# LARGE LANGUAGE MODELS FOR BIOMEDICAL RELATION EXTRACTION AND EXPLANATION

by

Nur Bengisu Çam

B.S., Computer Science & Engineering, Hacettepe University, 2020

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

2024

# ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my mentor, Associate Professor Arzucan Özgür, for her support and guidance throughout my Master's thesis. Her valuable vision has helped me a lot in my academic journey. I am honored to have had the opportunity to work with her.

I am also grateful that I had the opportunity to work with Professor Junguk Hur and Professor Yongqunh He. Their knowledge enlightened me during this time. It was also very helpful for me to discuss the process with Jie Zheng from the University of Michigan and Hasin Rehana from the University of North Dakota. Thank you all.

Words cannot express my gratitude to my parents Serap and Savaş, who have loved, supported and guided me at every stage of my life. I would also like to express my love for my sister Bilge Sena, who is always there for me.

Finally, I would like to thank my friends and colleagues for their existence and support.

# ABSTRACT

## LARGE LANGUAGE MODELS FOR BIOMEDICAL RELATION EXTRACTION AND EXPLANATION

Due to the rapid growth of biomedical literature, the extraction of useful information from biomedical texts using natural language processing (NLP) and text mining techniques is in high demand. One of the most important text mining tasks is relation extraction (RE), which involves identifying relationships between entities. Inspired by the success of Large Language Models (LLMs) in various NLP tasks, we propose an LLM-based approach to extract and explain protein-protein interactions (PPIs) from biomedical texts. For relation extraction, we fine-tuned the masked language models BioBERT, SciBERT and PubMedBERT on five PPI benchmark datasets and showed that they can effectively identify PPIs mentioned in a given text. Next, we developed a novel approach to further analyze the extracted PPIs using causal LLMs. We applied different learning strategies, namely in-context learning and parameter-efficient instruction fine-tuning for the Llama-2 chat models, to identify keywords in the text that reveal an interaction between a protein pair. Our results show that parameter-efficient fine-tuning leads to a performance gain even when the domain is new. The smaller fine-tuned models outperformed the zero-shot performance of much larger models. Our study suggests that a pipeline consisting of a masked LLM for relation extraction and a causal LLM for relation explanation can be an effective strategy for building a PPI analyser in real-world scenarios.

# ÖZET

## BİYOMEDİKAL METİNLERDE İLİŞKİ ÇIKARIMI VE AÇIKLANMASI İÇİN BÜYÜK DİL MODELLERİ

Biyomedikal literatürün hızlı gelişmesi nedeniyle, doğal dil işleme ve metin madenciliği teknikleri kullanılarak biyomedikal metinlerden yararlı bilgilerin çıkarılması büyük talep görmektedir. En önemli metin madenciliği görevlerinden biri, varlıklar arasındaki ilişkilerin tanımlanmasını içeren ilişki çıkarmadır. Büyük Dil Modellerinin (BDM) çeşitli doğal dil işleme görevlerindeki başarısından esinlenerek, biyomedikal metinlerden protein-protein etkileşimlerini çıkarmak ve bu etkileşimleri açıklamak için BDM tabanlı bir yaklaşım kullanılmasını öneriyoruz. İlişki çıkarımı için BioBERT, SciBERT ve PubMedBERT maskeli dil modellerine beş farklı kıyaslama veri kümesinde ince ayar yaptık ve bunların belirli bir metinde bahsedilen protein-protein etkileşimlerini etkili bir şekilde tanımlayabildiklerini gösterdik. Daha sonra, nedensel BDM'leri kullanarak çıkarılan protein-protein ilişkilerini daha ayrıntılı analiz etmek için yeni bir yaklaşım geliştirdik. Bir protein çifti arasındaki etkileşimi ortaya çıkaran metindeki anahtar kelimeleri belirlemek için, Llama-2 sohbet modellerine bağlam içi öğrenme ve parametre verimli talimat ince ayarı gibi farklı öğrenme stratejileri uyguladık. Sonuçlarımız, parametre açısından verimli ince ayarın, yeni bir eğitim alanında bile model performansında artış sağladığını gösteriyor. Daha küçük ince ayarlı modeller, çok daha büyük modellerin sıfır atış performansından daha iyi performans gösterdi. Çalışmamız, ilişki çıkarımı için maskeli ayarla ilişki çıkarımı ve ilişki açıklaması için nedensel BDM'den oluşan sistemin, gerçek dünya senaryolarında protein-protein ilişkilerinin analizinde kullanmak için etkili bir strateji olabileceğini öne sürüyor.

# TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . . . iii
ABSTRACT . . . . . iv
ÖZET . . . . . v
LIST OF FIGURES . . . . . viii
LIST OF TABLES . . . . . ix
LIST OF SYMBOLS . . . . . xi
LIST OF ACRONYMS/ABBREVIATIONS . . . . . xii
1. INTRODUCTION . . . . . 1
1.1. Contributions . . . . . 2
1.2. Outline . . . . . 3
2. RELATED WORK . . . . . 4
2.1. Relation Extraction . . . . . 4
2.2. Biomedical Relation Extraction . . . . . 6
2.3. Large Language Models for Biomedical Text Mining . . . . . 10
3. BACKGROUND THEORY . . . . . 13
3.1. Transformer Architecture . . . . . 13
3.2. Masked Language Modelling . . . . . 13
3.2.1. BERT-Based Models . . . . . 13
3.3. Causal Language Modelling . . . . . 15
3.3.1. GPT . . . . . 15
3.3.2. Llama . . . . . 16
3.4. Adapter-Based Fine-Tuning LLMs . . . . . 18
3.5. Low-Rank Adaptation of LLMs (LoRA) . . . . . 20
3.6. In-Context Learning . . . . . 20
4. DATASETS . . . . . 23
4.1. LLL . . . . . 23
4.2. LLL with INO . . . . . 23
4.3. HPRD50 . . . . . 24

4.4. IEPA . . . . . 24
4.5. BioInfer . . . . . 25
4.6. AIMed . . . . . 25
4.7. Analysis of the Datasets . . . . . 26
5. METHODOLOGY . . . . . 27
5.1. Processing the Datasets . . . . . 27
5.1.1. Protein Masking Setting . . . . . 28
5.1.2. Protein Tagging Setting . . . . . 29
5.2. Protein-Protein Relation Classifications . . . . . 32
5.2.1. Fine-Tuning BERT-based Models . . . . . 32
5.3. Explaining the Relations by Identifying Interaction Keywords . . . . . 34
5.3.1. In-Context Learning . . . . . 35
5.3.1.1. Zero-Shot Setting . . . . . 35
5.3.1.2. Few-Shot Setting . . . . . 36
5.3.2. Parameter-Efficient Instruction Fine-Tuning . . . . . 37
5.4. Evaluation Metrics . . . . . 38
6. EXPERIMENTS AND RESULTS . . . . . 43
6.1. Relation Classification Results . . . . . 43
6.1.1. Results on Protein Masking Setting . . . . . 43
6.1.2. Results on Protein Tagging Setting . . . . . 46
6.2. Relation Keyword Identification Results . . . . . 48
6.2.1. Zero-shot In-Context Learning Results . . . . . 49
6.2.2. Few-Shot In-Context Learning Results . . . . . 50
6.2.3. Parameter-Efficient Instruction Fine-Tuning Results . . . . . 52
7. CONCLUSION AND FUTURE WORK . . . . . 55
REFERENCES . . . . . 57
APPENDIX A: KEYWORD PREDICTIONS OF LLAMA-2-13B-CHAT MODEL 69

# LIST OF FIGURES

Figure 3.1. Adapter-tuning an LLM for different tasks. . . . . 22

Figure 5.1. **(a)** Distribution of word counts in the LLL. **(b)** Distribution of word counts in the HPRD50. **(c)** Distribution of word counts in the IEPA. **(d)** Distribution of word counts in the BIOINFER. **(e)** Distribution of word counts in the AIMED. . . . . 30

Figure 5.2. **(a)** Distribution of token counts in the LLL. **(b)** Distribution of token counts in the HPRD50. **(c)** Distribution of token counts in the IEPA. **(d)** Distribution of token counts in the BIOINFER. **(e)** Distribution of token counts in the AIMED. . . . . 31

Figure 5.3. Zero-shot prompt example. . . . . 40

Figure 5.4. Few-shot prompt example. . . . . 41

Figure 5.5. Instruction fine-tuning example. . . . . 42

# LIST OF TABLES

Table 4.1. Dataset details. . . . . 26

Table 5.1. Protein masking setting examples from LLL. . . . . 28

Table 5.2. Protein tagging setting examples from LLL. . . . . 29

Table 5.3. Specifications of BERT-based models. . . . . 33

Table 6.1. Hyper-parameter setting on LLL, IEPA and HPRD50. . . . . 44

Table 6.2. Performance of BERT models on the LLL, HPRD50 and IEPA. . . 44

Table 6.3. Macro-averaged scores on LLL, HPRD50 and IEPA. . . . . 45

Table 6.4. Hyper-parameter setting for BioInfer and AIMed. . . . . 45

Table 6.5. Performance of BERT-based models on the BioInfer and AIMed. . 46

Table 6.6. Macro-averaged scores on BioInfer and AIMed. . . . . 46

Table 6.7. Performance of BERT-based models on tagged LLL, HPRD50, and IEPA. . . . . 47

Table 6.8. Macro-averaged results on tagged LLL, HPRD50 and IEPA. . . . . 47

Table 6.9. INO keywords and IDs. . . . . 48

Table 6.10. Token-wise metric calculation. . . . . 49

Table 6.11. Zero-shot prompting inference scores. . . . . 50

Table 6.12. Few-shot prompting inference scores with the first set of context examples. . . . . 51

Table 6.13. Few-shot prompting inference scores with the second set of context examples. . . . . 52

Table 6.14. Few-shot prompting inference scores with the third set of context examples. . . . . 52

Table 6.15. Averaged few-shot prompting inference scores. . . . . 52

Table 6.16. Llama-2 fine-tuning hyper-parameters. . . . . 54

Table 6.17. Llama-2 fine-tune result. . . . . 54

Table A.1. Predicted keywords for LLL INO test split portion one. . . . . 69

Table A.2. Predicted keywords for LLL INO test split portion two. . . . . 70

Table A.3. Predicted keywords for LLL INO test split portion three. . . . . 71

# LIST OF SYMBOLS

| | |
|---|---|
| a | An input vector |
| b | Bias |
| c | Bias |
| $K$ | Number of classes in a multi-class classifier |
| $V$ | Weight matrix representation |
| $W$ | Weight matrix representation |
| $\overrightarrow{x}$ | The input vector |
| y | True labels as a vector |
| $\hat{\text{y}}$ | Predicted labels as a vector |
| $y_i$ | $i$th element of true labels vector |
| $\hat{y}_i$ | $i$th element of predicted labels vector |
| | |
| $\sigma$ | Sigmoid function |

# LIST OF ACRONYMS/ABBREVIATIONS

| | |
|---|---|
| ADVBERT | Adversarial Bidirectional Encoder Representations from Transformers |
| A-GCN | Attentive Graph Convolutional Network |
| AIMed | Artifical Intelligence in Medicine |
| BERT | Bidirectional Encoder Representations from Transformers |
| BioBERT | Bidirectional Encoder Representations from Transformers for Biomedical Text Mining |
| BLURB | Biomedical Language Understanding Reasoning Benchmark |
| BiLSTM | Bidirectional Long Short-Term Memory |
| BioInfer | Bio Information Extraction Resource |
| ChatGPT | Generative Pre-trained Transformer for Chat |
| ChemProt | Text Mining Chemical-protein Interactions Dataset |
| CNN | Convolutional Neural Network |
| DCNN | Deep Convolutional Neural Network |
| DDI | Drug-Drug Interaction |
| DDN | Deep Neural Network |
| GAD | Gene Associations Database |
| GAtt | Ghost Attention |
| GCN | Graph Convolutional Network |
| GLU | Gated Linear Unit |
| GPT | Generative Pre-trained Transformer |
| HPRD50 | Human Protein Reference Database |
| IEPA | Interaction Extraction Performance Assessment |
| INO | Interaction Network Ontology |
| LAMBADA | Language Modeling Broadened to Account for Discourse Aspects |
| LBERT | Lexically aware Transformer-based Bidirectional Encoder Representation from Transformers |
| Llama | Large Language Model Meta AI |

| | |
|---|---|
| Llama-2-7b-chat | Llama-2 model with seven billion parameters |
| Llama-2-70b-chat | Llama-2 model with seventy billion parameters |
| Llama-2-13b-chat | Llama-2 model with thirteen billion parameters |
| LLL | Learning Language in Logic |
| LLL05 | Learning Language in Logic 2005 |
| LLM | Large Language Model |
| LoRA | Low-Rank Adaptation |
| LSTM | Long Short-Term Memory |
| McDepCNN | Multi-channel Dependency-based Convolutional Neural Network |
| NLP | Natural language processing |
| PEFT | Parameter-Efficient Fine-Tuning |
| PPI | Protein-protein interaction |
| PMC | PubMed Central |
| PubMedBERT | Bidirectional Encoder Representations from Transformers for PubMed |
| Q&A | Question and Answering |
| RE | Relation Extraction |
| RLHF | Reinforcement Learning from Human Feedback |
| RMSNorm | Root Mean Square Layer Normalization |
| RNN | Recurrent Neural Network |
| RoBERTA | A Robustly Optimized BERT Pretraining Approach |
| SciBERT | Bidirectional Encoder Representations from Transformers for Scientific Text |
| SemEval | International Workshop on Semantic Evaluation |
| SVM | Support Vector Machine |
| SwiGLU | Swish Gated Linear Unit |
| TACRED | TAC Relation Extraction Dataset |

# 1. INTRODUCTION

Natural language processing (NLP) is a key area of computer science that is increasingly being applied in various fields, including biomedical research, which has experienced massive growth in recent years. Relation extraction, a widely used NLP method, aims to identify relationships between two or more entities in biomedical texts to support the automatic analysis of documents in this field [1].

Biomedical relation extraction focuses on detecting high-quality relations in biomedical texts, where the quality of relations is often measured using F-score, precision and recall metrics. The entities that form a relationship pair in the biomedical domain are usually genes, proteins, drugs, cells and diseases. There are several categories for relationships. For example, binary relations such as protein-protein interactions (PPI) and complex relations such as molecular events, where these events represent changes in molecules that occur during protein binding or RNA production [2].

The detection of protein-protein interactions is an important and widely studied topic in biomedical NLP [1]. This is because protein-protein interactions (PPIs) play a crucial role in various biological functions, including DNA replication, transcription, signaling pathways, cell metabolism and the conversion of genotype to phenotype. Studying these interactions provides valuable insights into the underlying biological processes, signaling pathways and networks associated with both healthy and diseased conditions. This helps researchers to search better and faster. The relationships can also be useful for gaining new insights into drug discovery, disease and various protein interactions. Therefore, the demand for extracting biomedical relations is increasing.

In biomedical text mining, the relations between the biomedical entities are expressed by words, which can be called as relation keywords. These relation keywords can be used to infer the type of the relation between the entity pairs. The Interaction Network Ontology (INO) is an ontology that represents the interactions in a domain.

However, when searching biomedical literature, the keywords of the relations are not carefully examined. For this reason, the gene-gene INO was created [3]. It is a helpful resource to better analyze the gene-gene or protein-protein interactions.

In this thesis, we first dealt with the detection of protein-protein interactions. We carefully fine-tuned the three BERT-based models, namely BioBERT [4], SciBERT [5] and PubMedBERT [6]. In addition, we have applied different strategies to process entity names. The entity masking setting performed better than the entity tagging setting. Our experiments shows that the BioBERT model was superior to the others in entity masking setting on LLL and HPRD50, while the PubMedBERT model was superior in BioInfer, AIMed and IEPA.

Second, our goal was to explain relations by identifying keywords for interactions. Here we have harnessed the power of causal LLMs. We chose the Llama-2 model because it is one of the most popular open-source LLMs. Our methods include contextual learning with zero-shot and few-shot settings as well as parameter-efficient instruction fine-tuning. We compared the models using these learning techniques. Our goal was to show that the models with smaller parameters, when fine-tuned even with parameter efficient techniques, can outperform the larger models. Here, the fine-tuned Llama-2-13b-chat model was superior to the Llama-2-70b-chat model with few-shot learning.

### 1.1. Contributions

Our contributions to the literature are as follows:

- We have performed a comprehensive evaluation of the fine-tuned masked language models BioBERT, SciBERT and PubMedBERT using the five benchmark PPI corpora. As far as we are aware, there is no other study that compares the performance of these BERT-based models on all these corpora.

- We have applied different entity name processing strategies where we found that protein names are not crucial for PPI.
- We have explained the relations by identifying the interaction keywords using causal LLMs. To our knowledge, the analysis of relations with LLMs is the first of its kind. We believe that this will be an initial study for future work.
- We applied different learning strategies to LLMs and concluded that parameter-efficient instruction fine-tuning of smaller LMs is superior to contextual learning without gradient updating of much larger LMs.

## 1.2. Outline

This thesis is outlined as follows:

- Chapter 2 presents the literature on the relation extraction, biomedical relation extraction and large language models on the biomedical domain.
- Chapter 3 presents the theory behind the methods used in this thesis for the detection of the protein-protein interactions and INO keywords.
- Chapter 4 explains the details of the datasets used
- Chapter 5 gives detailed information about the experiments and the results.
- Chapter 6 presents the final thoughts on the results and the explains possible future directions.

# 2. RELATED WORK

This chapter discusses some of the related studies in the literature. Here we will briefly introduce relation extraction in the general NLP domain. Next, we will focus on relation extraction studies in biomedicine. Finally, we will look for examples of large language models for the biomedical domain.

## 2.1. Relation Extraction

Since the topic of biomedical relation extraction is a specialized version of relation extraction from documents in NLP, we will present the studies on relation extraction to get a better overview of the task.

Distant Supervision [7] is one of the most commonly used techniques when manually labeled training data for a task is scarce or expensive to obtain. It uses the existing knowledge base to create new labeled examples without the need for manual labeling. This is done by matching entities with known relationships in a large corpus providing supervision to new examples. However, the examples created may not be labeled correctly. This problem of incorrect labeling often occurs with distant supervision frequently. It was found that although the entities of a relation occur in a sentence, they may not share the relation, but only the context [8]. For this reason, the researchers applied the idea of multi-instance learning and compared the performance of the models with and without multi-instance learning. According to their observations, precision and recall values increase when the multi-instance is added to their network. In previous studies [9], Wikidata or Wikipedia were used as a source of unannotated data to create the datasets. Some researchers adjusted the manually annotated data to create less erroneous samples by using distant supervision aiming to create more accurate models [10]. In addition, the datasets which are augmented with distant supervision are often used. RELX-Distant, a multilingual relationship extraction dataset, was created using the Distant Supervision technique [11].

Neural architectures such as Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) models have often been used to extract relationships. These can be grouped as feature-based neural network models. In some studies, CNN architectures were used by modifying the convolutional layer by applying a piecewise convolution operation to the hidden feature vector representation [8]. In another study, RNN and CNN models were used for open-type relation extraction problems [9]. According to their observations, the RNN models provided better results than the CNN models in most of their experiments. In another study, the idea of realistic few-shot learning was applied to many different neural architectures, including the CNN model and transformer-based models [12].

In addition to the feature-based neural network models mentioned above, the Long Short-Term Memory Networks (LSTMs) [13] and the Bi-directional Long Short-Term Memory Networks (BiLSTMs) are commonly used for relation extraction. Xu et al. [14] used shortest dependency paths in a directed graph to find the most relevant information and applied multi-channel LSTMs to find the relations on the SemEval 2010 Task 8 (SemEval) [15] Some works combined the distant supervision technique with these models. Banerjee and Tsioutsiouliklis [16] assigned confidence scores to the automatically labeled samples and applied the BiLSTM model to classify the relationships.

In the models of the Graph Convolutional Network (GCN), the sentences are represented as dependency trees. Here, the edges between the nodes have the same weight, so that the important relations cannot be captured. To solve this problem, the researchers have introduced attentive graph convolutional networks (A-GCN) to capture the relationship information with the center of gravity [17]. The proposed A-GCN model achieved the highest F1-score in SemEval 2010 Task 8 (SemEval) [15].

After the development of the transformer architecture [18], transformer-based models such as Bidirectional Encoder Representation Transformer (BERT) [19], a kind of masked language model, emerged. Previously, recurrent architectures were designed

to handle long dependencies. Transformer models, on the other hand, were developed to capture the contextual information of the text. This allows them to understand the relations between different entities from the context. The LUKE model has been applied to various NLP tasks, such as entity typing, relation classification, named entity recognition, and question answering [20]. LUKE was compared with the popular transformer-based models such as ROBERTA and other contextualized representation models such as BERT, SpanBERT and KnowBERT. The results showed that LUKE outperformed ROBERTA on all of the above problems. On the TACRED dataset, a commonly used benchmark dataset for relation extraction, KnowBERT outperforms LUKE. Similar results were obtained in the few-shot learning experiments [12]. The SpanBERT model outperformed the BERT, Roberta-base and LUKE models in only one of the experiments, while the LUKE model outperformed all other experiments.

Although the tasks of relation extraction and entity recognition are usually treated separately, some studies have focused on the joint recognition of both tasks. Xiao et al. [21] proposed a hybrid model with a transformer encoder and an LSTM model for the joint task of entity recognition and relation extraction. Feng et al. [22] used reinforcement learning with BiLSTM with attention and Tree-LSTM for entity identification and relation extraction, respectively.

## 2.2. Biomedical Relation Extraction

In the biomedical field, relationship extraction aims to identify relationships between two or more entities in biomedical texts to support the automatic analysis of documents in this field [1].

Previous approaches used classical machine learning approaches that require feature engineering. Erkan et al. [23] applied dependency parsing to the AIMed and CB datasets, resulting in parse trees. These parse trees are then fed into Support Vector Machines (SVMs) and the k-nearest-neighbor algorithm for protein-protein interaction classification. The proposed model obtained 59.62% F1-score. In another study [24],

the SVM was applied to the selected relationship pairs from the five benchmark datasets LLL, HPRD50, IEPA, BioInfer and AIMed. They reported the best F1-score of 84.1% for the LLL dataset. Kernel-based approaches were widely used for biomedical text mining. In an example study, graph kernels were used for the protein-protein interaction task [25] on five benchmark datasets.

As with relation extraction in the general domain, it is common to use feature-based neural models for biomedical relation extraction. Zhao et al. [26] a three-channel convolutional neural network for biomedical relation extraction. Their proposed system includes a named entity recognizer, a feature extractor, an auto-encoder, and a neural network model applied to the five benchmark protein-protein interaction corpora. Choi [27], used a Deep Convolutional Neural Network (DCNN) model for protein-protein interaction recognition on AIMed. Peng and Lu [28] used a multi-channel dependency-based convolutional neural network (McDepCNN) on the AIMed and BioInfer datasets. They applied a 10-fold cross-validation on these two protein-protein interaction datasets to be comparable with the previous ones by outperforming the results of the previous kernel-based methods.

The researchers fine-tuned the BioBERT and SciBERT models using the AIMed dataset [29]. It was found that the BioBERT model was superior with a 2.2% percent increase in F1- score. They processed the protein names similar to the other papers by replacing the names with the tag *@PROTEIN$* for all protein entities. In another study, BioBERT, SciBERT and PubMedBERT were applied to the AIMed corpus [30]. It has been shown that the entity-oriented representation improves the results for PPI and can therefore be promising for DDI tasks.

Li et al. [1] proposed a semantic multigranularity fusion model for protein-protein interaction classification. This model uses BioBERT [4] to extract the contextual representation of the words in the sentences as transformer embeddings. These embeddings are then fed into a CNN-based multi-channel model. Finally, the outputs of the convolutional layers are given to the max-pooling layer. The softmax function was used

to classify the raw outputs. The five benchmark protein-protein interaction corpora LLL [31], HPRD50 [32], IEPA [33], BioInfer [34] and AIMed [35] were used. The names of the entities are wrapped with the tags "<e1>", "</e1>" and "<e2>", "</e2>" for the first or second entity in a relation pair.

Warikoo et al. [36] created the Lexically aware Transformer-based Bidirectional Encoder Representation (LBERT) model with lexical embeddings and distance-driven attentional inputs. They improved classification performance on the five protein-protein interaction benchmark datasets by pre-training and fine-tuning the baseline BERT and BioBERT models. They reported that the best F1-score for the LLL dataset with the LBERT model was 86.00%, while the F1-score of the BioBERT model was 15.6% lower than the performance of LBERT.

Tang et al. [37] proposed the adversarial BERT model (ADVBERT) for the protein-protein interaction task. Since a protein name generalization approach in which protein names are replaced by PROTEIN1 and PROTEIN2 tags can be problematic in terms of the robustness of the trained model, adversarial training can solve this problem. This is done by adding adversarial values to token embeddings that cause bias and thus avoid overfitting. They reported that using adversarial perturbations of token embedding weights increases the resilience of the protein-protein interaction extraction model, especially when dealing with samples that have small differences [37]. Their proposed ADVBERT model includes a simple classification layer that uses the outputs of the final hidden BERT layer. The best F1-score for the LLL dataset was reported as 89.62%, while the best F1-score for BioInfer was reported as 90.31%.

Park et al . [38] have experimented with transformer-based architectures that use attention-based relational context information to capture the relationships between protein entities in the five benchmark protein-protein interaction corpora. They shared the unified version of these corpora as well as the interaction type annotation for some of them. Besides protein-protein interaction recognition, their transformer-based model can also work with other biomedical relationship datasets such as ChemProt [28],

GAD [39] and DDI [40]. Instead of applying a very common classification strategy with transformer-based models, namely using the [CLS] token in BERT embeddings to capture all the input information, they used entity mention and added entity start token information to the BERT embeddings. Then they used a binary classifier to recognize the relation interactions. They utilized the entity-mention attentions and the entity-start embedding of the BERT model. They achieved 89.4% F1-score for the LLL dataset, they reported the best F1-score of 92.0% for the AIMed dataset. This surpassed the protein-protein interaction detection performance of the LBERT [36] and ADVBERT [37] models from previous studies.

Roy and Mercer [41], used two different tree transformers and a heterogeneous neural graph network to obtain a better syntactic and semantic understanding of the texts. They experimented with both protein-protein interaction and drug-drug interaction (DDI) benchmark datasets. They outperformed the classification performance of previous studies on the five protein-protein interaction benchmark datasets. They reported F1-scores of 94.14%, 94.01%, 93.47%, 97.81% and 94.66% for the LLL, HPRD50, IEPA, BioInfer and AIMed datasets, respectively. As far as we know, this study has the highest scores in the protein-protein interaction literature.

In addition to biomedical text datasets, several studies have experimented with protein sequences [42, 43]. Zhang et al. [42] used an ensemble of deep neural networks (DNNs) for protein sequence datasets. At the same time, Gao et al. [43] utilized ensemble CNN architectures with residual connections on protein sequence datasets to mitigate the problem of gradient vanishing/exploding that can occur when training deep models. The research introduced a novel CNN method for predicting protein-protein interaction, which improves the accuracy of positive samples by incorporating the binding propensity of residues, thus addressing the cost and time issues of experimental methods. Meanwhile, Soleymani et al. [44] a framework that combines an auto-encoder with a deep CNN, achieving significant accuracy and efficiency in protein-protein interaction prediction.

## 2.3. Large Language Models for Biomedical Text Mining

Following the development of the transformer architecture, the era of large language models (LLMs) has begun. The LLMs are known for their large corpus pre-training, their enormous parameter size and their massive data processing capabilities for identifying connections in text elements. Therefore, the pre-training and subsequent fine-tuning of language models became popular for various NLP domains such as biomedical text mining.

By utilizing the transformer architecture, an encoder-only transformer models such as BERT (Bidirectional Encoder Representations from Transformers) have emerged [19]. BERT is an LLM trained on the general corpora. Various BERT-based models have emerged and become popular for biomedical text mining. BioBERT [4], Bidirectional Encoder Representations from Transformers for Biomedical Text Mining, is an LLM specifically pre-trained on biomedical text from PubMed articles. The model was not trained from scratch, but built on the BERT [19] model, which is a general domain LLM, through continuous training with PubMed abstracts and PubMed Central (PMC) articles. ClinicalBERT [45] further pretrained the BERT model for comparing the effect of domain-specific contextual embeddings. SciBERT [5], on the other hand, is an LLM pre-trained on scientific texts that also contain biomedical data. The studies have shown that pre-training with biomedical text shows superior performance on domain-specific tasks [4, 5] compared to pre-training with general domain information. In contrast, another study considered further pre-training as transfer learning and focused on the effects of pre-training on domain-specific sources only [6]. They reported that PubMedBERT, a biomedical domain-specific pre-trained LLM, outperformed the other LLMs on most tasks, such as biomedical named-entity recognition, biomedical relation extraction, and question answering.

Decoder-only transformer models such as the generative pre-trained transformer (GPT) is an autoregressive language model that generates human-like text through the acquisition of linguistic patterns and structures [46]. GPT is a set of language models

developed by OpenAI in 2018 based on transformer architecture [18]. Over the years, the number of parameters in the GPT models has evolved. The GPT-1 [46], GPT2 [47] and GPT-3 [48] have 117 million, 1.5 billion and 175 billion parameters respectively. OpenAI has released ChatGPT, a natural and engaging conversation tool that can produce contextually relevant responses based on text data [49]. It is available in two different variants, the GPT-3.5 and the GPT-4 architecture at the time of writing.

In our previous study [50], we evaluated the zero-shot performance of ChatGPT and compared the results with the fine-tuned BERT-based models BioBERT, SciBERT and PubMedBERT on protein-protein interaction tasks using the LLL, HPRD50 and IEPA datasets. We also worked on the prompt design for communication with ChatGPT. Our results showed that while ChatGPT performed competitively with carefully designed prompts, the BERT-based models were superior.

Another study investigated the performance of ChatGPT and fine-tuned BERT-based models in biomedical text mining and generation, including biomedical relation extraction [51]. Their results in the Biomedical Language Understanding and Reasoning Benchmark (BLURB) showed that ChatGPT performed remarkably low compared to the BERT-based models.

Llama [52] is another decoder-based LLM developed by Meta. It was trained on a very large collection of corpora from general domains. One of the main advantages of Llama over GPT models is that Llama is completely open source. This allows researchers to make further improvements. It was reported that Llama performs better than GPT-3 on almost all benchmarks [52]. In the same year, the researchers developed a newer version of Llama, which they called Llama-2 [53]. They published the 7, 13 and 70 billion parameterized models as well as the chat models. The chat models were trained on dialog-oriented corpora and methodically improved using Reinforcement Learning from Human Feedback (RLHF).

PMC-Llama [54] was developed specifically for medical applications, as the LLMs of the general domain lack medical information. Two different training approaches were used. First, the Llama model was trained with a large unsupervised dataset consisting of 4.8 million articles and 30,000 books from the biomedical domain. In order to improve the task-specific performance of the model, instruction tuning was then applied to corpora from the medical domain. According to the results, the fine-tuned PMC-Llama model with 13 billion parameters outperformed the Llama-2 model with 70 billion parameters by 13.43% and the ChatGPT model by 9.46% on Q&A benchmarks.

MEDITRON-7B and MEDITRON-70B [55] are a Llama-2-based, 7 billion and 70 billion Llama-2 LLM that was further pre-trained on carefully constructed medical corpora collected from PubMed Central and PubMed articles. It was reported that the in-context learning performance of MEDITRON-7B outperformed PMC-Llama by 10% on various biomedical benchmarks. Compared to the fine-tuned baseline models, MEDITRON-7B showed a performance gain of 5% and MEDITRON-70B showed a performance gain of 2%. It is worth noting that the code for distributed training and the dataset were made openly available.

Some studies revealed the instruction-based biomedical datasets they used to fine-tune LLMs [56]. BioInstruct is a large dataset consisting of 25,005 instructions. They evaluated the zero-shot and one-shot performances of different Llama models. The results show that both the Llama-1 and Llama-2 models tuned on BioInstruct significantly outperform the baseline models.

Although LLMs show promise in the biomedical field, their use in such a critical area without verification of responses could lead to misleading information. Some researchers investigated the opportunities and obstacles of LLMs in the biomedical field [57]. It was suggested to train a domain-specific LLM to reduce the pitfalls and misleading information [57, 58].

# 3. BACKGROUND THEORY

This chapter introduces the transformers and the models based on transformers that are experimented with in this thesis. It also describes the fine-tuning process and the methods of learning from context.

## 3.1. Transformer Architecture

The transformer architecture proposed by Vaswani et al. consists of a stack of encoder and decoder, where the encoder produces hidden representations and a decoder produces output sequences using multi-head attention that prioritizes data over inductive biases, enabling large-scale pre-training and parallelization [18]. The mechanism of self-attention enables neural networks to determine the importance of input elements, making them ideal for language translation, text classification and generation.

There are two types of language models in NLP. The first is masked language modeling and the second is causal language modeling. These categories represent the training objective of the models. In masked language modeling, the goal is to predict the masked tokens in a sequence during training. On the other hand, causal language modeling aims at the next token in a sequence. In the following Section 3.2 and Section 3.3 we will talk about the language models we have experimented with that belong to the causal and masked language modeling classes.

## 3.2. Masked Language Modelling

### 3.2.1. BERT-Based Models

BERT [19], Bidirectional Encoder Representations from Transformers, is a type of encoder-only transformer LLM that falls into the category of masked-language mod-

eling. It is trained on both masked word prediction and next sentence prediction. It uses the transformer architecture [18] by utilizing only the encoder part. Since BERT is a bidirectional model, it can attend to the tokens from both the left and right contexts in a sequence. Thus, the BERT embeddings can include the information from the entire context in a sequence, which leads to better context understanding. The ability to attend the entire context is especially useful for natural language understanding, e.g. named entity recognition, relation extraction and part-of-speech tagging.

BERT tokenizer adds special tokens to the specified input sequence. The token *[CLS]* is added at the beginning of each input sequence. This special *[CLS]* token represents the entire input sequence by capturing all the semantics. The embedding vector of this token is used as input for the classifier during classification. Another special token that BERT uses is the *[SEP]*, which comes from "separation". It is an important token that represents the separation of the input sequences. It is therefore added after each input sequence.

The BERT model can only process a fixed length of tokens in an input sequence. As the input sequences are hardly ever the same length, there is an approach to make them all the same length. The token *[PAD]* is added to the sequences that are shorter than the specified length of the input sequence. If the input sequences are longer than the specified input length, the input is truncated accordingly. Since BERT is a masked language model, 15% of the input tokens are masked during pre-training with the special token *[MASK]*. It is noteworthy that the token *[MASK]* does not appear in the fine-tuning process.

The unknown tokens are handled by the token *[UNK]*. This stands for the unseen tokens in an input sequence. However, it is not ideal to mask the unknown tokens with the *[UNK]* token. Therefore, there are coding algorithms that solve this problem of unknown tokens. The BERT tokenizer uses the WordPiece tokenization algorithm, which is a type of partial word tokenization. It represents the words as partial word units and thus makes it possible to deal with rare words in large vocabularies.

Here is an example of a padded input sequence in BERT:

> [CLS] This is the first sentence [SEP] [CLS] This is the second sentence [SEP] [PAD], [PAD], ..., [PAD].

The training objective of the BERT-based models is cross-entropy loss. The formula of the categorical cross-entropy loss is given as

$$L(\mathbf{P}, \hat{\mathbf{P}}) = -\sum_{i=1}^{N} P_i \cdot \log(\hat{P}_i), \tag{3.1}$$

where $P_i$ is the true class distribution and $\hat{P}_i$ is the predicted class distribution.

It is common to use BERT for classification tasks such as relation extraction, emotion recognition and natural language inference. The model can easily be converted into a classification model by adding a classification layer BERT. The pooled hidden output from the last layer of BERT can be used as input to the classification layer. This pooled hidden output contains the information of *[CLS]* token for each sequence in a batch. The output of the classifier then contains the raw outputs, which can be converted into probability distributions over the number of classes using the Softmax function. The formula of the Softmax function is given as

$$\text{Softmax}(\overrightarrow{x})_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}, \tag{3.2}$$

where $x$ is an input vector and $K$ is number of classes of a classifier.

## 3.3. Causal Language Modelling

### 3.3.1. GPT

GPTs, Generative Pre-trained Transformers, are a type of decoder-only transformer LLM that falls into the category of causal language modeling. They can also be categorized as autoregressive language models, where the goal is to predict the next word based on the previous words. Similar to the BERT model, a GPT model has transformer architecture [18] by utilizing only the decoder part. However, GPTs differ

from BERT in their unidirectional attention mechanism. As we have already mentioned, BERT uses the bidirectional attention mechanism to fully cover the context information. GPT, on the other hand, only uses the previous context as it specializes in predicting the next word. Due to its autoregressive nature, GPT is beneficial in text generation in NLP, e.g. text summarizing, machine translation, question answering and language modeling [46].

Unlike BERT, GPT uses a byte-pair encoding and sub-word tokenization algorithm to handle the unknown words in a text. In the byte-pair encoding algorithm, the most frequent pairs of characters are joined together until a desired vocabulary is obtained. During training, the cross-entropy loss between the predicted token and the actual token is minimized.

GPT models are characterized by their large number of parameters. GPT-1 [46] had 117 million, GPT-2 [47] had 1.5 billion, GPT-3 [48] had 175 billion parameters. Recently, GPT-4 [59] was introduced by OpenAI. However, the model parameters and architectural details were not disclosed.

### 3.3.2. Llama

At the beginning of 2023, Meta introduced a new language model called Llama-1 [52]. Llama stands for Large Language Model Meta AI. It is a completely open source LLM. Although it is usually assumed that the larger the model parameters, the better the model, it is not possible to train or run these LLMs on personal computers with limited memory. The researchers wanted the model to be accessible and useful to the entire community [52]. Therefore, in addition to the large models, they have also published comparatively smaller models. There are 7, 13, 33 and 65 billion parameterized Llama models.

Later this year, the Llama-2 [53] was introduced by Meta. They announced 7, 13 and 70 billion parameterized Llama-2 models along with the specialized chat models.

The main contribution was to create an LLM that is tuned to human preferences. This is done through Reinforcement Learning from Human Feedback (RLHF) in the fine-tuning phase for the chat models. RLHF, or Reinforcement Learning from Human Feedback, is a machine learning technique that uses human feedback to improve the self-learning efficiency of machine learning models. In this approach, reinforcement learning techniques are used to train the software and guide it to make decisions that optimize rewards and consequently improve the accuracy of results. They have also changed the attention mechanism for fine-tuning the base models with chat instructions by using Ghost Attention (GAtt). It is said that the GAtt helps the model to follow the instructions and control the dialog flow [53]. Another difference to the Llama-1 [52] model is that Llama-2 [53] was pre-trained on a 40% larger corpus, which refers to the general domain. The corpus comes from the open source material. The pre-training of the Llama-2 model was performed with 2 trillion tokens. The context length was also increased compared to the previous model. Llama-1 has 2048 tokens as context length, while Llama-2 has 4096 tokens as context length.

Last but not least, the researchers modified the Llama-1 model using Root Mean Square Layer Normalization (RMSNorm) [60] and Swish GLU (SwiGLU) [61]. RMSNorm is a regularization of the summed inputs to a neuron with the Root Mean Square function [60]. It was developed to reduce the computational effort for layer norms when training neural models. RMSNorm is given as

$$\text{RMSNorm}(\mathbf{x}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}, \tag{3.3}$$

where x is a vector with $N$ number of elements.

SwiGLU is a variant of the GLU [61] activation function. The transformer architecture [18] uses the activation function Rectified Linear Units (ReLU) [62]. This is a non-linear function in which the negative values are mapped directly to zero.

Swish [63], on the other hand, is a smooth version of the ReLU activation function as proposed. The researchers explained that Swish tends to work better for deep neural

models [63]. The formula of the Sigmoid function is given as

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{3.4}$$

and the Swish activation function is given as

$$f(x) \ = x \ \sigma(x), \tag{3.5}$$

where $x$ is an input vector.

GLU [64], Gated Linear Units, is actually a neural layer that is a product of a linear function with a non-linear function, where the non-linear function is a linear function with sigmoid. Shazeer, proposed to use the variations of GLU for the sublayers of the transformer-based models [61]. It was reported that the results obtained with GLU were better than those obtained with ReLU. The GLU is defined as

$$\mathrm{GLU}(x, W, V, b, c) = \ \sigma(xW + b) \otimes (xV + c), \tag{3.6}$$

where the learned parameters are $W \in \mathbb{R}^{k \times m \times n}$, $b \in \mathbb{R}^n$, $V \in \mathbb{R}^{k \times m \times n}$, $c \in \mathbb{R}^n$.

SwiGLU [61] is a combination of Swish and GLU, utilizing the advantages of both functions. The formula of the SwiGLU is given as

$$\mathrm{SwiGLU}(x, W, V, b, c) = \mathrm{Swish}(xW + b) \otimes (xV + c), \tag{3.7}$$

where the learned parameters are $W \in \mathbb{R}^{k \times m \times n}$, $b \in \mathbb{R}^n$, $V \in \mathbb{R}^{k \times m \times n}$, $c \in \mathbb{R}^n$.

### 3.4. Adapter-Based Fine-Tuning LLMs

In the vanilla fine-tuning approach, the weights of a pre-trained model or the new weights of the top layer added to the pre-trained model are updated based on the new corpus for each new task. However, this approach is inefficient as it requires many parameter updates. Furthermore, with the larger parameter sizes, it is not possible to fully adjust the weights of the model with vanilla fine-tuning. Therefore, an alternative to this approach was developed. Parameter-efficient transfer learning [65], is a method in which the smaller weights in the adapter layers are trained or fine-tuned for the new task. The weights of the adapters are randomly initialized and trained with the

new corpora. The original weights of the trained models are not changed, only the adapter layers are adjusted. Next, the adapter layers are added between the layers of the pre-trained model.

Consider a pre-trained model W($x$). In vanilla fine-tuning, the weights of W are updated using the new data $y$. In adapter-tuning, new adapter layers Q are trained on the new data $y$, which can be represented as Q($y$).

There are two important points about adapters. The first is the addition of a small number of adjusted parameters compared to the original model weights. This allows not only parameter efficient fine-tuning, but also allows parameter sharing. The second important point is the almost identical initialization of the adapter weights. When fine-tuning begins, the model is almost unaffected by the new weights. This is also important for stable fine-tuning. The researchers applied the adapter-tuning to BERT for 26 different NLP tasks, including machine translation, classification and question-answering [65]. According to these experiments, adapter-tuning achieved state-of-the-art performance while only a modest amount of additional parameters were trained.

The effectiveness of adapter-based fine-tuning or adapter-tuning over vanilla fine-tuning was investigated [66]. It was found that adapter tuning outperforms the classical fine-tuning approach when the new task is either multilingual or resource-poor. Furthermore, the researchers have shown that adapter tuning is more resilient against over-fitting and leads to improved learning while retaining prior knowledge. Another downside of vanilla fine-tuning is that LLMs can be subject to "catastrophic forgetting".

Parameter Efficient Fine-Tuning (PEFT) [67], is a library provided by the HuggingFace library that can be used for various LLMs or neural models. It provides various parameter-efficient training strategies, including Low-Rank Adaptation (LoRA). The details of LoRa are described in the Section3.5.

In Figure 3.1, we illustrated the parameter-tuning of an LLM with smaller adjusted weights.

## 3.5. Low-Rank Adaptation of LLMs (LoRA)

Low-rank Adaptation of LLMs (LoRA) [68], proposes to train a small number of weight matrices to adapt an LLM to different tasks. The idea and motivation are similar to the parameter-efficient transfer learning approach mentioned in Section 3.4. LoRA overcomes the inference latency of the previously proposed methods, such as parameter-efficient adaptation [65].

Hu et al. [68] hypothesized that during fine-tuning, the updated weights are of "intrinsic rank", where the significant changes can be represented with a minimum number of vectors. This leads to lower dimensional weight matrices. As it is known, the rank of a matrix expresses the number of linearly independent columns or rows. Therefore, low-rank matrices refer to matrices with a small number of independent rows or columns. LoRA enables the training of certain dense layers of a neural network. This is achieved by optimizing the low-rank decomposition matrices during adaptation. Again, the weights of the pre-trained model are frozen. Therefore, the pre-trained weights can be used for different tasks as we have shown in Figure 3.4.

The main advantage of LoRA over adapter tuning is that LoRA overcomes the inference latency that occurs when the model is run without model parallelism and the inputs are given one at a time [68].

## 3.6. In-Context Learning

LLMs are pre-trained using a large number of samples. This can make them useful for downstream tasks. However, for most tasks, from grammar correction to text generation, a specialized LLM is required to achieve the best results. This is usually done by fine-tuning a model against a task-specific corpus consisting of thousands to

hundreds of examples. Conversely, collecting a sufficient number of examples for each task is an expensive process. This can be even more difficult and not worthwhile if the downstream task has a limited scope, as the expensive data collection has to be repeated for each individual task. Therefore, fine-tuning an LLM may not be possible. As a solution to this problem, the notion of in-context learning has been proposed, which hypothesis that the LLMs are able to follow instructions due to the unsupervised training process [48].

In the in-context learning the model is prompted with a few examples to describe what is expected of the model. It does not include a gradient update of the pre-trained model. The model is conditioned on a task using the given context instructions and examples. This is referred to as " few-shot prompting", whereby the number of examples can vary between 10 and 100. If there are no examples, it is called "zero-shot" and if the number of examples is one, it is called "one-shot".

The researchers explained the prompts as instruction formats [69] that can be understood by humans. These include input-label pairs and a final example without a label, where the label is expected as a response. In their study, they found that there is not enough research on PEFT with a small number of samples. Therefore, they investigated the advantages of PEFT over in-context learning with few examples when the corpus contains a limited number of examples. It was found that PEFT is superior in contextual learning with few recordings.

The researchers applied the in-context learning approach to GPT-3 for different benchmarks [48]. They reported that the few-shot setting almost achieved state-of-the-art performance on datasets such as LAMBADA [70], HellaSwag [71] and StoryCloze [72]. Interestingly, it has been reported that GPT-3 model with the few-shot in-context learning setting outperformed state-of-the-art models which are specifically trained for machine translation for French-English and English-German language pairs.

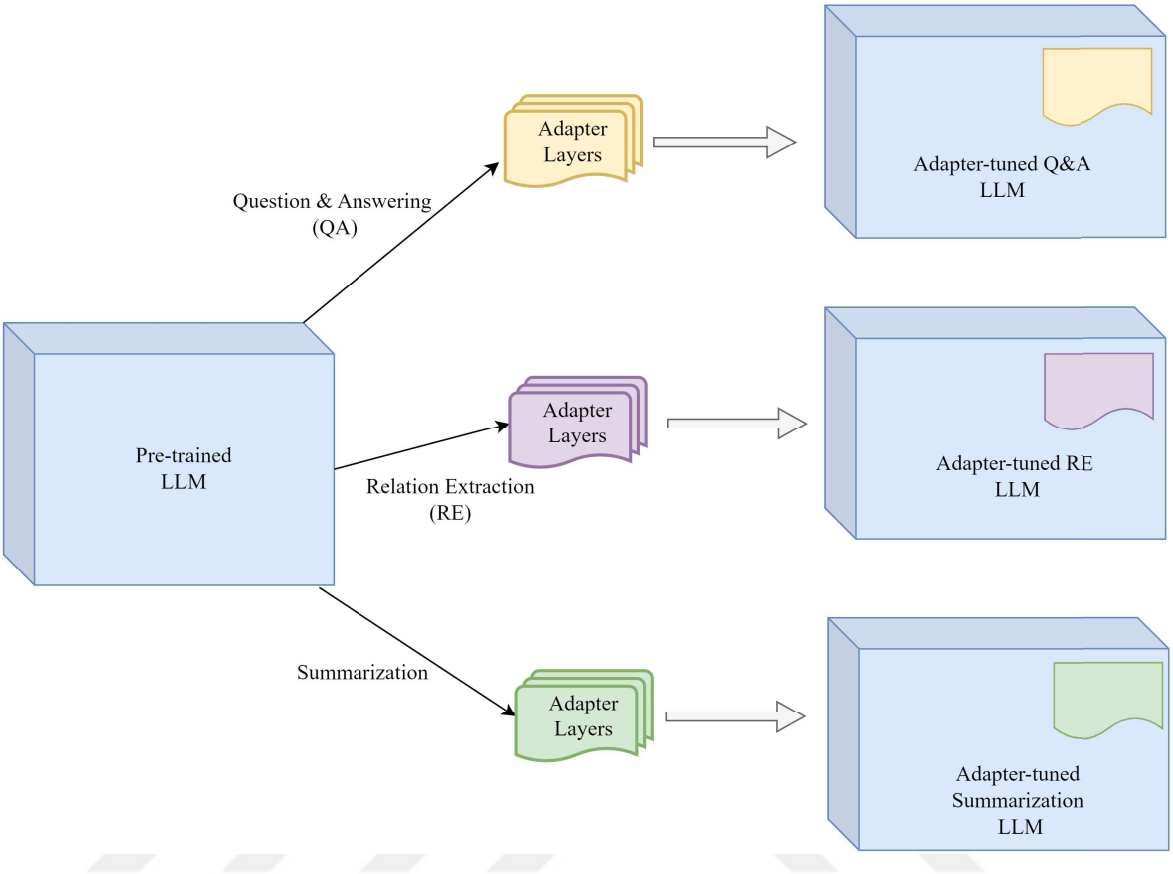


Figure 3.1. Adapter-tuning an LLM for different tasks.

# 4. DATASETS

In this chapter you will find detailed information and analysis on the datasets that are commonly used as benchmarks for protein-protein interaction detection. There is one dataset that we have experimented with for keyword identification.

## 4.1. LLL

The LLL dataset was originally introduced as part of the LLL Shared Task Challenge in 2005 (Learning Language in Logic 2005 - LLL05) [31]. It focuses on the extraction of protein and gene interactions from biological abstracts related to Bacillus subtilis. Bacillus subtilis, commonly referred to as hay bacillus or grass bacillus, is a Gram-positive bacterium that is catalase-positive. It is widely distributed in soil and can be found in the gastrointestinal tract of ruminants, humans and marine sponges. Belonging to the genus Bacillus, B. subtilis exhibits a rod-shaped morphology and has the ability to form resistant endospores that give it the ability to withstand harsh environmental conditions. This adaptability contributes to the bacterium's ability to survive in different habitats [73].

This dataset consists of 77 sentences, this dataset is enriched with 164 carefully annotated positive protein pairs, which are referred to as positive pairs in this study. In addition, it contains detailed annotations of various gene-protein interactions and specifies interaction types, including explicit interaction, binding protein-gene interaction, and regulatory family membership.

## 4.2. LLL with INO

This dataset contains the sentences from the LLL dataset (Learning Language in Logic). However, the gene interaction sentences are examined and the interaction types are analyzed using the Interaction Network Ontology (INO) [3]. The INO plays

an important role in the logical representation of biological interactions, pathways and networks. As of now, the INO ontology comprises a total of 575 terms, of which 202 terms fall into the interaction branch. Its importance is underlined by its ability to provide a structured set of ontological terms and associated keywords to support literature mining of gene-gene interactions in the biomedical literature. In many cases of biomedical relationships, two or more interaction keywords are used in combination. Therefore, the LLL dataset was annotated with one or more ontological keywords.

### 4.3. HPRD50

The HPRD50 dataset, a subset derived from the Human Protein Reference Database (HPRD), consists of 50 randomly selected abstracts and stands as a valuable resource for the study of protein-protein interactions [32]. These abstracts, identified from the Human Protein Reference Database, contribute to a dataset that is meticulously annotated for various interactions, including direct physical interactions, regulatory relationships, and modifications such as phosphorylation.

In total, the dataset comprises 145 records with 163 positive pairs and 270 negative pairs. The creation of the dataset involves a first step of automatic software annotation of protein and gene names, followed by a subsequent manual annotation of interactions involving these 25 entities. Of note is the inclusion of interaction certainties for each entity pair, deepening the dataset's ability to elucidate and analyze protein-protein interactions in the context of the selected abstracts.

### 4.4. IEPA

The Interaction Extraction Performance Assessment (IEPA) is a comprehensive collection of nearly three hundred abstracts from MEDLINE [33]. This corpus, intentionally designed for biomedical research, has been carefully curated from sentences in the abstracts that contain ten pairs of co-occurring biochemical entities. The entity pairs and their interactions within the IEPA are manually annotated, which increases

their suitability for in-depth investigations in the biomedical field. Furthermore, this dataset comprises 486 records, including 335 positive and 482 negative pairs, making it a valuable tool for evaluating the efficiency of protein-protein interaction extraction models. The diversity of abstracts and the detailed annotation of protein interactions in the IEPA dataset contribute to its importance as a resource for the advancement of research in the biomedical sciences.

## 4.5. BioInfer

The BioInfer dataset (Bio Information Extraction Resource) is a collection of abstracts of biomedical research articles from the PubMed articles. The protein pairs in the collected abstracts were extracted from the DIP (Database of Interacting Proteins). The dataset contains 1100 senteces with a total of 2662 entity relationships. The sentences had been manually annotated with great effort. In addition to the information on the entities and entity relationships, the dataset also contains the entity type ontology. The entity type ontology integrates the well-established Genia ontology [74] and focuses in particular on physical types.

## 4.6. AIMed

The AIMed corpus, a remarkable resource in the field of biomedical text mining, consists of 225 carefully selected PubMed abstracts [35]. These abstracts represent a rich source of information and contain a total of 1,987 sentences covering a wide range of biomedical topics. Within this corpus, a total of 4,075 protein mentions were identified and documented through a careful annotation process, contributing to the importance of the dataset for advancing research and applications related to protein-related interactions and functions. The comprehensiveness of the AIMed corpus, both in terms of abstract selection and detailed annotation of protein mentions, makes it a valuable dataset for various scientific purposes.

## 4.7. Analysis of the Datasets

Pyysalo et al. [75] provided a detailed analysis of the five benchmark protein-protein interaction corpora which are LLL, HPRD50, IEPA, BioInfer, and AIMed. The statistical analysis of the datasets can be found in Table 4.1. The AIMED dataset is the largest of all, while the LLL dataset is the smallest. As we can see, most of the datasets are unbalanced, with the exception of the LLL dataset.The largest datasets, BioInfer and AIMed, are the most unbalanced datasets of all.

Table 4.1. Dataset details.

| **Dataset** | **Sentences** | **Positive Pairs** | **Negative Pairs** | **Total Pairs** | **Positive to Negative Ratio** |
|---|---|---|---|---|---|
| **LLL** | 77 | 164 | 166 | 330 | 1:1.0 |
| **IEPA** | 486 | 335 | 482 | 817 | 1:1.4 |
| **HPRD50** | 145 | 163 | 270 | 433 | 1:1.6 |
| **BIOINFER** | 1100 | 2534 | 7132 | 8908 | 1:2.3 |
| **AIMED** | 1995 | 1000 | 4834 | 5775 | 1:4.8 |

# 5. METHODOLOGY

This chapter explains the techniques we used to solve the problem of identifying protein-protein interaction and relation keywords. It also mentions the evaluation metrics we used.

## 5.1. Processing the Datasets

The original datasets are available in XML format. This format has a tree-like structure and can be very informative. The XML dataset files contain multiple document entries with entity names with character offsets, protein-protein interaction pairs with entity IDs and sentences in each document. We have done some preprocessing steps to create only the sentences with and without interaction. First of all, we focused on creating the training sets, as the training set should consist of both positive and negative interaction examples. However, the XML files only contain the information on the positive interactions. To create the negative interaction sentences, we created a set of all entity-pair combinations for each sentence, separately. The positive interaction pairs are already given in the XML files. Among all the entity pair combinations, if they are not given as a protein-protein interaction in the XML file, we have marked this pair as a negative interaction. For all positive and negative interaction pairs, we created a separate sentence with a corresponding interaction label.

Each corpus contains a different set of biomedical sentences with the biomedical entities and relations. Some of the corpora contain additional information such as the entity type ontology and interaction direction information. However, in our work we only used the information about the biomedical relationships. The details of the preprocessed datasets can be found in Table 4.1. The distribution of the word counts of the datasets is given in Figure 5.1. The distribution of the token counts of the datasets after applying BERT tokenizer is given in Figure 5.2.

### 5.1.1. Protein Masking Setting

For the protein masking settings, we used the same masking setting as in our previous work [50]. We replaced the entity pair names with the keywords PROTEIN1, PROTEIN2 and PROTEIN. This approach generalized the datasets so that the proposed models using this setting also work on datasets with other protein or entity names. Here, the replacement is done by replacing the protein names with a true interaction with the placeholder PROTEIN1 and PROTEIN2 keywords. Other than the pair, the protein names in the sentence were substituted with the PROTEIN keyword. In Table 5.1, we have provided example sentences after applying the protein masking setting.The protein entities are given as an entity pair. The label column indicates whether there is a PPI or not. The original sentence is the one that occurs in the data set. The masked sentence is created after masking the given entities by the PROTEIN1 and PROTEIN2 tokens. The PROTEIN token masks the unrelated proteins in the original sentence.

Table 5.1. Protein masking setting examples from LLL.

| **Entity Pair** | **Label** | **Original Sentence** | **Masked Sentence** |
|---|---|---|---|
| SigK, ykvP | True | Both SigK and GerE were essential for ykvP expression, and this gene was transcribed from T5 of sporulation. | Both PROTEIN1 and PROTEIN were essential for PROTEIN2 expression, and this gene was transcribed from T5 of sporulation. |
| SigK, GerE | False | Both SigK and GerE were essential for ykvP expression, and this gene was transcribed from T5 of sporulation. | Both PROTEIN1 and PROTEIN2 were essential for PROTEIN expression, and this gene was transcribed from T5 of sporulation. |
| sigma(H), spo0A | True | In the shaA mutant, sigma(H)-dependent expression of spo0A and spoVG at an early stage of sporulation was sensitive to external NaCl. | In the shaA mutant, PROTEIN1 dependent expression of PROTEIN and PROTEIN2 at an early stage of sporulation was sensitive to external NaCl. |
| spo0A, spoVG | False | In the shaA mutant, sigma(H)-dependent expression of spo0A and spoVG at an early stage of sporulation was sensitive to external NaCl. | In the shaA mutant, PROTEIN dependent expression of PROTEIN1 and PROTEIN2 at an early stage of sporulation was sensitive to external NaCl. |

### 5.1.2. Protein Tagging Setting

In the protein tagging settings, we followed a similar strategy to Li et al. (2021) [1] for preprocessing the entities. Here, we enclosed the protein names in the sentences with the tags [PROTEIN1]-[/PROTEIN1] and [PROTEIN2]-[/PROTEIN2] for a relation pair. This we called tagging. We have not tagged the non-relative protein names. The reason for this is that this setting aims to indicate the position of the proteins in the interaction by leaving the protein names as they are. Since the generalization of the entity names is not a concern here, we have only processed the relation protein pairs. In Table 5.2, we have provided example sentences after applying the protein tagging setting.

Table 5.2. Protein tagging setting examples from LLL.

| **Entity Pair** | **Label** | **Original Sentence** | **Tagged Sentence** |
|---|---|---|---|
| SigK, ykvP | True | Both SigK and GerE were essential for ykvP expression, and this gene was transcribed from T5 of sporulation. | Both [PROTEIN1] SigK [/PROTEIN1] and GerE were essential for [PROTEIN2] ykvP [/PROTEIN2] expression, and this gene was transcribed from T5 of sporulation. |
| SigK, GerE | False | Both SigK and GerE were essential for ykvP expression, and this gene was transcribed from T5 of sporulation. | Both [PROTEIN1] SigK [/PROTEIN1] and [PROTEIN2] GerE [/PROTEIN2] were essential for ykvP expression, and this gene was transcribed from T5 of sporulation. |
| sigma(H), spo0A | True | In the shaA mutant, sigma(H)-dependent expression of spo0A and spoVG at an early stage of sporulation was sensitive to external NaCl. | In the shaA mutant, [PROTEIN1] sigma(H) [/PROTEIN1] -dependent expression of [PROTEIN2] spo0A [/PROTEIN2] and spoVG at an early stage of sporulation was sensitive t o external NaCl. |

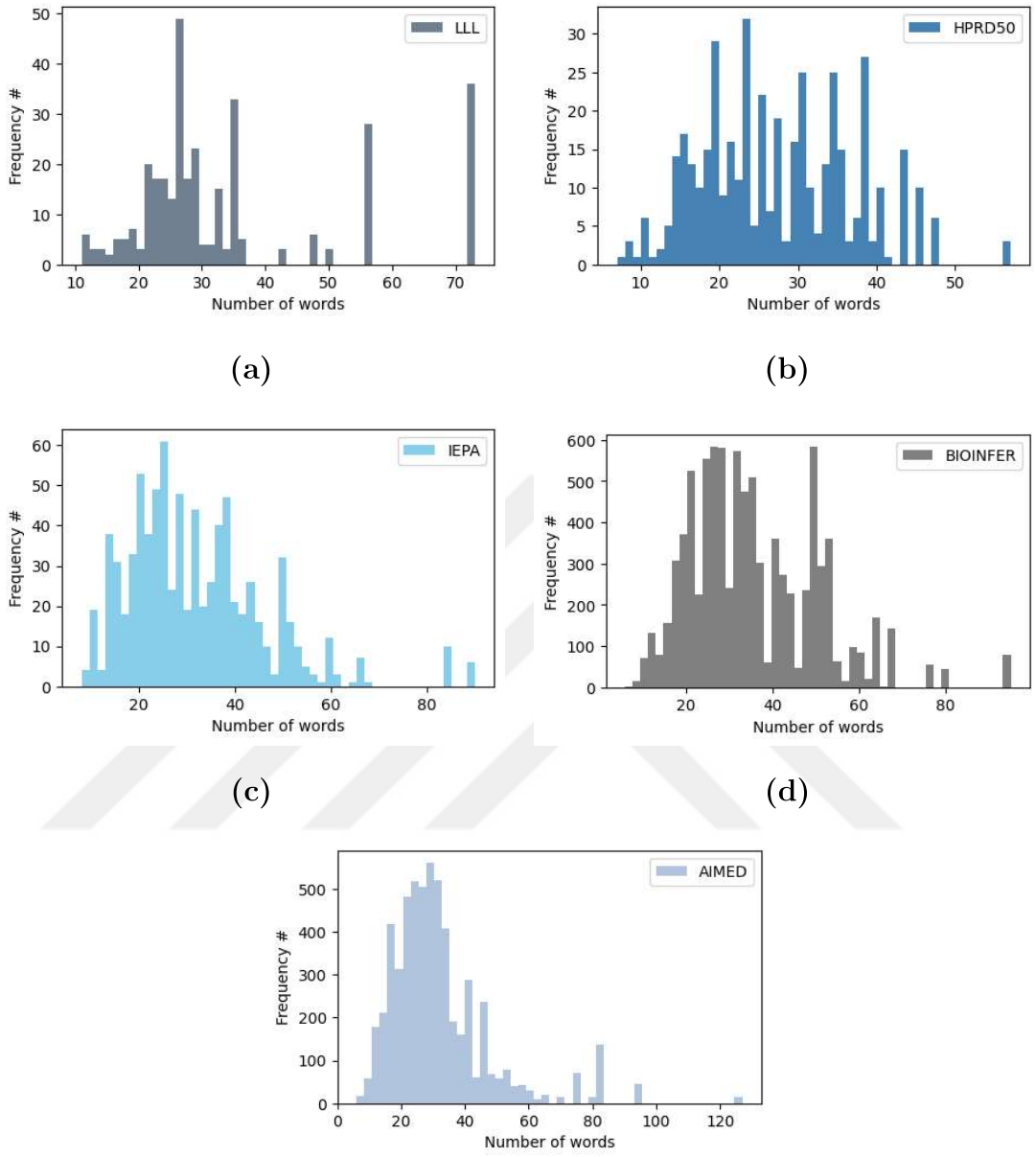


Figure 5.1. **(a)** Distribution of word counts in the LLL. **(b)** Distribution of word counts in the HPRD50. **(c)** Distribution of word counts in the IEPA. **(d)** Distribution of word counts in the BIOINFER. **(e)** Distribution of word counts in the AIMED.

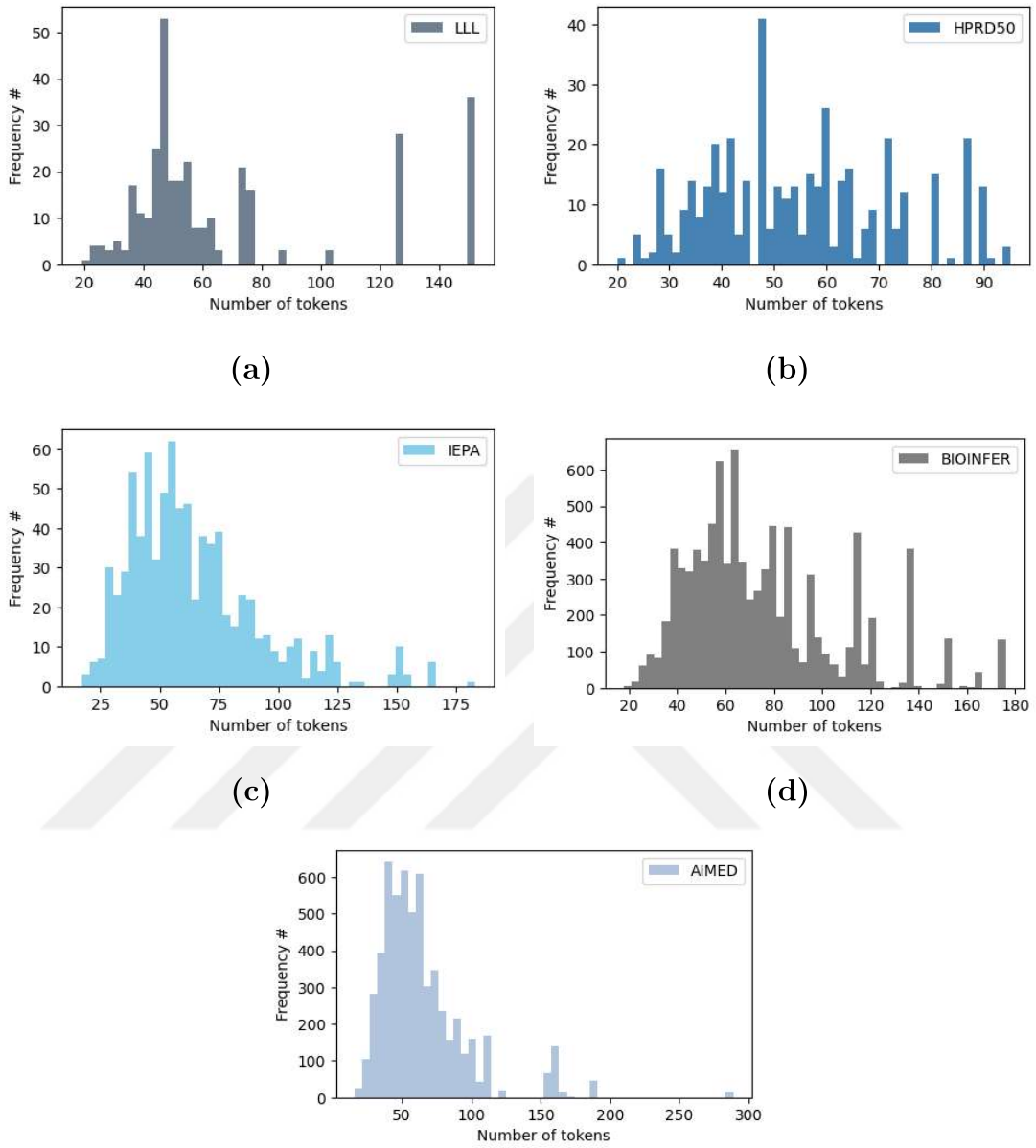


Figure 5.2. **(a)** Distribution of token counts in the LLL. **(b)** Distribution of token counts in the HPRD50. **(c)** Distribution of token counts in the IEPA. **(d)** Distribution of token counts in the BIOINFER. **(e)** Distribution of token counts in the AIMED.

## 5.2. Protein-Protein Relation Classifications

### 5.2.1. Fine-Tuning BERT-based Models

We used BERT-based models to find the protein-protein interaction relations. The BERT model is an encoder-only transformer model, as described previously. It is one of the most widely used LLMs since the day it was published. Although the training objective is different and the pre-training strategy is masked language modeling, the BERT models can be seamlessly transformed into classifiers. This transformation involves the addition of a classification layer on top of the BERT architecture. The hidden BERT layer, characterized by a 768-dimensional vector, serves as input to the classification layer. The classifier, which is tailored to the specific number of classes in the given problem, generates an output vector with the corresponding dimensions. This output vector, often referred to as raw outputs or logits, contains the predictions of the model and is crucial to the classification process. The raw outputs are then fed into the Softmax function, which calculates the probability distribution across multiple classes. The selection of the final class is based on the probabilities assigned to each class.

We experimented with three different BERT-based models in our study. Even though there are various BERT-based models that are either trained or fine-tuned on biomedical corpora, we have selected the BioBERT [4], SciBERT [5], and PubMedBERT [6] models, which are commonly used for biomedical text processing. They have the same model architecture with the base BERT model, which is an encoder architecture of transformer with 12 layers and hidden size of 768.

- BioBERT [4]: a BERT model pre-trained on PubMed abstracts and PubMed Central (PMC) full-text articles for different NLP tasks to measure performance. The initial version, BioBERT v1.0, used >200K abstracts and >270K PMC articles. An expanded version of BioBERT v1.1 was fine-tuned using >1M PubMed abstracts and was included in our study. The model was accessed from the fol-

lowing HuggingFace repository: dmis-lab/biobert-v1.1.

- SciBERT [5]: a BERT model pre-trained on random Semantic Scholar articles [76]. While pretraining with the articles, the entire text was used. The researchers created SCIVOCAB from scientific articles of the same size as BASEVOCAB, the BERT-base model's vocabulary. Uncased SCIVOCAB was used in the current study. The model was accessed from the following Hugging-Face repository: allenai/scibert_scivocab_uncased.

- PubMedBERT [6]: a BERT model trained explicitly on the BLURB (Biomedical Language Understanding  Reasoning Benchmark). In this study, we used PubMedBERT, which was trained only on the abstracts of the articles and not on the full text. We also used the "uncased" version. The model was accessed from the following HuggingFace repository: microsoft/BiomedNLP-PubMedBERT-base-uncased-abstract.

The summary of the details of the BioBERT, SciBERT and PubMedBERT models we used can be found in Table 5.3.

Table 5.3. Specifications of BERT-based models.

| **Model Name** | **Year Released** | **Number of Parameters** | **Context Window** | **Access Through** |
|---|---|---|---|---|
| BioBERT | 2019 | 108 million | 512 tokens | Huggingface (dmis-lab/biobert-v1.1) |
| SciBERT | 2019 | 108 million | 512 tokens | Huggingface (alle-nai/ scibert_scivocab_uncased) |
| PubMedBERT | 2020 | 108 million | 512 tokens | Huggingface (microsoft/BiomedNLP-PubMedBERT-base-uncased-abstract) |

We fine-tuned these models in a 10-fold cross-validation setting. The folds were were created at the document level, as introduced above in previous study [25]. This strategy employed document-level fold splitting, which ensured the sentences from one document were used only either in the training or testing set to avoid over-fitting [77].

To provide fixed length input to the models, 128 and 256 were used as the maximum context length. When each example in each dataset was tokenized with the BERT tokenizers, the token length was less than 200, with one small exception. If the token length of the sentence is greater than the length of the context window, the sentence is truncated. This leads to a loss of information. Since we want to include the entire sentence in the model, we used 256 as the context length, which results in better performance compared to 128. On the other hand, we do not want to create input representations that consist mainly of fill information. Therefore, we did not choose 512 as the context length. Notably that fine-tuning with a maximum context length of 256 is faster than 512. We selected the best hyper-parameters based on the average F1-score for models on all datasets' validation splits. These are given in the Chapter 7.

### 5.3. Explaining the Relations by Identifying Interaction Keywords

In this part of the thesis, we worked on explaining the relations by identifying relational keywords. Our goal was to find the relation keywords in a protein-protein interaction sentence. We have used the LLL INO dataset [78]. As we mentioned earlier, this is the only dataset with INO keywords that we can experiment with. Here, we chose Llama-2, a causal large language model, for several reasons. First, because Llama-2 has a decoder-transformer architecture like GPT and Llama language models are excellent for generation tasks due to their architecture and training properties [46].

For this task, we used in-context learning and parameter-efficient instruction tuning approaches. Recent studies have investigated the performance of LLMs in zero-shot and few-shot learning for various tasks such as analyzing, producing, production and

comprehension [48, 79]. It was reported that the zero-shot and few-shot performances outperformed the results of the fine-tuned models [48]. Therefore, we investigated the in-context learning methods, zero-shot and few-shot learning, of the Llama-2 models on LLL INO. On the other hand, the advantages of parameter-efficient fine-tuning over in-context learning were investigated on a small dataset [69]. It was shown that PEFT outperforms in-context learning methods.

Of the various LLMs, Llama-2 variants were selected for this task because it has the same architecture as GPT-3 and the model is available as an open source model. The Llama-2 chat models were specifically fine-tuned to the following dialog datasets using RLHF. The study suggests that models trained with RLHF are better in terms of lower bias and harmful content generation [80]. Therefore, we selected chat models from the Llama-2 model family.

### 5.3.1. In-Context Learning

5.3.1.1. Zero-Shot Setting. In this setting, we experimented with zero-shot prompting for inferences with Llama-2 chat models.

In our prompt we have an introduction statement, instruction statement, input statement and a response statement. The introduction statement consists of the introduction key and the introduction sentence. Next, we have the instruction statement. The previously mentioned studies have also experimented with similar instructions with chat models [51]. Our previous experiments have shown that the instruction needs to be as clear and informative as possible [50]. The instruction statement consists of the instruction key and the instruction sentence. The instruction sentence simply tells the model what to do for our specific task. The input statement consists of input key and input sentence. This statement provides the sentence we want to investigate to the model. Finally, the response statement consists of only a response key. This indicates that the model is being required to response after the response key.

For each test example in the LLL INO dataset, we have created the prompt by combining the introduction statement, instruction statement, input statement and response statement. In Figure 5.3 you can see an illustration of the prompt with zero-shot setting. The prompt is created by combining the introduction, instruction and input for each sentence in the test split of the LLL INO dataset.

5.3.1.2. Few-Shot Setting. In few-shot prompting, we provide a series of examples in the prompt. This helps the model to learn from a small number of examples without the need for training or fine-tuning. With few-shot prompting, we take advantage of the power of the in-context learning [48].

Here we make the inference similar to the zero-shot prompting. The main difference is that, we have a context with more than one input and response statements. In the context, several examples are listed one after the other with their responses, consecutively. This is actually a list of input and response statements where it stands as a series of examples for the given instruction statement. The model understands the task based on the instruction and the specified example-response pairs in the context. After the context, we have an additional input statement which has the test sentence. Finally, we have the response statement which is simply a response key.

While selecting the number of examples in the context statement, we leveraged the long context of the Llama-2 models. As we mentioned, Llama-2 [53] has a large context length of 4096 tokens to be exact. This makes it possible to process more input tokens at once. In most cases, the more examples, the better the predictions [48]. Therefore, we have chosen the number of examples so that the context length is almost fully utilized. From the Figure 5.2, we can infer that the average token length of the LLL sentences is around 80. We decided that the context should contain 40 sentence-response pairs and one test sentence. These prompt examples were selected from the train split of the LLL-INO dataset.

In experiments with few-shot prompting, selecting the examples should be handled carefully. Depending on how many and how complex or representative examples we have in our context, the model may react differently. Therefore, we conducted three different few-shot prompting experiments in which we tried to use almost all the examples we had available. While reporting, we have included the scores of each few-shot experiment in different tables. The overall performance of the few-shot prompting can be considered as the average score of these three experiments. Therefore, the final score of the in-context learning is reported in a different table. We have provided the illustration of the few-shot prompting in the Figure 5.4. The prompt is created by combining the introduction statement, instruction statement, input statements and response statements as context, a test sentence and a response key. This prompt-style was constructed for each sentence in the test split of the LLL INO dataset.

### 5.3.2. Parameter-Efficient Instruction Fine-Tuning

In our work, we have used the PEFT from the HuggingFace library [67]. We used LoRa configuration from the PEFT library to implement the parameter-efficient instruction fine-tuning with Llama-2.

Instruction fine-tuning is simply a fine-tuning technique where the inputs are instructions. Instructions are not only used for inference, but can also be helpful in the training phase. Popular LLMs also have instruction fine-tuned versions [49,53], [81,82]. They are often referred to as chat or instruct models. Even if the name of the model does not explicitly include the keywords chat or instruct, they can still be trained with the instructions that follow the examples [82]. Training with instructions helps the models to generate better answers. Some studies have shown that model size is not as important as we believe. Once the model is trained to follow the instruction, the results can outperform the the performance of much larger LLMs [81]. Other studies revealed that if the model is instruction-tuned, the zero-shot performance increases on unseen tasks [83].

Creating the instruction following datasets is simple yet effective. One can convert a dataset into an instruction following format by simply combining instructions with the examples of a dataset. Here, we have created the prompt similar to the zero-shot setting described earlier. The main difference is in the response statement. Since the model needs to be trained using the training examples and consecutive labels, we provided the response after the response key. This prompt-style was applied for every training example in the batch for several epochs. The process of creating prompts and fine-tuning with instructions is described in Figure 5.5.

## 5.4. Evaluation Metrics

For evaluating the results of both detecting protein-protein interactions and identifying relation keywords, we used precision, recall and F1 metrics. These are commonly used to calculate the model performance in classification tasks in NLP. The scores are calculated using the elements of a confusion matrix for a model.

- True Positives (TP): The number of instances that are actually positive and are predicted by the model to be positive, which is correct.
- False Positives (FP): The number of samples that are actually negative but are predicted by the model to be positive, which is wrong.
- False Negative (TN): The number of samples that are actually negative and are predicted to be negative, which is correct.
- False Negative (FN): The number of samples that are actually positive but are predicted to be negative, which is wrong.

Precision calculates the proportion of samples correctly predicted as positive that are actually positive. It is often used to indicate that the predictions made by the model are accurate. Recall calculates the proportion of correctly predicted samples that belong to a class of interest. It is also known as sensitivity. The F1-score is a harmonic mean of precision and recall that provides a comprehensive performance evaluation.

The formula for precision is given as,

$$\textit{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}. \tag{5.1}$$

The formula for recall is given as,

$$\textit{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{5.2}$$

Finally, the formula for F1-score is given as,

$$\textit{F1-score} = 2 \cdot \frac{\mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}, \tag{5.3}$$

where it is the harmonic mean of precision and recall scores.

The metrics for precision, recall and F1- score are calculated for the positive class. It is also possible to calculate them for other classes, e.g. for the negative class. This can be done by a macro-averaged score, which is actually an unweighted mean of the metrics for each class. It is excellent for classification problems with multiple classes as well as for unbalanced datasets. Therefore, we have also reported the results for macro-averaged precision, recall and F1-score.

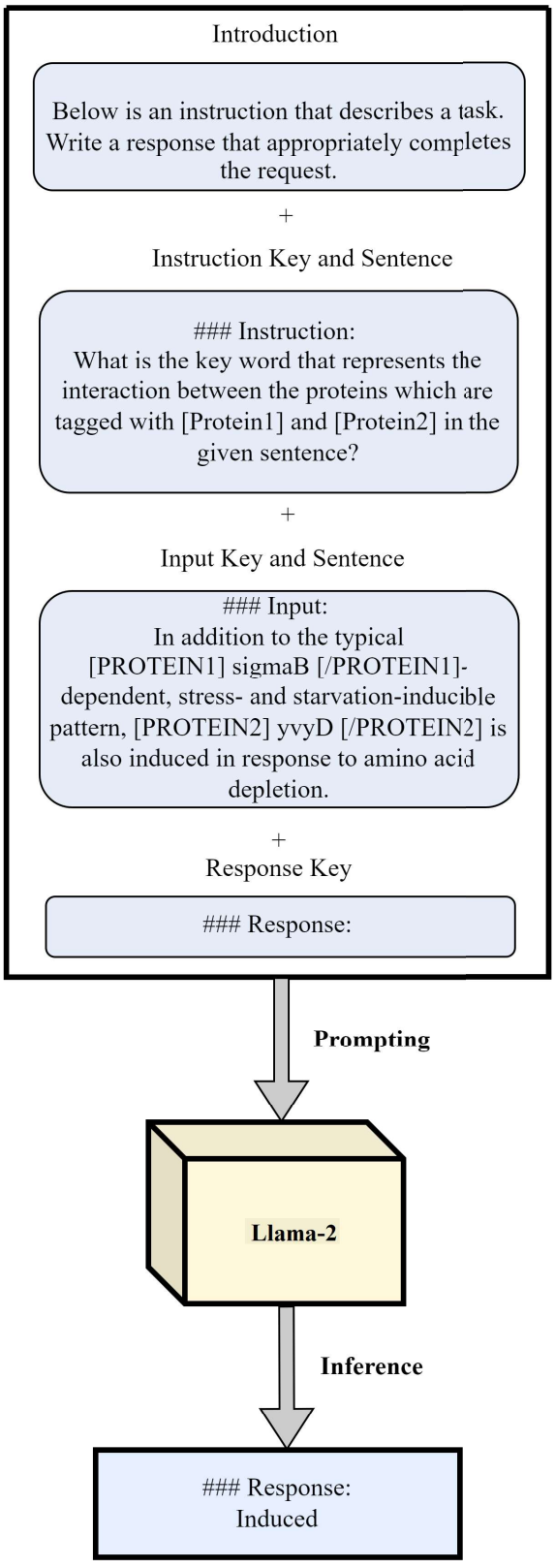


Figure 5.3. Zero-shot prompt example.

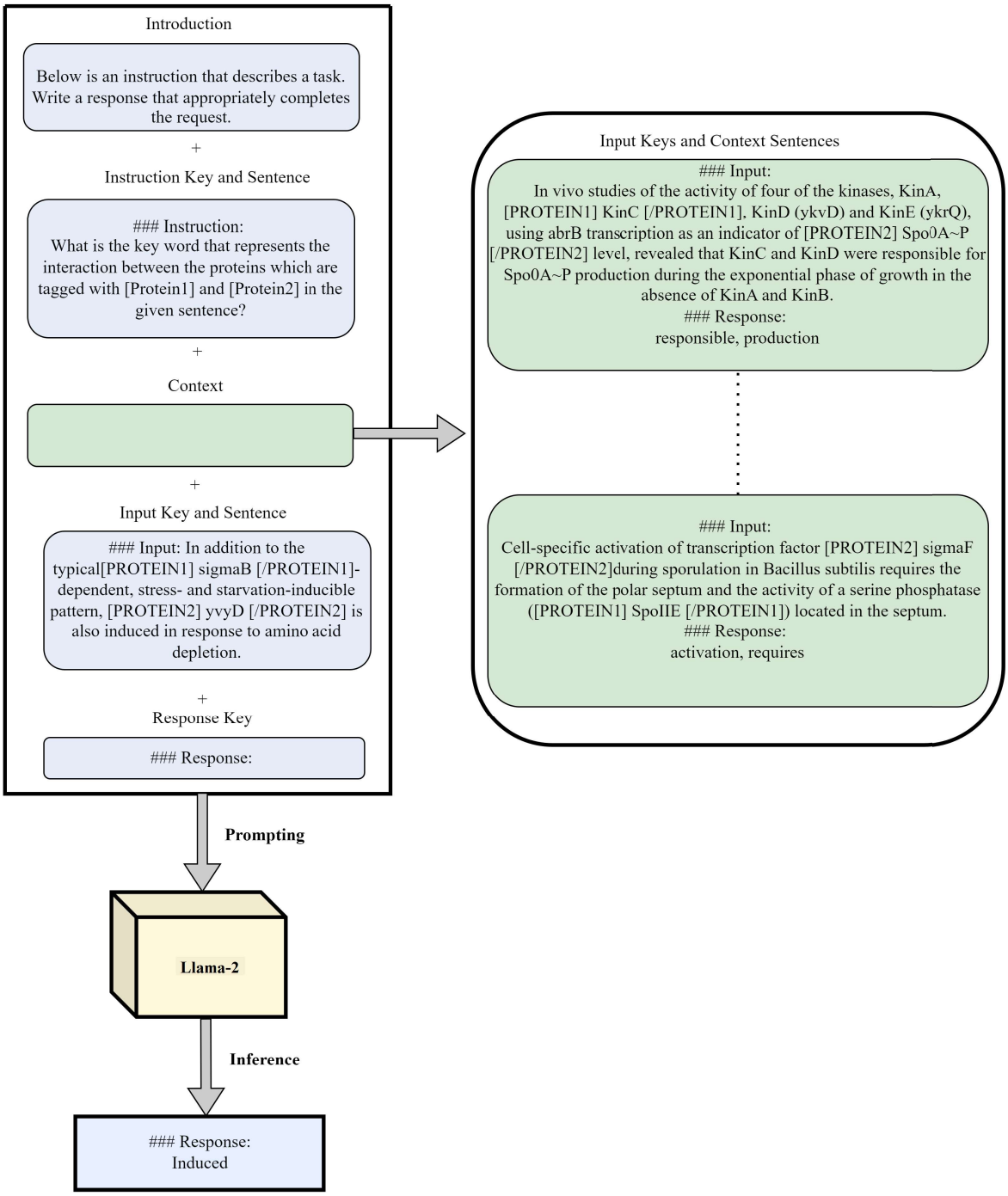


Figure 5.4. Few-shot prompt example.

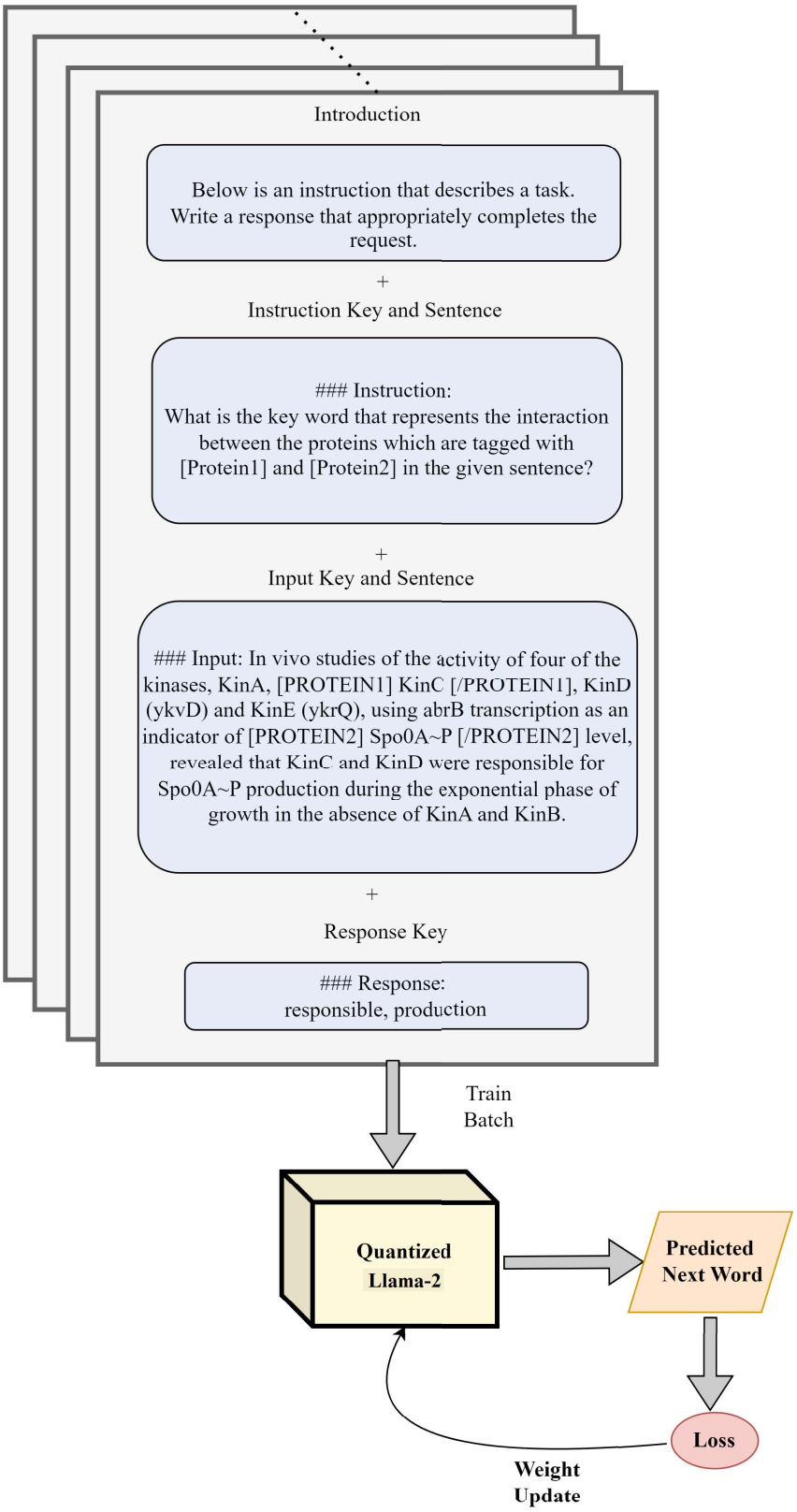


Figure 5.5. Instruction fine-tuning example.

# 6. EXPERIMENTS AND RESULTS

In this chapter, we will report on our results from the methods we used to identify protein-protein interactions and relation keywords. The experiments were conducted both on Bogazici University TETAM high performance computing clusters and graphic processors owned by TUBITAK BILGEM.

## 6.1. Relation Classification Results

In this section, the results of the BERT-based models on the protein-protein interaction detection task are reported. Here, we utilized

### 6.1.1. Results on Protein Masking Setting

As we mention in the Section 5.2.1, we applied 10-fold cross-validation on each dataset. We experimented with 1e-5, 2e-5, and 5e-5 as learning rate, 0.1, 0.01, and 0.001 as weight decay, and 4, 6, 8, and 32 as batch size. We selected the best hyper-parameters based on the average F1-score for models on all test split of the datasets. We got the best hyper-parameters on the LLL, IEPA, and HPRD50 datasets are shown in Table 6.1.

In Table 6.2 the performance of BioBERT, PubMedBERT, and SciBERT are listed. The results are given as percentages. As we can see, in terms of precision, recall and F1-score results all the models performed better on LLL compared to other datasets. As we mentioned earlier in Section 4, LLL dataset is balanced in terms of class distribution and it is also a smaller compared to the other datasets. We believe this can be the reason why the BERT-based models performed highest F1 score on the LLL dataset. It was therefore expected that BERT models would perform better in LLL.

Table 6.1. Hyper-parameter setting on LLL, IEPA and HPRD50.

| **Hyper-parameter** | **Value** |
|---|---|
| Optimizer | AdamW |
| Weight Decay | 0.01 |
| Learning Rate | 2e-5 |
| Batch Size | 4 |
| Epochs per Fold | 8 |
| Sequence Length | 256 |

Table 6.2. Performance of BERT models on the LLL, HPRD50 and IEPA.

| | **LLL** | | | **HPRD50** | | | **IEPA** | | |
|---|---|---|---|---|---|---|---|---|---|
| **Model** | **P %** | **R %** | **F1 %** | **P %** | **R %** | **F1 %** | **P %** | **R %** | **F1 %** |
| BioBERT | 82.82 | 91.95 | 86.84 | 75.79 | 77.63 | 74.95 | 75.80 | 83.21 | 78.81 |
| PubMedBERT | 85.25 | 87.35 | 85.42 | 78.81 | 82.71 | 79.65 | 77.70 | 81.05 | 78.49 |
| SciBERT | 84.54 | 86.07 | 84.66 | 73.03 | 79.02 | 74.67 | 73.29 | 79.32 | 75.53 |

When we analyze the F1-scores given the Table 6.2 on LLL dataset BioBERT model performed highest. Although the PubMedBERT and SciBERT models performed lower, trailing by mere 1.42% and 2.18% respectively, the margin is not very high. On HPRD50 dataset the PubMedBERT model performed highest. The margin between PubMedBERT's F1-score with BioBERT's and SciBERT's is higher, 4.7% and 4.98% respectively. On IEPA dataset the BioBERT model performed slightly higher, 0.32% to be precise, than PubMedBERT. The SciBERT, on the other hand, performed 3.28% lower than the PubMedBERT. We can conclude that the BioBERT model performed the best average F1-score on all the three protein-protein interaction datasets.

The macro-averaged scores of the BERT-based models on protein masked LLL, HPRD50, and IEPA are given in Table 6.3.

Table 6.3. Macro-averaged scores on LLL, HPRD50 and IEPA.

| | **LLL** | | | **HPRD50** | | | **IEPA** | | |
|---|---|---|---|---|---|---|---|---|---|
| **Model** | **P %** | **R %** | **F1 %** | **P %** | **R %** | **F1 %** | **P %** | **R %** | **F1 %** |
| BioBERT | 86.80 | 85.92 | 85.66 | 77.27 | 81.23 | 78.01 | 81.55 | 83.21 | 78.81 |
| PubMedBERT | 86.44 | 85.08 | 84.79 | 78.37 | 79.14 | 77.28 | 82.20 | 81.56 | 81.15 |
| SciBERT | 86.16 | 82.95 | 83.52 | 75.62 | 73.09 | 73.16 | 79.08 | 79.55 | 78.72 |

Since the BioInfer and AIMed datasets are much larger compared to LLL, IEPA and HPRD50 datasets the hyper-paramaters given in Table 6.1 did not work well on them. Therefore, we did a separate hyper-parameter tuning for BioInfer and AIMed datasets. Again, we followed the same methodology where we averaged the F1-scores for all BERT-based models on the two datasets. The best hyper-parameters for BERT-based models on the BioInfer and AIMed datasets are given in Table 6.4

Table 6.4. Hyper-parameter setting for BioInfer and AIMed.

| **Hyper-parameter** | **Value** |
|---|---|
| Optimizer | AdamW |
| Weight Decay | 0.01 |
| Learning Rate | 1e-5 |
| Batch Size | 8 |
| Epochs per Fold | 6 |
| Sequence Length | 256 |

In Table 6.5, the results of BioBERT, PubMedBERT and SciBERT on BioInfer and AIMed datasets are given. According to that table, PubMedBERT performed highest F1-score on both the two datasets. While SciBERT achieved the second-best F1-score on BioInfer, trailing by a mere 0.59%, BioBERT achieved the second-best score on AIMed, trailing by a mere 2.59%.

Table 6.5. Performance of BERT-based models on the BioInfer and AIMed.

| | **BioInfer** | | | **AIMed** | | |
|---|---|---|---|---|---|---|
| **Model** | **P %** | **R %** | **F1 %** | **P %** | **R %** | **F1 %** |
| BioBERT | 73.09 | 71.80 | 72.29 | 71.22 | 73.20 | 69.69 |
| PubMedBERT | 73.27 | 72.90 | 73.08 | 69.00 | 76.75 | 72.28 |
| SciBERT | 72.47 | 72.85 | 72.49 | 65.64 | 70.58 | 67.74 |

The macro-averaged scores of the BERT-based models on protein masked BioInfer and AIMed datasets are given in Table 6.6.

Table 6.6. Macro-averaged scores on BioInfer and AIMed.

| | **BioInfer** | | | **AIMed** | | |
|---|---|---|---|---|---|---|
| **Model** | **P %** | **R %** | **F1 %** | **P %** | **R %** | **F1 %** |
| BioBERT | 81.06 | 80.69 | 80.78 | 81.94 | 83.12 | 82.39 |
| PubMedBERT | 81.60 | 81.23 | 81.24 | 81.91 | 84.52 | 82.89 |
| SciBERT | 80.81 | 80.89 | 80.75 | 79.50 | 81.17 | 80.14 |

All the precision, recall and F1-scores are calculated for test splits after each 10 train steps. While reporting, each 10 test precision, recall and F1 metric scores are averaged. Therefore, the F1-score reported in the table might seem different than what we would get if we used averaged precision and average recall scores to calculate it. However, we are reporting all the metrics as an average over the 10 folds.

### 6.1.2. Results on Protein Tagging Setting

For this setting we start experimenting with the first three protein interaction datasets, LLL, IEPA and HPRD50, since these are smaller compared to BioInfer and AIMed. We experimented with 1e-5, 2e-5, and 5e-5 as learning rate, 0.1, 0.01, and 0.001 as weight decay, and 4, 6, 8, and 32 as batch size. When we did the hyper-

parameter tuning for protein tagged version of LLL, HPRD50 and IEPA we ended up with the same hyper-parameter setting on Table 6.1. On average, the tagged setting performed 5% lower on F1-score compared to the masked setting. Therefore, we did not analyze the BioInfer and AIMed datasets on the protein tagging setting.

Table 6.7. Performance of BERT-based models on tagged LLL, HPRD50, and IEPA.

| | **LLL** | | | **HPRD50** | | | **IEPA** | | |
|---|---|---|---|---|---|---|---|---|---|
| **Model** | **P %** | **R %** | **F1 %** | **P %** | **R %** | **F1 %** | **P %** | **R %** | **F1 %** |
| BioBERT | 81.63 | 89.26 | 84.48 | 69.78 | 68.87 | 67.21 | 67.80 | 73.92 | 70.06 |
| PubMedBERT | 86.07 | 89.12 | 86.13 | 74.45 | 74.45 | 73.49 | 70.15 | 78.94 | 73.60 |
| SciBERT | 81.49 | 80.15 | 78.55 | 68.04 | 75.34 | 70.92 | 71.94 | 71.87 | 71.60 |

The macro-averaged scores of the BERT-based models on protein tagged LLL, HPRD50, and IEPa datasets are given in Table 6.8.

Table 6.8. Macro-averaged results on tagged LLL, HPRD50 and IEPA.

| | **LLL** | | | **HPRD50** | | | **IEPA** | | |
|---|---|---|---|---|---|---|---|---|---|
| **Model** | **P %** | **R %** | **F1 %** | **P %** | **R %** | **F1 %** | **P %** | **R %** | **F1 %** |
| BioBERT | 84.06 | 82.52 | 81.57 | 75.43 | 74.69 | 73.65 | 74.37 | 74.87 | 73.98 |
| PubMedBERT | 88.86 | 83.86 | 84.21 | 79.42 | 79.17 | 78.67 | 77.49 | 77.89 | 76.88 |
| SciBERT | 79.90 | 77.75 | 75.39 | 76.14 | 76.67 | 75.85 | 76.23 | 76.43 | 76.12 |

Our results have shown that protein names are not as important for the detection of protein-protein interactions as we thought. We can conclude that the masked setting with the BioBERT model is best suitable for the task of protein-protein interaction detection.

## 6.2. Relation Keyword Identification Results

This section presents the results of the Llama-2 models in the task of identifying keywords for relations. Here, the results are computed using token-wise precision, recall and F1-scores. In computing the metrics, we first processed the predicted keywords. In this processing step, we mapped the keywords to the corresponding INO IDs by using the INO file [3]. Here you can find the INO keywords and the corresponding IDs. Some example of the INO IDs and keywords can be found in Table 6.9. The token-wise calculation explained with the examples using the Table 6.10.

Table 6.9. INO keywords and IDs.

| **Term (Keyword)** | **ID** |
|---|---|
| expressed | INO_0000102 |
| expresses | INO_0000102 |
| expression | INO_0000102 |
| gene expression | INO_0000102 |
| immuno-precipitate | INO_0000116 |
| immunopreticipating | INO_0000116 |
| immunoprecipitating | INO_0000116 |
| carboxylases | MI_1139 |
| carboxylation reaction | MI_1139 |

The INO file has not been updated. Therefore, some of the listed INO keywords in the LLL INO dataset are not available in the INO file. To handle these words, we used the WordNet Lemmatizer [84]. If at least one of the keywords from the predicted and the actual is not present in the INO file, the lemmas of the interaction words are compared. If the lemmas are same, it means a true prediction. Otherwise, it is a false prediction.

Table 6.10. Token-wise metric calculation.

| **Actual Keywords** | **Actual Keyword IDs** | **Predicted Keywords** | **Predicted Keyword IDs** | **P (%)** | **R (%)** | **F1 (%)** |
|---|---|---|---|---|---|---|
| dependent, drives, expression, promoter | INO_0000157, INO_0000157, INO_0000102, INO_T000008 | dependent, promoter | INO_0000157, INO_T000008 | 1 | 0.5 | 0.667 |
| gene expression, action | INO_0000102, action | gene expression, action | INO_0000102, action | 1 | 1 | 1 |
| dephosphorylation | MI_0203 | dependent | INO_0000157 | 0 | 0 | 0 |

### 6.2.1. Zero-shot In-Context Learning Results

In zero-shot in-context learning setting, the test examples from the LLL INO dataset are given one by one to the Llama-2 models. The model is loaded into the CPU for inference since the gradients were not updated. The details of the in-context learning was explained in the Section 5.

In Table 6.11 the zero-shot in-context learning performance of Llama-2-7b-chat, Llama-2-13b-chat and Llama-2-70b-chat on LLL INO dataset are given. Since the Llama-2-70b-chat model is much more large compared to the others, it yields the highest F1-score as expected. The Llama-2-13b-chat model achieved the second highest F1-score, trailing by 14.27%. Llama-2-7b-chat achieved the lowest F1-score, trailing by 41.2%. The models did not adhere to the output format in which interaction keywords were expected to be answered after the "Response" key. Therefore, the generated responses were processed manually. The Llama-2-7b-chat model generated responses that deviated far from the expected format. The Llama-2-70b chat model also responded with long paragraphs justifying the responses. This shows that the zero-shot in-context learning setting is not sufficient to use LLMs directly.

Table 6.11. Zero-shot prompting inference scores.

| | LLL INO | | |
|---|---|---|---|
| **Model** | **P%** | **R%** | **F1%** |
| Llama-2-7b-chat | 20.00 | 14.00 | 16.00 |
| Llama-2-13b-chat | 56.00 | 37.00 | 42.93 |
| Llama-2-70b-chat | 78.00 | 48.00 | 57.20 |

An example output response by the Llama-2-70b-chat model is as follows: The key word that represents the interaction between the proteins which are tagged with [Protein1] and [Protein2] in the given sentence is "induced." The sentence suggests that the expression of protein2 (yvyD) is induced in response to amino acid depletion, which implies a regulatory interaction between protein1 (sigmaB) and protein2. Therefore, the key word that represents this interaction is "induced."

### 6.2.2. Few-Shot In-Context Learning Results

In Table 6.12 the results of Llama-2-7b, Llama-2-13b, and Llama-2-70b models on few-shot in-context learning setting are given for each set of context examples. Since the model can react differently when different examples are given in context, the selection of context examples is not trivial.

The first set of examples are the first forty samples from the training split of the LLL INO dataset. Similarly, the second and the third set of context examples are the subsequent forty samples. The selection of context examples were explained in Section 5.

In the Table 6.12 the performances of the Llama-2 chat models are given. We can be seen that the results of the zero-shot setting of the Llama-2-70b-chat gave the highest F1-score. The Llama-2-13b-chat achieved the second highest F1-score, with a gap of 2%. In turn, the Llama-2-7b-chat achieved the lowest F1-score with a difference

of 17.74%. As we can see, the gap between the results of the individual models is much smaller. This indicates that the examples with in-context learning make the model perform better on the unseen tasks.

The results of the few-shot setting with the second set of context examples are given in Table 6.13. The Llama-2-13b-chat model achieved the highest F1-score, followed by the 7 billion model. Interestingly, the Llama-2-70b-chat model lags behind the 7 billion and 13 billion parameterized models by 1.2% and 1.54% respectively.

From the Table 6.14 we can see that the Llama-2-7b-chat model outperformed the Llama-2-13b-chat and Llama-2-70b-chat models. The Llama-2-70b model fall lag behind by 4.14%, and the Llama-2-13b model fall lag begind by 6.27%.

Table 6.15 shows the averaged scores from the Table 6.12, Table 6.13, Table 6.14. The Llama-2-70b-chat model achieved the highest F1-score overall. As expected, the Llama-2-13b-chat model was in second place, while the Llama-2-7b-chat model came last.

The models follow the output format better than the zero-shot in-context learning setting. However, human judgment or a post-processing step is still required to find the answer from the output. Therefore, the generated responses were processed manually to understand the model output and label them.

Table 6.12. Few-shot prompting inference scores with the first set of context examples.

| | **LLL INO** | | |
|---|---|---|---|
| **Model** | **P %** | **R %** | **F1 %** |
| Llama-2-7b-chat | 56.00 | 45.00 | 46.93 |
| Llama-2-13b-chat | 68.00 | 62.00 | 62.67 |
| Llama-2-70b-chat | 72.00 | 64.00 | 64.67 |

Table 6.13. Few-shot prompting inference scores with the second set of context examples.

| | LLL INO | | |
|---|---|---|---|
| **Model** | **P %** | **R %** | **F1 %** |
| Llama-2-7b-chat | 68.67 | 54.00 | 58.53 |
| Llama-2-13b-chat | 67.67 | 55.00 | 58.87 |
| Llama-2-70b-chat | 62.00 | 57.00 | 57.33 |

Table 6.14. Few-shot prompting inference scores with the third set of context examples.

| | LLL INO | | |
|---|---|---|---|
| **Model** | **P %** | **R %** | **F1 %** |
| Llama-2-7b-chat | 70.00 | 82.00 | 72.67 |
| Llama-2-13b-chat | 62.67 | 76.00 | 66.40 |
| Llama-2-70b-chat | 61.67 | 84.00 | 68.53 |

Table 6.15. Averaged few-shot prompting inference scores.

| | LLL INO | | |
|---|---|---|---|
| **Model** | **P %** | **R %** | **F1 %** |
| Llama-2-7b-chat | 64.89 | 60.33 | 59.38 |
| Llama-2-13b-chat | 66.11 | 64.33 | 62.65 |
| Llama-2-70b-chat | 65.22 | 68.33 | 63.51 |

### 6.2.3. Parameter-Efficient Instruction Fine-Tuning Results

Here we experimented with 7 and 13 billion parametered chat version of Llama-2 models. Due to limited resources, we could not fine-tune the Llama-2 model with 70 billion parameters. We used the 4-bit quantization technique, where the model

is loaded into the device with a precision of 4 bits. This configuration is available in the BitsAndBytes [85] library. In addition to quantization, we also used the LoRa configuration. As we have already mentioned, the LoRa configuration allows us to train only a small part of the weights. These trained weights are then multiplied by the pre-trained model parameters. By training only a smaller weight matrix, the process is fast and memory efficient.

The Llama-2 chat model with 13 billion parameters is not small enough to fit into a single 48 GB GPU. Even if the pre-trained weights of the model with the 4-bit quantization configuration fit in memory, it was not possible to use a single 48 GB GPU once the training starts and the gradient computations take place. Therefore, we used parallelism techniques to fit the quantized Llama-2 chat model with 13 billion parameters. Since the model is large and the dataset is small, we used the model parallelism technique. This splits the model across the available GPUs. With this setting, we can simply use all available GPU sources in parallel. This is called model parallelism.

For the training, we used the training and validation examples from the LLL INO dataset. The validation examples are used to tune the hyper-parameters. We also applied the early stopping strategy on the validation loss. Whenever the validation loss increases by two consecutive steps, the training is stopped. We also tuned the hyper-parameters for the training parameters and the PEFT-LoRa parameters. These are given in the Table 6.16.

The training results of the individual Llama-2 chat models are shown in Table 6.17. As we can see, the Llama-2-7b-chat and Llama-2-13b-chat performed similar after fine-tuning. The Llama-2-13b-chat was more precise of its predictions while the Llama-2-7b-chat predicted more of the true interaction keywords. Overall, the Llama-2-13b-chat outperformed the Llama-2-7b-chat by 1.33% on F1-score. The generated responses for the LLL INO test dataset are given in Appendix A.

Table 6.16. Llama-2 fine-tuning hyper-parameters.

| | **Hyper-parameter** | **Value** |
|---|---|---|
| **Training Parameters** | Epoch for Llama-2-7b | 5 |
| | Epoch for Llama-2-13b | 4 |
| | Batch Size per Device | 1 |
| | Gradient Accumulation Steps | 4 |
| | Learning Rate | 2e-4 |
| **PEFT - LoRa Parameters** | $\alpha$ | 32 |
| | R | 8 |

Table 6.17. Llama-2 fine-tune result.

| **Model** | **P %** | **R %** | **F1 %** |
|---|---|---|---|
| Llama-2-7b-chat | 79.33 | 84.00 | 80.15 |
| Llama-2-13b-chat | 86.00 | 80.00 | 81.33 |

The models followed the expected output format much more better compared to the in-context learning approaches. However, this time models generated unnecessary phrases such as ” END of input”, ” End of response” and ” End of task”. The outputs were processed manually.

The predictions of the Llama-2-13b chat model for the relation pairs in the test split of LLL INO dataset are given in Table A.1, Table A.2 and Table A.3. In order to be able to show all 25 test examples, we have divided the dataset into three portions.

# 7. CONCLUSION AND FUTURE WORK

In this work, we first investigated the detection of protein-protein interactions using the five benchmark datasets. We applied different strategies to process protein names. Our results showed that protein names are not important for detecting interactions with the BERT-based models. Of the models we tried, BioBERT performed best on the LLL dataset with an F1-score of 86.84%. To the best of our knowledge, this is the first work evaluating the fine-tuned performance of the BioBERT, SciBERT and PubMedBERT models for the five PPI benchmark corpora.

Next, we investigated the problem of interaction keyword identification. We leveraged the power of the causal language models by applying in-context learning and parameter-efficient instruction fine-tuning. We experimented with the Llama-2 chat models with 7, 13 and 70 billion. Since the model with 70 billion parameters requires a lot of memory compared to the others, it is not possible to use the largest model each time. Also, LLMs has an inference latency which is proportional to the size of the models. Our goal was to show the performance improvement by applying the in-context learning and fine-tuning methods to the Llama-2 chat models with 7 and 13 billion parameters. The results showed that the fine-tuned Llama-2 chat models with 7 and 13 billion parameters outperformed the largest model in in-context learning. The Llama-2-13b-chat model achieved the highest F1-score in fine-tuning and outperformed the Llama-2-70b-chat model in in-context learning with zero-shot and few-shot in F1-score by 24.13% and 17.82%, respectively.

We have contributed to the literature by investigating the detection of INO keywords using state-of-the-art NLP models. As far as we know, this problem has not been studied before. We believe that this work will lead to the biomedical problems requiring ontology being studied by LLMs. We would like to continue our research by applying the INO ontology to the problem of recognizing keywords in relations. As the context length of LLMs increases day by day, it is possible to feed the model with the

INO keywords by using the prompts. We believe that this will increase the performance of the model. Another possible work could concern the selection of the best prompts. A detailed analysis is needed here, which will take months. Also, different open source LLMs could be good to experiment with.

The two different tasks can be worked on in sequence. First, the fine-tuned BioBERT model can predict the protein-protein interactions. Second, these sentences can be passed to the fine-tuned Llama-2-chat-13b model to identify the keywords for the INO relation keywords. By combining the best performing models for these two tasks, we can create a comprehensive system for recognizing protein-protein relations and INO keywords. We believe this can help researchers to better study biomedical texts. It can also be used as an assistant to annotate the protein-protein relation datasets with the INO keywords.

# REFERENCES


1. Li, Y., Y. Chen, Y. Qin, Y. Hu, R. Huang and Q. Zheng, "Protein-Protein Interaction Relation Extraction Based on Multigranularity Semantic Fusion", *Journal of Biomedical Informatics*, Vol. 123, p. 103931, 2021.

2. Zhou, D., D. Zhong and Y. He, "Biomedical Relation Extraction: From Binary to Complex", *Computational and Mathematical Methods in Medicine*, Vol. 2014, pp. 298473–298473, 2014.

3. Hur, J., A. Özgür, Z. Xiang and Y. He, "Development and Application of an Interaction Network Ontology for Literature Mining of Vaccine-Associated Gene-Gene Interactions", *Journal of Biomedical Semantics*, Vol. 6, No. 1, pp. 1–10, 2015.

4. 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*, Vol. 36, No. 4, pp. 1234–1240, 2020.

5. Beltagy, I., K. Lo and A. Cohan, "SciBERT: A Pretrained Language Model for Scientific Text", ArXiv 1903.10676, 2019.

6. Gu, Y., R. Tinn, H. Cheng, M. Lucas, N. Usuyama, X. Liu, T. Naumann, J. Gao and H. Poon, "Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing", *Association for Computing Machinery Transactions on Computing for Healthcare*, Vol. 3, No. 1, pp. 1–23, 2021.

7. 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 Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Fed-*

*eration of Natural Language Processing*, Suntec, Singapore, pp. 1003–1011, 2009.

8. Zeng, D., K. Liu, Y. Chen and J. Zhao, "Distant Supervision for Relation Extraction via Piecewise Convolutional Neural Networks", *Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, pp. 1753–1762, 2015.

9. Roth, B., C. Conforti, N. Poerner, S. K. Karn and H. Schütze, "Neural Architectures for Open-Type Relation Argument Extraction", *Natural Language Engineering*, Vol. 25, No. 2, pp. 219–238, 2019.

10. Sun, C. and Y. Wu, "Distantly Supervised Entity Relation Extraction with Adapted Manual Annotations", *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, Hawaii, USA, Vol. 33, pp. 7039–7046, 2019.

11. Köksal, A. and A. Özgür, "The RELX Dataset and Matching the Multilingual Blanks for Cross-Lingual Relation Classification", ArXiv 2010.09381, 2020.

12. Brody, S., S. Wu and A. Benton, "Towards Realistic Few-Shot Relation Extraction", *Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic, pp. 5338–5345, 2021.

13. Hochreiter, S. and J. Schmidhuber, "Long Short-term Memory", *Neural Computation*, Vol. 9, No. 8, pp. 1735–1780, 1997.

14. Xu, Y., L. Mou, G. Li, Y. Chen, H. Peng and Z. Jin, "Classifying Relations via Long Short Term Memory Networks along Shortest Dependency Paths", *Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, pp. 1785–1794, 2015.

15. Hendrickx, I., S. N. Kim, Z. Kozareva, P. Nakov, D. O. Séaghdha, S. Padó, M. Pennacchiotti, L. Romano and S. Szpakowicz, "Semeval-2010 task 8: Multi-way Clas-

sification of Semantic Relations Between Pairs of Nominals", ArXiv 1911.10422, 2019.

16. Banerjee, S. and K. Tsioutsiouliklis, "Relation Extraction using Multi-Encoder LSTM Network on a Distant Supervised Dataset", *Institute of Electrical and Electronics Engineers 12th International Conference on Semantic Computing*, Laguna Hills, CA, USA, pp. 235–238, 2018.

17. Tian, Y., G. Chen, Y. Song and X. Wan, "Dependency-Driven Relation Extraction with Attentive Graph Convolutional Networks", *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, Vol. 1, pp. 4458–4471, 2021.

18. Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, "Attention is All You Need", *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Long Beach, California, USA, Vol. 30, pp. 6000–6010, 2017.

19. Devlin, J., M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, Minnesota, Vol. 1, pp. 4171–4186, 2019.

20. Yamada, I., A. Asai, H. Shindo, H. Takeda and Y. Matsumoto, "LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention", *Conference on Empirical Methods in Natural Language Processing*, pp. 6442–6454, 2020.

21. Xiao, Y., C. Tan, Z. Fan, Q. Xu and W. Zhu, "Joint Entity and Relation Extraction with a Hybrid Transformer and Reinforcement Learning Based Model", *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, pp. 9314–9321, 2020.

22. Feng, Y., H. Zhang, W. Hao and G. Chen, "Joint Extraction of Entities and Relations Using Reinforcement Learning and Deep Learning", *Computational Intelligence and Neuroscience*, Vol. 2017, p. 7643065, 2017.

23. Erkan, G., A. Özgür and D. Radev, "Semi-Supervised Classification for Extracting Protein Interaction Sentences Using Dependency Parsing", *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Prague, Czech Republic, pp. 228–237, 2007.

24. Bui, Q.-C., S. Katrenko and P. M. Sloot, "A Hybrid Approach To Extract Protein–Protein Interactions", *Bioinformatics*, Vol. 27, No. 2, pp. 259–265, 2011.

25. 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. 11, pp. 1–12, 2008.

26. 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, pp. 145–164, 2016.

27. 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, 2018.

28. Peng, Y. and Z. Lu, "Deep Learning for Extracting Protein-Protein Interactions From Biomedical Literature", ArXiv 1706.01556, 2017.

29. Su, P. and K. Vijay-Shanker, "Investigation of BERT Model on Biomedical Relation Extraction Based on Revised Fine-tuning Mechanism", *Institute of Electrical and Electronics Engineers International Conference on Bioinformatics and Biomedicine*, pp. 2522–2529, 2020.

30. Hu, Y., Y. Chen, Y. Qin and R. Huang, "Learning Entity-Oriented Representation

for Biomedical Relation Extraction", *Journal of Biomedical Informatics*, Vol. 147, p. 104527, 2023.

31. Nédellec, C., "Learning Language in Logic-Genic Interaction Extraction Challenge", *Proceedings of the 4th Learning Language in Logic Workshop*, Vol. 7, pp. 1–7, 2005.

32. Fundel, K., R. Küffner and R. Zimmer, "RelEx—Relation Extraction Using Dependency Parse Trees", *Bioinformatics*, Vol. 23, No. 3, pp. 365–371, 2007.

33. Ding, J., D. Berleant, D. Nettleton and E. Wurtele, "Mining MEDLINE: Abstracts, Sentences, or Phrases?", *Pacific Symposium on Biocomputing*, Vol. 7, pp. 326–337, 2002.

34. 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, pp. 1–24, 2007.

35. Ekbal, A., S. Saha and U. K. Sikdar, "Biomedical Named Entity Extraction: Some Issues of Corpus Compatibilities", *SpringerPlus*, Vol. 2, No. 1, pp. 1–12, 2013.

36. Warikoo, N., Y.-C. Chang and W.-L. Hsu, "LBERT: Lexically Aware Transformer-based Bidirectional Encoder Representation Model for Learning Universal Bio-Entity Relations", *Bioinformatics*, Vol. 37, No. 3, pp. 404–412, 2021.

37. Tang, Z., X. Guo, Z. Bai, L. Diao, S. Lu and L. Li, "A Protein-Protein Interaction Extraction Approach Based on Large Pre-Trained Language Model and Adversarial Training", *Korean Society for Internet Information Transactions on Internet and Information Systems*, Vol. 16, No. 3, pp. 771–791, 2022.

38. Park, G., S. McCorkle, C. Soto, I. Blaby and S. Yoo, "Extracting Protein-Protein Interactions (PPIs) from Biomedical Literature using Attention-based Relational Context Information", *Institute of Electrical and Electronics Engineers Interna-*

*tional Conference on Big Data*, pp. 2052–2061, 2022.

39. Bravo, À., J. Piñero, N. Queralt-Rosinach, M. Rautschka and L. I. Furlong, "Extraction of Relations Between Genes and Diseases From Text and Large-Scale Data Analysis: Implications for Translational Research", *BMC Bioinformatics*, Vol. 16, pp. 1–17, 2015.

40. Herrero-Zazo, M., I. Segura-Bedmar, P. Martínez and T. Declerck, "The DDI Corpus: An Annotated Corpus with Pharmacological Substances and Drug–Drug Interactions", *Journal of Biomedical Informatics*, Vol. 46, No. 5, pp. 914–920, 2013.

41. Singha Roy, S. and R. E. Mercer, "Extracting Drug-Drug and Protein-Protein Interactions from Text using a Continuous Update of Tree-Transformers", *The 22nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks*, Toronto, Canada, pp. 280–291, 2023.

42. Zhang, L., G. Yu, D. Xia and J. Wang, "Protein–Protein Interactions Prediction Based on Ensemble Deep Neural Networks", *Neurocomputing*, Vol. 324, pp. 10–19, 2019.

43. Gao, H., C. Chen, S. Li, C. Wang, W. Zhou and B. Yu, "Prediction of Protein-Protein Interactions Based on Ensemble Residual Convolutional Neural Network", *Computers in Biology and Medicine*, Vol. 152, p. 106471, 2023.

44. Soleymani, F., E. Paquet, H. L. Viktor, W. Michalowski and D. Spinello, "ProtInteract: A Deep Learning Framework for Predicting Protein–Protein Interactions", *Computational and Structural Biotechnology Journal*, Vol. 21, pp. 1324–1348, 2023.

45. Alsentzer, E., J. Murphy, W. Boag, W.-H. Weng, D. Jindi, T. Naumann and M. McDermott, "Publicly Available Clinical BERT Embeddings", *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, Minneapolis, Minnesota, USA, pp. 72–78, 2019.

46. Radford, A., K. Narasimhan, T. Salimans and I. Sutskever, "Improving Language Understanding by Generative Pre-Training", 2018, `https://openai.com/research/language-unsupervised`, accessed on Feb 20, 2024.

47. Radford, A., J. Wu, R. Child, D. Luan, D. Amodei and I. Sutskever, "Language Models are Unsupervised Multitask Learners", *OpenAI Blog*, Vol. 1, No. 8, p. 9, 2019.

48. Brown, T., B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever and D. Amodei, "Language Models are Few-Shot Learners", *Advances in Neural Information Processing Systems*, Punta Cana, Dominican Republic, Vol. 33, pp. 1877–1901, 2020.

49. Schulman, J., B. Zoph, C. Kim, J. Hilton, J. Menick, J. Weng, J. F. Ceron Uribe, L. Fedus, L. Metz, M. Pokorny, R. Gontijo Lopes, S. Zhao, A. Vijayvergiya, E. Sigler, A. Perelman, C. Voss, M. Heaton, J. Parish, D. Cummings, R. Nayak, V. Balcom, D. Schnurr, T. Kaftan, C. Hallacy, N. Turley, N. Deutsch, V. Goel, J. Ward, A. Konstantinidis, W. Zaremba and et al., "Introducing ChatGPT", 2022, `https://openai.com/blog/chatgpt`, accessed on Feb 20, 2024.

50. Rehana, H., N. B. Çam, M. Basmaci, Y. He, A. Özgür and J. Hur, "Evaluation of GPT and BERT-based Models on Identifying Protein-Protein Interactions in Biomedical Text", ArXiv 2303.17728, 2023.

51. Chen, Q., H. Sun, H. Liu, Y. Jiang, T. Ran, X. Jin, X. Xiao, Z. Lin, H. Chen and Z. Niu, "An Extensive Benchmark Study on Biomedical Text Generation and Mining with ChatGPT", *Bioinformatics*, Vol. 39, No. 9, p. btad557, 2023.

52. Touvron, H., T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin and G. Grave, Edouard Lample, "Llama: Open and Efficient Foundation Language Models", ArXiv 2302.13971, 2023.

53. Touvron, H., L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. Moya Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann and et al., "Llama 2: Open Foundation and Fine-Tuned Chat Models", ArXiv 2307.09288, 2023.

54. Wu, C., X. Zhang, Y. Zhang, Y. Wang and W. Xie, "PMC-Llama: Further Finetuning Llama on Medical Papers", ArXiv 2304.14454, 2023.

55. Chen, Z., A. H. Cano, A. Romanou, A. Bonnet, K. Matoba, F. Salvi, M. Pagliardini, S. Fan, A. Köpf, A. Mohtashami, A. Sallinen, A. Sakhaeirad, V. Swamy, I. Krawczuk, D. Bayazit, A. Marmet, S. Montariol, M.-A. Hartley, M. Jaggi and A. Bosselut, "MEDITRON-70B: Scaling Medical Pretraining for Large Language Models", ArXiv 2311.16079, 2023.

56. Tran, H., Z. Yang, Z. Yao and H. Yu, "BioInstruct: Instruction Tuning of Large Language Models for Biomedical Natural Language Processing", ArXiv 2310.19975, 2023.

57. Thapa, S. and S. Adhikari, "ChatGPT, Bard, and Large Language Models for Biomedical Research: Opportunities and Pitfalls", *Annals of Biomedical Engineering*, Vol. 51, No. 12, pp. 2647–2651, 2023.

58. Pal, S., M. Bhattacharya, S.-S. Lee and C. Chakraborty, "A Domain-Specific Next-Generation Large Language Model (LLM) or ChatGPT Is Required for Biomedical Engineering and Research", *Annals of Biomedical Engineering*, Vol. 52, No. 3, pp.

1–4, 2023.

59. Achiam, J., S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu, H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bogdonoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button and et al., "GPT-4 Technical Report", ArXiv 2303.08774, 2023.

60. Zhang, B. and R. Sennrich, "Root Mean Square Layer Normalization", *Advances in Neural Information Processing Systems*, Vol. 32, pp. 12360–12371, 2019.

61. Shazeer, N., "Glu Variants Improve Transformer", ArXiv 2002.05202, 2020.

62. Agarap, A. F., "Deep Learning Using Rectified Linear Units (RELU)", ArXiv 1803.08375, 2018.

63. Ramachandran, P., B. Zoph and Q. V. Le, "Searching For Activation Functions", ArXiv 1710.05941, 2017.

64. Dauphin, Y. N., A. Fan, M. Auli and D. Grangier, "Language Modeling With Gated Convolutional Networks", *International Conference on Machine Learning*, pp. 933–941, 2017.

65. Houlsby, N., A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan and S. Gelly, "Parameter-Efficient Transfer Learning for NLP", *International Conference on Machine Learning*, pp. 2790–2799, 2019.

66. He, R., L. Liu, H. Ye, Q. Tan, B. Ding, L. Cheng, J.-W. Low, L. Bing and L. Si, "On the Effectiveness of Adapter-Based Tuning for Pretrained Language Model Adaptation", ArXiv 2106.03164, 2021.

67. Mangrulkar, S., S. Gugger, L. Debut, Y. Belkada, S. Paul and B. Bossan,

"PEFT: State-of-the-art Parameter-Efficient Fine-Tuning Methods", 2022, https://github.com/huggingface/peft, accessed on Feb 20, 2024.

68. Hu, E. J., Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang and W. Chen, "LoRa: Low-Rank Adaptation of Large Language Models", ArXiv 2106.09685, 2021.

69. Liu, H., D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal and C. A. Raffel, "Few-Shot Parameter-Efficient Fine-Tuning Is Better and Cheaper Than In-Context Learning", *Advances in Neural Information Processing Systems*, Vol. 35, pp. 1950–1965, 2022.

70. Paperno, D., G. Kruszewski, A. Lazaridou, Q. N. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda and R. Fernández, "The LAMBADA dataset: Word Prediction Requiring a Broad Discourse Context", ArXiv 1606.06031, 2016.

71. Zellers, R., A. Holtzman, Y. Bisk, A. Farhadi and Y. Choi, "Hellaswag: Can a Machine Really Finish Your Sentence?", ArXiv 1905.07830, 2019.

72. Mostafazadeh, N., N. Chambers, X. He, D. Parikh, D. Batra, L. Vanderwende, P. Kohli and J. Allen, "A Corpus and Evaluation Framework for Deeper Understanding of Commonsense Stories", ArXiv 1604.01696, 2016.

73. Wikipedia, "Bacillus subtilis — Wikipedia, The Free Encyclopedia", 2023, https://en.wikipedia.org/wiki/Bacillus_subtilis, accessed on Feb 20, 2024.

74. Ohta, T., Y. Tateisi and J.-D. Kim, "The GENIA corpus: an annotated research abstract corpus in molecular biology domain", *Proceedings of the Second International Conference on Human Language Technology Research*, San Francisco, CA, USA, pp. 82–86, 2002.

75. Pyysalo, S., A. Airola, J. Heimonen, J. Björne, F. Ginter and T. Salakoski, "Com-

parative Analysis of Five Protein-Protein Interaction Corpora", *BMC Bioinformatics*, Vol. 9, pp. 1–11, 2008.

76. Ahmed, M., J. Islam, M. R. Samee and R. E. Mercer, "Identifying Protein-Protein Interaction Using Tree LSTM and Structured Attention", *The Institute of Electrical and Electronics Engineers 13th International Conference on Semantic Computing*, pp. 224–231, 2019.

77. Bunescu, R. C. and R. J. Mooney, "Subsequence Kernels for Relation Extraction", *Proceedings of the 18th International Conference on Neural Information Processing Systems*, Vancouver, British Columbia, Canada, p. 171–178, 2005.

78. Özgür, A., J. Hur and Y. He, "The Interaction Network Ontology-Supported Modeling and Mining of Complex Interactions Represented with Multiple Keywords in Biomedical Literature", *BioData Mining*, Vol. 9, No. 1, pp. 1–17, 2016.

79. Kojima, T., S. S. Gu, M. Reid, Y. Matsuo and Y. Iwasawa, "Large Language Models are Zero-Shot Reasoners", *Advances in Neural Information Processing Systems*, Vol. 35, pp. 22199–22213, 2022.

80. Ganguli, D., A. Askell, N. Schiefer, T. I. Liao, K. Lukošiūtė, A. Chen, A. Goldie, A. Mirhoseini, C. Olsson, D. Hernandez, D. Drain, D. Li, E. Tran-Johnson, E. Perez, J. Kernion, J. Kerr, J. Mueller, J. Landau, K. Ndousse, K. Nguyen, L. Lovitt, M. Sellitto, N. Elhage, N. Mercado, N. DasSarma, O. Rausch, R. Lasenby, R. Larson, S. Ringer, Kundu and et al., "The Capacity for Moral Self-Correction in Large Language Models", ArXiv 2302.07459, 2023.

81. Ouyang, L., J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. F. Christiano, J. Leike and R. Lowe, "Training Language Models To Follow Instructions With Human Feedback", *Advances in Neural Information Processing Systems*, Vol. 35, pp. 27730–27744, 2022.

82. Taori, R., I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang and T. B. Hashimoto, "Stanford Alpaca: An Instruction-following LLaMA model", 2023, `https://github.com/tatsu-lab/stanford_alpaca`, accessed on Feb 20, 2024.

83. Wei, J., M. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai and Q. V. Le, "Finetuned Language Models are Zero-Shot Learners", *International Conference on Learning Representations*, 2022.

84. Fellbaum, C., "WordNet", *Theory and Applications of Ontology: Computer Applications*, pp. 231–243, Springer, 2010.

85. Dettmers, T., M. Lewis, Y. Belkada and L. Zettlemoyer, "LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale", ArXiv 2208.07339, 2022.

# APPENDIX A: KEYWORD PREDICTIONS OF LLAMA-2-13B-CHAT MODEL

In Table A.1, Table A.2, and Table A.3 the Llama-2-13b chat model predictions are given for the LLL INO test set.

Table A.1. Predicted keywords for LLL INO test split portion one.

| Id | Keywords | Predicted Keywords | Sentence |
|---|---|---|---|
| 135 | dependent, inducible | dependent, induced | In addition to the typical [PROTEIN1] sigmaB [/PROTEIN1]-dependent, stress- and starvation-inducible pattern, [PROTEIN2] yvyD [/PROTEIN2] is also induced in response to amino acid depletion. |
| 115 | recognize, promoter | recognize, promoter | We overproduced and purified sigma(X) from Escherichia coli and demonstrate that in vitro, both sigma(A) and [PROTEIN1] sigma(X) [/PROTEIN1] holoenzymes recognize promoter elements within the sigX-[PROTEIN2] ypuN [/PROTEIN2] control region. |
| 131 | dependent, drives, expression, promoter | dependent, promoter | The sigmaB-dependent promoter drives expression of yvyD under stress conditions and after glucose starvation, whereas a [PROTEIN1] sigmaH [/PROTEIN1]-dependent promoter is responsible for [PROTEIN2] yvyD [/PROTEIN2] transcription following amino acid limitation. |
| 55 | dependent | dependent | Three new [PROTEIN1] sigmaB [/PROTEIN1]-dependent genes (ydaE, [PROTEIN2] ydaG [/PROTEIN2] and yfkM) encoding proteins with still unknown functions were also described. |
| 95 | transcription | transcription | Transcription of the cotB, cotC, and [PROTEIN2] cotX [/PROTEIN2] genes by final [PROTEIN1] sigma(K) [/PROTEIN1] RNA polymerase is activated by a small, DNA-binding protein called GerE. |
| 29 | requires, expression | expression, requires | The [PROTEIN2] rocG [/PROTEIN2] gene of Bacillus subtilis, encoding a catabolic glutamate dehydrogenase, is transcribed by SigL-containing RNA polymerase and requires for its expression [PROTEIN1] RocR [/PROTEIN1], a member of the NtrC/NifA family of proteins that bind to enhancer-like elements, called upstream activating sequences (UAS). |
| 157 | regulon, includes | regulon, includes | The [PROTEIN1] sigma W [/PROTEIN1] regulon includes a penicillin binding protein ([PROTEIN2] PBP4* [/PROTEIN2]) and a co-transcribed amino acid racemase (RacX), homologues of signal peptide peptidase (YteI), flotillin (YuaG), ABC transporters (YknXYZ), non-haem bromoperoxidase (YdjP), epoxide hydrolase (YfhM) and three small peptides with structural similarities to bacteriocin precursor polypeptides. |
| 51 | dephosphorylation | dependent | Dephosphorylation of [PROTEIN2] SpoIIAA-P [/PROTEIN2] by [PROTEIN1] SpoIIE [/PROTEIN1] is strictly dependent on the presence of the bivalent metal ions Mn2+ or Mg2+. |

Table A.2. Predicted keywords for LLL INO test split portion two.

| Id | Keywords | Predicted Keywords | Sentence |
|---|---|---|---|
| 101 | gene expression, action | gene expression, action | In vitro transcription experiments suggest that the differential pattern of [PROTEIN2] cot [/PROTEIN2] gene expression results from the combined action of GerE and another transcription factor, [PROTEIN1] SpoIIID [/PROTEIN1]. |
| 145 | inhibits | inhibits | The class includes AsiA form bacteriophage T4, which inhibits Escherichia coli sigma 70; FlgM, present in both gram-positive and gram-negative bacteria, which inhibits the flagella sigma factor sigma 28; [PROTEIN1] SpoIIAB [/PROTEIN1], which inhibits the sporulation-specific sigma factor,[PROTEIN2] sigma F [/PROTEIN2] and sigma G. of Bacillus subtilis; RbsW of B. subtilis. which inhibits stress response sigma factor sigma B; and DnaK, a general regulator of the heat shock response, which in bacteria inhibits the heat shock sigma factor sigma 32. |
| 19 | dependent, expression | dependent | Therefore, the physiological role of [PROTEIN1] sigmaB [/PROTEIN1]-dependent [PROTEIN2] katX [/PROTEIN2] expression remains obscure. |
| 85 | expression, induced, depended, presence | expression. depended | The expression of the [PROTEIN2] bkd [/PROTEIN2] operon was induced by the presence of isoleucine or valine in the growth medium and depended upon the presence of the sigma factor [PROTEIN1] SigL [/PROTEIN1], a member of the sigma 54 family. |
| 15 | expression, control | expression. control | The expression of [PROTEIN2] rsfA [/PROTEIN2] is under the control of both sigma(F) and [PROTEIN1] sigma(G) [/PROTEIN1]. |
| 66 | repressed, transcription | activated. transcription | A low level of GerE activated transcription of cotD by sigmaK RNA polymerase in vitro, but a higher level of [PROTEIN1] GerE [/PROTEIN1] repressed [PROTEIN2] cotD [/PROTEIN2] transcription. |
| 24 | controlled | controlled | [PROTEIN2] sigmaF [/PROTEIN2] is controlled by a regulatory cascade involving an anti-sigma factor, [PROTEIN1] SpoIIAB [/PROTEIN1], an anti-anti-sigma factor, SpoIIAA, and a membrane-bound phosphatase, SpoIIE, which converts the inactive, phosphorylated form of SpoIIAA back to the active form. |
| 30 | dependent | dependent | Unlike the case for other [PROTEIN1] sigma(54) [/PROTEIN1]-dependent genes, [PROTEIN2] rocG [/PROTEIN2] has no UAS; instead, its expression depends on a sequence located 1.5 kilobases downstream of the rocG promoter, beyond the end of the rocG coding region. |
| 132 | member of, regulons | member of, regulons | This indicates that the [PROTEIN2] yvyD [/PROTEIN2] gene product, being a member of both the [PROTEIN1] sigmaB [/PROTEIN1] and sigmaH regulons. might negatively regulate the activity of the sigmaL regulon. |
| 105 | activate. transcription | activate, transcription | A low concentration of [PROTEIN1] GerE [/PROTEIN1] activated cotB transcription by final sigma(K) RNA polymerase. whereas a higher concentration was needed to activate transcription of cotX or [PROTEIN2] cotC [/PROTEIN2]. |

Table A.3. Predicted keywords for LLL INO test split portion three.

| Id | Keywords | Predicted Keywords | Sentence |
|---|---|---|---|
| 152 | recognized | recognized | The sspH and [PROTEIN2] sspL [/PROTEIN2] genes are recognized primarily by the forespore-specific sigma factor for RNA polymerase, [PROTEIN1] sigmaG [/PROTEIN1]; the sspN-tlp operon is recognized equally well by sigmaG and the other forespore-specific sigma factor, sigmaF. |
| 16 | dependent | expression, dependent | Expression of the [PROTEIN1] sigma(K) [PROTEIN1]-dependent [PROTEIN2] cwlH [/PROTEIN2] gene depended on gerE. |
| 75 | increasing, production | stimualting, transcription | [PROTEIN1] ComK [/PROTEIN1] negatively controls the transcription of hag by stimulating the transcription of comF-[PROTEIN2] flgM [/PROTEIN2], thereby increasing the production of the FlgM antisigma factor that inhibits sigmaD activity. |
| 18 | level, depends | depends, level | Our results indicate that the level of [PROTEIN2] KatX [/PROTEIN2] level in outgrowing spores depends mainly on [PROTEIN1] EsigmaF [/PROTEIN1], because sigB mutants show normal KatX activity in dormant and outgrowing spores. |
| 12 | dependent, expression | dependent, expression | In the shaA mutant, [PROTEIN1] sigma(H) [/PROTEIN1]-dependent expression of [PROTEIN2] spo0A [/PROTEIN2] and spoVG at an early stage of sporulation was sensitive to external NaCl. |
| 9 | expression, controlled | controlled | Analysis of the expression of a translational ywhE-lacZ fusion showed that [PROTEIN2] ywhE [PROTEIN2] expression is sporulation-specific, and is controlled predominantly by the forespore-specific sigma factor sigma(F), and to a lesser extent by [PROTEIN1] sigma(G) [/PROTEIN1]. |
| 31 | controlling, transcription | control, expression | In addition to controlling transcription of [PROTEIN2] phrC [/PROTEIN2], [PROTEIN1] sigmaH [/PROTEIN1] appears to control expression of at least one other gene required for production of CSF. |