60.3
LLL	IEPA	50.5	89.3	64.5
LLL	LLL	70.3	86.7	77.6

6.3.2. Combined-Corpus Results

Since the performance of the fine-tuned models highly depends on the training data used, we also investigated the effect of combining the training datasets of five corpora as a single source. This time we fine-tuned the pre-trained BioBERT(v1.1+PubMed) model on a single combined dataset source and obtained a single model. And then, as our third experiment, we evaluated the performance of our single trained model on test sets of five corpora. The results suggest that when the model is trained on a larger dataset, the performance increases on the smaller corpora: HPRD50 and LLL, and decreases on larger corpora: AIMed and BioInfer. While corpus size has a positive effect on the performance of our BioBERT model, inter-corpus features can degrade performance in already large datasets. The Table 6.5 summarizes the overall comparison.

Table 6.5. Combined corpus evaluation results on Five Benchmarking PPI corpora.

Training Corpus	**Test Corpus**	**Precision**	**Recall**	**F1-Score**
AIMed	AIMed	**78.4**	79.6	**79.0**
Combined Corpora	AIMed	60.8	**84.3**	70.6

Table 6.5. Combined corpus evaluation results (cont.)

Training Corpus	**Test Corpus**	**Precision**	**Recall**	**F1-Score**
BioInfer	BioInfer	**81.2**	**78.9**	**79.9**
Combined Corpora	BioInfer	80.3	76.4	78.0
HPRD50	HPRD50	**82.6**	73.1	77.6
Combined Corpora	HPRD50	77.8	**80.8**	**79.2**
IEPA	IEPA	74.2	**82.1**	**78.0**
Combined Corpora	IEPA	**75.4**	76.8	76.1
LLL	LLL	70.3	**86.7**	77.6
Combined Corpora	LLL	**78.1**	83.3	**80.6**
Combined Corpora	Combined test	75.1	76.2	75.7

6.4. Results of The Second Stage

In the second stage, we trained our GAN model by combining (i) the training datasets of five corpora with ground truth labels and (ii) the test datasets of five corpora with BioBERT annotations. At the end of the training phase, we changed the labels of some test sentences to negative, in which our generator has assigned low probabilities, but BioBERT has annotated positive. Then we calculate F1-score for every five datasets by comparing the ground truth test labels.

We conducted two experiments for the second stage evaluation. First, we run our second stage on top of the BioBERT model trained on every five corpora separately.

Second, we run our second stage on top of the BioBERT model trained on the combined corpus. In both experiments, we tested our second stage on the test datasets of five corpora. Hence, we report ten different results: five for the first case experiments and five for the second case experiments.

Moreover, since our GAN model learns each time differently, it produces different results. Therefore, we got five different runs for each test scenario, and the average of these runs was reported as an overall second stage performance. Each individual run provides the precision, recall, and F1 score results of the second stage. Since our second stage changes some positive-labelled predictions of the BioBERT model to negatives, we also report the number of changes in false-positives and true-positives. Lastly, we conduct McNemar's Test to evaluate whether we can obtain statistically significant improvements on each second stage run compared to first stage results. The contingency table, chi-squared value, and p-values of McNemar test statistics were all provided in the final result table.

6.4.1. Second Stage Results (First Stage trained on separate corpora)

This section provides our second stage results when the BioBERT model is trained on five corpora separately.

6.4.1.1. McNemar Test Results on AIMed. In this experiment, we use the first stage predictions of our BioBERT model trained on AIMed corpus. The AIMed training dataset with ground truth labels and AIMed test dataset with BioBERT predictions are used during adversarial learning. The experimental results of five different runs are given in Table 6.6. The results show that the F1 score drops 6.9% on average compared to the first stage. The p-value of the McNemar test, which is $0.015 < 0.05$, also proves that the performance loss is statistically significant.

Table 6.6. The second stage results on AIMed Corpus.

Run	**contingency table**	**chi-sq**	**p-value**	**FP↓**	**TP↓**	**P**	**R**	**F1**
1	[[992, 10] [22, 71]]	3.78	0.052	10	22	**80.2**	68.1	73.7
2	[[991, 8] [23, 73]]	6.32	0.012	8	23	**79.1**	67.5	72.9
3	[[983, 9] [31, 72]]	11.03	0.001	9	31	**78.6**	63.4	70.1
4	[[993, 6] [21, 75]]	7.26	0.007	6	21	78.4	68.6	73.2
5	[[984, 10] [30, 71]]	9.03	0.003	10	30	**79.2**	63.9	70.7
avg. (2nd Stg.)	-	7.48	0.015	9	25	**79.1**	66.3	72.1
1st Stg.	-	-	-	-	-	78.4	**79.6**	**79.0**

6.4.1.2. McNemar Test Results on BioInfer. In this experiment, we use the first stage predictions of our BioBERT model trained on the BioInfer corpus. The BioInfer training dataset with ground truth labels and BioInfer test dataset with BioBERT predictions are used during adversarial learning. The experimental results of five different runs are given in Table 6.7. The results demonstrate that although we can obtain performance improvement on some runs, the F1 score drops very slightly on average compared to the first stage. None of each run can obtain statistically significant changes, i.e., all p-values are greater than 0.05, and the null hypothesis is not rejected. Therefore, we can conclude that our second stage does not contribute to our first stage when we train BioBERT on BioInfer corpus.

Table 6.7. The second stage results on BioInfer Corpus.

Run	**contingency table**	**chi-sq**	**p-value**	**FP↓**	**TP↓**	**P**	**R**	**F1**
1	[[1303, 6] [5, 227]]	0.0	1.0	6	5	**81.5**	78.5	79.8
2	[[1297, 16] [11, 217]]	0.59	0.441	16	11	**82.2**	78.2	79.8
3	[[1282, 29] [26, 204]]	0.07	0.787	29	26	**82.8**	77.0	79.1
4	[[1305, 9] [3, 224]]	2.08	0.149	9	3	**82.0**	78.9	**80.2**
5	[[1296, 20] [12, 213]]	1.53	0.216	20	12	**82.6**	78.3	**80.0**
avg. (2nd Stg.)	-	0.85	0.519	16	11	**82.2**	78.2	79.8
1st Stg.	-	-	-	-	-	81.2	**78.9**	**79.9**

6.4.1.3. McNemar Test Results on HPRD50. In this experiment, we use the first stage predictions of our BioBERT model trained on the HPRD50 corpus. The HPRD50 training dataset with ground truth labels and HPRD50 test dataset with BioBERT predictions are used during adversarial learning. The experimental results of five different runs are given in Table 6.8. Similar to BioInfer, all p-values of the McNamer test are greater than 0.05, which means we can not reject the null hypothesis. Consequently, our second stage can not obtain significant differences compared to the first stage results.

Table 6.8. The second stage results on HPRD50 Corpus.

Run	**contingency table**	**chi-sq**	**p-value**	**FP↓**	**TP↓**	**P**	**R**	**F1**
1	[[58, 1] [1, 10]]	0.5	0.480	1	1	**85.7**	69.2	76.6
2	[[57, 2] [2, 9]]	0.25	0.617	2	2	**89.5**	65.4	75.6
3	[[57, 1] [2, 10]]	0.0	1.0	1	2	**85.0**	65.4	73.9
4	[[59, 1] [0, 10]]	0.0	1.0	1	0	**86.4**	**73.1**	**79.2**
5	[[59, 2] [0, 9]]	0.5	0.480	2	0	**90.5**	**73.1**	**80.9**
avg. (2nd Stg.)	-	0.25	0.715	1	1	**87.4**	69.2	77.2
1st Stg.	-	-	-	-	-	82.6	**73.1**	**77.6**

6.4.1.4. McNemar Test Results on IEPA. In this experiment, we use the first stage predictions of our BioBERT model trained on the IEPA corpus. The IEPA training dataset with ground truth labels and IEPA test dataset with BioBERT predictions are used during adversarial learning. The experimental results of five different runs are given in Table 6.9. Similar to BioInfer and HPRD50, all p-values of the McNamer test are greater than 0.05; the null hypothesis is not rejected. Therefore, our second stage can not obtain significant differences compared to the first stage results.

Table 6.9. The second stage results on IEPA Corpus.

Run	**contingency table**	**chi-sq**	**p-value**	**FP↓**	**TP↓**	**P**	**R**	**F1**
1	[[107, 5] [3, 21]]	0.13	0.724	5	3	**79.6**	76.8	**78.2**
2	[[106, 3] [4, 23]]	0.0	1.0	3	4	**76.4**	75.0	75.7
3	[[106, 6] [4, 20]]	0.1	0.752	6	4	**80.8**	75.0	77.8
4	[[107, 3] [3, 23]]	0.17	0.683	3	3	**76.8**	76.8	76.8
5	[[107, 2] [3, 24]]	0.0	1.0	2	3	**75.4**	76.8	76.1
avg. (2nd Stg.)	-	0.08	0.832	4	3	**77.8**	76.1	76.9
1st Stg.	-	-	-	-	-	74.2	**82.1**	**78.0**

6.4.1.5. McNemar Test Results on LLL. In this experiment, we use the first stage predictions of our BioBERT model trained on the LLL corpus. The LLL training dataset with ground truth labels and LLL test dataset with BioBERT predictions are used during adversarial learning. The experimental results of five different runs are given in Table 6.10. Similar to BioInfer, HPRD50, and IEPA, all p-values of the McNamer test are greater than 0.05; the null hypothesis is not rejected. Therefore, our second stage can not obtain significant differences compared to the first stage results.

Table 6.10. The second stage results on LLL Corpus.

Run	**contingency table**	**chi-sq**	**p-value**	**FP↓**	**TP↓**	**P**	**R**	**F1**
1	[[45, 0] [1, 15]]	0.0	1.0	0	1	69.4	83.3	75.8
2	[[44, 1] [2, 14]]	0.0	1.0	1	2	**70.6**	80.0	75.0
3	[[45, 2] [1, 13]]	0.0	1.0	2	1	**73.5**	83.3	**78.1**
4	[[44, 2] [2, 13]]	0.25	0.617	2	2	**72.7**	80.0	76.2
5	[[43, 2] [3, 13]]	0.0	1.0	2	3	**71.9**	76.7	74.2
avg. (2nd Stg.)	-	0.05	0.923	1	2	**71.6**	80.7	75.9
1st Stg.	-	-	-	-	-	70.3	**86.7**	**77.6**

6.4.2. Second Stage Results (First Stage trained on combined corpus)

This section provides our second stage results when the BioBERT model is trained on the training dataset of five corpora as a single source.

6.4.2.1. McNemar Test Results on AIMed. Here, we use the first stage predictions of our BioBERT model trained on the combined corpus. The AIMed train set with ground truth labels and AIMed test dataset with BioBERT predictions are used during adversarial learning. The experimental results of five different runs are given in Table 6.11. The results show that the second stage achieves 1% F1-score and 7.2% precision score improvements compared to the first stage. The p-value of the McNemar test, which is $0.012 < 0.05$, also proves that the performance improvement is statistically significant.

Table 6.11. The second stage results on AIMed Corpus.

Run	**contingency table**	**chi-sq**	**p-value**	**FP↓**	**TP↓**	**P**	**R**	**F1**
1	[[953, 21] [8, 113]]	4.97	0.026	21	8	**64.8**	80.1	**71.7**
2	[[935, 48] [26, 86]]	5.96	0.014	48	26	**70.7**	70.7	**70.7**
3	[[942, 39] [19, 95]]	6.22	0.012	39	19	**68.6**	74.3	**71.4**
4	[[945, 38] [16, 96]]	8.17	0.004	38	16	**68.8**	75.9	**72.1**
5	[[950, 30] [11, 104]]	7.9	0.005	30	11	**67.0**	78.5	**72.3**
avg. (2nd stg.)	-	6.64	0.012	35	16	**68.0**	75.9	**71.6**
1st Stg.	-	-	-	-	-	60.8	**84.3**	70.6

6.4.2.2. McNemar Test Results on BioInfer. In this experiment, we use the first stage predictions of our BioBERT model trained on the combined corpus. The BioInfer training dataset with ground truth labels and BioInfer test dataset with BioBERT predictions are used during adversarial learning. The experimental results of five different runs are given in Table 6.12. The results demonstrate that we obtain performance improvement on precision, recall and F1 scores for each run. All p-values are lower than 0.05, and the alternative hypothesis is accepted. Hence, we can conclude that our second stage contributes significantly to our first stage when we train BioBERT on the combined corpus.

Table 6.12. The second stage results on BioInfer Corpus.

Run	**contingency table**	**chi-sq**	**p-value**	**FP↓**	**TP↓**	**P**	**R**	**F1**
1	[[1292, 8] [1, 240]]	4	0.046	8	1	**81.2**	**76.7**	**78.5**
2	[[1283, 25] [10, 223]]	5.6	0.017	25	10	**82.8**	76.4	**78.7**
3	[[1291, 14] [2, 234]]	7.56	0.006	14	2	**81.9**	**76.8**	**78.8**
4	[[1287, 18] [6, 230]]	5.04	0.025	18	6	**82.1**	**76.5**	**78.6**
5	[[1290, 13] [3, 235]]	5.06	0.024	13	3	**81.7**	**76.7**	**78.6**
avg. (2nd Stg.)	-	5.45	0.023	16	4	**81.9**	**76.6**	**78.6**
1st Stg.	-	-	-	-	-	80.3	76.4	78.0

6.4.2.3. McNemar Test Results on HPRD50. In this experiment, we use the first stage predictions of our BioBERT model trained on the combined corpus. The HPRD50 training dataset with ground truth labels and HPRD50 test dataset with BioBERT predictions are used during adversarial learning. The experimental results of five different runs are given in Table 6.13. The results demonstrate that although we obtain performance improvement on the second stage, none of the p-values is lower than 0.05, and the null hypothesis is not rejected. Consequently, our second stage does not contribute significantly to our first stage.

Table 6.13. The second stage results on HPRD50 Corpus.

Run	**contingency table**	**chi-sq**	**p-value**	**FP↓**	**TP↓**	**P**	**R**	**F1**
1	[[58, 1] [1, 10]]	0.5	0.480	1	1	**80.0**	76.9	78.4
2	[[59, 2] [0, 9]]	0.5	0.480	2	0	**84.0**	80.0	**82.4**
3	[[59, 1] [0, 10]]	0.0	1.0	1	0	**80.8**	80.8	**80.8**
4	[[57, 1] [2, 10]]	0.0	1.0	1	2	**79.2**	73.1	76.0
5	[[58, 2] [1, 9]]	0.0	1.0	2	1	**83.3**	76.9	**80.0**
avg. (2nd Stg.)	-	0.2	0.792	1	1	**81.5**	77.5	**79.5**
1st Stg.	-	-	-	-	-	77.8	**80.8**	79.2

6.4.2.4. McNemar Test Results on IEPA. In this experiment, we use the first stage predictions of our BioBERT model trained on the combined corpus. The IEPA training dataset with ground truth labels and IEPA test dataset with BioBERT predictions are used during adversarial learning. The experimental results of five different runs are given in Table 6.14. Similar to HPRD50, all p-values of the McNamer test are greater than 0.05; the null hypothesis is not rejected. Therefore, our second stage can not obtain significant differences compared to the first stage results.

Table 6.14. The second stage results on IEPA Corpus.

Run	**contingency table**	**chi-sq**	**p-value**	**FP↓**	**TP↓**	**P**	**R**	**F1**
1	[[107, 6] [2, 21]]	1.13	0.289	6	2	**83.7**	73.2	**78.1**
2	[[107, 2] [2, 25]]	0.25	0.617	2	2	**77.4**	73.2	75.2
3	[[107, 4] [2, 23]]	0.17	0.683	4	2	**80.3**	73.2	**76.6**
4	[[105, 1] [1, 26]]	0.5	0.478	1	1	**76.4**	75.0	75.7
5	[[106, 2] [3, 25]]	0.0	1.0	2	3	**76.9**	71.4	74.1
avg. (2nd Stg.)	-	0.41	0.613	3	2	**78.94**	73.2	75.9
1st Stg.	-	-	-	-	-	75.4	**76.8**	**76.1**

6.4.2.5. McNemar Test Results on LLL. In this experiment, we use the first stage predictions of our BioBERT model trained on the combined corpus. The LLL training dataset with ground truth labels and LLL test dataset with BioBERT predictions are used during adversarial learning. The experimental results of five different runs are given in Table 6.15. The results demonstrate that none of the p-values is lower than 0.05, and the null hypothesis is not rejected. Consequently, our second stage does not contribute significantly to our first stage.

Table 6.15. The second stage results on LLL Corpus.

Run	**contingency table**	**chi-sq**	**p-value**	**FP↓**	**TP↓**	**P**	**R**	**F1**
1	[[45, 2] [4, 10]]	0.17	0.683	2	4	**80.8**	70.0	75.0
2	[[48, 0] [1, 12]]	0.0	1.0	0	1	77.4	80.0	78.7
3	[[47, 0] [2, 12]]	0.5	0.480	0	2	76.7	76.7	76.7
4	[[48, 1] [1, 11]]	0.5	0.480	1	1	**80.0**	80.0	80.0
5	[[45, 1] [4, 11]]	0.8	0.371	1	4	77.8	70.0	73.7
avg. (2nd Stg.)	-	0.39	0.603	1	2	**78.5**	75.3	76.8
1st Stg.	-	-	-	-	-	78.1	**83.3**	**80.6**

6.4.2.6. McNemar Test Results on Combined Test Set. In this experiment, we use the first stage predictions of our BioBERT model trained on the combined corpus. The training dataset of combined corpus with ground truth labels and combined test set with BioBERT predictions are used during adversarial learning. The experimental results of five different runs are given in Table 6.16. The results demonstrate that the second stage achieves 0.8% F1-score and 4% precision score improvements compared to the first stage. The p-value of the McNemar test, which is $0.004 < 0.05$, also proves that the performance improvement is statistically significant.

Table 6.16. The second stage results on combined test set.

Run	**contingency table**	**chi-sq**	**p-value**	**FP↓**	**TP↓**	**P**	**R**	**F1**
1	[[2539, 43] [18, 303]]	9.44	0.002	43	18	**79.6**	73.9	**76.6**
2	[[2542, 36] [15, 310]]	7.84	0.005	36	15	**78.8**	74.3	**76.5**
3	[[2549, 25] [8, 321]]	7.76	0.005	25	8	**77.8**	75.3	**76.5**
4	[[2544, 33] [13, 313]]	7.85	0.005	33	13	**78.6**	74.6	**76.5**
5	[[2532, 54] [25, 292]]	9.92	0.002	54	25	**80.7**	72.9	**76.6**
avg. (2nd Stg.)	-	8.56	0.004	38	16	**79.1**	74.2	**76.5**
1st Stg.	-	-	-	-	-	75.1	**76.2**	75.7

7. A CASE STUDY: HOST-PATHOGEN INTERACTION EXTRACTION ON COVID-19 DATASET

In infectious diseases, pathogens such as bacteria, viruses, protozoans infect host cells and establish themselves with high rates of reproduction within the host [70]. These microparasites may be resistant to host immunity, and during their lifespan, protein interactions are established between host and pathogen proteins. Although the presence of pathogens may not always cause disease in hosts, their interaction with the host cell is adequate to define Host-Pathogen Interaction (HPI).

The current coronavirus pandemic, COVID-19, is a global pandemic of coronavirus disease, which is caused by Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2) [71]. During its lifespan, including attachment and entry to the human host cell, the replication and transcription phases, some structural proteins of human coronavirus (HCoV) such as spike (S) protein, envelope (E), and membrane (M) interact with several host proteins and cell receptors such as, angiotensin-converting enzyme 2 (ACE2) and dipeptidyl peptidase 4 (DPP4) etc. [72]. The human coronavirus and host interaction has been investigated in many biomedical studies and, with the current pandemic, the number of COVID-19 related papers are growing extremely fast. This makes it increasingly difficult for researchers to uncover the relevant findings of COVID-19.

In this case study, we apply our system to extract Host-Pathogen Interactions from coronavirus related sentences. Since there is no previous study on this topic, we created our own dataset by using a biomedical literature mining tool, SciMiner [73]. After candidate sentences with host and pathogen entities are automatically extracted, they all are examined one by one and annotated manually. At the end of the data curation, we obtain a golden standard HPI dataset for COVID-19. We apply similar preprocessing steps as we did in the PPI extraction task. Since the corpus size is too small, we excluded our GAN model on experiments.

7.1. Dataset Curation

We have created our own COVID-19 dataset for Host-Pathogen Interaction Extraction Task. Sentences were pre-extracted from PubMed publications related to COVID-19, which were published until June 2020, using a biomedical literature mining tool, SciMiner [73]. The pre-extracted data contain the following properties:

- sentence ID,
- PubMed ID of the corresponding sentence,
- type of entity representation (symbol or name),
- Entity ID,
- match entity term,
- actual entity term,
- raw sentence.

Sample raw sentences extracted by SciMiner can be seen in Figure 7.1.

SentID	PMID	Type	HUGOID	Symbol	MatchTerm	ActualTerm	Raw Sentence
40	32427429	NAME	9958	REN	renin	renin	Renin-Angiotensin-Aldosterone System Inhibitors in Covid-19.
92	32269089	SYMBOL	13557	ACE2	ACE2	ACE-2	ACE-2 expression in the small airway epithelia of smokers and COPD patients: im
166	32510973	SYMBOL	13557	ACE2	ACE2	ACE2	SARS-CoV-2 may manipulate mitochondrial function indirectly, first by ACE2 reg
168	32510973	SYMBOL	13557	ACE2	ACE2	ACE2	We argue that a decline in ACE2 function in aged individuals, coupled with the a
202	32422146	SYMBOL	13557	ACE2	ACE2	ACE2	Type 2 inflammation modulates ACE2 and TMPRSS2 in airway epithelial cells.
202	32422146	SYMBOL	11876	TMPRSS2	TMPRSS2	TMPRSS2	Type 2 inflammation modulates ACE2 and TMPRSS2 in airway epithelial cells.

Figure 7.1. Sample sentences extracted by SciMiner.

Since the file data format is entity-based, the same sentence can be seen several times if the sentence contains multiple host or pathogen entities. Two separate files exist for host and pathogen entities, respectively. Some statistics about host and pathogen files are given in Table 7.1. We apply the following steps in order to match the host and pathogen entities in two separate files:

- We read both files and store them in memory.
- We extract sentence ids that occur in both host and pathogen files.

- For each common sentence, we retrieve related host and pathogen entities. Then for each host and pathogen pair combination, we process the sentence. We apply the same preprocessing steps that we have applied for the PPI dataset. We replace the entity pair terms with PROTEIN1 and PROTEIN2 keywords and other entities rather than entity pairs with a PROTEIN keyword. For each entity pair combination, we obtain different versions of the same sentence.

Table 7.1. Same statistics for host and pathogen files.

File	Host	Pathogen
Number of Pubmed Articles	2698	332
Number of Unique Sentences	6106	631
Number of Extracted Entities	11262	949

After we obtain candidate host and pathogen entity pairs for each sentence, we further eliminate some noisy data. Since SciMiner may produce wrong identifications due to syntactic difficulty in processing some sentences, we need to manually eliminate the wrong identification of entities. These difficulties can be summarized as follows;

(i) The frequently seen ACE-2 protein may written in long form with Roman numerals as angiotensin-converting enzyme II, so it is wrongly identified as angiotensin converting enzyme.

(ii) Some complex protein structures like Microsomal prostaglandin E2 synthase-1 (mPGES-1) could be identified as two seperate entities since an entity named E2 also exists.

(iii) Some complex protein representations like B(0)AT1 could be partially identified as AT1.

(iv) Although it is less frequently seen some host entities may wrongly identified as pathogens and vice versa.

After we eliminate the wrongly identified host-pathogen pairs, we obtain 96 final sentences. Finally, these pairs are manually annotated regarding they exhibit protein-protein interaction semantically. Among the 96 sentences, we have labelled 64 host and pathogen pairs that exhibit interactions semantically.

7.2. Experiments and Results

We have used our BioBERT model trained on Five Benchmarking PPI corpora with our hyperparameter settings given in Table 5.3 to extract host-pathogen entities. We excluded the second stage since it requires separate training data for learning and our dataset is not large enough to achieve good results on this dataset. BioBERT model, on the other hand, is pre-trained on large biomedical corpora and can also achieve good results on small datasets. As an evaluation metric, we used F1-score and obtained 65.8% which is very promising considering it is a type of cross-corpus evaluation. Figure 7.2 provides the confusion matrix of our predictions as well. Interestingly, we have obtained zero false positives, and therefore our precision score is 100%.

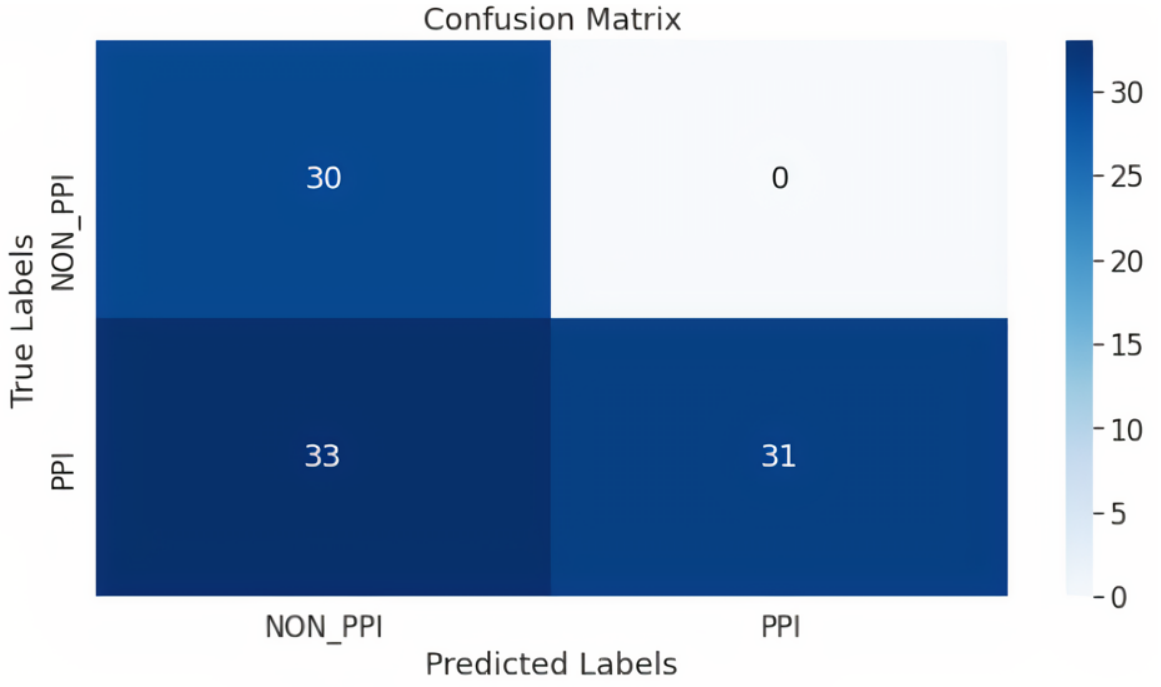


Figure 7.2. Confusion matrix of our model for Host-Pathogen Interaction Extraction.

8. CONCLUSION

The publications on the biomedical domain increase very rapidly. Automatic biomedical relation extraction systems are crucial to detect information that may be hidden in long biomedical publications. Protein-Protein interaction extraction is an important use case for biomedical relation extraction. In this thesis, we have introduced a hybrid BERT-GAN system for Protein-Protein Interaction extraction. We have designed a two-staged system so that our two models, BERT and GAN, work consecutively. As dataset, we used five benchmark PPI corpora: AIMed, BioInfer, HPRD50, IEPA, and LLL. These corpora provide ground truth entity annotations and sentence labels for both models.

In the first stage, we aimed to predict PPIs from biomedical sentences. Therefore, we have used a pre-trained biomedical language model, BioBERT, and applied transfer learning. In order to adopt the BioBERT model for the PPI extraction task, we fine-tuned it on our PPI dataset. We analyzed its performance with several experiments:

- The performance of the first stage was evaluated on five corpora separately.
- The cross-corpus evaluation was performed in order to investigate the characteristic similarities of the five corpora.
- These five corpora were combined as a single data source, and the model performance was evaluated on the combined corpus.

Our first stage results demonstrated that our BioBERT model itself achieves great success on Five Benchmarking PPI corpora. Our first stage has outperformed all previous studies on AIMed corpus, one of the most enormous and complex corpus, and we obtained a state-of-the-art F1-score with a relative improvement of 2.1%. Furthermore, we achieved comparable results compared to previous studies on BioInfer, HPRD50 and IEPA corpora. On the other hand, we obtained a lower F1-score on the LLL corpus. The reason would be the size of the LLL corpus. It is the smallest cor-

pus among five corpora, which has 300 sentences, and other kernel-based approaches and sentence representations could better handle the LLL corpus's characteristics than BioBERT.

Our cross-corpus evaluation on the first stage suggested that the performance may increase when the BioBERT is trained on larger datasets. When we trained our BioBERT model on a large BioInfer corpus and tested it on small HPRD50, IEPA and LLL corpora, we obtained promising results. AIMed, on the other hand, failed on smaller corpora contrary to expectation. The reason was not apparent, and further analysis on AIMed dataset may be required as future work. Furthermore, when BioBERT was trained on smaller corpora and tested on the AIMed and BioInfer, it has failed as expected.

Next, we evaluated our first stage on the combined corpus. Our experiments revealed that when the model is trained on the combined corpus, the performance increases on the smaller corpora: HPRD50 and LLL, and decreases on larger corpora: AIMed and BioInfer. We can conclude that while corpus size has a positive effect on the performance of our BioBERT model on small datasets, inter-corpus features can degrade the performance in already large datasets.

In the second stage, we aimed to increase the performance of the first stage by applying an adversarial process between two generative and discriminator models. The training datasets of five corpora with ground truth labels and the test sentences with predicted labels of the BioBERT model were together used. We assumed that the positive sentences of this dataset mainly consist of true positives since the majority of the sentences are coming from ground truth training dataset and most of the BioBERT predictions are correct, and there exist some false positives coming from BioBERT predictions. At the end of the adversarial learning phase, our generator captures the true positive data distribution by the feed-backs of the discriminator and assigns low probabilities to the false-positive sentences. We defined a probability threshold and changed the labels of some test sentences to negative, in which our generator has

assigned low probabilities, but BioBERT has annotated positive. In order to evaluate whether the second stage makes a statistically significant improvement on the first stage, we applied McNemar's significance test. We evaluated the performance of the second stage in two cases;

- when the BioBERT models trained on separate corpora were used on the first stage.
- when the BioBERT model trained on the combined corpus was used on the first stage.

Our second stage results revealed that the GAN model could not significantly improve the first stage performance when the first stage model was powerful. The second stage results were nearly equal to the first stage results when the BioBERT model was trained on each separate corpus. According to McNemar's test, we could not reject the alternative hypothesis that the second model is statistically different from the first model, so we can conclude that the second stage did not make a significant impact in this particular case.

On the other hand, when we trained our BioBERT model on combined corpus and ran our second stage on top of the first stage predictions, we have obtained significant improvements with the GAN model on larger corpora: AIMed and BioInfer. However, the second stage contribution was not significant on small corpora: HPRD50, IEPA, and LLL. Considering the first stage results on the combined corpus and the second stage results together, we have achieved three conclusions:

- Our BioBERT model had relatively worse results when trained on combined corpus and tested on AIMed and BioInfer than when it was trained on these corpora separately. Therefore, its performance may decrease when it is trained on a huge general PPI corpus.
- When the first stage's model performance is slightly low, our GAN model finds an open point to improve the first stage significantly on large corpora.

- Regardless of the success of our first-stage model, our GAN model does not make a significant impact on small datasets.

Lastly, we applied our system to a case study for Host-Pathogen Interaction extraction on COVID-19 dataset. First, we have obtained COVID-19 related candidate sentences from PubMed articles using a biomedical literature mining tool, SciMiner. After some preprocessing steps, we filtered sentences having at least one host and pathogen entities. Then, we have manually annotated interacting pairs and obtained a ground truth HPI dataset for COVID-19. The results revealed that our model is capable of handling different biomedical relation extraction tasks. Our first stage has achieved a 65.8% F1 score and 100% precision score for the HPI extraction task. The second stage evaluation was excluded due to two reasons; (i) the curated dataset is too small, and (ii) no false positives were obtained on the first stage, so there was no improvement point.

In the light of these conclusions, we can claim that our hybrid model is well fitted for general PPI corpora and can be used as a successful PPI extraction system. As future work, we want to mine a wide range of biomedical publications using the SciMiner tool and run our hybrid system on obtained candidate sentences. We will manually evaluate the performance of our system on a random subset of predictions. In case of obtaining good results as if we expect, we will provide an updated open-source PPI database for researchers.

REFERENCES

1. Phizicky, E. M. and S. Fields, "Protein-Protein Interactions: Methods for Detection and Analysis", *Microbiological Reviews*, Vol. 59, No. 1, pp. 94–123, Mar. 1995.

2. Kuzmanov, U. and A. Emili, "Protein-Protein Interaction Networks: Probing Disease Mechanisms Using Model Systems", *Genome Medicine*, Vol. 5, No. 4, p. 37, 2013.

3. PubMed, *PubMed Home Page*, 2021, `https://pubmed.ncbi.nlm.nih.gov`, accessed in May 2021.

4. U.S. National Library of Medicine, *Citations Added to MEDLINE® by Fiscal Year*, 2021, `https://www.nlm.nih.gov/bsd/stats/cit_added.html`, accessed in May 2021.

5. Jensen, L. J., M. Kuhn, M. Stark, S. Chaffron, C. Creevey, J. Muller, T. Doerks, P. Julien, A. Roth, M. Simonovic, P. Bork and C. von Mering, "STRING 8–A Global View on Proteins and Their Functional Interactions in 630 Organisms", *Nucleic Acids Research*, Vol. 37, No. Database, pp. D412–D416, Jan. 2009.

6. Stark, C., "BioGRID: A General Repository for Interaction Datasets", *Nucleic Acids Research*, Vol. 34, No. 90001, pp. D535–D539, Jan. 2006.

7. Hermjakob, H., "IntAct: An Open Source Molecular Interaction Database", *Nucleic Acids Research*, Vol. 32, No. 90001, pp. 452D–455, Jan. 2004.

8. Bader, G. D., "BIND: The Biomolecular Interaction Network Database", *Nucleic Acids Research*, Vol. 31, No. 1, pp. 248–250, Jan. 2003.

9. Alonso-López, D., M. A. Gutiérrez, K. P. Lopes, C. Prieto, R. Santamaría and J. D. L. Rivas, "APID Interactomes: Providing Proteome-based Interactomes with

Controlled Quality for Multiple Species and Derived Networks", *Nucleic Acids Research*, Vol. 44, No. W1, pp. W529–W535, Apr. 2016.

10. Salwinski, L., "The Database of Interacting Proteins: 2004 Update", *Nucleic Acids Research*, Vol. 32, No. 90001, pp. 449D–451, Jan. 2004.

11. Erkan, G., A. Özgür and D. R. Radev, "Semi-Supervised Classification for Extracting Protein Interaction Sentences using Dependency Parsing", *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 228–237, Association for Computational Linguistics, Prague, Czech Republic, Jun. 2007.

12. Lee, J., W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So and J. Kang, "BioBERT: A Pre-trained Biomedical Language Representation Model for Biomedical Text Mining", *Bioinformatics*, Sep. 2019.

13. Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, "Generative Adversarial Nets", *Advances in Neural Information Processing Systems*, Vol. 27, 2014.

14. Szklarczyk, D., A. L. Gable, K. C. Nastou, D. Lyon, R. Kirsch, S. Pyysalo, N. T. Doncheva, M. Legeay, T. Fang, P. Bork, L. J. Jensen and C. von Mering, "The STRING Database in 2021: Customizable Protein–Protein Networks, and Functional Characterization of User-uploaded Gene/Measurement Sets", *Nucleic Acids Research*, Vol. 49, No. D1, pp. D605–D612, Nov. 2020.

15. STRING, *STRING Database — Statistics*, 2021, `https://string-db.org/cgi/about?footer_active_subpage=statistics`, accessed in May 2021.

16. BioGRID, Database of Protein, Genetic and Chemical Interactions, *BioGRID Database Statistics*, 2021,

https://wiki.thebiogrid.org/doku.php/statistics, accessed in May 2021.

17. IntAct Molecular Interaction Database, *Statistics, Total Numbers*, 2021, https://www.ebi.ac.uk/intact/about/statistics?conversationContext=2, accessed in May 2021.

18. Benson, D. A., M. Cavanaugh, K. Clark, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell and E. W. Sayers, "GenBank", *Nucleic Acids Research*, Vol. 41, No. D1, pp. D36–D42, Nov. 2012.

19. Alfarano, C., "The Biomolecular Interaction Network Database and Related Tools 2005 Update", *Nucleic Acids Research*, Vol. 33, No. Database issue, pp. D418–D424, Dec. 2004.

20. Alonso-López, D., F. J. Campos-Laborie, M. A. Gutiérrez, L. Lambourne, M. A. Calderwood, M. Vidal and J. D. L. Rivas, "APID Database: Redefining Protein–Protein Interaction Experimental Evidences and Binary Interactomes", *Database*, Vol. 2019, Jan. 2019.

21. Wu, C. H., "The Protein Information Resource", *Nucleic Acids Research*, Vol. 31, No. 1, pp. 345–347, Jan. 2003.

22. Bairoch, A., "The SWISS-PROT Protein Sequence Data Bank and Its New Supplement TREMBL", *Nucleic Acids Research*, Vol. 24, No. 1, pp. 21–25, Jan. 1996.

23. Mikolov, T., K. Chen, G. Corrado and J. Dean, "Efficient estimation of word representations in vector space", *arXiv preprint arXiv:1301.3781*, 2013.

24. Mikolov, T., W. tau Yih and G. Zweig, "Linguistic Regularities in Continuous Space Word Representations", *HLT-NAACL*, 2013.

25. Mikolov, T., M. Karafiát, L. Burget, J. Cernocký and S. Khudanpur, "Recurrent

Neural Network Based Language Model", T. Kobayashi, K. Hirose and S. Nakamura (Editors), *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pp. 1045–1048, ISCA, 2010.

26. Bengio, Y., R. Ducharme, P. Vincent and C. Janvin, "A Neural Probabilistic Language Model", *J. Mach. Learn. Res.*, Vol. 3, No. null, p. 1137–1155, Mar. 2003.

27. LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition", *Neural Computation*, Vol. 1, No. 4, pp. 541–551, 12 1989.

28. Goodfellow, I., Y. Bengio and A. Courville, *Deep Learning*, MIT Press, 2016, `http://www.deeplearningbook.org`.

29. Wu, J., "Introduction to Convolutional Neural Networks", *National Key Lab for Novel Software Technology. Nanjing University. China*, Vol. 5, p. 23, 2017.

30. Zhou and Chellappa, "Computation of Optical Flow Using a Neural Network", *IEEE International Conference on Neural Networks*, IEEE, 1988.

31. Yani, M., M. B. I. S, Si. and M. C. S. S.T., "Application of Transfer Learning Using Convolutional Neural Network Method for Early Detection of Terry's Nail", *Journal of Physics: Conference Series*, Vol. 1201, p. 012052, May 2019.

32. Devlin, J., M.-W. Chang, K. Lee and K. Toutanova, "Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding", *arXiv preprint arXiv:1810.04805*, 2018.

33. Mintz, M., S. Bills, R. Snow and D. Jurafsky, "Distant Supervision for Relation Extraction without Labeled Data", *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 1003–1011, Association for Computational

Linguistics, Suntec, Singapore, Aug. 2009.

34. Roth, B., T. Barth, M. Wiegand and D. Klakow, "A Survey of Noise Reduction Methods for Distant Supervision", *Proceedings of the 2013 Workshop on Automated Knowledge Base Construction - AKBC '13*, ACM Press, 2013.

35. Riedel, S., L. Yao and A. McCallum, "Modeling Relations and Their Mentions without Labeled Text", *Machine Learning and Knowledge Discovery in Databases*, pp. 148–163, Springer Berlin Heidelberg, 2010.

36. Surdeanu, M., J. Tibshirani, R. Nallapati and C. D. Manning, "Multi-instance Multi-label Learning for Relation Extraction", *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 455–465, Association for Computational Linguistics, Jeju Island, Korea, Jul. 2012.

37. Mei, S. and H. Zhu, "AdaBoost Based Multi-Instance Transfer Learning for Predicting Proteome-Wide Interactions between Salmonella and Human Proteins", *PLOS ONE*, Vol. 9, No. 10, pp. 1–12, 10 2014.

38. Zeng, D., K. Liu, Y. Chen and J. Zhao, "Distant Supervision for Relation Extraction via Piecewise Convolutional Neural Networks", *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2015.

39. Airola, A., S. Pyysalo, J. Björne, T. Pahikkala, F. Ginter and T. Salakoski, "All-paths Graph Kernel for Protein-Protein Interaction Extraction with Evaluation of Cross-corpus Learning", *BMC Bioinformatics*, Vol. 9, No. S11, Nov. 2008.

40. Miwa, M., R. Sætre, Y. Miyao and J. Tsujii, "Protein–Protein Interaction Extraction by Leveraging Multiple Kernels and Parsers", *International Journal of Medical Informatics*, Vol. 78, No. 12, pp. e39–e46, Dec. 2009.

41. Palaga, P., "Extracting Relations from Biomedical Texts Using Syntactic Information", *Mémoire de DEA, Technische Universität Berlin*, Vol. 138, 2009.

42. Kim, S., J. Yoon, J. Yang and S. Park, "Walk-weighted Subsequence Kernels for Protein-Protein Interaction Extraction", *BMC Bioinformatics*, Vol. 11, No. 1, Feb. 2010.

43. Zhang, Y., H. Lin, Z. Yang and Y. Li, "Neighborhood Hash Graph Kernel for Protein–Protein Interaction Extraction", *Journal of Biomedical Informatics*, Vol. 44, No. 6, pp. 1086–1092, Dec. 2011.

44. Peng, Y., S. Gupta, C. Wu and V. Shanker, "An Extended Dependency Graph for Relation Extraction in Biomedical Texts", *Proceedings of BioNLP 15*, Association for Computational Linguistics, 2015.

45. Chang, Y.-C., C.-H. Chu, Y.-C. Su, C. C. Chen and W.-L. Hsu, "PIPE: A Protein–Protein Interaction Passage Extraction Module for BioCreative Challenge", *Database*, Vol. 2016, Aug. 2016.

46. Kim, S., R. Islamaj Doğan, A. Chatr-Aryamontri, C. S. Chang, R. Oughtred, J. Rust, R. Batista-Navarro, J. Carter, S. Ananiadou, S. Matos, A. Santos, D. Campos, J. L. Oliveira, O. Singh, J. Jonnagaddala, H.-J. Dai, E. C.-Y. Su, Y.-C. Chang, Y.-C. Su, C.-H. Chu, C. C. Chen, W.-L. Hsu, Y. Peng, C. Arighi, C. H. Wu, K. Vijay-Shanker, F. Aydın, Z. M. Hüsünbeyi, A. Özgür, S.-Y. Shin, D. Kwon, K. Dolinski, M. Tyers, W. J. Wilbur and D. C. Comeau, "BioCreative V BioC Track Overview: Collaborative Biocurator Assistant Task for BioGRID", *Database*, Vol. 2016, Sep. 2016.

47. Murugesan, G., S. Abdulkadhar and J. Natarajan, "Distributed Smoothed Tree Kernel for Protein-Protein Interaction Extraction from the Biomedical Literature", *PLOS ONE*, Vol. 12, No. 11, pp. 1–14, 11 2017.

48. Zhao, Z., Z. Yang, H. Lin, J. Wang and S. Gao, "A Protein-Protein Interaction Extraction Approach based on Deep Neural Network", *International Journal of Data Mining and Bioinformatics*, Vol. 15, No. 2, p. 145, 2016.

49. Choi, S.-P., "Extraction of Protein–Protein Interactions (PPIs) from the Literature by Deep Convolutional Neural Networks with Various Feature Embeddings", *Journal of Information Science*, Vol. 44, No. 1, pp. 60–73, Nov. 2016.

50. Hua, L. and C. Quan, "A Shortest Dependency Path Based Convolutional Neural Network for Protein-Protein Relation Extraction", *BioMed Research International*, Vol. 2016, pp. 1–9, 2016.

51. Quan, C., L. Hua, X. Sun and W. Bai, "Multichannel Convolutional Neural Network for Biological Relation Extraction", *BioMed Research International*, Vol. 2016, pp. 1–10, 2016.

52. Bunescu, R., R. Ge, R. J. Kate, E. M. Marcotte, R. J. Mooney, A. K. Ramani and Y. W. Wong, "Comparative Experiments on Learning Information Extractors for Proteins and Their Interactions", *Artificial Intelligence in Medicine*, Vol. 33, No. 2, pp. 139–155, Feb. 2005.

53. Pyysalo, S., F. Ginter, J. Heimonen, J. Björne, J. Boberg, J. Järvinen and T. Salakoski, "BioInfer: A Corpus for Information Extraction in the Biomedical Domain", *BMC Bioinformatics*, Vol. 8, No. 1, Feb. 2007.

54. Peng, Y. and Z. Lu, "Deep Learning for Extracting Protein-Protein Interactions from Biomedical Literature", *BioNLP 2017*, Association for Computational Linguistics, 2017.

55. Hsieh, Y.-L., Y.-C. Chang, N.-W. Chang and W.-L. Hsu, "Identifying Protein-Protein Interactions in Biomedical Literature using Recurrent Neural Networks with Long Short-Term Memory", *Proceedings of the Eighth International Joint*

Conference on Natural Language Processing (Volume 2: Short Papers), pp. 240–245, Asian Federation of Natural Language Processing, Taipei, Taiwan, Nov. 2017.

56. Zhang, Y., H. Lin, Z. Yang, J. Wang, S. Zhang, Y. Sun and L. Yang, "A Hybrid Model Based on Neural Networks for Biomedical Relation Extraction", *Journal of Biomedical Informatics*, Vol. 81, pp. 83–92, May 2018.

57. Ahmed, M., J. Islam, M. R. Samee and R. E. Mercer, "Identifying Protein-Protein Interaction Using Tree LSTM and Structured Attention", *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, IEEE, Jan. 2019.

58. Zhang, H., R. Guan, F. Zhou, Y. Liang, Z.-H. Zhan, L. Huang and X. Feng, "Deep Residual Convolutional Neural Network for Protein-Protein Interaction Extraction", *IEEE Access*, Vol. 7, pp. 89354–89365, 2019.

59. Ding, J., D. Berleant, D. Nettleton and E. Wurtele, "Mining Medline: Abstracts, Sentences, or Phrases?", *Biocomputing 2002*, World Scientific, Dec. 2001.

60. Fundel, K., R. Kuffner and R. Zimmer, "RelEx–Relation Extraction Using Dependency Parse Trees", *Bioinformatics*, Vol. 23, No. 3, pp. 365–371, Dec. 2006.

61. Nédellec, C., "Learning Language in Logic - Genic Interaction Extraction Challenge", *4. Learning language in logic workshop (LLL05)*, Proceedings of the learning language in logic (LLL05) workshop joint to ICML'05, ACM - Association for Computing Machinery, Born, Germany, Aug. 2005.

62. Pyysalo, S., R. Sætre, J. Tsujii and T. Salakoski, "Why Biomedical Relation Extraction Results are Incomparable and What to Do About It", *Proceedings of the Third International Symposium on Semantic Mining in Biomedicine (SMBM 2008). Turku*, pp. 149–152, Citeseer, 2008.

63. Pyysalo, S., A. Airola, J. Heimonen, J. Björne, F. Ginter and T. Salakoski, "Comparative Analysis of Five Protein-Protein Interaction Corpora", *BMC Bioinfor-*

matics, Vol. 9, No. S3, Apr. 2008.

64. Wolf, T., L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz *et al.*, "Huggingface's Transformers: State-of-the-art Natural Language Processing", *arXiv preprint arXiv:1910.03771*, 2019.

65. Loshchilov, I. and F. Hutter, "Decoupled Weight Decay Regularization", *arXiv preprint arXiv:1711.05101*, 2017.

66. Qin, P., W. Xu and W. Y. Wang, "DSGAN: Generative Adversarial Training for Distant Supervision Relation Extraction", *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, 2018.

67. Zeng, D., K. Liu, S. Lai, G. Zhou and J. Zhao, "Relation Classification via Convolutional Deep Neural Network", *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 2335–2344, Dublin City University and Association for Computational Linguistics, Dublin, Ireland, Aug. 2014.

68. Zhang, Y., Q. Chen, Z. Yang, H. Lin and Z. Lu, "BioWordVec, Improving Biomedical Word Embeddings with Subword Information and MeSH", *Scientific Data*, Vol. 6, No. 1, May 2019.

69. McNemar, Q., "Note on the Sampling Error of the Difference Between Correlated Proportions or Percentages", *Psychometrika*, Vol. 12, No. 2, pp. 153–157, Jun. 1947.

70. Anderson, R. M. and R. M. May, "Population Biology of Infectious Diseases: Part I", *Nature*, Vol. 280, No. 5721, pp. 361–367, Aug. 1979.

71. Varghese, P. M., A. G. Tsolaki, H. Yasmin, A. Shastri, J. Ferluga, M. Vatish, T. Madan and U. Kishore, "Host-Pathogen Interaction in COVID-19: Pathogene-

sis, Potential Therapeutics and Vaccination Strategies", *Immunobiology*, Vol. 225, No. 6, p. 152008, Nov. 2020.

72. Lim, Y., Y. Ng, J. Tam and D. Liu, "Human Coronaviruses: A Review of Virus–Host Interactions", *Diseases*, Vol. 4, No. 4, p. 26, Jul. 2016.

73. Hur, J., A. D. Schuyler, D. J. States and E. L. Feldman, "SciMiner: Web-based Literature Mining Tool for Target Identification and Functional Enrichment Analysis", *Bioinformatics*, Vol. 25, No. 6, pp. 838–840, Feb. 2009.