Biomedical Event Extraction Based on Transformers and Knowledge Graphs

Laura A. Zanella Calzada

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Biomedical Event Extraction Based on Transformers and Knowledge Graphs

THÈSE

présentée et soutenue publiquement le 15 décembre 2023

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

Laura A. Zanella Calzada

Composition du jury

Président :	Thierry Charnois	Pr. Université de Paris 13
Rapporteurs :	Natalia Grabar	C.R. CNRS, Université de Lille
	Anne Vilnat	Pr. Université Paris-Saclay
Examinateurs :	Claire Gardent	D.R. CNRS, Université de Lorraine
	Mathieu Roche	D.R. CIRAD
	Thierry Charnois	Pr. Université de Paris 13
Directeur :	Yannick Toussaint	Pr. Université de Lorraine

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thesul.

Acknowledgements

Yannick, merci beaucoup ! Merci de m'avoir donné l'opportunité d'entrer dans le monde du doctorat. Merci pour toutes les discussions, les enseignements, les défis, ton amitié et tout ce processus que tu as vécu avec moi. Je sais que tu sais, mais parfois c'était difficile et frustrant. D'autres fois, je me sentais vraiment épanouie. Il n'y a rien dans cette étape que je regrette.

Merci à mon jury d'avoir accepté de participer à cette soutenance de thèse. Merci Anne Vilnat et Natalia Grabar pour la révision de ma thèse et pour tous vos commentaires. Merci également à Claire Gardent, Mathieu Roche et Thierry Charnois d'avoir accepté d'être mes examinateurs.

Merci au LORIA. J'ai beaucoup apprécié mon séjour ici. Mes collègues, qui sont devenus mes amis (je ne les nommerai pas, vous savez qui vous êtes), sont la meilleure partie. Je vous aime beaucoup. Merci pour toute la compagnie et le soutien, ce chemin est beaucoup moins solitaire et difficile grâce à vous. Merci de m'avoir patiemment écouté dans les moments difficiles, à cause de la thèse ou d'autres choses de la vie. Merci pour toutes les sorties, la compagnie, les bonnes discussions et les rires. Un merci spécial à tous ceux qui m'ont aidé à relire et corriger mon manuscrit, vous êtes magnifiques ! Merci à Caro, Floriane, Isabelle et à toutes les personnes de la cantine et de la cafétéria de toujours nous accueillir avec un immense sourire. Quoi qu'il en soit, je ne pourrai jamais finir de dire combien je suis reconnaissant d'avoir retrouvé tous ces gens.

Merci à ma maman, qui me soutient toujours dans tous mes projets. Merci de croire en moi plus qu'en moi-même. Je sais que la distance est difficile. Tu me manques tous les jours et je t'aime plus que tu ne peux l'imaginer.

«*Gracias a mi mamá, que siempre me apoya en todos mis proyectos. Gracias por creer más en mí que yo misma. Yo sé que la distancia es dura. Te extraño todos los días y te amo más de lo que te puedas imaginar.*»

Un merci infini à toutes les personnes qui m'ont aidé à m'intégrer dans ce pays et qui m'ont permis de me sentir à ma place. Le processus d'adaptation à une nouvelle culture, un nouvel emploi et une nouvelle vie n'est pas toujours facile. Encore moins si ce processus débute par une pandémie.

A., mais quelle belle surprise ! Merci d'être ici.

I dedicate this thesis to
all the women who have
gone far from home to
pursue their dreams.

Especially to my mom.

Je dédie cette thèse à
toutes les femmes parties
loin de chez elles pour
poursuivre leurs rêves.

Spécialement à ma maman.

Dedico esta tesis a
todas las mujeres que han
ido lejos de casa para
perseguir sus sueños.

Especialmente a mi mamá.

Résumé

L'extraction d'événements biomédicaux peut être divisée en trois sous-tâches principales : (1) la détection de déclencheurs d'événements, (2) l'identification d'arguments et (3) la construction d'événements. Dans cette étude, pour la première sous-tâche, nous analysons un ensemble de modèles de langage transformer couramment utilisés dans le domaine biomédical afin d'évaluer et de comparer leur capacité à détecter les déclencheurs d'événements. Nous affinons les modèles en utilisant sept corpus annotés manuellement pour évaluer leurs performances dans différents sous-domaines biomédicaux. SciBERT s'est révélé être le modèle le plus performant, présentant une légère amélioration par rapport aux modèles de référence. Pour la deuxième sous-tâche, nous construisons un graphe de connaissances (*KG*, en anglais) à partir des corpus biomédicaux et intégrons ses *KG embeddings* à SciBERT pour enrichir son information sémantique. Nous démontrons que l'ajout des *KG embeddings* au modèle améliore la performance de l'identification d'arguments d'environ 20 %, et d'environ 15 % par rapport à deux modèles de référence. Pour la troisième sous-tâche, nous utilisons le modèle génératif, ChatGPT, basé sur des invitations, pour construire l'ensemble final d'événements extraits. Nos résultats suggèrent que l'affinage d'un modèle transformateur pré-entraîné à partir de zéro avec des données biomédicales et générales permet de détecter les déclencheurs d'événements et d'identifier des arguments couvrant différents sous-domaines biomédicaux, améliorant ainsi sa généralisation. De plus, l'intégration des *KG embeddings* dans le modèle peut significativement améliorer la performance de l'identification d'arguments d'événements biomédicaux, surpassant les résultats des modèles de référence.

Mots-clés: Extraction d'événements biomédicaux, Détection de déclencheurs d'événements, Identification d'arguments, Modèles de langage transformer, Graphe de connaissances.

Abstract

Biomedical event extraction can be divided into three main subtasks; (1) event trigger detection, (2) argument identification and (3) event construction. In this work, for the first subtask, we analyze a set of transformer language models that are commonly used in the biomedical domain to evaluate and compare their capacity for event trigger detection. We fine-tune the models using seven manually annotated corpora to assess their performance in different biomedical subdomains. SciBERT emerged as the highest-performing model, presenting a slight improvement compared to baseline models. For the second subtask, we construct a knowledge graph (KG) from the biomedical corpora and integrate its KG embeddings to SciBERT to enrich its semantic information. We demonstrate that adding the KG embeddings to the model improves the argument identification performance by around 20 %, and by around 15 % compared to two baseline models. For the third subtask, we use the generative model, ChatGPT, based on prompts to construct the final set of extracted events. Our results suggest that fine-tuning a transformer model that is pre-trained from scratch with biomedical and general data allows to detect event triggers and identify arguments covering different biomedical subdomains, and therefore improving its generalization. Furthermore, the integration of KG embeddings into the model can significantly improve the performance of biomedical event argument identification, outperforming the results of baseline models.

Keywords: Biomedical Event Extraction, Event Trigger Detection, Argument Identification, Transformer Language Models, Knowledge Graphs.

Contents

List of Figures

List of Tables

Glossary

Bi-LSTM Bidirectional Long Short Term Memory.

CG Cancer Genetics.

DL Deep Learning.

EE Event Extraction.

EPI Epigenetics and Post-translational Modifications.

GE11 GENIA 2011.

GE13 GENIA 2013.

ID Infectious Diseases.

KG Knowledge Graph.

LSTM Long Short Term Memory.

ML Machine Learning.

MLEE Multi-Level Event Extraction.

NER Named Entity Recognition.

NLP Natural Language Processing.

PC Pathway Curation.

RE Relation Extraction.

Introduction

A significant part of the world's biomedical knowledge is documented in natural language text, ranging from scientific publications and medical reports to electronic health records and biomedical literature. These documents containing biomedical knowledge are steadily increasing, making it difficult to read them at the same speed as they are produced. Therefore, much of this information remains stagnant in details in the text that are not exploitable for further analysis or application. Effectively harnessing and utilizing this information in its unstructured format presents a major challenge. Natural Language Processing (NLP) techniques offer powerful tools and methodologies to unlock the potential of this textual knowledge by transforming it into a structured format that is suitable for computer manipulation and analysis. This process, known as *information extraction*, plays a crucial role in converting unstructured text into actionable insights.

Information extraction techniques allow researchers and domain experts to extract structured knowledge from biomedical texts, making it available for various computational analyses, knowledge discovery, and decision support systems. By automatically identifying and extracting relevant information, such as biomedical entities and their relationships, these techniques can be implemented for advanced data mining ends, semantic search, literature-based discovery, clinical decision support, and drug discovery. Among the techniques to extract information are Named Entity Recognition (NER), Relation Extraction (RE), and Event Extraction (EE). NER involves the scanning of unstructured text to identify entities for term normalization and classification into pre-defined categories, e.g., proper names, organizations, dates, and locations. These categories can be related to a specific domain, e.g., Figure 1[1] shows a sentence annotated with two named entities in the biomedical domain, *Simple_chemical* and *Gene_or_gene_product*.

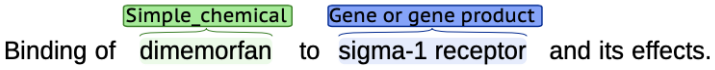


Figure 1: Example of a sentence annotated with named entities through NER.

When NER is applied to the biomedical domain, entities are usually categorized into genes and proteins, drugs, adverse effects, diseases, tissues, organs, pathways, or metabolites. Nevertheless,

[1]The visualization of the annotated sentence is done using the visualization tool *brat (brat.nlplab.org/)*

the non-standard usage of abbreviations, synonymous, homonyms, ambiguities, and multi-words entities makes biomedical NER a challenging task.

RE is a task that usually follows NER. Its purpose is to identify which entities are connected to find meaningful interactions. Figure 2 shows an example where the two named entities recognized earlier are related with the *Agonist* relation type. The term "Agonist" refers to a substance that binds to a receptor and activates it to produce a biological response. In this context, if *dimemorfan* binds to the *sigma-1 receptor* and leads to a response or activation, it would be considered an agonist for the *sigma-1 receptor*.

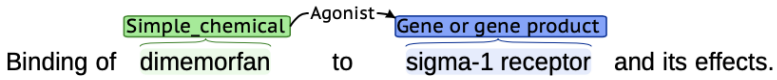


Figure 2: Example of a sentence annotated with relations through RE.

For instance, in biomedical RE the identification of interactions between proteins allows to construct networks of protein-protein interactions. In the same way, the location of gene-disease relations can bridge molecular and phenotype information. These relation networks give the possibility of exploring connections that were previously unknown and associating them with established relationships. However, one of the main problems in biomedical RE is that it loses worthy details about the context while trying to keep accurate interactions between biologically relevant entities. This can cause the practical usefulness of the extracted relations to be limited. Some of the details that can be missed in RE are sub-relations (e.g. *At Loc*, to refer to the location of the event) and co-participation of more than two entities that cover separated functions (e.g. while *gene_1* is the regulation cause, *protein_1* and *protein_2* are affected by this regulation) [Perera et al., 2020].

EE is a more expressive method to capture natural language statements that allow to formally represent biological processes, such as the study of biomolecular mechanisms or epigenetic changes. Figure 3 shows an example where the two named entities recognized earlier are the arguments of the *Binding* event. The event is built from the trigger word "Binding". One of the arguments, referring to the *Simple_chemical* entity, plays the role of *Theme* (answering the question "What was bound?"). The other argument, referring to the *Gene_or_gene_product* entity, plays the role of *Site* (answering the question "Where did the bind occur?").

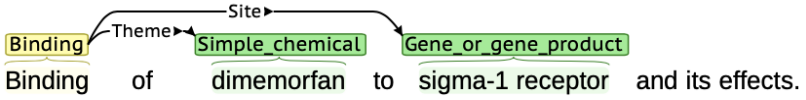


Figure 3: Example of a sentence annotated with an event through EE.

Currently, events are the most complex units of information that can be extracted from text, since they can capture n-ary dynamic relations between entities and also between events [Frisoni et al., 2021]. Events' structures are composed of multiple participants with a specific semantic role, including nested events and overlapping definitions. In RE, the structure is composed only of the pair of participants that form the relation. In this way, events provide flexibility to interconnect entities allowing to construct a finer description taking into account all the elements of the context. The dynamic properties make events the closest equivalent to information extracted directly from humans [Zerva and Ananiadou, 2015]. Therefore, EE allows to obtain exhaustive, interpretable, and quantifiable interactions describing implicit and emerging knowledge that otherwise would be buried in the text.

A biomedical event is constructed from an event trigger and one or more arguments that orbit around the trigger. Event triggers generally refer to nouns or verbs that express an action, circumstance, or eventuality, while the arguments refer either to biomedical entities or to other events, called nested events. To acquire these elements and build an event, the task is usually divided into three main subtasks, event trigger detection, argument identification, and event construction. Event trigger detection recognizes and classifies the trigger words into a set of predefined types of event triggers, while argument identification recognizes and classifies the roles between the event triggers and their respective arguments [Shen et al., 2019]. Event construction refers to the unmerging of the arguments that correspond to the same event, i.e., overlap events are merged into a unified node along with its corresponding set of outgoing edges [Björne and Salakoski, 2011].

Event trigger detection has a fundamental role in the construction of events. Indeed, the triggers are the targets that allow us to know that an event may exist [Cui et al., 2020]. This subtask is usually considered as classification problem, where each word needs to be classified into a predefined set of trigger types. The difficulty for trigger detection comes from the sensitivity to the domain or subdomain (text can present specialized language), linguistic forms (triggers can be single words, multi-words, discontinuous markers), and ambiguity on the trigger class (a trigger word can be given different trigger classes) [Zerva and Ananiadou, 2015].

Argument identification can be considered a multi-category classification problem, where the directed relation between a trigger and an entity or other event needs to be classified into a predefined set of role types. When these arguments are correctly identified, the event extracted has the potential to provide a reliable means of improving domain knowledge. One of the main complexities in identifying arguments is that they can be part of one or multiple events (one-to-one and one-to-multiple relations), where they play the same or different roles.

Among the main challenges in EE is to identify where events are located in the text and to classify them, since events can belong to different categories, assuming different roles [Ramponi et al., 2020]. This causes the failure to retrieve some of the information incomplete or erroneous.

Although such errors may not always be vitally important, they can be extremely detrimental when the information is used for a biomedical application, as they can lead to erroneous or unhelpful claims. This is one of the main reasons why the development and improvement of automatic biomedical information extraction models is a key issue in scientific research.

Transformer language models have been widely adopted to try to reduce errors in event extraction due to their positive achievements in performance for solving different types of NLP tasks. BERT [Devlin et al., 2018], which stands for Bidirectional Encoder Representations from Transformers, is a language model designed to pre-train bidirectional representations of words, taking into account the semantics by considering both the left and right directions of the text. From this pre-training, BERT can be fine-tuned by including additional layers on top of the model to solve new specific tasks. Furthermore, several domain-specific BERT variants have been developed by being trained on large corpora with the same context, such as the biomedical domain. However, since the learning of the models is limited to the specific domain in which they were trained, they present limitations in performance when using them in different biomedical subdomains.

To improve the integration of domain knowledge, knowledge graph (KG) models have been implemented along with language models for different information tasks in the biomedical domain [Huang et al., 2020, Yang et al., 2020, Dasgupta et al., 2021, Roy and Pan, 2021, Milošević and Thielemann, 2023]. Biomedical KGs are a resource for the integration of one or more sources of information (often manually curated datasets) into a graph, where biomedical entities are represented by nodes and the relations between them by edges [Nicholson and Greene, 2020]. KG models integrate nodes and edges into a low-dimensional vector space, known as embeddings while preserving the structure of the KG and its semantic information. This vector space plays a pivotal role in the automated augmentation of the KG, where the process involves predicting missing connections between nodes, commonly referred to as a link prediction task. The objective of link prediction is to infer new relations between the entities, based on the previously learned relations from the graph. By leveraging the semantic information encoded in the embeddings, the system can infer and incorporate new relationships, thus enhancing the overall comprehensiveness and utility of the graph. Therefore, with the new connections found, new information is integrated into the graph, and knowledge is enhanced.

1 Proposal and contributions

In this thesis, we analyze the performance of five previously trained transformer language models for two EE subtasks, even trigger detection and argument identification. We first identify whether transformer models allow the detection of event triggers in different biomedical subdomains. Then, we enrich the semantic information of the best-performing model with KG

embeddings to assess whether integrating these embeddings improves the model's ability to identify biomedical arguments and their roles. For this purpose, BERT, BioBERT, SciBERT, PubMedBERT, and BioMedRoBERTa are fine-tuned using two different classifiers, a linear layer, and a Bidirectional Long Short Term Memory (Bi-LSTM) layer, to detect biomedical event triggers. These BERT variants are chosen for comparison since they share the same BERT architecture but have previously been pre-trained with different data in the biomedical and/or general domain, showing positive results in biomedical information extraction tasks [Lee et al., 2020, Beltagy et al., 2019, Erdengasileng et al., 2022]. Models are learned using seven manually annotated data sets merged, which are Cancer Genetics 2013 (CG) [Nédellec et al., 2013], Epigenetics and Post-translational Modifications 2011 (EPI) [Ohta et al., 2011], GENIA 2011 (GE11) [Kim et al., 2011], GENIA 2013 (GE13) [Kim et al., 2013], Infectious Diseases 2011 (ID) [Pyysalo et al., 2011], Pathway Curation 2013 (PC) [Nédellec et al., 2013] and Multi-Level Event Extraction (MLEE) [Pyysalo et al., 2012]. These corpora were originally developed for the event extraction task in different biomedical subdomains. In addition to these data, two features are included as lexical and syntactical extra-information to the models, the stems and the parts-of-speech (POS), respectively. Then, a KG is constructed from the biomedical events contained in the biomedical corpora, and its KG embeddings are computed. These embeddings are integrated into the transformer language model to classify the roles between the previously identified triggers and the biomedical entities and/or other triggers, to detect the event arguments.

The main contributions of this thesis are summarized as follows:

- Evaluation and comparison of five transformer language models based on BERT for the detection of biomedical event triggers in EE;
- Assessment of whether adding lexical and syntactic information for fine-tuning the models improves biomedical event detection for EE;
- Proposal of a novel strategy to integrate KG embeddings into transformer language models to identify biomedical event arguments in EE;
- Empirical analysis of the impact of merging different annotated corpora to detect biomedical event triggers and identify arguments on different biomedical subdomains.

2 Thesis overview

This thesis is structured as follows:

- **Chapter 1.** The goal of this chapter is to explore the advancements in biomedical information extraction using NLP techniques, including biomedical EE. We describe the state-of-the-art approaches and methodologies employed in information extraction to overcome current challenges and extract valuable knowledge from biomedical texts;

- **Chapter 2.** This chapter describes the advancements made in biomedical EE using transformer language models, including BERT and some of its biomedical variants. We discuss the principles and methodologies underlying transformer-based approaches, highlighting the adaptations and advancements made for biomedical EE. By examining the capabilities and limitations of transformer language models, we aim to provide a comprehensive understanding of their application in extracting biomedical events from unstructured textual data;

- **Chapter 3.** This chapter focuses on the recent developments in utilizing KGs for biomedical knowledge discovery. We explore the principles and methodologies followed in KGs and their potential applications in the biomedical domain. By examining the capabilities and limitations of KGs, we aim to provide a comprehensive understanding of their role in extracting and organizing biomedical knowledge from unstructured textual data;

- **Chapter 4.** This chapter describes the two first contributions of the thesis, where we compare the pre-training strategies of BERT and four of its variants, and their impact on event extraction on different biomedical subdomains. We analyze and evaluate how their different pre-training characteristics, such as the initial weights and training data, can influence the detection of biomedical event triggers. In addition, we assess the integration of lexical and syntactic information into the models to improve the detection of event triggers. Through this evaluation, we can observe how the pre-training of these models indeed affects trigger detection, which is an important point to consider when choosing the model to perform this task. Also, we find that the addition of extra information does not improve significantly the performance of the models for this task;

- **Chapter 5.** In this chapter we present the third contribution, where we explore a novel approach to enhance argument identification on different biomedical subdomains using a KG-enriched transformer model. By merging the embeddings from transformer models and KGs, we aim to discover meaningful relations between event triggers and their corresponding arguments;

- **Chapter 6.** In this chapter we describe the fourth contribution, a qualitative and quantitative evaluation of the information integrated in the embeddings calculated with the transformer model and the KG. We develop a performance analysis of the detection of triggers and identification of arguments according to the different subdomains. In addition, we compare our model with baseline models and we show a use case of the application of the model in a biomedical text;

- **Chapter 7.** In this chapter we present an exploratory study focused on the extraction of biomedical events from biomolecular text. Our approach involves applying the proposed model, which combines embeddings from a transformer model and KGs to a set of biomedical abstracts chosen by an interdisciplinary biomolecular team. We also include the use of the generative language model, ChatGPT, to make the final construction of the events extracted. This study has as purpose to identify the potential application of our model in automating EE from complex biomedical texts.

Chapter 1

Biomedical information extraction and current limitations

Contents

Overview

Biomedical information extraction encompasses a sequence of procedures employing specialized methods. Initially, Named Entity Recognition (NER) is commonly employed, succeeded by Relation Extraction (RE) to discern the connections between significant biological entities within the biomedical domain. Event Extraction (EE) serves as an integrative technique, unifying the preceding two methods into a cohesive framework, thereby generating intricate units of knowledge that describe biological mechanisms.

The following sections describe the steps comprising a comprehensive information extraction pipeline based on NER, RE and EE. We discuss challenges in these methods and explore feature extraction techniques and modeling methods.

1.1 Main steps in biomedical information extraction

The main steps followed in biomedical information extraction systems are shown in Figure 1.1.

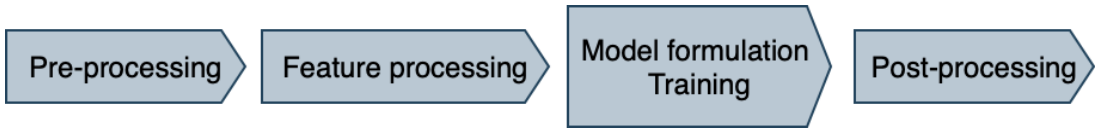


Figure 1.1: Pipeline followed for biomedical information extraction.

The pre-processing step refers to the data cleaning, tokenization, and normalization, to reduce ambiguities in the step of feature processing. In the feature processing step the main objective is to extract features that represent the text that is analyzed. For this, the text is converted into an appropriate representation so the system can read it and the modeling can be applied. The representation of the features can vary according to the modeling method. While dictionary and rule-based methods can use features in their textual format, Machine Learning (ML) methods need the text to be represented as real-value numbers. Once the features are processed, they are used for the training of the models and then, the output obtained may go through a post-processing step to reduce errors.

1.2 Pre-processing

Pre-processing steps include data cleaning, tokenization, removing stop words (e.g., "the", "a", "for", etc.), stemming (e.g., "changing" → "chang"), lemmatization (e.g., "changing" → "change"), sentence boundary detection and case normalization. However, the usage of these steps varies based on the application and the approach followed. In the case of NER, for example, the pre-processing steps that are usually applied comprise data cleaning, tokenization, name normalization, abbreviation, and head noun resolving measures. The main purpose of these steps is to reduce problems related to the data (e.g., noise, bias, missing data, sparseness) in the processing of the features [Perera et al., 2020]. Following the work of Mitrofan and Ion [Mitrofan and Ion, 2017], the standard pre-processing pipeline for biomedical information extraction consists of applying sentence splitting followed by word segmentation. This second step can be done by applying tokenization, part-of-speech (POS) tagging, and grouping the tokens that present similar meanings (based on linguistic rules).

1.3 Feature processing

Feature processing refers to the extraction of information in the form of features that represent the text that will be analyzed. The processing applied to these features depends on the system and the application for which they will be used. Systems that are based on rules and dictionaries mainly use orthographic and morphological features that are focused on word formations. For example, regular expressions to identify words beginning, suffixes, and prefixes, counting the number of characters and phrases based on the analysis of POS. The type of features extracted through this processing can be classified into three main categories, rich text features, vector representations of text, and word embeddings based on neural networks. The details of each category are described in Section 1.3.1, Section 1.3.2, and Section 1.3.3, respectively.

1.3.1 Rich text features

The rich text features that are the most commonly used for biomedical information extraction are described below.

Linguistic features focus on the text syntax, containing information about the sentence structures or the POS tags.

Orthographic features are related to the word formation, attempting to capture indicative characteristics of the words, e.g., the presence of a particular symbol or digit suggesting the presence of a named entity.

Morphological features contain information about the common characteristics that identify the essence of an entity, such as suffixes, prefixes, or n-grams.

Contextual features take into account the characteristics of the preceding and succeeding tokens of a word to enhance its representation.

Finally, lexicon features contain extra information about the domain, e.g., synonyms and trigger words that belong to specific fields [Campos et al., 2012].

1.3.2 Vector representations of text

In ML systems, feature processing focuses on extracting vector representations of text, i.e., real-value word representations, also known as word embeddings [Levy and Goldberg, 2014]. Among the simplest representations of words are bag-of-words, POS tagging with term frequencies, and one-hot encoding (binary representation). More complex formulations also include dimensional reduction, e.g. distributional or clustering-based representations [Turian et al., 2010].

One-hot vector word representation is the most basic method of word embedding. It consists of assigning a binary vector of length N to each word in a vocabulary of size N. The binary vector assigns one to the component that corresponds to the index of the word and zero to the rest. The disadvantage of this representation is the size of the vectors, since the larger the vocabulary, the larger the dimensions and sparsity of the word embeddings [Braud and Denis, 2015].

The clustering-based word representation consists of having clusters of words that describe information contextually similar. This approach presents affinities with one-hot vector word representation but in this case, the dimension of the vectors and the sparsity are reduced. Besides, it includes contextual information [Tang et al., 2014].

The distributional word representation is based on co-occurrence matrices with statistical approximations that extract latent semantic information. For this approach, the pre-processing consists of filtering the stop-words to avoid the high frequencies of unrelated words affecting the results. The rest of the words are represented in a co-occurrence matrix, where the rows represent the words in the vocabulary, and the columns represent the counts of each vector. Then, the dimension of the matrix is reduced by applying a statistical approximation (or unsupervised learning function) [Turian et al., 2010, Sahlgren, 2006].

1.3.3 Word embeddings based on neural networks

Currently, the state-of-the-art methods to process and extract features for biomedical information extraction applications are word embeddings based on neural networks, since they capture deep details of the syntax and semantics of text [Wang et al., 2020b]. Here, word embeddings are real-value vectors representing words learned under an unsupervised or semi-supervised approach from a text corpus.

There word representations can be categorized into context-independent word embeddings and context-dependent word embeddings. For the first category, the vector of a word will be the same regardless of the context. For the second category, the vector of a word will vary depending on the context, e.g. the vector embedding of the word *park* in "I went to the park last weekend." will be different from the same word in "I want to park my car next to yours.". In the first category *park* has the same word vector in both sentences, even if the meaning is not the same. The main models of context-independent word embeddings and context-dependent word embeddings are described in Section 1.3.3.1 and Section 1.3.3.2, respectively.

1.3.3.1 Context-independent word embeddings

Word2Vec [Mikolov et al., 2013a] is one of the first-word representation models based on a two-layer shallow neural network. The main idea behind the model is to take a text as input, create a vocabulary from this text, and produce a multidimensional vector representation for

each word in the vocabulary as output. The word vectors are positioned in the vector space in a way that words with common contextual meaning are close to each other. Word2Vec can follow two different algorithms to obtain the word embeddings according to the application, Continuous Bag-of-Words (CBoW) and Continuous Skip-Gram. In CBoW the principle is to predict the current word based on a window that encloses its close contextual words in the space, without considering the order of those words. Conversely, Continuous Skip-Gram uses the current word to predict the words that are contextually close to it. Therefore, the output obtained from the neural network is a vector that represents a word for CBoW or a vector that represents a set of words for Continuous Skip-Gram.

In GloVe [Pennington et al., 2014], which stands for Global Vectors, the word representation captures the global statistics of the text, contrary to Word2Vec which captures the local statistics. This model is based on an unsupervised learning algorithm that evaluates the word-word co-occurrence probabilities of a text looking for the interpretation of the semantic dependence between the words. In other words, GloVe is trained using log-bi-linear modeling with weighted least-squared error to learn the contextual word vectors. The contextual distance between the word vectors creates a linear sub-structural pattern in the vector space, that is defined as logarithmic probability. Therefore, GloVe learns the contextual word vectors in a way that the logarithmic probability of the word-word occurrences equals the dot product of the word vectors.

FastText is an extension of Word2Vec, introduced by Facebook Research [Bojanowski et al., 2017]. In this model, the learning of the word vector representations is based on N-gram characters. For example, to obtain the vector embedding of the word *cellular* using a 3-gram character representation, the representation would be *<ce*, *cel*, *ell*, *llu*, *lul*, *ula*, *lar*, *ar>*, where '<' and '>' indicate the boundaries of the word. The N-grams are then used to train the model based on the Skip-gram algorithm for the computation of the word embeddings. One of the advantages of FastText is that it is very effective in representing suffixes and prefixes, and also the meaning of short words, rare words, and even words that are not present in the vocabulary, since it learns from characters and not from words.

1.3.3.2 Context-dependent word embeddings

ELMo, which stands for Embedding from Language Models, is a word representation model developed by AllenNLP [Peters et al., 2018]. Following the principle of FastText, ELMo uses a character representation of the text as input to a Convolutional Neural Network (CNN), which is then passed to a two-layer Bidirectional Long Short Term Memory (Bi-LSTM) model. In this way, the embedding vectors contain information about a specific word and its context (the words before and after the specific word). One of the main advantages of ELMo is that it successfully addresses polysemy, which is not especially the case with FasText. An example of polysemy is

shown in "The head of my team was calling me this morning." and "He turned his head to get a better view of the painting.", where the word *head* does not have the same sense, since in the first sentence it refers to the boss of the team while in the second it refers to the body part.

GPT stands for Generative Pre-Training and it is a model developed by OpenAI [Radford et al., 2018]. Its architecture is based on an auto-regressive transformer decoder, where each token is predicted and conditioned on the previous token. Here, an encoder is not required because the previous tokens are received by the decoder itself. GPT learning consists of two steps, generative pre-training, and discriminative fine-tuning, both of which can be done following an unsupervised approach. One advantage of this model is its ability to generate long sequences of text without decreasing accuracy or coherence. However, its performance for classification tasks is comparatively inferior to tasks related to text generation.

BERT is a more recent model, similar to GPT, developed by Google AI [Devlin et al., 2018] that has shown positive results in different biomedical information extraction applications [Peng et al., 2019, Zhao et al., 2021, Nejadgholi et al., 2020, Chen et al., 2022a]. It learns through a transformer model the contextual token embeddings of a sentence bidirectionally, i.e., from both left and right. The architecture of the transformer model consists of a set of encoders and decoders and it is trained with the task of Masked Language Modeling to predict the original text. BERT was first pre-trained in an unsupervised way using English corpora from the general domain and, then, fine-tuned for a specific task using labeled data. This model and some of its variants will be described in more depth later.

1.4 Model formulation and Training

After obtaining the word vector representations through feature processing, the next step is to train a model for a specific information extraction task, such as biomedical NER (Section 1.4.1), biomedical RE (Section 1.4.2), and biomedical EE (Section 1.4.3).

1.4.1 Biomedical named entity recognition

Biomedical NER is a task focused on extracting keywords from biomedical texts to identify and classify biomedical concepts, called named entities, such as genes, proteins, diseases, treatments, and pathways. Figure 1.2[2] shows the example of a sentence annotated with four biomedical named entities, *GGP* (which stands for *gene or gene product*), *Organism substance*, *Cancer* and *Organism*.

[2]The visualization of the annotated sentence is done using the visualization tool *brat (brat.nlplab.org/)*

Figure 1.2: Example of sentence annotated with biomedical named entities.

Biomedical NER methods can be mainly divided into four categories, rule-based (Section 1.4.1.1), dictionary-based (Section 1.4.1.2), ML-based (Section 1.4.1.3), and hybrid models (Section 1.4.1.4). However, the current state-of-the-art systems are especially focused on ML and hybrid models, where rules and dictionaries are combined with ML learning methods. In biomedical NER supervised learning is mainly used for ML approaches since they are based on manual annotations made by experts in the domain. Although semi-supervised and unsupervised learning is also used less importantly. The earliest approaches of biomedical NER include Support Vector Machines (SVM), Hidden Markov Models (HMM), and Decision Tree models, while the most recent are focused on deep learning (DL) models with sequential data, Conditional Random Fields (CRF), and, more recently, on prompts [Perera et al., 2020].

1.4.1.1 Rule-based models

The main principle of rule-based models is to use hand-crafted rules based on orthographic and morphological features to identify and classify named entities. For example, to identify proper names in a text in English, a simple rule would be to identify named entities starting with a capital letter. Therefore, the named entities usually present features like upper-case letters, symbols, digits, suffixes, and prefixes, that can be captured with regex expressions, for example. Rules also include POS features, where POS taggers can be used to fragment sentences to capture noun phrases that represent named entities.

Rule-based models are specially designed for cases where the entity categories and entity boundaries are well-defined, showing both high precision and recall. One of the first rule-based biomedical NER approaches is PASTA (Protein Activate Site Template Acquisition) [Gaizauskas et al., 2003], a system that heuristically classifies named entities. Here, the biomedical documents are first analyzed to identify the sections presenting technical text. This text is then split into tokens and analyzed to identify semantic and syntactic features and from this extract morphological and lexical features. A set of hand-crafted rules are created to identify and classify entities into 12 categories of technical terms.

However, purely rule-based models are not very numerous. Rather, the models use heuristic rules combined with dictionaries [Wei et al., 2012, Eftimov et al., 2017], since this has shown better results in performance. One advantage of combining these methods is that rules compensate for exact dictionary matches and dictionaries refine the results extracted with the rules. On the other hand, the main weak points of rule-based systems are the time-consuming process of

hand-crafting the rules looking to cover all patterns of interest and the null effectiveness in unseen terms.

1.4.1.2 Dictionary-based models

In the case of dictionary-based models, the principle is to use large databases of named entities belonging to different categories taken as references to locate and classify entities in text. One of the strategies followed by this approach is to scan the text for the exact matching of words with the terms in the dictionary to detect the named entities. However, even if this strategy is precise for biomedical NER, the recall tends to be lower.

To handle this issue the dictionaries have been expanded, increasing biomedical terms, their synonyms, spelling, and word order differences. Also, some systems use flexible or fuzzy matching, where they automatically generate extended dictionaries to take into account spelling variations and partial matches.

An example of a dictionary-based system is Polysearch [Cheng et al., 2008], an association mining tool. This approach combines several comprehensive dictionary thesauri to tag and normalize entities. Whatizit [Rebholz-Schuhmann, 2013] is an online available tool that annotates text with separate modules for different named entity types. This system is based on vocabularies extracted from standard databases. For example, WhatizitChemical uses a vocabulary from ChEBI[3] and OSCAR3[4], WhatizitDisease uses diseases terms from a vocabulary extracted from MedlinePlus[5], WhatizitDurgs uses a vocabulary extracted from DrugBrank[6] and WhatizitOrganism uses a vocabulary extracted from the NCBI taxonomy[7]. Another dictionary-based model is LINNAEUS [Gerner et al., 2010]. It recognizes and normalizes named entities of species and applies regex heuristics to solve ambiguities.

More recent tools have been developed under a hybrid approach using dictionaries since the combination of other methods together with dictionaries has shown improvement in performance. Besides, since the evaluation of these systems uses exact and fuzzy matching, the main requirement for high performance is to compose an extended dictionary with possible related terms.

1.4.1.3 ML-based models

Currently, ML models are the most frequently used for biomedical NER due to their positive results in performance. Overall, the systems based on ML or hybrid approaches combining ML with rule-based or dictionary-based models are the present state-of-the-art methods. ML can

[3] www.ebi.ac.uk/chebi/
[4] www-pmr.ch.cam.ac.uk/wiki/Oscar3
[5] medlineplus.gov
[6] go.drugbank.com
[7] www.ncbi.nlm.nih.gov/Taxonomy/taxonomyhome.html/

follow three methodologies, supervised, semi-supervised, and unsupervised learning.

In supervised learning the training of the models is done using labeled data. The first supervised ML methods were based on SVM [Makino et al., 2002], HMM [Shen et al., 2003], Decision Trees, and Naive Bayesian methods [Nobata et al., 1999]. CRF [Lafferty et al., 2001], which considers the probability of contextual dependency words, is a method that has been considered as a milestone of these models. Here, the independence assumptions made by Bayesian inference and directed graphical models are shifted away. CRFs are a case of conditionally-trained finite-state machines, where the final result is a statistical-graphical model that has shown high performance in sequential data. It makes it a good option for tasks of language modeling, such as NER.

DL [LeCun et al., 2015, Emmert-Streib et al., 2020] or neural network systems have become state-of-the-art models in several information extraction tasks, including biomedical NER [Furrer et al., 2019, Zhu et al., 2018]. Among the first DL architectures used for extracting biomedical entities are Feed Forward Neural Networks (FFNNs), Recurrent Neural Networks (RNNs), and CNNs.

FFNNs are the earliest neural network models, introduced by Bengio et al. [Bengio et al., 2000], who focused on "fighting the curse of dimensionality". This model learns from a distributed continuous space of word vectors to estimate the conditional probability of each word appearing with the others. However, one constraint of this model is that it is limited to pre-defined contextual information and it is not compatible with timing and sequential information, so language cannot be represented as a sequence of words but as a probable word space. Figure 1.3 shows the simplest form of an FFNN, a single-layer perceptron.

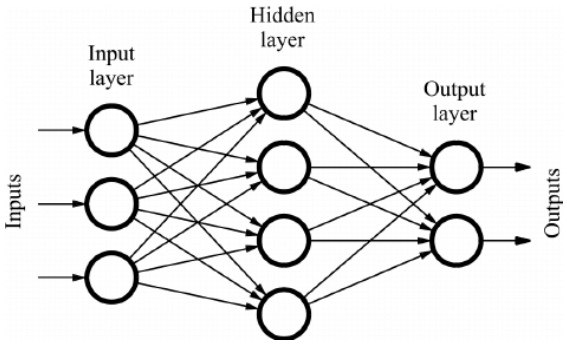


Figure 1.3: Example of an FFNN model (Figure taken from www.researchgate.net/figure/Sample-of-a-feed-forward-neural-network_fig1_234055177).

On the other hand, CNNs are models used to extract contextual information from word embeddings. Figure 1.4 shows an example of the structure of a typical CNN, where two convolutional layers apply a filter to the input data, and a fully connected layer is added at the

end for the prediction task. A pioneering application in language modeling with CNNs was introduced by Kim et al. [Kim et al., 2016]. This approach involves representing English words as character embeddings, which are subsequently inputted into a CNN. The neural model filters the embeddings to create feature vectors that represent the words. This work was later extended to the biomedical domain by Zhu et al. [Zhu et al., 2018], who obtained and combined the embeddings of characters, words, and POS tags from biomedical text. The embedding vectors are fed to a CNN with multiple filters, giving as output a vector that represents the local features of each term. This output is finally tagged using a CRF layer.

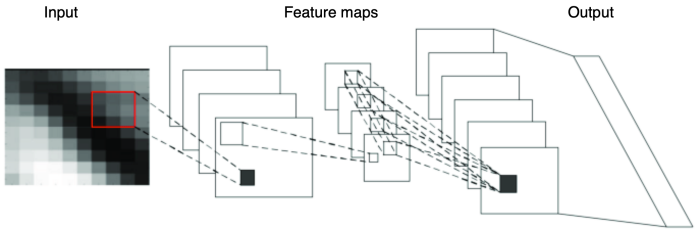


Figure 1.4: Example of a CNN model (Figure taken from www.researchgate.net/figure/Structure-of-a-typical-convolutional-neural-network-CNN_fig3_323938336).

RNNs were also explored to represent language as a collection of sequential tokens. LSTM, a variation of RNNs, is the most frequently used for modeling language. The architecture of LSTMs contains memory cells, which can learn long-term dependencies by retaining information long time and controlling which inputs and outputs are preserved or forgotten in the memory. Figure 1.5 shows an example of the structure of a LSTM model with three units. Each LSTM unit is composed of three gates. The first of the three gates, called the forget gate is the one that decides which information is discarded from the current internal state using a sigmoid function. The second gate called the input gate, updates the internal state with new information using a sigmoid function and a Tanh function combined. The last gate, called output gate, creates the hidden representation to be passed to the next step based on the current internal state.

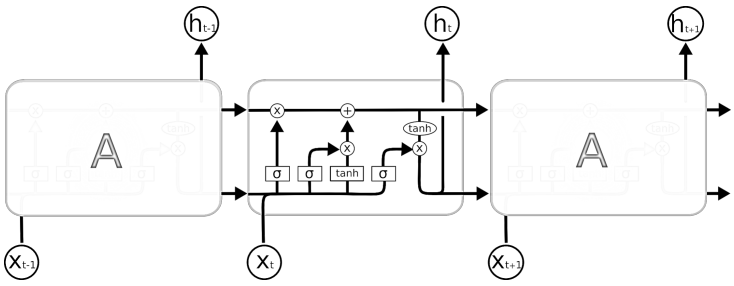


Figure 1.5: Example of a LSTM model (Figure taken from www.linkedin.com/pulse/look-rnns-lstm-gru-attention-mechanism-ayushee-mittal).

Bidirectional LSTMs (Bi-LSTMs) are an extension of LSTMs that learn from past and future information, instead of only learning from past data as in LSTMs. This bidirectional learning helps to better enrich and give more freedom for the development of a contextual language model [Li et al., 2016]. Later, different works [Habibi et al., 2017, Yoon et al., 2019, Wang et al., 2019, Giorgi and Bader, 2020, Weber et al., 2020] proposed to combine Bi-LSTMs and CRFs using the embeddings of words and characters showing positive effects in performance. The main idea behind these works is to first obtain the word embeddings from a pre-trained model and a Bi-LSTM to obtain the character embeddings for each word sequence. Both embeddings are combined and given as input to a Bi-LSTM, followed by a CRF layer. The objective of the CRF layer is to classify each word based on an IOB (Inside-Outside-Beginning) approach. An example of this approach is shown below.

's- **VEGF** in **peripheral blood samples** was analyzed in 40 **RCC** ...'

'O' 'B-GGP' 'O' 'B-Org_subs' 'I-Org_subs' 'I-Org_subs' 'O' 'O' 'O' 'O' 'B-Cancer'

The words that appear in bold in the sentence are the named entities identified and classified into the entity types shown in the labels below the words. For example, the label *B-Org_subs* of the word "peripheral" indicates that this word is the beginning of the named entity of the type *Org_subs*. Then, the labels *I-Org_subs* of the subsequent words "blood" and "samples", indicate that these words are inside of the same named entity of the type *Org_subs*. Therefore, it results in a multi-word named entity of the type *Org_subs*. The words in the sentence that are not in bold are labeled as *O*, indicating that they are not a named entity.

The most recent models developed for biomedical NER are based on transfer learning, a method where a model developed for a task is used to improve the generalization on a different task [Pan and Yang, 2010]. One of the main advantages of this approach is that it has shown positive performance, particularly for cases where the labeled data is reduced [Symeonidou et al., 2019]. BERT [Devlin et al., 2018] is one of the earliest models using this approach, achieving outstanding results in several NLP tasks, including NER.

The architecture of BERT has been used to develop similar models that are specific to the biomedical domain, such as BioBERT [Lee et al., 2020]. This model was trained on PubMed abstracts and PMC (PubMed Central)[8] articles for different tasks, achieving high performance in biomedical NER on multiple benchmarks [Symeonidou et al., 2019]. Transformer models and other BERT-derived models that have been developed specifically for the biomedical domain are described in more detail in Chapter 2.

Semi-supervised learning is an approach mostly used when the amount of unlabeled data in the available data collection is larger than the labeled data, which is very often the case in the

[8] www.ncbi.nlm.nih.gov/pmc/

biomedical domain. Here, the tasks are extended to the development of a model that maps the unlabeled data to the labels provided by the labeled data. An example of this approach is the work of Munkhdalai et al. [Munkhdalai et al., 2015], who used it to incorporate chemical and biomedical knowledge into the BANNER [Leaman and Gonzalez, 2008] system for NER. The system runs in two parts, in the first part it processes the labeled data using NLP techniques to extract rich features, i.e., word and character n-grams, lemma, and orthographic information. In the second part, it pre-processes the unlabeled data and extracts word representations using hierarchical clustering and Word2Vec. Then, both representations extracted from the labeled and unlabeled data are used to train a CRF model.

On the other hand, unsupervised methods are used when only unlabeled data is available. This learning presents the advantage of organizing unseen information without previous processing. Following this approach, Zhang et Elhadad [Zhang and Elhadad, 2013] proposed a system based on seed knowledge and signature similarities between entities. The seed knowledge is taken from the seed concepts, semantic types, and semantic groups of the UMLS (Unified Medical Language System)[9] for each entity type, e.g., cell types, DNA, RNA, and proteins. This information represents the domain knowledge. This candidate corpus is processed to obtain word representation vectors based on a clustering technique, which is later used to generate the signature similarity vectors for each entity class. The intuition behind these vectors is that the same class has contextually similar words. They finally calculate the similarities of the candidate signatures of the named entities and the signatures of the entity class for comparison. Similarly, Sabbir et al. [Sabbir et al., 2017] proposed an approach for a word disambiguation system using a knowledge base of concepts from the UMLS. Their results showed that unsupervised learning can deal with ambiguous biomedical entities.

1.4.1.4 Hybrid models

Hybrid models combine rule-based models, dictionary-based models, and ML-based models to gain performance. Most of the models combine ML with either dictionaries or rules, showing an improvement, especially in recall. OrganismTagger [Naderi et al., 2011] is a hybrid model that uses binomial rules of species to tag organisms. It uses also an SVM model to capture the organism names that do not follow these rules for optimization. Following this idea, SR4GN [Wei et al., 2012] is a system that uses rules to tag species and then, a dictionary to evaluate the accuracy of the entities tagged. Other tools based on hybrid models are Gimli [Campos et al., 2013], Chemspot [Rocktäschel et al., 2012] and DNorm [Leaman et al., 2013], which use CRFs with a thesaurus of taxonomy, while OGER++ [Furrer et al., 2019] uses FFNNs and a dictionary and Soomro et al. [Soomro et al., 2017] use ML models and rules.

[9] www.nlm.nih.gov/research/umls/index.html.

1.4.2 Biomedical relation extraction

Biomedical RE is the task that usually follows NER. The main objective is to identify the association between the biomedical named entities. Among the principal interests of the ongoing biomedical RE research is to find the roles of genes and the interaction of proteins in different biological processes. Figure 1.6 shows the example of a sentence annotated with two directed relations of the type *Inhibitor*. These relations represent the fact that the word "benzamidine", annotated as a *Simple_chemical* named entity *inhibits* the two named entities of the type *GGP* (which stands for *Gene_or_gene_product*).

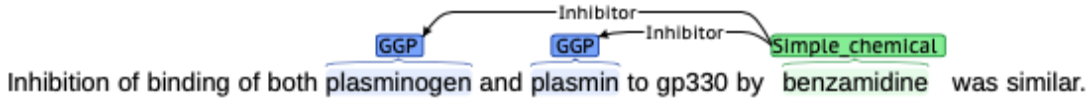


Figure 1.6: Example of biomedical RE (Figure taken from [Mahendran et al., 2021]).

The techniques used to extract relations are classified into three main categories, co-occurrence-based models (Section 1.4.2.1), rule-based models (Section 1.4.2.2), and ML-based models (Section 1.4.2.3).

1.4.2.1 Co-occurrence-based models

Approaches based on co-occurrence are the earliest and simplest models for RE. Its principle is to consider that entities are associated if they occur together in the text. The more frequently the entities occur together, the higher the probability that they are associated. For example, Chen et al. [Chen et al., 2008] introduce a method based on co-occurrence statistics to calculate and evaluate the associations between diseases and relevant drugs from clinical text and biomedical literature.

1.4.2.2 Rule-based models

Models based on rules use the syntactic and semantic information of the text for RE. For example, after applying POS tagging, these methods can look for verbs and prepositions that correlate two nouns or noun phrases labeled as named entities to identify an association between them. Figure 1.7 shows the syntactic parse tree of a simple sentence. The arrows connecting the words represent the syntactic dependency between them and below the words are shown the POS tags. The work of Fundel et al. [Fundel et al., 2007] proposes to use syntactic parse trees to identify in the sentences noun phrases and a verb that associates them, indicating a relationship. An example of a list of verbs can be considered to show implications between nouns, e.g., a list of verbs including "regulates", "increases", "influences", and "catalyzes".

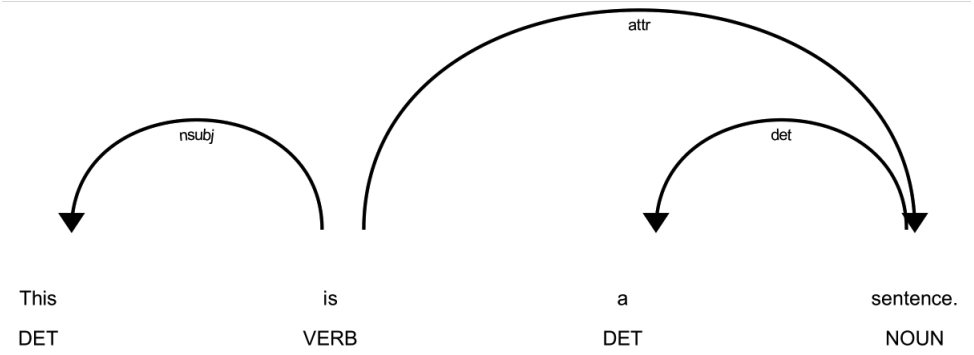


Figure 1.7: Example of syntactic parse tree (Figure taken from spacy.io/usage/visualizers).

1.4.2.3 ML-based models

These models, as in the case of NER, are mainly used in a supervised manner. An annotated corpus with pre-defined relations is often used to learn the model to extract relations. However, the main challenge of this approach is to acquire the annotated data. The datasets that are currently considered as baselines for the development of these models have been generated by BioCreative and BioNLP competitions [Sahu et al., 2019, Van Mulligen et al., 2012, Shardlow et al., 2018]. Which have been of great support to develop more accurate and effective models of biomedical RE.

SVMs were the first ML models to be used primarily for biomedical RE as they showed high performance on text classification tasks and a low tendency to overfit. The crucial process in this model is feature selection since accuracy and relationship mining depend on features. One of the earliest methods using SVM [Özgür et al., 2008] proposed to combine different methods to evaluate and find an appropriate kernel function to predict gene-disease relations. The kernel function is based on a similarity measure of the distances between the paths of two genes in a dependency parse tree. Later, it is proposed to extend the last study using an ensemble of SVMs trained in small samples of data [Bhasuran and Natarajan, 2018]. They included a Word2Vec representation combined with semantic and syntactic features, which improved the performance of extracting disease-gene associations.

Other traditional ML models used for biomedical RE are Naive-Bayes [Jensen et al., 2014], Max-entropy based classifier with Latent Dirichlet Allocation (LDA) [Quan et al., 2016] and CRF [Bundschus et al., 2008].

DL approaches have become the most used for biomedical RE since they do not require complicated feature processing and present positive results in performance. The most widely used DL methods include CNNs, RNNs, transformers, and hybrid models of these models combined [Jettakul et al., 2019, Zhang et al., 2019, Hong et al., 2020, Raza and Schwartz, 2023]. The information given as input to the models includes sentence-level, word-level, and lexical-level

features, which are represented as vectors, and the position of the named entities. The vectors are calculated from a pre-trained word and positional vector space obtained from single or multiple corpora. Also, the information on the shortest path between entities from dependency graphs, POS tags, and chunk tags are commonly used as input features. The first approaches use CNNs having as input features the dependency paths of the sentences represented with a vector word space. Since these models require that all the input samples present the same size in the training, the instances with a smaller size are padded with zeros. CNN layers are followed by a pooling layer, and the output is used as input to a classification layer, which is usually an FFNN layer with soft-max activation function [Hua and Quan, 2016].

LSTM models, including Bi-LSTM, have also been used to learn features of sentences. One of the advantages of these models compared to CNNs, is that they do not require all the samples at the input to have the same size. In addition, they work well with long sentences since the input is sequentially processed, which helps to associate named entities that are far from each other in the text. The work of Hsieh et al. [Hsieh et al., 2017] propose to extract drug-drug and protein-protein interactions using Bi-LSTM. Then, Zhang et al. [Zhang et al., 2018] extend the last work comparing different architectures using two hierarchical layers. The first architecture uses two simple RNNs, the second one two Gated Recurrent Units (GRUs), and the last one two Bi-LSTMs, being the model with the best performance. The work of Zheng et al. [Zheng et al., 2018] proposes a Convolutional Recurrent Neural Network (CRNN) to extract chemical-disease relations. CRNN is a hierarchical hybrid model composed of a CNN followed by a soft-max and two Bi-LSTM layers. One of the main interests of using a hybrid architecture is to combine the CNN's ability to learn local lexical and syntactic features in short sentences and the RNN's capacity to learn dependency features in long sentences with complicated sequences of words.

In addition to RNNs, Graph Neural Networks (GNN) have also been used to encode syntactic and semantic relationships preserving the structure of graphs. These models are designed to operate on graphs looking to capture dependencies by passing messages between the nodes. One advantage of GNNs is that they can cover deeper contextual information in the nodes since the features are calculated from all the neighboring nodes. Graph Convolutional Networks (GCNs) [Kipf and Welling, 2016] are a more recent approach for RE based on GNNs and CNNs, that use graphs as input, representing the text and preserving its structure. Figure 1.8 shows the example of a GCN model with two hidden layers. In graphs, named entities represent the nodes and syntactic or semantic structures that connect them are the edges. An advantage of this representation is that it can identify patterns that indicate significant associations. The work of Zhao et al. [Zhao et al., 2019] presents a model that combines GCNs and a Bi-GRU layer used to identify drug-drug interactions, finding that it outperformed other similar methods.

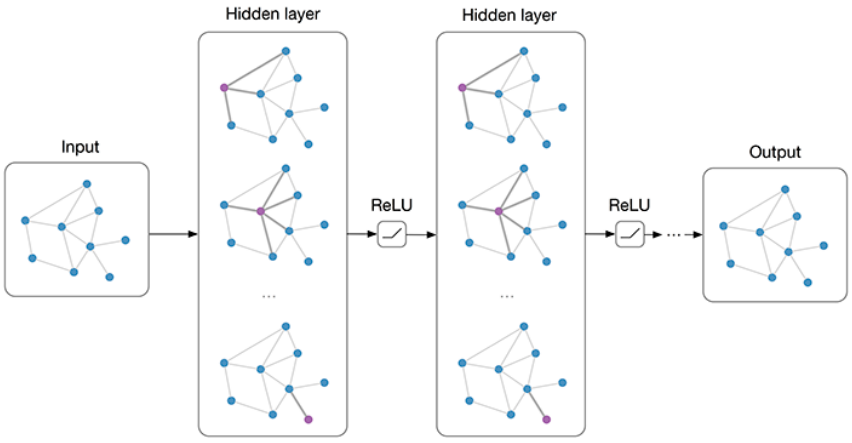


Figure 1.8: Example of GCN model (Figure taken from tkipf.github.io/graph-convolutional-networks/).

1.4.3 Biomedical event extraction

Biomedical EE aims at detecting the existence of events in the text and, if it is the case, to discover the semantic-related information that commonly answers questions such as *who*, *when*, *where*, *what*, *why* and *how*. According to Frisoni et al. [Frisoni et al., 2021], an event can be defined as *a specific occurrence of something that happens and involves an arbitrary number of attributes and participants covering a specific semantic role, depending on the event type. The interaction (i.e., dynamic relation) modeled by an event represents or leads to some state change.*

The main purpose of extracting biomedical events is to be able to have detailed information on the behavior of biomolecules and their constant changes that are spread in the literature. Among the biomedical events that can be extracted are expression, transcription, localization, binding, phosphorylation, and regulation.

The structure of an event consists of three main elements, an event trigger, one or more event arguments, and their corresponding argument roles. The event trigger is a textual mention that expresses the occurrence of an event, being commonly a verb or a noun. Event arguments are named entities serving as participants or attributes with specific roles in the events. These argument roles are the semantic relationships between the arguments and the event in which they participate [Xiang and Wang, 2019]. Events can be also taken as arguments of other events, called nested events, while if the arguments consist only of biomedical named entities, they are called flat events.

Figure 1.9[10] shows the example of a sentence containing two biomedical events, *-Reg* (which stands for *Negative regulation*) and *Locl* (which stands for *Localization*). The event constructed from the trigger word "excretion" of type *Locl* (the event is given the same type as the trigger)

[10]The visualization of the annotated sentence is done using the visualization tool *brat (brat.nlplab.org/)*.

presents as a single argument the biomedical entity of type *D/C* (which stands for "Drug or compound"), playing the role of *Th* (which stands for *Theme*). This role allows answering the question "What is excreted?". On the other hand, the event constructed from the trigger word "reduces" of the type *-Reg*, presents two arguments. The first argument is a biomedical entity of the type *Drug or compound*, playing the role of *Cause*. This role allows answering the question "What causes the reduction?". The second argument is the nested event *Locl* described before, playing the role of *Theme*, answering the question "What is reduced?".

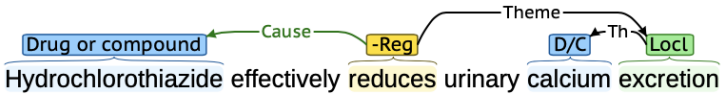


Figure 1.9: Example of event extraction; the *-Reg* (*negative regulation*) event has the *Locl* (*localization*) nested event as argument.

Biomedical EE and RE are tasks that share some similarities, such as extracting information from the text and transforming it into structured relational knowledge. However, RE focuses on discovering semantic information between two named entities forming directed or non-directed binary relations. In contrast, EE allows finding n-ary relations between a trigger and its arguments (named entities or other triggers).

Figure 1.10 shows the example of two biomedical sentences (a) and (b) annotated with relations (left side) and events (right side). In sentence (a) the RE system is not able to capture essential contextual information, such as "over-expressed", which is a condition for the relation of the type *+Reg* to be true. While the EE system considers the context words to build the events. In sentence (b) RE builds two binary relations that provide incomplete information, while EE builds a single event with three arguments, taking into account the context to indicate the role of each argument.

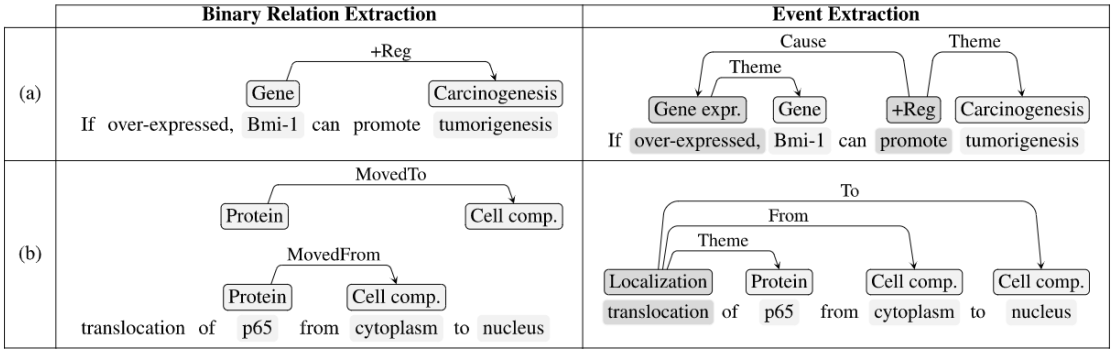


Figure 1.10: Comparison between biomedical RE and biomedical EE (Figure taken from [Frisoni et al., 2021]).

EE is usually divided into three main subtasks, event trigger detection, argument identification, and event construction.

Event trigger detection identifies and classifies the trigger words into a set of predefined types of event triggers, while argument identification identifies and classifies the roles between the event triggers and their respective arguments [Shen et al., 2019]. Finally, event construction refers to the unmerging of the arguments that correspond to the same event for its construction [Björne and Salakoski, 2011].

Event trigger detection has a fundamental role in the construction of events. Indeed, the triggers are the targets that allow us to know that an event may exist [Cui et al., 2020]. This subtask is usually considered a classification problem, where each word needs to be classified into a predefined set of trigger types. The difficulty in detecting triggers comes from the sensitivity to the domain or subdomain (text can present very specialized language), linguistic forms (triggers can be single words, multi-words, discontinuous markers), and ambiguity on the trigger class (a trigger word can be given different trigger classes) [Zerva and Ananiadou, 2015]. Among the solutions to tackle these problems, it is proposed to include extra features to provide syntactic information about the text, such as the parts-of-speech (POS) tags (token function in meaning and grammar within the sentence), which have been demonstrated to help detect event triggers [Shen et al., 2019].

Argument identification is also considered a multi-category classification problem, where the directed relation between a trigger and an entity or other event needs to be classified into a predefined set of role types. One of the main complexities in identifying arguments is that they can be part of one or multiple events (one-to-one and one-to-multiple relations), where they play the same or different roles. Following the work of Ramponi et al. [Ramponi et al., 2020], event trigger detection is the main source of errors in event extraction, where around 31 % of the errors correspond to non-detection of triggers and 28 % to over-detection of triggers. Further, the non-detection of arguments represents around 23 % of errors, and the over-detection of arguments around 7 %.

Various models have been developed to address these subtasks and attempt to overcome these errors. They can be classified into three categories: rule-based models (Section 1.4.3.1), ML-based models (Section 1.4.3.2), and hybrid models (Section 1.4.3.3).

1.4.3.1 Rule-based models

Analogously to biomedical NER and biomedical RE the earliest models attempting biomedical EE used human-crafted templates to match the events. Templates are constituted of rules or patterns that encode domain and linguistic knowledge, specific lexical items, and semantic classes. Following this approach, KrakeN [Akbik and Löser, 2012] identifies n-ary facts using hand-written

extraction rules based on dependency parses. Similarly, Exemplar [Mesquita et al., 2013] is a system to detect relations between triggers and multiple arguments with their corresponding roles. Later, it was introduced NESTIE [Bhutani et al., 2016] to identify nested relations using manually designed rules based on dependency parse trees. Then, MONTEE [de Vroe et al., 2021] was introduced to identify the modality of events (i.e. that indeed took place or did not take place) and their degree of uncertainty. Their system is based on a modality lexicon and a set of dependency graph patterns.

Rule-based models exhibit high extraction accuracy when rules are meticulously refined. However, they come with drawbacks. They tend to be costly to create and upkeep. They lack flexibility and struggle with generalizability across different domains, potentially limiting their ability to handle semantics.

1.4.3.2 ML-based models

Most of the biomedical EE methods are developed based on ML models, mainly following supervised learning. The ground truth labels used to train the models are obtained from manual or semi-automatic annotated corpora. Among the information that can be extracted from these annotated data are the lexical (e.g., n-grams, lemmas, POS tags), syntactic (e.g., dependency parsing relations), and semantic (e.g., synonyms and event/entity types) features.

The earliest ML approaches adopted pipeline architectures and classifiers based on SVM. The first version of the Turku Event Extraction System (TEES-SVM) [Björne and Salakoski, 2011], one of the first state-of-the-art biomedical EE models, reformulates biomedical EE as a graph construction problem decomposed in sequential nodes and edges that are classified with SVMs. EventMine [Miwa and Ananiadou, 2013] is another system based on SVMs that obtained the first and second ranks in two biomedical tasks from BioNLP'13 [Pyysalo et al., 2015]. In the case of [Liu et al., 2013b], they decomposed EE into a sequence of binary classification problems of relations extracted (using a (trigger, argument) structure). When the events presented multiple arguments, they multiplied the number of samples by the number of arguments and added a fusion step to extend the tuples.

More recent works use DL models, which present the advantage of automatically extracting features that in traditional ML models need to be manually defined. In the work of Nguyen and Grishman [Nguyen and Grishman, 2015] was introduced a pioneering approach that employs CNNs with non-consecutive convolution for event trigger detection, spanning various domains. The advantage of this model is that it addresses the limitation of CNNs, which tend to model words consecutively, potentially overlooking non-consecutive words that hold significance for event detection. Then, TEES-CNN was introduced [Björne and Salakoski, 2018], an upgraded version of TEES-SVM that incorporates CNNs and word embeddings to extract biomedical events. Wang

et al. [Wang et al., 2017a] proposed also a CNN model with multiple distributed features that include semantic information and the word-entity distance in a dependency tree.

RNNs have also been widely used for biomedical EE. In the work of Nguyen et al. [Nguyen et al., 2016], it was developed an RNN model based on Bi-GRUs. Similarly, Jagannatha et al. [Jagannatha and Yu, 2016] employed Bi-LSTMs and Bi-GRUs for extracting event triggers from Electronic Health Records (EHR). They utilized a skip-gram model trained on biomedical data to acquire word embeddings and then compared the system's performance with that obtained using CRF. Additionally, Rahul et al. [Rahul et al., 2017] proposed the use of RNNs for detecting biomedical event triggers. They integrated word and entity type embeddings as features, without relying on any hand-designed features. Given that event trigger detection is typically done at the sentence level, some works [Duan et al., 2017, Zhao et al., 2018] investigated enhancing context information by incorporating the complete document representation. Both studies advocate employing RNNs to extract cross-sentence features, all without relying on external resources.

On the other hand, Nguyen and Grishman [Nguyen and Grishman, 2018] introduced a GCN model that exploits syntactic dependency relations, using dependency trees to link words to their informative context for event detection. In Yan et al. [Yan et al., 2019], it is proposed a GCN model that integrates aggregative attention to effectively model and aggregate multi-order syntactic representations of sentences. In the case of Cui et al. [Cui et al., 2020], they augmented the GCN with a relation-aware concept, which leverages syntactic relation labels to model the connections between words.

In addition to conventional neural network models, the integration of pre-trained language models based on transformers has become common in event detection due to their ability to enhance the performance of existing systems. DeepEventMine [Trieu et al., 2020] stands as an end-to-end event extraction system composed of four primary modules: BERT model, trigger and entity detection and classification, relation extraction, and event identification. BERT serves as the foundational model for each module, supplemented by the addition of a linear layer. A key objective of this system is to enhance the extraction of nested events, a feat it accomplished by achieving a new state-of-the-art performance across seven biomedical nested event extraction tasks. The work of Portelli et al. [Portelli et al., 2021] proposed a comparison between six transformers models, BERT and five of its variants, for the identification of Adverse Drugs and Events (ADEs). They showed that span-based pre-training, from spanBERT [Joshi et al., 2020], provides an improvement in the recognition of ADEs, and that the pre-training of the models in the specific domain is particularly useful in comparison to training the models from scratch. In the case of Ramponi et al. [Ramponi et al., 2020], they developed BEESL, a neural network model based on a sequence labeling system for the extraction of events. Their system converts the event structures into a format of sequence labeling and uses BERT as a language model. Also, Chen [Chen, 2021] proposed the Multi-Source Transfer Learning-based Trigger Recognizer system,

which is an extension of transfer learning using multiple source domains. All the datasets from the different domains are used to jointly train the neural network, achieving a higher recognition performance in the biomedical domain, and showing a wide coverage of events.

1.4.3.3 Hybrid models

Various hybrid models integrating different approaches for biomedical EE have been proposed. For instance, Liu et al. [Liu et al., 2013a] implemented exact and approximate subgraph matching techniques to search for isomorphisms between graph-based patterns and sentence dependency graphs, enabling the extraction of biomedical events. Furthermore, Wang et al. [Wang et al., 2018] introduced a Bi-LSTM with CRF model for detecting biomedical event triggers. In the work of He et al. [He et al., 2021], event trigger detection was divided into two stages. They used a Bi-LSTM model with an attention mechanism for event trigger detection, coupled with a passive-aggressive online algorithm for event categorization. Additionally, Abdulkadhar et al. [Abdulkadhar et al., 2021] leveraged graph-based kernels and lexico-syntactic patterns for event trigger detection and argument identification. This involved the combination of Multiscale Laplacian Graph (MLG) and Feature-based Linear Graph (FLG) kernels. These approaches conducted a comprehensive analysis of the structure of biomedical sentences across multiple scales, capturing topological relationships between event nodes and various token-based, sentence-based, parsing-based, and domain-specific features.

1.5 Post-processing

Post-processing serves as the final step in the information extraction pipeline, aiming to enhance the quality and precision of system output through various processes. This includes tasks like resolving abbreviation ambiguities, disambiguating classes and terms, and addressing issues with parenthesis matching. For instance, in biomedical NER, post-processing involves identifying co-referring terms of an untagged named entity and labeling them with the appropriate class. In the context of biomedical EE, post-processing can be employed to assess argument candidates of partial events, ultimately aiding in the completion of constructed events.

1.6 Current limitations of biomedical information extraction

There are many current limitations on biomedical information extraction systems. In the case of biomedical NER, one of the main challenges is the large number of synonyms, alternative expressions, or abbreviations with non-standard terms of the same word that induce the explosion of word vocabulary, e.g., the expressions *HIV-1 enhancer* and *HIV 1 enhancer* [Gridach, 2017]. Oppositely, the same word or sentence can refer to more than one type of named entity, causing

an ambiguity or polysemy problem. For example, the term *TNF alpha* can correspond to the *protein* type or the *DNA* type, depending on the context. Moreover, many entities are composed of long sequences of tokens, complicating the detection of boundaries [Zhao et al., 2021].

Biomedical RE systems encounter similar issues. The variation in the non-standard expressions of biomedical entities may represent a limitation in RE as in NER. Also, the availability of well-annotated biomedical data is reduced since it requires domain expertise for manual annotation. Therefore, it becomes difficult to have enough data for the training of more complex DL models.

Extracting biomedical events introduces further challenges, such as the existence of multiple events with the same word trigger but different types or the same argument participating in different events with different roles. Also, it requires capturing complex relations, i.e., n-ary and nested relations, connected to a trigger, defining multi-relational structures. These complex relations may involve long-range dependencies, where arguments are distant from each other or the trigger [Frisoni et al., 2021]. Moreover, the frequent use of linguistic modalities to express uncertainty in biomedical texts requires careful consideration for constructing accurate biomedical events. For example, in the phrase "We examined whether [...]" it can be understood that the subject is under examination and it is not an affirmation, while the phrase "The results suggest that [...]", expresses speculation [Zerva and Ananiadou, 2015]. Addressing these challenges is crucial for advancing biomedical information extraction and enabling more precise and comprehensive biomedical knowledge extraction.

Furthermore, the evolving nature of biomedical research and the continuous influx of new scientific literature pose additional challenges for biomedical information extraction systems. Keeping up with the latest advancements, terminology, and domain-specific knowledge requires regular updates and adaptations of these systems. Additionally, the integration of diverse data sources, such as electronic health records, clinical trial data, and biomedical literature, presents a need for interoperability and harmonization of information extraction approaches across different data modalities.

Despite these challenges, significant progress has been made in the development of biomedical information extraction systems. The combination of ML techniques, NLP algorithms, and domain-specific knowledge has shown promising results in extracting valuable information from biomedical texts. Continued research efforts and collaborations between domain experts, data scientists, and computational linguists hold the potential to further enhance the performance and applicability of these systems.

Chapter 2

Transformer language models for biomedical event extraction

Contents

Overview

Transformer models have shown an important impact on various NLP tasks, leveraging self-attention mechanisms and parallel processing to achieve notable results. With their ability to capture long-range dependencies and contextual information, transformer models have achieved remarkable success in tasks such as information extraction, sentiment analysis, and text generation. This chapter explores the fundamental principles, architecture, and training procedures of transformer language models, aiming to describe their implications in NLP and biomedical applications.

2.1 Pre-trained transformer language models

Pre-trained language models that are based on transformers, e.g., BERT, RoBERTa, and T5, have started a new approach in modern NLP. These models acquire the advantages of transformers,

transfer learning, and self-supervised learning.

Transfer learning refers to a model that transfers the knowledge learned from a source task into a new task. For example, in computer vision, models are trained using a large set of labeled images. Then, these pre-trained models are used for similar tasks when the datasets are reduced. Among the main advantages of reusing these pre-trained models is that they learn language representations that are useful across different tasks and also avoid having to train downstream models from scratch. However, in NLP it is difficult to train models using large sets of data since annotated databases are small. To solve this problem, transformer language models are pre-trained using self-supervised learning on large unlabeled datasets. Self-supervised learning consists of using labels that are created automatically based on the common patterns extracted from the input data, and not manually as in supervised learning. After the model is pre-trained over a large set of text, it can be used for different downstream tasks through a process of fine-tuning by adding task-specific layers.

This adaptable framework allows for efficient and effective utilization of pre-trained transformer models across a wide range of NLP applications. The next section delves into the foundational principles of transformers, providing a deeper understanding of their inner workings and capabilities.

2.1.1 Foundation principles of transformer models

Transformer models, such as BERT, comprise two fundamental elements: the embedding layer and the transformer encoding layers. The embedding layer processes input tokens and produces a vector for each token. Within this layer, there are three or more sub-layers, each dedicated to generating a specific type of embedding for every input token. The initial sub-layer transforms the input tokens into a sequence of vectors. The subsequent sub-layers contribute supplementary details, such as the token's position within the input text. The final representation X for the given input tokens x_1, x_2, ..., x_n is calculated by summing the embedding from all the sub-layers, as shown in Equation (2.1), assuming that only three embedding types are summed, $I \in R^{n \times e}$, $P \in R^{n \times e}$ and $S \in R^{n \times e}$. Here $X \in R^{n \times e}$ represents the final input embeddings matrix, n represents the number of tokens and e represents the dimension of the embeddings.

$$X = I + P + S \tag{2.1}$$

This representation is then passed through a sequence of transformer encoder layers. Each transformer encoder layer consists of three key components: a Multi-Head Self Attention (MHSA) mechanism, a Position-Wise Feed Forward Network (PFN), and Add and Norm operations, as observed in Figure 2.1.

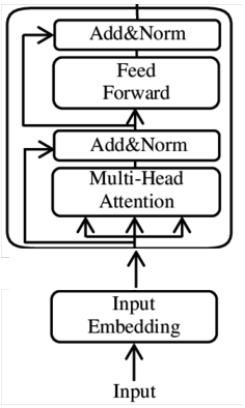


Figure 2.1: Example of transformer encoder.

MHSA independently applies self-attention multiple times in the input sequence to relate all the tokens to each other. Self-attention has shown to be a better alternative than convolution and recurrent layers to encode global contextual information [Kalyan et al., 2022]. In self-attention, each input token vector is updated with the contextual information applying a weighted sum of all the token vectors. These weights are obtained through attention scores. The input representation matrix X is then transformed into three matrices, Query ($Q \in R^{n \times q}$), Key ($K \in R^{n \times k}$) and Value ($V \in R^{n \times v}$), using three weight matrices $W^Q \in R^{e \times q}$, $W^K \in R^{e \times k}$ and $W^V \in R^{e \times v}$. Here $q = k = v = \frac{e}{h}$, where h represents the number of self-attention heads. If only one self-attention layer is used, the meaning of a word would mainly depend on the word itself. Then, by applying self-attention multiple times in parallel using the different weight matrices, as in MHSA, it is possible to take into account multiple positions while encoding a word.

PFN consists of two linear layers with a non-linear activation function, which may vary depending on the model. This step is applied to each input token vector, sharing the same set of parameters in all of them. Then, in the Add and Norm layer, Add represents residual connection and Norm represents layer normalization. Add and Norm are applied to both, MHSA and PFN, to avoid vanishing and exploding gradient problems. Therefore, for each transformer encoder layer, the input tokens are updated by the encoding of the global contextual information. In that way, the sequence of transformer encoders allows the models to encode deeper information. This encoding is done with Equation (2.2) and Equation (2.3), where LN refers to layer normalization, $\hat{E}_{m-1}$ represents the output after applying Add and Norm to the output from MHSA and E_m represents the output after applying Add and Norm to the output from PFN in the m^{th} encoder layer. Overall, E_m is the output from the m^{th} encoder layer with E_{m-1} as input, and the input for the first layer is $E_0 = X$.

$$\hat{E}_{m-1} = LN(E_{m_1} + MHSA(E_{m-1})) \tag{2.2}$$

$$E_m = LN(\hat{E}_{m-1} + PFN(\hat{E}_{m-1})) \tag{2.3}$$

2.1.2 Core concepts of transformer models

The main core concepts of pre-trained language models based on transformers correspond to the embedding vectors (Section 2.1.2.1), the pre-training methods (Section 2.1.2.2), and the fine-tuning methods (Section 2.1.2.3).

2.1.2.1 Embeddings

The embeddings are representations of data in a low-dimensional space. In pre-trained language models based on transformers, embeddings can be categorized into two types: main embeddings and auxiliary embeddings. Main embeddings are responsible for mapping the input sequence to a sequence of vectors, capturing the primary information. Whilst auxiliary embeddings provide supplementary and valuable information that enhances the understanding of the input data [Gopalakrishnan et al., 2019].

In the main embeddings, the mapping of the sequence of words into a sequence of vectors can be done based on characters (character embeddings), subwords (subword embeddings), or codes (code embeddings). Character embeddings or char-based embeddings, are designed to represent individual characters within a vocabulary, including letters, punctuation symbols, special characters, and numbers. These embeddings are initialized randomly and then refined through model pre-training. The primary benefit of character embeddings lies in their compact vocabulary size, which solely encompasses characters. However, a drawback is the increased duration of pre-training. As the sequence length expands when using character-level embeddings, the pre-training process becomes more time-consuming, resulting in slower model training. The subword embeddings involve constructing a vocabulary that includes characters, as well as the most common subwords and words. The principle of constructing a subword embedding vocabulary is to represent frequent words as complete words, while rare words are represented using meaningful subword units. The size of subword embedding vocabularies is typically moderate since they leverage subwords to capture infrequent and misspelled words. Several popular algorithms used for generating subword embedding vocabularies include Byte-Pair Encoding (BPE), Byte-Level BPE, Word-Piece, Unigram, and Sentencepiece. In the case of code embeddings, they are designed to encode sequences of codes into corresponding vectors. These embeddings are employed in models that are specifically tailored for scenarios where the input is not a sequence of words, but rather information represented as sequences of codes. The number of code embeddings incorporated in these models varies depending on the specific implementation. For instance, a model can focus solely on disease codes. Whereas another may include embeddings for a broader range of codes,

encompassing disease, medication, procedure, and clinical notes [Kalyan et al., 2022].

In summary, main embeddings serve as representations of the input sequence in a lower-dimensional space, capturing the essential information. Whereas auxiliary embeddings enrich the main embeddings by providing additional information to enhance the model's learning capabilities. To obtain a representation vector for each input token, the main embedding is combined with two or more auxiliary embeddings, allowing the model to leverage the comprehensive information encoded in these embeddings. Some examples of auxiliary embeddings are position embeddings, segment embeddings, age embeddings, gender embeddings, and semantic group embeddings. In the case of position embeddings, they enrich the representation of each token in the input sequence by incorporating its positional information. Since transformer models lack convolution or recurrence layers that can inherently capture token order, position embeddings are used to explicitly encode the token's position in the sequence. These embeddings can either be pre-defined or learned during the model's pre-training, allowing the model to effectively capture the positional relationships between tokens.

2.1.2.2 Pre-training

Pre-training based on self-supervised learning involves large volumes of unlabeled data. The pre-training allows the model to learn language patterns and representations that will be useful across tasks. Besides, it avoids initializing the training of other models from scratch and overfitting when the data is reduced [Zhao et al., 2021].

Pre-training methods can be classified into three categories, mixed-domain, domain-specific, and task adaptive. In mixed-domain pre-training, the model is trained using general and in-domain text. If initially the model was pre-trained over the general domain and then adapted to the biomedical domain, it is called continual pre-training. This is the approach generally adopted in the biomedical domain. For example, in the case of BioBERT, its pre-training is initialized with the BERT weights and then, further pre-trained on PubMed abstracts and PubMed Central (PMC) full-text articles. If both domains are combined for the whole pre-training, it is called simultaneous pre-training. This approach is especially used when the in-domain text is reduced. Generally, the in-domain data is over-sampled to ensure a balanced pre-training.

In domain-specific pre-training, the model is pre-trained using only in-domain data, increasing the in-domain vocabulary. For example, PubMedBERT was trained from scratch using PubMed abstracts and PMC full-text articles. Finally, in task adaptive pre-training the model is pre-trained using task-specific unlabeled data to learn both domain and task-specific knowledge. This type of pre-training is less expensive in comparison to the two last pre-training methods since it requires a relatively small corpus of unlabeled sentences related to the task.

During the pre-training process, models learn language representations based on the supervision

given by the pre-training task. This pre-training task is a pseudo-supervised task, where the labels are generated automatically. Depending on the approach, the pre-training tasks are classified into three main categories, word-level, phrase-level, and sentence-level. Among the most commonly used pre-training tasks are masked language modeling (MLM), replaced token detection (RTD), sentence boundary objective (SBO), next sentence prediction (NSP), and sentence order prediction (SOP).

MLM refers to predicting the missing tokens based on the left and right contexts. Here, for a given sequence x with the tokens x_1, x_2, ..., x_m, a subset of tokens are randomly chosen and replaced with a special token, e.g., $[MASK]$, to corrupt the input. RTD involves verifying if a token in the input was replaced or not. Here, some input tokens are replaced using words predicted from a generator network, and then the model acts as a discriminator, looking to predict the status of the words as replaced or not. These two approaches are developed at the word level. SBO, which is developed at the phrase level, consists of predicting an entire masked span based on the context. Initially, a span of tokens is randomly chosen and masked. Then, the model is asked to predict those masked tokens based on the token representations at the boundary. The main difference between this approach and MLM is that MLM predicts the masked token based on the final hidden vector of that masked token, while in SBO, the model predicts the masked token based on the final hidden vectors of the boundary tokens and the position embeddings of the masked token. In NSP the pre-training is done at the sentence level, involving the prediction of whether two sentences are consecutive or not. This classification task is also called a two-way sentence pair. SOP is also a sentence-level classification task, focused on modeling inter-sentence coherence. This approach is also a two-way sentence pair classification. Together with these pre-training tasks, the auxiliary pre-training tasks help to enrich models' knowledge using manually annotated in-domain sources. For example, the biomedical meta-thesaurus Unified Medical Language System (UMLS)[11] has been used to inject biomedical knowledge by further pre-training models in the domain, such as BioBERT and PubMedBERT.

Pre-training plays a crucial role in equipping the model with general or in-domain knowledge that can be applied to various tasks. However, to achieve optimal performance on a specific task, the model must acquire task-specific knowledge with the additional process of fine-tuning.

2.1.2.3 Fine-tuning

The task-specific knowledge is acquired through a process known as fine-tuning, where the model is trained on task-specific datasets. Pre-trained language models based on transformers incorporate task-specific layers on top of the pre-trained transformer. For instance, in text classification, the model requires both a contextual encoder to learn contextual representations of input tokens

[11] www.nlm.nih.gov/research/umls/index.html

and a classifier to project the final sequence vector and generate the probability distribution. The classifier acts as the task-specific layer, typically implemented as a softmax layer in text classification.

Fine-tuning methods can be broadly categorized into two approaches, intermediate fine-tuning (IFT) and multi-task fine-tuning. The IFT approach allows the model to learn knowledge specific to the domain or the task from large corpora and improve the performance on small target datasets. IFT can be done in four different ways, (1) same task different domain, (2) same task same domain, (3) different task same domain, and (4) different task different domain. Firstly the source and the target datasets are from the same task but correspond to different domains. The model can be first fine-tuned using a general domain dataset and then, fine-tuned on a small in-domain dataset. The second way uses the source and target datasets from the same task and domain. Thirdly the source and the target datasets are from different tasks but in the same domain. This allows the model to obtain more specific knowledge about the domain improving the performance on in-domain target tasks. For the last way, the source and the target datasets are from different tasks and different domains.

The multi-task fine-tuning approach allows the model to be fine-tuned on multiple tasks simultaneously. In this setup, the embedding and transformer encoder layers are shared across all tasks, while each task is associated with its dedicated task-specific layer. The use of multi-task fine-tuning enables the model to acquire both domain-specific and task-specific reasoning abilities by leveraging information from multiple tasks. Additionally, the larger training set resulting from the inclusion of multiple tasks helps mitigate the risk of overfitting. This fine-tuning approach proves particularly valuable in low-resource scenarios, which are frequently encountered in the biomedical domain. In addition, utilizing a single model for multiple tasks eliminates the need to deploy separate models for each task, leading to significant savings in computational resources, time, and deployment costs. However, it is important to acknowledge that multi-task fine-tuning does not guarantee optimal results in all cases. In situations where the initial fine-tuning does not yield the desired outcomes, an iterative approach can be employed, repeatedly applying multi-task fine-tuning to identify the most effective subset of tasks. This iterative process allows for the refinement and optimization of the model's performance [Kalyan et al., 2022].

Following these core concepts, different models of transformers have been developed for various purposes. One of the notable models that emerged from this progression is BERT, which introduced bidirectional context and pre-training on vast amounts of text data. The details of this model are described below.

2.1.3 BERT: a pre-trained transformer model

BERT [Devlin et al., 2018] is a contextualized word representation model based on a masked language model pre-trained with bidirectional transformers [Lee et al., 2020]. The architecture of BERT is based on the multi-layer bidirectional transformer model. It presents two model sizes, $BERT_{BASE}$ (shown in Figure 2.2), with 12 layers and 768 hidden size embeddings (110 million trained parameters in total), and $BERT_{LARGE}$, with 24 layers and 1024 hidden size embeddings (340 million parameters). Initially, BERT tokenizes the input text into words or sub-words to calculate the embeddings based on WordPiece, with a vocabulary of 30,000 tokens. The first token of each statement is a special classification token ($[CLS]$) and its final hidden state represents the aggregate sequence representation that is used for classification tasks. The sequence of these input tokens is constituted with initial vectors that are the combination of the token embeddings, the (token) position embeddings, and the segment embeddings (text segment to which the token corresponds) through element-wise summation. The embeddings of extra features can be computed and included in this summation, such as the POS embeddings (token function in meaning and grammar within the sentence) [Shen et al., 2019].

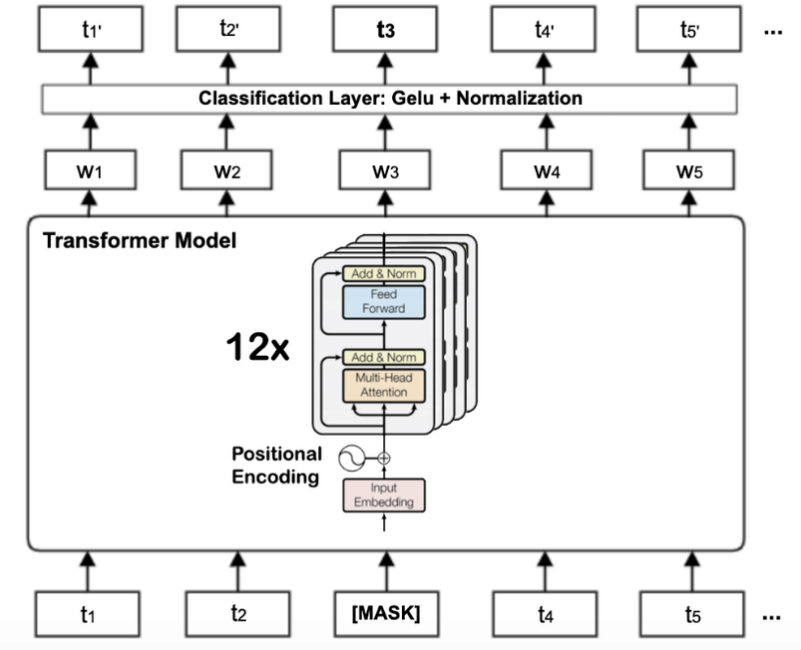


Figure 2.2: BERT architecture based on a transformer model (Figure taken from [Khalid et al., 2021]).

The embeddings are then passed to a set of layers of transformer modules. Figure 2.3 shows the example of a transformer encoding layer. It is mainly composed of two parts, an encoder and a decoder. The encoder processes the input word embeddings to generate encodings that

contain complex information from the inputs. These encodings are passed to the decoder layer, which uses the contextual information from the encodings to generate an output sequence. Each transformer layer generates a contextual representation of every token by summing the non-linear transformation of the tokens' representations from the previous layer. This representation is weighted by the attention calculated using the representations of the previous layer as a query. The last layer generates the contextual representations for all the tokens, where the information of the whole text span is combined [Gu et al., 2021].

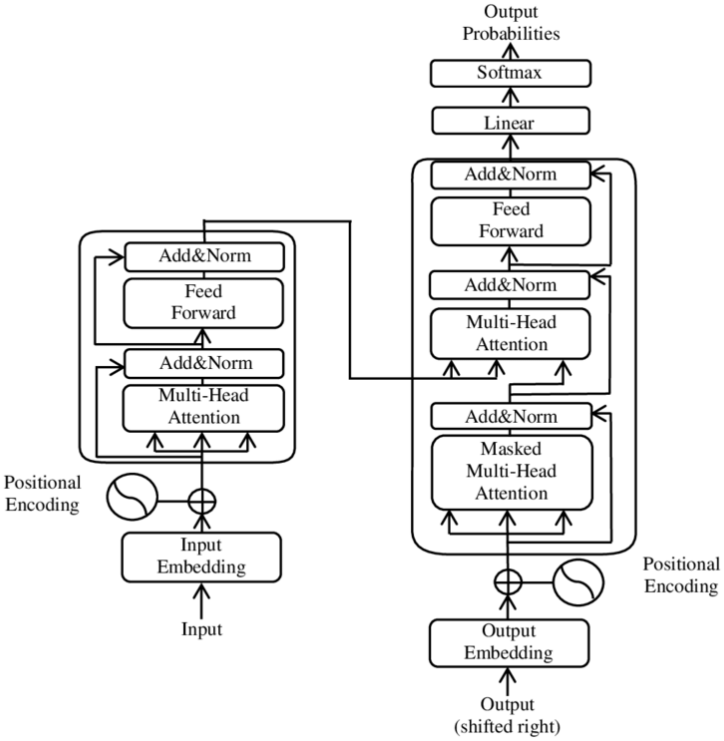


Figure 2.3: Example of transformer model (Figure taken from en.wikipedia.org/wiki/Transformer_(machine_learning_model)).

The framework of BERT is composed of two stages, pre-training, and fine-tuning, as shown in Figure 2.4. In the pre-training, BERT is trained on unlabeled data over two unsupervised tasks, MLM and NSP. In MLM, 15 % of the tokens are randomly masked of all WordPiece tokens in each sequence. Then, these tokens are replaced 80 % of the time with the $[MASK]$ token, 10 % of the time with a random token, and 10 % of the time the token is unchanged. The tokens that were replaced with $[MASK]$ are used to predict the original token using cross-entropy loss. In NSP, the model is pre-trained for a binary NSP task, where choosing sentences A and B for each sample, 50 % of the time B is the next sentence following A and 50 % of the time it is a random sentence. The data used for this pre-training correspond to the BookCorpus [Zhu et al., 2015]

(800 million words) and English Wikipedia (2,500 million words).

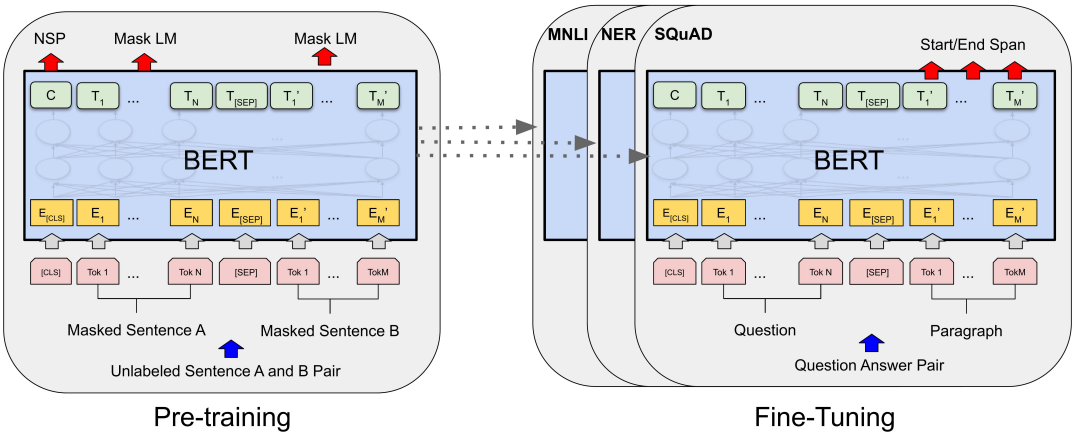


Figure 2.4: Representation of pre-training and fine-tuning of BERT (Figure taken from [Devlin et al., 2018]).

For the fine-tuning step, the model is initialized with the parameters from the pre-training which are fine-tuned with labeled data from downstream tasks, e.g., QA or NER. This step is relatively inexpensive in comparison to pre-training. Here, BERT uses the self-attention mechanism to encode the sequence of tokens and model downstream tasks. For each task, the model receives the task-specific inputs and fine-tune all the parameters end-to-end. The representation of the tokens is finally fed into an output layer relative to a specific task. For instance, the $[CLS]$ representation is fed to a classification layer for tasks such as sentiment analysis or RE. There is a specific fine-tuned model for each downstream task, even if all of them are initialized with the same pre-trained parameters. Therefore, the architecture of BERT may present minimal differences between the pre-trained architecture and the final downstream architecture, depending on the downstream task [Koroteev, 2021].

2.2 Biomedical language models based on BERT

Building upon the BERT framework, other transformer-based language models have emerged, pre-trained not only on general domain data but also on specific domains like biomedical data. This specialized pre-training allows for more effective adaptation to in-domain tasks within the biomedical field.

Intuitively, pre-training the models with biomedical data would help for biomedical applications. This has been shown by prior works, where pre-training using text from PubMed has allowed a better performance in biomedical NLP tasks [Peng et al., 2019, Beltagy et al., 2019, Lee et al., 2020]. Biomedical pre-trained language models present key differences from the original BERT model. For example, the hyperparameters specific to biomedical models have been fine-tuned

to optimize their performance on biomedical text data. These hyperparameters include batch size, learning rate, and the number of training epochs, tailored to the specific characteristics of biomedical text corpora. For instance, the input vocabulary of these models is expanded to include biomedical terms, such as medical entities, disease names, and gene/protein mentions, enabling better representation of biomedical concepts. This expanded vocabulary ensures that the models capture the nuances and domain-specific information present in biomedical texts. Additionally, biomedical models often leverage domain-specific external resources, such as biomedical ontologies or databases. These resources help enrich the models' contextual understanding, enabling them to capture the intricacies of relationships between biomedical entities and domain-specific knowledge.

BioBERT [Lee et al., 2020] was the first pre-trained transformer model that incorporated large-scale biomedical corpora for its pre-training, being initialized from BERT, as shown in Figure 2.5. This model uses WordPiece tokenization for the input in its pre-training to mitigate the OOV issue, since any new word can be represented with frequent subwords. Then, for its fine-tuning, it uses the base architecture of BERT requiring minor modifications for the different downstream tasks. However, it outperforms BERT in three main tasks in the biomedical domain, NER, RE, and QA. Where, for example, in the case of NER it directly learns WordPiece embeddings in both, pre-training and fine-tuning. This suggests that pre-training BERT on biomedical corpora indeed helps to better understand complex biomedical texts. After BioBERT more than 40 different biomedical models have been proposed, becoming the first choice for biomedical NLP tasks. Among the most used models for biomedical purposes are SciBERT [Beltagy et al., 2019], PubMedBERT [Gu et al., 2021], and BioMedRoBERTa [Liu et al., 2019].

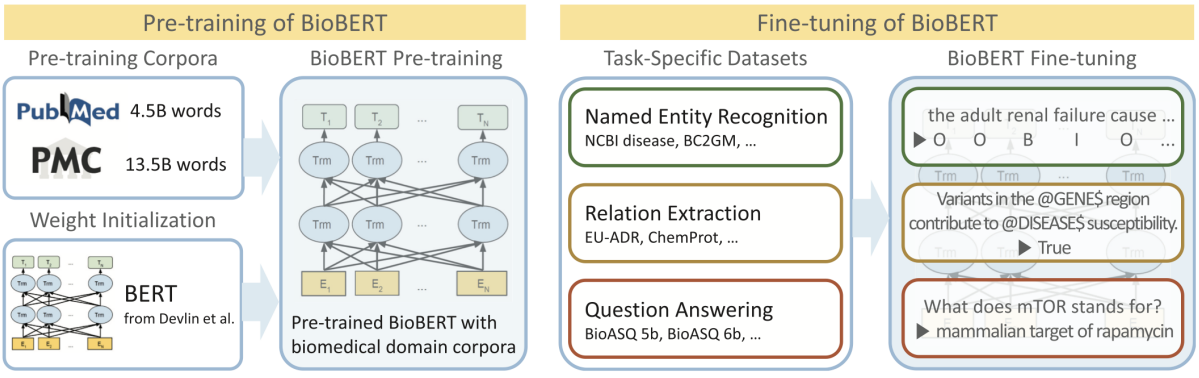


Figure 2.5: Overview of the pre-training and fine-tuning of BioBERT (Figure taken from [Lee et al., 2020]).

SciBERT [Beltagy et al., 2019] is another language model based on BERT, pre-trained on a large multi-domain corpus of scientific texts. This model uses a new vocabulary from scientific

text using WordPiece tokenization for the input. The corpus is a random sample of 1.14 million full scientific articles from Semantic Scholar[12] and PMC, where 18 % of them are in the computer science domain and 82 % are in the biomedical domain. SciBERT uses the architecture and configuration of BERT and starts its pre-training from scratch. For the fine-tuning, it follows almost the same architecture, optimization, and hyperparameters used in BERT for five NLP tasks, including NER, RE, and text classification. According to the task, minimal changes are applied, e.g., for text classification and RE, a linear classification layer is added on the top of BERT to classify the *[CLS]* token. While for NER, each token is classified using a linear classification layer with softmax output. In comparison to BERT, SciBERT obtains higher or very similar performance for the different NLP tasks in the general domain. While, in comparison to BioBERT, SciBERT obtains a higher or slightly lower performance in NER and RE for different biomedical corpus.

PubMedBERT [Gu et al., 2021] is a model pre-trained from scratch using a biomedical vocabulary from PubMed. It is pre-trained on a corpus that collects 14 million PubMed abstracts and it uses the same architecture of BERT. For fine-tuning, PubMedBERT uses hyperparameters tuned for each specific task and selects a set of common hyperparameters for the different datasets and the out-domain and in-domain language models. The performance of this model was evaluated alongside six other BERT-based models, including the original, across six distinct downstream tasks including NER, RE, QA, and document classification. Notably, this model demonstrated superior performance across most tasks, with the most significant improvement observed when compared to BERT-based models pre-trained exclusively on out-domain text.

BioMedRoBERTa [Gururangan et al., 2020] is a domain-adaptive pre-training approach based on ROBERTA [Liu et al., 2019], which uses the same architecture as BERT but different hyperparameters and training data size. Here, ROBERTA had a second phase of pre-training on an unlabeled large corpus of biomedical text. This supplementary pre-training was also extended to three other domains: computer science, news, and reviews. Within the biomedical domain, the model's performance was evaluated on two classification tasks: RE and abstract sentence roles. Additionally, an alternative strategy was explored for scenarios where resources for domain-adaptive pre-training were not accessible, opting for a task-adapting pre-training approach. They show the effectiveness of adapting the model to a task corpus using simple data selection techniques. The main contribution of this model is to show that multi-phase adaptive pre-training allows to gain in task performance significantly.

[12] www.semanticscholar.org

2.2.1 Biomedical event extraction using transformer models

Pre-trained language models based on transformers are often involved in the extraction of events since they have been shown to improve the performance of other current systems. In the work of Wang et al. [Wang et al., 2020c] is presented BEE, an approach to identify biomedical events based on a QA system. They described the task as a set of questions that are iteratively answered using SciBERT. These questions are represented by a template that defines different types of questions and their sequence, following a recursive procedure. The two main question types have as purpose of detecting event triggers and arguments. They compared their proposal with TEES SVM [Björne and Salakoski, 2013] and TEES CNN [Björne and Salakoski, 2018], surpassing both systems and obtaining an F1-score of 0.62 using the GENIA 2011 [Kim et al., 2011] corpus and 0.59 using the PC [Nédellec et al., 2013] corpus. One of the state-of-the-art systems for biomedical event extraction is DeepEventMine [Trieu et al., 2020], an end-to-end method that consists of four main modules; BERT model, trigger and entity detection and classification, relation extraction, and event identification. For each of the modules, BERT is used as a base model, and a linear layer is added. One of the main objectives of this system is improving the extraction of nested events, where it has achieved the new SOTA performance on seven biomedical nested event extraction tasks, including CG [Nédellec et al., 2013] with an F1-score of 61.74, GE13 [Kim et al., 2013] with 56.72, PC [Nédellec et al., 2013] with 57.72 and EPI [Ohta et al., 2011] with 65.57. Proposing an alternative approach, Ramponi et al. [Ramponi et al., 2020] developed BEESL, a neural network model based on a sequence labeling system for the extraction of events. The system converts the event structures into a format of sequence labeling and uses BERT as language model. On the other hand, Chen [Chen, 2021] proposed the Multi-Source Transfer Learning-based Trigger Recognizer system, which is an extension of transfer learning using multiple source domains. All the datasets from the different domains are used to jointly train the neural network, achieving a higher recognition performance in the biomedical domain, and having a wide coverage of events.

To address the recurrent issue of low recall in biomedical event extraction systems due to unrecognized events, Su et al. [Su et al., 2022] proposed a multi-task approach using transformer models. Here, the lexical and syntactic information of the data encoded is shared by multiple subtasks, including trigger detection and argument identification. In addition, they added an auxiliary subtask to identify proteins that participate in the events, helping to decrease the errors related to these specific events. The third subtask corresponds to event construction, where they especially focused on complex nested events. This end-to-end method allowed to balance the precision and recall of biomedical event extraction and outperformed the recall of DeepEventMine [Trieu et al., 2020] on two biomedical datasets by 4.65 % and 5.0 %. MTTLADE [El-allaly et al., 2021] is a multi-task learning method developed for the specific case of extracting events of Adverse Drug Events (ADEs). This system converted the ADEs extraction task into

a dual-task sequence labeling that consists of extracting the ADEs source mention and the ADEs attribute-relation. Both tasks are processed simultaneously using a multi-task transfer learning approach based on the representations obtained from a pre-trained transformer model and then fine-tuned for the specific task. For this, they evaluated five pre-trained transformer models for comparison. In the last step, MTTLADE produces sequences for each task to extract ADEs mentions and relations. They finally concluded that combining transfer and multi-task learning is an effective method to extract intricate ADEs. Later, it was proposed BEEDS [Wang et al., 2022], a new approach to extract events based on their last system, BEE. This method consists of a three-step pipeline following a document retriever, a document reader, and an entity normalizer using transformer models. The events are extracted as triples that involve protein or gene entities and stored in a knowledge base. To mitigate the problem of having reduced data, they applied data augmentation using a distant supervision approach. Levitated Context Markers (LCMs) [Vasilakes et al., 2023] is an adaptation of levitated markers, originally used for RE tasks. LCMs focus on word sub-spans that are related to events using an attention mechanism and also use the global input representations from transformer models simultaneously. The levitated markers are related to a single disposition event to force the model to be focused on potentially useful sub-spans. In addition, they showed that sparse attention allows to provide interpretable predictions through the detection of relevant context cues.

Selecting a transformer model from the models available in the literature can be challenging, as it is often unclear which one provides the best performance and the underlying rationale for this choice. To tackle this, Portelli et al. [Portelli et al., 2021] compared BERT and five of its variants for the identification of ADEs. They showed that span-based pre-training, from spanBERT, provides an improvement in the recognition of ADEs, and that the pretraining of the models in the specific domain is particularly useful in comparison to train the models from scratch. Following this approach, Scaboro et al. [Scaboro et al., 2023] performed an evaluation of 19 transformer models for ADEs extraction on informal text based on posts and tweets. They compared the performance of the models adding a CRF and a LSTM layers, and applied a feature importance technique to correlate the performance of the models with the features that describe them. In their analysis, they considered the model category, pre-training domain, training from scratch, and model size. From their results, they conclude that Auto-Encoding models are the best technique for ADEs extraction, and text-generation models do not perform well on long texts or texts with multiple ADEs. Besides, when all the features in the models are the same, larger models present the highest performance. For the pre-training, if the models use social media text, the performance consistently increases, while using medical text is only effective on more formal language.

As described in the works above, transformer models have achieved significant advancement in biomedical event extraction. Different transformer-based architectures, such as BERT, BioBERT,

PubMedBERT, and T5, have demonstrated the ability to tackle the complexities of event extraction, showing remarkable capabilities in understanding and capturing intricate relationships between biomedical entities, events, and their contextual information. BERT-based models have demonstrated their ability to pre-train in large corpora of biomedical text, enabling them to effectively handle small and unbalanced datasets during fine-tuning. In contrast to BERT-based models, the T5-based generative models have shown lower performance in event extraction. However, these models have shown significant performance in other complex tasks requiring generation capabilities. Taking advantage of the potential of transformer models, it is possible to seek improvement in biomedical event extraction models. This can lead to increased medical knowledge discovery, drug development, and patient care.

2.2.2 Challenges of transformer models in the biomedical domain

As mentioned before, the two primary approaches for developing pre-trained language models based on transformers are mixed-domain and domain-specific pre-training. These approaches involve extensive pre-training on large volumes of in-domain text using high-performance GPUs or TPUs over an extended period. While these approaches have shown great success in developing pre-trained language models, they come with high computational costs and long pre-training duration [Poerner et al., 2020]. For instance, adapting general BERT to the biomedical domain using eight GPUs in BioBERT took approximately ten days. Additionally, domain-specific pre-training is more resource-intensive as it requires learning model weights from scratch. Therefore, there is a need for cost-effective domain adaptation methods to tailor general BERT models to the biomedical domain. Two such low-cost domain adaptation methods are task adaptive pre-training and extending the embedding layer. Task adaptive pre-training involves further pre-training on task-related unlabeled instances. This method allows models to acquire both domain and task-specific knowledge by pre-training on a relatively small task-related unlabeled corpus. While extending the embedding layer, the general transformer models can be adapted to the biomedical domain by refining the embedding layer through the addition of new in-domain vocabulary. The new in-domain vocabulary can be generated, for example, either by using Word2Vec over in-domain text and aligning it with the general word-piece vocabulary, or by generating it using word-piece over in-domain text [Kalyan et al., 2022].

Further, the size of the data that is used for pre-training affects how well the model learns to represent the language. It is generally not possible to have biomedical databases large enough for models to generalize and be useful across different tasks. A possible solution to these scenarios is to apply simultaneous pre-training, training the models with combined in-domain and general data [Wada et al., 2020]. Also, having access to large volumes of data for fine-tuning the models allows to learn enough knowledge to perform well in specific NLP tasks. However, in the biomedical

domain, there are few and small manually annotated databases that can be used as target data. Some of the proposed solutions to deal with the small size of the target data are intermediate fine-tuning or multi-task fine-tuning, being the second approach more widely used in the biomedical domain. There is also the data augmentation option, which allows to create new samples from existing ones. The new instances created are similar to the original data, being especially useful when these are few. Two approaches that have been used in the biomedical domain for data augmentation are back translation and the use of in-domain ontologies. Semi-supervised learning also allows data augmentation from pseudo-labeled instances, since the model that is fine-tuned with the original data is used to annotate the in-domain unlabeled data. This process can be repeated multiple times until the model converges [Pruksachatkun et al., 2020].

Another challenge that is observed in pre-trained models based on transformers is that their performance is limited when noisy data is used for testing. This occurs because models are generally trained on data with little or no noise, so they are not trained to deal with these cases. One solution to this problem is adversarial training, where the model is exposed to noisy data or "adversarial samples" that the model will learn to detect and thus improve its performance when tested on noisy data [Araujo et al., 2020].

The challenges mentioned above represent a problem that limits the performance of pre-trained transformer models, in addition to involving large resources that cause other external consequences. Research in this domain continues to develop, making significant progress in creating more reliable and versatile language models that can effectively address real-world scenarios while promoting fairness, robustness, and inclusiveness in NLP tasks.

Chapter 3

Knowledge graphs and biomedical information extraction

Contents

Overview

This chapter details the fundamental principles of KGs and their applications in the biomedical domain. We explain how KGs have emerged as a practical knowledge representation framework that leverages the interconnections in data to discover information. KGs represent complex information in a structured and semantically rich manner by capturing relationships between entities, concepts, and edges. We also describe the implications of using KGs for biomedical knowledge discovery. By synthesizing information from multiple sources, these knowledge representations facilitate hypothesis generation, drug discovery, disease diagnosis, and biomolecular interactions.

3.1 KG definition

KGs are semantic networks that contain the different correlations between different types of entities. They enable reasoning and explanation of data. A KG is a knowledge base presented in a graphical structure format, composed of multiple types of relations between entities.

Formally, a *knowledge graph* G is defined as a couple $G = (\mathcal{E}, \mathcal{R})$, where $\mathcal{E} = \{e_1, e_2, ..., e_n\}$ (with $n = |\mathcal{E}|$) is a set of *nodes* (entities) and where $\mathcal{R}$ is a set of binary relation over $\mathcal{E}$ (ie. $\forall r \in \mathcal{R}, r \subseteq \mathcal{E} \times \mathcal{E}$), said to be the *relation set* of G. The *adjacency matrix* $A \in \mathbb{R}^{n \times n}$ of G is the $n \times n$ matrix such that for all $(i, j) \in \{1, \ldots, n\}^2$, $A[i, j]$ is the number of relations of $\mathcal{R}$ that contain (e_i, e_j) (ie. $A[i, j] = |\{r \in \mathcal{R} \mid (e_i, e_j) \in r\}|$). For all $i \in \{1, \ldots, n\}$, the *degree* of an entity $e_i \in \mathcal{E}$ is defined as $\sum_{j=1}^{n} A[i, j]$, and the *degree matrix* $D \in \mathbb{R}^{n \times n}$ of G is the diagonal matrix with for all $i \in \{1, \ldots, n\}$, $D[i, i] = \sum_{j=1}^{n} A[i, j]$ is the degree of e_i [Wang et al., 2021]. The set of *edges* or *facts* of G is defined as the set $\mathcal{F}$ of triples $(h, r, t) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ such that $(h, t) \in r$. Given an edge (h, r, t) of G, h is said to be the *head entity* of the edge (h, r, t), and t is said to be the *tail entity* of (h, r, t).

KGs are composed of edges (also called facts) defined as triples (h, r, t), where h represents the head entity, t represents the tail entity, and r represents the relation between them. To use the knowledge integrated into the KG, these edges are modeled using KG embeddings, mapping the content of entities and relations to low-dimensional vectors. Then, the vectors can be used for different tasks, including KG completion, link prediction, and entity classification.

KG models usually consider entities h and t as their embeddings vectors $\mathbf{h}$ and $\mathbf{t}$, whereas the relations r will often be seen as the affine operator $x \mapsto x + \mathbf{r}$, which are used to calculate the relation between the nodes. For instance, it is common to expect from a relevant embedding of G that for all $(h, r, t) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, $\mathbf{h} + \mathbf{r}$ is "close" to $\mathbf{t}$ regarding a norm in the vector space F (which we will denote by $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$) if and only if (h, r, t) is an edge in G.

Therefore, to calculate the relation between the nodes, the KG model first replaces each triple with all the possible relations in the KG to obtain negative samples and embeds the knowledge of the triples. Most of the current techniques calculate the embeddings based only on the observed edges. Then, a scoring function is used to measure the reliability of the triples based on the embeddings, where a higher score means that the triple is more likely to be true. A loss function is used to optimize the reliability of all triples in the KG.

The evaluation of the model is usually done by recording the ranks of the correct predictions in an ordered list to see whether the correct predictions are ranked before the incorrect ones [Wang et al., 2021].

3.2 KG embeddings

KG embeddings allow mapping the content of entities and their relations contained in a KG to a low-dimensional vector space. Most techniques calculate KG embeddings using the KG edges to obtain the vectors, enforcing embeddings to be compatible with these edges. Figure 3.1 presents the process followed to map a KG into KG embeddings.

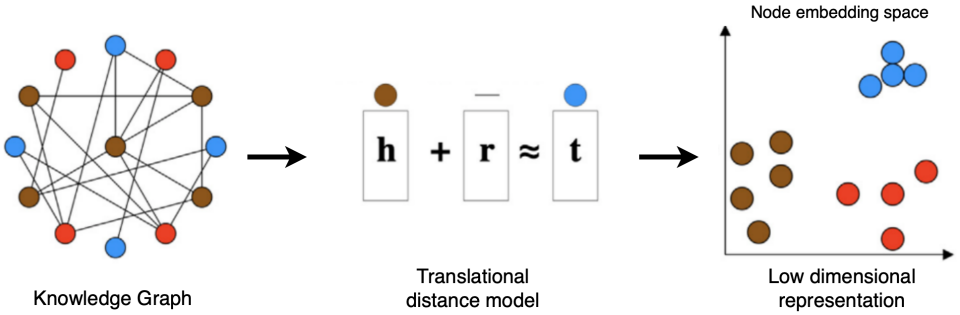


Figure 3.1: General pipeline to represent the calculation of KG embeddings (Figure taken from [Nicholson and Greene, 2020]).

A general KG embedding technique consists of three main steps: (1) representing entities and relations, (2) defining a scoring function, and (3) learning nodes and relations representations. The first step defines the continuous vector space. Entities are usually represented as deterministic points in the vector space that can further consider other information, such as uncertainties modeled using multivariate Gaussian distributions [Wang et al., 2017b]. Relations are generally considered operations in the vector space that can be represented as vectors, matrices, tensors, and multivariate Gaussian distributions. For the second step, a scoring function $f_r(\mathbf{h}, \mathbf{t})$ is defined on the different triples (h, r, t) to measure their plausibility. The edges observed in the KG will have a higher score than those not observed. Then, the third step consists of learning these entity and relation representations, i.e., embeddings, by solving an optimization problem to maximize the plausibility of the observed edges.

The techniques to calculate the KG embeddings are commonly categorized into three main groups, translational distance models (Section 3.2.1), semantic matching models (Section 3.2.2), and neural network models (Section 3.2.3).

3.2.1 Translational distance models

Translational distance models use scoring functions based on distance. Here, the plausibility of an edge refers to the distance between two entities measured through the translation of the relation. TransE [Bordes et al., 2013] is the most representative translational distance model. It represents the entities and relations as vectors in the same space. Given a triple $(h, r, t) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, the

relation refers to the translation vector $\mathbf{r}$ that connects $\mathbf{h}$ and $\mathbf{t}$, i.e., $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$, if and only if (h, r, t) is a fact.

Figure 3.2 (a) shows a simple illustration of TransE. The intuition behind this model is to learn distributed word representations to capture linguistic regularities such as *virus* - *antiviral* ≈ *bacteria* - *antibiotic*. Multi-relational data holds this type of analogy through a certain relation, e.g., treatment. Therefore, it is possible to get *virus* + *treatment* ≈ *antiviral* and *bacteria* + *treatment* ≈ *antibiotic*. The scoring function of TransE is defined in Equation (3.1), which calculates the opposite of the distance between $\mathbf{h} + \mathbf{r}$ and $\mathbf{t}$. A high score will be obtained if (h, r, t) is an edge.

$$f_r(\mathbf{h}, \mathbf{t}) = -\|\mathbf{h} + \mathbf{r} - \mathbf{t}\| \tag{3.1}$$

Here, $\|\cdot\|$ can refer either to the L_1 norm $\|\cdot\|_1$ defined for all $\mathbf{x} \in \mathbb{R}^d$ by $\|\mathbf{x}\|_1 = \sum_{i=1}^{d} |[\mathbf{x}]_i|$, or to the L_2 norm $\|\cdot\|_2$ defined for all $\mathbf{x} \in \mathbb{R}^d$ by $\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{d} |[\mathbf{x}]_i|^2}$.

However, even if TransE is efficient and straightforward (in terms of the number of parameters), it can only handle one-to-one relations. For example, in the case of one-to-n-ary relations, TransE enforces $\mathbf{h} + \mathbf{r} \approx \mathbf{t}_i$ for all $i \in \{1, \ldots, p\}$, and then $\mathbf{t}_1 \approx \cdots \approx \mathbf{t}_p$. Therefore, the model might learn very similar vector representations for different nodes, even if they are semantically different from each other. One strategy to overcome this problem is to calculate a different representation of the same entity according to the relation to which it is involved.

TransH [Wang et al., 2014] follows the principle of incorporating relation-specific hyperplanes to generate entity embeddings. This model calculates entities as vectors and each relation as a vector $\mathbf{r}$ on a hyperplane with a normal vector $\mathbf{w}_r$. Figure 3.2 (b) illustrates these vectors. Therefore, given a triple (h, r, t), the representations of h and t are projected in a hyperplane, resulting in Equation (3.2).

$$\mathbf{h}_\perp = \mathbf{h} - (\mathbf{w}_r^\top \mathbf{h}) \cdot \mathbf{w}_r, \qquad \mathbf{t}_\perp = \mathbf{t} - (\mathbf{w}_r^\top \mathbf{h}) \cdot \mathbf{w}_r \tag{3.2}$$

These projections are then connected by $\mathbf{r}$ on the hyperplane if (h, r, t) is an edge, i.e., $\mathbf{h}_\perp + \mathbf{r} \approx \mathbf{t}_\perp$. The scoring function of TransH is defined in Equation (3.3).

$$f_r(\mathbf{h}, \mathbf{t}) = -\|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2 \tag{3.3}$$

The TransR [Lin et al., 2015] model follows a similar principle to TransH but uses relation-specific spaces instead of hyperplanes. This model represents entities as vectors in an entity space, $\mathbb{R}^d$, while each relation is associated with a specific relation space, $\mathbb{R}^k$, and modeled as a translational vector. Therefore, given a triple (h, r, t), TransR projects the entity representations $\mathbf{h}$ and $\mathbf{t}$ into the space of the specific relation, as shown in Equation (3.4) and illustrated in

Figure 3.2 (c). Here, $\mathbf{M}_r \in \mathbb{R}^{k \times d}$ represents the projection matrix from the entity space to the relation space of r. The score function is calculated with Equation (3.5).

$$\mathbf{h}_\perp = \mathbf{M}_r \mathbf{h}, \qquad \mathbf{t}_\perp = \mathbf{M}_r \mathbf{t} \tag{3.4}$$

$$f_r(\mathbf{h}, \mathbf{t}) = -\|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2 \tag{3.5}$$

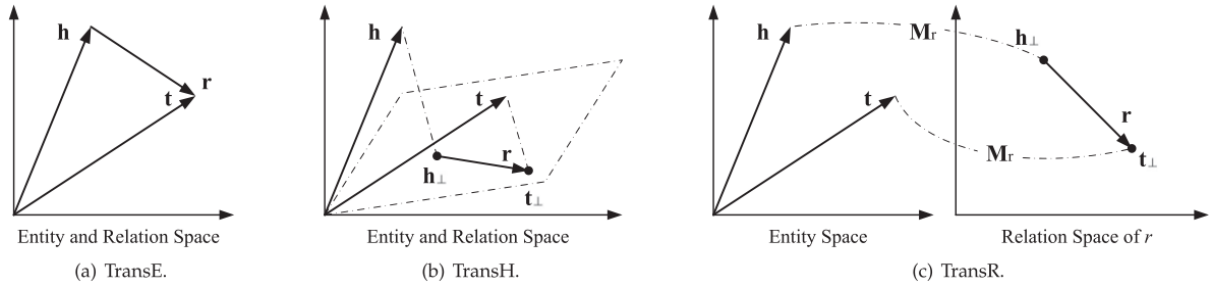


Figure 3.2: Representations of TransE, TransH, and TransR (Figure taken from [Wang et al., 2017b]).

Since TransR introduces a projection matrix for each relation, its simplicity and efficiency are reduced. To tackle this, TransD [Ji et al., 2015] simplifies TransR by decomposing the projection matrix into a product of vectors. This model introduces additional mapping vectors, $(\mathbf{w}_h, \mathbf{w}_t) \in (\mathbb{R}^d)^2$ and $\mathbf{w}_r \in \mathbb{R}^k$ for each triple (h, r, t), together with entity-relation representations $(\mathbf{h}, \mathbf{t}) \in (\mathbb{R}^d)^2$ and $\mathbf{r} \in \mathbb{R}^k$. The projection matrices $\mathbf{M}_r^1$ and $\mathbf{M}_r^2$ are calculated with Equation (3.6).

$$\mathbf{M}_r^1 = \mathbf{w}_r \mathbf{w}_h^\top + I, \qquad \mathbf{M}_r^2 = \mathbf{w}_r \mathbf{w}_t^\top + I \tag{3.6}$$

Then, the projections of these projection matrices are calculated with Equation (3.7), and the score function is calculated in the same way as in TransR.

$$\mathbf{h}_\perp = \mathbf{M}_r^1 \mathbf{h}, \qquad \mathbf{t}_\perp = \mathbf{M}_r^2 \mathbf{t} \tag{3.7}$$

TransM [Fan et al., 2014] is another translational model associating a weight θ_r specific to r to each triple (h, r, t). The score function is calculated with Equation (3.8). Therefore, by assigning lower weights to one-to-n-ary, n-ary-to-one, and n-ary-to-n-ary relations, the model allows t to be further away from $h + r$.

$$f_r(\mathbf{h}, \mathbf{t}) = -\theta_r \|\mathbf{h} + \mathbf{r} - \mathbf{t}\| \tag{3.8}$$

Again, $\|\cdot\|$ can either denote the L_1 norm $\|\cdot\|_1$ or the L_2 norm $\|\cdot\|_2$.

TransF [Feng et al., 2016] follows the same principle of TransM but only requires $\mathbf{t}$ to lie in the same direction as $\mathbf{h}+\mathbf{r}$ and $\mathbf{h}$ in the same direction as $\mathbf{t}-\mathbf{r}$. Therefore, $\mathbf{t}$ is matched with $\mathbf{h}+\mathbf{r}$ and $\mathbf{h}$ with $\mathbf{t}-\mathbf{r}$ in the scoring function, as shown in Equation (3.9).

$$f_r(\mathbf{h},\mathbf{t}) = (\mathbf{h}+\mathbf{r})^\top \mathbf{t} + (\mathbf{t}-\mathbf{r})^\top \mathbf{h} \tag{3.9}$$

TransA [Xiao et al., 2015] introduces the concept of relaxed translation. This approach relaxes the overly rigid constraint of $\mathbf{h}+\mathbf{r} \approx \mathbf{t}$. These models offer a straightforward and intuitive approach to representing relations in KGs. They provide geometric interpretations of these relations, facilitating the understanding of the underlying data. Moreover, these models demonstrate efficiency in training, requiring fewer computational resources than more complex models. Their scalability is a notable advantage, enabling them to handle large knowledge graphs effectively. However, they are limited in representing complex or non-linear relations and may present problems with n-ary-to-n-ary relations. Additionally, they may not capture contextual or semantic information as effectively as more advanced models, limiting their applicability in tasks requiring a deeper understanding of the data.

3.2.2 Semantic matching models

Models based on semantic information can outperform translational distance models by capturing semantics. This type of model usually uses similarity-based functions for the calculation of the scoring functions. There are two main types: traditional semantic matching models and models that introduce additional information for more knowledge. Traditional semantic matching models, such as DistMult [Yang et al., 2014], match the latent semantics of the entities and the relations to measure the plausibility of a triple. They only focus on the information of the triple itself and do not consider any additional information. Models that introduce additional information, such as SimplE [Sorber et al., 2013], mine deeper information in graphs. The additional information may include path information, order information, and, entity types.

RESCAL [Nickel et al., 2011] is a semantic matching model that associates entities with vectors to capture latent semantics. Relations are represented as a matrix that models pairwise interactions between the latent factors. The scoring function is defined by the bilinear function shown in Equation (3.10), where $\mathbf{h}$,$\mathbf{t} \in \mathbb{R}^d$ refers to the vector representations of entities and $\mathbf{M}_r \in \mathbb{R}^{d\times d}$ refers to the matrix associated with the relation. In this way, the score captures the pairwise interactions between all the components $\mathbf{h}$ and $\mathbf{t}$, as shown in Figure 3.3 (a).

$$f_r(\mathbf{h},\mathbf{t}) = \mathbf{h}^\top \mathbf{M}_r \mathbf{t} = \sum_{i=0}^{d-1}\sum_{j=0}^{d-1} [\mathbf{M}_r]_{i,j} \cdot [\mathbf{h}]_i \cdot [\mathbf{t}]_j \tag{3.10}$$

DistMult [Yang et al., 2014] follows the same principle of RESCAL but simplifies the scoring

function by restricting $\mathbf{M}_r$ to diagonal matrices. Each relation is represented by a vector embedding $\mathbf{r} \in \mathbb{R}^d$ and $\mathbf{M}_r = diag(\mathbf{r})$. The scoring function, defined with Equation (3.11), captures pairwise interactions between the elements of $\mathbf{h}$ and $\mathbf{t}$ in the same dimension, as shown in Figure 3.3 (b), reducing the number of parameters per relation. However, since $\mathbf{h}^\top diag(\mathbf{r})\mathbf{t} = \mathbf{t}^\top diag(\mathbf{r})\mathbf{h}$ for any entities h and t, this model can only deal with symmetric relations, which is a limitation for general KGs since most of them are not symmetric.

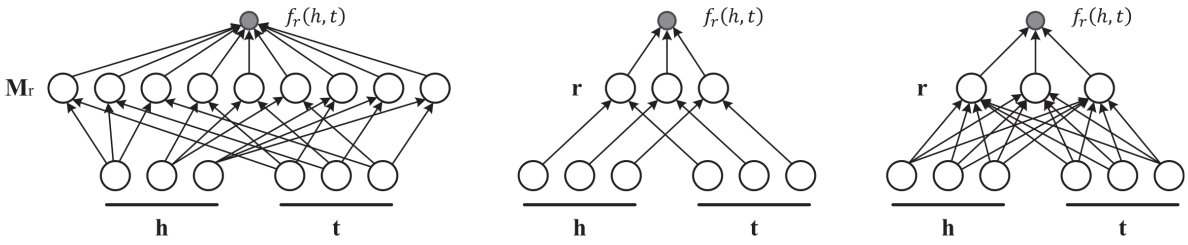
$$f_r(\mathbf{h}, \mathbf{t}) = \mathbf{h}^\top diag(\mathbf{r})\mathbf{t} = \sum_{i=0}^{d-1} [\mathbf{r}]_i \cdot [\mathbf{h}]_i \cdot [\mathbf{t}]_i \tag{3.11}$$

Alternatively, HolE [Nickel et al., 2016], which stands for Holographic Embeddings, represents a combination of the representational capacity of RESCAL and the efficiency and simplicity of DistMult, representing entities and relations as vectors in $\mathbb{R}^d$. Given a triple (h, r, t), the representations of entities are calculated as $\mathbf{h} \star \mathbf{t} \in \mathbb{R}^d$, using the circular correlation operation, $\star$, of Equation (3.12).

$$[\mathbf{h} \star \mathbf{t}]_i = \sum_{k=0}^{d-1} [\mathbf{h}]_k \cdot [\mathbf{t}]_{(k+i) \bmod d} \tag{3.12}$$

Then, the scoring function is calculated using the compositional vector matched with the relation representation, as shown in Equation (3.13). Circulation correlation represents a compression of pairwise interactions, as observed in Figure 3.3 (c), which makes HolE require fewer parameters than RESCAL but be more efficient. In addition, since circular correlation is not commutative, i.e., $\mathbf{h} \star \mathbf{t} \neq \mathbf{t} \star \mathbf{h}$, this model can deal with asymmetric relations.

$$f_r(\mathbf{h}, \mathbf{t}) = \mathbf{r}^\top (\mathbf{h} \star \mathbf{t}) = \sum_{i=0}^{d-1} [\mathbf{r}]_i \sum_{k=0}^{d-1} [\mathbf{h}]_k \cdot [\mathbf{t}]_{(k+i) \bmod d} \tag{3.13}$$



(a) RESCAL. (b) DistMult. (c) HolE.

Figure 3.3: Representations of RESCAL, DistMult, and HolE (Figure taken from [Wang et al., 2017b]).

ComplEx [Trouillon et al., 2016], which stands for Complex Embeddings, is an extension of DistMult that introduces embeddings with complex values to improve the performance of asymmetric relations. Here, entity and relation embeddings lie in a complex space $\mathbb{C}^d$. The scoring function of a triple (h, r, t) is defined with Equation (3.14), where $\overline{\mathbf{t}}$ is the conjugate of $\mathbf{t}$ and $\mathrm{Re}(\cdot)$ refers the real part of the complex value. Since the scoring function is not symmetric, edges of asymmetric relations may receive different scores according to the order of the entities involved.

$$f_r(\mathbf{h}, \mathbf{t}) = \mathrm{Re}(\mathbf{h}^\top \mathrm{diag}(\mathbf{r})\overline{\mathbf{t}}) = \mathrm{Re}\left(\sum_{i=0}^{d-1} [\mathbf{r}]_i \cdot [\mathbf{h}]_i \cdot [\ \overline{\mathbf{t}}\ ]_i\right) \tag{3.14}$$

ANALOGY [Liu et al., 2017] extends RESCAL to model analogical properties of entities and relations, e.g., *virus* is to *antiviral* as *bacteria* is to *antobiotic*. Following the RESCAL principle, its bilinear scoring function is calculated with Equation (3.15), where $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$ represents a linear map associated with the relation.

$$f_r(\mathbf{h}, \mathbf{t}) = \mathbf{h}^\top \mathbf{M}_r \mathbf{t} \tag{3.15}$$

The analogical structures are modeled based on two requirements, the relations of the linear maps need to be normal and mutually commutative, as shown below.

$$\text{normality} : \mathbf{M}_r \mathbf{M}_r^\top = \mathbf{M}_r^\top \mathbf{M}_r, \text{ for all relation } r$$
$$\text{commutativity} : \mathbf{M}_r \mathbf{M}_{r'} = \mathbf{M}_{r'} \mathbf{M}_r, \text{ for all relations } r \text{ and } r'$$

SimplE [Sorber et al., 2013] is a model based on canonical polyadic decomposition, a tensor factorization approach. It learns independent embedding vectors for the entities h and t, even if they are tied. SimplE encodes the vector embeddings of both entities by parameter sharing, allowing them to integrate their dependencies into a relation vector $\mathbf{v}_r$ and $\mathbf{v}_{r-1}$ for the inverse relation. These embeddings are optimized by satisfying the scoring function of Equation (3.16), where for all $(\mathbf{x}, \mathbf{y}, \mathbf{z}) \in (\mathbb{R}^d)^3$, $\langle \mathbf{x}, \mathbf{y}, \mathbf{z} \rangle = \sum_{i=0}^{d-1} [\mathbf{x}]_i \cdot [\mathbf{y}]_i \cdot [\mathbf{z}]_i$.

$$f_r(\mathbf{h}, \mathbf{t}) = \frac{\langle \mathbf{h}, \mathbf{v}_r, \mathbf{t} \rangle + \langle \mathbf{t}, \mathbf{v}_{r-1}, \mathbf{h} \rangle}{2} \tag{3.16}$$

In summary, compared to translational distance models, semantic matching models have generally been shown to mine deeper semantic information, e.g., patch information, contextual information, and entity types, allowing to obtain more accurate KG representations.

3.2.3 Neural network models

Neural networks-based KG models proposed an approach with the abilities of self-study, high-speed optimization, and associative storage. These models make non-linear transformations of

high dimensional data by commonly using structures based on Word2vec [Mikolov et al., 2013b]. These models aim to project words into a low-dimensional space that input downstream models, such as FFNNs or RNNs. One of the main characteristics of these models is the propagation of neighborhood information, obtaining better and more efficient entity and relation embeddings [Wang et al., 2021]. However, their performance largely depends on the structure of nodes and edges in the KG. Furthermore, for this approach, only the nodes are represented by vectors, which means that KG embeddings no longer meet the definition of a KG [Nicholson and Greene, 2020].

SME [Bordes et al., 2014], which stands for Semantic Matching Energy, is a model that performs semantic matching using neural networks. This model first projects entities and relations to their vector embeddings in the input layer. Then, the embedding $\mathbf{h}$ of the head h and the embedding $\mathbf{r}$ of the relation r of the triple (h, r, t) are combined to get $g_u(\mathbf{h}, \mathbf{r})$, as well as the embedding $\mathbf{t}$ of the tail t, and $\mathbf{r}$ to get $g_v(\mathbf{t}, \mathbf{r})$ in the hidden layer. The scoring function is calculated by the dot product of matching g_u and g_v, as shown in Equation (3.17).

$$f_r(\mathbf{h}, \mathbf{t}) = g_u(\mathbf{h}, \mathbf{r})^\top \cdot g_v(\mathbf{t}, \mathbf{r}) \tag{3.17}$$

SME has two versions: linear and bilinear. The linear version is defined by Equation (3.18),

$$\begin{aligned} g_u(\mathbf{h}, \mathbf{r}) &= \mathbf{M}_u^1 \mathbf{h} + \mathbf{M}_u^2 \mathbf{r} + \mathbf{b}_u, \\ g_v(\mathbf{t}, \mathbf{r}) &= \mathbf{M}_v^1 \mathbf{t} + \mathbf{M}_v^2 \mathbf{r} + \mathbf{b}_v \end{aligned} \tag{3.18}$$

while the bilinear version is defined by Equation (3.19). Here, $\circ$ designates the element-wise product, $\mathbf{M}_u^1$, $\mathbf{M}_u^2$, $\mathbf{M}_v^1$ and $\mathbf{M}_v^2 \in \mathbb{R}^{d \times d}$ represent weight matrices and $\mathbf{b}_u$, $\mathbf{b}_v \in \mathbb{R}^d$ represent bias vectors for the different relations. An illustration of SME is shown in Figure 3.4 (a).

$$\begin{aligned} g_u(\mathbf{h}, \mathbf{r}) &= (\mathbf{M}_u^1 \mathbf{h}) \circ (\mathbf{M}_u^2 \mathbf{r}) + \mathbf{b}_u, \\ g_v(\mathbf{t}, \mathbf{r}) &= (\mathbf{M}_v^1 \mathbf{t}) \circ (\mathbf{M}_v^2 \mathbf{r}) + \mathbf{b}_v \end{aligned} \tag{3.19}$$

The model NTN [Socher et al., 2013], which stands for Neural Tensor Network, first projects the entities to the vector embeddings in the input layer. The two embeddings $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$ of the entities h and t are then combined by a tensor $\underline{\mathbf{M}}_r \in \mathbb{R}^{d \times k \times d}$ specific to the relation and mapped to a non-linear hidden layer. Then, the output layer gives the score through a linear calculation specific to the relation, as shown in Equation (3.20). Here, $\mathbf{M}_r^1, \mathbf{M}_r^2 \in \mathbb{R}^{k \times d}$ and $\mathbf{b}_r \in \mathbb{R}^k$ refer to weight matrices and bias vectors specific to the relation, while the bilinear tensor product $\mathbf{h}^\top \underline{\mathbf{M}}_r \mathbf{t}$ produces a vector for each entry. A simple illustration of NTN is shown in Figure 3.4 (b).

$$f_r(\mathbf{h}, \mathbf{t}) = \mathbf{r}^\top \tanh(\mathbf{h}^\top \underline{\mathbf{M}}_r \mathbf{t} + \mathbf{M}_r^1 \mathbf{h} + \mathbf{M}_r^2 \mathbf{t} + \mathbf{b}_r) \tag{3.20}$$

Recall that the vector $\mathbf{h}^\top \underline{\mathbf{M}}_r \mathbf{t}$ belongs to $\mathbb{R}^k$ and is defined by Equation (3.21).

$$\forall l \in \{0, \ldots, k-1\}, [\mathbf{h}^\top \underline{\mathbf{M}}_r \mathbf{t}]_l = \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} [\mathbf{h}]_i [\underline{\mathbf{M}}_r]_{i,l,j} [\mathbf{t}]_j \tag{3.21}$$

In the case of MLP [Dong et al., 2014], which stands for Multi-Layer Perceptron, is a model that associates each entity and relation to a single vector. Given a triple (h, r, t), the vector embeddings $\mathbf{h}, \mathbf{r}$ and $\mathbf{t}$ are concatenated in the input layer and mapped to a non-linear layer, as shown in Figure 3.4 (c). The score is calculated in the output layer by the linear function of Equation (3.22), where $\mathbf{M}^1, \mathbf{M}^2, \mathbf{M}^3 \in \mathbb{R}^{d \times d}$ represent the first layer weights and $\mathbf{w} \in \mathbb{R}^d$ the second layer weights.

$$f_r(\mathbf{h}, \mathbf{t}) = \mathbf{w}^\top \tanh(\mathbf{M}^1 \mathbf{h} + \mathbf{M}^2 \mathbf{r} + \mathbf{M}^3 \mathbf{t}) \tag{3.22}$$

In Equation (3.22), the vector $\tanh(\mathbf{M}^1 \mathbf{h} + \mathbf{M}^2 \mathbf{r} + \mathbf{M}^3 \mathbf{t})$ is obtained by applying the function $\tanh : x \mapsto \frac{e^x - e^{-x}}{e^x + e^{-x}}$ (hyperbolic tangent) to every coordinate of the vector $\mathbf{M}^1 \mathbf{h} + \mathbf{M}^2 \mathbf{r} + \mathbf{M}^3 \mathbf{t}$.

NAM [Liu et al., 2016], which stands for Neural Association Model, follows a semantic matching approach with a deep architecture. NAM first concatenates the vector embeddings of the head and the relation in the input layer, $\mathbf{z}^{(0)} = [\mathbf{h}; \mathbf{r}] \in \mathbb{R}^{2d}$. Then, $\mathbf{z}^{(0)}$ is fed into a deep neural network that consists of L rectified linear hidden layers, as shown in Equation (3.23), where $\mathbf{M}^{(\ell)}$ and $\mathbf{b}^{(\ell)}$ represent the weight matrix and bias of the ℓ^{th} layer.

$$\begin{aligned} \mathbf{a}^{(\ell)} &= \mathbf{M}^{(\ell)} \mathbf{z}^{(\ell-1)} + \mathbf{b}^{(\ell)}, \qquad \ell = 1, ..., L, \\ \mathbf{z}^{(\ell)} &= \mathrm{ReLU}(\mathbf{a}^{(\ell)}), \qquad \ell = 1, ..., L \end{aligned} \tag{3.23}$$

In Equation (3.23), ReLU designates the rectified linear unit: $\mathrm{ReLU}: x \mapsto \frac{x + |x|}{2}$. Also, the vector $\mathrm{ReLU}(\mathbf{a}^{(\ell)})$ is obtained by applying the function ReLU to every coordinate of $\mathbf{a}^{(\ell)}$.

The score is calculated after the feed-forward process with Equation (3.24), by matching the output of the last hidden layer and the embedding of the tail entity. Figure 3.4 (d) illustrates this process.

$$f_r(\mathbf{h}, \mathbf{t}) = \mathbf{t}^\top \mathbf{z}^{(L)} \tag{3.24}$$

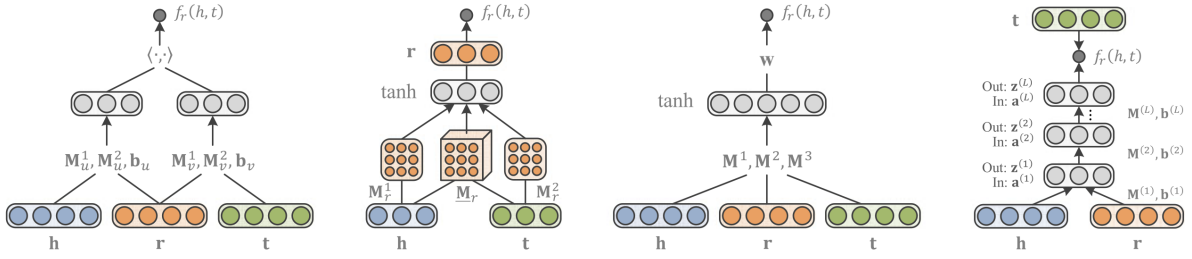


Figure 3.4: Representations of SME, NTN, MLP, and NAM (Figure taken from [Wang et al., 2017b]).

Other neural networks-based models are ConvE [Dettmers et al., 2018], which was the first to use CNNs for KG completion, and ConvKB [Nguyen et al., 2017] which combines CNNs and the translation principle of TransE. In summary, KG models based on neural networks allow to learn complex relationships and patterns within KGs. They capture intricate semantic information and can adapt to various data types. However, they can be computationally intensive and require substantial data for efficient training. Additionally, these models generally present low interpretability, making it challenging to understand their decision-making process compared to semantic matching models.

3.3 Model training and evaluation

The training procedure of KG models can be performed under two assumptions, closed world and open world. In closed-world assumptions, all the edges that are not contained in the KG are assumed to be false. In open-world assumption, the edges contained in the KG are true, and the edges not contained in the KG can be either false or missing.

In the case of open-world assumption, $\mathbb{D}^+$ stores the positive examples, while the negative examples are later generated using heuristics. Therefore, given the positive set $\mathbb{D}^+$ and negative set $\mathbb{D}^-$, the model learns an embedding Θ by minimizing a loss function, e.g, logistic function, calculated with Equation (3.25). Here, $\tau = (h, r, t)$ refers to a training example in $\mathbb{D}^+ \cup \mathbb{D}^-$, and $y_{hrt} = \pm 1$ refers to the label, positive or negative, of the example. One of the advantages of using logistic loss is that it can help to find representations of complex patterns such as transitive relations.

$$\Theta = \underset{\Theta'}{\operatorname{argmin}} \sum_{\tau \in \mathbb{D}^+ \cup \mathbb{D}^-} \log(1 + \exp(-y_{hrt} \cdot f_r(\mathbf{h}_{\Theta'}, \mathbf{t}'_{\Theta'}))) \tag{3.25}$$

In Equation (3.25), the embeddings $\mathbf{h}_{\Theta'}$ and $\mathbf{t}_{\Theta'}$ of the entities h and t are computed with respect to the embedding Θ'.

The alternative pairwise ranking loss function allows to make higher the scores of positive edges than those of negatives. It is calculated with Equation (3.26), where $\tau^+ = (h, r, t)$ represents a positive example and $\tau^- = (h', r', t')$ a negative one, and γ represents the margin separating them. The pairwise ranking loss's main advantage is that it does not assume that negative samples are systematically false, but that they are less valid than the positive ones.

$$\Theta = \underset{\Theta'}{\operatorname{argmin}} \sum_{\tau^+ \in \mathbb{D}^+} \sum_{\tau^- \in \mathbb{D}^-} \max(0, \gamma - f_r(\mathbf{h}_{\Theta'}, \mathbf{t}_{\Theta'}) + f_{r'}(\mathbf{h}'_{\Theta'}, \mathbf{t}'_{\Theta'})) \tag{3.26}$$

Again, in Equation (3.26), the embeddings $\mathbf{h}_{\Theta'}$ and $\mathbf{t}_{\Theta'}$ of the entities h and t are computed with respect to the embedding Θ'.

These optimization functions are implemented using stochastic gradient descent (SGD) using mini-batches. Once the entity and relation embeddings are initialized, a small set of positive edges is sampled from $\mathbb{D}^+$ at each iteration, and one or more negative edges are generated for each positive edge. These positive and negative edges are the training samples in a mini-batch. Then, a gradient step with constant or adaptive learning rates is applied to update the entity and relation embeddings. These embeddings are usually initialized randomly from uniform or Gaussian distributions. Other solutions to initialize the embeddings are using the results of simpler models, e.g., TransE, or using the word vectors from the pre-trained embeddings of a text corpus.

Further, the generation of negative edges can be done by replacing either the head h or the tail t of a given positive edge $\tau^+ = (h, r, t) \in \mathbb{D}^+$ with a random entity sampled uniformly from $\mathcal{E}$, as shown below,

$$\begin{aligned}\mathbb{D}^- = &\{(h', r, t) \mid h' \in \mathcal{E} \wedge h' \neq h \wedge (h, r, t) \in \mathbb{D}^+\} \\ &\cup \{(h, r, t') \mid t' \in \mathcal{E} \wedge t' \neq t \wedge (h, r, t) \in \mathbb{D}^+\}\end{aligned}$$

and, sometimes, the relation can be also randomly corrupted to generate the negative edges, as shown below.

$$\begin{aligned}\mathbb{D}^- = &\{(h', r, t) \mid h' \in \mathcal{E} \wedge h' \neq h \wedge (h, r, t) \in \mathbb{D}^+\} \\ &\cup \{(h, r, t') \mid t' \in \mathcal{E} \wedge t' \neq t \wedge (h, r, t) \in \mathbb{D}^+\} \\ &\cup \{(h, r', t) \mid r' \in \mathcal{E} \wedge r' \neq r \wedge (h, r, t) \in \mathbb{D}^+\}\end{aligned}$$

However, random negative sampling might introduce false negatives, a drawback that can be avoided by giving different replacement probabilities to the head and tail. For example, a higher

head replacement probability for a one-to-n-ary relation, or a higher tail replacement probability for a n-ary-to-one relation. Another strategy is to corrupt the position of h or t using only entities that have appeared in that position for the same relation. Following the survey of Wang et al. [Wang et al., 2017b], the number of negative edges generated per positive edge influences the model performance, where having more negative samples usually allows having better results. A good balance between accuracy and training time is around 50 negative samples per positive sample.

3.4 Link prediction based on KG embeddings

The main training tasks for KGs are link prediction, triple classification, entity classification, and entity resolution. These tasks represent a refinement of the KG input adapted to the specific application.

Link prediction is usually used for measuring the performance of KG models. The objective is to predict entities that have specific relations with others, based on the existing relations in the KG. This task is formally defined as entity prediction or entity ranking, and relation prediction. In entity prediction, we predict h or t, given $(?, r, t)$ or $(h, r, ?)$, respectively. While in relation prediction, we predict r, given $(h, ?, t)$. For example, in (?, *treatment*,*antiviral*) the objective is to predict what the treatment fights against, while in (*virus*, ?,*antiviral*) the objective is to predict the relation between the infectious agent and the medication.

The first step in the task is to learn the entity and relation representations in the KG and then, conduct link prediction as a ranking procedure. Taking the prediction of relations, $(h, ?, t)$, as an example, the KG model replaces r in each triple with all the possible relations r' in the KG as candidate triples to obtain negative samples. Then, a score is used to measure the plausibility of each (h, r', t) to determine its validity. This is calculated by training the model on the KG to get the embeddings and then, applying the scoring function. Ranking the scores obtained in descending order will provide a ranked list of candidate answers. For instance, given the prediction task $(h, ?, t)$, the model will generate an ordered list of relation candidates, e.g., *treatment*, *regulation*, *pathway*, *cause*, using this ranking procedure. The prediction tasks $(?, r, t)$ and $(h, r, ?)$ can be performed following a similar pipeline.

The evaluation of link prediction is usually based on a ranking of correct answers in the ordered list to assess whether the correct answers are ranked before the incorrect answers. Lower ranks indicate better performance. Among the evaluation metrics designed to measure ranks are mean rank (MR) to get the average of predicted ranks, mean reciprocal rank (MRR) to get the average reciprocal ranks, Hits@z to get the proportion of ranks no larger than z, and AUC-PR to get the area under the precision-recall curve.

3.5 Discovering biomedical knowledge using KG embeddings

KGs have been used for diverse biomedical applications, including identifying proteins' functions, improving drug administration, and recommending more adapted and safer drugs for patients. Some examples of KG used for biomedical applications are shown in Figure 3.5.

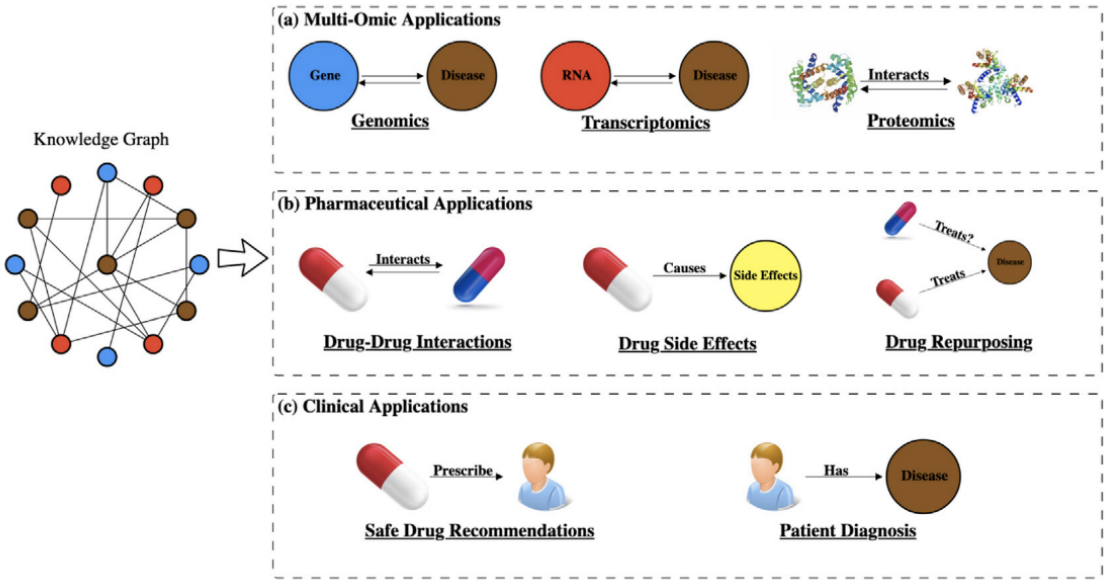


Figure 3.5: Overview of biomedical applications using KGs (Figure taken from [Nicholson and Greene, 2020]).

For instance, Dai et al. [Dai et al., 2015] proposed to apply KGs to identify new properties of drugs. They developed a collaborative filtering system to infer novel associations between drugs and diseases. They first constructed a KG using two bipartite networks, one containing drug-gene interactions and the other disease-gene interactions. Both networks were integrated assuming the drugs are associated with a disease that interacts with the same gene. The rows of the KG adjacency matrix were represented by drugs and the columns by the diseases. The matrix was decomposed based on SVD, and the resulting matrices were used to calculate the similarity scores between drugs and diseases. Other biomedical applications of KGs are the study of the genome, the expression of genes in the transcriptome, and the interaction of the products of these transcripts. Here, KGs represent the connections between the genes and diseases, to predict protein-protein interactions, gene-symptom prioritization, and RNA-disease associations. In the work of Shen et al. [Shen et al., 2017] is proposed to use KGs together with collaborative filtering to establish associations between miRNA and diseases. They constructed a KG using the Human MicroRNA Disease database (HMDD)[13] and represented the adjacency matrix with the

[13] www.cuilab.cn/hmdd

miRNAs as the rows and the diseases as the columns. This matrix is then decomposed using SVD and the new matrices are used to calculate the similarity scores between the miRNAs and the diseases. The higher the score, the higher the likelihood that a miRNA is associated with a given disease. The work of Yang et al. [Yang et al., 2018] proposed to use Node2vec to infer associations between genes and disease symptoms. They first constructed a KG of gene-disease symptoms combining two bipartite graphs, genes with diseases and diseases with disease symptoms. The KG embeddings are obtained, and for each gene-symptom pair is calculated the similarity score. High similarity scores refer to a high association likelihood. This work outperformed methods that did not use KGs, but the validation was challenging since it required manual annotation of the literature. Similarly, Zong et al. [Zong et al., 2019] used a model based on Node2vec to predict drug-target associations. They constructed a heterogeneous network incorporating 12 repositories, including seven types of biomedical entities. Their proposed model integrates an inference-based model and a classification model, that is later fine-tuned using a negative sample selection algorithm. This method performed better than the last methods, obtaining 95.3 % AUC-PR.

Later, Wang et al. [Wang et al., 2020a] developed a knowledge discovery framework called COVID-KG, that extracts multimedia knowledge elements, e.g., entities and their visual chemical structures, relations, and events, from the scientific literature. Then, they used a KG constructed based on this framework to solve the tasks of QA and report generation for drug repurposing to show an example of the exploitation of the KG. Then, Huang et al. [Huang et al., 2020] proposed the graph neural network Graph Edge-conditioned Attention Networks (GEANet), a model that incorporates the representation of hierarchical graphs to infer complex events. They first constructed a KG by integrating biomedical text and the semantic information from the UMLS[14] to incorporate domain knowledge into the pre-trained language model, SciBERT. By evaluating GEANet on the BioNLP 2011 GENIA Event Extraction task, it achieved 1.41 % and 3.19 % F1-score improvement on all events and complex events, respectively, compared to prior baseline models. The work of Sastre et al. [Sastre et al., 2020] proposed a DL model based on Bi-LSTM to extract drug information and structure it in a KG embedding space. The main objective of this system is to evaluate the consistency of drug labeling with ground truth knowledge and the performance of drug interaction prediction. Their results demonstrate that using the embedding space of a KG helps to evaluate drug label accuracy to improve biomedical information extraction systems.

Lai et al. [Lai et al., 2021] presented a framework that incorporates external knowledge for NER and RE, called Knowledge-Enhanced Collective Inference (KECI). This system first constructs a span graph that represents the initial understanding of a text. Then, it links the entities of the text to form a KG containing background knowledge related to these entities.

[14] www.nlm.nih.gov/research/umls

For the predictions, KECI combines the span graph and the KG into another graph using an attention mechanism and feeds it to a GCN. Their results improved the results of baseline models in two benchmark datasets, BioRelEx[15] and ADEs. In the case of Fei et al. [Fei et al., 2021], they evaluated different extraction methods for biomedical information extraction tasks, e.g., NER, RE, and EE. The evaluated methods included vanilla neural networks, general language models, and pre-trained contextualized language models. Then, they proposed to integrate the biomedical KG, BioKGLM[16], into the pre-trained language model BioBERT to improve information extraction. The model integrates the KG using different fusion strategies, and the experimental results show that BioKGLM outperforms baseline extraction models. In the work of Chen et al. [Chen et al., 2022b], they have addressed the high cost and low success rate of clinical trials by inferring knowledge from existing clinical trials to design new ones. They constructed the Clinical Trials Knowledge Graph (CTKG), which consists of nodes representing medical entities from clinical trials (e.g., studies, drugs, and conditions) and edges representing relationships among these entities (e.g., drugs used in studies). The study includes an embedding analysis demonstrating the potential applications of CTKG in various areas, such as drug repurposing and similarity search. Among the potential applications of CTKG we can mention developing more efficient and cost-effective approaches to drug development and clinical trial design.

Further, different methods were discussed for scalable RE from biomedical literature and their integration into KGs [Milošević and Thielemann, 2023]. The methods compared include rule-based approaches and ML-based techniques such as Naive Bayes and Random Forests, as well as transformer models, such as DistilBERT, PubMedBERT, T5, and SciFive. These models were evaluated to test their resilience to unbalanced and relatively small datasets. The results revealed that transformer models perform well on both small datasets (thanks to pre-training on large datasets) and unbalanced datasets. The best-performing model was PubMedBERT, achieving an F1-score of 0.92 when fine-tuned on balanced data. DistilBERT obtained an F1-score of 0.89, offering faster performance and requiring fewer computational resources. Finally, the paper demonstrated that BERT-based models outperform T5-based generative models in this context.

As described in the works above, biomedical information extraction using KG embeddings demonstrates the impact of KGs in the biomedical domain. Integrating diverse medical entities and relationships into a unified KG enables efficient analysis and knowledge discovery. For example, the understanding and treatment of complex diseases, and leads to more informed decision-making in clinical trial design and medical research.

[15] github.com/YerevaNN/BioRelEx
[16] github.com/Baxelyne/BioKGLM

Chapter 4

Comparing pre-trained transformer models for event trigger detection

Contents

Overview

This chapter describes the approach followed to detect biomedical triggers in diverse biomedical domains based on the comparison of different transformer models. We fine-tuned BERT and four of its variants (BioBERT, SciBERT, PubMedBERT, and BioMedRoBERTa) with a set of seven biomedical corpora for event trigger detection. These corpora correspond to different biomedical subdomains. In addition to these data, we included a syntactic and a lexical feature to fine-tune the transformer models, to evaluate if this information improves the detection of triggers. This

stage is developed based on the NER task to identify and classify the triggers in the text. The details of the experiments performed are described below.

4.1 Fine-tuning transformer models to detect event triggers

Biomedical event trigger detection refers to finding the triggers in the text that will allow to construct an event. This information extraction task can be tackled as a multi-class classification problem. Given a sentence $S = s_1, s_2, ..., s_n$ (n refers to the number of words in the sentence), each word s_i is categorized into a class $l \in L$, where L refers to the predefined collection of trigger classes, including the no-trigger class. Downstream tasks, such as event trigger detection, can be performed by making minimal modifications to the transformer architecture, through a process of fine-tuning. One of the main downstream tasks of biomedical text mining is NER, which aims to recognize domain-specific nouns in a biomedical corpus.

Here, we adapted NER to detect event triggers, which implies not only identifying nouns, but also verbs and in some cases adjectives. For this purpose, the biomedical text is first fed to the embedding layer of the transformer models to obtain the contextual representation vectors of the tokens. In addition, we integrated into these embeddings the lexical and syntactic information of the text. These vectors are passed to a classification layer that is added on top of the transformer model to classify the vectors into the event trigger classes. We used two different classification layers separately for comparison, a linear layer and a Bi-LSTM layer. The final output is a sequence of IOB (inside-outside-beginning) labels at the word level, which are the labels of the detected triggers. Since the output from the transformer model produces a sequence of sub-word tokens, labels are only obtained for the first sub-word of the words. Then, all non-first sub-words that follow a first sub-word are given the same label. Figure 4.1 shows an illustration of the description given above.

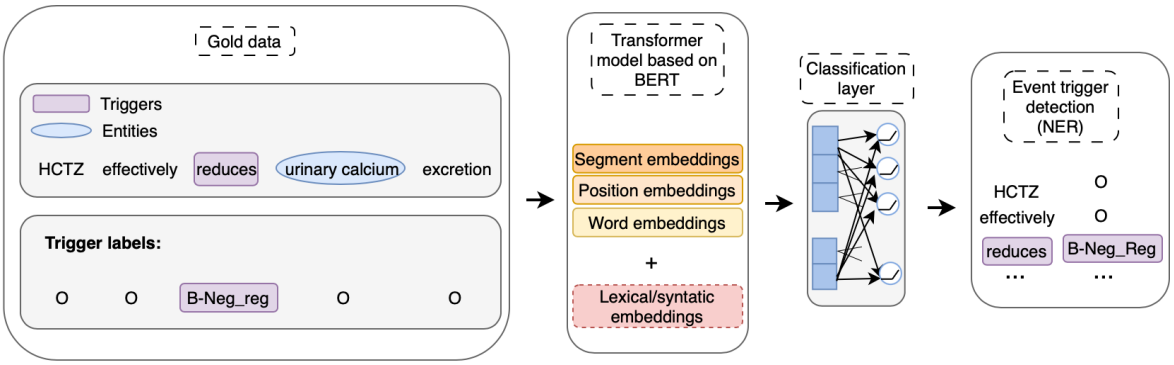


Figure 4.1: Overview of the workflow followed to detect event triggers.

4.1.1 Corpora

Table 5.1 presents the seven corpora used for the fine-tuning of the transformer models. All are in English and publicly available. These corpora were manually or semi-manually annotated by experts and released to be used for the development and improvement of event extraction models. For each corpus are mentioned the number of triggers and events contained, the number of trigger classes, the type of documents, and the size of the train, development and test sets (referring to the number of documents). A brief description of these corpora is included below.

Cancer Genetics (CG) 2013 [Nédellec et al., 2013] contains events of physiological and pathological processes at various levels of biological organization.

Epigenetics and Post-translational Modifications (EPI) 2011 [Ohta et al., 2011] contains the representations of proteins and DNA modification events and the catalysis of these reactions.

GENIA 2011 (GE11) [Kim et al., 2011] and GENIA 2013 (GE13) [Kim et al., 2013] contain general biomolecular events, but the latter was updated with more recent papers.

Infectious Diseases (ID) 2011 [Pyysalo et al., 2011] contains events of biomolecular mechanisms of infectious diseases, virulence, and resistance.

Pathway Curation (PC) 2013 [Nédellec et al., 2013] contains events of target reactions relevant to the development of biomolecular pathway models.

Multi-Level Event Extraction (MLEE) [Pyysalo et al., 2012] contains events of different levels of biological organization ranging from the subcellular to the organism level.

More information about the corpora is presented in Appendix A.1, together with the data statistics in Appendix A.3.

Table 4.1: Statistics of the biomedical corpora.

Dataset	No. Triggers	Trig Classes	No. Events	Documents	Train/Dev/Test
CG	9,790	35	17,248	PubMed abstracts	300/100/200
EPI	2,035	14	2,453	PubMed abstracts	600/200/400
GE11	10,210	10	13,560	MEDLINE abstracts	1,000 (total)
GE13	4,676	12	6,016	PMC full-text	34 (total)
ID	2,155	10	2,779	PMC full-text	15/5/10
PC	6,220	22	8,121	PubMed abstracts	260/90/175
MLEE	5,554	15	6,677	PubMed abstracts	131/44/87

4.1.2 Syntactic and lexical features added for detecting triggers

In addition to the biomedical corpora, we extracted from the text the information on the stems and POS tags to be fed to the transformer models. The stems provide lexical information that corresponds to the words reduced to their word roots, without needing to be an existing word in the dictionary (e.g., the stem of the words "repay" and "payment" is *pay*). This is obtained through stemming, which usually consists of applying a set of rules to remove attached suffixes and prefixes (affixes) from terms without considering the POS or the context of the word occurrence [Jivani et al., 2011]. The POS tags represent syntactic information that provides the categorical differences of the words according to their functions in meaning and grammar within the sentence. POS tagging consists of automatically obtaining the POS tag of each word among the different POS categories corresponding to their syntactical role [Petrov et al., 2011].

The stems of the words were obtained using the 'Snowball Stemmer' module from NLTK-3.4.5[17], while the POS were obtained using spaCy-3.0.0[18] (following the Universal POS tags[19]), using "en_core_web_sm", a pipeline developed for biomedical data. The stems and POS tags were integrated into the transformer model to calculate their embeddings. Then, these embeddings were summed to the text embeddings (token, position, and segment) and fed to the classification layer for fine-tuning. To be able to identify if adding the syntactic and lexical features improves the detection of triggers, we fine-tune the models with and without those features to compare the results.

4.1.3 Transformer language models used for comparison

For the detection of event triggers, we fine-tuned the transformer model, BERT [Devlin et al., 2018], and four BERT variants pre-trained in the biomedical domain, BioBERT [Lee et al., 2020], SciBERT [Beltagy et al., 2019], PubMedBERT [Kalyan et al., 2022], and BioMedRoBERTa [Gururangan et al., 2020].

These models differ from each other by the corpora in which they were trained, the type of pre-training, and the size of the vocabulary, as shown in Table 4.2. SciBERT and PubMedBERT were pre-trained from scratch, meaning that they use a unique vocabulary on their pre-training corpus and include embeddings that are specific for in-domain words. BioBERT and BioMedRoBERTa were pre-trained initializating their embeddings from the BERT.

[17] www.nltk.org/_modules/nltk/stem/snowball.html
[18] spacy.io/
[19] universaldependencies.org/u/pos/

Table 4.2: Pretrained language models based on transformers used for comparison.

Model	Version	Pretraining	Corpus	Text size	Vocab size
BERT	base uncased	from scratch	WikiPedia + BookCorpus	3.3B words	30,522
BioBERT	base v1.1	from BERT	PubMed	4.5B words	28,996
SciBERT	scivocab uncased	from scratch	PMC + Semantic scholar	3.2B words	31,116
PubMedBERT	base uncased	from scratch	PMC + PubMed	3.1B words	30,522
BioMedRoBERTa	base	from BERT	Semantic scholar	7.55B tokens	50,265

4.1.4 Evaluation metrics

We measured three metrics to evaluate the performance of the transformer models, precision (P), recall (R), and F1-score (F1). They were calculated with the equations in 4.1, where TP are the true positives, i.e., the positive samples correctly classified. Positive samples refer to the samples that correspond to the specific class that is being evaluated, while the rest of the samples are negative. FP are the false positives, i.e., the negative samples incorrectly classified and FN are the false negatives or the positive samples incorrectly classified.

$$P = \frac{TP}{TP + FP} \qquad R = \frac{TP}{TP + FN} \qquad F1 = \frac{2 * P * R}{P + R} \tag{4.1}$$

4.2 Experiments settings

4.2.1 Data pre-processing

For the development of the experiments, the training and development sets of all the corpora were merged into one single dataset. This dataset was split into sentences, obtaining a total of 24,819 sentences. Then, a random data partition was applied to use 80% of the data for training and 20% for testing, containing 19,855 and 4,964 sentences, respectively. Each sentence is further split into words by spaces and then, each word into sub-words or tokens following the setting of the BERT tokenization. The sentences split into tokens were then given as input to the transformer model.

All the classes from the trigger labels of each corpus were considered for the final trigger classification, presenting a final set of 58 trigger types (some of these types overlap among the different corpora). The list of trigger types is presented in Appendix A.2.

4.2.2 Implementation details

The experiments were developed using NVIDIA GeForce GTX 1080 Ti (11 GiB) GPU and GeForce RTX 2080 Ti (11 GiB) GPU. The training time varies between models, ranging from 11

hours to 22 hours, with a total time of around 495 hours. All the implementations were done with PyTorch, using the Transformers[20] library, and the models were taken from Hugging Face[21].

Transformer models were trained using the original parameters from BERT, presenting a dropout probability for the attention heads and hidden layers of 0.1, a hidden size of 768, an initializer range of 0.02, a max position embeddings of 512, and an intermediate size of 3,072. The number of attention heads and hidden layers was 12 for each. Adam was used as optimizer and GELU as activation function.

The training parameters of the classification layers, both linear and Bi-LSTM, were set as follows; batch size of training and testing sets of 16, learning rate of 1e-05, and max gradient norm of 10 (since gradient clipping was included). The maximum length of the sentences was set to 256. All the models were trained during 100 epochs on the training set without applying early stopping, and evaluated by measuring the precision, recall, and F1-score.

4.3 Results and perspectives

4.3.1 Fine-tuning transformer models to detect biomedical event triggers

The evaluation results of the fine-tuning of the models for event trigger detection are shown in Table 4.3. The highest values obtained in epochs 10, 30, and 100 are presented in bold, and the highest overall results are presented in bold and underlined.

First, we observe that SciBERT, which is pre-trained from scratch using biomedical and general data, obtained the best overall values in precision, recall, and F1-score. It presented higher values when Bi-LSTM is used as classifier, and especially when extra features are not used. A similar performance is shown when the lexical feature is added but only for the first 10 epochs. However, when the training is done for more than 10 epochs, the performance between SciBERT+POS (syntactic feature) and SciBERT+stem (lexical feature) is very similar. When SciBERT is fine-tuned using a linear classifier, SciBERT+POS achieves the best results, having a difference of around 10 % in comparison to when the lexical feature (SciBERT+stem) is added.

PubMedBERT, a model pre-trained from scratch using biomedical data, achieved the second best performance, being below SciBERT by 4 % when the training is done for 30 epochs, using Bi-LSTM as classifier and without adding the extra features. When PubMedBERT uses Bi-LSTM as classifier, the results are very similar between not adding extra features and adding any of the syntactic and lexical features. These results are also similar to when a linear classifier is used and the extra features are added, noticing that the result is worse when no features are added.

In the case of BERT, which is trained from scratch using data from the general domain, it presents lower results than PubMedBERT by around 5 %. The best results of BERT are obtained

[20] github.com/huggingface/transformers
[21] huggingface.co/

using a linear classifier and not adding extra features, while the results of BERT+POS and BERT+stem are slightly lower and very similar to each other. The same behavior can be noticed when Bi-LSTM is used as classifier. The three last transformer models, SciBERT, PubMedBERT, and BERT, present some similarities in that they were trained from scratch, used very comparable text sizes for their pre-training, and have similar vocabulary sizes.

The two models that presented the lowest performance are BioBERT and BioMedRoBERTa, both pre-trained from the BERT weights, using biomedical and, biomedical and general data, respectively. They present the largest text sizes of all the models. BioBERT used the smallest vocabulary for its pre-training, while BioMedRoBERTa used the largest in comparison with the rest of the models. In both models, it is observed that there is no important difference in performance when the extra features are added, although there is an improvement of around 7 % when using a Bi-LSTM classifier compared to a linear classifier.

In general, the results of all the models are comparable when not adding extra features and adding the syntactic or lexical feature, if the fine-tuning is done for more than 10 epochs. This suggests that adding the syntactic or lexical feature can help to improve performance when the model learning is done over a reduced number of epochs, i.e., less than 10. Also, for most cases, using a Bi-LSTM classifier achieves better results than using a linear classifier, as shown in Figure 4.2. The biggest change in the F1-score of each model is shown during the first 30 epochs, and then there is no improvement of more than 10 % between epoch 30 and epoch 100. This suggests that the most important learning of the models takes place during the first 10 to 30 epochs, being less significant when the number of epochs is greater.

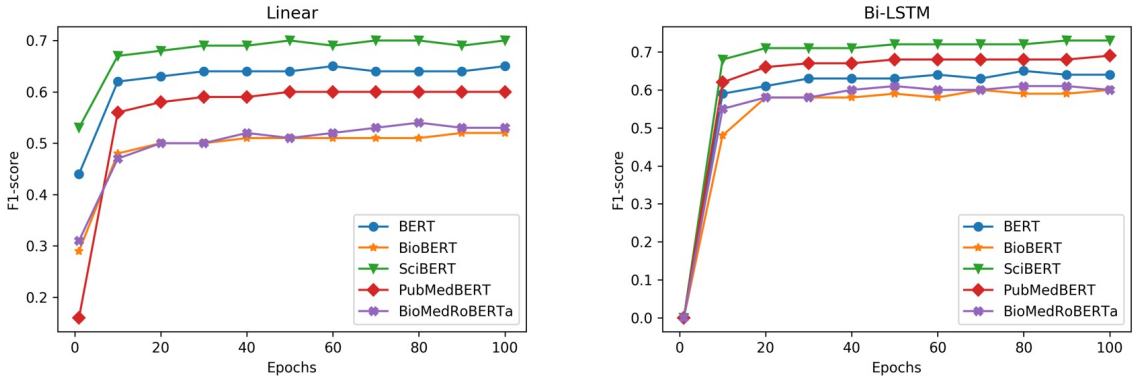


Figure 4.2: F1-score using a linear classifier and a Bi-LSTM classifier (without adding extra features).

Table 4.3: Results of the models' fine-tuning for event detection. Results values and time are obtained from a single run.

Model	**Classifier**	**10 epochs**			**30 epochs**			**100 epochs**			**Time**
		P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**(h)**
BERT	Linear	0.57	0.67	0.62	0.60	0.68	0.64	0.62	0.68	0.65	13
BERT+POS		0.58	0.61	0.59	0.62	0.63	0.62	0.64	0.64	0.64	14
BERT+stem		0.62	0.58	0.59	0.67	0.57	0.61	0.66	0.62	0.63	18
BERT	Bi-LSTM	0.59	0.57	0.57	0.67	0.58	0.62	0.65	0.64	0.64	19
BERT+POS		0.46	0.59	0.51	0.58	0.62	0.60	0.61	0.63	0.62	21
BERT+stem		0.57	0.59	0.57	0.63	0.61	0.62	0.67	0.60	0.63	15
BioBERT	Linear	0.49	0.49	0.48	0.52	0.50	0.50	0.56	0.49	0.51	19
BioBERT+POS		0.54	0.44	0.47	0.49	0.51	0.49	0.51	0.51	0.51	16
BioBERT+stem		0.48	0.50	0.47	0.52	0.46	0.49	0.53	0.48	0.50	18
BioBERT	Bi-LSTM	0.60	0.39	0.45	0.60	0.56	0.58	0.64	0.56	0.59	14
BioBERT+POS		0.57	0.39	0.44	0.59	0.55	0.57	0.61	0.55	0.58	15
BioBERT+stem		0.54	0.50	0.50	0.61	0.52	0.56	0.59	0.57	0.58	20
SciBERT	Linear	0.59	0.64	0.61	0.61	0.65	0.63	0.70	0.70	0.70	11
SciBERT+POS		**0.67**	**0.72**	**0.69**	0.69	0.71	0.70	0.72	**0.73**	**0.72**	16
SciBERT+stem		0.56	0.62	0.58	0.61	0.62	0.61	0.64	0.62	0.63	13
SciBERT	Bi-LSTM	0.65	0.71	0.68	0.71	**0.73**	**0.72**	0.74	0.71	**0.72**	19
SciBERT+POS		0.55	0.56	0.54	0.70	0.71	0.70	0.73	0.70	0.71	22
SciBERT+stem		**0.67**	0.68	0.67	**0.72**	0.68	0.70	**0.75**	0.68	0.71	16
PubMedBERT	Linear	0.49	0.61	0.54	0.58	0.66	0.61	0.58	0.62	0.60	14
PubMedBERT+POS		0.63	0.68	0.65	0.64	0.68	0.66	0.68	0.67	0.67	16
PubMedBERT+stem		0.62	0.66	0.64	0.66	0.67	0.66	0.70	0.67	0.68	18
PubMedBERT	Bi-LSTM	0.57	0.65	0.61	0.66	0.69	0.67	0.67	0.69	0.68	19
PubMedBERT+POS		0.58	0.65	0.61	0.67	0.66	0.66	0.69	0.67	0.68	17
PubMedBERT+stem		0.59	0.66	0.61	0.66	0.69	0.67	0.70	0.66	0.68	18
BioMedRoBERTa	Linear	0.48	0.49	0.47	0.52	0.52	0.51	0.55	0.50	0.52	14
BioMedRoBERTa+POS		0.52	0.56	0.53	0.55	0.51	0.52	0.55	0.53	0.54	13
BioMedRoBERTa+stem		0.50	0.53	0.51	0.51	0.51	0.51	0.53	0.54	0.53	18
BioMedRoBERTa	Bi-LSTM	0.58	0.50	0.53	0.60	0.57	0.58	0.69	0.53	0.59	19
BioMedRoBERTa+POS		0.51	0.56	0.52	0.61	0.53	0.56	0.62	0.56	0.58	15
BioMedRoBERTa+stem		0.51	0.54	0.52	0.57	0.59	0.57	0.60	0.59	0.59	15

In Figure 4.3 and Figure 4.4, we seek to identify the impact of each corpus on the model for the detection of event triggers. For this purpose, SciBERT-*Bi-LSTM* was fine-tuned over 30 epochs without adding extra features by cumulatively adding each corpus one by one and by using each corpus separately.

Figure 4.3 shows the impact of fine-tuning the model by cumulatively adding the corpora. Below each corpus is shown the total number of trigger types after adding the corpus. The recall is improved when CG and EPI are used together and then reduced as the rest of the corpora are added. The precision is affected when EPI is added and also more importantly when GE11 is added. The behavior of the recall and the precision vary differently depending on the added corpus, although when GE13 is added both values are comparable, and as might be expected according to the observed on Figure 4.4 when MLEE is added the values are negatively affected. This behavior may be because when adding a new corpus for fine-tuning the model, some trigger types overlap between the corpora but new ones are also added. This may cause the new trigger types to be less representative in comparison to the number of samples of the other trigger types, affecting the balance of the data. In addition, the context of the different biomedical subdomains of the corpora can play an important role as well, since BERT and its variants build embeddings considering the semantics.

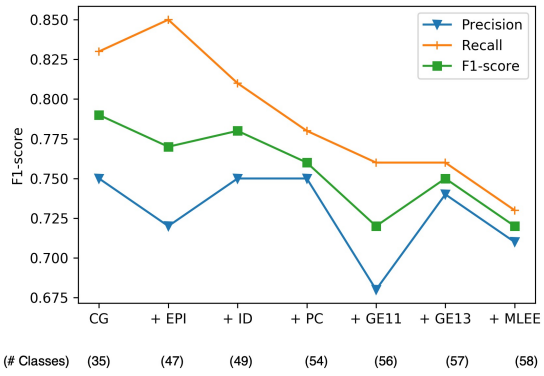


Figure 4.3: Fine-tuning SciBERT-*Bi-LSTM* by cumulatively adding the corpora.

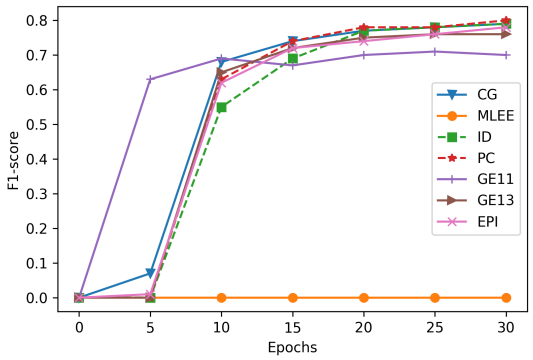


Figure 4.4: Fine-tuning SciBERT-*Bi-LSTM* on the different corpus.

4.3.2 Conclusion and perspectives

This chapter presents the analysis of BERT and four of its variants for detecting biomedical events across diverse biomedical subdomains. Through a comparative evaluation of the pre-trained transformer models and the integration of lexical and syntactic features, we conclude that fine-tuning SciBERT with a Bi-LSTM classifier stands out as the most effective strategy

for event detection in various biomedical subdomains, without the need for additional features. Additionally, we found that fine-tuning the models for 10 to 30 epochs captures most of the learning, with little improvement if the fine-tuning is done for more than 30 epochs, which may even lead to overfitting.

One of the limitations of this analysis is the data imbalance. Some trigger types overlap in the corpora, leading to augmented samples for these classes. As a result, there may be fewer samples for unique classes in each corpus. This could affect the models' performance in the detection of event triggers, as their behavior might differ across subdomains based on available samples. In addition, the use of external tools for obtaining the POS tags and stems may introduce errors. These errors might contribute to the higher performance observed without additional features.

Chapter 5

Integrating KGs into transformer models for argument identification

Contents

Overview

In this chapter, we describe the approach to identify biomedical arguments. Firstly, we construct a KG using semantic types and entity instances from the biomedical corpora. From this KG, we compute the KG embeddings using a set of KG models for comparative analysis. These embeddings are then integrated into the pre-trained transformed model, SciBERT. The transformer model

is then fine-tuned for argument identification, based on the RE task, where the objective is to classify the roles between triggers and candidate arguments.

5.1 Integrating KG embeddings into transformer models to identify arguments

Biomedical argument identification refers to finding the arguments that belong to an event and the role they play in it. This task can be considered as multi-class classification problem. Given a sentence $S = s_1, s_2, ..., s_n$ (n refers to the number of words in the sentence), a pair of named entities in the sentence, s_i and s_j, are related by a relation r_k that is categorized into a class $k \in K$. Here, K refers to the predefined collection of relation classes, including the no-relation class.

Our contribution consists of first representing events as KGs, where triggers and arguments are considered as nodes, and links between them are considered as directed relations from the triggers to the arguments. These links represent the role that the argument is playing concerning the trigger. Each argument can be part of different events and assume different roles in each event. Also, events, which are complete information units, can act as arguments to other events.

From the KG constructed, we calculate its embeddings based on the link prediction task, where the goal is to predict the links between the triggers and the arguments. For this, every event in the corpora is decomposed into simple events, i.e., with a single argument. Then, the simple events are represented as triples (h, r, t), which correspond to the edges in the KG. The heads, h, in the triples, represent the triggers; the relations, r, represent the roles; and the tails, t, represent the arguments. Then, the set of triples is fed to the KG model to calculate the KG embeddings. The output of the model is a low-dimension vector for each element in the triple, i.e., one vector for h, one for r, and one for t. Simultaneously, the word embeddings of the biomedical corpora are calculated using SciBERT, since it was the model that presented the best performance to detect biomedical event triggers in Chapter 4. The transformer model is fine-tuned for the RE task, where the objective is that the model learns to classify the role between the trigger and the argument. For this, the text embeddings and the KG embeddings that correspond to the trigger and the argument are concatenated. These embeddings are then fed to the classification layer. Figure 5.1 shows an illustration of the approach described.

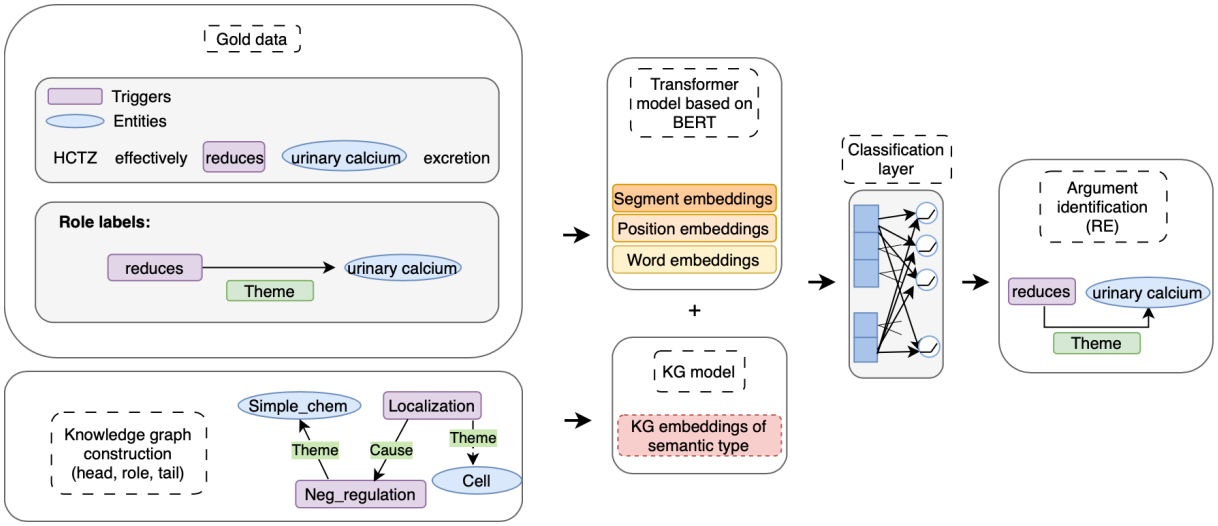


Figure 5.1: Overview of the workflow followed to identify arguments.

5.1.1 Corpora

The corpora used for the development of the experiments, presented in Table 5.1, are the same as those described in Chapter 4. These corpora are used for the construction of the KG, the calculation of the KG embeddings, and the fine-tuning of the transformer model for RE. In Appendix A are presented more statistics about the corpora.

Dataset	No. Triggers	Trig Classes	No. Events	Documents	Train/Dev/Test
CG	9,790	35	17,248	PubMed abstracts	300/100/200
EPI	2,035	14	2,453	PubMed abstracts	600/200/400
GE11	10,210	10	13,560	MEDLINE abstracts	1,000 (total)
GE13	4,676	12	6,016	PMC full-text	34 (total)
ID	2,155	10	2,779	PMC full-text	15/5/10
PC	6,220	22	8,121	PubMed abstracts	260/90/175
MLEE	5,554	15	6,677	PubMed abstracts	131/44/87

Table 5.1: Statistics of the biomedical corpora.

5.1.2 KG models

The KG embeddings are calculated based on the link prediction task using five different KG models for comparison. Three models are based on translational distance: TransE [Bordes

et al., 2013], TransH [Wang et al., 2014], and TransD [Ji et al., 2015], and two models are based on semantic information: SimplE [Kazemi and Poole, 2018] and DistMult [Yang et al., 2014]. Table 5.2 presents a summary of the scoring functions of the KG models.

Table 5.2: Scoring functions of the knowledge graph models used.

Model	Scoring function
TransE	$\|\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|\|$
TransH	$\|\|\mathbf{h}_\perp + \mathbf{d}_r - \mathbf{t}_\perp\|\|$
TransD	$\|\|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|\|$
SimplE	$\frac{1}{2}(\langle \mathbf{h}, \mathbf{v}_r, \mathbf{t} \rangle + \langle \mathbf{t}, \mathbf{v}_{r-1}, \mathbf{h} \rangle)$
DistMult	$(\langle \mathbf{h}, \mathbf{r}, \mathbf{t} \rangle)$

5.1.3 Integration of the KG embeddings in the transformer model

After obtaining the KG embeddings, they are integrated into the contextual representation of the text obtained by SciBERT. This integration was done following the three different strategies described below. Where v_{tr} and v_{ar} represent the embeddings of the trigger and the argument from the transformer language model, respectively, and kg_{tr}, kg_{ar}, kg_r represent the KG embeddings of the trigger, argument, and role, respectively.

$$KG_{tr,ar} = [v_{tr}; kg_{tr}; v_{ar}; kg_{ar}]$$

$$KG_r = [v_{tr}; v_{ar}; kg_r]$$

$$KG_{tr,r,ar} = [v_{tr}; kg_{tr}; v_{ar}; kg_{ar}; kg_r]$$

In $KG_{tr,ar}$, the text embeddings and the KG embeddings that represent the trigger and the argument are concatenated. In KG_r, the KG embeddings that represent the role are concatenated to the text embeddings of the trigger and the argument. Finally, in $KG_{tr,r,ar}$ the text embeddings that represent the trigger and the argument are concatenated to the KG embeddings that represent the trigger, the role, and the argument.

5.1.4 Evaluation metrics

The evaluation of the KG models is based on the standard setup used in the literature. Given a triple (h, r, t) in the test set, a set of corrupted triples is generated by either replacing the entity in h or in t, with any other entity in the total set. Then, the scores of the corrupted triples are calculated along with the score of the true triple. The model is evaluated by ranking all

the triples according to their scores and by calculating the standard evaluation metrics Mean Reciprocal Rank (MRR) and Hits@z, where $z \in 1, 3, 10$ [Kristiadi et al., 2019, Islam et al., 2021].

MRR is a statistic measure for evaluating a process that produces a list of possible responses to a positive test triple, ordered by probability of correctness. It is calculated with Equation (5.1), where Q represents the sample of queries. A query refers to the operation made to retrieve specific knowledge, e.g., $\mathbf{h}, ?, \mathbf{t}$.

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{5.1}$$

Hits@z refers to the average number of times that the positive test triple is among the z highest ranked triples and it is calculated with Equation (5.2). As suggested by the literature for link prediction in KGs, the three Hits@z measured were $z \in \{1, 3, 10\}$.

$$Hits@z = \frac{1}{|Q|} \sum_{i=1}^{|Q|} hit_i, \qquad hit_i = \begin{cases} 1, & \text{if rank}_i \leq z \\ 0, & \text{otherwise} \end{cases} \tag{5.2}$$

Both metrics scores range [0,1], where the higher value demonstrates the better ranking of positive test triples, which means a better prediction performance.

For the evaluation of the transformer model, the performance is based on the three evaluation metrics, precision, recall, and F1-score, described in Chapter 4.

5.2 Experiments settings

5.2.1 Data pre-processing

We used the same merging and split process described in Chapter 4 to obtain the train and test sets used for the development of these experiments. These datasets were obtained by initially merging the seven corpora into one single dataset and splitting it into sentences, obtaining a total of 24,819 sentences. The sentences were then split to have a set of 80 % to train and 20 % to test, containing 19,855 and 4,964 sentences, respectively. The original test datasets were not used for the experiments since the annotations are not publicly available.

We added a set of markers in the sentences to identify the location of the triggers and the arguments. Below is shown an example of a sentence with the markers added, where `[H1]` and `[\H1]` are the markers that allow to identify the trigger and `[H2]` and `[\H2]` allow to identify the argument.

```
HCTZ effectively [H1]reduces[\H1] [H2]urinary calcium[\H2] excretion
```

The trigger, argument, and role types of all corpora were considered for the experiments, presenting a set of 58 trigger types, 22 entity types, and 12 role types. The list of trigger, entity, and role types is presented in Appendix A.2. Figure 5.2 shows the proportion of data per semantic type. Triggers in Figure 5.2 (a) are mainly part of the semantic types *Positive_regulation*, *Gene_expression*, *Negative_regulation*, *Regulation* and *Binding*, while the remaining 35.4 % are part of the other 44 trigger types. In Figure 5.2 (b), entities are mainly part of the semantic types *Protein* and *Gene_or_gene_product*, and the remaining 19.5 % are part of the other 19 entity types. Roles in Figure 5.2 (c) are mainly part of the semantic types *Theme* and *Cause*, while the remaining 10 % are part of the other 10 role types.

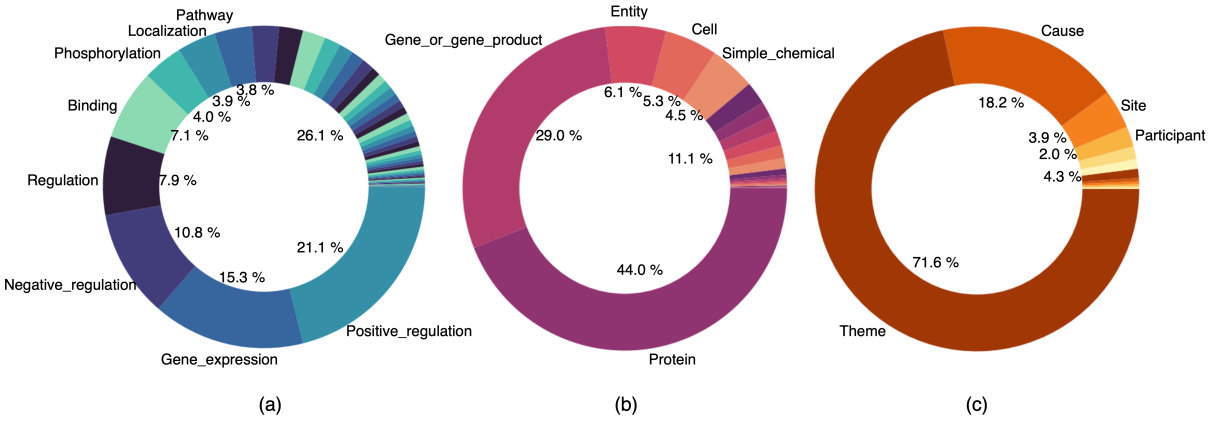


Figure 5.2: Statistics of the dataset, (a) trigger types, (b) entity types, and (c) relation types.

5.2.2 KG construction

For the KG construction, we first constructed a set of KGs from the events found at the document level, where nodes represent the triggers and arguments, and edges represent the roles. The direction of the edges was taken from the direction of the roles in the events. Nodes contain the semantic type of the trigger or argument and the word instance, while edges contain the semantic type of the role. From this step, we obtained 3,453 graphs. Then, the graphs were post-processed, keeping only those that presented at least two nodes and one relation, reducing the total number of graphs to 2,781. All KGs were finally merged into one single graph using the disjoint union function from NetworkX[22]. This final KG contained 104,990 nodes and 60,030 edges or triples.

The edges of the KG were represented in two ways. In the first one, the heads and tails contained the entity or trigger semantic types (e.g. (*Binding*, Theme, *Simple_chemical*), from the event *Binding* in Figure 5.3). Meanwhile, in the second one, the heads and tails contained the

[22] networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.operators.all.disjoint_union_all.html

entity or trigger word instances (e.g. (*associations*, Theme, *O3*), from the event in Figure 5.3).

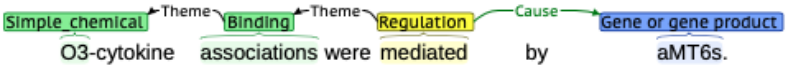


Figure 5.3: Example of sentence annotated with biomedical events.

Table 5.3 presents the statistics of the different edges and the average samples per edge in the KG. We can observe that when edges contain the semantic types of the entities, the average number of samples per edge is higher than when the edges contain the word instances.

Table 5.3: Statistics of the triples obtained from the knowledge graph constructed.

	Types	**Instances**
# of different edges	591	25,051
Avg. of samples per edge	96.32	2.27

5.2.3 Implementation details

Experiments were developed using NVIDIA GeForce GTX 1080 Ti (11 GiB) GPU and GeForce RTX 2080 Ti (11 GiB) GPU. The KG was constructed with NetworkX[23] and the models used to calculate the KG embeddings were implemented with PyTorch, using the base code from [Islam et al., 2021]. They were trained using Adam as optimizer and pair as loss function with a margin value of 4, a learning rate of 1e-04, and a weight decay value of 0.01. The training was conducted during 500, 1000, and 2000 epochs, setting 100, 200, 300, and 768 (to match with the hidden size of the transformer language model) as hidden size (embedding dimension).

The fine-tuning of SciBERT was done with PyTorch, using the Transformers package and the model was taken from Hugging Face[24]. Only the last layer of the model was fine-tuned using the original parameters from BERT. The training was conducted during 10 epochs on the train set and evaluated on the test set, with a batch size of 16, a learning rate of 1e-05, and a max gradient norm of 10. The maximum length of the sentences was set to 256.

5.3 Results and perspectives

5.3.1 KG embeddings calculation based on link prediction

Table 5.5 presents the results obtained with the edges containing the semantic types, while Table 5.6 presents the results with the edges containing the word instances. The highest results

[23] networkx.org
[24] huggingface.co/

are shown in bold.

Table 5.5 shows that for most of the models, the values of MRR and Hits@z increase as the number of training epochs increases. The performance improvement is more significant when the number of training epochs increases from 500 to 1000 than when it increases from 1000 to 2000, where the performance improvement is less important or it even decreases. Regarding the hidden size, having a higher hidden size value does not necessarily imply better performance than having a lower value. Besides, the difference in performance when the hidden size is modified is less important than when the number of epochs is modified.

When the models are trained for 500 epochs, TransE, TransH, and TransD present very similar MRR and Hits@z values, showing an improvement when increasing the hidden size in all cases. In SimplE, the values of MRR and Hits@1 are slightly higher than in the previous models, while for Hits@3 and Hits@10 the opposite effect occurs, obtaining lower values. However, when the hidden size is set to 768, the MRR and Hits@z values increase significantly, obtaining the highest results in the table. In the case of training with 1000 epochs, TransH and TransD present very similar values of MRR and Hits@1, which are the lowest of all the models, while TransE and DistMult present a relatively better performance, especially when the hidden size is set to 200 and 300, respectively.

SimplE presents the highest MRR and Hits@1 when the hidden size is set to 200 and 300, obtaining a value of 0.96 and 0.95, respectively. Hence, the probability of correctly predicting the role between a trigger and an argument reaches around 96 %, while the average number of times a role is correctly predicted is approximately 95 %. On the other hand, it can be observed that when the models are trained for 2000 epochs, TransE, TransH, and TransD improve their MRR and Hits@1 between 0.1 and 0.2 points for all the hidden sizes, in comparison to the training with 500 and 1000 epochs. In the case of SimplE and DistMult, these values are slightly reduced when the hidden size is set to 200 and 300.

Table 5.6 shows that all the results obtained using word instances are worse than those obtained in Table 5.5, using semantic types. This can be partially explained by the number of samples per different edge in each case. The number of different edges using the word instances is much greater than the number of samples that each edge contains, which does not allow the model to have enough information per edge for generalization. Similarly to the results in Table 5.5, the values of MRR and Hits@z tend to improve when the number of epochs and hidden size increase; however, the performance improvement is less significant than in the previous results.

In general, a very similar performance is noticed in the TransE, TransH, and TransD models for all the different hidden sizes and number of epochs, showing a slight improvement of around 0.5 points in the MRR values obtained between 500 epochs and 1000 epochs. Concerning the Hits@z, there is no significant improvement in the values obtained between the different numbers of epochs. However, by increasing the hidden size there is an improvement of around 1.5 points on

average for the three models, especially between the hidden size of 200 and 300. DistMult presents a slightly better performance than the previous models, following the same trend regarding the increase of the epochs and the hidden size, and the values of MRR and Hits@z. However here the improvement in the values obtained between 500 and 2000 epochs is more important, being of around 0.12 points on average. SimplE again showed the highest results, getting the best MRR and Hits@z results when trained for 2000 epochs with a hidden size of 768. In comparison to the rest of the models, its MRR value is higher by around 0.2 points on average, while its Hits@1 is higher than TransE, TransH, and TransD by around 0.45 points on average. From all the results in both tables, SimplE showed the best performance, although the values obtained using word instances are significantly low compared to the use of semantic types.

The advantage of the SimplE results might be because it calculates the embeddings of the inverse relations separately from the relations, which provides different embeddings for entities in the h of the edge and for entities in the t. This gives information about the context and considers the fact that some entities can appear in both, h and t (as in the case of triggers), while others can only appear in t (as in the case of biomedical entities).

Using the semantic type allows words to be grouped into classes and thus generalize according to their context, reducing the space of variables. This provides a greater number of samples per class and, thus, improves the training of the models for the prediction of links. Therefore, this shows that it is not necessary to know an exact word to be able to determine its role as an argument in an event, but knowing its semantic group will allow a better result.

Considering that the data presents an important imbalance between the classes, the results obtained using the semantic types allow to observe that the type of arguments in an event are very related to event types with similar contexts. For example, an argument of the semantic type *Gene_or_gene_product* could be very likely related to an event of type *Gene_expression*. However, it would not be very likely related to an event of type *Localization*, while an argument of the type *DNA_domain_or_region* would most likely be related to an event of type *Localization*. Therefore, using the semantic type helps to know which entities can be related to each other and avoids the problem of data imbalance. However, balancing the data could benefit the model training, being planned to be attempted in the future. Some strategies for data augmentation can be Random Insertion, Deletion, and Swap [Wei and Zou, 2019], Semantic Text Exchange [Feng et al., 2019], or Keyword Replacement [Feng et al., 2020]. Another strategy to balance the data may be automatic data augmentation using a language generation model such as GPT-2 [Radford et al., 2019] or BART [Lewis et al., 2019], which have shown significant performance in data augmentation [Kumar et al., 2020, Anaby-Tavor et al., 2020, Papanikolaou and Pierleoni, 2020].

5.3.2 Incorporating KG embeddings to transformer models

The KG embeddings obtained with SimplE trained during 500 epochs with a hidden size of 768, were used to enrich the transformer model for biomedical argument identification since it presented the best performance for link prediction. The transformer model used was SciBERT since it presented the best performance for biomedical event trigger detection (Chapter 4).

Table 5.4 compares the results of the different strategies followed to integrate the KG embeddings to SciBERT. SciBERT-$KG_{tr,r,ar}$, which integrates the KG embeddings of the trigger, the role, and the argument, presented the highest F1-score. This strategy improved the F1-score by around 18 % compared to when KG embeddings were not integrated. However, we observe that when the KG embeddings of the trigger and the argument were integrated into SciBERT-$KG_{tr,ar}$, the performance improved by only 1 %, while when the KG embeddings of the role were integrated into SciBERT-KG_r, the improvement was of 17 %. This reveals that integrating the KG embeddings into SciBERT improves the performance in the identification of biomedical arguments, especially when the KG embeddings of the role are used.

Table 5.4: Macro-average performance of biomedical argument identification (RE) evaluated on the test corpora.

Model	P	R	F1
SciBERT	0.77	0.70	0.73
SciBERT-$KG_{tr,ar}$	0.80	0.71	0.74
SciBERT-KG_r	**0.93**	0.88	0.90
SciBERT-$KG_{tr,r,ar}$	**0.93**	**0.90**	**0.91**

5.3.3 Conclusion and perspectives

In this chapter, we proposed an approach to identify biomedical arguments using a transformer model enriched with KG embeddings. The KG embeddings were computed through the link prediction task, employing various KG models for comparative analysis. Upon comparing the performance of the different KG models, SimplE emerged as the top performer in terms of MRR and Hits@z (where z takes values from 1, 3, and 10). This suggests that even in cases of imbalanced data within the KG, a KG model can predict links between triggers and arguments, especially if the KG contains semantic information. This happens because semantic information allows grouping different entities into categories, addressing the imbalance issue and consequently mitigating data sparseness.

On the other hand, when the transformer model, SciBERT, is enriched with these KG embeddings, particularly those representing the roles, there is a notable improvement in the

performance to identify biomedical arguments. This suggests that the incorporation of the KG embeddings into the transformer model integrates richer information about the relationships between entities. Thereby allowing for a more precise understanding of the relations between triggers and arguments in events.

Table 5.5: Results of link prediction using semantic entity types.

Model	Hid size	500 epochs				1000 epochs				2000 epochs			
		MRR	**Hits@1**	**Hits@3**	**Hits@10**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
TransE	100	0.42	0.21	0.56	0.81	0.66	0.50	0.78	0.92	0.84	0.77	0.90	0.97
	200	0.51	0.24	0.74	0.92	0.72	0.53	0.88	0.97	0.92	0.89	0.95	0.98
	300	0.54	0.24	0.82	0.93	0.69	0.44	0.92	0.97	0.91	0.87	0.95	0.98
	768	0.59	0.26	0.93	0.98	0.69	0.44	0.96	0.99	0.93	0.87	0.98	0.99
TransH	100	0.36	0.11	0.52	0.82	0.54	0.22	0.86	0.95	0.71	0.48	0.95	0.99
	200	0.50	0.17	0.79	0.93	0.59	0.26	0.92	0.98	0.66	0.37	0.96	0.98
	300	0.56	0.22	0.88	0.97	0.61	0.25	0.95	0.98	0.65	0.33	0.95	0.98
	768	0.59	0.24	0.96	0.98	0.62	0.26	0.97	0.99	0.69	0.39	0.98	0.99
TransD	100	0.42	0.13	0.67	0.86	0.54	0.23	0.84	0.95	0.78	0.61	0.95	0.99
	200	0.42	0.08	0.73	0.92	0.60	0.26	0.95	0.98	0.71	0.44	0.96	0.98
	300	0.57	0.24	0.90	0.99	0.61	0.26	0.95	0.98	0.69	0.43	0.96	0.98
	768	0.60	0.25	0.93	0.98	0.61	0.25	0.95	0.99	0.79	0.61	0.96	0.99
SimplE	100	0.55	0.42	0.60	0.79	0.92	0.89	0.94	0.97	0.94	0.92	0.96	0.98
	200	0.52	0.39	0.58	0.79	0.96	0.95	0.96	0.98	0.91	0.88	0.94	0.97
	300	0.62	0.50	0.69	0.85	0.96	0.95	0.97	0.98	0.94	0.92	0.97	0.98
	768	**0.98**	**0.98**	**0.98**	**0.99**	0.91	0.86	0.94	0.98	0.92	0.89	0.95	0.99
DistMult	100	0.36	0.23	0.40	0.65	0.70	0.60	0.75	0.89	0.82	0.75	0.87	0.98
	200	0.65	0.52	0.74	0.89	0.85	0.79	0.88	0.95	0.83	0.75	0.88	0.98
	300	0.79	0.72	0.82	0.92	0.86	0.81	0.90	0.97	0.81	0.73	0.85	0.96
	768	0.87	0.82	0.91	0.97	0.82	0.75	0.87	0.98	0.82	0.73	0.88	0.99

Table 5.6: Results of link prediction using entity instances.

Model	Hid size	500 epochs				1000 epochs				2000 epochs			
		MRR	**Hits@1**	**Hits@3**	**Hits@10**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
TransE	100	0.28	0.18	0.32	0.43	0.27	0.13	0.36	0.48	0.28	0.13	0.39	0.51
	200	0.28	0.11	0.41	0.55	0.31	0.90	0.49	0.59	0.32	0.10	0.52	0.59
	300	0.31	0.09	0.49	0.59	0.33	0.10	0.55	0.60	0.34	0.11	0.56	0.60
	768	0.33	0.07	0.57	0.61	0.34	0.10	0.58	0.61	0.35	0.12	0.58	0.60
TransH	100	0.32	0.23	0.37	0.47	0.29	0.14	0.39	0.49	0.28	0.11	0.40	0.53
	200	0.28	0.08	0.44	0.57	0.30	0.08	0.50	0.59	0.32	0.09	0.52	0.60
	300	0.29	0.04	0.50	0.59	0.31	0.07	0.54	0.60	0.33	0.09	0.56	0.60
	768	0.32	0.06	0.57	0.60	0.33	0.07	0.58	0.60	0.34	0.08	0.58	0.60
TransD	100	0.30	0.20	0.36	0.46	0.29	0.15	0.39	0.50	0.29	0.13	0.40	0.52
	200	0.27	0.06	0.45	0.56	0.30	0.08	0.50	0.58	0.32	0.09	0.52	0.59
	300	0.29	0.07	0.53	0.56	0.31	0.12	0.39	0.59	0.33	0.13	0.55	0.60
	768	0.32	0.06	0.57	0.60	0.33	0.06	0.58	0.60	0.34	0.08	0.58	0.60
SimplE	100	0.33	0.27	0.34	0.43	0.37	0.31	0.39	0.49	0.46	0.40	0.49	0.58
	200	0.37	0.32	0.39	0.47	0.46	0.41	0.48	0.57	0.55	0.51	0.58	0.62
	300	0.40	0.35	0.42	0.51	0.50	0.45	0.53	0.60	0.58	0.55	0.59	0.63
	768	0.48	0.43	0.50	0.58	0.57	0.54	0.59	0.63	**0.60**	**0.58**	**0.60**	**0.63**
DistMult	100	0.26	0.21	0.28	0.36	0.28	0.23	0.30	0.38	0.33	0.27	0.35	0.46
	200	0.30	0.25	0.31	0.39	0.35	0.29	0.36	0.46	0.41	0.35	0.45	0.54
	300	0.32	0.27	0.33	0.41	0.38	0.33	0.40	0.49	0.46	0.41	0.49	0.56
	768	0.38	0.35	0.41	0.50	0.45	0.40	0.48	0.55	0.50	0.47	0.53	0.57

Chapter 6

Evaluating the knowledge embedded by the transformer model and KG

Contents

Overview

This chapter presents an evaluation of the knowledge embedded by the transformer model and the KG model used for the experiments in Chapters 4 and 5. We first evaluated the performance of the models on the different trigger and role types through a category-grained performance analysis. Afterward, we assessed our model against a baseline model to detect event triggers and identify arguments in biomedical text. The description of the evaluation is presented in the sections below.

6.1 Detecting biomedical event triggers with a transformer model

This section describes the evaluation of SciBERT-*Bi-LSTM* for event trigger detection. We first present a category-grained analysis, where we measured the performance of the transformer model for the different trigger types. Then, we compare our model with two baseline models that presented the state-of-the-art results in the CG task. Finally, to validate our model on a biomedical text, we present a use case where we compared the gold data (manually annotated triggers), with the triggers detected by our model and by a baseline model. The details of each evaluation are described below.

6.1.1 Category grained performance analysis

This analysis consisted of evaluating each trigger type considered for the detection of event triggers. The evaluation was done with SciBERT-*Bi-LSTM* fine-tuned for 30 epochs, using the corpora of the seven biomedical subdomains together. The result values are shown in Table 6.1, in descending order according to the F1-score. The column of support refers to the number of occurrences of each trigger type in the corpora.

From these results, it is observed that most of the trigger types with low support (≤ 5) have also a low or zero F1-score (the last six trigger types in the lower right of the table). This can be explained because the model has not had enough examples of these trigger types to learn how to classify them. However, we can also notice that the support of each trigger type does not necessarily influence the ability of the model to classify the event triggers. Trigger types with high support (≥ 100) do not exceed an F1-score of 0.78, except for *Deglycosylation*, *Process* and *Gene_expression*, which presented F1-scores of 0.91, 0.90 and 0.85, respectively. The event trigger type with the highest support is *Positive_regulation*, with an F1-score of 0.70. These results are significantly lower in comparison to the two trigger types with the highest F1-score, *Amino_acid_catabolism*, and *Glycolysis*, that obtained F1-score of 1.00 and 0.95, respectively, even if they presented a low support of 1 and 10. However, analyzing the corpus, the trigger word for the *Amino_acid_catabolism* type is always "glutaminolysis" in the different sentences of the training and testing sets, which facilitates its detection. For the *Glycolysis* type, the situation is similar, having always as trigger the word *glycolysis* included: "glycolysis", "glycolysis pathway", "aerobic glycolysis", or a variant of the word: "glycolytic".

In contrast, similar to *Positive_regulation*, the trigger types *Negative_regulation* and *Regulation* present a high support (586 and 556, respectively) but relatively lower F1-score (0.75 and 0.61, respectively). This shows that even having a large number of occurrences of some trigger types, the model presents problems in classifying them. This may be because the triggers that correspond to these categories are usually similar words that are modified by negation, in the case of *Negative_regulation*. It also depends on the context to know if they belong to *Regulation*

or *Positive_regulation*, which may be difficult for the model to identify.

Table 6.1: Performance of event detection for each trigger type.

Trigger type	P	R	F1	Support	Trigger type	P	R	F1	Support
Amino_acid_catabolism	1.00	1.00	1.00	1	Entity	0.63	0.74	0.68	398
Glycolysis	1.00	0.90	0.95	10	Degradation	0.68	0.68	0.68	19
Acetylation	0.86	0.99	0.92	82	Transcription	0.65	0.70	0.67	175
Phosphorylation	0.89	0.94	0.91	207	Synthesis	1.00	0.50	0.67	2
Deglycosylation	0.83	1.00	0.91	5	Conversion	0.55	0.75	0.64	28
Process	0.84	0.96	0.90	136	Regulation	0.66	0.57	0.61	556
Deacetylation	0.81	1.00	0.90	13	Blood_vessel_development	0.52	0.72	0.60	18
Metastasis	0.84	0.92	0.88	53	Transport	0.62	0.54	0.59	42
Methylation	0.85	0.90	0.87	73	Planned_process	0.65	0.54	0.59	104
Demethylation	0.75	1.00	0.86	3	Metabolism	0.57	0.57	0.57	7
Ubiquitination	0.82	0.90	0.86	67	Cell_death	0.56	0.58	0.57	43
Gene_expression	0.82	0.88	0.85	754	Growth	0.50	0.67	0.57	3
Hydroxylation	0.82	0.85	0.84	27	DNA_demethylation	0.40	1.00	0.57	2
Glycosylation	0.81	0.84	0.82	67	DNA_domain_or_region	0.57	0.57	0.57	7
DNA_methylation	0.82	0.82	0.82	77	Development	0.49	0.54	0.51	39
Cell_differentiation	0.92	0.73	0.81	15	Dephosphorylation	0.33	1.00	0.50	1
Carcinogenesis	0.78	0.81	0.79	31	Deubiquitination	1.00	0.33	0.50	3
Activation	0.78	0.80	0.79	65	Inactivation	0.44	0.53	0.48	15
Protein_catabolism	0.70	0.87	0.78	30	Catalysis	0.38	0.56	0.45	16
Pathway	0.79	0.76	0.78	168	Breakdown	0.40	0.50	0.44	4
Cell_proliferation	0.77	0.73	0.75	37	Mutation	0.45	0.41	0.43	32
Binding	0.72	0.79	0.75	434	Protein_processing	0.25	1.00	0.40	1
Negative_regulation	0.71	0.79	0.75	586	Anaphora	0.23	0.14	0.18	49
Localization	0.71	0.77	0.74	164	Protein_domain_or_region	0.00	0.00	0.00	5
Infection	1.00	0.56	0.71	9	Cell_division	0.00	0.00	0.00	2
Cell_transformation	0.76	0.67	0.71	39	Catabolism	0.00	0.00	0.00	5
Positive_regulation	0.72	0.68	0.70	1,276	Remodeling	0.00	0.00	0.00	1
Dissociation	0.64	0.78	0.70	9	Translation	0.00	0.00	0.00	2
Death	0.69	0.69	0.69	16	Dehydroxylation	0.00	0.00	0.00	1

Figure 6.1 shows in sentence (1) an example of a *Negative_regulation* trigger, where the trigger instance 'becomes' was incorrectly classified by the model as *Regulation*. The word that provides the information about a negation in the sentence is "unable". However, it is not annotated as part of the trigger, which can make it difficult for the model to differentiate from another type of regulation trigger.

On the other hand, in sentence (2), the *Positive_regulation* trigger "unable to induce" was incorrectly classified as *Negative_regulation*. This can be because the word "unable" is part of the trigger and it represents a negative action, which can also create confusion in the model. A possible solution to this problem is including a module for negation detection, to identify negations even if they are not part of the annotated triggers. Sentence (3) shows an example of ambiguous information, with the *Regulation* trigger, and sentence (4) with the *Positive_regulation* trigger, where for both cases the trigger is "role". This can create confusion in the model, as the same

word is annotated with a different trigger type.

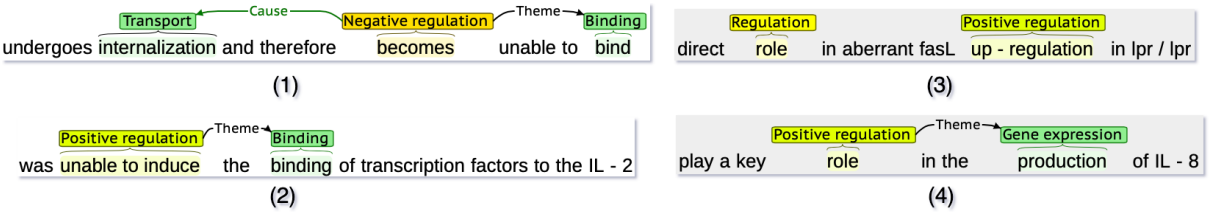


Figure 6.1: Examples of sentences with annotated event triggers.

6.1.2 Performance comparison with baseline models

Table 6.2 compares SciBERT-*Bi-LSTM* fine-tuned in the CG corpus with two baseline EE models, TEES-CNN [Björne and Salakoski, 2018] and DeepEventMine [Trieu et al., 2020] for the detection of event triggers. TEES-CNN is a pipeline model based on a CNN architecture for EE that sequentially applies event trigger detection, argument identification, and the construction of events. DeepEventMine is a joint model based on BERT for EE that simultaneously detects event triggers, identifies arguments, and constructs the final events. Both models presented state-of-the-art results in the CG task.

The results reveal that SciBERT-*Bi-LSTM* achieved a better F1-score than TEES-CNN by around 3% and than DeepEventMine by around 1%. This argues that transformer models achieve slightly better performance than a CNN-based model, as DeepEventMine had demonstrated before. In the case of DeepEventMine, which also uses SciBERT as a base model with a linear classifier to detect event triggers, it obtains a lower recall than SciBERT-*Bi-LSTM*, while the precision is higher. However, the final result of the three models showed very similar values, without presenting a significant advantage in the performance of any of the models.

Table 6.2: Comparison of results of biomedical event trigger detection (NER) on the CG corpus.

Model	**P**	**R**	**F1**
TEES-CNN	0.77	0.81	0.79
DeepEventMine	**0.79**	0.83	0.81
SciBERT-*Bi-LSTM* (ours)	0.78	**0.85**	**0.82**

6.1.3 Use case: validation of event trigger detection in a biomedical text

This section presents the use case of SciBERT-*Bi-LSTM* to detect event triggers in the PubMed abstract with PMID: 19338980[25]. The manual annotations of the abstract were obtained from the development dataset of the CG corpus[26] and were used for evaluation. The event triggers were detected using DeepEventMine and our proposed model for comparison. Then, we developed an error analysis of the models, where we quantified the missing triggers, the incorrectly classified triggers, and the incorrectly detected triggers. Missing triggers are those words that were not identified as triggers, i.e., false negatives, so it is not possible to construct a potential event. Incorrectly classified triggers are those words that were correctly identified as triggers but classified with an incorrect class. These triggers are considered errors in a strict quantitative analysis. However, they are still candidates for the construction of a potential event. Incorrectly detected triggers refer to words detected as triggers that are not in the gold data, i.e., false positives.

Figure 6.2 presents the triggers of the gold data, which were manually annotated. Figure 6.3 presents the results of the event triggers detected with DeepEventMine (using the model pre-trained on the CG task), Figure 6.4 presents the results obtained with SciBERT-*Bi-LSTM* (CG) fine-tuned in the CG corpus and Figure 6.5 presents the results obtained with SciBERT-*Bi-LSTM* fine-tuned in the seven corpora. Figure 6.3 shows that DeepEventMine detected most of the triggers of the gold data, detecting and classifying correctly 13 triggers out of 17. However, it did not detect the triggers "inducible" (sentence 2), "extended" (sentence 4) and "pathway" (sentence 5). It incorrectly classified "knock down" (sentence 4) and it incorrectly detected "required" (sentence 3), "stimulation" (sentence 3), and "role" (sentence 5). Figure 6.4 shows that SciBERT-*Bi-LSTM* (CG) presented slightly better results than DeepEventMine, detecting and classifying correctly 14 triggers. Here, the missing triggers are "inducible" (sentence 2) and "extended" (sentence 4), the incorrectly classified is "knock down" (sentence 4) and the incorrectly detected are "required" and "stimulation" (both in sentence 3). Figure 6.5 shows that SciBERT-*Bi-LSTM* fine-tuned in the seven corpora was overall less performant compared to previous models. It only detected three triggers correctly and it incorrectly detected "down" (sentence 4).

In the case of SciBERT-*Bi-LSTM* (CG) and DeepEventMine, the results were very similar, both detecting some words as triggers (false positives) that do not appear in the gold data. However, although these detected triggers are considered as errors in a quantitative analysis, in a qualitative analysis they may be less penalized. As they are words that can build new events, they allow more information to be extracted and thus, potentially discover new knowledge. This can enrich the information that had been initially identified through manual annotation.

[25] pubmed.ncbi.nlm.nih.gov/19338980/
[26] 2013.bionlp-st.org/tasks/cancer-genetics-cg-task

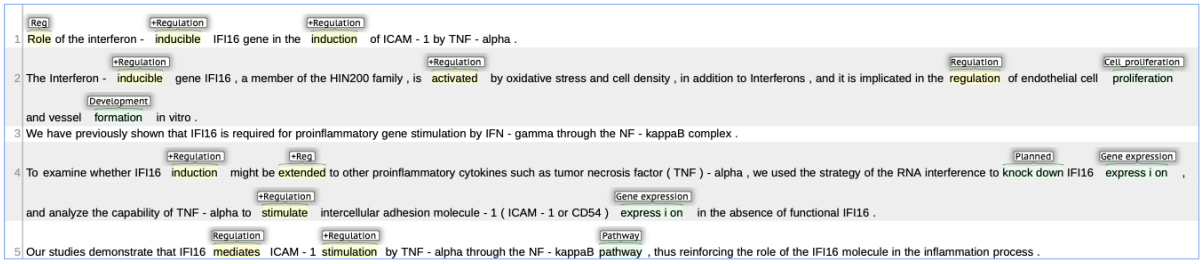


Figure 6.2: Abstract with manually annotated triggers (PMID:19338980).

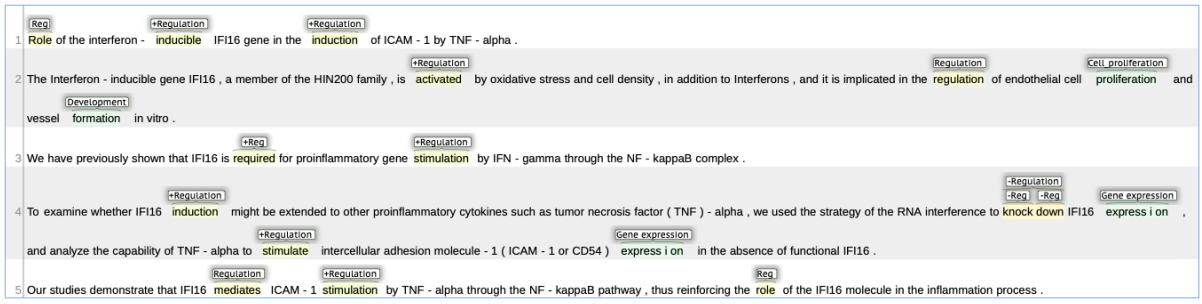


Figure 6.3: Detection of biomedical event triggers with DeepEventMine (PMID:19338980).

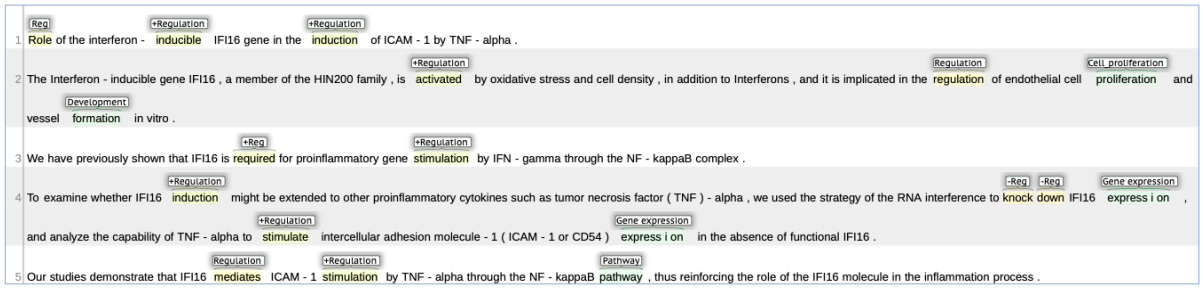


Figure 6.4: Detection of biomedical event triggers with SciBERT-*Bi-LSTM* (CG) fine-tuned in the CG corpus (PMID:19338980).

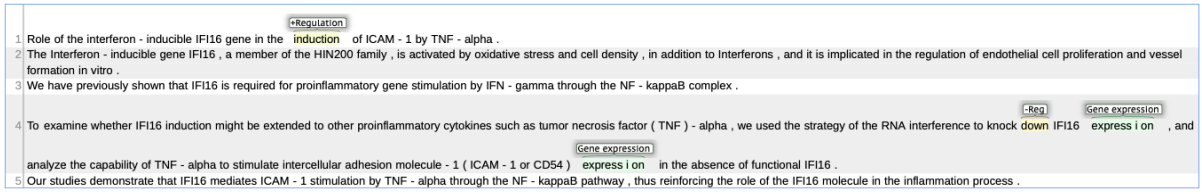


Figure 6.5: Detection of biomedical event triggers with SciBERT-*Bi-LSTM* fine-tuned in the seven corpora (PMID:19338980).

Table 6.3 summarizes a quantitative error analysis of these results, where SciBERT-*Bi-LSTM* (CG) represents the results obtained with SciBERT-*Bi-LSTM* fine-tuned in the CG corpus and SciBERT-*Bi-LSTM* fine-tuned in the seven corpora. In the case of DeepEventMine and SciBERT-*Bi-LSTM* (CG), the biggest sources of errors were missing triggers and incorrectly detected triggers. However, SciBERT-*Bi-LSTM* (CG) presented the highest percentage of correctly detected triggers, and missing and incorrectly detected triggers were lower than in DeepEventMine. In SciBERT-*Bi-LSTM* the largest source of errors was missing triggers, exceeding by around 60 % of the triggers that were correctly detected. The percentage of triggers incorrectly classified was the same for the three models, corresponding to the same word trigger in all the cases. With these results, it can be seen that even if SciBERT-*Bi-LSTM* was trained with the seven corpora, SciBERT-*Bi-LSTM* (CG) has a better performance for the detection of biomedical event triggers. This is demonstrated by comparing the results with the gold data, in addition to the fact that SciBERT-*Bi-LSTM* (CG) has detected new triggers that can potentially build events. This may be because when the model is trained with the seven corpora, the data may become more dispersed since no corpus is balanced, and therefore it fails to generalize. In addition, it is possible that by taking an abstract from the CG corpus as a use case, the model fine-tuned specifically in the CG corpus is much more refined to detect triggers in text containing information of this specific subdomain. While the fine-tuned model with the seven corpora may be suffering from under-fitting in all the subdomains.

Table 6.3: Error analysis of event trigger detection.

Error type	**DeepEventMine**	**SciBERT-*Bi-LSTM* (CG)**	**SciBERT-*Bi-LSTM***
Correct triggers	76.47 % (13/17)	82.30 % (14/17)	17.64 % (3/17)
Missing trigger	17.65 % (3/17)	11.76 % (2/17)	76,47 % (13/17)
Incorrect trigger class	5.88 % (1/17)	5.88 % (1/17)	5.88 % (1/17)
Incorrect detection	3	2	0

6.2 Identifying biomedical arguments with a KG-enriched transformer model

This section describes the evaluation of SciBERT-*Bi-LSTM* and SciBERT-$KG_{tr,r,ar}$ on the identification of event arguments. We first present a category-grained analysis, when we measured the performance of the model for the different role (relation) types. Then, we compared our model with the two baseline models presented in the last section. Finally, we present a use case where we compared the gold data (manually annotated roles), with the arguments identified with

our models and with a baseline model. The details of each evaluation are described below.

6.2.1 Category grained performance analysis

This analysis consisted of evaluating each role type considered for the identification of arguments, i.e., relations between triggers and the arguments. The evaluation was done using the corpora of the seven biomedical subdomains together and SciBERT-$KG_{tr,r,ar}$ as model, trained for 10 epochs. For each role type are calculated the precision, recall, and F1-score. The result values are shown in Table 6.4, in descending order according to the F1-score.

Table 6.4: Performance of argument identification for each role type.

Role type	**P**	**R**	**F1**	**Support**
Theme	0.99	1.00	0.99	11,222
Cause	0.99	0.97	0.98	2,835
Negative	0.98	0.98	0.98	4,260
Site	0.96	0.97	0.97	529
Participant	0.96	0.96	0.96	328
AtLoc	0.92	0.92	0.92	118
Instrument	0.87	0.92	0.90	98
Contextgene	0.90	0.90	0.90	21
ToLoc	0.90	0.86	0.88	95
Product	0.92	0.78	0.84	58
Sidechain	0.80	0.89	0.84	9
FromLoc	1.00	0.54	0.70	24

The support refers to the number of occurrences of each role type in the corpora. These results show the imbalance of the data in the support of each role type, where the role *Theme* presents the highest support with a value of 11,222, while *Sidechain* presents the lowest support with a value of 9. It is also possible to observe a relationship between support and F1-score, where the higher the support, the better the performance. However, the difference in performance between the different roles is less significant than the difference in support. While *Theme* presents an F1-score of 0.99, *FromLoc* presents the lowest F1-score with a value of 0.70 and support of 24. The F1-score of this last is negatively affected by the recall, presenting a significantly low value compared to the other role types. This means that the model has not been able to correctly identify the roles with this class, i.e., this role type presents a high level of false negatives.

If we analyze the corpus, the *FromLoc* role is most of the time part of a *Localization* or *Transport* event. Furthermore, most of the time it relates triggers to arguments with the semantic

type *Cellular_component* or *Cell*. These are characteristics that this role shares with the *ToLoc* role, which makes the classification of *FromLoc* confusing for the model since it is very close in context to *ToLoc* and has fewer learning samples. The difference between these two types of roles depends on the context of the text and the keywords that help to classify. For example, in some cases, the trigger words give enough context to identify the role, such as the trigger word "export" in sentence (1) of Figure 6.6, where the *Cellular_component* argument plays the *FromLoc* role. Prepositions close to the triggers may also give relevant information to identify the type of roles, such as "from", "in" and "to", as shown in sentences (2) and (3) of Figure 6.6. However, there are some cases where the same argument plays both types of roles in the same event, as shown in Figure (4) of Figure 6.6, which makes the classification much more complex. This can create confusion in the model, as the same argument plays two different roles in the event.

In the case of the *Sidechain* role, despite having so few samples, its performance is comparable to the *Product* role, which has more than nine times its number of samples. This is because it appears in a very specific context, always being part of *Glycosylation* and *Deglycosylation* type events. In addition, it always appears to relate these triggers to arguments with the semantic type *Entity*. This makes it easier for the model to learn how to classify it by having a constant context, achieving better generalization.

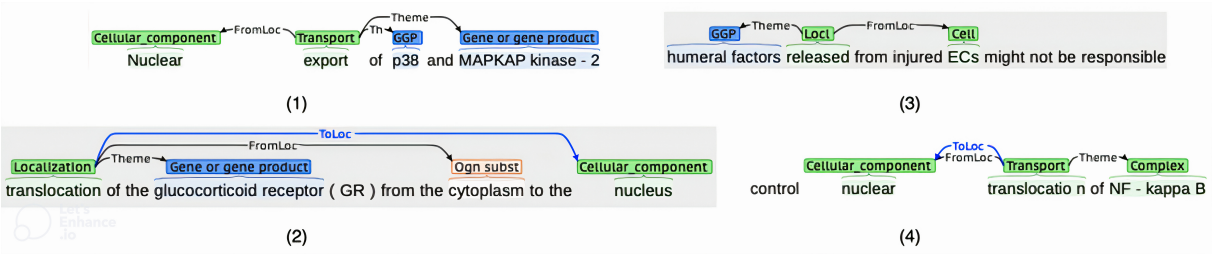


Figure 6.6: Examples of sentences with annotated event triggers.

6.2.2 Symbolic representation of role predictions

In this section, we present a symbolic representation of the predictions obtained for argument identification. The main objective of presenting this representation is to be able to compare the results of each model and to identify patterns in these results. A pattern refers to a distinctive arrangement of elements within a set of information. It is a recognizable configuration that can be repeated or exhibit consistent characteristics, making it distinguishable from other elements within the context of the problem. For this, we compare the results of SciBERT-*Bi-LSTM*, KG and SciBERT-$KG_{tr,r,ar}$. The predictions of KG were calculated with the link prediction task, using SimplE as the KG model. The models were fine-tuned in the seven corpora, using the train and validation tests, except for the CG corpus, from which we only used the train set. The

validation set of the CG corpus was used for the predictions, to compare the results with the manual annotations of the dataset. The CG validation dataset contains 3,736 roles annotated with eight different role types. The statistics of the dataset and the complete set of predictions are included in Appendix B.

Table 6.5 shows a symbolic representation of the predictions obtained by each model, as well as the real value (True) and the number of samples for each pattern followed in the predictions. If the model predicted the correct role, it has a value of 1, while if the prediction is incorrect, it has a value of 0.

Table 6.5: Symbolic representation of role predictions on the CG corpus (validation dataset).

True	**SciBERT-*Bi-LSTM***	**KG**	**SciBERT-$KG_{tr,r,ar}$**	**No. Samples**
1	0	0	0	28
1	0	0	1	48
1	0	1	0	23
1	0	1	1	177
1	1	0	1	533
1	1	1	0	1
1	1	1	1	2926
		Total		3736

Of all the predictions obtained by the different models, we identified seven patterns when comparing the results between them. We can observe in the first pattern of the table that when there is indeed an argument related to the trigger (True), there are 28 samples where none of the models managed to predict the correct role. The second pattern shows that in 48 samples only SciBERT-$KG_{tr,r,ar}$ predicted the roles correctly. In the third pattern, it is observed that there were 23 samples where only KG predicted the roles correctly. The fourth pattern shows that there were 177 samples where KG and SciBERT-$KG_{tr,r,ar}$ correctly predicted the roles, while SciBERT-*Bi-LSTM* did not. In the fifth pattern, 533 samples were correctly predicted by SciBERT-*Bi-LSTM* and SciBERT-$KG_{tr,r,ar}$, but not by KG. The sixth pattern shows that there was only one sample where SciBERT-*Bi-LSTM* and KG correctly predicted the role and SciBERT-$KG_{tr,r,ar}$ did not. Finally, the seventh pattern shows when the three models correctly predicted the roles, to which a large part of the samples correspond. As the results show, about 78 % (2,926 samples) of the roles were correctly predicted by all models.

This suggests that both models, KG and SciBERT-*Bi-LSTM*, integrate information about the text that allowed the identification of most of the arguments in the dataset. Furthermore, by joining the information from both models, the roles are also correctly predicted, suggesting that

the information integrated into the embedding vectors of both models agrees.

On the other hand, about 14 % (533) of the roles were correctly predicted by SciBERT-*Bi-LSTM* and SciBERT-$KG_{tr,r,ar}$, regardless of whether KG did not correctly predict the role. This suggests that the contribution of the transformer model is what mostly defines the final prediction when both models are integrated. The same is observed in the third pattern, corresponding to around 0.6 % of the samples, where the prediction of SciBERT-$KG_{tr,r,ar}$ is incorrect even when the KG embeddings, which correctly predicted the role, were integrated. However, in around 5 % (177) of the roles, the incorrect prediction of SciBERT-*Bi-LSTM* was modified when integrating the KG embeddings, now predicting the correct role.

6.2.3 Performance comparison with baseline models

Table 6.6 compares our models, SciBERT-*Bi-LSTM* and SciBERT-$KG_{tr,r,ar}$, fine-tuned in the CG corpus with TEES-CNN [Björne and Salakoski, 2018] and DeepEventMine [Trieu et al., 2020] for the identification of arguments.

The results reveal that SciBERT-*Bi-LSTM* achieves better performance than the two baseline models, showing that our proposal based on a transformer model improves the F1 of the CNN-based model by around 10 %. In the case of DeepEventMine, which is also based on the transformer model, SciBERT, it uses a linear classifier while our model uses a Bi-LSTM classifier. Furthermore, DeepEventMine follows a joint approach, i.e., the three stages of EE (event trigger detection, argument identification, and event construction) are trained simultaneously, while we follow a pipeline approach, training each stage separately with a different model. The contrast in the learning approaches of each model and the type of classifier can explain the difference in the performance of both, where SciBERT-*Bi-LSTM* surpassed the F1-score of DeepEventMine by 9 %. The best performance was achieved by SciBERT-$KG_{tr,r,ar}$, obtaining an F1-score that surpassed TEES-CNN by 16%, DeepEventMine by 15%, and SciBERT-*Bi-LSTM* by 6 %. This shows that the integration of the KG embeddings of the semantic types in the transformer model enriches the vector information. This allows the model to better classify the roles of the arguments and reduce EE errors related to argument identification.

Table 6.6: Comparison of results of biomedical argument identification (RE) on the CG corpus.

Model	**P**	**R**	**F1**
TEES-CNN	0.65	0.63	0.64
DeepEventMine	0.63	0.67	0.65
SciBERT-*Bi-LSTM* (ours)	0.80	0.71	0.74
SciBERT-$KG_{tr,r,ar}$ (ours)	**0.87**	**0.75**	**0.80**

6.2.4 Use case: validation of argument identification in a PubMed abstract

This section presents the use case of SciBERT-$KG_{tr,r,ar}$ to identify event arguments in the PubMed abstract with PMID: 19338980[27]. The manual annotations of the abstract were obtained from the development dataset of the CG corpus[28] and were used for evaluation. The biomedical entities were automatically annotated with DeepEventMine and the event triggers were taken from the use case of subsection 6.1.3 to be given as input to the model, together with the text. The event arguments were identified using DeepEventMine and our proposed model for comparison. We developed an error analysis of the models, where we quantified the missing arguments, the incorrectly classified roles, and the incorrectly identified arguments. Missing arguments refer to when the relationship between a trigger and an argument is not identified, i.e., false negatives. Incorrectly classified roles are those relationships between triggers and arguments that were correctly identified but classified with an incorrect class. These roles are considered errors in a strict quantitative analysis. However, they still construct a candidate event that shows a relationship between the trigger and the argument. Incorrectly identified arguments refer to identified relationships between triggers and arguments that are not in the gold data, i.e., false positives.

Figure 6.7 presents the triggers, the arguments, and their roles (relationships between triggers and arguments) in the gold data. Figure 6.8 presents the results of the arguments identified with DeepEventMine. Figure 6.9 presents the results obtained with SciBERT-$LSTM$ and Figure 6.10 presents the results obtained with SciBERT-$KG_{tr,r,ar}$, both fine-tuned in the seven corpora. Figure 6.11 presents the results obtained with SciBERT-$LSTM$ and Figure 6.12 presents the results obtained with SciBERT-$KG_{tr,r,ar}$, both fine-tuned in the CG corpus.

It can be seen that all figures have missing biomedical entities (automatically identified with DeepEventMine), which has automatically caused errors due to missing arguments. These are some of the sources of errors affecting the results since the same biomedical entities were used as input for all models. Figure 6.8 shows that DeepEventMine identified most of the arguments and their roles present in the gold data.

However, having identified the triggers with the same model, causes another source of errors due to missing triggers, in addition to missing biomedical entities, since they make it impossible to create a candidate event. For example, in sentence 1 it can be seen that the only missing role is the *Cause* relation of the *Positive_Regulation* event candidate, corresponding to the "inducible" trigger word. However, this error is due to the missing biomedical entity, causing the error to propagate to the argument identification. Similarly, in sentence 2 the candidate event *Positive_Regulation* corresponding to the "inducible" trigger word was not detected. This

[27] pubmed.ncbi.nlm.nih.gov/19338980/
[28] 2013.bionlp-st.org/tasks/cancer-genetics-cg-task

causes there to be two unidentified arguments, the argument *GGP* that corresponds to the missing entity "interferon" playing the role of *Cause* and the argument *GGP* that corresponds to the entity "IFI16" playing the role of *Theme*. On the other hand, some arguments have been identified that do not appear in the gold data. For example, the arguments of sentence 3, which have been related to the previously detected false positive triggers.

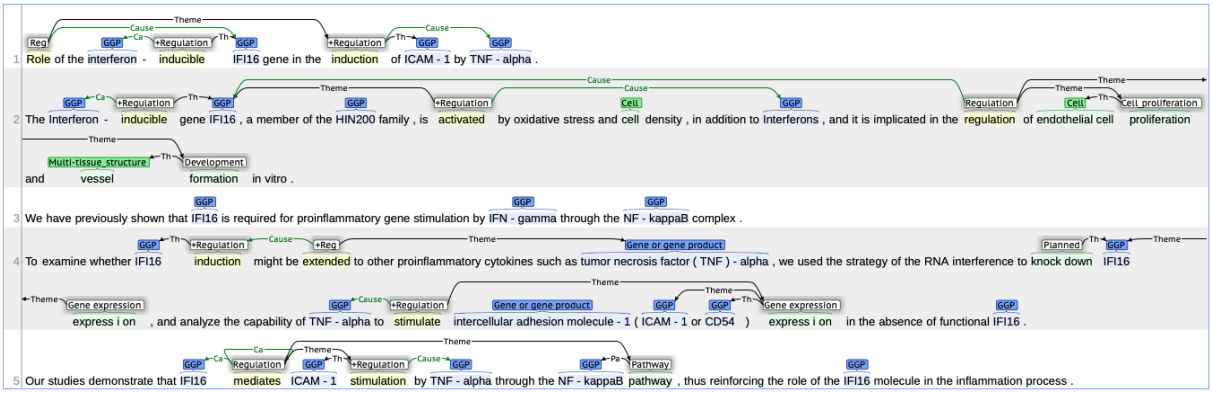


Figure 6.7: Abstract with manually annotated relations (PMID:19338980).

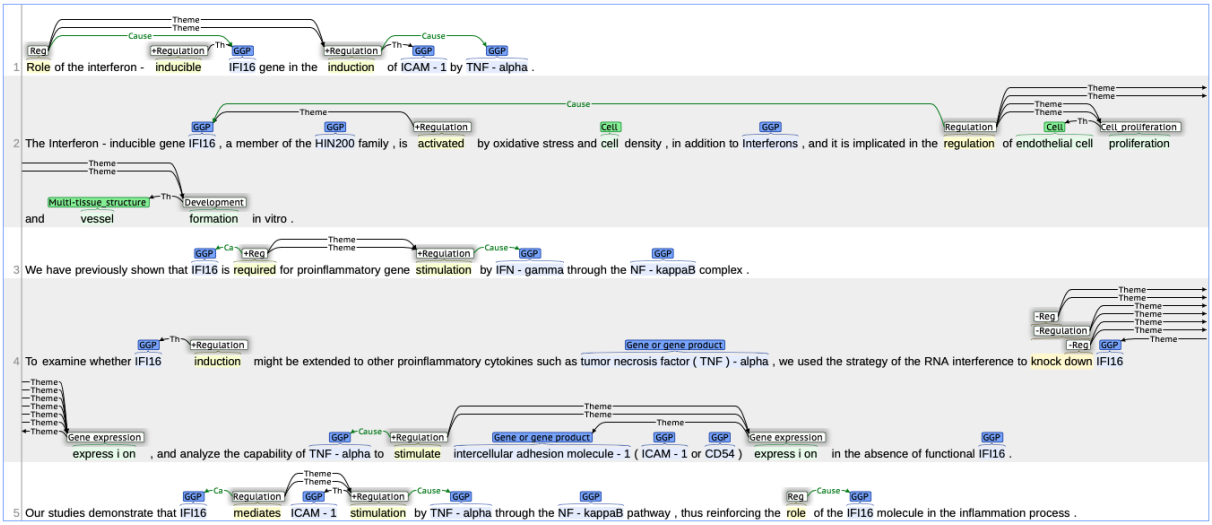


Figure 6.8: Identification of biomedical arguments with DeepEventMine (PMID:19338980).

In Figure 6.9 is observed that there were some missing arguments because the trigger (detected with SciBERT-*Bi-LSTM*) or the biomedical entity (detected with DeepEventMine) had not been previously detected. For example in sentence 1, with the missing biomedical entity that corresponds to the word "interferon", or in sentence 2, with the missing trigger that corresponds to the word

"inducible". Also, there are many more arguments identified than in the gold data, that is, false positives. This can be seen more evidently in sentences 2 and 4. In Figure 6.10 the problem of missing arguments due to missing triggers (detected with SciBERT-*Bi-LSTM*) and biomedical entities (detected with DeepEventMine) is also observed. It is also possible to witness the problem of false positives in sentences 2 and 4, as in Figure 6.9.

However, these results present more missing arguments than the previous two models, even if the KG embeddings were integrated into the model. For example, the missing argument *GGP* corresponds to the "IFI16" word that should play the *Theme* role in the *Positive_Regulation* candidate event in sentence 1. Also, the two arguments related to the *Regulation* event candidate correspond to the trigger word "Role" in the same sentence. Errors due to false negatives can be observed in all sentences, except in sentence 2. However, in this last sentence arises the problem of creating incomplete candidate events, since, despite that two roles were identified in the sentence, these correspond to two interrelated triggers, i.e., they are nested events. Nevertheless, as they are candidate events that are not related to any biomedical entity, the information they present does not give any biomedical details. Therefore, the unit that is built with these events does not extract precise information that can be used later to discover knowledge.

Figure 6.9: Identification of biomedical arguments with SciBERT-*Bi-LSTM* fine-tuned in the seven corpora (PMID:19338980).

Figure 6.10: Identification of biomedical arguments with SciBERT-$KG_{tr,r,ar}$ fine-tuned in the seven corpora (PMID:19338980).

In Figures 6.11 and 6.12 some arguments were unidentified because the triggers (detected with SciBERT-*Bi-LSTM* (CG)) or the biomedical entities (detected with DeepEventMine) were not previously detected. It can be observed that the arguments identified are very similar in both models, but SciBERT-*Bi-LSTM* (CG) shows more false positives. In the case of SciBERT-$KG_{tr,r,ar}$ (CG), the model predicts fewer relationships but is more selective in identifying true arguments. This means that, by adding the information from the KG embeddings, the model has reduced false positives and false negatives in its predictions.

On the other hand, it can be noted in both models that training with the CG corpus presents better performance than when training is done with the seven corpora. This is observed in candidate events that were incompletely constructed due to missing arguments, e.g., missing arguments in sentence 2 or incomplete nested events in sentence 3. This supports the behavior of the models presented previously in event detection, where the fine-tuned model with the CG corpus presents better performance than the fine-tuned model with the seven corpora. The fine-tuned model in the CG corpus, by learning from data in a single subdomain (i.e., with a smaller vocabulary, fewer labels, and less dispersed), can generalize more easily. Therefore, it will be able to identify arguments with better performance than a model that has learned in the seven unbalanced corpora with subdomains and, consequently, much more dispersed data.

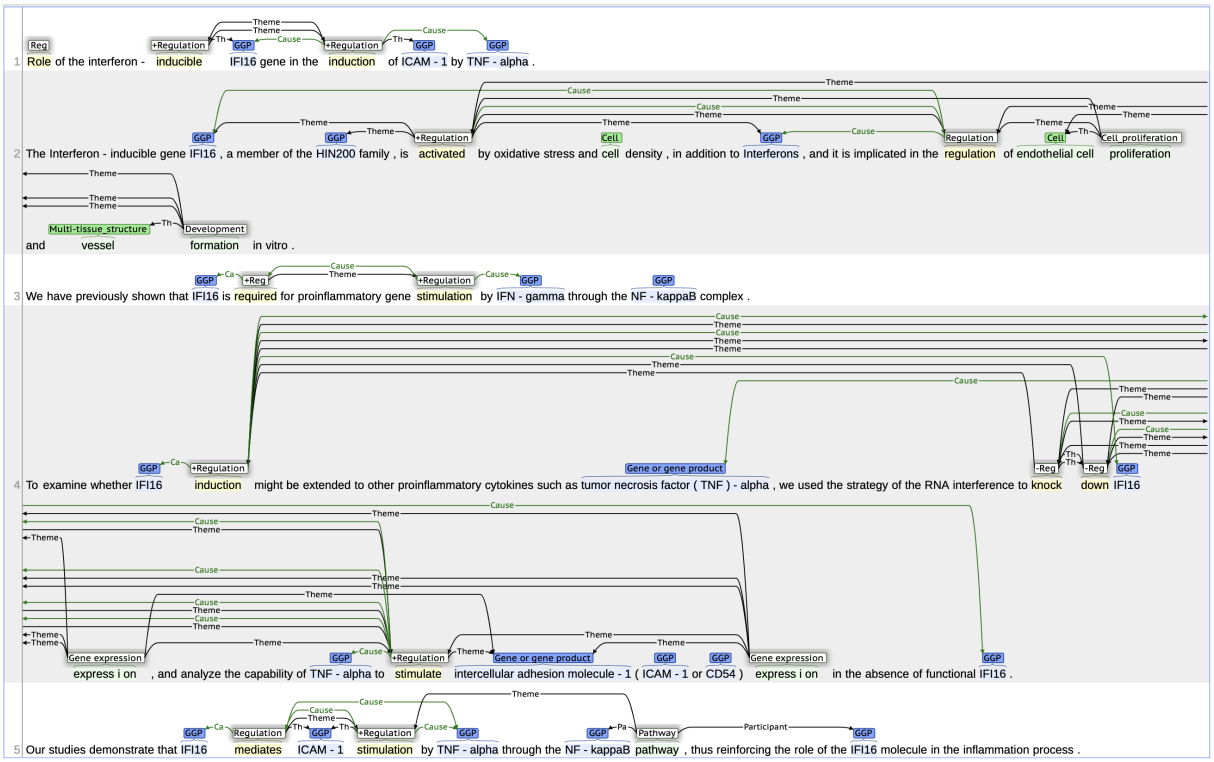


Figure 6.11: Identification of biomedical arguments with SciBERT-*Bi-LSTM* (CG) trained in the CG corpus (PMID:19338980).

Figure 6.12: Identification of biomedical arguments with SciBERT-$KG_{tr,r,ar}$ (CG) fine-tuned in the CG corpus (PMID:19338980).

Table 6.7 summarizes a quantitative analysis error of the results presented above. The two models that had the most correctly identified arguments are DeepEventMine and SciBERT-*Bi-LSTM*. In both models, there was the same number of missing arguments, while the most notable difference between them is the incorrectly identified arguments, i.e., false positives. SciBERT-*Bi-LSTM* was the model that presented the most false positives among all the models. The model that presented the lowest percentage of arguments classified correctly was SciBERT-$KG_{tr,r,ar}$, with missing arguments being its first source of error. It can be noticed that the only two models that presented errors in the classification of the roles were the fine-tuned models in the CG corpus. That is, these models predicted those relationships between triggers and arguments correctly but the assigned class was incorrect.

Table 6.7: Error analysis of event argument identification.

Error type	**DeepEventMine**	**SciBERT-*Bi-LSTM* (CG)**	**SciBERT-$KG_{tr,r,ar}$ (CG)**	**SciBERT-*Bi-LSTM***	**SciBERT-$KG_{tr,r,ar}$**
Correct argument	61.29 % (19/31)	45.16 % (14/31)	35.48 % (11/31)	61.29 % (19/31)	32.26 % (10/31)
Missing argument	38.71 % (12/31)	41.93 % (13/31)	48.39 % (15/31)	38.71 % (12/31)	67.74 % (21/31)
Incorrect role class	0 % (0/31)	13.00 % (4/31)	16.13 % (5/31)	0 % (0/31)	0 % (0/31)
Incorrect identification	8	26	22	29	25

Furthermore, it is notable that all our proposed models presented more false positives than DeepEventMine. These identified arguments that do not exist in the gold data can be new potential candidate events that allow new events to be built and thus discover new knowledge. However, they can also create new errors in the construction of these units of knowledge that may be difficult to identify by not having a baseline for comparison. It is important to note that in event extraction, being a complex task that is made up of different subtasks, qualitative analysis is an important part when evaluating events. Since, by having candidate events that are not in the manually annotated data used for the evaluation, these will count as errors. However, these candidate events may contain relevant and true information, allowing the information contained in the annotated data to be completed or enriched.

6.3 Conclusion and perspectives

The evaluation of the event trigger detection shows that the fact that some trigger types have low support is related to the poor performance of the model to detect them correctly. A possible solution to deal with trigger types that present very low support and performance is to integrate them into subcategories that present similar semantic types. However, we can also notice that the correct detection of the different trigger types is not always directly related to the number of samples of the trigger type contained in the data. Analyzing the gold data, it was observed that

some samples were annotated in a complex way (e.g., the same trigger word annotated with a different type), which could be confusing for the model and make generalization difficult. When comparing our model with two baseline models (one based on CNNs and the other on transformer models) it was observed that our model presented slightly better performance. However, this difference in F1-score is not highly significant between the models. The use case of event detection allowed a qualitative and quantitative analysis of the model applied in a biomedical text. Here it was found that our fine-tuned model in the CG corpus presented the best performance, detecting a majority of the gold data triggers. In addition, it has identified some false positives that can help enrich the manually annotated data. On the other hand, the fine-tuned model in the seven biomedical sub-domains presented the poorest performance.

In the case of the evaluation of the identification of arguments, it was observed that there is a relationship between the number of samples per role type in the data and the performance of the model for their identification. However, when analyzing the gold data, samples with complex annotations were also observed (e.g., the same argument playing two different roles concerning the same trigger), which can cause confusion in the training of the model. When our model was compared with the two baseline models, our model enriched with the KG embeddings obtained the best performance. However, the use case where the model is applied in a biomedical text, showed that our KG-enriched model presented the poorest performance. Our model without KG embeddings fine-tuned in the seven biomedical subdomains presented the best performance together with the baseline model.

Chapter 7

Exploratory study: extracting biomedical events from PubMed abstracts

Contents

Overview

This chapter describes the use case of the proposed model to extract biomedical events from text in the biomolecular domain. The text consists of a set of PubMed abstracts selected by a group of domain experts. These abstracts were subjected to a simple manual selection of keywords and phrases by the experts, which were then used to compare with the automatic extraction of events. We used three different models for event trigger detection and argument identification for comparison, (1) SciBERT-*LSTM*, (2) SciBERT-$KG_{tr,r,ar}$ and (3) DeepEventMine. The construction of the final events (the last step in event extraction) was done using prompt-based ChatGPT (GPT-3.5). Model (3) follows an end-to-end approach based on transformer models. We describe the details of these experiments in the following sections.

7.1 Can we discover knowledge about biomolecules with event extraction?

In the context of biomolecular research, events encapsulate intricate interactions between genes, proteins, chemicals, and biological processes. These relationships are often crucial to understanding cellular mechanisms and disease pathways. Biomolecular literature represents a large source of information. However, extracting practical insights from it requires extensive and in-depth searches. The efficient extraction of biomolecular knowledge from the literature has immense potential to accelerate research in various domains. For example, the discovery of drugs, the comprehension of the disease mechanisms, the formulation of hypotheses, the design of experiments, and the validation of findings [Trieu et al., 2020].

EE techniques make it possible to capture complex relationships that might otherwise remain submerged within the text. By automating the knowledge discovery process, EE enables the detection of event triggers (eg., protein-protein interactions) and their associated events (eg., participating proteins and their functions). This process transforms unstructured textual data into structured knowledge, allowing a more systematic and interpretable understanding of biomolecular events.

In this chapter, we propose the extraction of biomedical events from PubMed abstracts with a biomolecular context. Figure 7.1 shows the pipeline we followed for this purpose. The biomolecular text corresponds to a set of six PubMed abstracts chosen by domain experts. Then, the experts were asked to manually select the keywords or phrases that they considered important for the understanding of the main contributions of the abstracts.

For the automatic extraction of events, we conducted a comparative analysis using three models: (1) SciBERT-*Bi-LSTM*, (2) SciBERT-$KG_{tr,r,ar}$ and (3) DeepEventMine (baseline). Models (1) and (2) are used for detecting event triggers and identifying arguments. The details about these models are described in Chapters 4 and 5. Then, for the last step of EE, which corresponds to the final construction of events, we proposed to use the generative language model, ChatGPT (GPT-3.5). Here, ChatGPT (GPT-3.5) allows to unmerging the relations extracted from the identification of arguments into events. For this purpose, we learned ChatGPT (GPT-3.5) to construct events from the relations extracted using a one-shot training based on prompts. The objective is that ChatGPT (GPT-3.5) seeks to generate contextually relevant event descriptions from the relations. This step allows to improve the integrity and comprehensibility of the elements extracted before.

In the case of the model (3), DeepEventMine, it adopts an end-to-end strategy based on the transformer model, SciBERT. The model is simultaneously fine-tuned across the three EE stages: event detection, argument identification and event construction.

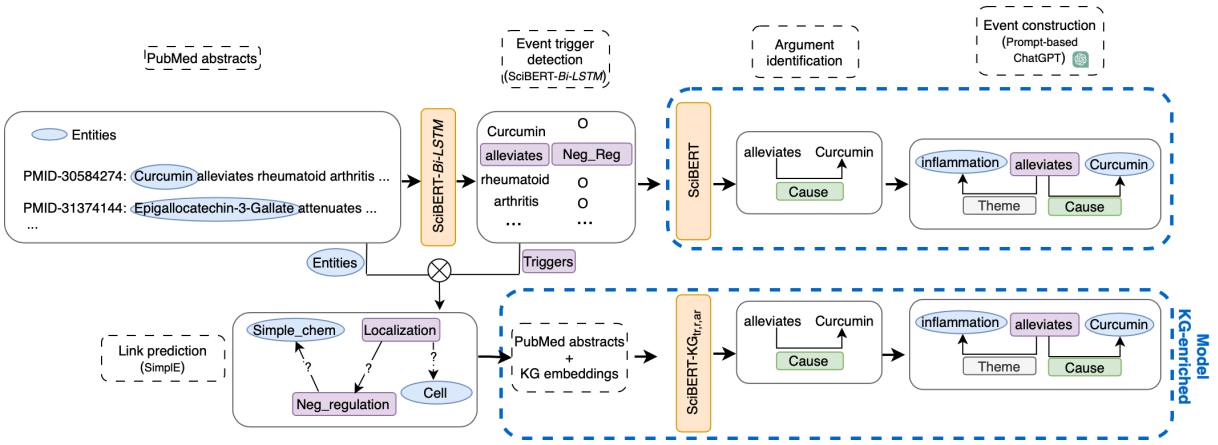


Figure 7.1: Approach followed for EE from PubMed abstracts.

The results and perspectives of this chapter include the visualization and comparison of the extracted events using the different models. This visualization provides a qualitative assessment of the models, enabling a deeper understanding of their strengths and limitations. The events extracted by each model offer insights into the relationships between biomolecular entities, actions, and contextual information. The main purpose of this comparison is to evaluate the models that we proposed and to identify trends, discrepancies, and areas of improvement for each approach in the extraction of biomedical events.

7.2 Experiments settings

7.2.1 Manual selection of PubMed abstracts

The six PubMed abstracts used for the development of these experiments were selected by a multidisciplinary group of four persons in the domain of biomolecules. The set of abstracts is listed below;

- `PMID-30584274`[29]
- `PMID-31374144`[30]
- `PMID-15707690`[31]

[29] "Curcumin alleviates rheumatoid arthritis-induced inflammation and synovial hyperplasia by targeting mTOR pathway in rats", pubmed.ncbi.nlm.nih.gov/30584274/

[30] "Epigallocatechin-3-Gallate Attenuates Microglial Inflammation and Neurotoxicity by Suppressing the Activation of Canonical and Noncanonical Inflammasome via TLR4/NF-kB Pathway", pubmed.ncbi.nlm.nih.gov/31374144/

[31] "Biocatalytic preparation of acylated derivatives of flavonoid glycosides enhances their antioxidant and antimicrobial activity", pubmed.ncbi.nlm.nih.gov/15707690/

PMID-22610280[32]

PMID-22884864[33]

PMID-26986801[34]

After the selection of abstracts, the experts were asked separately to make a selection of the keywords or phrases that describe the most relevant information in the texts. This selection was made by people with no experience in text annotation, but with experience in the biomedical domain. The selection of words and phrases was made according to the information that they considered important in agreement with their knowledge of the domain. Then, the experts made a second selection of words and phrases from what was previously chosen, where all of them agreed. This final selection is shown in Figure C.1, Figure C.13, Figure C.5, Figure C.17, Figure 7.2, Figure C.9.

7.2.2 Trigger detection and argument identification based on the proposed models

First, unlike usual EE-shared tasks, biomedical entities are not given in our context. Hence extracting these entities is our first step. This task can be done simultaneously with the detection of event triggers, although generally the texts are already annotated with the biomedical entities for the EE-shared tasks. Since our set of PubMed abstracts did not have the biomedical entities previously annotated, we made an automatic extraction of biomedical entities using DeepEventMine[35]. We used the pre-trained DeepEventMine model for the CG task.

After the automatic extraction of biomedical entities, we used SciBERT-*Bi-LSTM* fine-tuned in the CG corpus during 30 epochs to detect event triggers. This model presented the best performance of all the models in the experiments of Chapter 4.

Once the triggers have been detected, we used them together with the biomedical entities to build a KG. The information in the KG nodes corresponds to the semantic type of the triggers and the entities. The links between the triggers and the arguments are inferred through link prediction, using the pre-trained model obtained with SimplE, previously described in Chapter 5, trained on the seven corpora from Table 5.1. The KG model was trained during 500 epochs, with a hidden size of 768. Through the link prediction task, we obtained the KG embeddings that were later integrated into the transformer model to identify arguments.

[32] "Peroxiredoxin family proteins are key initiators of post-ischemic inflammation in the brain", pubmed.ncbi.nlm.nih.gov/22610280/

[33] "How to boost antioxidants by lipophilization?", pubmed.ncbi.nlm.nih.gov/22884864/

[34] "Encapsulation of Antioxidant Gallate Derivatives in Biocompatible Poly(ϵ-caprolactone)-b-Pluronic-b-Poly(ϵ-caprolactone) Micelles", pubmed.ncbi.nlm.nih.gov/26986801/

[35] github.com/aistairc/DeepEventMine

For the identification of arguments, we first predicted the roles between triggers and arguments using the SciBERT-*Bi-LSTM* model. Then we used the SciBERT-$KG_{tr,r,ar}$ model, where the transformer model is enriched with the KG embeddings of the h, r and t, for comparison. The details of both models are described in Chapter 5. After this step, it is obtained a list of relationships between triggers and arguments, which represents the candidates that can be used for the final construction of events. Here, the objective is to select the candidates that better describe the context of the text and to use them for constructing the final events.

7.2.3 Event construction based on prompts

For the last step of the extraction of biomedical events, we constructed the events using the generative model, ChatGPT (GPT-3.5)[36] based on prompts. A prompt refers to a template generated manually or automatically with instructions expressed in natural language to learn a model. This template contains one or more examples for training and a test instance as input, and the model generates the output for the test instance directly, without updating its parameters [Webson and Pavlick, 2021, Liu et al., 2023].

We chose this approach following the project "Tackling biomedical event extraction with prompts and deep learning models" that we developed together with the two master's students, Hajar Hajji and Nadir Birem (École Nationale Supérieure des Mines de Nancy, 2022-2023). The objective of this project was to identify the potential of prompts to develop each stage of biomedical EE, i.e., entities and triggers detection, argument identification, and events construction. For this, the biomedical entities, triggers, and arguments were detected simultaneously using a prompt. The generative model used, *Cohere*[37], was trained based on a few-shot approach. The results were compared with those obtained using ChatGPT (GPT-3.5), as proposed in [Gao et al., 2023], showing that this last model allowed to have results that were closer to the gold data.

Following this perspective, we tend to identify the most relevant events and use them to construct the final events. For this, we trained ChatGPT (GPT-3.5) based on a one-shot training, where the model learns how to build an event. In this training, the model is asked to construct the events using the candidate events that better describe the context of the sentence. Henceforth, the training examples contain the sentence, candidate events, and the list of biomedical entities and triggers (following the standoff format[38]), as shown below in Section 7.2.3.1.

[36] chat.openai.com
[37] cohere.com
[38] brat.nlplab.org/standoff.html

7.2.3.1 Example for one-shot training

```
The objective is to construct biomedical events.  I have the triggers,
the arguments (biomedical named entities and other triggers), and
the candidate events (relations between triggers and arguments that
can be used to construct an event).  Here is an example of the events
constructed from a sentence.

Sentence:  « Tumors of the retinal pigment epithelium metastasize to
inguinal lymph nodes and spleen in tyrosinase - related protein 1 / SV40
T antigen transgenic mice .  »

Entities and triggers:
T1 Cancer 0 6 Tumors
T2 Multi-tissue_structure 65 76 lymph nodes
T3 Organ 81 87 spleen
T57 Metastasis 41 52 metastasize

Event candidates:
R1 ToLoc Arg1:T57 Arg2:T2
R2 ToLoc Arg1:T57 Arg2:T3
R3 Theme Arg1:T57 Arg2:T1

Events constructed:
E1 Metastasis:T57 Theme:T1 ToLoc:T2
```

After the training based on this example, the model is asked to construct the events from the candidate events predicted before. The prompt used for constructing events is shown below in Section 7.2.3.2.

7.2.3.2 Prompt

```
Select the event candidates that can be used to construct biomedical
events. Consider the context of the sentence and avoid redundant or
contradictory events.

Sentence: « »

Entities and triggers:

Event candidates:

What are the events constructed from these event candidates?
```

This step is done at the sentence level. All the entities, triggers, and event candidates corresponding to a sentence are given in the prompt, together with the sentence. These elements need to follow the format presented in the example used for the training to make sure that the model makes the prediction based on what it learned.

7.3 Results and Discussion

This section presents the results obtained for the EE task applied to the PubMed abstract `PMID: 30584274`. The results obtained for the other abstracts selected by the experts are presented in Appendix C. The two first sub-tasks of event extraction, event trigger detection, and argument identification, were obtained using three different models, SciBERT-*Bi-LSTM*, SciBERT-KG$_{tr,r,ar}$ and DeepEventMine. While the results of the third sub-task, event construction, were obtained using ChatGPT (GPT-3.5). Figure 7.2 shows the manual annotation of keywords and phrases done by the experts. For this annotation, the experts were asked to select all the words and phrases that they considered pertinent, according to their domain knowledge. They have not been given any rules for the selection of this information and can choose any words or phrases they consider relevant. As can be observed, they have selected words that can correspond to biomedical entities, e.g., "curcumin". Also, they have selected phrases that can correspond to multi-word entities, e.g., "rheumatoid arthritis", or complete events, e.g., "effect of curcumin to alleviate inflammation".

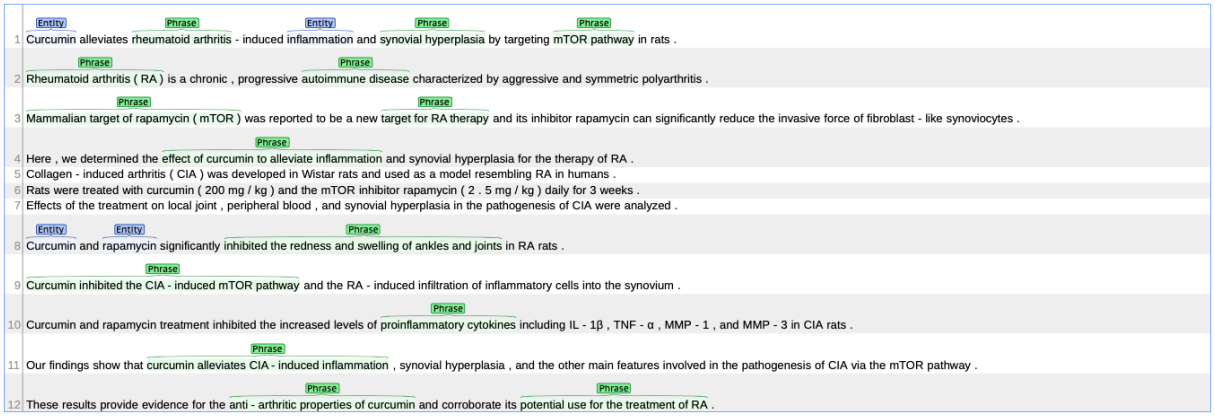


Figure 7.2: Manual annotation of the abstract `PMID: 30584274`.

149 On the other hand, Figure 7.3 shows the biomedical entities automatically extracted using
150 DeepEventMine. These entities were used later for the identification of arguments. It can
151 be observed that some of the information that was manually annotated was recognized as a
152 biomedical entity by DeepEventMine, e.g., "curcumin", "inflammation", "mTOR" and "rapamycin".
153 However, some important biomedical entities were not recognized, such as "rheumatoid arthritis",
154 "RA", "redness", "swelling", "ankles", "synovial hyperplasia", "proinflammatory cytokines" and
155 "anti-arthritic".

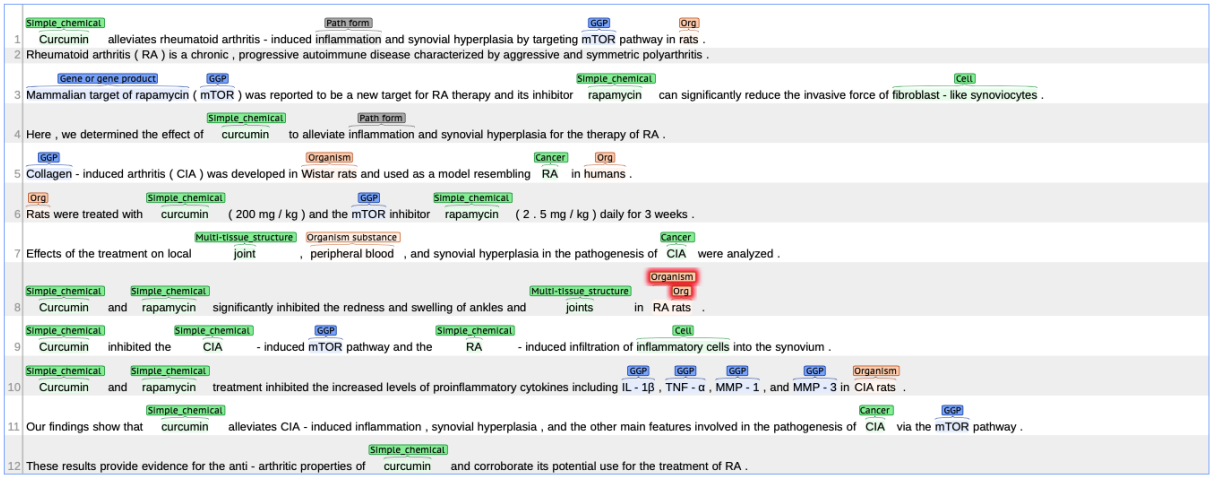


Figure 7.3: Automatic annotation of biomedical entities with DeepEventMine (`PMID: 30584274`).

156 The next step was the detection of trigger events. In Figure 7.4 are shown the results using
157 DeepEventMine. All the detected triggers are verbs conjugated in various tenses (present in
158 singular, e.g., "alleviates"; present participle; e.g., "targeting") or nouns ("effect", "pathway"). It

can also be observed that in comparison with the manually selected information, the detected triggers allow to construct events that cover most of this information in the text, e.g., “alleviates”, “pathway”, “therapy”, “effect”, “alleviate”, “inhibited” and “induced”.

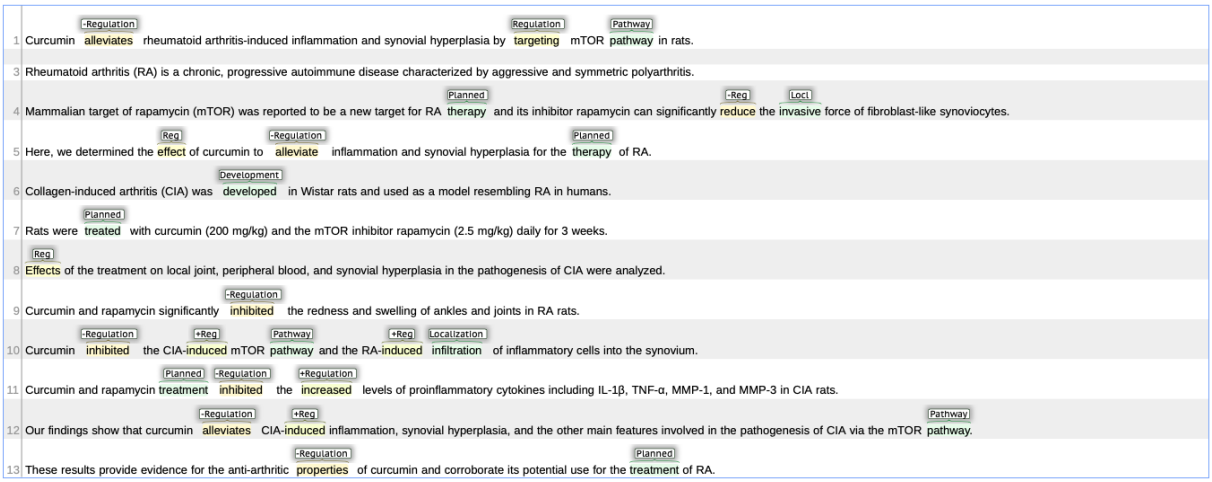


Figure 7.4: Detection of biomedical event triggers with DeepEventMine(PMID: 30584274).

We show in Figure 7.5 the results obtained using SciBERT-*Bi-LSTM*. We observed that the results are very similar to those obtained with DeepEventMine, where most of the detected triggers are the same in both models. As shown in Chapter 6, both models presented very similar performance when both were fine-tuned with the CG corpus. One reason is that the two models are based on SciBERT, although DeepEventMine uses a linear classifier and SciBERT-*Bi-LSTM* a Bi-LSTM classifier.

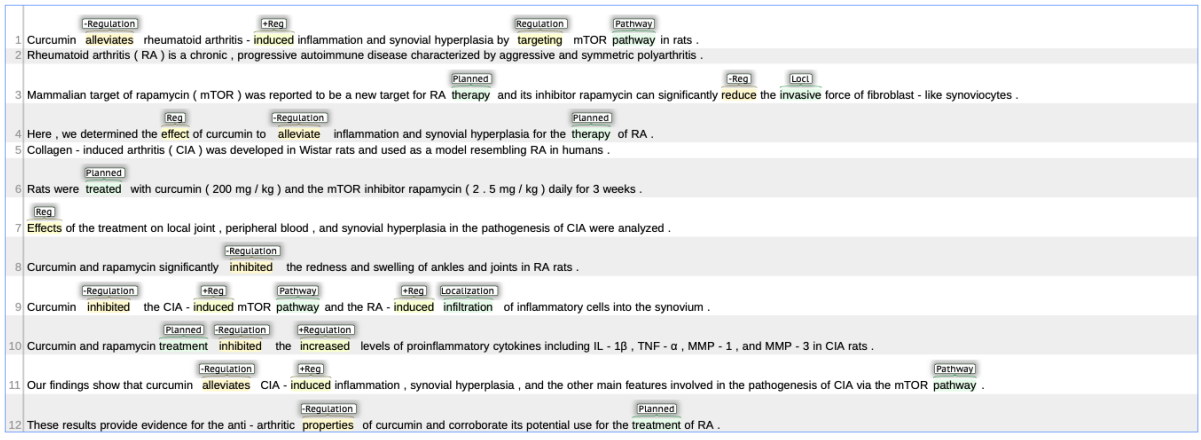


Figure 7.5: Detection of biomedical event triggers with SciBERT-*Bi-LSTM*(`PMID: 30584274`).

The next step was the identification of arguments. We show in Figure 7.6 the results using DeepEventMine. It is observed that the relations identified between the triggers and the arguments are candidate events that allow to detect some of the information manually selected. However, some missing relations could cause to have incomplete events. For example, in sentence 3, the missing relation excludes information on "mTOR" reported as a therapy target for rheumatoid arthritis (RA). In other cases, the candidate events are incomplete due to the missing biomedical entities that do not allow to have a relation. For example, in sentence 8, the missing entities "redness", "swelling", and "ankles" cause to exclude the information about what is inhibited by "curcumin" and "rapamycin". This is an error inherited from the first stage of the process, which will affect the results of all the models since the recognition of the biomedical entities was always done using DeepEventMine.

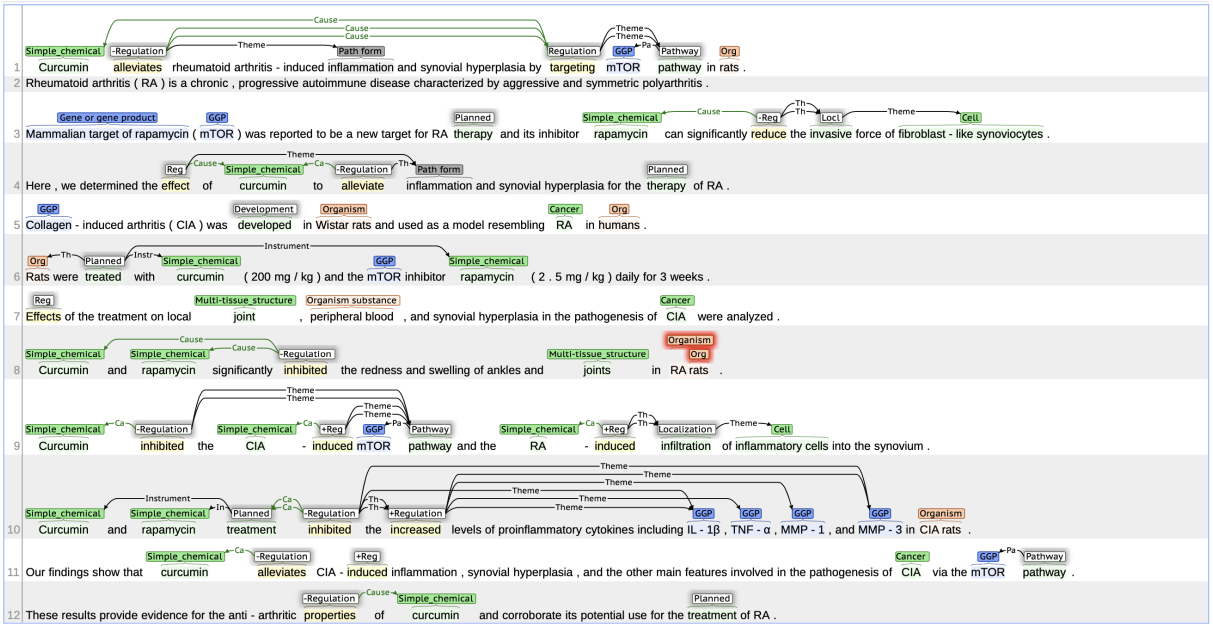


Figure 7.6: Identification of biomedical arguments with DeepEventMine (`PMID: 30584274`).

Figure 7.7 shows the results obtained using SciBERT-*Bi-LSTM*. As previously observed in Chapter 6, this model has identified more arguments than DeepEventMine. This can cause a greater number of false positives when constructing the final events, but it can also help to reduce false negatives. Besides, this might give rise to extract information that had not been manually selected but that may be important to have a more detailed description of the text. However, although this model identified more arguments than DeepEventMine, there are missing relations that should have been identified, according to manual selection. For example, as in the previous results, the relation that allows to identify "mTOR" as a target therapy for RA in sentence 3. On

the other hand, among the arguments that were identified by this model that were not previously identified are those in sentence 12. Here, the candidate event allows to know that it can be used as treatment a property of curcumin. Nevertheless, information is incomplete since "anti-arthritic" and "RA" are biomedical entities non identified.

Figure 7.7: Identification of biomedical arguments with SciBERT-*Bi-LSTM* (`PMID: 30584274`).

We show in Figure 7.8 the results obtained using SciBERT-$KG_{tr,r,ar}$. It is observed that this model obtained very similar results to those from SciBERT-*Bi-LSTM*. However, there are cases where the candidate events that were detected incompletely in the previous results have now been completely detected. For example, in sentence 3, the biomedical entity "rapamycin", labeled as a *Simple_ chemical* is identified as an argument of the trigger word "reduce", labeled as *Negative_ regulation*, playing the role of *Cause*. This candidate event allows to know that rapamycin is the cause of the reduction, together with Mammalian target of rapamycin. However, some arguments were not identified by this model but were identified by SciBERT-*LSTM* before. For example, in sentence 8 two event candidates were previously detected, presenting as arguments the biomedical entities "Curcumin" and "rapamycin", both labeled as *Simple_ chemical*. But, since this model did not detect the trigger word "inhibited", the arguments cannot be identified.

Figure 7.8: Identification of biomedical arguments with SciBERT-$KG_{tr,r,ar}$ (PMID: 30584274).

The last step for the event extraction consisted of the events construction. This step is carried out by first selecting candidate events, i.e., relations that were previously extracted for the identification of arguments, that better describe the context of the text. These candidate events are then merged for the construction of the final events. Here, the arguments that are related to the same trigger and that belong to the same event are put together. Thus, the final events can be events with a single argument (simple events) or with multiple arguments (complex events). If an event is the argument of another event, then it is called a nested event.

Figure 7.9 shows the results of event construction based on the previous triggers and arguments identified using DeepEventMine. This model performs this step simultaneously with the two other sub-tasks. It can be observed that in some of the sentences, the selected candidate events have allowed the construction of events, avoiding relationships that give repetitive information. For example, in sentences 1, 3, 4, and 9, where the repetitive relations have been filtered and the constructed events contain information that describes the text. Some of this information is part of the manually selected phrases, while another, although it is not present in the manual annotation, allows to complement the information. In the case of sentences 6 and 10, it can be noted that the constructed events presented repetitive information.

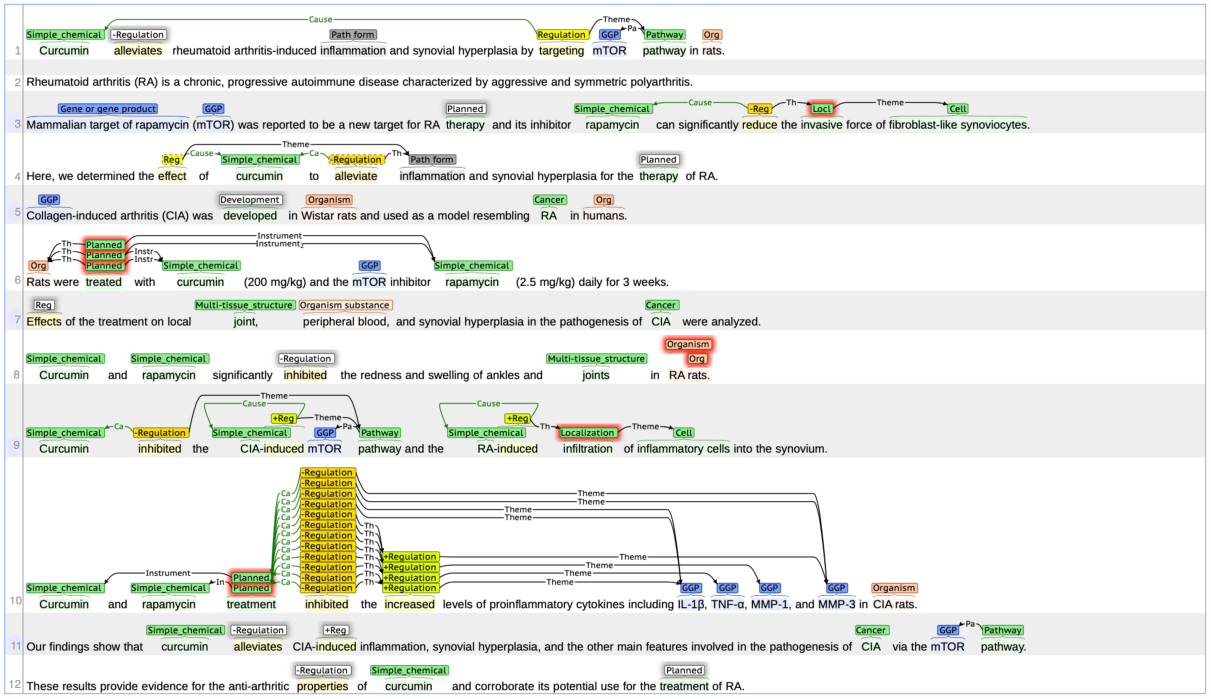


Figure 7.9: Automatic event extraction with DeepEventMine (`PMID: 30584274`).

For example, in sentence 6 there were three events of type "Planned" that give redundant information and that can be confusing as each event gives different details. The solution would be to keep only the event with the three arguments, the word entity "Rats" of type *Org* (which stands for organism) playing the role of *Theme*, the word entity "curcumin" of type *Simple_chemical* playing the role of *Instrument* and the word entity "rapamycin" of type *Simple_chemical* playing the role of $Instrument_2$. In sentence 10 occurs the same problem with the *-Regulation* (which stands for negative regulation) events, where some events present as an argument the nested event *+Regulation* (which stands for positive regulation), while others present as an argument the biomedical entities of type *GGP* (which stands for Gene or gene product). The difference between both types of events is that if the argument is the nested event, the *-Regulation* event describes that it affects the *+Regulation* event which refers to the increased levels of the *GGP* biomedical entities. This means that the *-Regulation* event refers to the inhibition of increased levels of the biomedical entities. When the *-Regulation* event has the *GGP* biomedical entities as an argument, this means that the biomedical entities are inhibited. So the information related to the increased levels that are inhibited is lost. On the other hand, it is also possible to observe that in some of the sentences, such as 8, 11, and 12, no event has been constructed even though they presented candidate events. What is observed in these cases is that the candidate events had only arguments of the *Cause* type and not of the *Theme* type. This could be causing

the candidate events to not be used to construct the final event because they may be considered incomplete.

Figure 7.10 shows the results based on the triggers and arguments identified using SciBERT-*Bi-LSTM*. The events were constructed using ChatGPT (GPT-3.5). Among the constructed events it can be noted that some arguments are incorrect. For example, in sentence 1, the event $+Reg$ (which stands for positive regulation) presented as an argument the entity "Curcumin" of type $Simple\_chemical$ playing the role of $Cause$, while the argument of type $Theme$ is the entity "inflammation" of type $pathform$ (which stands for pathological form). This event describes that curcumin is the cause that induces inflammation. However, according to the text, the cause that induces inflammation is rheumatoid arthritis and curcumin relieves it. Therefore, the information contained in the event is not adequate. A similar error occurs in sentence 9, with events of type $+Reg$. On the other hand, some other events were incompletely constructed. For example, in sentence 4, the event Reg (which stands for Regulation) has as an argument the entity of type $Pathform$ playing the role of $Theme$, but the argument that plays the role of cause is missing, which corresponds to the entity "curcumin". Likewise, in sentence 10, the event $-Regulation$ is constructed correctly but incompletely. Since, the $-Regulation$ event is missing the argument that plays the role of $Theme$, which should be the nested event $+Regulation$. In this way, the event would not only describe that the cause of the inhibition is curcumin and rapamycin, but also that what is inhibited is the increased levels of the GGP entities. Sentences 7, 11, and 12 also present incompletely constructed events.

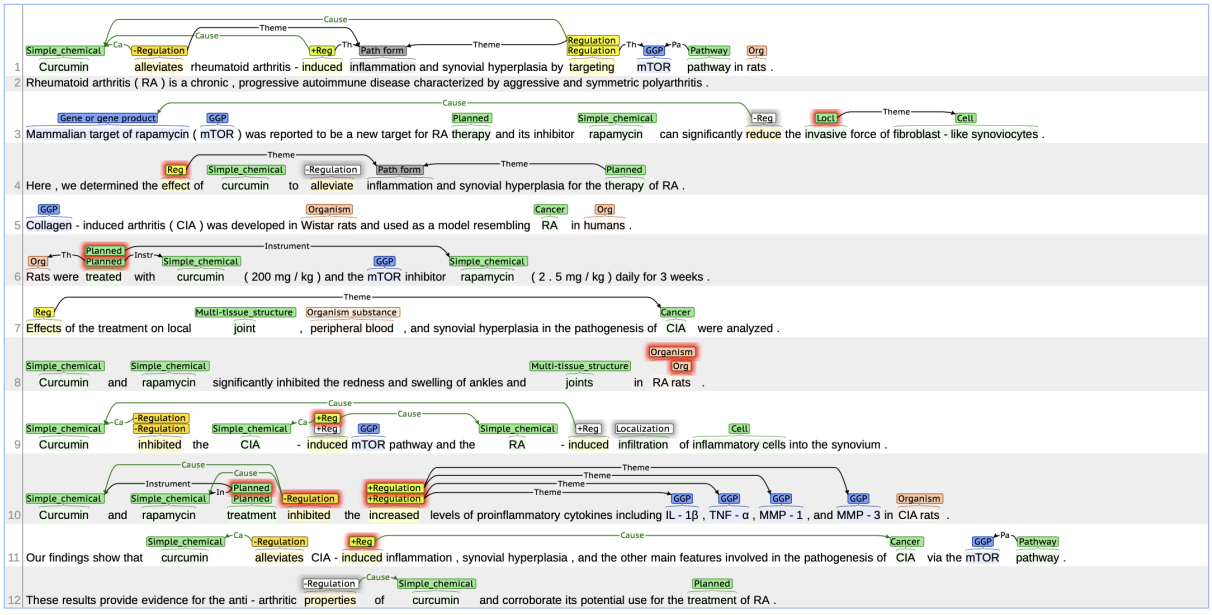


Figure 7.10: Automatic event extraction with SciBERT-*Bi-LSTM* (`PMID: 30584274`).

We show in Figure 7.11 the results based on the triggers and arguments identified using SciBERT-$KG_{tr,r,ar}$. Here, the events were also constructed using ChatGPT (GPT-3.5). In general, it can be observed that most of the constructed events describe the context of the information contained in the text. However, in some cases, the details may be incomplete due to missing biomedical entities and unidentified or incorrect arguments. In sentence 1, the constructed events allow us to know that there is relief of an inflammation that was induced and that the targeting of the mTOR pathway is done. However, the argument is missing in the *-Regulation* event that allows us to know the cause of the relief, which is the word entity "Curcumin".

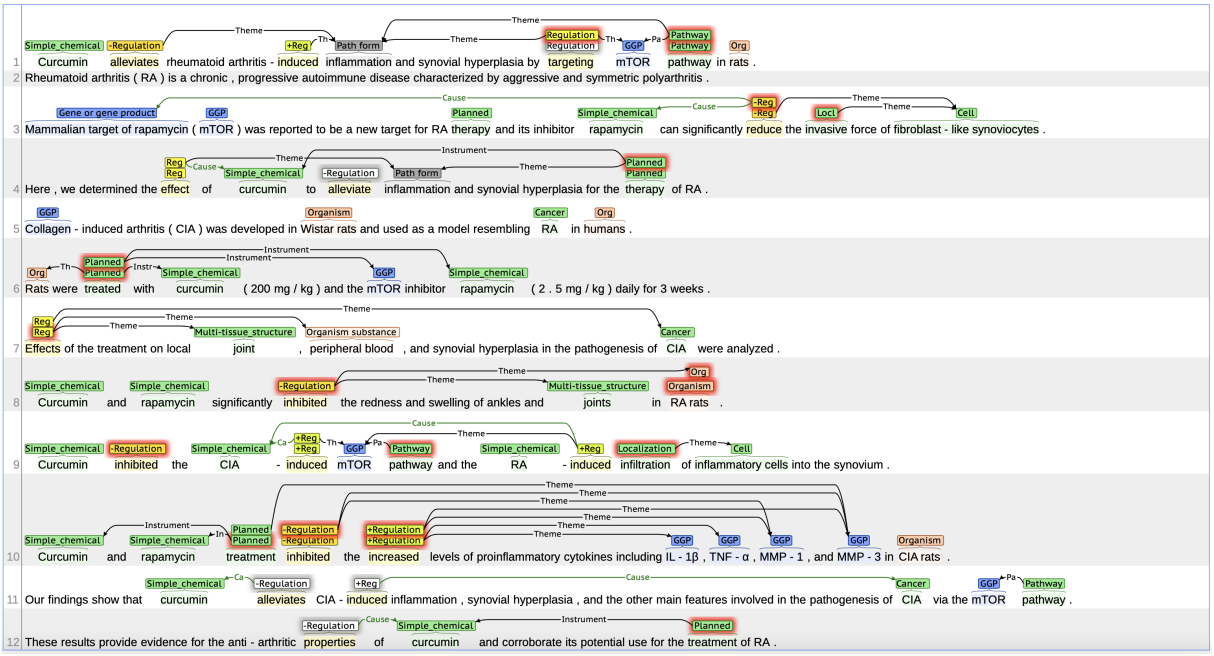


Figure 7.11: Automatic event extraction with SciBERT-$KG_{tr,r,ar}$ (ours) (`PMID: 30584274`).

On the other hand, in the *Regulation* and *Pathway* events, the *Path form* argument should not play a role concerning these triggers. In sentences 3, 4, 6, 7, 8, 9, 11, and 12, the constructed events are correct and descriptive of the text, although in some situations the events may be incomplete due to missing biomedical entities. For example, in sentence 3, the entity "RA" was not recognized, so it does not allow the construction of the *Planned* event that describes the RA therapy. In sentence 7, even if the events are correct, they do not allow to describe the main information of the sentence since there are missing entities and undetected triggers, e.g., "treatment", "synovial hyperplasia", and "analyzed". A similar problem occurs in sentence 8, where the missing entities corresponding to "redness", "swelling" and "ankles" do not allow to have the correct information about what is inhibited. Also, there are two missing arguments of the *-Regulation* event, "Curcumin" and "rapamycin", which play the role of *Cause* of the inhibition.

In sentence 12, the missing entity that corresponds to "anti-arthritic" should play the role of *Theme* in the *-Regulation* event, allowing completion of the information about the properties of curcumin. Similarly, the missing entity "RA" should play the role of *Theme* in the *Planned* event to know what the treatment is used for.

7.4 Conclusion and Perspectives

When comparing the results of the three models, it can be seen that the SciBERT-*Bi-LSTM* model presents the worst overall results among the three. This is highlighted by being the model that presented more events constructed with erroneous arguments, therefore giving incorrect information about the text. In the case of DeepEventMine and SciBERT-$KG_{tr,r,ar}$, both models presented events that allowed the text to be described, however, in both cases, there were incomplete events. Most of these incomplete events were due to missing biomedical entities or roles, so the final result did not depend on the event construction stage, but on the previous stages. SciBERT-*LSTM* and SciBERT-$KG_{tr,r,ar}$ are models with a pipeline approach, where each subtask is done separately, following a sequence. Therefore, errors from one subtask are going to be passed to subsequent sub-tasks in this sequence. In the case of DeepEventMine, which is a model that follows a joint approach, this problem should be reduced since all sub-tasks are carried out simultaneously. Therefore, the errors of the different sub-tasks allow for an optimization of the general performance and thus reduce errors. However, despite the different implementations of both approaches, pipeline, and joint, the problem of incomplete events is present. On the other hand, when comparing SciBERT-*LSTM* and SciBERT-$KG_{tr,r,ar}$, the events built in SciBERT-$KG_{tr,r,ar}$ show a significant improvement by having fewer erroneous arguments than SciBERT-*LSTM*. So integrating KG embeddings can, in effect, enrich the context of the model and thus improve performance.

One of the points that can be highlighted when building events with ChatGPT (GPT-3.5) is that, although most of the time they present correct information, there are arguments that are missing, even if they had previously been identified in the previous subtask. So the model is probably not considering all the information necessary to construct the events. It must be taken into account that this is a generative language model that has been trained on a huge amount of data, on general tasks, and in multiple domains. The construction of biomedical events is a very specific task in a domain that presents a particular vocabulary, which can become a complex task for the model. Besides, since we used one-shot training, the model had to learn the construction of events based on a single example. On the other hand, since the prompt used to train ChatGPT (GPT-3.5) was built manually, this can induce errors depending on how the model has "understood" it. To resolve this, an optimization of the prompt could be used to improve the information given to the model so that it learns the task.

Conclusions and Perspectives

Conclusions

In this thesis, we have tackled biomedical event extraction, an information extraction task that can be divided into three primary sub-tasks: event trigger detection, argument identification, and event construction. Chapters 1, 2, and 3 provide a comprehensive literature review, while Chapters 4, 5, 6, and 7 describe the contributions made.

To address the challenge of biomedical event extraction, we proposed a method employing a pipeline approach that relies on transformer language models and KGs. Transformer language models stand as the prevailing architecture for large-scale language modeling. Within the biomedical domain, BERT and its variants have found extensive application across diverse information extraction tasks. While these models share a common architecture, they present different characteristics in their pre-training methodologies and the data used for this process. Consequently, determining the optimal model for a specific task, such as event extraction, can be a challenge. In addition, considering that the biomedical domain has very specific terms and vocabulary, models are usually fine-tuned for specific biomedical subdomains. Therefore, they cannot be used with different biomedical data expecting to have the same performance even if it is the same task.

In this context, in Chapter 4 we propose a comparative analysis of BERT and four of its biomedical variants, aiming to identify the model that shows the best performance in biomedical event trigger detection. The models were fine-tuned using seven corpora from different biomedical subdomains together. With this approach, we analyze and evaluate the generalization of the models for different biomedical subdomains. Additionally, we developed an evaluation to assess the impact of incorporating lexical and syntactic information to improve biomedical event detection. From these analyses, our results revealed that fine-tuning SciBERT using a Bi-LSTM classifier emerges as the most effective strategy for detecting biomedical events across diverse biomedical subdomains. **With this result we provide the first contribution of this work, suggesting that a model pre-trained on biomedical and general data that starts its pre-training from scratch is the best strategy for biomedical event trigger detection**. Our results also

demonstrate that a training duration of 10 to 30 epochs is sufficient for optimal model learning of this sub-task. Training the model for more than 30 epochs leads to a slight improvement in performance, though not as substantial as the gains achieved in earlier epochs. Furthermore, we showed that models performance does not change significantly when additional syntactic or lexical features are included. **With this result we provide the second contribution of the thesis, suggesting that transformer models acquire the necessary linguistic information during their training**. Therefore, the syntactic and lexical features added probably give the model information that it already had previously integrated during its learning. This is advantageous because it eliminates the need for the automatic incorporation of features using external tools, preventing the introduction of potential sources of errors.

Transformer models present the advantage of not only incorporating syntactic information into their embeddings but also integrating semantic information. Despite this embedded knowledge, finding the appropriate relationships for the identification of biomedical arguments remains a challenging aspect, often leading to errors. In contrast, KGs integrate semantic information into their embeddings through interconnected graphs or semantic networks, rather than relying on textual data. These graphs consist of interlinked descriptions of entities, each represented by nodes, relationships, and labels. By representing biomedical events in the form of graphs, it is possible to obtain the KG embeddings through a link prediction task. This information can be further used to augment the semantic understanding of a transformer model. In Chapter 5, we propose incorporating KG embeddings into the transformer model through various strategies. The goal is to improve the second sub-task of biomedical event extraction, which is argument identification. First, we evaluated the different KG models in the link prediction task, where we compared the use of semantic types and entity instances in the nodes of the KG. This evaluation revealed that the KG model, SimplE, outperformed the other KG models in terms of MRR and Hits@z. This suggests that a KG model allows to find the relations between the triggers and the arguments through link prediction, especially when the KG nodes represent the semantic information of the entities. After having identified the KG model that presented the best performance in link prediction, the KG embeddings calculated were integrated into the transformer model. Then, the model was fine-tuned to identify biomedical arguments. **With these results we provide the third contribution of this thesis, showing that enriching SciBERT with KG embeddings, particularly those corresponding to the roles of arguments in events, improves the identification of biomedical arguments**. This demonstrates the fundamental role of semantic context in identifying relationships between the triggers and the arguments, and therefore, determining the roles they play in events. Finally, **for the fourth contribution of this thesis we found that when evaluating the performance of the transformer models using the corpora of the different subdomains together, the F1-score is negatively affected**. This was to be expected since the imbalance of the data is more pronounced, considering

that some labels are unique in each corpus while others overlap between the different corpora. This makes the sparsity of the data more important and therefore, it is more difficult for the model to generalize.

To validate the proposed framework, we analyzed and evaluated the knowledge that is embedded by the transformer language models and the KG models in Chapter 6. This evaluation consisted of a category grained analysis and a use case to validate the event trigger detection and argument identification in a biomedical text. Also, we compared our model with two baseline models (one based on CNN and the other based on transformer models) that presented the state-of-the-art performance in the CG task. The evaluation of event trigger detection revealed that the detection of triggers does not depend solely on the number of samples of trigger types in the corpus. Complex annotations in some samples, such as the same trigger word annotated with a different semantic type, can create confusion in the model and make generalization difficult. Compared to the baseline models, the performance is very similar in all of them. In the context of event detection in a biomedical abstract, our model showed the highest performance when it was fine-tuned on the CG task, successfully detecting most of the gold data triggers. Additionally, it identified some false positives that can be used to complement the manually annotated data. In contrast, the model fine-tuned on the seven biomedical subdomains showed the lowest performance. Regarding the evaluation of the identification of arguments, the number of samples per role type showed a correlation with the performance of the model. However, similar to event trigger detection, some samples presented complex annotations, such as the same argument with two different roles in the same event, which can lead to confusion in training. Compared with the two baseline models, our model enriched with the KG embeddings achieved the highest performance. However, when we used this model to identify arguments in the biomedical abstract, it presented the lowest performance. Interestingly, our model without KG embeddings, fine-tuned on all seven biomedical subdomains, showed the best performance along with the baseline model. This suggests that one of the limitations in the performance of the models is due to the gold data. The annotations of these corpora, since it is done manually, may be subject to human error, representing a source of errors for the fine-tuning of the models. On the other hand, incorporating the KG embeddings may lead to a decline in performance due to the introduction of inaccurately predicted links, which represent another source of errors. Therefore, the accumulation of errors in the gold data, the event triggers detected and the predicted links may be causing poorer inference than if the KG embeddings are not integrated.

Finally, in Chapter 7 we developed an exploratory study where we compared the manual selection of biomedical information with the events extracted using our model. For this, a multi-disciplinary group of experts has made a manual selection of keywords and phrases from a set of Pubmed abstracts. Afterward, we detected biomedical event triggers and identified arguments from these abstracts following our proposed pipeline method. When comparing the

performance of our model with and without the integration of the KG embeddings, the events generated by the KG-enriched model showed an improvement. This underlines the effectiveness of integrating KG embeddings to enrich the contextual understanding of the model and consequently improve performance.

The final events were constructed from the detected triggers and the identified arguments using a prompts approach with the generative language model, ChatGPT. We compared the results with a baseline model. In terms of performance, both the baseline model and our KG-enriched model, while proficient in the event description in the text, tend to generate incomplete events. The main cause of this problem lies in the absence of crucial biomedical entities or roles, which emphasizes that the final result depends also on the preceding sub-tasks and not only on the construction of the event. While the baseline model, which follows a joint approach, can mitigate error propagation, it may encounter difficulties in handling complex interactions between sub-tasks. On the other hand, our pipeline approach offers a more interpretable and modular workflow, allowing for fine-tuning and refinement of each sub-task independently. Nevertheless, the results using ChatGPT suggest also a potential limitation, showing that the model may not be considering all the relevant information required for the construction of the final event. It is critical to recognize that ChatGPT is a generative language model trained on a wide range of tasks and domains. Constructing biomedical events requires specialized knowledge and vocabulary, potentially representing a challenge for the model. Additionally, manually constructing the training prompt introduces the possibility of errors depending on the interpretation of the model.

Perspectives

The findings of this thesis offer different clues for future research directions in biomedical event extraction. Several key points deserve attention to improve the performance and robustness of our proposed approach.

The first of these points is the **adoption of joint models**. Future research could focus on the development and evaluation of joint models designed for the extraction of biomedical events. Given the observed advantages of a joint approach over a traditional pipeline model, further exploration into the implementation of joint models is recommended. By simultaneously addressing multiple sub-tasks, joint models have demonstrated the potential to mitigate error propagation and optimize overall performance. Another point is the **mitigation of data imbalance**. Addressing this is crucial for achieving reliable and consistent performance in event extraction. Strategies such as oversampling minority classes, undersampling majority classes, or employing techniques like SMOTE (Synthetic Minority Over-sampling Technique) could be strategies to investigate. Additionally, exploring techniques like data augmentation or leveraging transfer learning from related tasks can contribute to a more balanced and representative dataset. On the other hand, to

deepen the contextual understanding of models, the **KGs could be enriched**. The incorporation of external knowledge sources, such as additional KGs, knowledge bases, or specialized thesauri within the biomedical domain, can be potential directions. Augmenting the existing KG with domain-specific resources can provide models with a richer semantic understanding, likely leading to improved event extraction performance. Finally, considering the limitations of using ChatGPT, we should explore **alternative approaches for the construction of events**. Investigating models specifically designed for event construction or integrating the event construction step directly into the model architecture can be considered. This approach could potentially reduce reliance on external language models and provide a more tailored solution for the task at hand.

In summary, our vision for future work in biomedical event extraction lies in the exploration and integration of advanced modeling techniques, data balancing strategies, knowledge enrichment, and innovative approaches to constructing events. By addressing these key areas, we believe it is possible to contribute to the continued advancement and refinement of event extraction systems in the biomedical domain.

Publications

National conference

Zanella Laura and Toussaint Yannick. “Fine-tuning Pre-trained Transformer Language Models for Biomedical Event Trigger Detection.”, Workshop on Deep Learning for Natural Language Processing, 22ème édition de la conférence Extraction et Gestion des Connaissances (EGC’22). Blois, France, 2022. hal.science/hal-03984783/document

International conferences

Zanella Laura and Toussaint Yannick. “How Much do Knowledge Graphs Impact Transformer Models for Extracting Biomedical Events?”, The 22nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks, The 61st Annual Meeting of the Association for Computational Linguistics (ACL’23). Toronto, Canada, 2023. hal.science/hal-04192212/document

Zanella Laura and Toussaint Yannick. “Adding Linguistic Information to Transformer Models Improves Biomedical Event Detection?”, 1st Symposium on Challenges for Natural Language Processing, 18th Conference on Computer Science and Intelligence Systems (FedCSIS’23). Warsaw, Poland 2023. hal.science/hal-04197974/document

Appendix A

Supplementary materials

A.1 Corpora

Here are described the seven biomedical corpora used for the development of the experiments, Cancer Genetics (CG) 2013 [39], Epigenetics and Post-translational Modifications (EPI) 2011 [40], GENIA 2011 (GE11)[41], GENIA 2013 [42], Infectious Diseases (ID) 2011 [43], Pathway Curation (PC)[44] and Multi-Level Event Extraction (MLEE) [45].

The CG corpus was released for the BioNLP Shared Task 2013. It was focused on the automatic extraction of information containing statements of biological processes related to the development and progression of cancer. Figure A.1 shows a sentence with events present in the CG corpus.

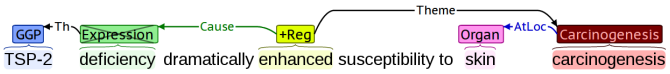


Figure A.1: Sample of events in the CG corpus. Figure taken from [39].

The EPI corpus was released for the BioNLP Shared Task 2013. The main objective was to extract events related to epigenetic changes, such as DNA methylation histone modification, and post-translational protein modifications. Figure A.2 shows a sentence with events present in the EPI corpus.

[39] 2013.bionlp-st.org/tasks/cancer-genetics-cg-task
[40] 2011.bionlp-st.org/bionlp-shared-task-2011/epigenetics-and-post-translational-modifications-task-epi
[41] sites.google.com/site/bionlpst/bionlp-shared-task-2011/genia-event-extraction-genia
[42] site not available
[43] 2011.bionlp-st.org/bionlp-shared-task-2011/infectious-diseases-task-id
[44] 2013.bionlp-st.org/tasks/pathway-curation-pc-task
[45] nactem.ac.uk/MLEE/

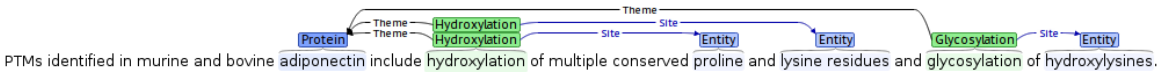


Figure A.2: Sample of events in the EPI corpus. Figure taken from [40].

The GE11 corpus was released for the BioNLP Shared Task 2011. The task has the same definition as the BioNLP Shared Task 2009, with the main purpose of measuring the progress of the community on this task. The objective was to extract events related to transcription factors in human blood cells.

The GE13 corpus was released for the BioNLP Shared Task 2013. This task had the same purpose as GENIA 2011, including an annotation scheme extended and new full-text articles.

The ID corpus was released for the BioNLP Shared Task 2011. It was mainly focused on the molecular mechanisms of infectious diseases. The task has the same definition as the BioNLP'09 Shared Task, including new entity and event categories.

The PC corpus was released for the BioNLP Shared Task 2011. The objective of the task was to develop event extraction systems to support the curation, evaluation, and maintenance of biomolecular pathway models. Figure A.3 shows a sentence with events present in the PC corpus.

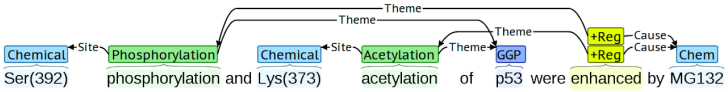


Figure A.3: Sample of events in the PC corpus. Figure taken from [44].

The MLEE corpus was developed to detect entities and extract events at different levels of biological organization, from the molecular to the organ system level. Figure A.4 shows a sentence with events present in the MLEE corpus.

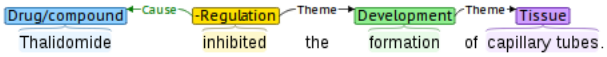


Figure A.4: Sample of events in the MLEE corpus. Figure taken from [45].

A.2 Entity, trigger and role types used for the experiments

“**trigger_types**": [
“Acetylation",
“Activation",
“Amino_acid_catabolism",
“Protein_catabolism",
“Binding",
“Blood_vessel_development",
“Breakdown",
“Carcinogenesis",
“Catabolism",
“Catalysis",
“Cell_death",
“Cell_differentiation",
“Cell_division",
“Cell_proliferation",
“Cell_transformation",
“Complex",
“Conversion",
“DNA_demethylation",
“DNA_methylation",
“Deacetylation",
“Death",
“Deglycosylation",
“Degradation",
“Dehydroxylation",
“Demethylation",
“Dephosphorylation",
“Deubiquitination",
“Development",
“Dissociation",
“Gene_expression",
“Glycolysis",
“Glycosylation",
“Growth",
“Hydroxylation",
“Inactivation",
“Infection",
“Localization",
“Metabolism",
“Metastasis",
“Methylation",
“Mutation",
“Negative_regulation",
“Pathway",
“Phosphorylation",
“Planned_process",
“Positive_regulation",
“Process",
“Protein_catabolism",
“Protein_modification",
“Protein_processing",
“Regulation",
“Regulon-operon",
“Remodeling",
“Reproduction",
“Synthesis",
“Transcription",
“Translation",
“Transport",
“Ubiquitination"
]

"**entity_types**": [
"Amino_acid",
"Anatomical_system",
"Cancer",
"Cell",
"Cellular_component",
"Chemical",
"DNA_domain_or_region",
"Developing_anatomical_structure",
"Drug_or_compound",
"Entity",
"Gene_or_gene_product",
"Immaterial_anatomical_entity",
"Multi-tissue_structure",
"Organ",
"Organism",
"Organism_subdivision",
"Organism_substance",
"Pathological_formation",
"Protein",
"Protein_domain_or_region",
"Simple_chemical",
"Tissue",
"Two-component-system"
],

"**role_types**": [
"AtLoc",
"Cause",
"Contextgene",
"FromLoc",
"Instrument",
"Participant",
"Product",
"Site",
"Sidechain",
"Theme",
"ToLoc"
]

A.3 Data statistics

Figure A.5 presents a graph with the number of samples per entity type contained in the seven corpora merged, where is possible to observe the data imbalance. The entity type with the largest number of samples is "Protein", followed by "Gene_or_gene_product".

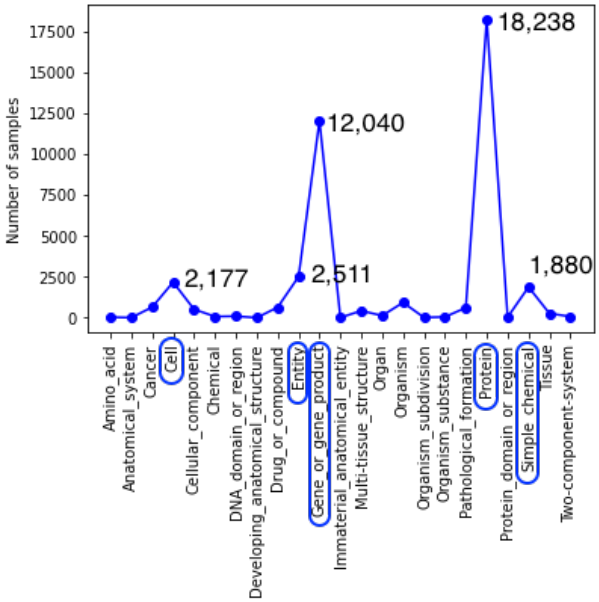


Figure A.5: Graph of the number of samples per entity type.

Figure A.6 presents a graph with the number of samples per role type. The most representative type is "Theme", while the second is "Cause". The rest of the types present much fewer samples and the amount is more balanced.

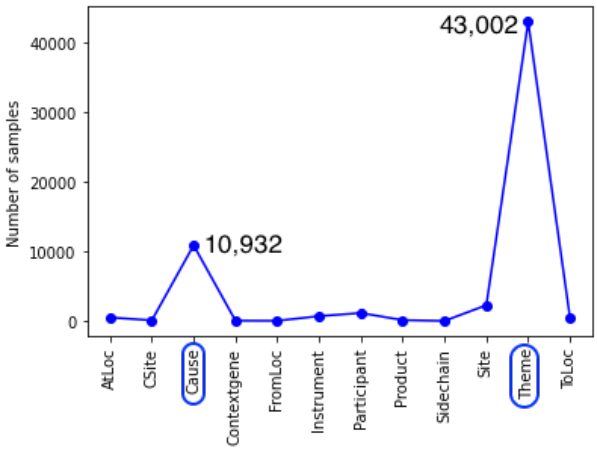


Figure A.6: Graph of the number of samples per role type.

Figure A.7 presents a graph with the number of samples per trigger type. The most representative type is "Positive_regulation", followed by "Gene_expression", "Negative_regulation", "Regulation" and "Binding".

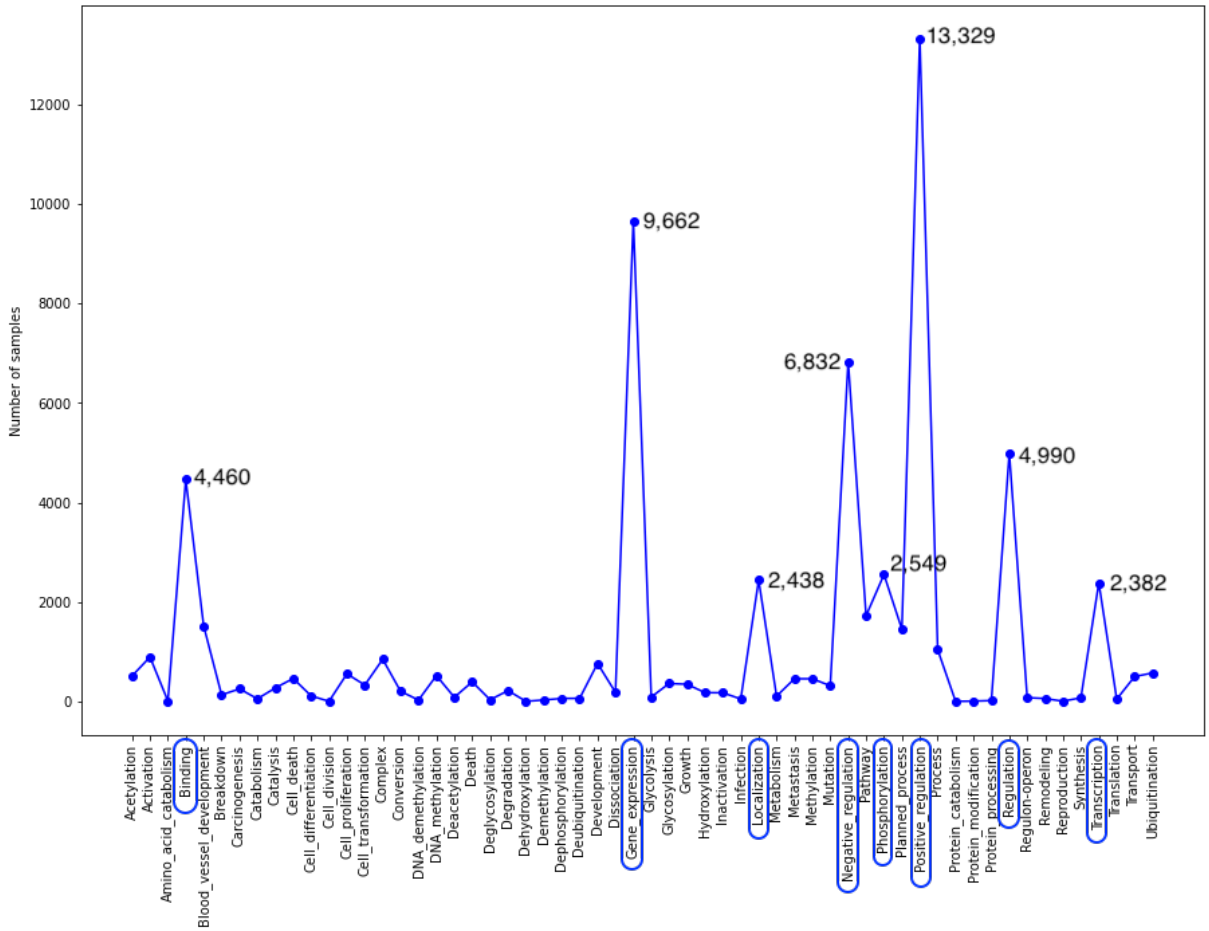


Figure A.7: Graph of the number of samples per trigger type.

Table A.1 and Table A.2 present the statistics of the different corpora.

Table A.1: Statistics of the corpora. The Shared Tasks did not release the gold data of the test sets.

Item	CG			EPI			GE11			GE13		
	Train	**Dev**	**Test**	**Train**	**Dev**	**Test**	**Train**	**Dev**	**Test**	**Train**	**Dev**	**Test**
# docs	300	100	200	600	200	400	908	259	347	222	249	305
# sents	3,031	1,005	1,923	5,698	1,955	4,122	8,691	2,900	3,371	2,446	2,733	3,380
# entities	11,034	3,665	6,984	8,226	2,712	5,737	12,105	4,871	5,301	3,797	4,569	4,359
# triggers	7,370	2,420	-	1,523	515	-	7,882	2,388	-	2,210	2,466	-
# relations	10,201	3,412	-	2,558	858	-	11,142	3,698	-	3,125	3,920	-
# events	8,803	2,915	5,530	1,852	601	-	10,310	3,250	-	2,817	3,199	-
# ent class	18	18	16	2	2	1	2	2	1	3	3	1
# trig class	40	37	-	15	14	-	9	9	-	11	13	-
# rel class	9	9	-	5	5	-	6	6	-	5	5	-
# flat ev	5,800	1,913	-	1,677	540	-	6,477	2,119	-	1,938	1,961	-
# nested ev	3,003	1,002	-	175	61	-	3,833	1,131	-	879	1,238	-

Table A.2: Statistics of the corpora. The Shared Tasks do not release the gold data of the test sets.

Item	PC			ID			MLEE		
	Train	**Dev**	**Test**	**Train**	**Dev**	**Test**	**Train**	**Dev**	**Test**
# docs	260	90	175	152	46	118	131	44	87
# sents	2,499	857	1,695	2,848	711	1,947	1,271	457	880
# entities	7,855	2,734	5,312	6,553	1,996	4,239	4,147	1,431	2,713
# triggers	4,603	1,617	-	1,578	577	-	2,767	932	1,855
# relations	7,751	2,705	-	1,906	715	-	3,783	1,334	2,471
# events	5,992,	2,129	-	2,088	691	-	3,296	1,175	2,206
# ent class	4	4	4	6	6	5	16	16	15
# trig class	24	21	-	1	9	-	26	23	22
# rel class	8	8	-	6	6	-	9	8	8
# flat ev	3,730	1,336	-	1,673	515	-	2,171	749	1,497
# nested ev	2,262	813	-	415	176	-	1,125	426	709

Appendix B

Predictions of roles for argument identification

B.1 Statistics of roles in the CG development set

Figure B.1 presents a graph with the number of samples per role type in the development set of the CG corpus. The most representative type is "Theme".

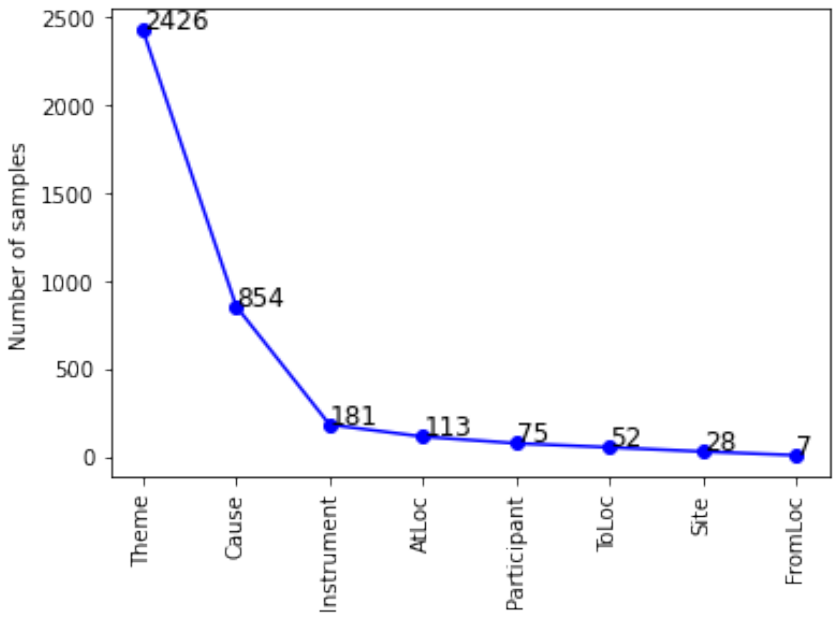


Figure B.1: Statistics of roles in the development set of the CG corpus.

B.2 Predictions of roles

Table B.2 presents the predictions of roles obtained by each model. The predictions were grouped into patterns, where **True** refers to the true role and **No. Samples** refers to the number of times each pattern was repeated in the predictions. The first pattern in the table, when the value True was *Theme* and all three models predicted *Theme*, was repeated 2046 times.

Table B.1: Prediction of roles of each model on the CG corpus (validation dataset).

True	**SciBERT-*Bi-LSTM***	**KG**	**SciBERT-$KG_{tr,r,ar}$**	**No. Samples**
Theme	Theme	Theme	Theme	2046
Theme	Theme	Cause	Theme	218
Theme	Negative	Theme	Theme	94
Theme	Cause	Theme	Theme	13
Theme	Theme	Instrument	Theme	12
Theme	Theme	AtLoc	Theme	10
Theme	Cause	Cause	Theme	8
Theme	Negative	Cause	Theme	7
Theme	AtLoc	Theme	Theme	3
Theme	Instrument	Theme	Theme	2
Theme	Negative	Instrument	Theme	2
Theme	ToLoc	Theme	Theme	2
Theme	Instrument	Cause	Instrument	1
Theme	Instrument	Instrument	Theme	1
Theme	Theme	Product	Theme	1
Theme	Theme	Site	Theme	1
Theme	Theme	Theme	Product	1
Theme	Cause	Cause	Cause	1
Theme	AtLoc	AtLoc	AtLoc	1
Theme	Negative	AtLoc	Theme	1
Theme	Theme	ToLoc	Theme	1
Cause	Cause	Cause	Cause	534
Cause	Cause	Theme	Cause	244
Cause	Negative	Cause	Cause	23
Cause	Theme	Cause	Cause	14
Cause	Theme	Theme	Cause	14

Table B.2: Prediction of roles of each model on the CG corpus (validation dataset).

True	**SciBERT-*Bi-LSTM***	**KG**	**SciBERT-$KG_{tr,r,ar}$**	**No. Samples**
Cause	Theme	Theme	Theme	11
Cause	Negative	Theme	Cause	5
Cause	Theme	Cause	Theme	3
Cause	Negative	Theme	Negative	2
Cause	Cause	Site	Cause	1
Cause	Negative	Cause	Negative	1
Cause	Negative	Cause	Participant	1
Cause	Instrument	Cause	Cause	1
Instrument	Instrument	Instrument	Instrument	135
Instrument	Instrument	Theme	Instrument	16
Instrument	Theme	Instrument	Instrument	7
Instrument	Negative	Instrument	Negative	4
Instrument	Theme	Instrument	Theme	4
Instrument	Negative	Instrument	Instrument	3
Instrument	Theme	Instrument	Negative	3
Instrument	Negative	Theme	Instrument	2
Instrument	Negative	Theme	Negative	2
Instrument	Cause	Instrument	Instrument	2
Instrument	AtLoc	Instrument	Instrument	1
Instrument	Theme	Theme	Instrument	1
Instrument	Negative	Theme	Theme	1
AtLoc	AtLoc	AtLoc	AtLoc	83
AtLoc	AtLoc	Theme	AtLoc	11
AtLoc	AtLoc	Negative	AtLoc	3
AtLoc	Negative	AtLoc	AtLoc	3
AtLoc	Negative	AtLoc	Negative	3
AtLoc	Theme	AtLoc	AtLoc	3
AtLoc	Theme	AtLoc	Negative	2
AtLoc	AtLoc	ToLoc	AtLoc	2
AtLoc	Theme	Theme	AtLoc	2
AtLoc	Theme	Theme	Negative	1

True	**SciBERT-*Bi-LSTM***	**KG**	**SciBERT-$KG_{tr,r,ar}$**	**No. Samples**
Participant	Participant	Participant	Participant	72
Participant	Negative	Participant	Participant	2
Participant	Theme	Theme	Participant	1
ToLoc	ToLoc	ToLoc	ToLoc	37
ToLoc	ToLoc	AtLoc	ToLoc	7
ToLoc	Negative	Theme	Negative	2
ToLoc	Theme	AtLoc	Theme	2
ToLoc	Theme	AtLoc	ToLoc	1
ToLoc	Theme	Theme	ToLoc	1
ToLoc	Theme	ToLoc	Negative	1
ToLoc	ToLoc	Theme	ToLoc	1
Site	Site	Site	Site	19
Site	Site	Negative	Site	3
Site	Negative	Site	Site	2
Site	Cause	Site	Site	1
Site	Negative	Negative	Site	1
Site	Theme	Site	Negative	1
Site	Theme	Site	Site	1
FromLoc	Negative	Theme	Negative	3
FromLoc	FromLoc	Theme	FromLoc	2
FromLoc	ToLoc	AtLoc	ToLoc	1
FromLoc	ToLoc	Theme	FromLoc	1

Appendix C

Examples of exploratory study

C.1 PMID: 15707690

Figure C.1 shows the abstract with the keywords and phrases manually highlighted by the experts.

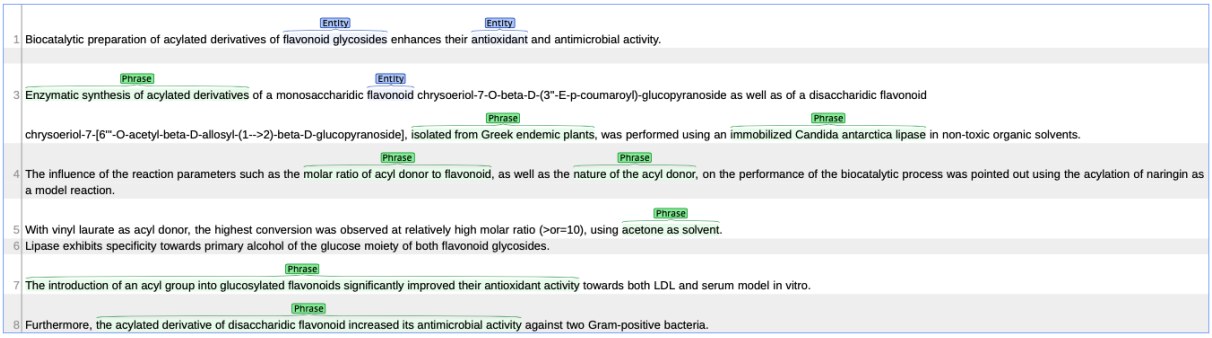


Figure C.1: Manual annotation of the abstract `PMID: 15707690`.

Figure C.2 shows the events automatically extracted from the abstract using DeepEventMine.

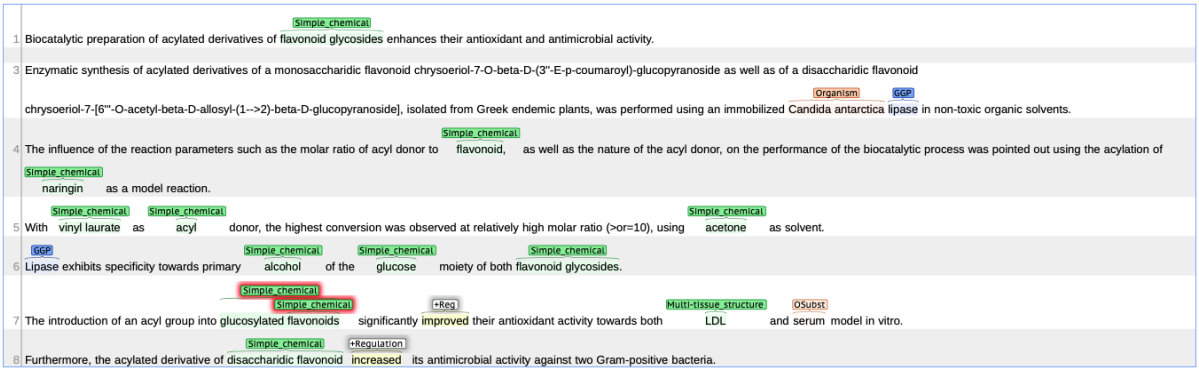


Figure C.2: Automatic event extraction with DeepEventMine (`PMID: 15707690`).

Figure C.3 shows the events automatically extracted from the abstract using SciBERT.

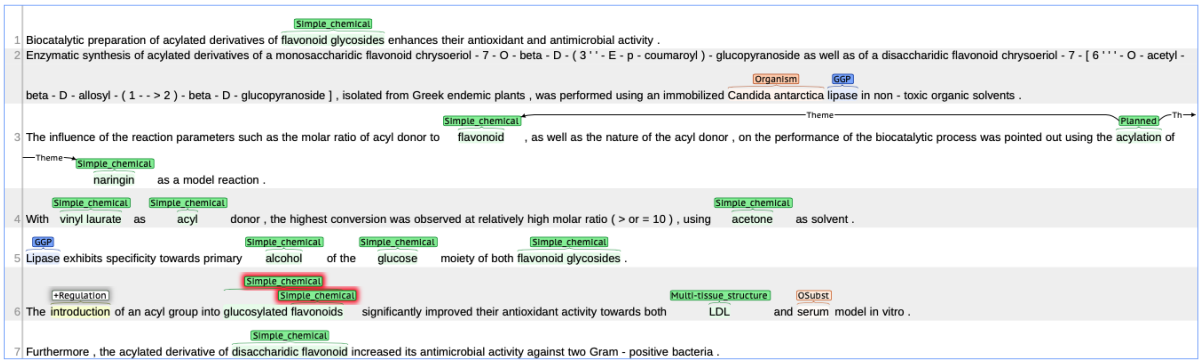


Figure C.3: Automatic event extraction with SciBERT (`PMID: 15707690`).

Figure C.4 shows the events automatically extracted from the abstract using SciBERT$-KG_{tr,r,ar}$.

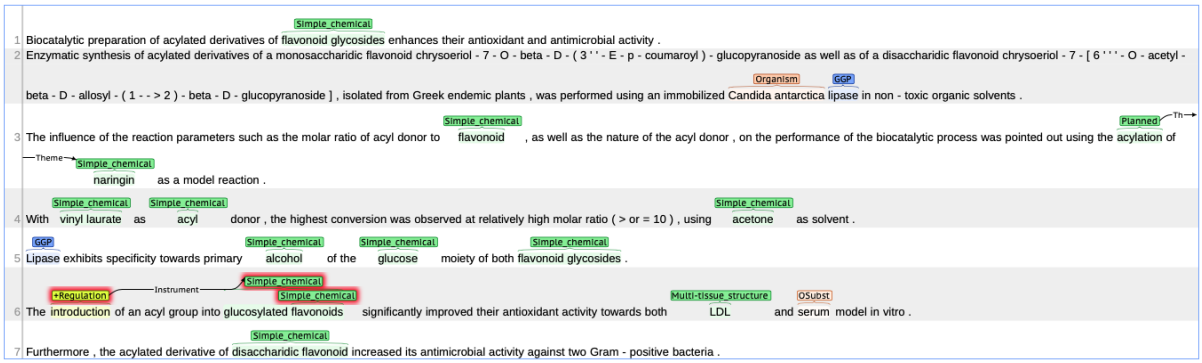


Figure C.4: Automatic event extraction with SciBERT-$KG_{tr,r,ar}$ (ours) (`PMID: 15707690`).

C.2 PMID: 22884864

Figure C.5 shows the abstract with the keywords and phrases manually highlighted by the experts.

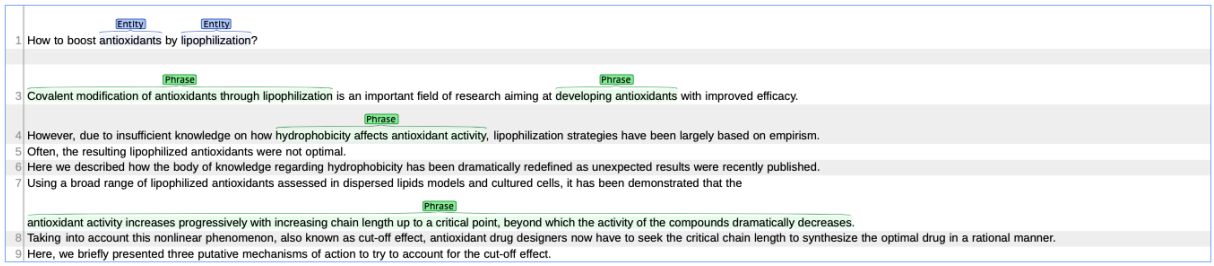


Figure C.5: Manual annotation of the abstract `PMID: 22884864`.

Figure C.6 shows the events automatically extracted from the abstract using DeepEventMine.

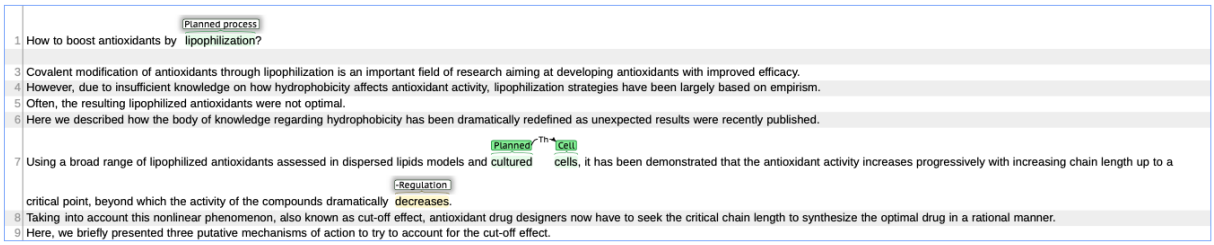


Figure C.6: Automatic event extraction with DeepEventMine (`PMID: 22884864`).

Figure C.7 shows the events automatically extracted from the abstract using SciBERT.

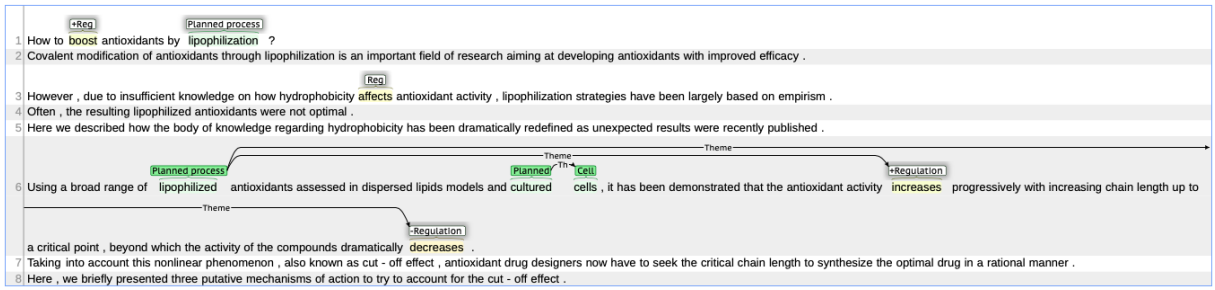


Figure C.7: Automatic event extraction with SciBERT (`PMID: 22884864`).

Figure C.8 shows the events automatically extracted from the abstract using SciBERT$-KG_{tr,r,ar}$.

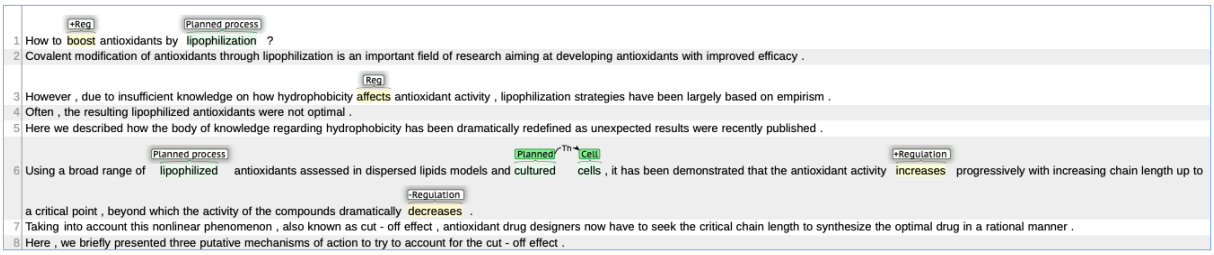


Figure C.8: Automatic event extraction with SciBERT-$KG_{tr,r,ar}$ (ours) (`PMID: 22884864`).

C.3 PMID: 31374144

Figure C.9 shows the abstract with the keywords and phrases manually highlighted by the experts.

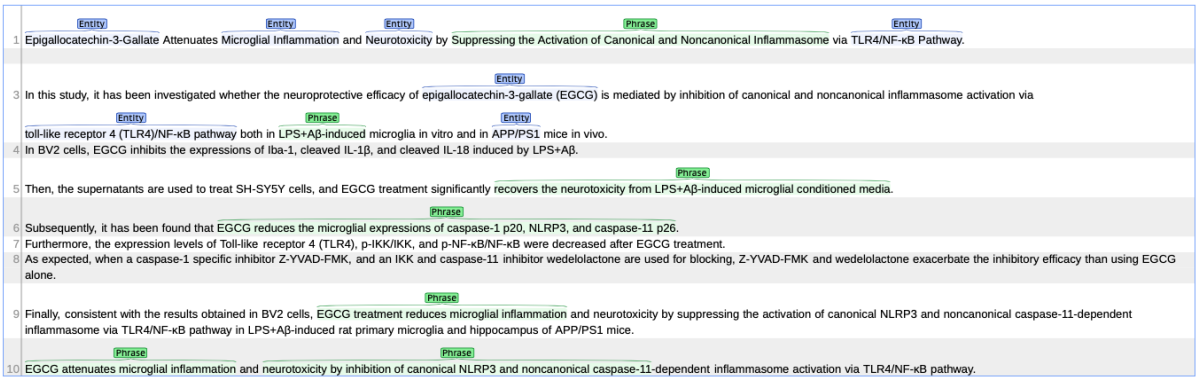


Figure C.9: Manual annotation of the abstract `PMID: 31374144`.

Figure C.10 shows the events automatically extracted from the abstract using DeepEventMine.

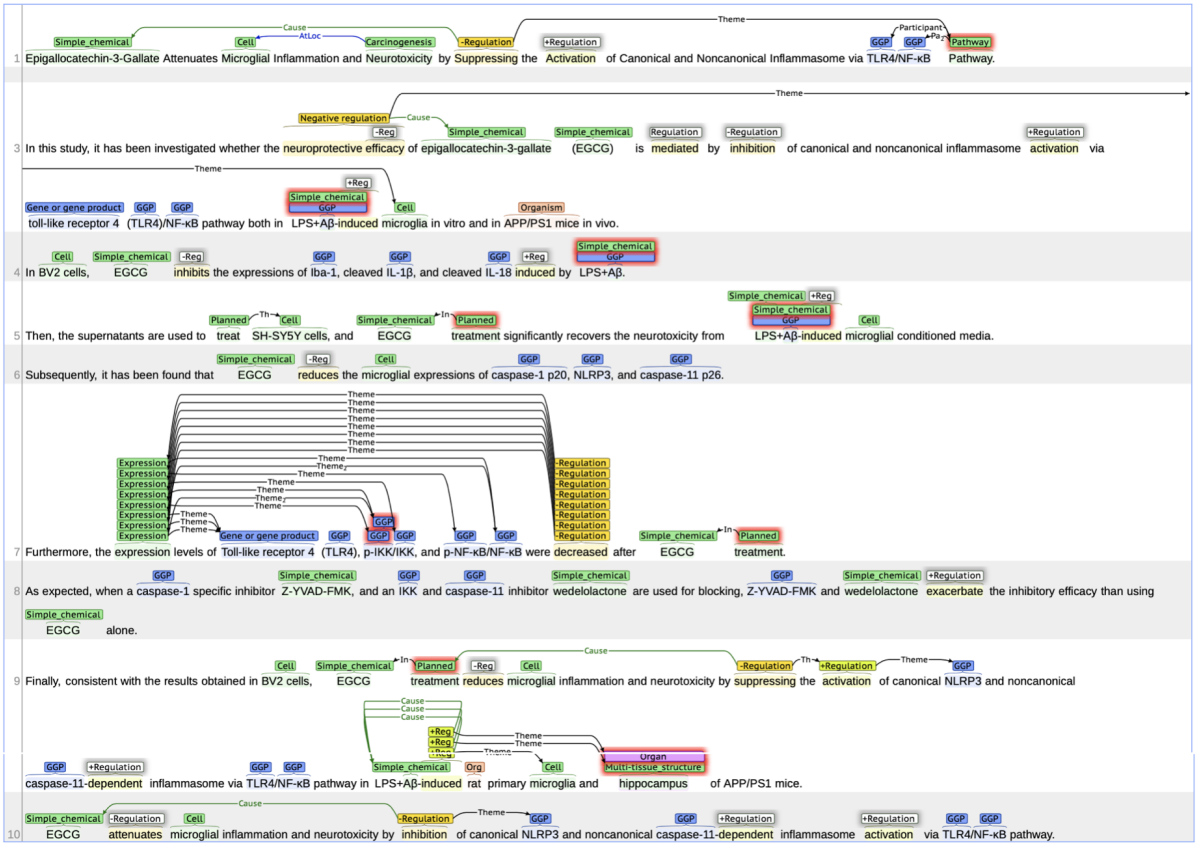


Figure C.10: Automatic event extraction with DeepEventMine (`PMID: 31374144`).

Figure C.11 shows the events automatically extracted from the abstract using SciBERT.

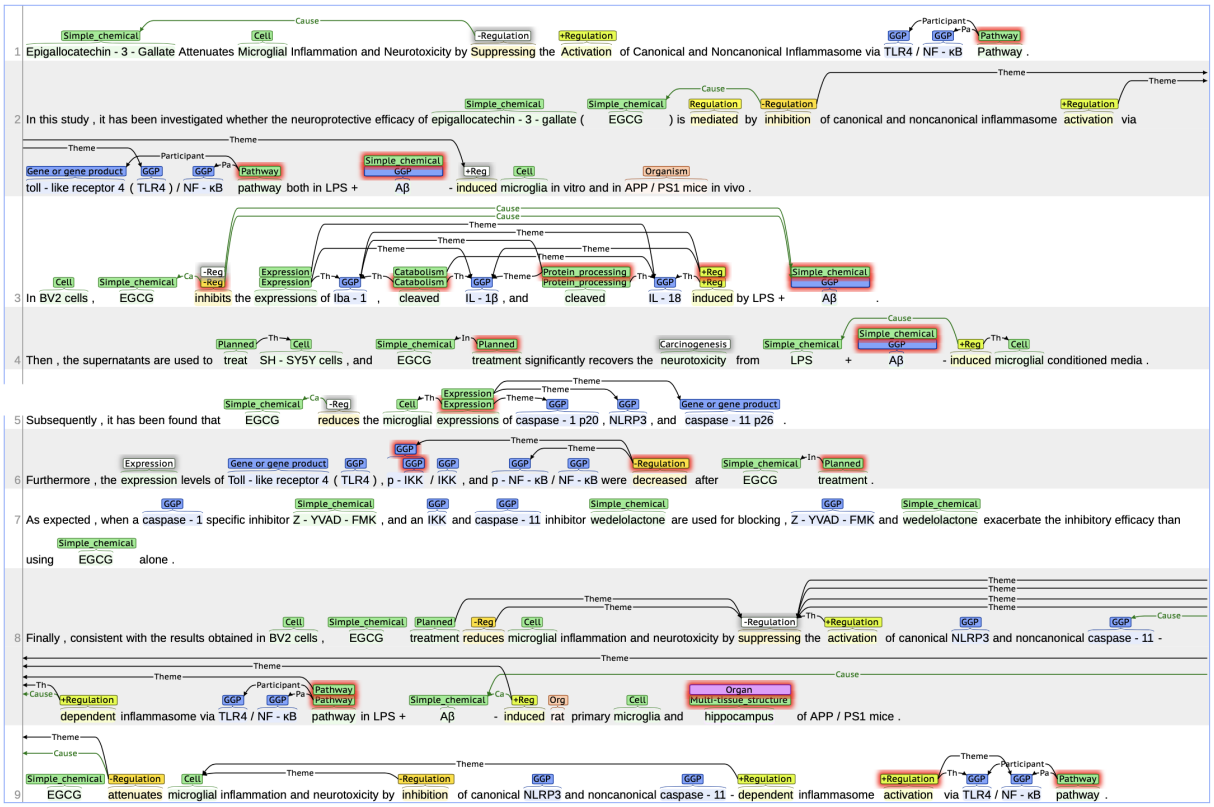


Figure C.11: Automatic event extraction with SciBERT (`PMID: 31374144`).

Figure C.12 shows the events automatically extracted from the abstract using SciBERT$-KG_{tr,r,ar}$.

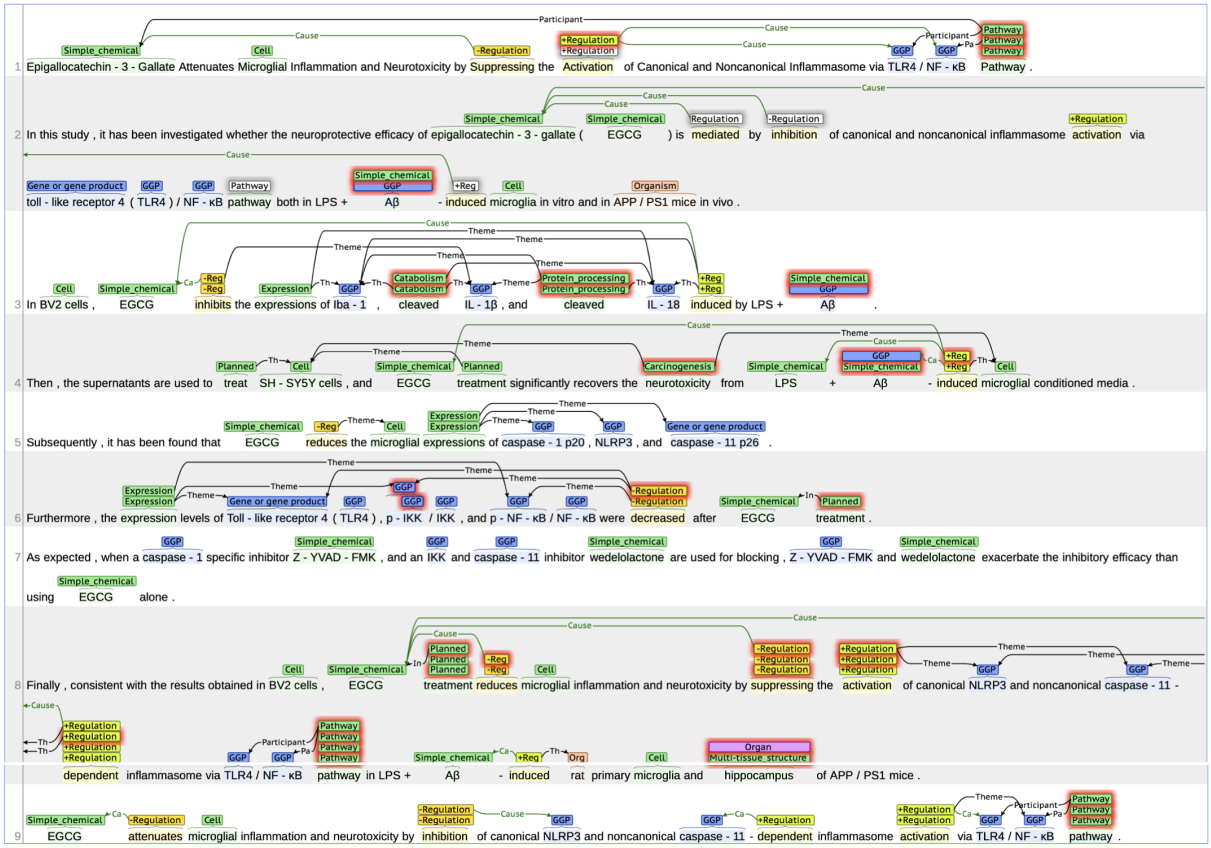


Figure C.12: Automatic event extraction with SciBERT-$KG_{tr,r,ar}$ (ours) (`PMID: 31374144`).

C.4 PMID: 22610280

Figure C.13 shows the abstract with the keywords and phrases manually highlighted by the experts.

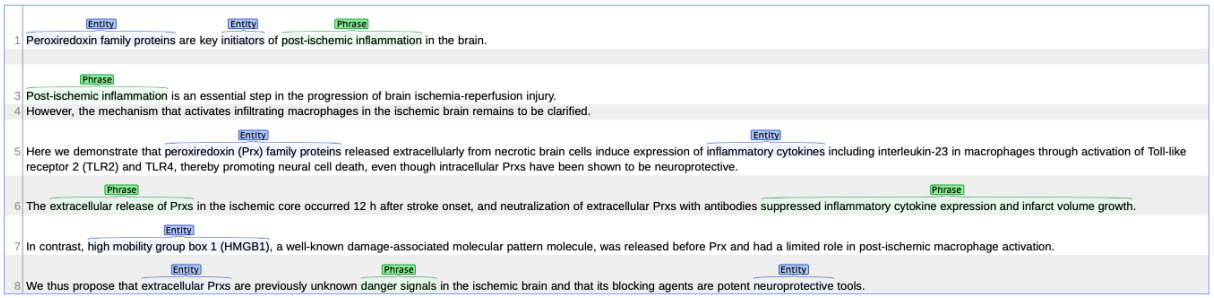


Figure C.13: Manual annotation of the abstract `PMID: 22610280`.

Figure C.14 shows the events automatically extracted from the abstract using DeepEventMine.

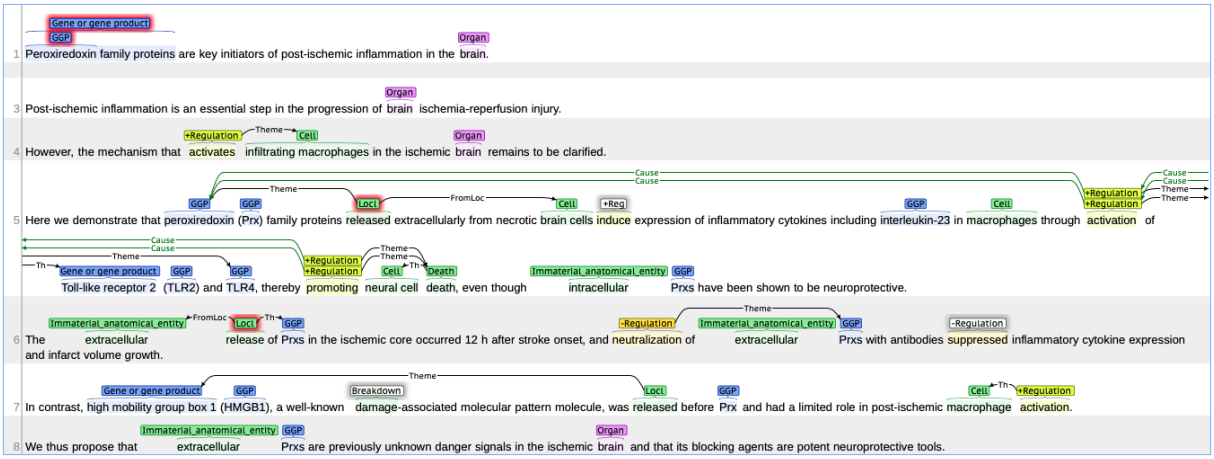


Figure C.14: Automatic event extraction with DeepEventMine (`PMID: 22610280`).

Figure C.15 shows the events automatically extracted from the abstract using SciBERT.

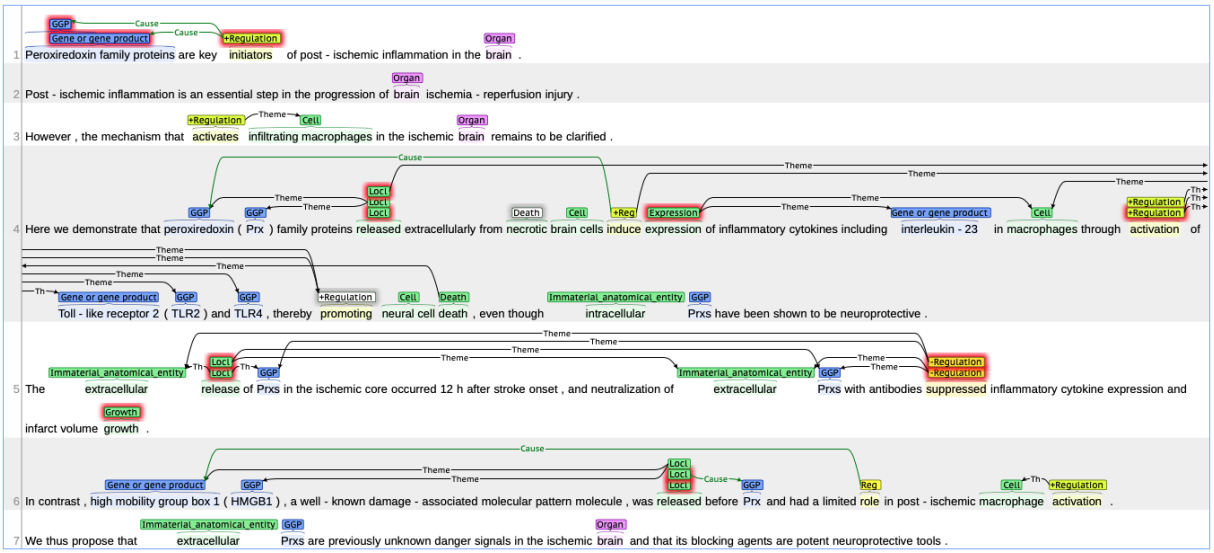


Figure C.15: Automatic event extraction with SciBERT (PMID: 22610280).

Figure C.16 shows the events automatically extracted from the abstract using SciBERT$-KG_{tr,r,ar}$.

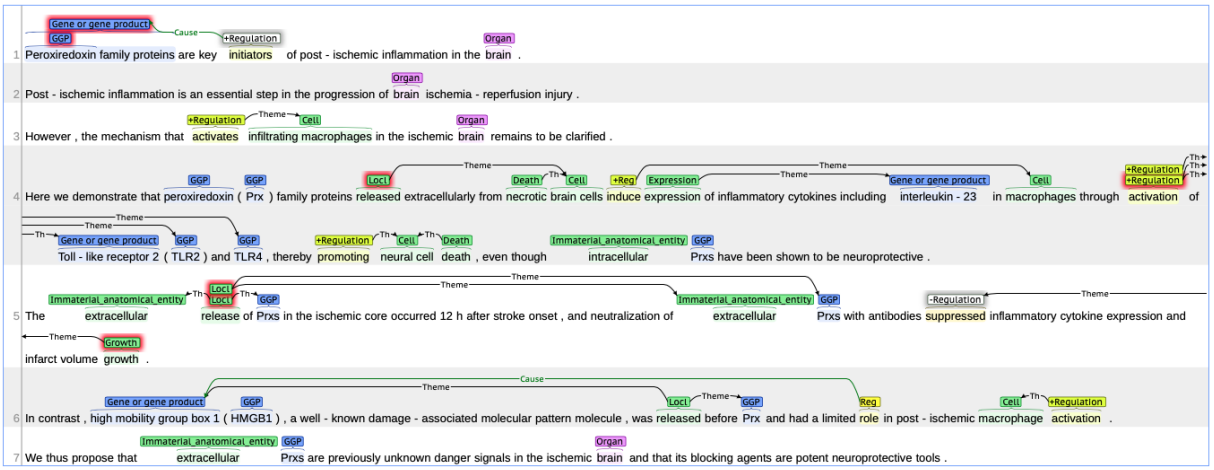


Figure C.16: Automatic event extraction with SciBERT-$KG_{tr,r,ar}$ (ours) (PMID: 22610280).

C.5 PMID: 26986801

Figure C.17 shows the abstract with the keywords and phrases manually highlighted by the experts.

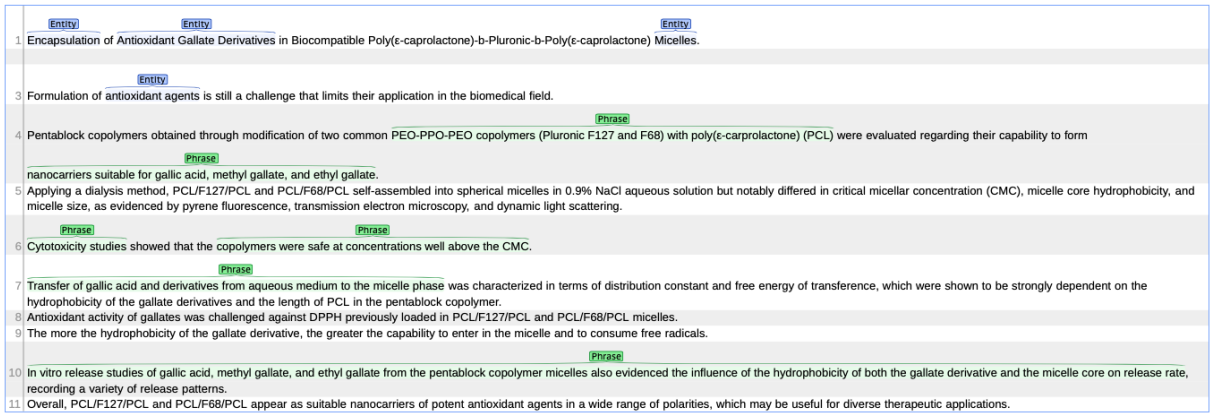


Figure C.17: Manual annotation of the abstract PMID: 26986801.

Figure C.18 shows the events automatically extracted from the abstract using DeepEventMine.

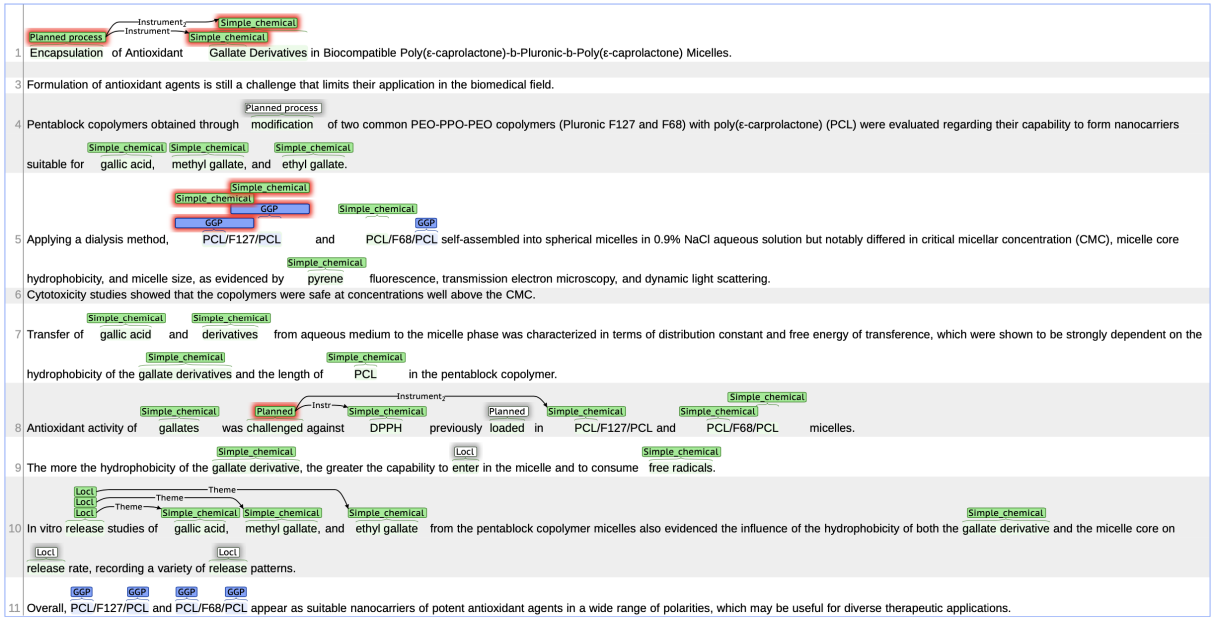


Figure C.18: Automatic event extraction with DeepEventMine (PMID: 26986801).

Figure C.19 shows the events automatically extracted from the abstract using SciBERT.

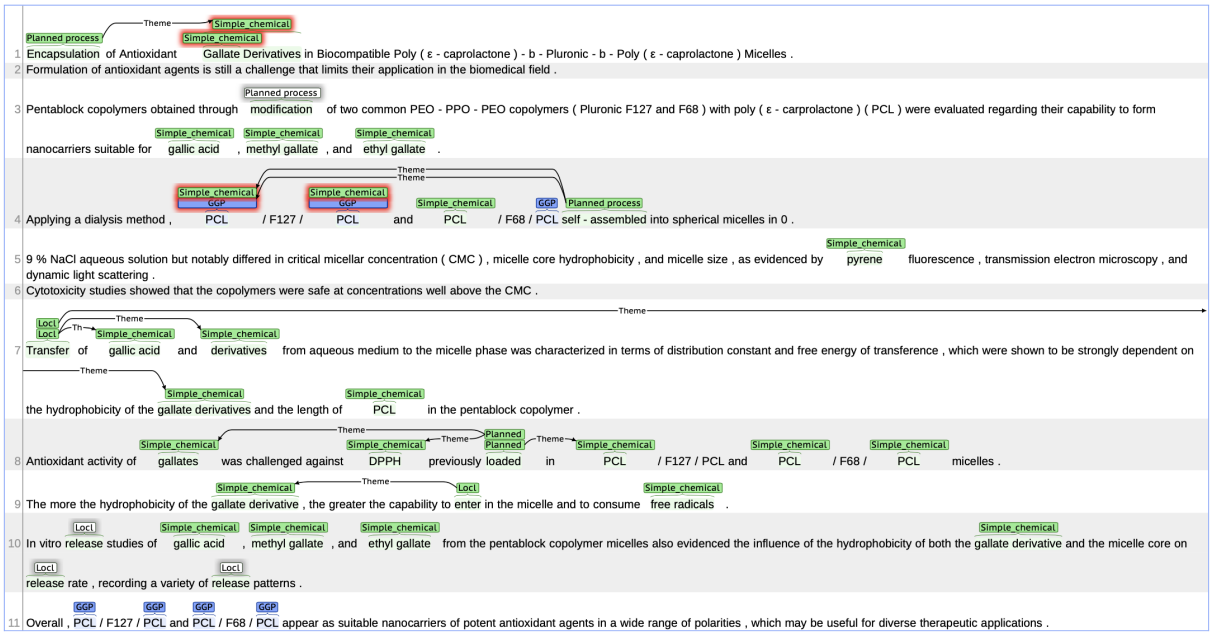


Figure C.19: Automatic event extraction with SciBERT (`PMID: 26986801`).

Figure C.20 shows the events automatically extracted from the abstract using SciBERT−$KG_{tr,r,ar}$.

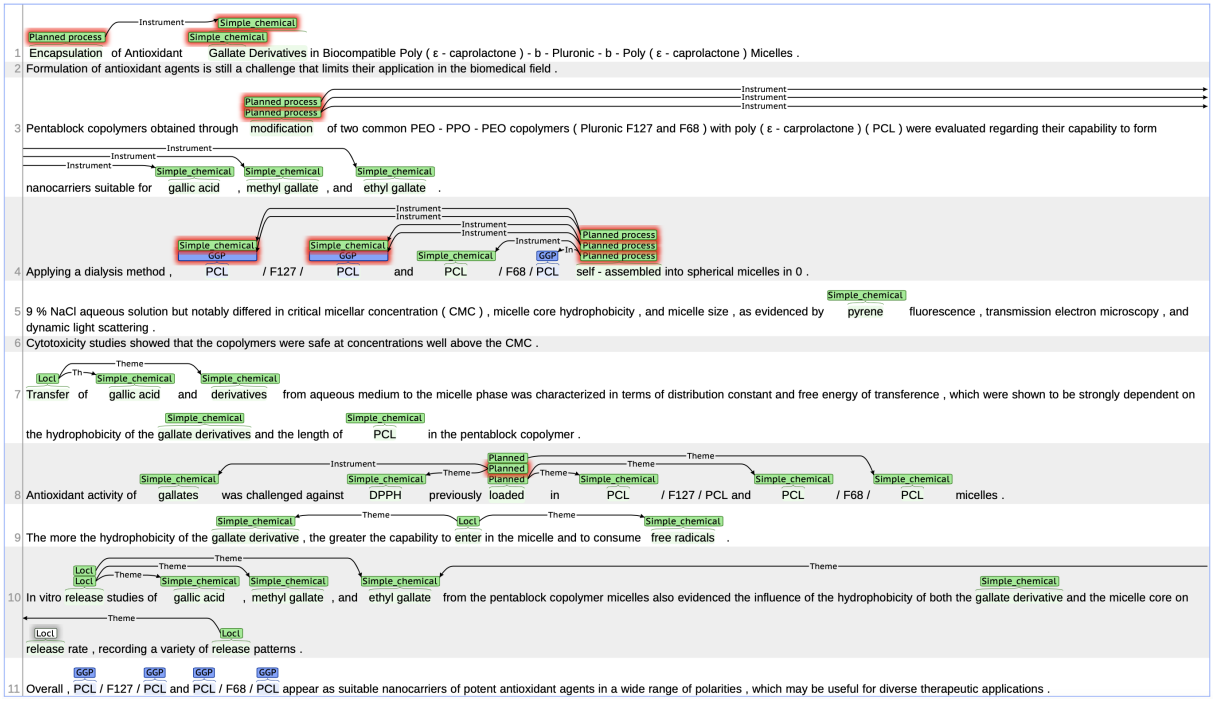


Figure C.20: Automatic event extraction with SciBERT-$KG_{tr,r,ar}$ (ours) (`PMID: 26986801`).

Résumé étendu

Une partie significative de la connaissance biomédicale mondiale est documentée sous forme de texte en langage naturel, allant des publications scientifiques et des rapports médicaux aux dossiers de santé électroniques et à la littérature biomédicale. Ces documents contenant des connaissances biomédicales sont en constante croissance, ce qui rend difficile de les lire à la même vitesse à laquelle ils sont produits. Par conséquent, une grande partie de cette information reste stagnante dans les détails du texte qui ne sont pas exploitables pour une analyse ou une application ultérieure. Utiliser efficacement cette information dans son format non structuré représente un défi majeur. Les techniques de Traitement Automatique des Langues (TAL, NLP en anglais) offrent des outils puissants et des méthodologies pour libérer le potentiel de cette connaissance textuelle en la transformant en un format structuré adapté à la manipulation et à l'analyse par ordinateur. Ce processus, connu sous le nom d'extraction d'informations, joue un rôle crucial dans la conversion de texte non structuré en connaissances exploitables.

Les techniques d'extraction d'informations permettent aux chercheurs et aux experts du domaine d'extraire des connaissances structurées à partir de textes biomédicaux, les rendant disponibles pour diverses analyses computationnelles, découvertes de connaissances et systèmes de support à la décision. En identifiant et extrayant automatiquement des informations pertinentes telles que les entités biomédicales et leurs relations, ces techniques peuvent être mises en œuvre à des fins avancées d'exploration de données, de recherche sémantique, de découverte basée sur la littérature, de support à la décision clinique et de découverte de médicaments.

Formellement, ces techniques d'extraction d'informations sont la Reconnaissance d'Entités Nommées (NER en anglais), l'Extraction de Relations (RE, en anglais) et l'Extraction d'Événements (EE, en anglais). Le NER implique l'analyse de texte non structuré pour identifier les entités pour la normalisation des termes et la classification en catégories prédéfinies, par exemple, noms propres, organisations, dates et lieux. Ces catégories peuvent être liées à un domaine spécifique, par exemple, la Figure 21 montre une phrase annotée avec deux entités nommées du domaine biomédical, *Simple_ chemical* et *Gene_ or_ gene_ product*.

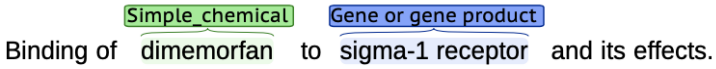


Figure 21: Exemple de NER.

Lorsque la NER est appliquée au domaine biomédical, les entités sont généralement catégorisées en gènes et protéines, médicaments, effets indésirables, maladies, tissus, organes, voies ou métabolites. Parmi les raisons qui rendent la REN biomédicale difficile, on trouve l'utilisation non standard d'abréviations, les synonymes, les homonymes, les ambiguïtés et les entités composées de plusieurs mots.

La RE est une tâche qui suit généralement la NER et son but est d'identifier quelles entités sont liées les unes aux autres pour trouver des interactions significatives. La Figure 22 montre un exemple où les deux entités nommées reconnues précédemment sont liées avec le type de relation *Agoniste*. Le terme « agoniste » fait référence à une substance qui se lie à un récepteur et l'active pour produire une réponse biologique. Dans ce contexte, si le *dimemorfan* se lie au *sigma-1 receptor* et entraîne une réponse ou une activation, cela serait considéré comme un agoniste du *sigma-1 receptor*.

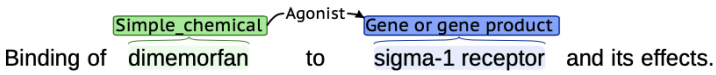


Figure 22: Exemple de RE.

Par exemple, dans la RE biomédicale, l'identification des interactions entre les protéines permet de construire des réseaux d'interactions protéine-protéine. De la même manière, la localisation des relations gène-maladie peut établir un lien entre l'information moléculaire et phénotypique. Ces réseaux de relations offrent la possibilité d'explorer des connexions qui étaient auparavant inconnues et de les associer à des relations établies. Cependant, l'un des principaux défis de la RE biomédicale est qu'elle perd des détails précieux sur le contexte tout en essayant de maintenir des interactions précises entre les entités biologiquement pertinentes. Cela peut limiter l'utilité pratique des relations extraites. Parmi les détails qui peuvent être perdus en RE, on trouve les sous-relations (par exemple, *At Loc*, pour faire référence à l'emplacement de l'événement) et la participation conjointe de plus de deux entités qui couvrent des fonctions séparées (par exemple, tandis que *gene_1* est la cause de régulation, *protein_1* et *protein_2* sont affectées par cette régulation) [Perera et al., 2020].

EE est une méthode plus expressive pour capturer des énoncés en langage naturel qui permettent de représenter formellement des processus biologiques, tels que l'étude des mécanismes biomoléculaires ou des changements épigénétiques. La Figure 23 montre un exemple où les deux entités nommées reconnues précédemment sont les arguments de l'événement *Binding*.

L'événement est construit à partir du mot déclencheur « Binding ». L'un des arguments, faisant référence à l'entité *Simple_ chemical*, joue le rôle de *Theme* (répondant à la question « Qu'est-ce que a été lié?"). L'autre argument, faisant référence à l'entité *Gene_ or_ gene_ product*, joue le rôle du *Site* (réponse à la question « Où s'est produite la liaison ? »)

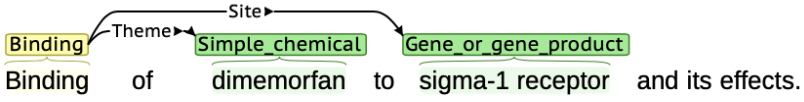


Figure 23: Exemple de EE.

Par conséquent, les structures des événements sont composées de multiples participants ayant un rôle sémantique spécifique, y compris des événements imbriqués et des définitions qui se chevauchent. Tandis que dans le RE, la structure est composée uniquement de la paire de participants qui forme la relation. Actuellement, les événements sont les unités d'information les plus complexes qui peuvent être extraites du texte, car ils sont capables de capturer des relations dynamiques n-aires entre les entités et également entre les événements [Frisoni et al., 2021].

De cette manière, les événements offrent une flexibilité pour interconnecter les entités, permettant de construire une description plus fine en prenant en compte tous les éléments du contexte. Les propriétés dynamiques font des événements l'équivalent le plus proche de l'information extraite directement des humains [Zerva and Ananiadou, 2015]. Par conséquent, le RE permet d'obtenir des interactions exhaustives, interprétables et quantifiables décrivant des connaissances implicites et émergentes qui seraient autrement enfouies dans le texte.

Un événement biomédical est construit à partir d'un déclencheur d'événement et d'un ou plusieurs arguments qui gravitent autour du déclencheur. Les déclencheurs d'événements se réfèrent généralement à des noms ou des verbes exprimant une action, une circonstance ou une éventualité, tandis que les arguments se réfèrent soit à des entités biomédicales, soit à d'autres événements, appelés événements imbriqués.

Pour acquérir ces éléments et construire un événement, la tâche est généralement divisée en trois sous-tâches principales : la détection des déclencheurs d'événements, l'identification des arguments et la construction de l'événement. La détection des déclencheurs d'événements identifie et classe les mots déclencheurs en un ensemble de types prédéfinis de déclencheurs d'événements, tandis que l'identification des arguments identifie et classe les rôles entre les déclencheurs d'événements et leurs arguments respectifs [Shen et al., 2019]. La construction de l'événement fait référence à la désagrégation des arguments qui correspondent au même événement, c'est-à-dire que les événements chevauchants sont fusionnés en un nœud unifié avec son ensemble correspondant d'arêtes sortantes [Björne and Salakoski, 2011].

La détection de déclencheurs d'événements joue un rôle fondamental dans la construction

des événements. En effet, les déclencheurs sont les cibles qui nous permettent de savoir qu'un événement peut exister [Cui et al., 2020]. Cette sous-tâche est généralement considérée comme un problème de classification, où chaque mot doit être classifié dans un ensemble prédéfini de types de déclencheurs. La difficulté de la détection des déclencheurs provient de la sensibilité au domaine ou sous-domaine (le texte peut présenter un langage spécialisé), des formes linguistiques (les déclencheurs peuvent être des mots simples, des expressions multiples, des marqueurs discontinus) et de l'ambiguïté sur la classe de déclencheur (un mot de déclenchement peut être attribué à différentes classes de déclencheurs) [Zerva and Ananiadou, 2015].

L'identification des arguments peut être considérée comme un problème de classification multi-catégories, où la relation dirigée entre un déclencheur et une entité ou un autre événement doit être classée dans un ensemble prédéfini de types de rôles. Lorsque ces arguments sont correctement identifiés, l'événement extrait a le potentiel de fournir un moyen fiable d'améliorer la connaissance du domaine. L'une des principales complexités dans l'identification des arguments est qu'ils peuvent faire partie d'un ou de plusieurs événements (relations un-à-un et un-à-plusieurs), où ils jouent des rôles similaires ou différents.

L'un des principaux défis en ER réside dans l'identification de l'endroit où les événements sont construits dans le texte et comment les classifier, car les événements peuvent appartenir à différentes catégories, assumer différents rôles [Ramponi et al., 2020]. Cela entraîne l'échec de la récupération de certaines informations, les rendant incomplètes ou erronées. Bien que de telles erreurs ne soient pas toujours d'une importance vitale, elles peuvent être extrêmement préjudiciables lorsque l'information est utilisée pour une application biomédicale, car elles peuvent conduire à des affirmations erronées ou non utiles. C'est l'une des principales raisons pour lesquelles le développement et l'amélioration de modèles automatiques d'extraction d'informations biomédicales constituent un enjeu clé dans la recherche scientifique.

Les modèles de langage basés sur les transformateurs ont été largement adoptés pour tenter de réduire les erreurs dans l'EE en raison de leurs réalisations positives en termes de performance pour résoudre différents types de tâches de TAL. BERT [Devlin et al., 2018], qui signifie *Bidirectional Encoder Representations from Transformers*, est un modèle de langage conçu pour pré-entraîner des représentations bidirectionnelles de mots, en tenant compte de la sémantique en considérant à la fois les directions gauche et droite du texte. À partir de cet pré-entraînement, BERT peut être affiné en incluant des couches supplémentaires au-dessus du modèle pour résoudre de nouvelles tâches spécifiques. De plus, plusieurs variantes de BERT spécifiques au domaine ont été développées en étant formées sur de grands corpus avec le même contexte, comme le domaine biomédical. Cependant, comme l'apprentissage des modèles est limité au sous-domaine dans lequel ils ont été formés, ils présentent des limitations de performance lorsqu'on les utilise dans différents sous-domaines biomédicaux.

Pour améliorer l'intégration de la connaissance du domaine, des modèles de graphe de

connaissances (KGs) ont été mis en œuvre conjointement avec des modèles de langage pour différentes tâches d'information dans le domaine biomédical [Huang et al., 2020, Yang et al., 2020, Dasgupta et al., 2021, Roy and Pan, 2021, Milošević and Thielemann, 2023].

Les KG biomédicaux sont une ressource d'intégration d'une ou plusieurs sources d'informations (souvent des ensembles de données soigneusement élaborés manuellement) dans un graphe, où les entités biomédicales sont représentées par des nœuds et les relations entre elles par des arêtes [Nicholson and Greene, 2020]. Les modèles de KG intègrent les nœuds et les arêtes dans un espace vectoriel de faible dimension, appelé embeddings, tout en préservant la structure du graphe de connaissances et son information sémantique. Cet espace vectoriel joue un rôle essentiel dans l'augmentation automatisée du KG. Ce processus implique la prédiction de connexions manquantes entre les nœuds, communément appelée tâche de prédiction de liens. En exploitant l'information sémantique encodée dans les embeddings, le système peut déduire et incorporer de nouvelles relations, améliorant ainsi la globalité et l'utilité du graphe. L'objectif de la prédiction de liens est d'inférer de nouvelles relations entre les entités, basées sur les relations précédemment apprises à partir du graphe. Ainsi, avec les nouvelles connexions trouvées, de nouvelles informations sont intégrées dans le graphe et la connaissance est enrichie.

6 Proposition et contributions

Dans cette thèse, nous commençons par analyser les performances de cinq modèles de langage transformateur préalablement entraînés pour déterminer s'ils permettent l'identification de déclencheurs dans différents sous-domaines biomédicaux. Ensuite, nous enrichissons l'information sémantique du modèle offrant les meilleures performances avec des plongements de KG pour évaluer si l'intégration de ces plongements améliore la capacité du modèle à identifier les arguments biomédicaux et leurs rôles. Dans ce but, BERT, BioBERT, SciBERT, PubMedBERT et BioMedRoBERTa sont affinés en utilisant deux classificateurs différents, une couche linéaire et une couche Bi-LSTM (*Bidirectional Long-Short Term Memory*) pour détecter les déclencheurs d'événements biomédicaux. Ces variantes de BERT sont choisies pour la comparaison car elles partagent la même architecture BERT mais ont été pré-entraînées auparavant avec différentes données dans le domaine biomédical et/ou général, montrant des résultats positifs dans les tâches d'extraction d'informations biomédicales [Lee et al., 2020, Beltagy et al., 2019, Erdengasileng et al., 2022]. Les modèles sont appris en utilisant sept ensembles de données annotées manuellement fusionnés, Cancer Genetics (CG) 2013 [Nédellec et al., 2013], Epigenetics and Post-translational Modifications (EPI) 2011 [Ohta et al., 2011], GENIA 2011 [Kim et al., 2011], GENIA 2013 [Kim et al., 2013], Infectious Diseases (ID) 2011 [Pyysalo et al., 2011], Pathway Curation (PC) 2013 [Nédellec et al., 2013], Multi-Level Event Extraction (MLEE) [Pyysalo et al., 2012]. Ces corpus ont été initialement développés pour la tâche d'extraction d'événements dans différents

sous-domaines biomédicaux. En plus de ces données, deux caractéristiques sont incluses en tant qu'informations lexicales et syntaxiques supplémentaires pour les modèles, les racines et les parties du discours (POS, en anglais), respectivement. Ensuite, un KG est construit à partir des événements biomédicaux contenus dans les corpus biomédicaux et ses plongements de KG sont calculés. Ces plongements sont intégrés dans le modèle de langage transformateur pour classifier les rôles entre les déclencheurs précédemment identifiés et les entités biomédicales et/ou autres déclencheurs, afin de détecter les arguments d'événements.

Les principales contributions de cette thèse sont résumées comme suit :

- Évaluation et comparaison de cinq modèles de langage transformateur basés sur BERT pour la détection des déclencheurs d'événements biomédicaux ;
- Évaluation de l'ajout d'informations lexicales et syntaxiques pour l'affinage des modèles afin d'améliorer la détection d'événements biomédicaux ;
- Proposition d'une nouvelle stratégie pour intégrer les plongements de KGs dans les modèles de langage transformateur pour identifier les arguments d'événements biomédicaux ;
- Analyse empirique de l'impact de la fusion de différents corpus annotés pour détecter les déclencheurs d'événements biomédicaux et identifier les arguments dans différents sous-domaines biomédicaux.

7 Aperçu de la thèse

L'EE biomédicale joue un rôle essentiel pour révéler les connaissances cachées contenues dans de vastes quantités de texte non structuré. Les événements biomédicaux, tels que les interactions gène-protéine, les relations médicament-maladie et les processus moléculaires, renferment des informations cruciales pour comprendre les mécanismes biologiques, identifier les cibles thérapeutiques et faire progresser la médecine de précision. Dans les sections suivantes, nous décrirons différentes techniques d'extraction d'informations et approches de TAL utilisées pour l'EE biomédicale, en mettant particulièrement l'accent sur les modèles transformateurs et leur intégration avec les KGs. En combinant la compréhension contextuelle des modèles transformateurs et la représentation structurée des graphes de connaissances, nous étudions leur capacité à améliorer la précision de l'ER biomédicale et à fournir une représentation globale des connaissances.

- **Chapitre 1.** L'objectif de ce chapitre est d'explorer les avancées dans l'extraction d'informations biomédicales, y compris l'EE biomédicale, en utilisant des techniques de TAL. Nous décrivons les approches et méthodologies de pointe utilisées dans l'extraction

d'informations pour surmonter les défis actuels et extraire des connaissances précieuses à partir de textes biomédicaux.

- **Chapitre 2.** Ce chapitre décrit les avancées réalisées dans l'EE biomédicale en utilisant des modèles de langage transformateur, y compris BERT et certaines de ses variantes biomédicales. Nous abordons les principes et les méthodologies sous-jacents aux approches basées sur les transformateurs, mettant en lumière les adaptations et les progrès réalisés pour l'ER biomédicale. En examinant les capacités et les limitations des modèles de langage transformateur, nous visons à fournir une compréhension globale de leur application dans l'extraction d'événements biomédicaux à partir de données textuelles non structurées.

- **Chapitre 3.** Ce chapitre se concentre sur les récents développements dans l'utilisation des KGs pour la découverte de connaissances biomédicales. Nous explorons les principes et les méthodologies suivies dans les graphes de connaissances, ainsi que leurs applications potentielles dans le domaine biomédical. En examinant les capacités et les limitations des KGs, nous visons à fournir une compréhension globale de leur rôle dans l'extraction et l'organisation des connaissances biomédicales à partir de données textuelles non structurées.

- **Chapitre 4.** Ce chapitre décrit les deux premières contributions de la thèse, où nous comparons les stratégies de pré-entraînement de BERT et de quatre de ses variantes, ainsi que leur impact sur l'extraction d'événements dans différents sous-domaines biomédicaux. Nous analysons et évaluons comment leurs différentes caractéristiques de pré-entraînement, telles que les poids initiaux et les données d'entraînement, peuvent influencer la détection de déclencheurs d'événements biomédicaux. De plus, nous évaluons l'intégration d'informations lexicales et syntaxiques aux modèles pour améliorer la détection des déclencheurs d'événements. À travers cette évaluation, nous avons observé comment le pré-entraînement de ces modèles influence effectivement la détection des déclencheurs, ce qui est un point important à prendre en compte lors du choix du modèle pour cette tâche. De plus, nous avons constaté que l'ajout d'informations supplémentaires n'améliore pas les performances des modèles pour cette tâche.

- **Chapitre 5.** Dans ce chapitre, nous présentons la troisième contribution, où nous explorons une approche novatrice pour améliorer l'identification des arguments dans différents sous-domaines biomédicaux en utilisant un modèle transformateur enrichi en connaissances de graphe. En fusionnant les plongements issus des modèles transformateurs et des KGs, notre objectif est de découvrir des relations significatives entre les déclencheurs d'événements et leurs arguments correspondants. À travers des expériences et des évaluations détaillées, nous mettons en lumière l'efficacité de cette approche intégrée pour identifier les arguments des événements biomédicaux.

- **Chapitre 6** Dans ce chapitre, nous décrivons la quatrième contribution, une évaluation qualitative et quantitative de l'information intégrée dans les plongements calculés avec le modèle transformateur et les KGs. Nous réalisons une analyse de performance de la détection des déclencheurs et de l'identification des arguments en fonction des différents sous-domaines. De plus, nous comparons notre modèle avec des modèles de référence et nous présentons un cas d'utilisation de l'application du modèle dans un texte biomédical.

- **Chapitre 7.** Dans ce chapitre, nous présentons une étude exploratoire axée sur l'extraction d'événements biomédicaux à partir de textes biomoléculaires. Notre approche consiste à appliquer le modèle proposé, qui combine les plongements d'un modèle transformateur et les KGs, à un ensemble de résumés biomédicaux sélectionnés par une équipe biomoléculaire interdisciplinaire. Cette étude vise à identifier l'application potentielle de notre modèle dans l'automatisation de l'extraction d'événements à partir de textes biomédicaux complexes.

Bibliography

[Abdulkadhar et al., 2021] Abdulkadhar, S., Bhasuran, B., and Natarajan, J. (2021). Multiscale laplacian graph kernel combined with lexico-syntactic patterns for biomedical event extraction from literature. *Knowledge and Information Systems*, 63:143–173.

[Akbik and Löser, 2012] Akbik, A. and Löser, A. (2012). Kraken: N-ary facts in open information extraction. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 52–56.

[Anaby-Tavor et al., 2020] Anaby-Tavor, A., Carmeli, B., Goldbraich, E., Kantor, A., Kour, G., Shlomov, S., Tepper, N., and Zwerdling, N. (2020). Do not have enough data? deep learning to the rescue! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7383–7390.

[Araujo et al., 2020] Araujo, V., Carvallo, A., Aspillaga, C., and Parra, D. (2020). On adversarial examples for biomedical nlp tasks. *arXiv preprint arXiv:2004.11157*.

[Beltagy et al., 2019] Beltagy, I., Lo, K., and Cohan, A. (2019). Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

[Bengio et al., 2000] Bengio, Y., Ducharme, R., and Vincent, P. (2000). A neural probabilistic language model. *Advances in neural information processing systems*, 13.

[Bhasuran and Natarajan, 2018] Bhasuran, B. and Natarajan, J. (2018). Automatic extraction of gene-disease associations from literature using joint ensemble learning. *PloS one*, 13(7):e0200699.

[Bhutani et al., 2016] Bhutani, N., Jagadish, H., and Radev, D. (2016). Nested propositions in open information extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 55–64.

[Björne and Salakoski, 2011] Björne, J. and Salakoski, T. (2011). Generalizing biomedical event extraction. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 183–191.

[Björne and Salakoski, 2013] Björne, J. and Salakoski, T. (2013). Tees 2.1: Automated annotation scheme learning in the bionlp 2013 shared task. In *Proceedings of the BioNLP shared task 2013 workshop*, pages 16–25.

[Björne and Salakoski, 2018] Björne, J. and Salakoski, T. (2018). Biomedical event extraction using convolutional neural networks and dependency parsing. In *Proceedings of the BioNLP 2018 workshop*, pages 98–108.

[Bojanowski et al., 2017] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

[Bordes et al., 2014] Bordes, A., Glorot, X., Weston, J., and Bengio, Y. (2014). A semantic matching energy function for learning with multi-relational data: Application to word-sense disambiguation. *Machine Learning*, 94:233–259.

[Bordes et al., 2013] Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

[Braud and Denis, 2015] Braud, C. and Denis, P. (2015). Comparing word representations for implicit discourse relation classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2201–2211.

[Bundschus et al., 2008] Bundschus, M., Dejori, M., Stetter, M., Tresp, V., and Kriegel, H.-P. (2008). Extraction of semantic biomedical relations from text using conditional random fields. *BMC bioinformatics*, 9(1):1–14.

[Campos et al., 2012] Campos, D., Matos, S., and Oliveira, J. L. (2012). Biomedical named entity recognition: a survey of machine-learning tools. *Theory and Applications for Advanced Text Mining*, 11:175–195.

[Campos et al., 2013] Campos, D., Matos, S., and Oliveira, J. L. (2013). Gimli: open source and high-performance biomedical name recognition. *BMC bioinformatics*, 14(1):1–14.

[Chen et al., 2008] Chen, E. S., Hripcsak, G., Xu, H., Markatou, M., and Friedman, C. (2008). Automated acquisition of disease–drug knowledge from biomedical and clinical documents: an initial study. *Journal of the American Medical Informatics Association*, 15(1):87–98.

[Chen et al., 2022a] Chen, Q., Allot, A., Leaman, R., Islamaj, R., Du, J., Fang, L., Wang, K., Xu, S., Zhang, Y., Bagherzadeh, P., et al. (2022a). Multi-label classification for biomedical literature: an overview of the biocreative vii litcovid track for covid-19 literature topic annotations. *Database*, 2022.

[Chen, 2021] Chen, Y. (2021). A transfer learning model with multi-source domains for biomedical event trigger extraction. *BMC genomics*, 22(1):1–18.

[Chen et al., 2022b] Chen, Z., Peng, B., Ioannidis, V. N., Li, M., Karypis, G., and Ning, X. (2022b). A knowledge graph of clinical trials (ctkg). *Scientific reports*, 12(1):4724.

[Cheng et al., 2008] Cheng, D., Knox, C., Young, N., Stothard, P., Damaraju, S., and Wishart, D. S. (2008). Polysearch: a web-based text mining system for extracting relationships between human diseases, genes, mutations, drugs and metabolites. *Nucleic acids research*, 36(suppl_2):W399–W405.

[Cui et al., 2020] Cui, S., Yu, B., Liu, T., Zhang, Z., Wang, X., and Shi, J. (2020). Event detection with relation-aware graph convolutional neural networks. *arXiv e-prints*, pages arXiv–2002.

[Dai et al., 2015] Dai, W., Liu, X., Gao, Y., Chen, L., Song, J., Chen, D., Gao, K., Jiang, Y., Yang, Y., Chen, J., et al. (2015). Matrix factorization-based prediction of novel drug indications by integrating genomic space. *Computational and mathematical methods in medicine*, 2015.

[Dasgupta et al., 2021] Dasgupta, S., Jayagopal, A., Hong, A. L. J., Mariappan, R., Rajan, V., et al. (2021). Adverse drug event prediction using noisy literature-derived knowledge graphs: algorithm development and validation. *JMIR Medical Informatics*, 9(10):e32730.

[de Vroe et al., 2021] de Vroe, S. B., Guillou, L., Stanojević, M., McKenna, N., and Steedman, M. (2021). Modality and negation in event extraction. *arXiv preprint arXiv:2109.09393*.

[Dettmers et al., 2018] Dettmers, T., Minervini, P., Stenetorp, P., and Riedel, S. (2018). Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

[Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[Dong et al., 2014] Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., and Zhang, W. (2014). Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610.

[Duan et al., 2017] Duan, S., He, R., and Zhao, W. (2017). Exploiting document level information to improve event detection via recurrent neural networks. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 352–361.

[Eftimov et al., 2017] Eftimov, T., Koroušić Seljak, B., and Korošec, P. (2017). A rule-based named-entity recognition method for knowledge extraction of evidence-based dietary recommendations. *PloS one*, 12(6):e0179488.

[El-allaly et al., 2021] El-allaly, E.-d., Sarrouti, M., En-Nahnahi, N., and El Alaoui, S. O. (2021). Mttlade: A multi-task transfer learning-based method for adverse drug events extraction. *Information Processing & Management*, 58(3):102473.

[Emmert-Streib et al., 2020] Emmert-Streib, F., Yang, Z., Feng, H., Tripathi, S., and Dehmer, M. (2020). An introductory review of deep learning for prediction models with big data. *Frontiers in Artificial Intelligence*, 3:4.

[Erdengasileng et al., 2022] Erdengasileng, A., Han, Q., Zhao, T., Tian, S., Sui, X., Li, K., Wang, W., Wang, J., Hu, T., Pan, F., et al. (2022). Pre-trained models, data augmentation, and ensemble learning for biomedical information extraction and document classification. *Database*, 2022.

[Fan et al., 2014] Fan, M., Zhou, Q., Chang, E., and Zheng, F. (2014). Transition-based knowledge graph embedding with relational mapping properties. In *Proceedings of the 28th Pacific Asia conference on language, information and computing*, pages 328–337.

[Fei et al., 2021] Fei, H., Ren, Y., Zhang, Y., Ji, D., and Liang, X. (2021). Enriching contextualized language model from knowledge graph for biomedical information extraction. *Briefings in bioinformatics*, 22(3):bbaa110.

[Feng et al., 2016] Feng, J., Huang, M., Wang, M., Zhou, M., Hao, Y., and Zhu, X. (2016). Knowledge graph embedding by flexible translation. In *Fifteenth International Conference on the Principles of Knowledge Representation and Reasoning*.

[Feng et al., 2020] Feng, S. Y., Gangal, V., Kang, D., Mitamura, T., and Hovy, E. (2020). Genaug: Data augmentation for finetuning text generators. *arXiv preprint arXiv:2010.01794*.

[Feng et al., 2019] Feng, S. Y., Li, A. W., and Hoey, J. (2019). Keep calm and switch on! preserving sentiment and fluency in semantic text exchange. *arXiv preprint arXiv:1909.00088*.

[Frisoni et al., 2021] Frisoni, G., Moro, G., and Carbonaro, A. (2021). A survey on event extraction for natural language understanding: Riding the biomedical literature wave. *IEEE Access*, 9:160721–160757.

[Fundel et al., 2007] Fundel, K., Küffner, R., and Zimmer, R. (2007). Relex—relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–371.

[Furrer et al., 2019] Furrer, L., Jancso, A., Colic, N., and Rinaldi, F. (2019). Oger++: hybrid multi-type entity recognition. *Journal of cheminformatics*, 11(1):1–10.

[Gaizauskas et al., 2003] Gaizauskas, R., Demetriou, G., Artymiuk, P. J., and Willett, P. (2003). Protein structures and information extraction from biological texts: the pasta system. *Bioinformatics*, 19(1):135–143.

[Gao et al., 2023] Gao, J., Zhao, H., Yu, C., and Xu, R. (2023). Exploring the feasibility of chatgpt for event extraction. *arXiv preprint arXiv:2303.03836*.

[Gerner et al., 2010] Gerner, M., Nenadic, G., and Bergman, C. M. (2010). Linnaeus: a species name identification system for biomedical literature. *BMC bioinformatics*, 11(1):1–17.

[Giorgi and Bader, 2020] Giorgi, J. M. and Bader, G. D. (2020). Towards reliable named entity recognition in the biomedical domain. *Bioinformatics*, 36(1):280–286.

[Gopalakrishnan et al., 2019] Gopalakrishnan, V., Jha, K., Jin, W., and Zhang, A. (2019). A survey on literature based discovery approaches in biomedical domain. *Journal of biomedical informatics*, 93:103141.

[Gridach, 2017] Gridach, M. (2017). Character-level neural network for biomedical named entity recognition. *Journal of biomedical informatics*, 70:85–91.

[Gu et al., 2021] Gu, Y., Tinn, R., Cheng, H., Lucas, M., Usuyama, N., Liu, X., Naumann, T., Gao, J., and Poon, H. (2021). Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23.

[Gururangan et al., 2020] Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., and Smith, N. A. (2020). Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of ACL*.

[Habibi et al., 2017] Habibi, M., Weber, L., Neves, M., Wiegandt, D. L., and Leser, U. (2017). Deep learning with word embeddings improves biomedical named entity recognition. *Bioinformatics*, 33(14):i37–i48.

[He et al., 2021] He, X., Ren, Y., Tai, P., and Shi, H. (2021). A two-stage biomedical event trigger detection method based on hybrid neural network and sentence embeddings. *IEEE Access*, 9:81926–81935.

[Hong et al., 2020] Hong, L., Lin, J., Li, S., Wan, F., Yang, H., Jiang, T., Zhao, D., and Zeng, J. (2020). A novel machine learning framework for automated biomedical relation extraction from large-scale literature repositories. *Nature Machine Intelligence*, 2(6):347–355.

[Hsieh et al., 2017] Hsieh, Y.-L., Chang, Y.-C., Chang, N.-W., and Hsu, W.-L. (2017). Identifying protein-protein interactions in biomedical literature using recurrent neural networks with long short-term memory. In *Proceedings of the eighth international joint conference on natural language processing (volume 2: short papers)*, pages 240–245.

[Hua and Quan, 2016] Hua, L. and Quan, C. (2016). A shortest dependency path based convolutional neural network for protein-protein relation extraction. *BioMed research international*, 2016.

[Huang et al., 2020] Huang, K.-H., Yang, M., and Peng, N. (2020). Biomedical event extraction with hierarchical knowledge graphs. *arXiv preprint arXiv:2009.09335*.

[Islam et al., 2021] Islam, M. K., Aridhi, S., and Smaïl-Tabbone, M. (2021). Simple negative sampling for link prediction in knowledge graphs. In *International Conference on Complex Networks and Their Applications*, pages 549–562. Springer.

[Jagannatha and Yu, 2016] Jagannatha, A. N. and Yu, H. (2016). Bidirectional rnn for medical event detection in electronic health records. In *Proceedings of the conference. Association for Computational Linguistics. North American Chapter. Meeting*, volume 2016, page 473. NIH Public Access.

[Jensen et al., 2014] Jensen, K., Panagiotou, G., and Kouskoumvekaki, I. (2014). Integrated text mining and chemoinformatics analysis associates diet to health benefit at molecular level. *PLoS computational biology*, 10(1):e1003432.

[Jettakul et al., 2019] Jettakul, A., Wichadakul, D., and Vateekul, P. (2019). Relation extraction between bacteria and biotopes from biomedical texts with attention mechanisms and domain-specific contextual representations. *BMC bioinformatics*, 20:1–17.

[Ji et al., 2015] Ji, G., He, S., Xu, L., Liu, K., and Zhao, J. (2015). Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, pages 687–696.

[Jivani et al., 2011] Jivani, A. G. et al. (2011). A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*, 2(6):1930–1938.

[Joshi et al., 2020] Joshi, M., Chen, D., Liu, Y., Weld, D. S., Zettlemoyer, L., and Levy, O. (2020). Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

[Kalyan et al., 2022] Kalyan, K. S., Rajasekharan, A., and Sangeetha, S. (2022). Ammu: a survey of transformer-based biomedical pretrained language models. *Journal of biomedical informatics*, 126:103982.

[Kazemi and Poole, 2018] Kazemi, S. M. and Poole, D. (2018). Simple embedding for link prediction in knowledge graphs. *Advances in neural information processing systems*, 31.

[Khalid et al., 2021] Khalid, U., Beg, M. O., and Arshad, M. U. (2021). Rubert: A bilingual roman urdu bert using cross lingual transfer learning. *arXiv preprint arXiv:2102.11278*.

[Kim et al., 2011] Kim, J.-D., Wang, Y., Takagi, T., and Yonezawa, A. (2011). Overview of genia event task in bionlp shared task 2011. In *Proceedings of BioNLP shared task 2011 workshop*, pages 7–15.

[Kim et al., 2013] Kim, J.-D., Wang, Y., and Yasunori, Y. (2013). The genia event extraction shared task, 2013 edition-overview. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 8–15.

[Kim et al., 2016] Kim, Y., Jernite, Y., Sontag, D., and Rush, A. (2016). Character-aware neural language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

[Kipf and Welling, 2016] Kipf, T. N. and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

[Koroteev, 2021] Koroteev, M. (2021). Bert: a review of applications in natural language processing and understanding. *arXiv preprint arXiv:2103.11943*.

[Kristiadi et al., 2019] Kristiadi, A., Khan, M. A., Lukovnikov, D., Lehmann, J., and Fischer, A. (2019). Incorporating literals into knowledge graph embeddings. In *International Semantic Web Conference*, pages 347–363. Springer.

[Kumar et al., 2020] Kumar, V., Choudhary, A., and Cho, E. (2020). Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

[Lafferty et al., 2001] Lafferty, J., McCallum, A., and Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

[Lai et al., 2021] Lai, T., Ji, H., Zhai, C., and Tran, Q. H. (2021). Joint biomedical entity and relation extraction with knowledge-enhanced collective inference. *arXiv preprint arXiv:2105.13456*.

[Leaman and Gonzalez, 2008] Leaman, R. and Gonzalez, G. (2008). Banner: an executable survey of advances in biomedical named entity recognition. In *Biocomputing 2008*, pages 652–663. World Scientific.

[Leaman et al., 2013] Leaman, R., Islamaj Doğan, R., and Lu, Z. (2013). Dnorm: disease name normalization with pairwise learning to rank. *Bioinformatics*, 29(22):2909–2917.

[LeCun et al., 2015] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

[Lee et al., 2020] Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., and Kang, J. (2020). Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

[Levy and Goldberg, 2014] Levy, O. and Goldberg, Y. (2014). Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308.

[Lewis et al., 2019] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

[Li et al., 2016] Li, L., Jin, L., Jiang, Y., and Huang, D. (2016). Recognizing biomedical named entities based on the sentence vector/twin word embeddings conditioned bidirectional lstm. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data: 15th China National Conference, CCL 2016, and 4th International Symposium, NLP-NABD 2016, Yantai, China, October 15-16, 2016, Proceedings 15*, pages 165–176. Springer.

[Lin et al., 2015] Lin, Y., Liu, Z., Sun, M., Liu, Y., and Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29.

[Liu et al., 2013a] Liu, H., Verspoor, K., Comeau, D. C., MacKinlay, A., and Wilbur, W. J. (2013a). Generalizing an approximate subgraph matching-based system to extract events in molecular biology and cancer genetics. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 76–85.

[Liu et al., 2017] Liu, H., Wu, Y., and Yang, Y. (2017). Analogical inference for multi-relational embeddings. In *International conference on machine learning*, pages 2168–2178. PMLR.

[Liu et al., 2023] Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., and Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

[Liu et al., 2016] Liu, Q., Jiang, H., Evdokimov, A., Ling, Z.-H., Zhu, X., Wei, S., and Hu, Y. (2016). Probabilistic reasoning via deep learning: Neural association models. *arXiv preprint arXiv:1603.07704*.

[Liu et al., 2013b] Liu, X., Bordes, A., and Grandvalet, Y. (2013b). Biomedical event extraction by multi-class classification of pairs of text entities. In *BioNLP shared task 2013 workshop*, pages 45–49.

[Liu et al., 2019] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

[Mahendran et al., 2021] Mahendran, D., Ranjan, S., Tang, J., Nguyen, M., and McInnes, B. T. (2021). Biocreative vii-track 1: A bert-based system for relation extraction in biomedical text. In *BioCreative VII Workshop*.

[Makino et al., 2002] Makino, T., Ohta, Y., Tsujii, J., et al. (2002). Tuning support vector machines for biomedical named entity recognition. In *Proceedings of the ACL-02 workshop on Natural language processing in the biomedical domain*, pages 1–8.

[Mesquita et al., 2013] Mesquita, F., Schmidek, J., and Barbosa, D. (2013). Effectiveness and efficiency of open relation extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 447–457.

[Mikolov et al., 2013a] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[Mikolov et al., 2013b] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

[Milošević and Thielemann, 2023] Milošević, N. and Thielemann, W. (2023). Comparison of biomedical relationship extraction methods and models for knowledge graph creation. *Journal of Web Semantics*, 75:100756.

[Mitrofan and Ion, 2017] Mitrofan, M. and Ion, R. (2017). Adapting the ttl romanian pos tagger to the biomedical domain. In *BiomedicalNLP@ RANLP*, pages 8–14.

[Miwa and Ananiadou, 2013] Miwa, M. and Ananiadou, S. (2013). Nactem eventmine for bionlp 2013 cg and pc tasks. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 94–98.

[Munkhdalai et al., 2015] Munkhdalai, T., Li, M., Batsuren, K., Park, H. A., Choi, N. H., and Ryu, K. H. (2015). Incorporating domain knowledge in chemical and biomedical named entity recognition with word representations. *Journal of cheminformatics*, 7:1–8.

[Naderi et al., 2011] Naderi, N., Kappler, T., Baker, C. J., and Witte, R. (2011). Organismtagger: detection, normalization and grounding of organism entities in biomedical documents. *Bioinformatics*, 27(19):2721–2729.

[Nédellec et al., 2013] Nédellec, C., Bossy, R., Kim, J.-D., Kim, J.-J., Ohta, T., Pyysalo, S., and Zweigenbaum, P. (2013). Overview of bionlp shared task 2013. In *Proceedings of the BioNLP shared task 2013 workshop*, pages 1–7.

[Nejadgholi et al., 2020] Nejadgholi, I., Fraser, K. C., and De Bruijn, B. (2020). Extensive error analysis and a learning-based evaluation of medical entity recognition systems to approximate user experience. *arXiv preprint arXiv:2006.05281*.

[Nguyen et al., 2017] Nguyen, D. Q., Nguyen, T. D., Nguyen, D. Q., and Phung, D. (2017). A novel embedding model for knowledge base completion based on convolutional neural network. *arXiv preprint arXiv:1712.02121*.

[Nguyen and Grishman, 2018] Nguyen, T. and Grishman, R. (2018). Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

[Nguyen et al., 2016] Nguyen, T. H., Cho, K., and Grishman, R. (2016). Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309.

[Nguyen and Grishman, 2015] Nguyen, T. H. and Grishman, R. (2015). Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 365–371.

[Nicholson and Greene, 2020] Nicholson, D. N. and Greene, C. S. (2020). Constructing knowledge graphs and their biomedical applications. *Computational and structural biotechnology journal*, 18:1414–1428.

[Nickel et al., 2016] Nickel, M., Rosasco, L., and Poggio, T. (2016). Holographic embeddings of knowledge graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

[Nickel et al., 2011] Nickel, M., Tresp, V., Kriegel, H.-P., et al. (2011). A three-way model for collective learning on multi-relational data. In *Icml*, volume 11, pages 3104482–3104584.

[Nobata et al., 1999] Nobata, C., Collier, N., and Tsujii, J.-i. (1999). Automatic term identification and classification in biology texts. In *Proc. of the 5th NLPRS*, pages 369–374.

[Ohta et al., 2011] Ohta, T., Pyysalo, S., and Tsujii, J. (2011). Overview of the epigenetics and post-translational modifications (epi) task of bionlp shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 16–25.

[Özgür et al., 2008] Özgür, A., Vu, T., Erkan, G., and Radev, D. R. (2008). Identifying gene-disease associations using centrality on a literature mined gene-interaction network. *Bioinformatics*, 24(13):i277–i285.

[Pan and Yang, 2010] Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

[Papanikolaou and Pierleoni, 2020] Papanikolaou, Y. and Pierleoni, A. (2020). Dare: Data augmented relation extraction with gpt-2. *arXiv preprint arXiv:2004.13845*.

[Peng et al., 2019] Peng, Y., Yan, S., and Lu, Z. (2019). Transfer learning in biomedical natural language processing: an evaluation of bert and elmo on ten benchmarking datasets. *arXiv preprint arXiv:1906.05474*.

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

[Perera et al., 2020] Perera, N., Dehmer, M., and Emmert-Streib, F. (2020). Named entity recognition and relation detection for biomedical information extraction. *Frontiers in cell and developmental biology*, page 673.

[Peters et al., 2018] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

[Petrov et al., 2011] Petrov, S., Das, D., and McDonald, R. (2011). A universal part-of-speech tagset. *arXiv preprint arXiv:1104.2086*.

[Poerner et al., 2020] Poerner, N., Waltinger, U., and Schütze, H. (2020). Inexpensive domain adaptation of pretrained language models: Case studies on biomedical ner and covid-19 qa. *arXiv preprint arXiv:2004.03354*.

[Portelli et al., 2021] Portelli, B., Lenzi, E., Chersoni, E., Serra, G., and Santus, E. (2021). Bert prescriptions to avoid unwanted headaches: A comparison of transformer architectures for adverse drug event detection. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1740–1747.

[Pruksachatkun et al., 2020] Pruksachatkun, Y., Phang, J., Liu, H., Htut, P. M., Zhang, X., Pang, R. Y., Vania, C., Kann, K., and Bowman, S. R. (2020). Intermediate-task transfer learning with pretrained models for natural language understanding: When and why does it work? *arXiv preprint arXiv:2005.00628*.

[Pyysalo et al., 2012] Pyysalo, S., Ohta, T., Miwa, M., Cho, H.-C., Tsujii, J., and Ananiadou, S. (2012). Event extraction across multiple levels of biological organization. *Bioinformatics*, 28(18):i575–i581.

[Pyysalo et al., 2015] Pyysalo, S., Ohta, T., Rak, R., Rowley, A., Chun, H.-W., Jung, S.-J., Choi, S.-P., Tsujii, J., and Ananiadou, S. (2015). Overview of the cancer genetics and pathway curation tasks of bionlp shared task 2013. *BMC bioinformatics*, 16(10):1–19.

[Pyysalo et al., 2011] Pyysalo, S., Ohta, T., Rak, R., Sullivan, D., Mao, C., Wang, C., Sobral, B., Tsujii, J., and Ananiadou, S. (2011). Overview of the infectious diseases (id) task of bionlp shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 26–35.

[Quan et al., 2016] Quan, C., Hua, L., Sun, X., and Bai, W. (2016). Multichannel convolutional neural network for biological relation extraction. *BioMed research international*, 2016.

[Radford et al., 2018] Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training.

[Radford et al., 2019] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

[Rahul et al., 2017] Rahul, P. V., Sahu, S. K., and Anand, A. (2017). Biomedical event trigger identification using bidirectional recurrent neural network based models. *arXiv preprint arXiv:1705.09516*.

[Ramponi et al., 2020] Ramponi, A., van der Goot, R., Lombardo, R., and Plank, B. (2020). Biomedical event extraction as sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5357–5367.

[Raza and Schwartz, 2023] Raza, S. and Schwartz, B. (2023). Entity and relation extraction from clinical case reports of covid-19: a natural language processing approach. *BMC Medical Informatics and Decision Making*, 23(1):20.

[Rebholz-Schuhmann, 2013] Rebholz-Schuhmann, D. (2013). Biomedical named entity recognition, whatizit. *Encyclopedia of Systems Biology*, pages 132–134.

[Rocktäschel et al., 2012] Rocktäschel, T., Weidlich, M., and Leser, U. (2012). Chemspot: a hybrid system for chemical named entity recognition. *Bioinformatics*, 28(12):1633–1640.

[Roy and Pan, 2021] Roy, A. and Pan, S. (2021). Incorporating medical knowledge in bert for clinical relation extraction. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 5357–5366.

[Sabbir et al., 2017] Sabbir, A., Jimeno-Yepes, A., and Kavuluru, R. (2017). Knowledge-based biomedical word sense disambiguation with neural concept embeddings. In *2017 IEEE 17th International Conference on Bioinformatics and Bioengineering (BIBE)*, pages 163–170. IEEE.

[Sahlgren, 2006] Sahlgren, M. (2006). *The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. PhD thesis, Institutionen för lingvistik.

[Sahu et al., 2019] Sahu, S. K., Christopoulou, F., Miwa, M., and Ananiadou, S. (2019). Inter-sentence relation extraction with document-level graph convolutional neural network. *arXiv preprint arXiv:1906.04684*.

[Sastre et al., 2020] Sastre, J., Zaman, F., Duggan, N., McDonagh, C., and Walsh, P. (2020). A deep learning knowledge graph approach to drug labelling. In *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 2513–2521. IEEE.

[Scaboro et al., 2023] Scaboro, S., Portelli, B., Chersoni, E., Santus, E., and Serra, G. (2023). Extensive evaluation of transformer-based architectures for adverse drug events extraction. *Knowledge-Based Systems*, page 110675.

[Shardlow et al., 2018] Shardlow, M., Nguyen, N., Owen, G., O'Donovan, C., Leach, A., McNaught, J., Turner, S., and Ananiadou, S. (2018). A new corpus to support text mining for the curation of metabolites in the chebi database. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

[Shen et al., 2019] Shen, C., Lin, H., Fan, X., Chu, Y., Yang, Z., Wang, J., and Zhang, S. (2019). Biomedical event trigger detection with convolutional highway neural network and extreme learning machine. *Applied Soft Computing*, 84:105661.

[Shen et al., 2003] Shen, D., Zhang, J., Zhou, G., Su, J., and Tan, C. L. (2003). Effective adaptation of hidden markov model-based named entity recognizer for biomedical domain. In *Proceedings of the ACL 2003 workshop on Natural language processing in biomedicine*, pages 49–56.

[Shen et al., 2017] Shen, Z., Zhang, Y.-H., Han, K., Nandi, A. K., Honig, B., Huang, D.-S., et al. (2017). mirna-disease association prediction with collaborative matrix factorization. *Complexity*, 2017.

[Socher et al., 2013] Socher, R., Chen, D., Manning, C. D., and Ng, A. (2013). Reasoning with neural tensor networks for knowledge base completion. *Advances in neural information processing systems*, 26.

[Soomro et al., 2017] Soomro, P. D., Kumar, S., Shaikh, A. A., Raj, H., et al. (2017). Bio-ner: biomedical named entity recognition using rule-based and statistical learners. *International Journal of Advanced Computer Science and Applications*, 8(12).

[Sorber et al., 2013] Sorber, L., Van Barel, M., and De Lathauwer, L. (2013). Optimization-based algorithms for tensor decompositions: Canonical polyadic decomposition, decomposition in rank-(l_r,l_r,1) terms, and a new generalization. *SIAM Journal on Optimization*, 23(2):695–720.

[Su et al., 2022] Su, F., Zhang, Y., Li, F., and Ji, D. (2022). Balancing precision and recall for neural biomedical event extraction. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:1637–1649.

[Symeonidou et al., 2019] Symeonidou, A., Sazonau, V., and Groth, P. (2019). Transfer learning for biomedical named entity recognition with biobert. In *SEMANTICS Posters&Demos*.

[Tang et al., 2014] Tang, B., Cao, H., Wang, X., Chen, Q., and Xu, H. (2014). Evaluating word representation features in biomedical named entity recognition tasks. *BioMed research international*, 2014.

[Trieu et al., 2020] Trieu, H.-L., Tran, T. T., Duong, K. N., Nguyen, A., Miwa, M., and Ananiadou, S. (2020). Deepeventmine: end-to-end neural nested event extraction from biomedical texts. *Bioinformatics*, 36(19):4910–4917.

[Trouillon et al., 2016] Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., and Bouchard, G. (2016). Complex embeddings for simple link prediction. In *International conference on machine learning*, pages 2071–2080. PMLR.

[Turian et al., 2010] Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394.

[Van Mulligen et al., 2012] Van Mulligen, E. M., Fourrier-Reglat, A., Gurwitz, D., Molokhia, M., Nieto, A., Trifiro, G., Kors, J. A., and Furlong, L. I. (2012). The eu-adr corpus: annotated drugs, diseases, targets, and their relationships. *Journal of biomedical informatics*, 45(5):879–884.

[Vasilakes et al., 2023] Vasilakes, J., Georgiadis, P., Nguyen, N. T., Miwa, M., and Ananiadou, S. (2023). Contextualized medication event extraction with levitated markers. *Journal of Biomedical Informatics*, 141:104347.

[Wada et al., 2020] Wada, S., Takeda, T., Manabe, S., Konishi, S., Kamohara, J., and Matsumura, Y. (2020). Pre-training technique to localize medical bert and enhance biomedical bert. *arXiv preprint arXiv:2005.07202*.

[Wang et al., 2017a] Wang, A., Wang, J., Lin, H., Zhang, J., Yang, Z., and Xu, K. (2017a). A multiple distributed representation method based on neural network for biomedical event extraction. *BMC medical informatics and decision making*, 17:59–66.

[Wang et al., 2021] Wang, M., Qiu, L., and Wang, X. (2021). A survey on knowledge graph embeddings for link prediction. *Symmetry*, 13(3):485.

[Wang et al., 2020a] Wang, Q., Li, M., Wang, X., Parulian, N., Han, G., Ma, J., Tu, J., Lin, Y., Zhang, H., Liu, W., et al. (2020a). Covid-19 literature knowledge graph construction and drug repurposing report generation. *arXiv preprint arXiv:2007.00576*.

[Wang et al., 2017b] Wang, Q., Mao, Z., Wang, B., and Guo, L. (2017b). Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743.

[Wang et al., 2020b] Wang, S., Zhou, W., and Jiang, C. (2020b). A survey of word embeddings based on deep learning. *Computing*, 102:717–740.

[Wang et al., 2019] Wang, X., Zhang, Y., Ren, X., Zhang, Y., Zitnik, M., Shang, J., Langlotz, C., and Han, J. (2019). Cross-type biomedical named entity recognition with deep multi-task learning. *Bioinformatics*, 35(10):1745–1752.

[Wang et al., 2022] Wang, X. D., Leser, U., and Weber, L. (2022). Beeds: Large-scale biomedical event extraction using distant supervision and question answering. In *Proceedings of the 21st Workshop on Biomedical Language Processing*, pages 298–309.

[Wang et al., 2020c] Wang, X. D., Weber, L., and Leser, U. (2020c). Biomedical event extraction as multi-turn question answering. In *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis*, pages 88–96.

[Wang et al., 2018] Wang, Y., Wang, J., Lin, H., Tang, X., Zhang, S., and Li, L. (2018). Bidirectional long short-term memory with crf for detecting biomedical event trigger in fasttext semantic space. *BMC bioinformatics*, 19:59–66.

[Wang et al., 2014] Wang, Z., Zhang, J., Feng, J., and Chen, Z. (2014). Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI conference on artificial intelligence*, volume 28.

[Weber et al., 2020] Weber, L., Münchmeyer, J., Rocktäschel, T., Habibi, M., and Leser, U. (2020). Huner: improving biomedical ner with pretraining. *Bioinformatics*, 36(1):295–302.

[Webson and Pavlick, 2021] Webson, A. and Pavlick, E. (2021). Do prompt-based models really understand the meaning of their prompts? *arXiv preprint arXiv:2109.01247*.

[Wei et al., 2012] Wei, C.-H., Kao, H.-Y., and Lu, Z. (2012). Sr4gn: a species recognition software tool for gene normalization. *PloS one*, 7(6):e38460.

[Wei and Zou, 2019] Wei, J. and Zou, K. (2019). Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

[Xiang and Wang, 2019] Xiang, W. and Wang, B. (2019). A survey of event extraction from text. *IEEE Access*, 7:173111–173137.

[Xiao et al., 2015] Xiao, H., Huang, M., Hao, Y., and Zhu, X. (2015). Transa: An adaptive approach for knowledge graph embedding. *arXiv preprint arXiv:1509.05490*.

[Yan et al., 2019] Yan, H., Jin, X., Meng, X., Guo, J., and Cheng, X. (2019). Event detection with multi-order graph convolution and aggregated attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5766–5770.

[Yang et al., 2014] Yang, B., Yih, W.-t., He, X., Gao, J., and Deng, L. (2014). Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*.

[Yang et al., 2018] Yang, K., Wang, N., Liu, G., Wang, R., Yu, J., Zhang, R., Chen, J., and Zhou, X. (2018). Heterogeneous network embedding for identifying symptom candidate genes. *Journal of the American Medical Informatics Association*, 25(11):1452–1459.

[Yang et al., 2020] Yang, S., Yoo, S., and Jeong, O. (2020). Denert-kg: Named entity and relation extraction model using dqn, knowledge graph, and bert. *Applied Sciences*, 10(18):6429.

[Yoon et al., 2019] Yoon, W., So, C. H., Lee, J., and Kang, J. (2019). Collabonet: collaboration of deep neural networks for biomedical named entity recognition. *BMC bioinformatics*, 20(10):55–65.

[Zerva and Ananiadou, 2015] Zerva, C. and Ananiadou, S. (2015). Event extraction in pieces: Tackling the partial event identification problem on unseen corpora. In *Proceedings of BioNLP 15*, pages 31–41.

[Zhang and Elhadad, 2013] Zhang, S. and Elhadad, N. (2013). Unsupervised biomedical named entity recognition: Experiments with clinical and biological texts. *Journal of biomedical informatics*, 46(6):1088–1098.

[Zhang et al., 2019] Zhang, Y., Lin, H., Yang, Z., Wang, J., Sun, Y., Xu, B., and Zhao, Z. (2019). Neural network-based approaches for biomedical relation classification: a review. *Journal of biomedical informatics*, 99:103294.

[Zhang et al., 2018] Zhang, Y., Zheng, W., Lin, H., Wang, J., Yang, Z., and Dumontier, M. (2018). Drug–drug interaction extraction via hierarchical rnns on sequence and shortest dependency paths. *Bioinformatics*, 34(5):828–835.

[Zhao et al., 2019] Zhao, D., Wang, J., Lin, H., Yang, Z., and Zhang, Y. (2019). Extracting drug–drug interactions with hybrid bidirectional gated recurrent unit and graph convolutional network. *Journal of Biomedical Informatics*, 99:103295.

[Zhao et al., 2021] Zhao, S., Su, C., Lu, Z., and Wang, F. (2021). Recent advances in biomedical literature mining. *Briefings in Bioinformatics*, 22(3):bbaa057.

[Zhao et al., 2018] Zhao, Y., Jin, X., Wang, Y., and Cheng, X. (2018). Document embedding enhanced event detection with hierarchical and supervised attention. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 414–419.

[Zheng et al., 2018] Zheng, W., Lin, H., Li, Z., Liu, X., Li, Z., Xu, B., Zhang, Y., Yang, Z., and Wang, J. (2018). An effective neural model extracting document level chemical-induced disease relations from biomedical literature. *Journal of biomedical informatics*, 83:1–9.

[Zhu et al., 2018] Zhu, Q., Li, X., Conesa, A., and Pereira, C. (2018). Gram-cnn: a deep learning approach with local context for named entity recognition in biomedical text. *Bioinformatics*, 34(9):1547–1554.

[Zhu et al., 2015] Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. (2015). Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *The IEEE International Conference on Computer Vision (ICCV)*.

[Zong et al., 2019] Zong, N., Wong, R. S. N., Ngo, V., Yu, Y., and Li, N. (2019). Scalable and accurate drug–target prediction based on heterogeneous bio-linked network mining. *bioRxiv*, page 539643.