KNOWLEDGE EXTRACTION FROM BIOMEDICAL LITERATURE WITH SYMBOLIC AND DEEP TRANSFER LEARNING METHODS

Alan Ramponi

A thesis submitted for the degree of
Doctor of Philosophy

Department of Information Engineering and Computer Science
IECS Doctoral School
University of Trento

June 30, 2021

Supervisors

Prof. Enrico Blanzieri
Dept. of Information Engineering and Computer Science, University of Trento

Dr. Rosario Lombardo
Fondazione The Microsoft Research – University of Trento COSBI

Prof. Corrado Priami
Dept. of Computer Science, University of Pisa
Fondazione The Microsoft Research – University of Trento COSBI

Final Examination Committee

Prof. Davide Bacciu
Dept. of Computer Science, University of Pisa

Prof. Andrea Passerini
Dept. of Information Engineering and Computer Science, University of Trento

Dr. Sumithra Velupillai
Institute of Psychiatry, Psychology and Neuroscience, King's College London

The work presented in this thesis has been funded by:

Fondazione The Microsoft Research – University of Trento
Centre for Computational and Systems Biology (COSBI)
Rovereto, Italy

Abstract

The available body of biomedical literature is increasing at a high pace, exceeding the ability of researchers to promptly leverage this knowledge-rich amount of information. Although the outstanding progress in natural language processing (NLP) we observed in the past few years, current technological advances in the field mainly concern newswire and web texts, and do not directly translate in good performance on highly specialized domains such as biomedicine due to linguistic variations along surface, syntax and semantic levels. Given the advances in NLP and the challenges the biomedical domain exhibits, and the explosive growth of biomedical knowledge being currently published, in this thesis we contribute to the biomedical NLP field by providing efficient means for extracting semantic relational information from biomedical literature texts. To this end, we made the following contributions towards the real-world adoption of knowledge extraction methods to support biomedicine: (i) we propose a symbolic high-precision biomedical relation extraction approach to reduce the time-consuming manual curation efforts of extracted relational evidence (Chapter 3), (ii) we conduct a thorough cross-domain study to quantify the drop in performance of deep learning methods for biomedical edge detection shedding lights on the importance of linguistic varieties in biomedicine (Chapter 4), and (iii) we propose a fast and accurate end-to-end solution for biomedical event extraction, leveraging sequential transfer learning and multi-task learning, making it a viable approach for real-world large-scale scenarios (Chapter 5). We then outline the conclusions by highlighting challenges and providing future research directions in the field.

Acknowledgments

I would first like to thank my Ph.D. supervisors Rosario Lombardo and Enrico Blanzieri for their guidance and support during this long journey. I am extremely thankful to them for sharing expertise, and sincere and valuable encouragement extended to me. Also, I would like to express my gratitude to Corrado Priami and Enrico Domenici for the opportunity they gave me to conduct my research at Fondazione Microsoft Research – The University of Trento Centre for Computational and Systems Biology, and to the reviewers, Sumithra Velupillai and Davide Bacciu, for their thoughtful and constructive comments towards improving my thesis.

I am also grateful to Barbara Plank for being an awesome mentor during my research stay at the IT University of Copenhagen and for helping me grow. I extend my special thanks to the wonderful people of the NLPnorth research group, especially Rob and Marija. My journey was much easier with you!

Further, I would like to thank all my friends, particularly Riccardo, Daniele and Enrico. It doesn't matter how far we are, there are so many things and amazing experiences that bind us together! A very profound gratitude goes to Francesca, my mother, Giovanni and Samuel, my brothers, and Maggie, my sister. A big thank you to Camilla for providing me with unfailing support and continuous encouragement throughout these years and for listening to me while rambling about research. Last but not least, I would like to thank my father. You are a stone, and stones live forever.

Alan

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Processing and understanding natural language is a central feature of human intelligence and therefore a main challenge towards the goal of artificial general intelligence (Eisenstein, 2019). Giving a machine the ability to perform human cognitive tasks such as tackling the complexity and ambiguity of natural languages is a long-standing problem and the main research focus of Natural Language Processing (NLP), a field at the intersection of linguistics and computer science. In the era of data, unstructured textual information pertaining to a variety of domains is produced at a very high rate, resulting in a large amount of potentially useful information to dig in. Biomedicine is no exception, in which the number of research publications indexed on bibliographic databases such as PubMed (Canese & Weis, 2013) is increasing steeply (Figure 1.1).

The availability of written biomedical knowledge and the fast progress in NLP have opened up the challenge and the opportunity to extract structured relational information from unstructured biomedical texts, in order to aid translating the research evidence into practice. Biomedical Natural Language Processing (BioNLP), a subfield of NLP in the biomedical application domain, has indeed been shown to be crucial for downstream bioinformatics applications such as the population of knowledge bases and the construction of biochemical pathways (Ananiadou et al., 2010; Li et al., 2019). Despite the attractive opportunity, NLP in the biomedical domain is lagging behind the "general

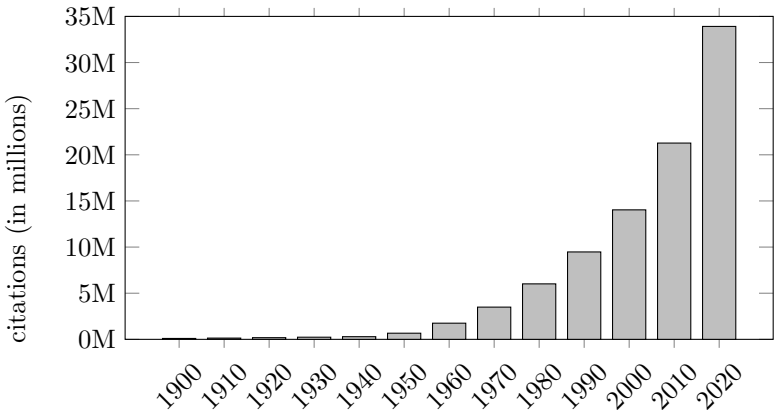


Figure 1.1: Cumulative number of citations indexed on PubMed over years.

domain" NLP. The main reason being that research in natural language processing has traditionally focused on newswire text (Cohen & Demner-Fushman, 2014), implicitly setting it as a standard, *canonical language*. This historical coincidence is mainly due to the early availability of resources in the newswire domain (Plank, 2016), that indirectly slowed down the progress in NLP for *non-canonical linguistic varieties*, including but not limited to biomedicine, legal texts, and social media.

Newswire text has indeed a simpler textual structure compared to biomedical text (Cohen & Demner-Fushman, 2014). For instance, it has been shown that texts in the newswire domain have an average sentence length of 19.1 words, whereas in biomedicine sentences are up to 70.7% longer (i.e., 24.5–27 words) (Lippincott et al., 2010). Further, the vocabulary overlap between newswire and biomedicine is only around 24.9% (Gururangan et al., 2020). Besides lexical and surface characteristics, biomedicine exhibits different syntactic, discourse and sentential features, which vary even within the broad biomedical domain itself (Lippincott et al., 2010). Particularly, scientific publications present additional challenges regarding the text types, namely abstracts and full-texts, the latter being harder to process due to the differences in structural and content aspects (Cohen et al., 2010). Biomedical texts are

thus by nature more challenging to deal with compared to newswire texts, due to both the linguistic variations along surface, syntax, and semantics levels (Cohen & Demner-Fushman, 2014), and the implicit linguistic bias of existing solutions which indirectly harms their repurposing to the biomedical domain (D. Q. Nguyen & Verspoor, 2019).

Early approaches to NLP – and thus, to BioNLP – have been mostly dominated by symbolic methods, namely the manipulation of linguistic information units by means of human hand-coded rules and patterns, to the goal of capturing the meaning of texts automatically (Henderson, 2020; Bender & Koller, 2020). The increase in computational power and the need for soft probabilistic classification in the late 1980s originated a shift towards statistical methods. As a result, in the past 20 years we have witnessed a first revolution, with statistical NLP that has become the dominant paradigm (Manning & Schutze, 1999). Statistical approaches such as machine learning have been commonplace since then due to their intrinsic ability to learn patterns from data in an automated fashion, dispensing the manual crafting of rules and patterns in favour of feature engineering (Mitchell, 1997).

A second revolution in NLP, or "tsunami" (Manning, 2015), dates back a few years ago when deep learning, a particular branch of machine learning, dramatically improved the performance across a variety of NLP tasks (LeCun et al., 2015). The breakthroughs in representation learning for NLP, such as word embeddings (Mikolov, Sutskever, et al., 2013; Mikolov, Chen, et al., 2013), have further strengthened this shift, showing that deep neural network approaches with dense word representations are effective and more robust across linguistic variations compared to former approaches (T. H. Nguyen & Grishman, 2015).

Although successful, deep learning methods require a large amount of labeled data for the specific task and domain at hand, as well as costly computational resources in order to train a reliable model. Transfer learning has recently emerged to mitigate these issues in NLP, providing an effective paradigm to repurpose models trained on a task or domain on a related task or domain (Pan & Yang, 2010; Ruder, 2019). Specifically, language models pre-trained on large collections of unlabeled texts such as BERT (Devlin et al., 2019) have shown to lessen the amount of data required to reach the same performance on a target

task (Howard & Ruder, 2018), while increasing the robustness on unseen texts belonging to different data distributions (Hendrycks et al., 2020). As a result, transfer learning can be referred to as the third revolution in NLP. The recent progress in NLP is in fact mostly ascribable to transfer learning (Ruder et al., 2019), which has led to unprecedented, near to human-level performance on a wide array of NLP tasks (A. Wang, Pruksachatkun, et al., 2019).

Given the advances in the general domain NLP and the challenges the biomedical domain exhibits, in this thesis we contribute to the BioNLP field by proposing effective methods for extracting semantic relational knowledge from the unstructured biomedical literature, in order to ultimately assist researchers in keeping pace with the growing volume of domain-relevant texts being published. Specifically, we design solutions for the tasks of biomedical relation extraction and biomedical event extraction, driving the progress in BioNLP as well as providing relevant insights to the broader NLP community.

Research Goal and Objectives

The focus of this thesis is on providing efficient means for extracting semantic relational information of domain interest from the biomedical literature. We firstly explore traditional symbolic approaches and the conceptually simpler relation extraction task, then focusing on the more complex event extraction task, thus employing deep and transfer learning methods. More specifically, in this thesis we attempt to answer the following research questions:

> **RQ**$_1$ Does a symbolic approach for biomedical relation extraction aid in mitigating the subsequent manual curation efforts by achieving a higher precision score compared to deep learning approaches? How does it compare to recent transfer learning methods?

In order to answer this question, we devise a symbolic approach that uses carefully designed patterns and rules to leverage surface linguistic information and syntactic dependency tree structures of the input texts. We compare it to state-of-the-art methods in literature as well as to transformer-based methods which we specifically fine-tune on the task. We also conduct a detailed error

analysis to investigate the limitations of the dependency tree-driven approach, highlighting interesting challenges and opening future directions.

> **RQ**$_2$ To which degree are deep learning methods robust in handling different linguistic varieties for the edge detection sub-task of biomedical event extraction? What is the expected drop in performance when these methods are applied out-of-domain?

We employ convolutional neural networks as the most successful deep learning approach for the task to study the cross-domain performance of edge detection on diverse corpora. We compare in-domain and out-of-domain performance of models, shedding light on the importance of domain variations to be tackled in future work, and releasing data to encourage research in this direction. We also assess the contribution of different syntactic and semantic input embeddings, providing results using multiple evaluation strategies.

> **RQ**$_3$ How can biomedical event extraction be tackled in an end-to-end fashion in order to improve the performance on the task compared to previous pipeline-based and joint learning alternatives? How does it compare to previous work in terms of speed efficiency?

We propose a joint, end-to-end solution for the highly complex task of event extraction by introducing a novel linearization approach to recast event structures into word-level labels, and leveraging deep transfer learning methods. We experiment with single task and diverse multi-task learning alternatives as well as multi-label decoding. We compare our approach to state-of-the-art methods in terms of both performance and speed, also providing a thorough error analysis and investigating the contribution of model components by performing ablation studies. Results are insightful and open directions in BioNLP and the broader NLP field.

From a practical perspective, we make the following contributions:

- We propose a high-precision biomedical relation extraction approach to reduce human curation efforts of the extracted information (Chapter 3);

- We provide state-of-the-art performance in biomedical relation extraction using either symbolic or transfer learning methods (Chapter 3);
- We quantify the cross-domain generalization of current deep learning for edge detection in biomedical event extraction (Chapter 4);
- We release standardized data to encourage future research in cross-domain biomedical edge detection for event extraction pipelines (Chapter 4);
- We propose a novel linearization approach and a end-to-end multi-task multi-label model for the biomedical event extraction task (Chapter 5);
- We provide insights on the importance of multi-task learning and multi-label decoding, as well as first results using non-gold entities (Chapter 5);
- We provide an efficient solution for biomedical event extraction that is both faster and more accurate compared to previous work (Chapter 5).

Thesis Outline

The organization of this thesis is schematically presented in Figure 1.2. After this introduction, the following chapters are presented:

- In Chapter 2, an overview of the concepts and methods relevant to the contents of this thesis is provided. We present fundamentals on natural language processing and relevant tasks, with particular emphasis on the biomedical domain. We then introduce symbolic and neural network methods, with details on layers and components we employ throughout this dissertation. Finally, we discuss transfer learning concepts and the notion of domain, providing details on sequential transfer learning methods such as BERT and the multi-task learning paradigm. Sections 2.2.4 and 2.2.4.1 of this Chapter are based on Ramponi and Plank (2020).
- In Chapter 3, a dependency tree-driven symbolic approach for high-precision biomedical relation extraction is presented. Specifically, we show that (i) carefully designed dependency tree rules provide the highest

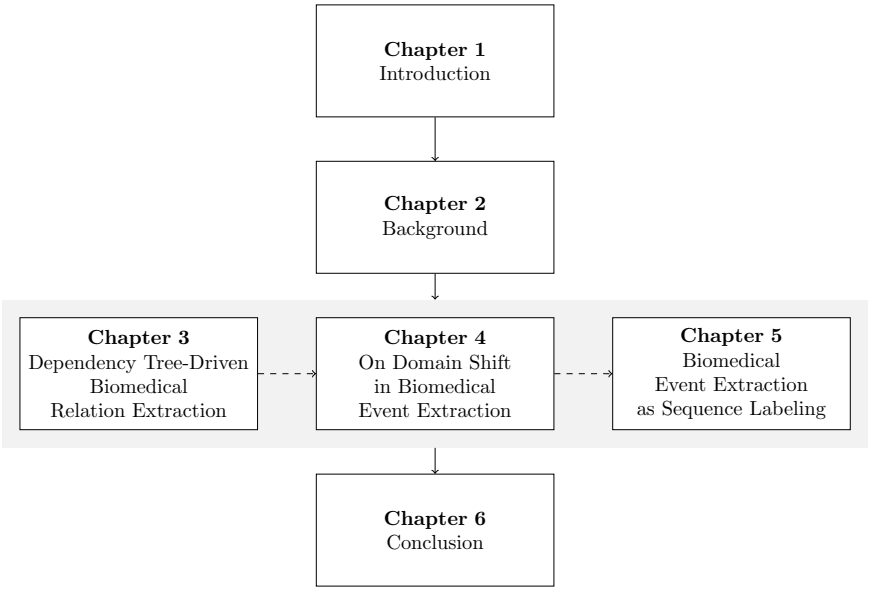


Figure 1.2: How to read this thesis: while we suggest reading Chapters in the order they appear, Chapters 3, 4 and 5 can be optionally read after Chapter 2.

precision compared to previous methods on the task, ultimately reducing the human curation efforts of the extracted information, (ii) fine-tuning transformer-based language models on the task provides higher performance than traditional machine learning approaches, and (iii) besides the good results, there are cases that a symbolic approach still does not capture due to the high variability of biomedical texts, and the relation extraction task itself cannot model potentially useful high-order associations. As a result, this study sets the motivation on focusing on deep and transfer learning solutions afterwards, for the semantically richer event extraction task. This Chapter is based on Ramponi, Giampiccolo, Tomasoni, Priami, and Lombardo (2020).

- In Chapter 4, a study on the cross-domain performance in biomedical event extraction of typical deep learning solutions is presented. As a case study, we experiment with the most challenging step of the biomedical

event extraction pipeline, namely edge detection, employing deep convolutional neural networks. We provide (i) the first cross-domain study of biomedical edge detection, quantifying the out-of-domain drop in performance of in-domain models, thus highlighting the importance of the linguistic varieties to be tackled in future work, (ii) a standardized cross-domain corpus we released freely to encourage future research in cross-domain edge detection for event extraction pipelines, and (iii) insights about how different syntactic and semantic input embeddings contribute to the performance. This study provides the motivation to abandon pipelined models in favour to end-to-end biomedical event extraction, to avoid cascading errors and to leverage inter-dependencies among subtasks. This Chapter is based on Ramponi, Plank, and Lombardo (2020).

- In Chapter 5, we study how to tackle biomedical event extraction as end-to-end solution using deep transfer learning in a fast and efficient way. We propose Biomedical Event Extraction as Sequence Labeling (BeeSL), whose main contributions are (i) a novel approach to linearize the event structures into word-level labels, thus to model the biomedical event extraction stages jointly and leverage subtask inter-dependencies, while mitigating the error cascading shortcoming of locally-optimized classifier pipelines, (ii) a multi-task, multi-head learning paradigm, which substantially reduces the label space for sequence labeling, while allowing to model multiple head relations for each word at once, (iii) state-of-the-art results on the standard GENIA event extraction benchmark, and an increase up to 5× in speed efficiency at inference time compared to the previous best solution, making it a viable solution for large-scale scenarios, and (iv) insights on the contribution of multi-task learning and multi-label decoding, as well as first results on the task without gold entity information, and a thorough error analysis. This Chapter is based on Ramponi, van der Goot, Lombardo, and Plank (2020).

Finally, Chapter 6 summarizes the main findings, outlining the conclusions, and presenting future directions in the field.

List of Publications

This dissertation is based on the following peer-reviewed publications:

1. **Ramponi, A.**, Giampiccolo, S., Tomasoni, D., Priami, C., and Lombardo, R. (2020). High-Precision Biomedical Relation Extraction for Reducing Human Curation Efforts in Industrial Applications. *IEEE Access*, 8, 150999–151011, doi: 10.1109/ACCESS.2020.3014862 `https://ieeexplore.ieee.org/document/9171821`.

2. **Ramponi, A.**, Plank, B., and Lombardo, R. (2020). Cross-Domain Evaluation of Edge Detection for Biomedical Event Extraction. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)* (pp. 1982–1989). Marseille, France: European Lang. Resources Association. `https://www.aclweb.org/anthology/2020.lrec-1.244`.

    - Formerly peer-reviewed and presented as a poster: **Ramponi, A.**, Plank, B., and Lombardo, R. (2019). On the Impact of Cross-Domain Edge Detection in Biomedical Event Extraction. *EurNLP Summit*. Facebook AI Research (FAIR), London, United Kingdom.

3. **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)* (pp. 5357–5367). Punta Cana, Dominican Republic (Online): Association for Computational Linguistics. `https://www.aclweb.org/anthology/2020.emnlp-main.431`.

4. **Ramponi, A.**, and Plank, B. (2020). Neural Unsupervised Domain Adaptation in NLP—A Survey. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)* (pp. 6838–6855). Barcelona, Spain (Online): International Comm. on Computational Linguistics. `https://www.aclweb.org/anthology/2020.coling-main.603`.

Chapter 2

Background

In this chapter we provide the fundamental background knowledge to understand the remainder of this dissertation. We start by introducing the broad goal of biomedical natural language processing, specifically focusing on relation and event extraction as the tasks of interest in this thesis. An introduction to traditional and neural network approaches in natural language processing is then provided, with details on layers, components and architectures we employ throughout this dissertation. Finally, we discuss relevant transfer learning concepts and introduce the notion of linguistic varieties (or domains), then focusing on sequential transfer learning approaches such as BERT and multi-task learning strategies. We conclude the chapter by providing key references to allow the reader to further deepen the presented topics.

2.1 Biomedical Natural Language Processing

Natural language processing (NLP) has shown tremendous advancements in the last decade, leading to unprecedented results in a wide array of tasks (A. Wang, Pruksachatkun, et al., 2019). Given the technical progress and the exponential growth of the biomedical literature in the form of electronic data resources (cf. Figure 1.1), there has recently been a surge of interest in NLP technology in order to extract structured information from the vast amount of unstructured biomedical texts. The study and development of NLP methods in the

biomedical application domain is typically referred to as biomedical natural language processing (BioNLP) (Cohen & Demner-Fushman, 2014).[1]

Although the recent progress in NLP is truly significant, current technological advances in the field have shown to not directly translate in better performance in BioNLP (Lee et al., 2020). Indeed, most research in natural language processing has traditionally focused on newswire and web texts, whose structural and content aspects are intrinsically different compared to biomedical texts (Cohen et al., 2010). Specifically, as opposed to newswire texts, biomedical texts exhibit different syntactic and sentential features (Lippincott et al., 2010), with only a very small percentage of word overlap with newswire texts due to the occurrence of highly-specialized terms (Gururangan et al., 2020) and substantially longer and more articulated statements (Lippincott et al., 2010; Cohen & Demner-Fushman, 2014). This indirectly harms the repurposing of existing solutions from the often termed *general* domain to the biomedical domain (D. Q. Nguyen & Verspoor, 2019).

As a running example, consider the text excerpts in Table 2.1. It appears immediately evident that biomedical text contents are drastically different and that a fine-grained description of biological processes (as the one in Table 2.1, below) will unlikely occur in newswire data, where on the other hand topics like money transfer operations find their natural place (Table 2.1, above). Besides the surface features of both the textual varieties (e.g., the approximately double length of the biomedical sentence compared to the newswire one, Table 2.1), biomedical statements are intrinsically more elaborated, exhibiting several coordinate and subordinate clauses. Further, most of the words are typically of domain relevance. Concretely, "rocG", "RocR", "NtrC", and "NifA" are highly-specialized biomedical terms that happen to be out-of-vocabulary in the general domain, and thus are mostly neglected by NLP methods – either rule-based or data-driven – which rely on newswire textual data. Further, some words belong to both the domain vocabularies, but they actually differ in their semantic content. For instance, the term "bank" predominantly refers to a

[1] In the context of this thesis, biomedical NLP entails biological entities and relationships from literature, such as protein-protein interactions and molecular biology reactions of genes or gene products (e.g., regulation, phosphorylation). We thus do not entail clinical NLP tasks concerning e.g., tests, treatments and medications from medical/health records.

Table 2.1: Excerpt examples of newswire text and biomedical text, from ACE corpus (Walker et al., 2006) and LLL corpus (Nédellec, 2005), respectively.

Newswire text
"*Citibank was involved in moving about $100 million for Raul Salinas de Gortari, brother of a former Mexican president, to banks in Switzerland.*"
Biomedical text
"*The rocG gene of Bacillus subtilis, encoding a catabolic glutamate dehydrogenase, is transcribed by SigL-containing RNA polymerase and requires for its expression RocR, a member of the NtrC/NifA family of proteins that bind to enhancer-like elements, called upstream activating sequences.*"

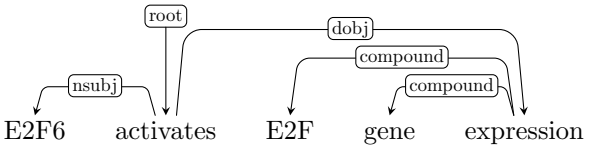
financial institution in newswire texts, whereas in biomedicine could indicate an environmental location, e.g., where particular bacterial species proliferate.

As a result, BioNLP as a field aims at accounting for both the challenges the biomedical domain exhibits, and the specifics about the biomedical task at hand. BioNLP being a strongly interdisciplinary field, it focuses in deriving means to allow biomedical researchers to better conduct their own research. Thus, a main goal in BioNLP is to design and develop methods capable of extracting relevant information from unstructured biomedical texts, in order to assist researchers in keeping pace with the increasing volume of domain-relevant texts being published. The main tasks to this purpose are *relation extraction* and *event extraction*. We provide details on these crucial tasks in Sections 2.1.1 and 2.1.2 after introducing core tasks and settings in NLP.

Core tasks in natural language processing Despite the domain differences, BioNLP shares the same fundamental tasks as the general domain NLP. In the following, we briefly describe the main preprocessing and syntactic tasks which are typically employed as building blocks in pipelined systems, e.g., to serve information for rule or feature design, providing examples in Table 2.2.

- **tokenization**: a common preprocessing task whose goal is to segment an input string into a sequence of linguistic units, called *tokens* (roughly words), in order to ease further processing;

Table 2.2: Typical tasks in natural language processing. The first row indicates an example of a raw text to be processed, whereas the following rows show the intended output for each of the tasks.

Task	**Example output**				
Raw text	*E2F6 activates E2F gene expression*				
Tokenization	E2F6	activates	E2F	gene	expression
Lemmatization	e2f6	activate	e2f	gene	expression
POS tagging	NN	VBZ	NN	NN	NN
Chunking	O	B-VP	O	B-NP	I-NP
Entity recognition	B-PRO	O	B-PRO	O	O
Dependency parsing					

- **lemmatization**: a task which aims at finding the *lemma* (i.e., the canonical form) of each token, so that inflected and derived forms of a word are processed consistently by further analysis components (e.g., "activate" = {"activates", "activated", "activate"}, "be" = {"is", "are", "be"});

- **part-of-speech tagging**: a task whose goal is to assign a descriptive syntactic category to each token in a sequence (e.g., verb, noun, adverb, adjective, etc.). Part-of-speech (POS) categories are typically drawn from tag inventories such as the coarse-grained Universal POS tag set (Nivre et al., 2016) and the fine-grained PENN Treebank POS tag set (Marcus et al., 1993) (refer to Appendix A for a detailed description of the tag sets). POS-tagged tokens are typically employed as central features to derive chunks or syntactic dependency parse trees (described below);

- **chunking**: a task which aims at identifying token segments in the input sequence that together constitute high-level syntactic units (e.g., ["gene", "expression"] $\mapsto$ ["gene expression"]). The output typically consists of

BIO-tagged[2] labels (i.e., B: begin, I: inside, O: outside) to distinguish between the first token and the other tokens of single chunks (e.g., B-NP refers to the first token of a noun phrase, and zero or more I-NP will follow it). Chunking is also typically referred to as *shallow parsing*;

- **entity recognition**: a task whose goal is to detect and assign a category to *entities* in an input sequence. As in chunking, BIO tags are typically employed to handle multi-token segments. Since entities differ based on the target application domain, there is no unified tag set. For instance, general domain entities are typically PERSON, ORGANIZATION and LOCATION, whereas in biomedicine these are, e.g., PROTEINS, as shown in Table 2.2 (B-PRO). Although named entity recognition (NER) is not a syntactic task per se, it is typically the input to extraction systems;

- **dependency parsing**: a task which aims to produce a grammatical dependency parse tree for an input sequence of tokens. The output structure encodes the relationships between tokens, and more specifically the grammatical dependency (in the form of a label) of each token to its *head* token (i.e., the parent in the tree). As for POS tagging, dependency labels are drawn from an inventory such as the ClearNLP tag set (refer to Appendix B for a detailed description of the tag set). Dependency parsing has been proven useful for information extraction tasks because dependencies loosely approximate semantic relationships (Ruder, 2019).

Deriving features for end tasks such as relation and event extraction (Sections 2.1.1 and 2.1.2) based on the output of some external syntactic modules in a *pipeline* fashion has been a de-facto standard in NLP (Tenney et al., 2019). We use syntactic modules for rule design in Chapter 3 and for providing features to a neural system in Chapter 4. With the recent advent of deep learning and specifically the availability of large, deeply contextualized pre-trained models (Chapter 2.2.4.2), *end-to-end* solutions have become commonplace due to their ability in adjusting low-level syntactic information to better capture high-level

[2] Alternatives to the BIO tagging scheme include IO (I: inside, O: outside) and BILOU (B: begin, I: inside, L: last, O: outside, U: unit) (see Jurafsky and Martin (2020) for details).

semantics (Tenney et al., 2019). We design an end-to-end solution for event extraction in Chapter 5, completely displacing external syntactic tools.

Problem settings Regardless of the technical approach that is employed (Section 2.2), natural language processing tasks can be categorized according to different methodological setups which reflect the specifics of the input and output relationships of the problem at hand. We can specifically distinguish:

- **text classification** (or *categorization*): the problem of categorizing a sequence of tokens based on its content. As a result, a label is assigned to describe the whole sequence. Example tasks include spam detection, topic identification, language detection, and sentiment classification;
- **sequence labeling** (or *tagging*): the problem of assigning a descriptive label to each of the tokens within an input sequence. POS tagging, chunking, and entity recognition in Table 2.2 are typical examples;
- **syntactic parsing**: the problem of deriving a structured output (e.g., in the form of a tree or graph) for an input sequence of tokens. Typical examples are dependency parsing in Table 2.2 and constituency parsing;
- **language modeling**: the problem of predicting the next token in a sequence given a sequence of previous tokens. It is typically employed for large language models pre-training (Chapter 2.2.4.2);
- **sequence to sequence** (or *seq2seq*): the problem of producing an output sequence of tokens from an input sequence of tokens. Specifically, seq2seq builds on top of language modeling. Examples include machine translation and text summarization.

While most of the NLP tasks have been historically framed into one of the aforementioned problem setups, this does not preclude reframing a task as a different problem. For instance, recent work has shown the effectiveness of reducing syntactic parsing tasks as tagging, specifically constituency (Gómez-Rodríguez & Vilares, 2018) and dependency parsing (Strzyz et al., 2019). We

also take an opposite but effective direction to current work in Chapter 5, where we cast biomedical event extraction as sequence labeling.

Given the background on core tasks in natural language processing and the problem settings, we now introduce the tasks we tackle in this thesis to the goal of extracting semantic relational information from the unstructured biomedical literature, namely relation (Section 2.1.1) and event extraction (Section 2.1.2).

2.1.1 Relation Extraction

Relation Extraction (RE) is the task of identifying the semantic relationships that hold between given entity mentions in an unstructured text. Specifically, the targeted information focus in biomedicine is in determining if an association exists between domain-specific entities such as PROTEINS, and the RE task is thus typically referred to as biomedical relation extraction. Due to the topic of this thesis, we hereafter use RE and biomedical RE interchangeably.

Formally, given an input sequence of tokens $S = [w_1, ..., w_N]$ and a set of given entity mentions $\{e_i\}_{i=1}^{E}$ where e_i is a token span in S, relation extraction aims at predicting a class $c_k \in C$ for each entity pair $(e_i, e_j)_{i \neq j}$. As a result, relation extraction is typically framed as a text classification task, where $\binom{|E|}{2}$ entity pairs, if $|E| \geq 2$, are independently evaluated and assigned a class $c_k \in C$.

For instance, given the sentence in Figure 2.1 in which $|E| = 3$ entity mentions are provided (i.e., the PROTEIN mentions "E2F6", "E2F" and "BDH1"), $\binom{|E|}{2} = 3$ candidate relation pairs are evaluated, namely ("E2F6","E2F"), ("E2F6","BDH1"), and ("E2F","BDH1") (Figure 2.1, dashed lines).

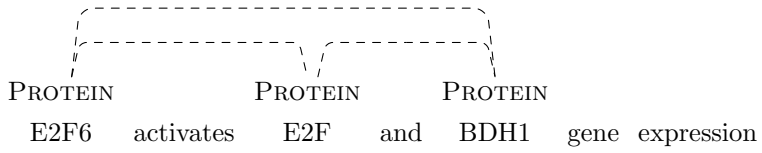


Figure 2.1: Example of entity pairs evaluation in biomedical relation extraction.

Specifically, in the relation extraction example in Figure 2.1, the candidate relations ("E2F6","E2F") and ("E2F6","BDH1") are positive relation instances (i.e., $c = 1$), whereas the pair ("E2F","BDH1") is not in a semantic

relationship (i.e., $c = 0$). Since biomedical relations of interest are often interactions, biomedical RE is typically framed as a binary problem in biomedicine. The main freely available annotated corpora for biomedical relation extraction are LLL (Nédellec, 2005), IEPA (Ding et al., 2001), and HPRD50 (Fundel et al., 2007). As opposed to early approaches which derive relations based on entity co-occurrences in the same token sequence (Cheng et al., 2008), annotated data for relation extraction aims at providing standardized means to encourage the development and evaluation of accurate information extraction methods. In this thesis we tackle biomedical relation extraction to the goal of providing high-precision results to biomedical practitioners, thus reducing the subsequent efforts in filtering spurious relation instances (Chapter 3).

Evaluation metrics In order to assess the goodness of relation extraction methods, a set of acknowledged metrics are typically employed. Given a set of human-annotated (i.e., *gold* standard) relation instances and the corresponding predictions, the *precision* (Eq. 2.1), *recall* (Eq. 2.2), and the harmonic mean of the two, referred to as the F_1 *score* (Eq. 2.3), are calculated as follows:

$$precision = \frac{TP}{TP + FP} \tag{2.1}$$

$$recall = \frac{TP}{TP + FN} \tag{2.2}$$

$$F_1\ score = \frac{2 * precision * recall}{precision + recall} \tag{2.3}$$

where TP is the number of *true positives* (i.e., the number of correctly identified instances), FP is the number of *false positives* (i.e., the number of wrongly identified instances), and FN is the number of *false negatives* (i.e., the number of missed instances). TN is also used to indicate the number of *true negatives* (i.e., the number of correctly identified negative instances).

Standard precision, recall and F_1 score metrics as defined in Eq. 2.1, 2.2 and 2.3 are typically the case of binary classification problems (i.e., when the output classes are two). When dealing with multi-class classification problems (i.e., when the output classes are more than two), the metrics are *macro-* or

micro-averaged over the classes. In the first case, metrics are computed at the instance-level, thus each instance contributes equally to the final score, regardless of the class it belongs. In the second case, per-class contributions are firstly aggregated and then averaged.

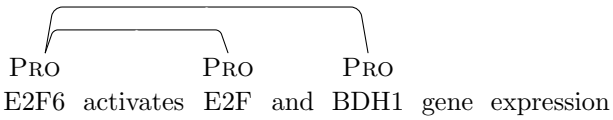
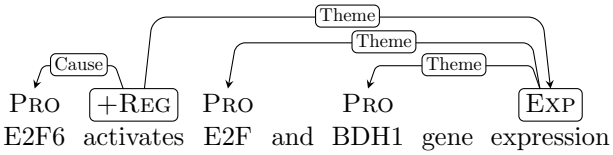
2.1.2 Event Extraction

Similarly to RE, Event Extraction (EE) is a task which aims at extracting relational knowledge about given entity mentions occurring in an unstructured text. In the context of BioNLP, EE is typically referred to as biomedical event extraction. Informally, a biomedical *event* is a formal description of a biomedical process or "happening" involving biomedical entities such as PROTEINS. As opposed to relations, events aim not only to capture which biomedical mentions are interacting, but also to describe which kind of interactions are actually occurring, and which role each mention has in them (Björne, 2014). This makes events suitable for capturing fine-grained descriptions of processes from the elaborate biomedical statements occurring in the scientific literature.

Specifically, in contrast to relations, which are high-level pairwise associations between entity mentions (Table 2.3, top), events are semantically rich, structured representations that (i) are rooted on *triggers*, i.e., tokens (typically verbs or nominalized verbs) that indicate the presence of a biomedical "happening" of a certain category, and thus determine the "center" and the semantic type of those events, and (ii) have multiple *arguments* (alternatively, *edges*), i.e., entities or other event triggers that participate in the events with a semantic role (Table 2.3, bottom). For instance, in Table 2.3 (bottom) a +REGULATION event is anchored to the trigger token "activates", and it has the "E2F6" PROTEIN as CAUSE argument, and the "expression"-centered EXPRESSION event as THEME argument. As a result, events naturally capture the association of more than two mentions, and each event can in turn be argument of other events, resulting in nested event structures (Björne & Salakoski, 2018).

Given an input text and marked entities, biomedical event extraction thus aims to recognize events by identifying *triggers* and *arguments* as relevant information units. Due to the increased complexity caused by the fine-grained representation of events, the task is typically tackled into two stages in a pipeline,

Table 2.3: Comparison of relation and event annotations given input entity mentions (i.e., PRO: proteins). Relations are pairwise associations of entity mentions, whereas events are structures rooted on *trigger* tokens with a semantic type (e.g., +REG: positive regulation; EXP: expression) which can have multiple *arguments* in different semantic roles (arrows and their labels). Event arguments can be entities or other events, resulting in nested event structures.

Task	**Example output**
Relation extraction	
Event extraction	

i.e., using two specifically purposed classifiers. We detail these stages below, using Figures 2.2 and 2.3 as running examples to explain each stage:

- **trigger detection**: a sub-task aiming at recognizing and classifying the tokens that may trigger events. For instance, a +REGULATION event trigger centered on the token "activates", and an EXPRESSION event trigger anchored on the token "expression", have to be recognized (Figure 2.2, solid boxes). Instead, other tokens such as "and" and "gene" do not trigger any biomedical event, and thus are not classified as triggers (Figure 2.2, dashed boxes). Sequence tagging is typically employed as problem setup (cf. Section 2.1) for tackling this event extraction stage;

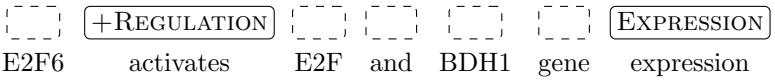


Figure 2.2: Example of trigger detection in biomedical event extraction. Solid boxes and dashed boxes indicate triggers to be recognized or not, respectively.

- **edge detection**: given the previously identified event triggers, this sub-task aims at identifying and classifying the semantic arguments (or edges) of each of the events centered on these triggers. Specifically, candidate edges from each event trigger to given entities or other event triggers are built and thus evaluated (Figure 2.3, both dashed and solid arrows). For instance, in Figure 2.3 four potential arguments are evaluated for the +Regulation trigger anchored on the "activates" token, namely the Protein entities "E2F6", "E2F", "BDH1", and the Expression event centered on "expression". Analogously, four arguments are evaluated for the Expression event. Each entity or trigger that is actually argument of events (Figure 2.3, solid arrows) is thus assigned a semantic role in the event itself (Figure 2.3, label on solid arrows), and negative edge instances are instead discarded (Figure 2.3, dashed arrows). Edge detection (also referred to as argument detection) is similar in spirit to relation extraction, where a pairwise evaluation between mentions has to be carried out.[3] As such, this sub-task is typically framed as text classification problem (cf. Section 2.1), where the token sequence is evaluated multiple times, one for each pair of properly marked mentions.[4]

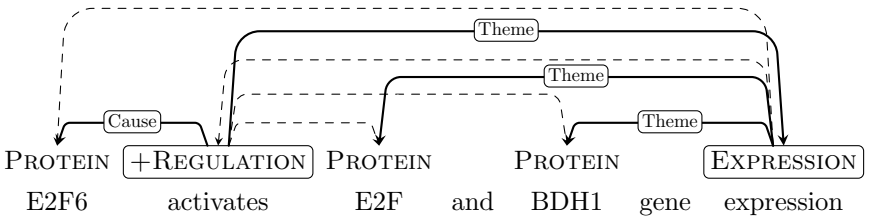


Figure 2.3: Example of edge detection in biomedical event extraction. Solid arrows and dashed arrows indicate event arguments to be recognized or not, respectively. Labels on solid arrows are the roles that entity or event trigger arguments (end of arrows) have in a specific event (start of arrows).

[3] However, relation extraction performs an "entity–entity" evaluation, whereas in edge detection the evaluated associations are between "trigger–[entity|trigger]" pairs.

[4] This is typically done by (a) using position embeddings (Zeng et al., 2014) as features to be added to word embeddings, or by (b) replacing mention tokens with a placeholder (Alt et al., 2019) (e.g., "`Mention`") when fine-tuning large pre-trained models.

The detection of both triggers and edges results in the identification of biomedical events. For instance, in Figure 2.3 we can see the following events: an EXPRESSION event anchored on "expression" having as THEME arguments the "E2F" and "BDH1" PROTEINS, and a +REGULATION event centered on "activates", having as CAUSE the "E2F6" PROTEIN, and as THEME argument the aforementioned EXPRESSION event.

As opposed to general domain event extraction (Walker et al., 2006), nested structures are possible and frequent in biomedical event extraction, making it a challenging task. The standard benchmark corpus for assessing advances in biomedical event extraction methods is the well-known GENIA event extraction corpus (J.-D. Kim et al., 2011), which comprises over 37% nested events. A variety of pipeline systems made up of trigger and edge detection modules has been proposed in recent years to tackle the complexity of the task, e.g., Björne and Salakoski (2018) and Li et al. (2019). In the direction of pipeline systems, in Chapter 4 we provide a thorough evaluation of edge detection performance in biomedical event extraction. We also provide in Chapter 5 a novel end-to-end solution for biomedical event extraction which jointly learns both trigger and edge information gaining in both performance and efficiency.

Evaluation metrics As for relation extraction, event extraction methods are evaluated using standard macro-averaged precision (Eq. 2.1), recall (Eq. 2.2), and F_1 score (Eq. 2.3). The evaluation is carried out at the event level, thus based on an equality criterion (detailed below) between each predicted event structure (i.e., comprising both the event trigger and its arguments) and its corresponding gold event annotation. As a result, the typical trigger and edge detection sub-tasks of pipelined event extraction systems are not formally evaluated per se, and their assessment is left to researchers. In Chapter 4, we contribute to the field by providing standardized means for evaluating biomedical edge detection performance in pipelined systems.

The standard event equality criterion employed in biomedical event extraction is called *approximate recursive span matching*. Specifically, a predicted event is said to match a gold event if all the following conditions hold: (i) the predicted event trigger is equivalent to the gold event trigger (i.e., the predicted

token span is entirely contained within the gold token span extended to the left and the right by one token); (ii) the predicted type of the event trigger is the same as the type of the gold event trigger; (iii) all the predicted arguments (either entities or triggers) are equivalent to the gold arguments; (iv) all the predicted role types of those event arguments are the same as the role types of gold event arguments; and (v) recursively, the aforementioned conditions hold for all the events referred as arguments, relaxing iii-iv to at least match the THEME argument as the main descriptive part of biomedical events (J.-D. Kim et al., 2011). As a result, the evaluation of biomedical events is very strict and requires a system to be carefully designed to reach good performance. We believe this strictness is necessary to avoid partial event matches which are unlikely to carry useful information for downstream uses (e.g., a REGULATION event with a CAUSE argument but no actual THEME argument).

2.2 Approaches in Natural Language Processing

In this section we provide an overview of the fundamental methods that are typically employed in NLP, specifically focusing on approaches for relation and event extraction of relevance to this thesis. In Section 2.2.1 we introduce symbolic approaches such as patterns and rules, then providing an overview of machine learning in Section 2.2.2. We then describe neural network approaches in Section 2.2.3, finally outlining transfer learning methods in Section 2.2.4.

2.2.1 Patterns and Rules

The *symbolic* approach to natural language processing, namely the manipulation of linguistic units such as words as discrete symbols, has been an initially prevalent paradigm in NLP. Concretely, in symbolic NLP hand-coded patterns and rules are designed by human experts over words and their categorial features (e.g., POS tags) to the goal of capturing the meaning of text and solve a task at hand. Methods exploiting patterns and rules date back to the dawn of NLP and to the interest in the test of machine intelligence proposed by Turing (1950), and have been commonplace until the rise of statistical approaches.

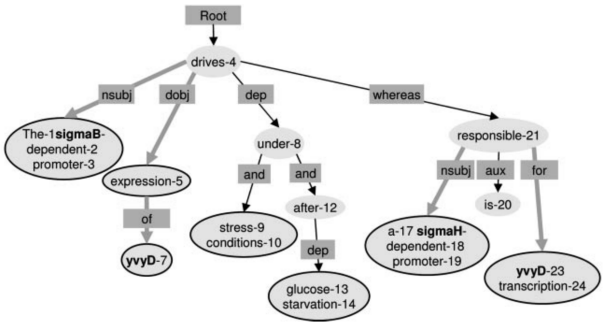


Figure 2.4: Chunked dependency parse tree. Tokens with their indices (ellipses) are linked by grammatical dependencies (type within rectangles). Entities are indicated in bold, whereas extracted paths by rules, denoting entity relationships, are indicated by thick grey arrows. *Figure source: Fundel et al. (2007).*

Since patterns and rules are injected by human experts to mimic their own reasoning behaviour in solving a specific task, symbolic NLP approaches are typically task-specific static programs (as opposed to statistical approaches which automatically learn patterns from data, thus allowing the repurposing of model architectures, see from Section 2.2.2 onwards). We here outline seminal approaches relevant to the extraction of relational information.

Lexico-syntactic patterns have been exploited to extract hyponymic lexical relations. For instance, given the text "Countries, including Canada and England", (Canada, country) and (England, country) hyponyms can be extracted using the pattern "`NP {,} including {NP ,}* {or|and} NP`", where *NP* is a noun phrase (Hearst, 1992). However, the interest in biomedicine in keeping pace with the scientific literature relies on findings such as biomedical interactions rather than lexical relations. Towards this goal, Ono et al. (2001) proposed patterns such as "`interaction (between|among) A and B`", "`A(-/)B complex`", and "`A and B association with each other`", where *A* and *B* are entity mentions. To further handle the variability of biomedical texts, recent methods leverage dependency parse trees. For instance, in Fundel et al. (2007) a dependency tree is built on top of noun chunk tokens, and rules are applied to identify semantic relations between entities based on their shortest path (Figure 2.4). We employ a similar approach for relation extraction in

Chapter 3. While symbolic approaches are still effective in relation extraction, their results in event extraction are far lower compared to statistical approaches due to the high complexity of the task (J.-D. Kim et al., 2011). We thus adopt deep and transfer learning methods for event extraction (Chapters 4 and 5).

2.2.2 Machine Learning

Machine Learning (ML) is a field of study aiming at building computational models for given tasks by automatically learning patterns from data (i.e., the previous experience on the tasks). More formally:

> *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks T, as measured by P, improves with experience E."* (Mitchell (1997))

For instance, in relation extraction (i.e., the task T) the goal is typically to maximize the F_1 score (i.e., the performance measure P) on a set of unseen relation examples by learning from a set of relation examples with given classification labels (i.e., the training experience E). Given $X = \{x_1, ..., x_n\}$ the training instances and $Y = \{y_1, ..., y_n\}$ the corresponding class labels, the goal is thus to learn a function f that generalizes to unseen instances, assigning the right labels to new examples. This setup in which training data consists of tuples $\{(x_i, y_i)\}_{i=1}^{n}$, where n is the number of instances, is called *supervised learning*. In contrast, in *unsupervised learning* $\{(x_i)\}_{i=1}^{n}$ are only available, thus the goal reduces to derive input structure or as a mean for feature learning.

Supervised learning deals with two common prediction problems, namely *classification* and *regression*. In classification, the label y_i is a discrete value belonging to a predefined set of classes (or categories) C, whereas in regression the label y_i is a continuous real value. The former can be further divided into *binary*, *multi-class*, and *multi-label* classification. In binary classification, $|C| = 2$ (e.g., determine whether an email is spam or not) whereas in multi-class classification, $|C| \geq 2$ (e.g., assigning to a word a part-of-speech class from a tag inventory). In both binary and multi-class classification an instance

is assigned exactly one label drawn from $|C|$. In multi-label classification, an instance can be instead assigned multiple classes from $|C|$ (e.g., determine the list of topics covered in an input text).

Due to the nature of relation and event extraction tasks, in this thesis we focus on classification problems, particularly binary classification (Chapter 3), multi-class classification (Chapter 4), and multi-class multi-label classification[5] (Chapter 5).

From symbols to vectors In machine learning, input, intermediate, and output representations are all encoded as vectors. Each input instance $x_i \in X$ is a vector of f *features* (or parameters), i.e., $x_i = [x_i^{(1)}, ..., x_i^{(f)}]$, whereas each output $y_i \in Y$ is a vector indicating the probability of the example x_i to belong to the c possible classes, i.e., $y_i = [y_i^{(1)}, ..., y_i^{(c)}]$. A feature is an informative property that is potentially discriminative in the learning process in order to assign the correct label y_i to each example x_i. For instance, in computer vision elemental features are the raw pixel values (e.g., the components of the RGB color space), which are naturally encoded in digital image data.

As opposed to computer vision, NLP requires words to be converted into vectorial representations before being fed into a machine learning model. A naïve solution is to represent each word as *one-hot* vector, i.e., a binary vector with the size of the vocabulary, where all dimensions are 0 except for the dimension corresponding to the actual word, which is 1. Treating words as *discrete* representations has shown to exhibit several shortcomings (Smith, 2020), leading to *distributed* representations such as word embeddings (Mikolov, Sutskever, et al., 2013; Mikolov, Chen, et al., 2013) (see Section 2.2.3.1).

Besides elemental features representing the input surface, a wide array of characteristics can be encoded. Examples are word attributes derived from a syntactic analysis such as lemmas and part-of-speech tags (Björne & Salakoski, 2018), as well as character n-grams and word suffixes, amongst others. In relation and edge detection tasks, features encoding the relative position of each

[5]Multi-class multi-label classification refers to the classification setup in which an instance can be assigned multiple labels (i.e., multi-label) from $|C| \geq 2$ possible ones (i.e., multi-class).

word to the mentions being evaluated have also been proven successful (Zeng et al., 2014; T. H. Nguyen & Grishman, 2015).

Learning and generalization Given X the input examples, and Θ some learnable parameters (or *weights*) of the model, the training objective is to find an optimal function $f(X;\Theta)$ that minimizes a specific *loss function* during the learning process. To meet this goal, optimization algorithms such as gradient descent are typically employed. A loss function measures the discrepancy between the predictions $\hat{y}_i$ compared to the ground truth y_i. Relevant loss functions for classification which are used in this thesis are the binary cross-entropy loss and the categorial cross entropy loss (or simply cross entropy loss).

The binary cross-entropy loss is used for binary classification problems (i.e., $|C| = 2$). Given y a binary value denoting the ground truth of the example (e.g., 1 for positive, 0 for negative), and $\hat{y}$ the predicted probability score of the example (i.e., in $[0, 1]$), the binary cross-entropy is expressed in Eq. 2.4:

$$CE_{binary} = -y \cdot log(\hat{y}) + (1 - y) \cdot log(1 - \hat{y}) \tag{2.4}$$

When dealing with $|C| \geq 2$ classes, the aforementioned loss generalizes to the categorial cross-entropy loss, typically referred to as cross-entropy loss. A separate loss value is computed for each class label $c \in C$, and the results are then summed together as follows (Eq. 2.5):

$$CE = -\sum_{c=1}^{C} y_c \cdot log(\hat{y}_c) \tag{2.5}$$

As a result, the loss value enlarges as the predicted probabilities diverge from the true labels, whereas it decreases as the predictions are close to the ground truth (i.e., a perfect classifier would have 0 as loss). Minimizing the cross-entropy is thus necessary for learning and generalization.

Generalization is a concept tied to the *bias-variance tradeoff*, namely the ability of a model to minimize both the error on training data, and the difference between training and test error (Goodfellow et al., 2016). *Bias* occurs when the error on training data is large, whereas *variance* occurs when the model exhibits a high gap between errors on training and test data, thus being too

sensitive to random perturbations in the training data. The model is said to face *underfitting* in the first case, and *overfitting* in the latter. The capacity of a model, i.e., the ability to fit different kinds of functions, is thus a crucial factor for generalization (Goodfellow et al., 2016; Ruder, 2019) and the main motivation for deep learning (Section 2.2.3).

Model assessment In assessing machine learning models, *training*, *development* (alternatively, validation) and *test* portions of the original dataset play a crucial role. The training set is used to train a model (and its potential variants) to learn the mapping between input data and the corresponding output labels; the development set is used to tune the hyper-parameters of the designed classifier and to select the best performing model; and the test set is used to assess the performance of the final model, thus the generalization to unseen data instances. These portions are typically given along the data itself to foster comparability of different approaches. However, when standard data splits are not available, *cross validation* techniques are typically employed.

The most common is *k-fold* cross validation, in which the original dataset is randomly partitioned into k subsets, and k experiments are conducted by using $k-1$ subsets as training data, and the remaining subset as test data. The resulting k performance scores are then averaged, providing an estimate of the accuracy of the model to new and previously unseen data.

Challenges and deep learning Traditional machine learning approaches, such as Kernels and Support Vector Machines (SVM) (Cortes & Vapnik, 1995), have been shown to work very well in a wide array of problems. However, they have not historically succeeded in complex AI fields such as NLP, where the target function to learn is often very complex (Goodfellow et al., 2016). The transition to deep neural network approaches has thus been mainly motivated by the need of more flexible learning functions which rely on compositionality, mitigating the limited representation power and the curse of dimensionality of traditional machine learning (Goodfellow et al., 2016). We focus on this family of approaches in the following sections.

2.2.3 Neural Networks

Neural Networks (NNs), also known as Artificial Neural Networks (ANNs), are a family of machine learning models originally intended to be computational abstractions of the learning process in the human brain. They can be seen as network-like structures composed by a collection of interconnected computing units, called *neurons* (or nodes), whose outputs feed other neurons.

Neural networks lay their foundations on early neuroscience-inspired linear models such as the McCulloch-Pitts neuron (McCulloch & Pitts, 1943) and the perceptron model (Rosenblatt, 1958). Despite these units are referred to as neurons, modern research in neural networks is no longer guided by the goal of understanding how the brain works, but is rather concerned on building systems to deal with tasks requiring intelligence (Goodfellow et al., 2016).

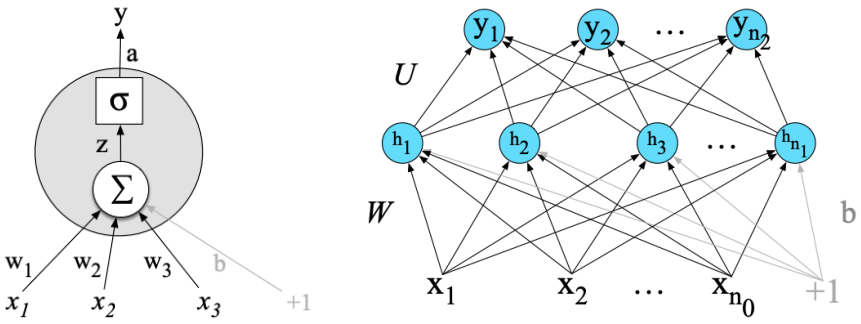
In these terms, a neuron can be seen as the building computational block of a neural network. A neural unit takes a set of n real valued scalar inputs $x_1, ..., x_n$ with their associated weights $w_1, ..., w_n$, and computes a weighted summation, adding a bias term b, a learnable parameter which determines the y-intercept of the function. The resulting sum z is thus passed through a non-linear function σ, giving a scalar *activation* value a as output (common activation functions are detailed in the next paragraphs). The computation of the neural unit is thus formally defined as follows (Eq. 2.6):

$$a = \sigma\Big(\overbrace{\sum_{i=0}^{n} w_i \cdot x_i + b}^{z}\Big) \tag{2.6}$$

whereas a graphical representation of the operations is provided in Figure 2.5a, where the unit's output a also corresponds to the output y of the network (i.e., $y = a$). For convenience, given D_n, D_m the dimensions of inputs and outputs, respectively, Eq. 2.6 can be rewritten using vector notation as follows (Eq. 2.7):

$$a = \sigma(\overbrace{W \cdot x + b}^{z}) \tag{2.7}$$

where $W \in \mathbb{R}^{D_n \times D_m}$ is the weight matrix, $x \in \mathbb{R}^{D_n}$ the input vector, and $b \in \mathbb{R}^{D_m}$ the bias vector, with $W, b \in \Theta$ the learnable parameters.



(a) A neural unit. (b) A feed-forward neural network.

Figure 2.5: A neural unit as a building block of neural networks, and a feed-forward neural network. *Figure source: Jurafsky and Martin (2020).*

Combining such units together allows to build multi-layer neural networks, in which outputs from units in one layer are fed to units in the next layer. In essence, a neural network is thus a composition of many affine functions (i.e., linear functions with a constant), interleaved with non-linear functions (i.e., activation functions) (Mitchell, 1997). Non-input and non-output layers, namely *hidden* layers, are the core of neural networks (Jurafsky & Martin, 2020). They are formed of a number of non-linear hidden units (as the one depicted in Figure 2.5a) which perform non-linear transformations on the data.

A simple neural network comprising hidden layers is the Feed-Forward Neural Network (FFNN). A FFNN with a single hidden layer is presented in Figure 2.5b. Given W the weight matrix of the hidden layer, U the weight matrix of the output layer, b and b' (omitted in Figure 2.5b) the bias vector of the hidden and the output layer, and σ and σ' the non-linear functions of the hidden and the output layer, the resulting output vector y of the two-layer[6] FFNN can be defined as follows (Eq. 2.8):

$$y = \sigma'(U \cdot \overbrace{\sigma(W \cdot x + b)}^{h} + b') \tag{2.8}$$

[6]The enumeration of the layers of a neural network does not include the input layer.

where h is the output vector of the hidden layer. Multiple layers can be stacked, resulting in neural architectures with many hidden layers, referred to as *deep* neural networks. The use of deep architectures is referred to as *deep learning*.

Activation functions The introduction of non-linearity on neural networks is crucial to better capture complex functions. Activation functions are in charge for adding this capability, as well as to squash real numbers to fixed intervals for computational reasons. Activation functions are used at hidden and output layers. For instance, an example activation function for hidden layers is the Rectified Linear Unit (ReLU), defined as in Eq. 2.9:

$$\sigma(z) = max(0, z) \tag{2.9}$$

which outputs z if $z > 0$, and 0 otherwise. Despite its simplicity, it typically overcomes the limitations of other common activation functions, such as the hyperbolic tangent and sigmoid, which are known to easily saturate and to be expensive to compute (Goldberg, 2017).[7]

As for the output layer, the most common activation functions are the sigmoid activation function (or logistic function) and the softmax. The sigmoid transforms each value z in $[0, 1]$, squashing outliers towards each end (Eq. 2.10):

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{2.10}$$

and thus is suitable for binary classification where the output can be interpreted as a probability value. As previously mentioned, it is currently deprecated in hidden layers (Goldberg, 2017). When dealing with multi-class classification problems, the softmax function is typically employed. The softmax normalizes each element z_i of the output vector z so that z_i relies in $[0, 1]$ and the values sum to 1 (Eq. 2.11):

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}} \tag{2.11}$$

[7]Note that ReLU is typically inappropriate for some neural architectures, such as Recurrent Neural Networks (RNNs). For further investigation, refer to Le et al. (2015).

This allows the output vector z of raw scores to be interpreted as a probability distribution over the possible classes C.

Specialized layers and architectures Simple feed-forward neural networks have been shown to be universal function approximators (Hornik et al., 1989) and as a result the composition of non-linearities in neural networks to have a high representation power (Goldberg, 2017). However, the non-specificity of FFNNs has given rise to more specialized layers and architectures that tackle specific challenges of language data. Indeed, a theoretical guarantee that FFNNs are universal approximators does not guarantee that the training algorithm will easily find a suitable function for all problems (Goldberg, 2017). Given the provided background on machine learning and neural networks, in the following sections we thus present layers and components that have been successful which are relevant to the remainder of this thesis.

In Section 2.2.3.1, we introduce the embedding layer, a layer that is necessary to model NLP problems, and aids in mapping word symbols to their corresponding vectorial representations. On top of the embedding layer, convolutional neural networks, which specifically model spatial properties of input data, can be employed. We introduce them in Section 2.2.3.2. Alternatively, to model time and long-range relationships between words, transformer models can be used (Section 2.2.3.3).

2.2.3.1 Embedding Layer

As introduced in Section 2.2.2, neural networks require input, intermediate, and output representations as vectors. However, words are discrete units of information, strictly requiring to be transformed into real-valued vectors beforehand in order to be used in neural networks. The *embedding layer* is the layer in which the transformation from symbols to vectors takes place.

Formally, given a vocabulary V consisting of all the words in the dataset, d the vector size for each word, and a matrix $M \in \mathbb{R}^{|V| \times d}$ storing the vectors for each word, a lookup table $L_M(\cdot)$ is used (Collobert & Weston, 2008) to turn each word $w_i \in V$ into a d-dimensional vectorial representation x_i, known as the *word embedding* of w_i (Eq. 2.12):

$$L_M(w_i) = x_i \tag{2.12}$$

where the embedding vector $x_i \in \mathbb{R}^d$ corresponds to the i^{th} row of the matrix M. Values in M are typically pre-computed and can be further tuned during training together with the other learnable parameters of the network.

The embedding layer can be used in the same way to also map other discrete, categorial features about w_i to vectors, e.g., its part-of-speech tag. As a result, x_i becomes the concatenation of the d-dimensional word vector and the d'-dimensional part-of-speech vector (where d' is an hyper-parameter), leading to a $(d + d')$-dimensional representation for the word w_i.

Distributed word representations Word representation research, namely *putting words into computers* (Smith, 2020), has a long-standing tradition. Not long ago, words were treated as discrete, integer representations, in which each word w_i was given a $|V|$-dimensional one-hot vector, i.e., only the dimension corresponding to the index of the word in the vocabulary V was activated, i.e., 1 for the identity dimension, and 0 for all others (Figure 2.6, left). However, discrete word representations are extremely sparse, leading to issues such as the curse of dimensionality (i.e., a high number of feature dimensions for a small set of examples). Moreover, they do not carry any information regarding the properties of words, such as their semantic similarity or relatedness. For instance, words such as "sternum" and "breastbone", which both refer to the bone located in the central region of the chest, were assigned different vectors as for "activates" and "suppresses", which instead express a divergent meaning.

To mitigate the issue, word representations that leverage the distributional properties and usage of linguistic items in large collections of data have been explored. The idea lies the foundation on the *distributional hypothesis* in linguistics, which states that words occurring in similar contexts tend to have similar meanings, mainly popularized by Harris (1954) and Firth (1957). As a result, a variety of models to derive *distributed* word representations – a representation of words' meaning that is distributed across the whole vector (Figure 2.6, right) (Smith, 2020) – have been proposed. Under this hypothesis, similar words get similar vectors (e.g., "E2F6" and "E2F" in Figure 2.6).

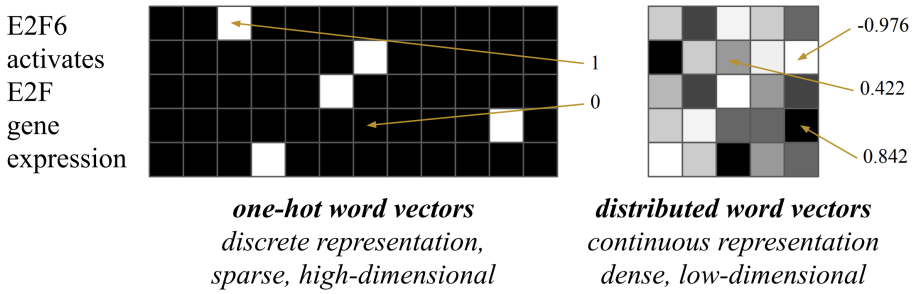
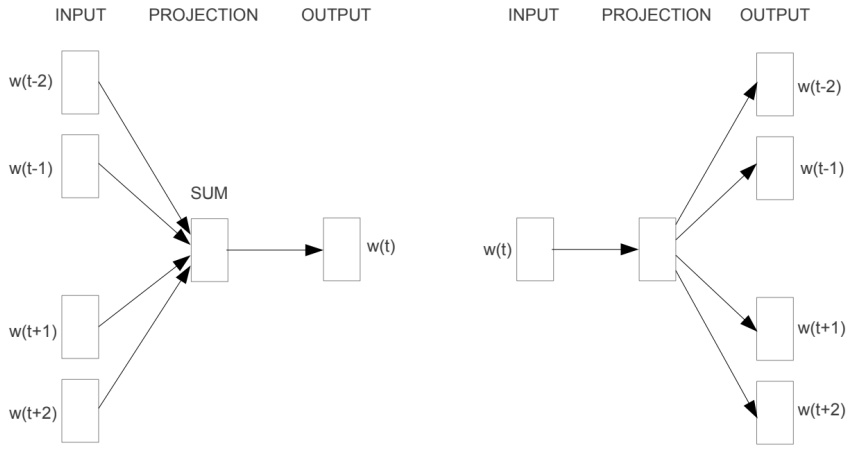


Figure 2.6: Word representations for the sentence "*E2F6 activates E2F gene expression*". Shades of grey represent values. One-hot word vectors are sparse, and features grow with the size of V. In distributed word vectors the word's meaning is instead distributed across the vector, whose dimensionality is fixed.

Seminal examples in distributional semantics are the Brown clusters (Brown et al., 1992), latent semantic analysis (Landauer et al., 1998), and latent Dirichlet allocation (Blei et al., 2003). By leveraging dimensionality reduction techniques, vectors could be compressed, mitigating the feature sparsity and the curse of dimensionality, ultimately leading to low-dimensional, continuous, *dense* vectors that model the distributional view of word meaning (Smith, 2020) and that can be inspected for similarities and analogies in a vector space model. The first work that showed the usefulness in neural networks of distributional semantic models, or *word embeddings*, is by Collobert and Weston (2008). However, with the growth of corpora, the scalability of these methods became an issue, especially in the case of limited computational resources.

This led to the popularization of the word2vec toolkit (Mikolov, Sutskever, et al., 2013; Mikolov, Chen, et al., 2013), a method that allows both the independent training of word embeddings as well as the use of word representations *pre-trained* by other researchers, thus mitigating the need of costly computing infrastructures. In their original paper, Mikolov, Sutskever, et al. (2013) proposed the continuous skip-gram and the continuous bag-of-words (CBOW) neural models (Figure 2.7). Given a set of raw texts and a context word window c (e.g., $c = 2$ for 2 words on both sides), d-dimensional word embeddings are created by (a) predicting the context words $w_{(t-c)}, ..., w_{(t+c)}$ given a target word



(a) CBOW model. (b) Skip-gram model.

Figure 2.7: The two proposed word2vec models to learn word embeddings (with word context window $c = 2$). *Figure source: Mikolov, Chen, et al. (2013).*

w_t (continuous skip-gram, Figure 2.7b), or by (b) predicting the target word w_t given the context words $w_{(t-c)}, ..., w_{(t+c)}$ (CBOW, Figure 2.7a), where c and d are hyper-parameters. The training objective is thus to learn word representations which maximize the probability of predicting context words or target words, respectively.

The use of word embeddings pre-trained on 6 billion tokens of the Google News corpus (Mikolov, Sutskever, et al., 2013) has became ubiquitous in a variety of NLP applications since then. However, some domains exhibit a specific jargon (Section 2.2.4.1) that is not included in the general purpose word embeddings derived from a news corpus, leading to a high fraction of missing word representations (referred to as *out-of-vocabulary* words). To mitigate the problem, domain-specific pre-trained word embeddings have been introduced. Of particular relevance to this thesis are the biomedical word embeddings by Pyysalo et al. (2013). They induced 200-dimensional word vectors by using the skip-gram model with a context size of 5 on biomedical publications from PubMed and PubMed Central. We employ them for biomedical edge detection in Chapter 4.

2.2.3.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) (Lecun et al., 1998) are a class of deep neural networks that have been successfully applied in a range of fields, from computer vision (Redmon et al., 2016; Krizhevsky et al., 2017) to a variety of natural language processing tasks, such as semantic role labeling (Collobert et al., 2011), part-of-speech tagging (Dos Santos & Zadrozny, 2014), event detection (T. H. Nguyen & Grishman, 2015), and question and sentiment classification (Kalchbrenner et al., 2014; Y. Kim, 2014).

The success of CNNs originates from their ability at capturing the spatial properties of the input data. As opposed to FFNNs, CNNs are in fact able to model local and position-invariant features in the text, thus acting as *ngram* detectors instead of naïve *bag-of-words* models (Goldberg, 2017). Accounting for the spatial locality correlation of words, regardless of their position in the text – a property named *translational invariance* – is indeed beneficial to handle the high variability of texts written in a natural language.

A convolutional neural network – also named *convolution-and-pooling* architecture (LeCun et al., 1995) – is made up by a sequence of layers, namely a *convolutional layer*, acting as a feature extractor (i.e., by identifying local ngrams that are informative for the task at hand), and a *pooling layer*, that combines the most informative features of the sequence. These two layers can be stacked into a hierarchy of convolution-and-pooling operations, allowing the network to combine long-range features (Goldberg, 2017). The resulting pooled representation is fed to a fully-connected layer in charge of making the final classification. See Figure 2.8 for a schematic overview.

It is important to note that the words in the input sequence have to be encoded into suitable word representations in order to be fed to a convolutional layer (such as word embeddings, see Section 2.2.3.1). As a result, CNNs are not self-contained, ready-to-use networks, but rather a collection of layers that can be plugged into larger neural architectures. In NLP, CNNs are typically employed on top of an embedding layer (Section 2.2.3.1, Figure 2.8).

From a closer perspective, given a sequence of N words $[w_1, ..., w_N]$,[8] the

[8]Typically N is set to the number of words of the longest sentence in the corpus, thus the sentence is 0-padded in the last unused row dimensions if necessary.

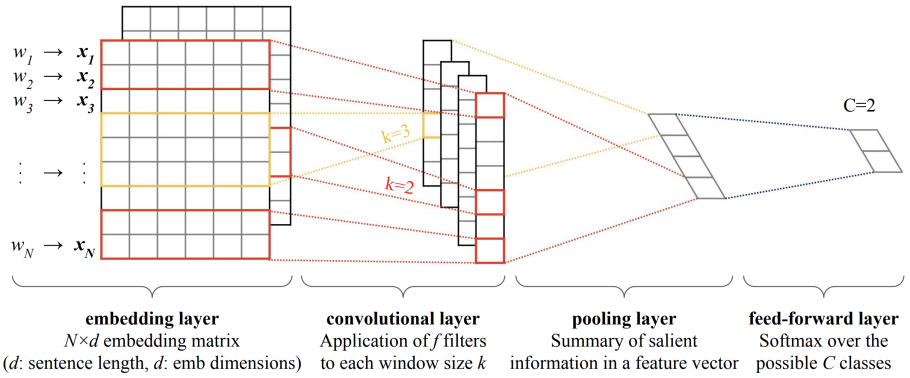


Figure 2.8: A convolutional neural network architecture with two filters and window sizes [2,3] (on top of the embedding layer) employed for binary sentence classification (output classes $C = 2$). *Figure source [adapted]: Y. Kim (2014).*

convolutional layer slides over a matrix $N \times d$ of the corresponding d-dimensional word representations $[x_1, ..., x_N]$ by applying non-linear functions (i.e., *filters*) to each of the k-size contiguous words, being k the *window size* (e.g., a window size of 3 considers *3*-grams). A scalar value results from the application of each filter to a k-words window, thus the application of multiple filters results in a vector summarizing the k-word windows properties. As a result, the size of the output vector is $n - k + 1$. When considering multiple window sizes, m $(n - k + 1)$-dimensional vectors are produced. The pooling layer is then used to condense the information of all the k-words window vectors m resulting from the convolutional layer into a single vector summarizing the salient information of the input sequence. The most widespread pooling operation is *max pooling*, which selects the maximum value across each dimension of the vectors, i.e., the most relevant indicators for each feature for solving the task at hand. The vector resulting from the pooling operation is then used as input to a simple feed-forward layer (i.e., a fully connected neural layer as introduced in Section 2.2.3, which outputs the most confident class $c \in C$ (e.g., by using the softmax function).

Convolutional neural networks have also shown to be more robust across language variations compared to traditional machine learning approaches on

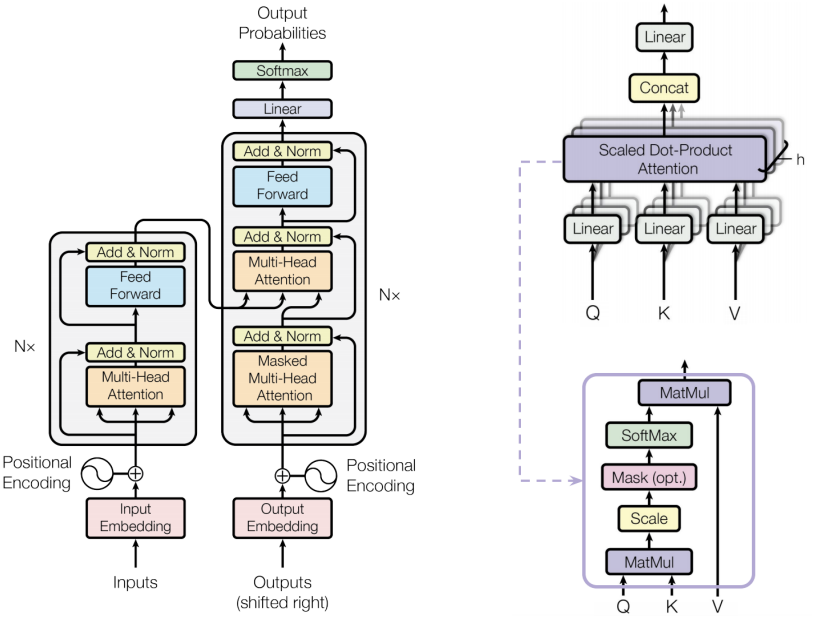
relational tasks (T. H. Nguyen & Grishman, 2015). We employ them for a cross-domain study on biomedical edge detection (Chapter 4). Details on domain variations are provided in Section 2.2.4.1, whereas for further details on CNNs we refer the reader to Goodfellow et al. (2016) and Goldberg (2017).

2.2.3.3 Transformers

A Transformer (Vaswani et al., 2017) is a neural network architecture specifically tailored to deal with sequence-to-sequence problems, such as machine translation and summarization (i.e., where both the input and the output are sequences). Formally, it is an *encoder-decoder* architecture (Sutskever et al., 2014), in which the encoder transforms an arbitrary-length sequence into a fixed dimensional vector representation, and the decoder uses this latent representation to produce another variable-length sequence as output. Both encoder and decoder can consist of multiple layers.

As opposed to former Recurrent Neural Network (RNN) approaches for processing temporal information, such as Long Short-Term Memory (LSTM) networks (Hochreiter & Schmidhuber, 1997), transformers do not require the input sequence to be processed word-by-word, allowing the parallelization of the computation and a significantly faster training (Vaswani et al., 2017). Moreover, transformers have shown to achieve high performance gains on a wide array of NLP tasks, motivating new developments such as BERT (Section 2.2.4.2).

Transformers process the input sequence as a whole, overcoming the limitations of directional approaches (e.g., left-to-right) by learning contextual information of words based on both their sides, and having direct access to all other words in the sequence. Specifically, transformers are based solely on an *attention* mechanism, which weighs the relevance of each input word to each other word in the sequence, in order to capture contextual information relevant for the task at hand. As a result, transformers completely dispense convolutions and recurrence (Vaswani et al., 2017). Figure 2.9a provides a high-level overview of the transformer architecture, which consists of $L = 6$ encoder layers and $L = 6$ decoder layers. The building blocks of the transformer architecture are introduced in the following.



(a) Transformer architecture. (b) Multi-head self-attention.

Figure 2.9: Transformer architecture, and detailed view of the multi-head self-attention sub-layer. *Figure source [adapted]: Vaswani et al. (2017).*

Word and positional embeddings Words in the input sequence are turned into d-dimensional word representations as described in Section 2.2.3.1 (Figure 2.9a, "Input Embedding"). Since the transformer dispenses recurrence, it requires information about the relative order of words to be injected.[9] The authors introduced d-dimensional *positional embeddings* to the goal, which encode the specific position of words in the sequence by using multiple sine and cosine functions of different wavelengths (refer to Vaswani et al. (2017) for details). Position embeddings are thus summed to word embeddings to equip word representations with order information (Figure 2.9a, "Positional Encoding").

[9] This is crucial because self-attention, described later on, is permutation-invariant.

Multi-head self-attention The embedding vector is fed to a transformer encoder (Figure 2.9a, grey box on the left). Each encoder consists of two layers: a multi-head self-attention layer (Figure 2.9a, orange box), and a fully connected feed-forward layer (Figure 2.9a, blue box). Both are followed by layer normalization which use residual connections (Figure 2.9a, yellow boxes). We here focus on the attention layer, detailing other layers in the following.

As previously introduced, attention is a mechanism that allows each token in a sequence to attend to relevant parts of a sequence. For instance, Bahdanau et al. (2015) introduced attention to score words between two input sequences for an English-to-French translation task. With *self-attention* (also known as intra-attention), Vaswani et al. (2017) extended the attention mechanism to work on the same input sequence, thus allowing words to focus on other contextual words that are relevant for the task at hand. They formally refer to their self-attention mechanism as to *scaled dot-product attention* (Figure 2.9b (top), purple box; expanded to show operations in Figure 2.9b (bottom)).

Formally, for each input word vector x_i, three different vectors are created: a query vector q_i, a key vector k_i, and a value vector v_i. Vectors are then packed into corresponding matrices Q, K and V to compute multiple vectors simultaneously. These three matrices, which form the input to the scaled dot-product attention component, are obtained by multiplying each input to three corresponding matrices of weights W_Q, W_K, and W_V, learned during training. The actual computation of the scaled dot-product attention is depicted in Figure 2.9b (bottom) and is defined as follows (Eq. 2.13):

$$Attention(Q, K, V) = \sigma\Big(\frac{QK^T}{\sqrt{d_k}}\Big)V \tag{2.13}$$

Specifically, the query Q and the key K undergo a matrix multiplication, giving a matrix of raw self-attention scores. For stabilizing gradients during training, results are scaled by dividing the result by $\sqrt{d_k}$, where d_k is the dimension of queries and keys. A softmax function σ is applied for normalizing self-attention scores in $[0, 1]$ such that they sum to 1. The scaled dot-product attention output is thus obtained by multiplying scores to the value matrix V.

In order to focus on multiple representations at multiple positions, Vaswani

et al. (2017) proposed *multi-head* self-attention (Figure 2.9b (top)). Specifically, Q, K and V are linearly projected h times, so that scaled dot-product attention is performed in parallel on all projections. Results are then concatenated and projected to get the final attention output.

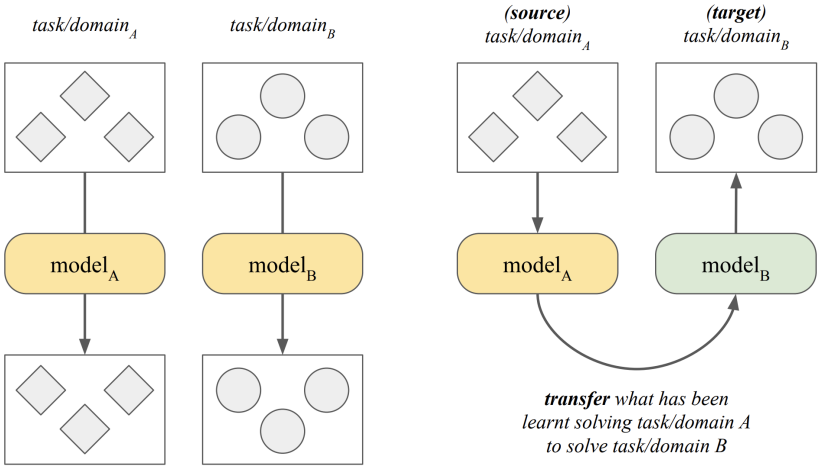
Residual connections, normalization and others The multi-head self-attention output is added to the original embedding vector, a process called *residual connection* that allows gradients to directly flow during training (He et al., 2016) (Figure 2.9a, incoming arrows to yellow boxes), then it undergoes *layer normalization* to speed-up the training process (Ba et al., 2016) (Figure 2.9a, yellow boxes). A feed-forward network consisting of two linear layers with ReLU activation (Figure 2.9a, blue box) transforms the output, which is again added to residual connections, and then normalized.

The resulting continuous representation comprising attention information is the output of the encoder, which is used for decoding. Multiple encoder blocks can be stacked, so that representations can be refined by flowing through them. The decoder (Figure 2.9a, grey box on the right) uses the same components described above, except for the addition of a dedicated encoder-decoder multi-head attention layer (Figure 2.9a, second orange box from bottom, on the right) that leverages both the output of the encoder (specifically, its transformation to Q and K matrices), and the output of the first multi-head attention[10] (Figure 2.9a, first orange box from bottom, on the right), whose calculations are performed on embeddings from the target sequence that are shifted one position right to let predictions rely on known outputs (Vaswani et al., 2017). The output of the decoder is then passed through a linear layer and softmax is applied (Figure 2.9a, layers on top of the grey box on the right) to output the next word with highest probability. Decoders, as encoders, can be stacked.

2.2.4 Transfer Learning

A default assumption in supervised machine learning is that the test data follows the same distribution as the training data, i.e., training and test examples

[10]The first multi-head self-attention layer of the decoder is masked to prevent conditioning the generation of the output sequence based on future words (Vaswani et al., 2017).



(a) Supervised machine learning setup. (b) Transfer learning setup.

Figure 2.10: Traditional supervised machine learning setup and transfer learning setup in comparison. *Figure inspired by: Ruder (2019).*

are assumed to be independently and identically (i.i.d.) sampled from the same underlying distribution. Given a particular task and domain (say, A), a model is trained and applied to the same task and domain A. When dealing with a different task and domain (say, B), another model is trained to deal with the peculiarities of the new data (Figure 2.10a). This setup assumes that a lot of labeled data is available for the task and domain at hand to train a reliable model, however for most tasks and domains this is not the case (Ruder, 2019).

Transfer Learning (TL) specifically tackles this issue, introducing ways in which a model trained on a task or domain (i.e., *source*) can be repurposed on a related task or domain (i.e., *target*) (Pan & Yang, 2010). As a result, instead of training a new model from scratch, a previous model can be employed and further trained (Figure 2.10b) on substantially fewer examples (Howard & Ruder, 2018). Besides the amount of required labeled data, transfer learning has shown several advantages, including time and cost efficient training of deep learning models, and unprecedented performance gains on a variety of NLP tasks (Devlin et al., 2019).

The paradigm draws inspiration from theories on *transfer of learning* in psychology, which state that knowledge gained in solving a problem in a particular context strengthens the learning of other problems in new situations (Perkins et al., 1992). In neural networks, the knowledge is encoded in the form of learned weights. Transfer learning is formally defined as follows:

> *"Given a source domain $\mathcal{D}_S$ and learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and learning task $\mathcal{T}_T$, transfer learning aims to help improve the learning of the target predictive function in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S = \mathcal{D}_T$, or $\mathcal{T}_S = \mathcal{T}_T$."*
>
> (Pan and Yang (2010))

The setup in which $\mathcal{T}_S \neq \mathcal{T}_T$ is referred to as *inductive* transfer learning, whereas when $\mathcal{D}_S \neq \mathcal{D}_T$ the setting is called *transductive* transfer learning (Pan & Yang, 2010). In the context of this thesis, we employ inductive transfer learning methods (Chapter 5), namely *sequential* transfer learning and *multi-task* learning (detailed in Section 2.2.4.2 and 2.2.4.3, respectively). We also investigate the need for transductive transfer learning (Chapter 4) to be tackled in future work, thus the importance of domains, which we define in the following section (Section 2.2.4.1).

2.2.4.1 Tasks and Domains

As introduced in Section 2.2.4, transfer learning involves the concepts of task and domain. More formally, a task (e.g., text classification) is defined as $\mathcal{T} = \{\mathcal{Y}, P(Y|X)\}$, where $\mathcal{Y}$ is the label space. Estimates for the prior distribution $P(Y)$ and the likelihood $P(Y|X)$ are learned from the training data $\{(x_i, y_i)\}_{i=1}^{n}$. A domain is instead defined as $\mathcal{D} = \{\mathcal{X}, P(X)\}$ where $\mathcal{X}$ is the feature space (e.g., the text representations), and $P(X)$ is the marginal probability distribution over that feature space.

While the concept of a task is fairly intuitive, this is not obvious for domains. Indeed, the term is quiet loosely used in NLP and there is no common ground on what constitutes a domain (Plank, 2016). Typically in NLP, domain is meant to refer to some coherent type of corpus, i.e., predetermined by the

given dataset (Plank, 2011). This may relate to topic, style, genre, or linguistic register. The notion of domain and what plays into it has though significantly changed over the last years, leading to relevant research lines.

The Penn Treebank WSJ (Wall Street Journal) corpus (Marcus et al., 1993) and the Brown corpus (Francis & Kucera, 1979) are prototypical examples, with the WSJ being considered widely as the canonical newswire domain. In the recent decade, there has been considerable work on what is considered *non-canonical* data. The dichotomy between canonical (typically considered well-edited English newswire) and non-canonical data arose with the increasing interest of working with *social media* with all its challenges related to the "noisiness" of the domain (Eisenstein, 2013; Baldwin et al., 2013). Models trained on canonical data failed in light of the challenges on, e.g., Twitter (Gimpel et al., 2011; Foster et al., 2011), or biomedical texts (Chiu et al., 2016).

Instead of relying on a coarse-grained definition of canonicity, recent work started to raise more awareness of the underlying variation in the data samples NLP works with. Indeed, NLP is pervasively facing heterogeneity in data along many underlying (often unknown) dimensions. A theoretical notion put forward by Plank (2016) is the *variety space*. In the variety space a corpus is seen as a subspace (subregion), a sample of the variety space. A corpus is a set of instances drawn from the underlying unknown high-dimensional variety space, whose dimensions (or latent factors) are fuzzy language aspects. These latent factors can be related to the notions discussed above, such as genre (e.g., scientific, newswire, informal), sub-domain (e.g., finance, immunology, politics, environmental law, molecular biology) and socio-demographic aspects (e.g., author's age and cultural identity), amongst others, as well as stylistic or data sampling impacts (e.g., sentence length, annotator bias).

For example, it is less known that the well-known Penn Treebank consists of multiple genres (Webber, 2009; Plank & van Noord, 2011), including reviews and some prose. It has almost universally been treated as prototypical news domain. Similarly, social media is typically considered only non-canonical data, but an analysis revealed the data to lie on a "continuum of similarity" (Baldwin et al., 2013). Even within the biomedical domain a variety of subdomains have shown to exhibit a different linguistic behaviour (Lippincott et al., 2010, 2011).

Understanding linguistic variations will ultimately help to not only overcome overfitting to overrepresented domains (e.g., the newswire bias (Plank, 2016)), but also work on robustness of models across domains. We specifically study cross-domain performance in biomedical edge detection in Chapter 4, providing insights for future work in transductive transfer learning.

2.2.4.2 Sequential Transfer Learning

Sequential transfer learning refers to the inductive transfer learning setup in which the different tasks are learnt in a sequence. More precisely, the knowledge acquired in learning a (source) task where abundant data is available is fully exploited to subsequently learn a (target) task in which data is potentially limited to train a reliable deep learning model.

The typical and most effective approach in NLP is to use a language modeling task[11] as source task (A. Wang, Hula, et al., 2019). Language modeling aids to not only leverage the large amount of easily accessible raw data – insofar as it requires no supervision – but also helps on learning general properties of language that can be exploited for a variety of target tasks. From a closer view, sequential transfer learning typically involves the following stages (Ruder, 2019):

- **pre-training**: a one-off, computationally costly stage in which a model is trained on a large unlabeled corpus, producing representations which are meant to capture general aspects of language;
- **fine-tuning**: a resource-efficient stage in which representations from a pre-trained model are used to initialize the model's weights, that are further adjusted on a supervised target task.

The sequential transfer learning paradigm has many advantages. First, compared to training a model from scratch, it lessens the amount of target data required for reaching the same performance on a target task (Howard & Ruder, 2018), while increasing the robustness on out-of-distribution data (Hendrycks

[11] Language modeling refers to the task of assigning a probability for a given word or sequence, based on a sequence of words (Goldberg, 2017).

et al., 2020). Second, the release of large pre-trained models allows researchers to directly focus on the target task at hand, designing effective solutions even if a costly computing infrastructure is not affordable. In addition to reducing the significant carbon emissions of large model pre-training (Strubell et al., 2019), the reuse of pre-trained models also represents a step towards the democratization of NLP (Riedl, 2020). Finally, the outbreaking advances in NLP that have been registered in the past two years are mostly due to sequential transfer learning approaches (Ruder et al., 2019), and specifically to a variety of task-specific solutions on top of transformer-based models such as BERT (Devlin et al., 2019), which we introduce in the following.

BERT The combination of sequential transfer learning ideas with the transformer architecture (Vaswani et al., 2017) (Section 2.2.3.3) has resulted in the development of BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019). As the name suggests, BERT is based on transformers, and particularly on transformer encoder layers (called *transformer blocks*).

Indeed, since BERT is meant to produce language representations that can then be used for arbitrary tasks, it does not require decoders. However, training encoders for the language modeling task (i.e., predicting the next word in the sequence) is intrinsically unidirectional, in that attending to future words would condition the predictions on the target words themselves. In transformers, bidirectional self-attention is possible just because the words fed to the encoder are not part of the prediction. To tackle the issue and make full use of left and right contexts, Devlin et al. (2019) introduced bidirectionality by using variants of the language modeling task during pre-training (detailed below).

The BERT architecture is schematically depicted in Figure 2.11. Input embeddings are fed to L stacked transformer blocks, which consist of H feed-forward hidden units, and A attention heads.[12] Special tokens are included in the input sequence to allow BERT to naturally apply to diverse tasks. Specifically, a classifier token `[CLS]` is introduced at the beginning of the whole sequence and is used for sentence classification tasks, whereas a separator `[SEP]`

[12] In its default variant, called BERT$_{Base}$, $L = 12$, $H = 768$, $A = 12$. The total number of learnable parameters is 110M.

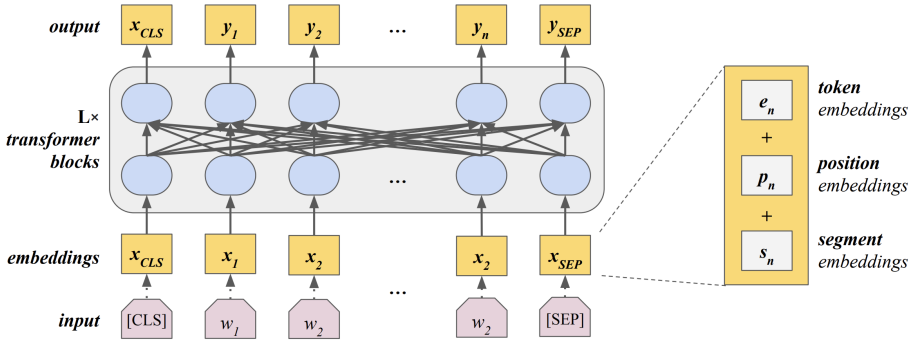


Figure 2.11: An high-level overview of the BERT architecture.

is added to mark the end of a sentence – and the start of the following one – in order to support tasks dealing with two sequences (e.g., question answering).

Similarly to the transformer (Vaswani et al., 2017), each token in the input sequence is converted to a embedded representation that makes use of both token and positional embeddings. In addition to token and position embeddings, BERT also introduces *segment* embeddings, which are markers used to distinguish the identity of two text segments in the input, in case of sentence pair classification or question answering (i.e., the one before `[SEP]`, and the one after it). Token representations are created using WordPiece embeddings, i.e., representations for subword-level linguistic units to effectively handle rare words in the input sequence (Wu et al., 2016). Token, position and segment embeddings are thus summed as in Vaswani et al. (2017) (Section 2.2.3.3). The input embeddings are then passed to each transformer layer, and thus undergo self-attention, whose results are fed through a feed-forward network to the next encoder. The output of the last transformer block is a H-dimensional representation that can be used for classification.

BERT being a sequential transfer learning approach, it consists of pre-training and fine-tuning stages (Figure 2.12). Specifically:

- **pre-training**: the model is trained simultaneously on two tasks. As introduced earlier, in order to support the bidirectional training of the encoders, variants of the language modeling task have been employed:

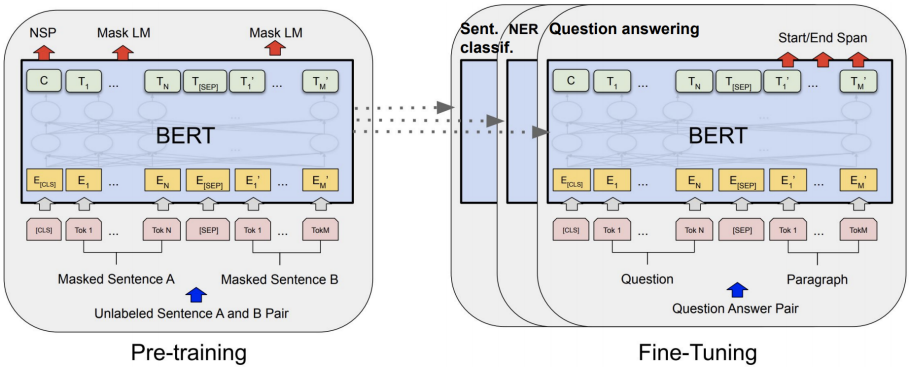


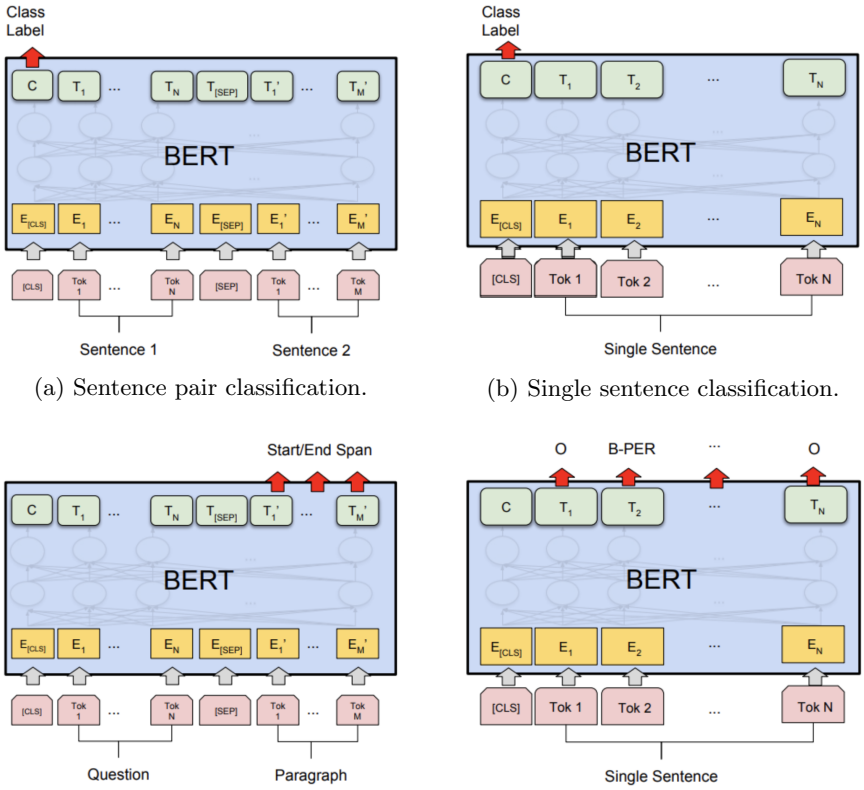
Figure 2.12: Overview of the sequential transfer learning stages in BERT. After pre-training on unsupervised language modeling objectives (left), the resulting model parameters are used as initialization and thus fine-tuned for a variety of target tasks (right). *Figure source [adapted]: Devlin et al. (2019).*

  - **masked language modeling** (Mask LM): instead of predicting the next token based on previous ones, 15% of the input tokens are randomly masked and need to be predicted (Figure 2.12, left, "Mask LM") by fully leveraging their context. Notably, 80% of the masked tokens are replaced with a special masked token `[MASK]`, 10% are replaced with another random token, and 10% are left unchanged. The rationale behind this choice is that the `[MASK]` token are never seen during fine-tuning (Devlin et al., 2019);
  - **next sentence prediction** (NSP): in order to also capture sentence relationships, useful for tasks involving text segment pairs, NSP aims at determining if the second sentence B in the input is actually following the first sentence A. This binary classification task is trivially generated from a large corpus. Specifically, 50% of the sentences are paired with actual following sentences, whereas 50% are paired with random sentences drawn from the corpus. The output of the special token `[CLS]` is used for classification (Figure 2.12, left, "NSP").

- **fine-tuning**: BERT weights resulting from the pre-training process are used for fine-tuning on a wide array of tasks (Figure 2.12, right). Inputs

pertaining to a particular downstream task are provided to the model following a specific input pattern, and a task-specific output layer is added on top of the core architecture. For instance, Figure 2.12 (right) shows fine-tuning on the question answering task, where the input consists of a question and a paragraph where to search for an answer (i.e., two `[SEP]`-separated text segments), and the output is made by labels denoting the start and end tokens of the answer within the paragraph segment. In practice, sophisticated task-specific layers can be designed on top of BERT in order to fully leverage both language representations and task properties (as we do in Chapter 5). BERT parameters and additional layers' weights are jointly updated for the supervised task at hand.

Different input and output patterns are handled in BERT to support fine-tuning on a variety of target tasks with only the addition of a single output layer (Figure 2.13). In sentence pair classification tasks (Figure 2.13a), such as textual entailment or duplicate question detection, two sentences are fed to BERT using the special `[SEP]` token to mark where a segment finishes and the other begins. The output is given by the final representation of the classifier token `[CLS]`, which aggregates sequence information. In single sentence classification tasks (Figure 2.13b), such as sentiment classification, the input is instead a single sentence, and similarly to sentence pair classification, `[CLS]` is used for prediction because the output is at a sequence level. As introduced earlier, for question answering tasks (Figure 2.13c) two segments are fed to BERT as in sentence pair classification tasks (Figure 2.13a), however the output is instead at a token level, where a start and end span of the question's answer to retrieve have to be get from token classes themselves. Lastly, for sequence labeling tasks (Figure 2.13d), such as POS tagging and NER, the input is a sequence of tokens and the outputs are token-level labels. In general, the output representations of actual tokens can be safely ignored in sequence-level tasks such as those illustrated in Figure 2.13a and 2.13b, whereas the aggregated information coming from the special token `[CLS]` are instead ignored in token-level tasks such as question answering and sequence labeling (Figure 2.13c and 2.13d).



(a) Sentence pair classification.

(b) Single sentence classification.

(c) Question answering.

(d) Sequence labeling.

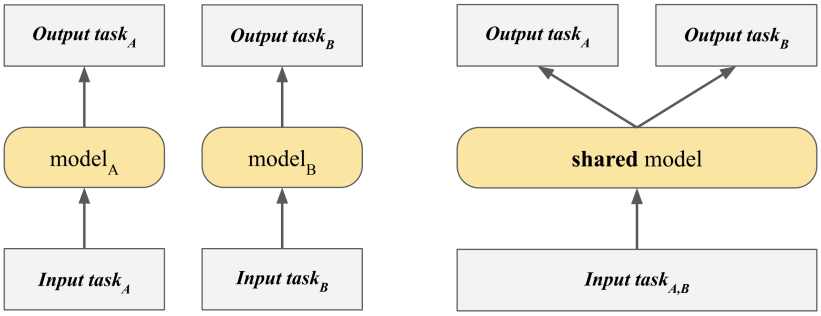
Figure 2.13: Example of diverse target tasks. *Figure source [adapted]: Devlin et al. (2019).*

A common practice is thus to reframe inputs and outputs for the downstream task at hand to the patterns proposed in Figure 2.13. For instance, relation extraction is typically framed as a single sentence classification task (Figure 2.13b). A text segment containing two marked entity mentions is fed to BERT and the aggregate information from the special token `[CLS]` is used to determine if the entities are in a semantic relationship. We use this approach in Chapter 3 to compare BERT-based models to our proposed symbolic approach. In event extraction, this is not that straightforward. As described

in Section 2.1.2, event extraction is a combination of sub-tasks, where trigger detection could be seen as a sequence labeling task as in Figure 2.13d, whereas edge detection is similar in spirit to relation extraction, and thus can be framed as a single sentence classification task (Figure 2.13b). Modeling these two sub-tasks in an end-to-end manner is thus not possible using the simple input patterns and output layers proposed by Devlin et al. (2019). We specifically tackle this issue in Chapter 5, proposing a novel linearization approach of event structures to token-level labels, and specific output layers to model nested events.

BERT has been originally pre-trained on a large amount of raw texts from BooksCorpus (Zhu et al., 2015) and English Wikipedia pages (Devlin et al., 2019). However, as introduced in Section 2.2.4.1, natural language exhibits important linguistic variations, and general domain language representations often obtain poor performance on domain-specific varieties such as biomedical texts (Lee et al., 2020). Similarly to word embeddings (Section 2.2.3.1), the release of BERT has motivated large domain-specific pre-trained models meant to be repurposed to tasks belonging to particular language varieties.

Pre-trained models of relevance to the biomedical domain are BioBERT (Lee et al., 2020) and SciBERT (Beltagy et al., 2019). BioBERT has been pre-trained on 4.5B PubMed abstract tokens and 13.5B PubMed Central full-text article tokens (Lee et al., 2020), whereas SciBERT data for pre-training comprises 3.2B SemanticScholar article tokens (of which about 82% are from biomedical papers) (Beltagy et al., 2019). The choice of the model also depends on the problem at hand; for instance, we found that BioBERT is more effective than SciBERT to our solution for event extraction (Chapter 5), whereas for relation extraction we obtained mixed results (Chapter 3). In general, the large performance advancements of pre-trained language models over pre-trained word embedding-based solutions in a variety of tasks and domains (Devlin et al., 2019; Beltagy et al., 2019; Lee et al., 2020) have made transformer-based solutions ubiquitous in NLP. This is also motivated by their increased ability in handling out-of-distribution data compared to previous approaches (Hendrycks et al., 2020), thus in their robustness across similar domains.



(a) Single task learning. (b) Multi-task learning.

Figure 2.14: Single and multi-task learning setups in comparison.

2.2.4.3 Multi-Task Learning

As opposed to sequential transfer learning, in multi-task learning the different tasks are learnt simultaneously. This approach aims at leveraging the training signals of multiple tasks at the same time, thus exploiting their mutual information as an inductive bias to help improve the learning and prediction performance for the tasks (Caruana, 1993, 1997). Figure 2.14 illustrates the difference between single and multi-task learning setups. Specifically, in contrast to the single task learning paradigm in which different models are built from scratch for each end task (Figure 2.14a, models A and B), in multi-task learning different tasks are learnt jointly by exploiting a shared representation in the model (Figure 2.14b, shared model).

Information sharing in the context of neural networks is typically distinguished in *soft* parameter sharing and *hard* parameter sharing (Ruder, 2017). In soft parameter sharing, each task is provided its own parameters, and related parameters from the tasks are constrained to be similar. In hard parameter sharing, hidden layers are shared between the tasks, and task-specific output layers are devised on top of shared representations. Hard parameter sharing is the most common multi-task learning approach since it substantially reduces overfitting issues by modeling representations which hold for all tasks (Ruder et al., 2019). With the advent of pre-trained language models, BERT (Devlin et al., 2019) represents a natural fit for hard parameter sharing.

Multi-task learning has also been used to provide a *main* task useful signals coming from tasks that are not of direct interest, namely *auxiliary* tasks. Specifically, the loss coming from each auxiliary task is typically given a smaller weight compared to the weight of the main task. For instance, POS tagging as an auxiliary task has shown to consistently improve the performance on chunking (Changpinyo et al., 2018) and dependency parsing (Y. Zhang & Weiss, 2016), whereas mixed results have been reported for the simultaneous learning of keyphrase extraction and chunking (Bingel & Søgaard, 2017). More in general, since multi-task learning is meant to provide additional related signals, it reduces the training data needed for the learning tasks at hand (Changpinyo et al., 2018). We employ multi-task learning in Chapter 5, where we jointly model the biomedical event extraction sub-tasks using BERT as hard parameter sharing architecture.

2.3 End Notes

For a contemporary overview of natural language processing as a field, we refer the reader to Eisenstein (2019). In-depth details of neural network methods in NLP are provided in Goldberg (2017) and Jurafsky and Martin (2020). A comprehensive deep learning overview is provided by Goodfellow et al. (2016), whereas mathematical optimization details are in Ruder (2016). The seminal paper by Pan and Yang (2010) provides an in-depth view of transfer learning, whereas a modern excursus of transfer learning in NLP is in Ruder (2019).

Sections 2.2.4 and 2.2.4.1 are based on the following scientific publication:

> **Ramponi, A.**, and Plank, B. (2020). Neural Unsupervised Domain Adaptation in NLP—A Survey. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)* (pp. 6838–6855). Barcelona, Spain (Online): International Committee on Computational Linguistics.

Chapter 3

Dependency Tree-Driven Biomedical Relation Extraction

In this chapter we focus on biomedical relation extraction aiming at cutting down the human curation efforts of the resulting semantic entity associations by providing highly-precise extractions. We propose a symbolic approach for the task, leveraging surface linguistic information and syntactic dependency tree structures. Experiments on gold-standard corpora show that the system achieves the highest precision compared with previous rule-based, kernel-based, and neural approaches, while maintaining a F1 score comparable or superior to other methods. The results are insightful and open interesting research directions we explore in the next chapters, including the need for a more expressive relational semantics formalism, and the importance of deep and transfer learning methods in mitigating linguistic variations across different domains.

This chapter is based on the following scientific publication:

Ramponi, A., Giampiccolo, S., Tomasoni, D., Priami, C., and Lombardo, R. (2020). High-Precision Biomedical Relation Extraction for Reducing Human Curation Efforts in Industrial Applications. *IEEE Access*, 8, 150999–151011. 10.1109/ACCESS.2020.3014862

3.1 Introduction and Motivation

Relation extraction has attracted a lot of interest in the BioNLP community for a variety of applications. For instance, biomedical relation extraction has been employed to dig into research questions ranging from the identification of protein-protein interactions (Szklarczyk et al., 2016; Saik et al., 2016), gene-disease associations (Zhou & Fu, 2018; Bhasuran & Natarajan, 2018; Piñero et al., 2015; Bundschus et al., 2008), adverse drug events (Tafti et al., 2017; Abacha et al., 2015), and protein subcellular localization (Binder et al., 2014). To encourage the development of highly performing relation extraction systems, in the last two decades several community challenges, namely shared tasks, have been designed (Huang & Lu, 2016).

Despite the great progress in the techniques (cf. Section 3.2), the results of relation extraction systems still need to undergo a manual scrutiny by field experts in order to make the information ready to be exploited. This resource-demanding manual scrutiny should ideally be avoided in real-world contexts, where biomedical relation extraction is the first step of a complex pipeline of biologically driven analyses which requires highly precise relations in order to produce reliable insights.[1] This is even more important because of the rapidly growing body of biomedical literature, which calls for frequent updates of the extracted evidence during a project life cycle. Highly precise relation extraction results, with a satisfactory recall,[2] are thus crucial in real-world scenarios to smoothly translate the extracted information into actionable knowledge.

In this chapter, we present a highly precise biomedical relation extraction system designed to reduce human curation efforts. Our approach is based on a sequence of NLP syntactic modules, and a novel dependency tree-based relation extraction engine that captures relations by means of syntactic rules based on

[1] In contrast to high-precision systems, high-recall systems have the advantage of being able to find most of the relevant results, at the cost of returning irrelevant ones. High recall is thus preferable in contexts in which the quantity is more important than quality, or when false positives do not involve some direct costs; however, when human resources are involved in the curation of false positives, high precision is preferable since it limits the manual efforts.

[2] We empirically define a recall score as "satisfactory" if it reaches at least 75% of the recall performance of the most performant system in terms of F_1 score. As we show in Table 3.5, our system achieves a relative recall of 77.8%, 77.1% and 85.4% on LLL, HPRD50, and IEPA, respectively, when compared to the recall of the best system on each corpus.

common linguistic patterns. The highly precise results largely limit the need for human manual curation, allowing scientists to quickly keep abreast of novel discoveries and thus to drive an effective research.[3]

The remaining part of this chapter is organized as follows. In Section 3.2 we list related work in the field. Section 3.3 describes the methods of our system, going through the syntactic preprocessing analysis and the relation extraction engine. Section 3.4 outlines the experimental setup, whereas Section 3.5 presents the results of our system, showing the quality of the method on well-established gold-standard corpora with respect to recent approaches. A detailed analysis of errors and an ablation study are discussed in Section 3.6. Finally, conclusions are in Section 3.7.

3.2 Related Work

A variety of methods has been adopted for biomedical relation extraction. These approaches can be mainly divided into three categories: rule-based methods, feature- and kernel-based methods, and neural methods. Rule-based approaches typically make use of linguistically-motivated patterns on dependency parse trees or surface words in order to capture semantic relationships. Fundel et al. (2007) showed how a small number of carefully designed rules based on the shortest dependency path (SDP) between two examined entities produces fairly good results. Yu et al. (2018) exploited dependency parse trees and a flexible pattern matching scheme, enriching the system with a decision tree classifier. Diverse syntactic and orthography features have been extensively used in feature- and kernel-based methods. Phan and Ohkawa (2016) proposed an automatic feature selection method based on the contribution levels of different feature groups, followed by a k-nearest neighbor (k-NN) classifier. A variety of kernel-based methods have been proposed too, ranging from the walk-weighted subsequence kernel (S. Kim et al., 2010) to a combination of kernels based on different parsers (Miwa et al., 2009). Other kernel-based approaches for biomedical relation extraction include a linguistic pattern-aware

[3] A docker container is available at: `https://www.cosbi.eu/research/prototypes/biomedical_knowledge_extraction`

dependency tree kernel combined with a tree kernel (Warikoo et al., 2018), a convolution tree kernel (Chang et al., 2016), and a distributed smoothed tree kernel combined with a feature kernel (Murugesan et al., 2017). In the rising wave of deep learning, Zhao et al. (2016) proposed a deep multi-layer neural network for the task. More recent neural methods use Recurrent Neural Networks (RNNs), including Bidirectional Long Short-Term Memory (LSTM) and tree LSTM networks, and Convolutional Neural Networks (CNNs). Y. Zhang et al. (2018) showed how leveraging the complementary advantages of RNNs and CNNs in a combined hybrid model improves biomedical relation extraction. Yadav et al. (2019) experimented with a bidirectional LSTM network with an attention mechanism, exploiting word sequences and the shortest dependency path between the entities, whereas H. Zhang et al. (2019) introduced a residual CNN to tackle the task. Ahmed et al. (2019) exploited a tree LSTM network using a structured attention architecture, showing how the attention mechanism improves the performance in relation extraction. A recent research line in NLP includes the Transformer, an encoder-decoder architecture which dispenses with convolutions and recurrence, being based solely on an attention mechanism (Vaswani et al., 2017). This architecture is the core of pre-trained language models such as BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019), and its adaptively pre-trained variants for biomedical texts, namely BioBERT (Lee et al., 2020) and SciBERT (Beltagy et al., 2019). Despite the recent advances in deep learning based techniques, in this work we rely on carefully designed syntactic rules on dependency parse trees in order to avoid being dependent on labeled data. The most similar approach to our work is thus represented by the work by Fundel et al. (2007).

3.3 Methods

Our approach to biomedical relation extraction includes two main steps, namely (i) text preprocessing, in which a sequence of syntactic NLP modules are applied to input texts (Section 3.3.1), and (ii) relation extraction, in which relationships between biomedical entities are identified and classified (Section 3.3.2). A schematic view of the system is presented in Figure 3.1.

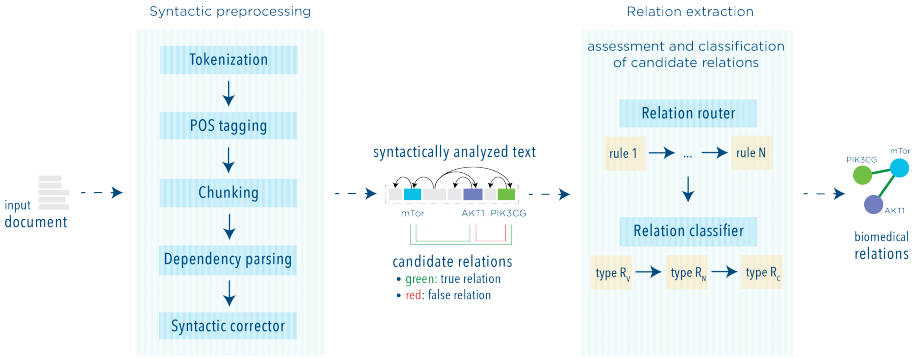


Figure 3.1: In our biomedical relation extraction approach each input document is firstly analyzed by syntactic preprocessing modules (i.e., tokenizer*, POS tagger, chunker*, dependency parser, and syntactic corrector*). The resulting syntactic dependency parse tree and token annotations, along with candidate entity pairs, are analyzed by a relation router to detect candidate relations. Actual relations are finally identified by a relation classifier by means of pattern matching rules on the dependency tree. *Custom implementation of preprocessing components. © 2020 IEEE.

3.3.1 Syntactic Preprocessing

A pipeline of syntactic modules is firstly applied to input texts. This allows the relation extraction engine to leverage syntactic information, such as part-of-speech tags and dependency trees, in order to extract semantic associations from input texts. We present the preprocessing modules in the following.

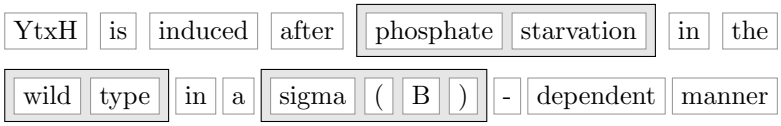
Tokenization The input text is separated into tokens using the spaCy[4] tokenizer, which we enrich with regular expressions to segment the text units also on intra-word punctuation (e.g., hyphens, slashes, etc.). This fine-grained approach to tokenization originates from the observation that not all the symbol-separated tokens are the smallest units of information to work with. For instance, "IL6-induced atrophy" is typically divided in two tokens ("IL6-induced" and "atrophy"). However, "IL6-induced" implicitly encodes relational information that is eventually desirable to analyze.

[4] `https://spacy.io/`

Part-of-speech tagging Each token is assigned a label describing its part-of-speech (POS) at two different granularities: a coarse-grained one – from the Universal POS tagset (Nivre et al., 2016) – and a fine-grained one – from the Penn Treebank tagset (Marcus et al., 1993). We use the spaCy neural model for POS tagging. These POS labels (refer to Appendix A for a complete list) serve to both the chunking and the syntactic dependency parsing steps.

Chunking Our approach to tokenization allows the system to ultimately merge only the tokens that together form a self-contained chunk of information. For instance, the tokens $T = \{(, AKT,), -, 1\}$ are part of the same concept "(AKT)-1", hence it is desirable to merge them into a single token. To the goal, we designed patterns on the orthography and on the fine- and coarse-grained POS tags of the tokens (see Table 3.1). The merged tokens store all attributes of their original constituents (e.g., POS tags, surface text, etc.).

We designed this module to reduce potential errors in syntactic parsing in case of long and articulated texts, and to easily process multi-token words (e.g., "Interleukin 6"), frequent in biomedical texts. For instance, given a list of tokens (i.e., white boxes), the following chunks (i.e., grey boxes) are produced:



The number of text units decreases from 19 to 14, allowing an easier parsing process, and multi-token words are produced. For simplicity, we hereafter refer to these text units as tokens and chunks indistinctly.

Syntactic dependency parsing A syntactic dependency parse tree of the text is built using the spaCy non-monotonic transition-based parser. We choose to rely on the spaCy parser since it has been benchmarked to be the fastest to date,[5] and thus it fully meets real-world application requirements. The grammatical dependencies (hereafter, *edges*) of the tokens or chunks (hereafter,

[5] `https://spacy.io/usage/facts-figures#benchmarks`

Table 3.1: The chunking patterns used by the system. Each token T_i in a candidate sequence $T_1...T_n$ must satisfy some matching rules on the orthography (*orth*), and coarse- and fine-grained part-of-speech (*pos* and *tag*, respectively) levels in order to be merged. Underlined tokens are the triggers of a pattern, which have to meet their restrictions in order to proceed. The rest of the sequence tokens are thus subsequently checked for matching. If a matching rule is satisfied, $T_1...T_n$ are merged into a single chunk.

Pattern name	Matching rule
PLUS_MATCHER$_1$ e.g., "CD19+", "CD20 +"	$\underline{T_1 T_2}$, where: $T_1(tag) \in \{NN, NNS, NNP, NNPS\}$ $T_2(orth) \in \{+\}$
PLUS_MATCHER$_2$ e.g., "CD19(+)", "CD20 (+)"	$\underline{T_1 T_2 T_3 T_4}$, where: $T_1(tag) \in \{NN, NNS, NNP, NNPS\}$ $T_2(tag) \in \{-LRB-\}$ $T_3(orth) \in \{+\}$ $T_4(tag) \in \{-RRB-\}$
HYPHEN_MATCHER$_1$ e.g., "IL-6", "AKT- 1"	$\underline{T_1 T_2 T_3}$, where: $T_1(tag) \in \{NN, NNS, NNP, NNPS\}$ $T_2(tag) \in \{HYPH\}$
HYPHEN_MATCHER$_2$ e.g., "(IL)-6", "(AKT) - 1"	$T_1 \underline{T_2 T_3 T_4 T_5}$, where: $T_1(tag) \in \{-LRB-\}$ $T_2(pos) \in \{PROPN\}$ $T_3(tag) \in \{-RRB-\}$ $T_4(orth) \in \{-\}$ $T_5(tag) \in \{CD\}$
ADJ_NN_MATCHER e.g., "Primary cell", "Tumor suppressor protein"	$\underline{[T_1, ..., T_k]}[T_{(k+1)}, ..., T_n]$, where: $T_{1,...,k}(tag) \in \{JJ, JJR, JJS\}$ $T_{(k+1),...,n}(pos) \in \{NOUN, PROPN\}$
ADJ_COMPOUND_MATCHER e.g., "Young adult"	$\underline{[T_1, ..., T_k]}[T_{(k+1)}, ..., T_n]$, where: $T_{1,...,n}(pos) \in \{ADJ\}$
NN_COMPOUND_MATCHER e.g., "Cell antibody"	$\underline{[T_1, ..., T_k]}[T_{(k+1)}, ..., T_n]$, where: $T_{1,...,n}(pos) \in \{NOUN, PROPN\}$

nodes) are drawn from the CLEAR tagset[6] for dependency parsing (we refer the reader to Appendix B for a comprehensive reference).

Syntactic corrector Since POS tags and grammatical dependencies are predicted, they are not always correct. Correcting POS tags or parse trees is an hard problem; however, some wrong labels can be easily detected. As a consequence, for the most trivial errors we automatically correct the labels, whereas in more complex cases we label the sentence as unreliable to avoid false positives in the relation extraction phase. The following corrections are applied:

- tokens that are heads of a direct object (*dobj*) or a nominal subject (*nsubj*) having a coarse-grained POS tag different from *verb* are assigned *verb* as a POS tag;
- tokens that are heads of an adjectival modifier (*amod*) having *verb* as a coarse-grained POS tag are assigned *adj* as a POS tag, since they are in most cases past participles used as adjectives.

3.3.2 Relation Extraction

The syntactic preprocessing provides the information needed to extract biomedical relationships between entities from text. Following previous work in biomedical relation extraction, we assume entities are given. We rely on syntactic rules, thus fully exploiting the dependency parse tree and the syntactic information encoded to each token. Our strategy involves a routing phase to detect candidate relation pairs (Section 3.3.2.1), and a classification phase to assess the actual presence of semantic relations (Section 3.3.2.2).

3.3.2.1 Relation Router

We analyze the minimum path of the dependency parse tree between entities to assess if the path is eligible for representing a candidate relation pair. We

[6] `https://github.com/clir/clearnlp-guidelines/blob/master/md/specifications/dependency_labels.md`

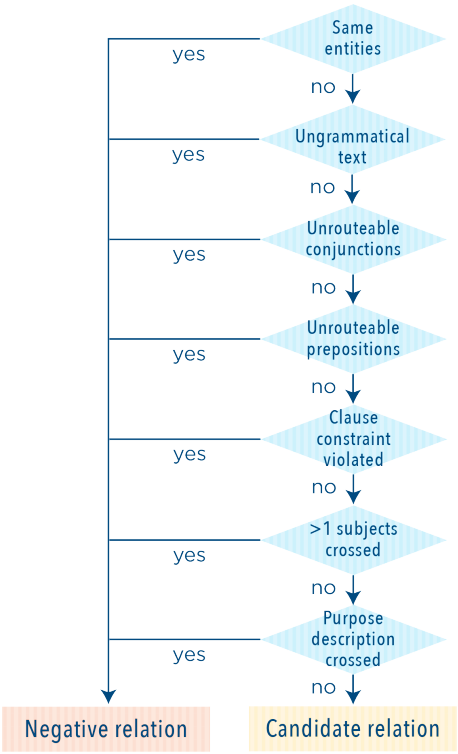


Figure 3.2: The logic of the relation router. Rhombus shapes indicate tested conditions, while arrows indicate the router flow. If all the conditions are negative, the entity pair is considered a relation candidate. Otherwise, the entity pair is labeled as a negative relation instance. © 2020 IEEE.

devise several rules to the goal, based on common linguistic constructs.[7] The process of routing a syntactic path involves both the analysis of crossed edges (i.e., dependency relations) and node attributes (e.g., coarse- and fine-grained POS tags, surface text, etc.). Figure 3.2 summarizes the workflow. In the case one of the following conditions is met, the router stops immediately labeling the candidate relation pair as negative:

[7]We took inspiration from relation filtering steps in Fundel et al. (2007), then identifying constructs after manual inspection of 25% of the examples from each dataset (see Section 3.4).

1. **Same entities.** If the lemmas of entities are the same, the candidate pair is labeled as negative;

2. **Ungrammatical text.** In the case the input text has no verb if not in subordinate clauses, the pair is considered unreliable and thus labeled as a negative instance;

3. **Unrouteable conjunctions.** If conjunctions introducing subordinate or coordinate prepositions are met (i.e., *but*, *whereas*, *if*, *therefore*, and *while*), the entities are unlikely to be related, thus the pair is negative;

4. **Unrouteable prepositions.** The prepositions *if*, *therefore*, *during*, *despite*, *from* and *at* typically introduce phrases that specify where – or when – a specific event occurs – or has occurred. If one of these prepositions is found in the path, the candidate pair is unlikely to be a relation and thus discarded (i.e., labeled as negative);

5. **Clause routing constraint.** Sentences in the biomedical literature are complex and articulated, with one or more coordinate and subordinate clauses. Entities in different clauses could be in a relation, but only under some conditions. We allow the router to cross a clause only if the target clause has no explicit subject dependency, and if the final path has exactly one subject. Otherwise, we consider the pair a negative instance;

6. **More than one subject crossed.** If more than a subject dependency relation is crossed we label the relation pair as negative, because the minimum path is typically crossing semantically independent phrases or clauses. For instance, in the sentence "A causes B and C triggers a D-reaction", the entity A is not related to the entity D;

7. **Purpose-description statements.** Some sentences express a broad research purpose (e.g., "In this paper we aim to demonstrate that tuberculosis could be prevented by prophylactic treatment"), instead of actual relations. When crossing the path between entities, the lemmas of the tokens are thus compared to a list of purpose-related words (Appendix C). If a match is found, the pair is labeled as negative.

While crossing the path, the relation router also checks if the relation is affirmed or negated. This is particularly useful to detect actual associations. Negations are detected using the following rules and lexicons (Appendix C):

1. **Negative auxiliary.** A crossed token node is incident to an edge having a negation modifier dependency tag (*neg*), or is adjacent to a token node with *no* lemma;
2. **Negative verb.** A crossed verb belongs to a lexicon of negative verbs;
3. **Negative adverb.** A crossed token node is incident to an edge having an adverb as target that belongs to a lexicon of negative adverbs;
4. **Negative noun.** A crossed noun belongs to a lexicon of negative nouns;
5. **Negative adjective.** A crossed adjective belongs to a lexicon of negative adjectives.

If the relation router navigates the whole path between the two entities without any of the routing conditions is met, the pair is considered a relation candidate and is analyzed by the relation classifier (Section 3.3.2.2).

3.3.2.2 Relation Classifier

The relation classifier analyzes the relation candidates the router detected, assigning the entities the *effector* and the *effectee* roles. We identified three categories of linguistic constructs that are typically used to express semantic relations in the English language. The categories are the following:

- **Relation expressed by a verb** (R_V). A generalized version of the *effector-relation-effectee* rule proposed by Fundel et al. (2007) that we enhanced to capture constructs of the form:

  $entity_A$*-[phrase]-verb-[phrase]-*$entity_B$

  where a *phrase* can appear zero, one, or multiple times. As a result, the rule matches elaborate statements with interleaved phrases such as

"A plays a big role in B assimilation" , and not only triples of the form *entity$_A$-verb-entity$_B$*.

- **Relation expressed by a nominalization or a participle** (R_N). Associations in the biomedical literature are often expressed by nominalizations or participles. We thus employ the following rules:

  (1) *nominalization-of-entity$_A$-by-entity$_B$*
  Example: "Activation of A by B"

  (2) *nominalization-between-entity$_A$-and-entity$_B$*
  Example: "Relation between A and B"

  (3) *nominalization-of-entity$_A$-on-entity$_B$*
  Example: "Effect of A on B"

  (4) *entity$_A$-participle-entity$_B$*
  Example: "A-dependent B"

  While rules (1) and (2) are inspired by the *relation-of-effectee-by-effector* and *relation-between-effector-and-effectee* proposed by Fundel et al. (2007), the rule (3) widens the scope of rule (1), and rule (4) allows the system to effectively handle nominalized adjectives expressing relations.

- **Relation expressed by a conjunction** (R_C). This category is designed to capture relations of entities that act together to do something, which are typically both subjects of a statement. We use the following pattern:

  entity$_A$-conjunction-entity$_B$-verb
  Example: "A and B associated"

As a result, if the path between entities contains a verb, we consider the candidate relation pair as a R_V relation. The verb found in the path is considered the verb for the relation, and if multiple verbs are found, we take the last one in the text order. To assign a role to the entities, we look at the verb voice. If the voice is active, the entity that appears first in the sentence is labeled as

the effector, while the second one is labeled as the effectee. Otherwise, the first entity is labeled as the effectee, and the second entity as the effector.

In the case no verb is found in the crossed path, but it contains (a) a past participle,[8] (b) an adjective ending in "ent" (e.g., A-dependent B), or (c) a nominalized verb, we consider the candidate pair as a R_N candidate. Additionally, we have to focus on the types of the edges crossed. During the routing we allow many edge types to be crossed, but a lot of them only exist in verb-expressed relations. Since a R_N relation represents a more compact connection between the entities, it should not contain verbs (if not the participle form), nor both subjects and objects. To model this additional restriction in terms of edge types and paths, we check whether the minimum path between the two entities only contains certain types of grammatical dependencies. Beyond links expressed by conjunctions and prepositions, only modifiers, compounds, and appositions should exist (i.e., *npmod*, *amod*, *compound*, *appos*, *punct*, *prep*, *pobj*, or *conj*). If condition (a) or (b) is satisfied, the effector and effectee roles are assigned according to the text order, whereas if condition (c) is met, roles are assigned by analyzing the preposition connecting the nominalization and the entities. Specifically, the effector is the entity that does not have a preposition *or*, *by* as ancestor, whereas the effectee is the entity that has a preposition amongst *on*, *of*, or *with* as ancestor.

If the crossed path only contains a conjunction, the remaining part of the text is analyzed for R_C relations. We check whether the top-level node of the path is incident to a verb node. In such case, we check if the verb lemma is *interact* or *form*, and if so, we consider the relation as a R_C type.[9] Note that in the R_C category the effector and effectee roles are not needed since both entities are interacting as both effectors.

Lastly, if all R_V, R_N, and R_C categories are not satisfied, the candidate relation pair is labeled as negative.

[8] This holds under some limitations: it should be incident to an edge with a *npadvmod* or an *amod* dependency relation.

[9] In contrast to R_V and R_N relation categories, we here look at the whole dependency tree, without restricting the focus to the minimum path.

Table 3.2: Statistics of the benchmark corpora. © 2020 IEEE.

	LLL	IEPA	HPRD50
Positive relations	164	335	163
Negative relations	166	482	270
Sentences	77	486	145

3.4 Experimental Setup

We evaluate our biomedical relation extraction method on different benchmark corpora annotated for biomedical relations: LLL (Nédellec, 2005), IEPA (Ding et al., 2001), and HPRD50 (Fundel et al., 2007). The corpora are about different topics in biomedicine, thus they represent a good evaluation benchmark for our system for diverse real-world applications. In particular, LLL is a corpus about the model bacterium *Bacillus subtilis*, focused on gene transcription and sporulation; HPRD50 is about regulatory relations, direct physical interactions and modifications on documents from the Human Protein Reference Database (Peri et al., 2004); and IEPA is a corpus focused on interactions between a restricted set of biochemicals (e.g., insulin, oxytoxin, leptin, etc.). Further details on data creation and annotation are in Appendix D. Relations between entities are annotated within the sentence boundaries, and entities offsets are provided with the raw texts. Given a set of entities $\{e_1, e_2, ..., e_n\} \in E$ belonging to an input sentence S, we generate $\binom{n}{2}$ candidate relation instances (if $n \geq 2$) for the sentence S. Following previous work, negative instances are represented by pairs that are not annotated as relations in the corpora. The statistics of the corpora are summarized in Table 3.2.

For the sake of comparison to previous work, we evaluate our relation extraction method using precision, recall, and F1 score. We compared our method to existing methods in literature, including rule-based approaches (Fundel et al., 2007; Yu et al., 2018), feature- and kernel-based approaches (Phan & Ohkawa, 2016; S. Kim et al., 2010; Miwa et al., 2009; Warikoo et al., 2018; Chang et al., 2016; Murugesan et al., 2017), and neural network approaches (Zhao et al., 2016; Y. Zhang et al., 2018; Yadav et al., 2019; H. Zhang et al., 2019; Ahmed et al., 2019). Additionally, we compared our system

to recent transformer-based methods pre-trained on biomedical texts, namely BioBERT (Lee et al., 2020) and SciBERT (Beltagy et al., 2019). We fine-tuned both BioBERT and SciBERT on each corpus, reporting the average performance using 10-fold cross validation. We used the official implementation and optimal hyper-parameters provided by the respective authors (Lee et al., 2020; Beltagy et al., 2019).

3.5 Empirical Results

Table 3.3 shows the performance of our system across corpora compared to other methods. Our system achieves the highest precision on all the corpora (93.2%, 90.7%, and 91.7% on LLL, HPRD50, and IEPA, respectively), outperforming by a large margin the BERT-based approaches in the precision metric while maintaining a F1 score comparable to other methods. The only exception is on the LLL corpus, where transformer-based methods and the "DSTK & feature kernel" approach achieve a very high F1 score. It is worth noting that our relation extraction approach also achieves the highest F1 score (81.1%) on the IEPA corpus. These results strongly meet our expectations, since our goal was developing a high-precision system to allow researchers, and in particular biologists, to obtain reliable information without having to manually review the results and discard all the false positive instances. The small size of corpora suggests that the superiority of our proposed method over deep learning approaches is confirmed when few annotated data is available. As future work, we aim to further investigate this advantage in higher-resource regimes.

3.6 Analysis and Discussion

In this section we present a detailed analysis of the errors (Section 3.6.1) and an ablation study to investigate the contribution of each rule component (Section 3.6.2).

Table 3.3: Performance of our system with other approaches on benchmark corpora. Precision (P), Recall (R), and F1 score (F1) are shown by percentage rounded with a single decimal. Best results for each metric are highlighted in bold. *Original implementation we fine-tuned on each corpus.

Method	LLL			HPRD50			IEPA		
	P	R	F1	P	R	F1	P	R	F1
Fundel et al. (2007) *Dependency tree rules*	79.0	85.0	82.0	79.0	78.0	78.0	-	-	-
Yu et al. (2018) *GRGT (rules & decision tree)*	91.2	77.1	83.6	86.5	50.8	64.0	91.0	63.6	74.9
Phan and Ohkawa (2016) *O2G (feature selection & k-NN)*	74.7	82.2	76.5	73.0	74.3	72.6	68.1	71.3	69.5
S. Kim et al. (2010) *Walk-weighted subsequence kernel*	76.9	91.2	82.4	66.7	69.2	67.8	73.8	71.8	72.9
Miwa et al. (2009) *Multiple kernels*	77.6	86.0	80.1	68.5	76.1	70.9	67.5	78.6	71.7
Warikoo et al. (2018) *LPTK (patterns & tree kernel)*	78.9	72.1	75.3	72.7	62.2	67.1	74.8	66.1	70.2
Chang et al. (2016) *Convolution tree kernel*	73.2	89.6	80.6	63.8	81.2	71.5	62.5	83.3	71.4
Murugesan et al. (2017) *DSTK & feature kernel*	87.3	91.2	89.2	76.3	84.2	80.0	75.9	85.2	80.2
Zhao et al. (2016) *Deep neural network*	80.7	84.4	81.0	58.7	92.4	71.3	68.7	83.5	74.2
Y. Zhang et al. (2018) *Combined RNN and CNN*	76.6	96.1	85.2	75.1	76.4	75.6	73.2	84.4	78.2
Yadav et al. (2019) *SDP-based Bi-LSTM w/ attention*	84.2	83.6	83.9	79.9	77.6	78.7	77.0	75.6	76.3
H. Zhang et al. (2019) *Residual CNN*	80.5	87.2	83.2	74.9	82.8	77.7	71.6	80.6	75.5
Ahmed et al. (2019) *Tree LSTM*	85.3	84.9	84.8	82.4	82.8	82.0	77.0	76.7	76.4
Ahmed et al. (2019) *Tree LSTM w/ structured attention*	84.8	84.3	84.2	81.7	82.3	81.3	78.6	78.7	78.5
Lee et al. (2020) *BioBERT**	87.0	91.8	89.4	83.1	81.8	**82.5**	79.7	82.2	80.9
Beltagy et al. (2019) *SciBERT**	87.3	94.0	**90.5**	79.1	76.9	78.0	81.6	77.7	79.6
Ours	**93.2**	73.1	81.9	**90.7**	70.8	79.5	**91.7**	72.8	**81.1**

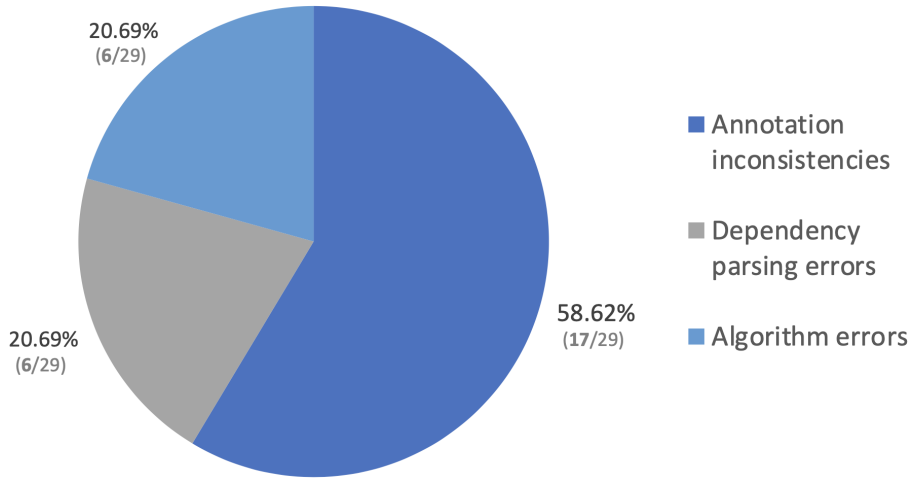


Figure 3.3: Distribution of sources of FP errors across corpora. © 2020 IEEE.

3.6.1 Error Analysis

To get additional insights on our approach, we analyzed the false positives and false negatives produced by the system in order to make room for future work.

False positives analysis. We identified three main sources of false positives (FPs), which are presented in the following and summarized in Figure 3.3:

- **Annotation inconsistencies.** Most false positive results (i.e., 58.62%) are caused by annotation inconsistencies in the corpora. We found sentences in which a relation, on a grammatical basis, actually exists, but it has not been annotated. For instance, in the following sentence:

  > *Several distinct mutations in exon2 of* ***VHL*** *disrupt binding of* ***pVHL*** *to TBP-1.* (Corpus: HPRD50, sentence ID: d26)

  a relation between "VHL" and "pVHL" has not been annotated even if it is stated in the text. This could be due to the complex mutation statement that the utterance describes, which may be better modeled as a biomedical event (Ananiadou et al., 2010);

- **Dependency parsing errors.** The 20.69% of false positives is due to errors in the dependency parse tree. For instance, in the sentence:

  > *A low level of GerE **activated transcription** of cotD by sigmaK RNA polymerase in vitro, but a higher level of GerE repressed cotD transcription.*
  >
  > (Corpus: LLL, sentence ID: d18)

  the verb "activated" has *amod* as the head relation label, denoting it is the adjectival modifier of "transcription". As a result, the system wrongly identifies a relation between "sigmaK" and the last occurrence of "cotD";

- **Algorithm errors.** Other sources of errors account for the 20.69% of the total, and are mainly due to articulated syntactic structures that our algorithm wrongly navigates. For example, in the following sentence:

  > *These results clearly demonstrate that **UCP3** gene expression is upregulated by TZDs in the WAT and BAT in Wistar fatty rats, an obese model with **leptin** receptor defect, and that adipose UCP3 gene expression is increased in response to TZDs in vitro.* (Corpus: IEPA, sentence ID: d88)

  our system incorrectly identifies a relation between the biomedical entities "UCP3" and "leptin".

False negatives analysis. False negatives (FNs) can be classified according to both the relation category they have been tested on, and their cause. Figure 3.4 summarizes the distribution of false negatives according to this classification. Particularly, the 87.50% of false negatives belong to the R_V category, the 11.03% belong to the R_N category, and the 0.74% fall into the R_C category. The remaining 0.74% are cases that do not belong to any of the previous categories. For each category, the causes we identified are the following:

- **Dependency parsing errors.** In the 64.71% of the total cases, false negatives are caused by errors in the dependency tree of the sentence being analyzed. For example, in the following sentence:

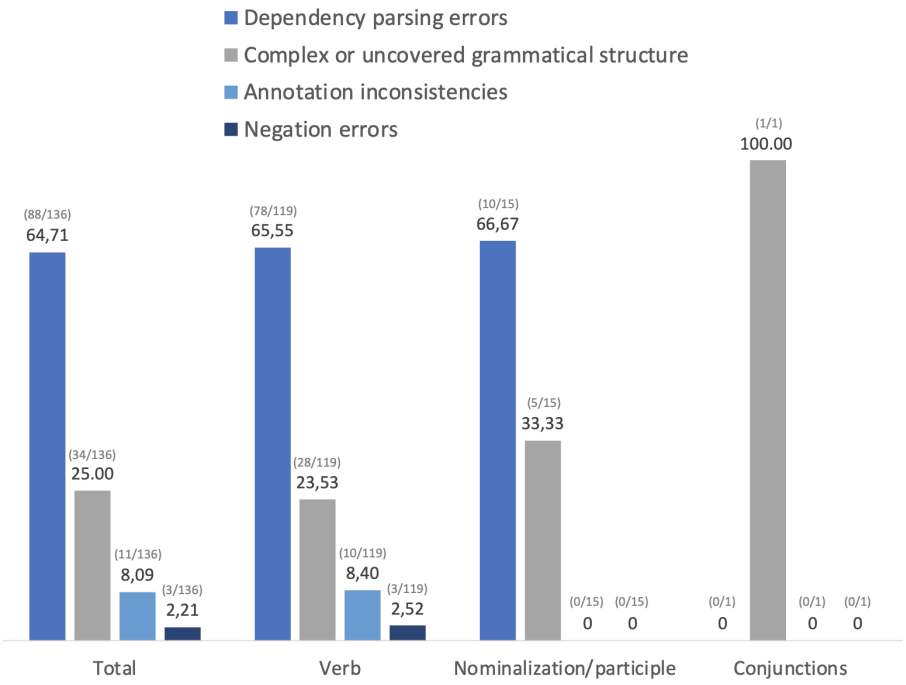


Figure 3.4: Distribution of sources of FN errors across corpora, according to both relation categories and their causes. The remaining 0.74% (1/136) are cases that do not belong to any category. © 2020 IEEE.

> *We have shown previously that the transcription of degR is driven by an alternative sigma factor,* ***sigmaD****.*
>
> (Corpus: LLL, sentence ID: d26)

"sigmaD" is labeled as an appositional modifier (i.e., *appos*) of the verb "shown"; however, its head should instead be "factor". This results in a wrong structure that prevents our algorithm to correctly navigate the tree. We found this kind of error particularly prominent within the R_V category (i.e., 65.55%) and the R_N category (i.e., 66.67%). No errors of this kind are found in R_C;

- **Complex or unconvered grammatical structure.** In the 25.00% of

the cases, the grammatical structure of the sentence has more than one subordinate or coordinate clause, and it is not easy to route. To give an example of this latter case, we can look at the following sentence:

> ***SpoIIID*** *at low concentration repressed cotC transcription, whereas a higher concentration only partially repressed* ***cotX*** *transcription and had little effect on cotB transcription.*
> (Corpus: LLL, sentence ID: d27)

where to identify the actual relation between "SpoIIID" and "cotX", the system should be able to figure out that "higher concentration" is actually referring to "SpoIIID". However, this is far beyond the capabilities of our algorithm. While this false negative cause accounts for all the error within the R_C category, it only accounts for the 23.53% and the 33.33% within the R_V and R_N categories, respectively;

- **Annotation inconsistencies.** Similarly to the false positive analysis, false negatives could also be due to annotation inconsistencies. These errors account for the 8.09% of the total false negatives, and an example of this error type is exemplified by the following sentence:

  > *The aim of this study was to investigate the effects of hCG, hCG plus oxytocin and* ***oxytocin*** *on [3H]* ***inositol phosphate*** *(IP) formations in porcine myometrial cells obtained from ovariectomized and cyclic gilts.* (Corpus: IEPA, sentence ID: d17)

  where "oxytocin" and "inositol phosphate", following the annotation standards of the corpora, are not actual relations, but instead statements about the purpose of the study. Fortunately, these errors are not common, representing only the 8.40% of the total errors in R_V;

- **Negation errors.** The remaining false negatives (i.e., 2.21%) are due to errors by our negation detector. For instance, in the sentence:

  > *From these results we conclude that* ***ComK*** *negatively regulates* ***degR*** *expression by preventing sigmaD-driven transcription of*

degR, possibly through interaction with the control region.
(Corpus: LLL, sentence ID: d26)

our system misses the relation between "ComK" and "degR". This is due to difficulties in discerning negated relations from negative relations. This error type is only present within the R_V category, accounting for a relative amount of 2.52% of the errors.

3.6.2 Ablation Study

In order to provide additional insights on our method, we investigate the contribution of each rule category on the final performance of the system. In Table 4.5 we report precision, recall and F1 score on all corpora when R_V, R_N, R_C, and negation rule components are individually removed. As expected, the negation rules are useful to the precision of the relation extraction system. In fact, when removed the precision score decreases on all the corpora (-3.1%, -8.3%, and -6.6% on LLL, HPRD50, and IEPA, respectively). We also notice a small increase in the F1 score on the LLL corpus (+2.0%). This is due to the characteristics of LLL, which exhibits few negated relations with respect to the other corpora. When removing R_V, R_N, and R_C, we obtain deeper insights about the importance of each relation category. For instance, the relation expressed by a verb (R_V) is by far the most important rule set. When removed, the precision increases on all the corpora (+6.8%, +3.7%, and +2.2% on LLL, HPRD50, and IEPA, respectively), while an important decrease appears evident in the recall metric (-58.0%, -58.4%, and -49.1% on LLL, HPRD50, and IEPA, respectively) and thus in the F1 score. This behaviour confirms that the R_V category is the primary source of errors of our system, but also the mean of a tradeoff between a very high precision and a satisfying recall. We notice a similar but less pronounced trend when removing relations expressed by nominalizations or participles (R_N). On the other hand, the category of relations expressed by conjunctions (R_C) contributes a little on all corpora. Particularly, it improves the precision (+0.1%), the recall (+0.7%), and the F1 score (+0.5%) on HPRD50, whereas it decreases the precision (-0.4%) and the F1 score (-0.2%) on IEPA.

Table 3.4: Ablation study on the contribution of each rule type. We report precision (P), recall (R), and F1 score (F1) of our biomedical relation extraction approach on all the corpora when each rule category is removed.

Corpus	Configuration	P	R	F1
LLL	**Complete rule set**	93.2	73.1	81.9
	– R_V category rules	100.0	15.1	26.2
	– R_N category rules	91.5	58.1	71.1
	– R_C category rules	93.2	73.1	81.9
	– Negation rules	90.1	78.5	83.9
HPRD50	**Complete rule set**	90.7	70.8	79.5
	– R_V category rules	94.4	12.4	21.9
	– R_N category rules	90.6	63.5	74.7
	– R_C category rules	90.6	70.1	79.0
	– Negation rules	82.4	71.5	76.6
IEPA	**Complete rule set**	91.7	72.8	81.1
	– R_V category rules	93.9	23.7	37.9
	– R_N category rules	91.5	50.2	64.8
	– R_C category rules	92.1	72.8	81.3
	– Negation rules	85.1	73.2	78.7

3.7 Summary and Conclusions

In this chapter we presented a high-precision relation extraction system aiming to speed up the time-consuming process of the manual curation of semantic biomedical associations. Experimental results on gold-standard corpora showed that our method outperforms existing rule-based, feature- and kernel-based, and neural-based biomedical relation extraction approaches on the precision metric, while reaching a comparable or superior F1 score. As a result, we met the requirement of limiting the expensive curation of semantic biomedical relationships to smoothly and reliably translate the extracted information into actionable knowledge. Besides the good results, this work offered useful insights and opened interesting research directions:

Expressivity of relation extraction Our system is able to extract highly precise binary relations, however there are use cases in which it would be useful to extract higher-order associations, making a relation the argument of another relation, or modeling relations with more than two arguments. This motivated us to focus on biomedical event extraction in the next chapters (Chapter 5 and 6), a more complex yet more powerful NLP task for capturing fine-grained semantic associations between biomedical entities from unstructured texts.

Deep learning for domain robustness We designed rules as general as possible, relying on syntactic information thus avoiding to overfit to words or corpus-specific constructs. However, as presented in Section 3.6, there are cases the system still does not capture due to the high variability of the biomedical texts and their articulated syntactic structures. Despite this limitation is little pronounced in relation extraction, it becomes important in more complex structured tasks such as event extraction, where symbolic methods have shown to be fairly limited in performance (J.-D. Kim et al., 2011). Deep learning methods have shown to be effective in learning complex patterns from data, particularly being more robust across domain variations compared to traditional methods in relation extraction (T. H. Nguyen & Grishman, 2015). We specifically investigate the robustness of deep learning methods across domain varieties in the next chapter for the task of edge detection in biomedical event extraction (Chapter 4).

Chapter 4

On Domain Shift in Biomedical Event Extraction

In this chapter we shift to biomedical event extraction, a richer NLP task for capturing semantic relationships, and we focus on robustness in face of linguistic domain variations. As a case study, we experiment on the most challenging step of the event extraction pipeline, namely edge detection. We contribute to the field by presenting the first cross-domain study of edge detection in biomedical event extraction, which highlights the importance of linguistic variations in the different corpora to be tackled in future work. We also encourage future research in cross-domain edge detection for event extraction pipelines by releasing a standardized benchmark dataset and a baseline neural model to the purpose. We conclude by outlining future directions in the field, while gaining the motivation for end-to-end biomedical event extraction and the use of novel transfer learning methodologies.

This chapter is based on the following scientific publication:

> **Ramponi, A.**, Plank, B., and Lombardo, R. (2020). Cross-Domain Evaluation of Edge Detection for Biomedical Event Extraction. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)* (pp. 1982–1989). Marseille, France: ELRA.

4.1 Introduction and Motivation

Biomedical event extraction is a crucial task in order to automatically extract information from the increasingly growing body of biomedical literature. Specifically, the goal of biomedical event extraction is to extract semantically rich, structured information from unstructured texts. Unlike traditional relation extraction, event representations can capture the association of one or more participants in different semantic roles, where each association in turn can be argument of higher-level associations (see Chapter 2).

Recent studies on biomedical event extraction have shown that supervised machine learning approaches and in particular neural methods provide state-of-the-art performance on the task (Björne & Salakoski, 2018; Li et al., 2019). However, a current limitation of machine learning is that models are trained under the implicit hypothesis that the test data (i.e., the *target*) follows the same underlying distribution of the training data (i.e., the *source*). In practice, this translates to a dramatic drop in performance when the model is applied – or evaluated – *into the wild* (i.e., *out-of-domain*), due to the differences of source and target corpora. This is the case of biomedicine, a field that is often seen as a domain per se, but instead comprises a lot of sub-domains, from molecular biology to genetics and physiology.

As a first step towards the open problem of out-of-domain generalization in machine learning (Bengio, 2019), in this Chapter we provide the first cross-domain evaluation study in the context of biomedical event extraction. Since biomedical event extraction is typically framed as a multi-stage task by means of classifier pipelines (Björne & Salakoski, 2018; Li et al., 2019), this evaluation could be carried out at each stage, similarly to what happens in the non-biomedical ACE event extraction challenge (Walker et al., 2006). However, the formal evaluation of biomedical event extraction is based on events as monolithic units by submitting predictions to shared task online evaluation services. This has three main consequences: (i) test data is blind and is meant for the evaluation of entire events, thus it cannot be used for intermediate stages; (ii) previous work using classifier pipelines report final scores only, making it difficult to evaluate and interpret how well the stages perform in isolation; and

(iii) even if those performances were reported, results are incomparable due the different preprocessing conditions, such as the generation of negative examples, and experimental setups.

In this work we focus on the edge detection stage of biomedical event extraction since we believe it represents the most important module of the event extraction pipeline. In fact, in addition to being a step where both input and output data are not explicitly available, we argue that it shares most of the incomparability issues with relation extraction, including the number of negative examples one could generate, and the independence of training and test data with regard to the examples within the same sentences (Pyysalo et al., 2008).

In the absence of explicit data and common means to evaluate edge detection in biomedical event extraction, we contribute to the field and provide (i) standardized training and test data for edge detection for five different gold-standard corpora enabling cross-domain experimentation, together with a characterization of the differences between the corpora; (ii) a model for edge detection based on recent advances in neural methods, setting it as a strong baseline for future research; and (iii) a thorough experimentation of edge detection in a cross-domain setting, quantifying the drop in performance of baseline models. To the best of our knowledge, we are the first to provide such insights. We thus believe our work could encourage future research on out-of-distribution generalization for biomedical event extraction, as well as in-depth evaluation of other stages. To the goal, we make both the data and the baseline available to the research community.[1]

The remaining part of this Chapter is organized as follows. Section 4.2 presents related work in the area. In Section 4.3 we present the differences in language aspects of biomedical event extraction corpora, and the methods to create a standardized benchmark dataset for edge detection. Section 4.4 describes the baseline method for the task, detailing the experimental setup and providing an ablation study. Section 4.5 presents the cross-domain results of the baseline along with a thorough discussion, whereas Section 4.6 outlines the conclusions.

[1]The data, splits and source code are available at `https://www.cosbi.eu/cfx/9985`

4.2 Related Work

In recent years, a number of shared tasks have been organized to promote the development of techniques for biomedical NLP, providing annotated corpora and evaluation means (Huang & Lu, 2016). Of particular interest for biomedical event extraction are the Genia corpus (GE11) (J.-D. Kim et al., 2011), which is the standard benchmark for assessing advances in the field, ID11 (Pyysalo et al., 2011), EPI11 (Ohta et al., 2011), PC13 (Ohta et al., 2013), and MLEE (Pyysalo et al., 2012). Several techniques have been employed for the task, ranging from rule-based to machine learning based systems (Vanegas et al., 2015). Recently, neural methods have shown to provide state-of-the-art performance on the task, using either a pipeline of Convolutional Neural Networks (CNNs) (Björne & Salakoski, 2018) or Long Short-Term Memory networks (Li et al., 2019).

Despite the efforts, biomedical corpora comprise many textual variations. According to the language variety space proposed by Plank (2016), each corpus could be characterized by several factors, including the *topic*, the *genre*, and the *language* used, amongst others. Although out-of-distribution generalization has received increasing importance in other fields, little and scattered work has been done so far in biomedical event extraction. Vlachos and Craven (2012) showed that a simple supervised domain adaptation approach (Daumé III, 2007) is beneficial in handling the differences between abstracts and full-texts, i.e., what we hereafter refer to as textual *scope*, in GE11. However, their work assumed labeled data is available in the target domain, and that the textual scope is the only source of language variation. T. H. Nguyen and Grishman (2015) conducted experiments in the newswire domain, showing that CNNs without any external features are more robust than other statistical approaches for the trigger detection stage. Miwa and Ananiadou (2015) integrated weighting and covariate shift into their system showing how these methods could improve recall at the cost of precision, whereas Miwa et al. (2013) proposed a multi-corpus learning approach combining semantic annotations shared across corpora, heuristically filtering corpus-specific annotation instances. Although these works are the closest to our goal, data and performance evaluation re-

sults for edge detection in isolation are not available. Further, since our goal is to provide standardized data and baseline models to enable future research on cross-domain generalization for edge detection on unseen and unannotated domains, we expressly avoid to filter likely spurious negative edge instances based on the knowledge of instances in other corpora (see Section 4.3.2).

4.3 Data

In this section we present the corpora used in this study, the commonalities and differences in their language aspects, as well as how we use them to generate standardized data for edge detection to enable cross-domain experimentation.

4.3.1 Corpora and Linguistic Variations

We focus on five biomedical corpora annotated for event structures. These corpora are the standard Genia benchmark GE11, ID11, EPI11, PC13, and MLEE. While the first four originate from shared tasks, MLEE results from an independent effort towards the annotation of events at various levels of the biological organization. All the corpora share the *genre* and the *language* aspects, since they all derive from scientific publications in English taken from PubMed (Canese & Weis, 2013) and PMC (Maloney et al., 2013).

Despite these commonalities, the corpora differ along many other language aspects. The main aspects – or linguistic variations – we examine are the *sub-domain* and the textual *scope* (Table 4.1). The sub-domain is the subject topic the corpus belongs to. It has been previously shown that different sub-domains exhibit different vocabulary, syntax, as well as discourse and sentential features (Lippincott et al., 2011). The sub-domain is a fuzzy aspect, since documents could span different topics with various degrees, and it is implicitly induced in the data collection step of the corpus creation. For example, all the five corpora under consideration are sampled according to different research questions, resulting in different sub-domains. Further, the textual scope of the documents in the corpora introduces another important variation, since abstracts and full-texts noticeably differ in content and structure (Cohen et al., 2010). While EPI11, PC13, and MLEE consist of abstracts only, ID11

Table 4.1: The linguistic aspects (or variations) in the corpora.

corpus	scope	sub-domain
GE11	full-texts, abstracts	reactions about transcription factors in human blood cells
ID11	full-texts	mechanisms of infectious diseases in two-component regulatory systems
EPI11	abstracts	epigenetic change events and post-translational modifications
PC13	abstracts	reactions about a selection of pathway models from BioModels and PantherDB
MLEE	abstracts	angiogenesis (i.e., formation of new blood vessels from pre-existing ones)

comprises full-text documents, and GE11 comprises both abstracts and full-texts. We present the statistics of the corpora in Table 4.2. Further details on data creation and annotation are in Appendix D.

These differences, reported in Table 4.1, show that there is no corpus that shares more than one language aspect with another one. The language variation among corpora provides the motivation for conducting a thorough cross-domain study for edge detection.

4.3.2 Data for Edge Detection

Candidate edge examples are required in order to train and evaluate an edge detector. However, since the standard evaluation in biomedical event extraction is about the final event structures, edges are not explicitly evaluated. Additionally, test data annotations in shared tasks corpora are blind. Thus, except for the MLEE corpus, we can only use the *train* and *dev* portions (see Table 4.2) to the purpose. Moreover, there are multiple ways one can generate negative examples, leading to incomparability issues among individual efforts (Pyysalo et al., 2008). In this section we outline how we dealt with this problem, creating standardized benchmark data from all five corpora. As corpora are stand-off annotations, we provide unified preprocessing, i.e., we devise the extraction of edges from event structures and the mapping to unified edge types (see "Pre-

Table 4.2: Statistics of the corpora. In addition to abstracts, GE11 includes full-text documents too (train: 5/805; development: 5/155; test: 4/264), whereas all documents in ID11 are full-texts. Density values are also indicated (S/D: average number of sentences per document; W/S: average number of words per sentence; E/S: average number of events per sentence).

corpus	set	docs	sents	words	events	S/D	W/S	E/S
GE11	train	805	8,656	205,729	10,310	10.75	23.77	1.19
	dev	155	2,888	64,132	3,250	18.63	22.21	1.13
	test	264	3,351	79,047	4,487	12.69	23.59	1.34
ID11	train	15	2,484	74,439	2,088	165.60	29.97	0.84
	dev	5	709	21,225	691	141.80	29.94	0.97
	test	10	1,925	57,489	1,371	192.50	29.86	0.71
EPI11	train	600	5,720	127,312	1,852	9.53	22.26	0.32
	dev	200	1,975	43,497	601	9.88	22.02	0.30
	test	400	4,132	82,819	1,261	10.33	20.04	0.31
PC13	train	260	2,525	53,811	5,992	9.71	21.31	2.37
	dev	90	869	18,579	2,129	9.66	21.38	2.45
	test	175	1,714	35,966	4,004	9.79	20.98	2.34
MLEE	train	131	1,271	27,875	3,296	9.70	21.93	2.59
	dev	44	457	9,610	1,175	10.39	21.03	2.57
	test	87	880	19,103	2,206	10.11	21.71	2.51

processing of event structures") and the generation of negative examples (see "Generation of negative examples"). In order to allow for the creation of a wide-coverage benchmark, we propose to focus on the most widely used edge types across all corpora (see "Merging of under-represented classes").

Preprocessing of event structures Each document in a corpus is accompanied by two annotation files, one for entities and one for both triggers and event structures. Since an edge is a subset of an event and its endpoints could be both triggers and entities, edges are implicitly encoded in both annotation files. We thus used these files in order to divide event structures into a

set of intra-sentence edge examples.[2] We use the scispaCy model with custom postprocessing rules for sentence segmentation (Neumann et al., 2019). Similarly to Miwa et al. (2013), we also handle name variations on the labels that refer to the same edge type,[3] mapping them to their canonical type (e.g., $\{Theme, Theme2, Theme3\} \mapsto Theme$). Due to both the differences in the topic of texts – thus, in the provided edge annotations – and the goal of a cross-domain study, we retain all the semantic edge types that are annotated in multiple corpora. These edge types are `Theme`, `Cause`, `Site`, `CSite`, `AtLoc`, `ToLoc`, and `FromLoc`. For the grouping of these edges refer to "Merging of under-represented classes" below, while for the formal definition of the edge types refer to the original publications of corpora.

Generation of negative examples For each sentence in the corpus, we generate edge pairs from each trigger to each of its potential arguments (i.e., triggers or entities). Similarly to previous work (Björne & Salakoski, 2015), we limit the generation of candidate edges to valid edges only, as defined in the guidelines of each corpus. This yields candidate pairs that are useful for learning, and avoids a highly unbalanced distribution of negative examples with respect to positive examples.[4] Then, each candidate edge which does not have a gold annotated type (e.g., `Theme`, `Cause`, etc.) is labeled as a `NoEdge` type (i.e., a negative instance). In the case an edge type is not among the overlapping edge types in the corpora (see "Preprocessing of event structures"), we discard the instance. As a result, we obtain a dataset of candidate edges for each corpus that can be used for training and testing.

Merging of under-represented classes Some classes are highly under-represented. For instance, in the ID11 training set, there is only one instance (0.02%) for both `AtLoc` and `ToLoc` edges, and there are no `AtLoc` instances at all in the dev set. In the same corpus, no `CSite` instances are present in the

[2]Recent work show performance degradation due to an high number of false positives on systems dealing with inter-sentence edges (Lever & Jones, 2016).

[3]Name variations on the labels are used in the corpora to arbitrarily enumerate multiple arguments of the same type starting from the same trigger.

[4]For instance, a `Binding` trigger cannot have a `Phosphorylation` as a `Theme` edge, thus the pair `(Trigger:Binding, Argument:Phosphorylation)` is not produced.

training set, while one instance (0.04%) is present in the dev set. In the MLEE training and dev sets, there are only 8 instances (0.04%) of `FromLoc` edges, and 5 (0.05%) in the test set. In general, `Site`, `CSite`, `AtLoc`, `ToLoc`, and `FromLoc` are minority classes which are difficult to learn due to the few number of examples. While for `Site` the problem is less pronounced, accounting on average for 3.16% of the edges among corpora, other edges are more problematic. On average, in the training sets there are 0.13% of `CSite`, 0.26% of `AtLoc`, 0.15% of `ToLoc`, and 0.08% of `FromLoc` instances. Since all these edges encode a similar semantic meaning of location, we created a new `Location` class, mapping them into it (i.e., $\{Site, CSite, AtLoc, ToLoc, FromLoc\} \mapsto Location$). This strategy overcomes the learning issues from under-represented classes and provides a mean for cross-domain experimentation, since all corpora now have a `Location` type.

Edge statistics across corpora The final statistics of the edges in all the corpora are presented in Table 4.3. The instances in the *train/dev* set are the ones generated from the original training set of the respective corpus, while the *test* instances are the ones coming from the original development set, since no event annotations are provided in the test sets. For the MLEE corpus, where test set annotations are available, the *train/dev* and the *test* sets reflect the original splits of the corpus.

4.4 Experiments

In this section we present the baseline model used in our experiments, the experimental setup, including the tuning of the hyper-parameters, and an ablation study to investigate the importance of different input embeddings.

4.4.1 Model Overview

We cast the edge detection problem as a multi-class classification problem where the labels to be predicted are `Theme`, `Cause`, `Location`, and `NoEdge`. We employ a CNN architecture as our framework, following its recent success in biomedical event extraction (Björne & Salakoski, 2018). Since our goal is to

Table 4.3: Statistics of edges in all the corpora in the newly created training/development (*train/dev*) and *test* sets.

Corpus	Set	Edges	Theme	Cause	Location	NoEdge
GE11	train/dev	26,718	9,027	1,082	487	18,122
			31.43%	3.77%	1.70%	63.10%
	test	9,083	2,905	442	181	5,555
			31.98%	4.87%	1.99%	61.16%
ID11	train/dev	6,430	1,270	212	28	4,920
			19.75%	3.30%	0.44%	76.52%
	test	2,805	453	113	21	2,218
			16.15%	4.03%	0.75%	79.07%
EPI11	train/dev	4,410	1,578	145	582	2,105
			35.78%	3.29%	13.20%	47.73%
	test	1,502	518	53	188	743
			34.49%	3.53%	12.52%	49.47%
PC13	train/dev	24,327	4,958	1,834	286	17,249
			20.38%	7.54%	1.18%	70.90%
	test	8,809	1,782	635	98	6,294
			20.23%	7.21%	1.11%	71.45%
MLEE	train/dev	19,903	3,482	1,001	219	15,201
			17.49%	5.03%	1.10%	76.38%
	test	9,415	1,688	466	91	7,170
			17.93%	4.95%	0.97%	76.16%

provide a baseline model for future research on cross-domain generalization for edge detection, we based architectural choices (e.g., number of convolutional layers) on recent work (T. H. Nguyen & Grishman, 2015), whose proposed network has been shown to provide an higher cross-domain robustness compared to traditional methods for relational semantics tasks. A graphical overview is presented in Figure 4.1. The neural network is composed of an input layer, a convolutional layer, a max-pooling layer, and a classification layer. To introduce a non-linearity, we use the ReLU activation function at each layer, except the output layer, which uses softmax. Given a sentence S containing a candi-

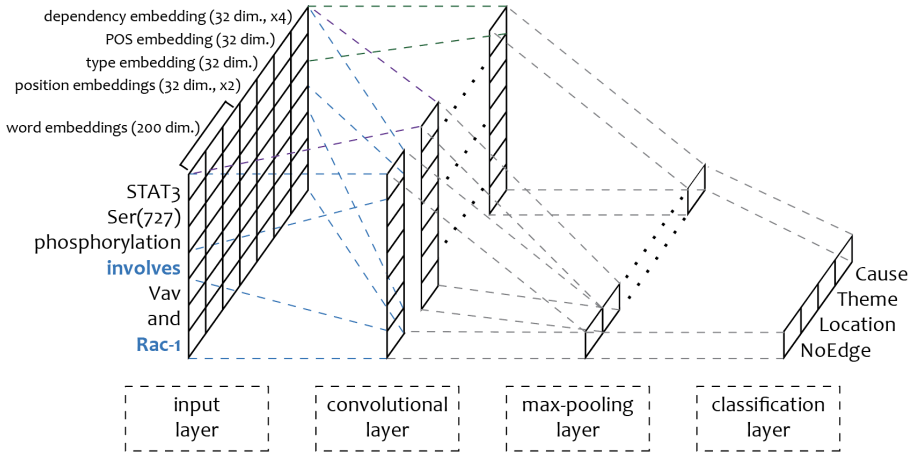


Figure 4.1: Architecture of the baseline model for biomedical edge detection. Tokens are turned into real-valued representations which are concatenated at the input layer. A convolutional layer and a max-pooling layer extract and summarize features, whereas the 4-class classification is done after the fully-connected classification layer.

date edge, an example is modeled as its sequence of tokens $\{w_i, ..., w_n\} \in S$. Each token w_i is turned into a real-valued, vectorial representation x_i representing its different syntactic and semantic characteristics. This token-wise representation is the result of the concatenation of different embeddings:

- **Word embedding**: a vectorial representation for the token from pre-trained word embeddings resulting from millions of PubMed abstracts, PMC full-texts, and English Wikipedia texts (Pyysalo et al., 2013). Out-of-vocabulary tokens are randomly initialized;
- **Position embedding**: a vector encoding the relative position of the current token from each target (Zeng et al., 2014). Since the targets are two – the source and the target of the edge to guess – two embeddings are used, one for the source and one for the target;
- **Type embedding**: a vector for the trigger type (or the named entity

type) associated with the token, available in the gold annotations of the corpora;

- **POS embedding**: a vector for the part-of-speech tag the token is assigned. We predict POS tags using a biomedical model trained on GENIA 1.0 Treebank and OntoNotes 5.0 (Neumann et al., 2019) (Appendix A);

- **Dependency embedding**: we use the path embeddings by Björne and Salakoski (2018), encoding the shortest undirected dependency path from each token to the source and target tokens of the edge candidate. We set the path depth to 2, since a depth $d > 2$ has been reported to hurt the performance in edge detection. As a result, we employ a total of four embeddings (one for the source and one for the target, both at a path distance 1 and 2). Similarly to POS embeddings, we use a model trained on biomedical texts to predict dependency trees (Neumann et al., 2019).

The sentence representation is thus passed through the convolutional layer (operating on 1D) and the max-pooling layer. The 4-class classification is done at the classification layer using softmax. Similarly to Nguyen and Grishman (T. H. Nguyen & Grishman, 2015) we used shuffled mini-batches of size 50 during training and a dropout regularization rate $\rho = 0.5$ to avoid overfitting. All the weights of the network are updated at training time, except for the 200-dimensional pre-trained word embeddings.

4.4.2 Experimental Setup

Before training, we tuned the hyper-parameters of the network under a 5-fold *stratified group* cross-validation setting on the *train/dev* set of GE11 (Table 4.3).[5] We designed this multifaceted cross validation setting (i) to account for the class imbalance, ensuring different splits have examples from all the classes, especially the under-represented ones, and (ii) to avoid the same document falling into different splits, a long-standing issue in comparability of

[5]We use the standard GE11 benchmark since it represents the largest biomedical event extraction corpus and it includes both abstract and full-text documents.

Table 4.4: The search space for the best hyper-parameter configuration, along with the optimal values.

Hyper-parameter	Value	Space
Optimizer	Adam	Adadelta, Adagrad, Adam, Adamax, RMSProp, SGD
Learning rate	$5e^{-4}$	1, $1e^{-1}$, $1e^{-2}$, $1e^{-3}$, $5e^{-4}$, $1e^{-4}$
Batch size	50	50, 64
Window sizes	[3,4,5]	[3,4,5], [1,3,5,7]
Filters	150	32, 150
Embedding size	32	8, 16, 32, 50, 64

relation extraction systems (Pyysalo et al., 2008), which we extend to the document scope. In fact, not only the same sentence but contiguous sentences could share common information that could lead to an overestimation of performance.

We collect hyper-parameter choices that have been employed in related work (T. H. Nguyen & Grishman, 2015; Björne & Salakoski, 2018)[6] for the optimizer, the learning rate, the batch size, the window size, and the number of filters. Additionally, we search for the optimal dimension of the input embeddings, which we concatenate to the 200-dimensional word embeddings. We perform a grid search to select the values, averaging the performance of models for each combination of input embeddings across the five executions. To prevent overfitting, we use early stopping with a patience value of 5 epochs, choosing models from the epoch with the highest micro F_1 score on the development set. Table 4.4 depicts the search space, where we highlight the best hyper-parameter values we choose. We also find that no significant differences in performance are given by different dimensions for each input embedding.

Finally, we train the network on the whole *train/dev* set of each corpus, evaluating it on the respective *test* set (Table 4.3). For cross-domain evaluation, we instead test all the models – trained on a source *train/dev* corpus – on the *test* set of all the other corpora.

To give a detailed picture of the performance, we report both the micro F_1

[6] We acknowledge that optimal values for hyper-parameters might be outside this search space; however, for the purpose of a baseline model and due to the exploding combinations of values to be tested, we leave extensive hyper-parameter tuning for future work.

and macro F_1 scores, while we use micro F_1 for model selection. We believe reporting both is useful to the community since the evaluation is typically specific to the use case – in fact, one could be more interested in good performance among all classes rather than at an instance level. Additionally, for each metric we also provide the scores considering negative instances in the evaluation (i.e., with `NoEdge`) and without considering them (i.e., without `NoEdge`). We believe this sheds light on the impact of negative examples in edge detection. For the sake of comparability of future results, we also make the *stratified group* splits for each dataset publicly available.

4.4.3 Ablation Study

We investigate the contribution of different combinations of input embeddings (i.e., *POS*: part-of-speech, *TYP*: type, *DEP*: dependency) to the performance of the model on the GE11 development splits (Table 4.5). We average micro F_1 scores of each variant of the model on the five development splits, also reporting the standard deviation. This experiment shows that (i) the most informative input embedding is DEP, which individually contributes 1.96 F_1, followed by TYP, which contributes 1.68 F_1; (ii) POS is the least informative embedding since whether it is removed individually or with other embeddings it decreases the performance only slightly (from 0.09 to 0.42 F_1);[7] (iii) the behaviour of the different input embeddings is consistent both individually and in group, with POS and DEP being highly independent from TYP due to their semantic interdependency; (iv) the inclusion of all the embeddings contributes to a gain of 4.28 F_1 points with respect to the baseline with only word and position embeddings. It is worth noting that DEP is the most informative embedding despite being a predicted feature. To sum up, the semantic and syntactic features together help edge detection, with the dependency path being the most informative feature.

[7]The max decrease in performance of 0.42 F_1 points occurs when POS is removed with TYP, DEP ("POS, TYP, DEP", Table 4.5) compared to keeping it ("TYP, DEP", Table 4.5).

Table 4.5: Ablation study on input embeddings. We report mean and standard deviation of micro F_1 scores on the development splits for each variant of the model, as well as the performance loss (Δ) with respect to the complete model.

Model	Micro F_1 score	Δ
All the input embeddings	**88.83** ± 0.35	
– POS	88.74 ± 0.58	-0.09
– TYP	87.15 ± 0.66	-1.68
– DEP	86.87 ± 0.33	-1.96
– POS, TYP	86.76 ± 0.52	-2.07
– POS, DEP	86.67 ± 0.51	-2.16
– TYP, DEP	84.97 ± 0.61	-3.86
– POS, TYP, DEP	84.55 ± 0.57	-4.28

4.5 Results and Discussion

In-domain and out-of-domain results of the baseline model across the five corpora are reported in Table 4.6. Particularly, we present four evaluation strategies to guide the reader in choosing the most appropriate approach for assessing edge detection performance according to its use case. To give additional insights on the results, we also present per-class F_1 scores of each model trained on a *source* corpus when applied to each *target* corpus (Figure 4.2). Thus, we hereafter use both views to complement the discussion of the results.

In-domain results As we can see in Table 4.6, regardless of the metric and the classes used, in-domain results (with a *grey* background) are consistently better than out-of-domain results. This is not surprising since corpora are characterized by important linguistic variations. The only exception is the ID11 corpus: a model trained on ID11 seems not enough to provide the highest macro F_1 score on the ID11 test set. This is due to both the relatively small size of the ID11 *train/dev* set and the very few training examples having `Location` as a label, clearly insufficient to learn the patterns that characterize the class (Table 4.3). This only impacts the macro F_1 score, where under-represented classes such as `Location` are given the same weight as dominant classes such as `Theme` and `NoEdge`. As a matter of fact, the GE11 model achieves a higher

Table 4.6: Cross-domain performance of the baseline model for edge detection. Different performance views are presented, according to both the evaluation metrics used – i.e., micro F_1 score (top table), or macro F_1 score (bottom table) – and whether the scores consider the classification of negative examples – i.e., with `NoEdge` or without `NoEdge`, on the rows of both tables. In-domain results are on the diagonals (with a *grey* background), while best results on target corpora are in bold. For each metric and evaluation strategy, we indicate the average out-of-domain drop (in italic).

	target →	micro F_1					
	source ↓	GE11	ID11	EPI11	PC13	MLEE	Avg
with NoEdge	GE11	**88.65**	86.67	84.35	84.36	84.79	*-3.71*
	ID11	80.48	**90.05**	71.50	81.84	84.07	*-10.58*
	EPI11	73.67	78.00	**87.88**	76.17	71.41	*-13.07*
	PC13	83.87	86.95	73.10	**88.11**	87.15	*-5.34*
	MLEE	81.22	88.20	70.24	84.54	**90.20**	*-9.15*
without NoEdge	GE11	**83.66**	69.31	83.43	66.14	63.39	*-13.09*
	ID11	70.67	**72.63**	59.72	61.54	61.82	*-9.19*
	EPI11	63.69	53.01	**87.17**	52.18	47.93	*-32.97*
	PC13	74.92	68.09	59.90	**76.22**	70.08	*-7.97*
	MLEE	69.15	67.68	53.48	64.75	**75.95**	*-12.08*

	target →	macro F_1					
	source ↓	GE11	ID11	EPI11	PC13	MLEE	Avg
with NoEdge	GE11	**81.01**	**74.28**	77.09	53.28	50.85	*-17.13*
	ID11	56.90	65.76	52.33	48.61	48.17	*-14.01*
	EPI11	62.22	54.93	**82.19**	48.95	42.78	*-29.97*
	PC13	56.81	54.38	54.57	**77.48**	56.00	*-22.04*
	MLEE	55.54	55.42	52.54	57.75	**74.59**	*-19.28*
without NoEdge	GE11	**77.47**	**68.39**	74.35	40.66	37.37	*-22.28*
	ID11	47.17	56.20	42.99	36.33	33.96	*-16.09*
	EPI11	56.23	44.53	**80.06**	36.70	29.76	*-38.25*
	PC13	46.01	41.70	44.51	**72.42**	43.79	*-28.42*
	MLEE	44.92	42.79	42.58	46.67	**68.08**	*-23.84*

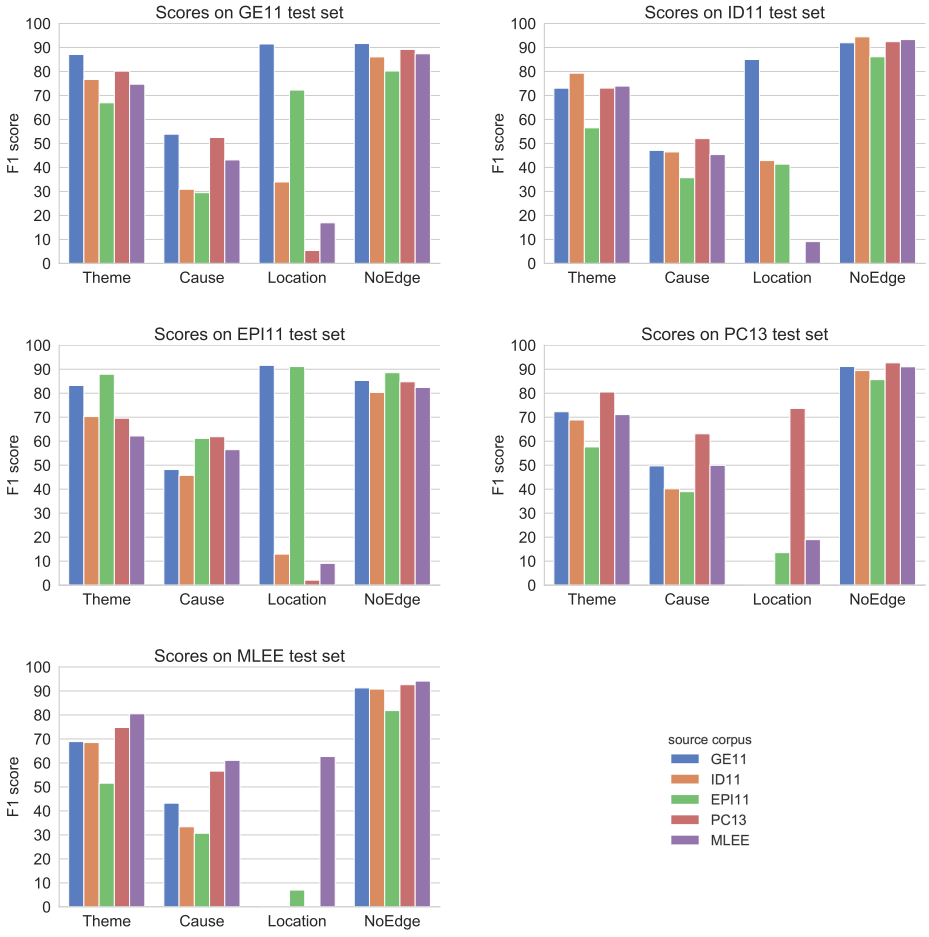


Figure 4.2: Performance of in-domain models for each *source* corpus on detecting and classifying edge labels on all the other corpora (*target*). Each plot indicates a *target* corpus the *source* model is tested.

macro F_1 score on the ID11 test set only because of the `Location` classification performance, almost two times the one provided by the in-domain ID11 model (Figure 4.2, "Scores on ID11 test set"). On average over all five corpora, the in-domain micro F_1 is 88.98 and 79.13 with and without considering `NoEdge`, respectively, while the in-domain macro F_1 is 76.21 and 70.85 with and without negative instances, respectively.

Out-of-domain results A large drop in performance occurs when in-domain models are applied on *out-of-domain* corpora. As reported in Table 4.6, the drop in micro F_1 score is from 3.71 to 13.07 points if we consider negative instances in the evaluation, and from 7.97 to 32.97 without considering them. Regarding the macro F_1 score, the drop is even more pronounced, going from 14.01 to 29.97 considering negative edges, and from 16.09 to 38.25 without them. From a closer point of view, we notice EPI11 is the most difficult domain a model could be applied to, as shown by the highest drop in out-of-domain performance across all metrics and classes (i.e., -13.07 and -29.97 considering the `NoEdge` class, and -32.97 and -38.25 without considering the `NoEdge` class, for micro and macro F_1 scores, respectively). This could be due to how EPI11 was constructed, since it is the only corpus that was built avoiding a sample selection bias towards particular proteins or event expressions (Ohta et al., 2011). Another interesting finding is about the ability of some in-domain models to generalize reasonably well to a specific target domain. This is the case of the GE11 model, when applied to EPI11 as a target (i.e., GE11→EPI11), and of the PC13 model, when applied to MLEE (i.e., PC13→MLEE). Although they are far from the performance of the target in-domain models – especially under the macro F_1 metric – they consistently achieve better results than other in-domain models on all metrics and classes. As we can see in Figure 4.2, "Scores on EPI11 test set", in the GE11→EPI11 case the GE11 model obtains lower performance mainly due to the classification of the `Cause` class, while maintaining close performance on `Location` and `NoEdge` classes. Regarding PC13→MLEE, the difference in performance with respect to the MLEE in-domain model could be explained by the `Location` score, which is 0% (Figure 4.2, "Scores on MLEE test set").

`Location` seems to be the trickiest class to predict especially in the MLEE target test set, where the only source that achieves a score greater that 0% is EPI11 (7.02%) (Figure 4.2, "Scores on MLEE test set"). In general, our experiments highlight that there is no single source corpus in which a model could be trained to robustly and consistently achieve good performance on all target corpora. This highlights the need for future research in out-of-distribution robustness to make in-domain models able to generalize better across linguistic varieties, even within biomedicine itself.

Metrics and classes We notice important distinctions when using micro F_1 or macro F_1 score as the evaluation metric. Firstly, the scores using macro F_1 are generally lower than the scores using micro F_1. This is because over-represented classes (e.g., `Theme`, `NoEdge`) dominate the micro F_1 score, while the correct classification of under-represented classes (e.g., `Location`) is central to obtain a high macro F_1 score. Secondly, considering negative instances (i.e., `NoEdge`) in computing the averaged scores of edge detection leads to an overestimation of the performance. This could be explained by the fact that `NoEdge` is the majority class in all the corpora, thus giving a high contribution on both micro and macro F_1 scores. Despite it is a common practice to consider only true annotated labels (i.e., `Theme`, `Cause`, and `Location`) in the evaluation – using wrongly predicted negative instances as *false negatives* for the actual class, and treating as *false positives* for the actual class the instances that are negatives, but classified in that class – we believe considering the negative class in the evaluation could be beneficial in developing real world applications. Whatever evaluation strategy is used, we see the trend of the scores is consistent across metrics and classes.

4.6 Summary and Conclusions

We provided the first cross-domain evaluation study for biomedical edge detection, together with standardized data from five gold-standard corpora to enable further progress in comparable edge detection. We proposed different evaluation strategies to assess the performance of models, together with an in-domain

baseline for edge detection, for which we assessed the contribution of different combinations of input embeddings, finding syntactic and semantic features to be particularly helpful. We used in-domain models to assess the performance drop across five datasets, shedding light on the importance of developing robust models that could deal with the linguistic variations in different corpora. We believe this work could encourage future work in out-of-distrubution generalization, and could sensitize an awareness about the language differences within the biomedical domain. The data, the splits, and the baselines for edge detection are publicly available. This work motivated us to explore complementary research directions, outlined in the following:

End-to-end event extraction State-of-the-art systems currently employ classifier pipelines (Björne & Salakoski, 2018; Li et al., 2019), of which edge detection is one of these classifiers. This study clearly showed that even the stage of edge detection alone is far from being perfect in performance. In classifier pipelines, errors typically propagate between modules – i.e., an incorrect prediction from a module (say, trigger detection) inevitably affects the performance of the following stage (i.e., edge detection). We thus investigate an end-to-end solution for the whole task in the next chapter (Chapter 5).

Transfer learning for domain robustness For the purpose of this work, we used a CNN baseline since it has been shown to provide an higher cross-domain robustness compared to traditional methods for relational semantics tasks (T. H. Nguyen & Grishman, 2015). Recently, transfer learning has arisen as an effective solution to mitigate the out-of-domain performance drop issue, by transferring previous knowledge acquired from a related domain to a target domain (Ruder, 2019). Pre-trained models such as BERT (Devlin et al., 2019) are typically employed to the purpose of language variations and data mismatch. In the next chapter (Chapter 5), we thus employ transfer learning in biomedical event extraction for the first time, while providing an efficient, high-performance, and end-to-end solution.

Chapter 5

Biomedical Event Extraction as Sequence Labeling

*In this chapter we present a novel end-to-end biomedical event extraction approach, namely Biomedical Event Extraction as Sequence Labeling (*BeeSL*). We contribute to the field by recasting the event structures into a representation suitable for sequence labeling, thus modeling trigger and argument information jointly through a shared encoder, experimenting with multi-task learning strategies and multi-label aware decoding.* BeeSL *is both fast and accurate, and unlike current methods does not require any external knowledge base or preprocessing tools.* BeeSL *outperforms the current best system on the standard Genia 2011 benchmark by 1.57% absolute F1 score, establishing a new state of the art for the task. Empirical results show that* BeeSL*'s speed and accuracy makes it a viable approach for large-scale real-world scenarios.*

This chapter is based on the following scientific publication:

> **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)* (pp. 5357–5367). Punta Cana, Dominican Republic (Online): Association for Computational Linguistics.

5.1 Introduction and Motivation

Biomedical event extraction is a complex task that requires to detect – and assign a semantic type to – both event triggers (i.e., the words indicating the presence of a biomedical event) and their arguments (i.e., which are participating in those events). As introduced in Chapter 2, biomedical events are typically highly complex and nested structures (Miwa et al., 2014) where long-distance arguments are frequent (Li et al., 2019), which thus require deep contextual knowledge to resolve. While the task has received significant attention in research over the last decade, it remains challenging. Progress has been rather stagnating (see Figure 5.1).

State-of-the-art biomedical event extraction systems still work as classifier pipelines, and extract event triggers and their arguments independently (Björne & Salakoski, 2018; Li et al., 2019). They typically employ dependency parsing as features in a CNN model ensemble (Björne & Salakoski, 2018) or in Tree-LSTMs with knowledge bases (Li et al., 2019). We instead argue that modeling triggers and arguments jointly can lead to a more efficient model in terms of both speed and accuracy due to the sharing of information between sub-tasks.

In this work we propose a new approach for biomedical event extraction by casting it as a sequence labeling task (BeeSL). Our approach is conceptually simple: we convert the event structures into a representation suitable for sequence labeling, and leverage a multi-label aware decoder with BERT (Devlin et al., 2019) in a multi-task sequence labeling model. This reduces the problem to predicting a structured output for an input sequence to word-level tagging decisions. Compared to previous alternatives (cf. Section 5.2) which cast event extraction as syntactic or semantic tree- or graph-parsing task, this leads to a faster, joint model which also mitigates error propagation of locally-optimized classifier pipelines (Björne & Salakoski, 2018; Li et al., 2019). Our empirical evaluation shows the effectiveness of BeeSL for biomedical event extraction. A quantitative and qualitative analysis shows that BeeSL is fast and effective. Despite the model's simplicity, BeeSL outperforms the previous best model (Li et al., 2019) on most event categories.

To the best of our knowledge, we are the first to cast biomedical event ex-

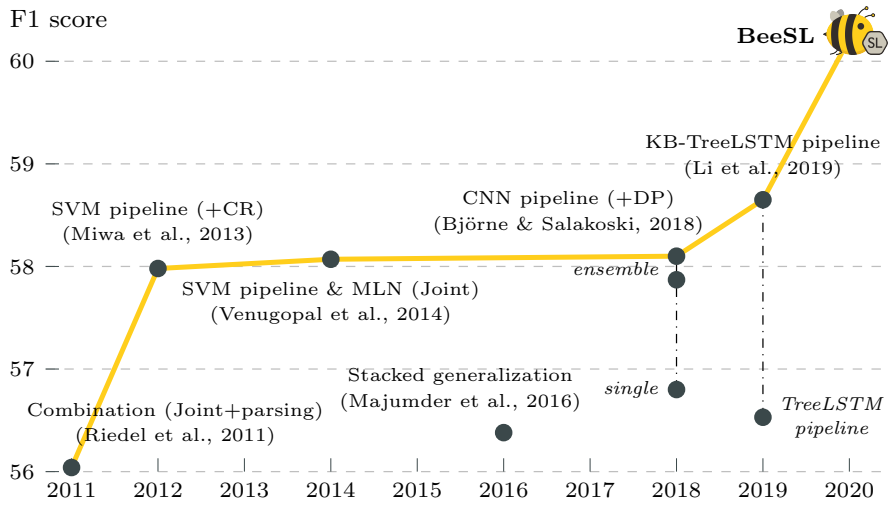


Figure 5.1: Performance of biomedical event extraction on the standard BioNLP Genia 2011 test set over time.

traction as sequence labeling. We demonstrate that BeeSL is an attractive and efficient solution to extract biomedical events. We evaluate it on the standard BioNLP Genia 2011 benchmark, obtaining a new state of the art, while gaining on efficiency. We additionally provide empirical results of the impact of alternative multi-task encodings, and to the best of our knowledge, the first results of biomedical event extraction *without* assuming entities are given.[1]

The remaining part of this chapter is organized as follows. Section 5.2 outlines related work in biomedical event extraction. In Section 5.3 we revise the details on the task, presenting our encoding strategy for sequence labeling and the model. Section 5.4 illustrates the experimental setup, whereas in Section 5.5 we present the empirical results, analyzing the multi-task learning variants and the multi-label decoding feature, as well as providing a comparison of BeeSL to the state-of-the-art in terms of accuracy and speed. Section 5.6 provides a detailed analysis and discussion on the importance of multi-task learning, the stability of the multi-label decoder, the robustness to non-gold entity informa-

[1]The source code is available at `https://github.com/cosbi-research/beesl`

tion, and an error analysis. Finally, conclusions are in Section 5.7.

5.2 Related Work

Biomedical event extraction has a long-standing tradition. In the last decade, several methods based on traditional machine learning approaches have been proposed (Riedel et al., 2011; Miwa et al., 2012; Vlachos & Craven, 2012; Venugopal et al., 2014; Majumder et al., 2016). Current work in the field has explored neural methods and uses multiple classification stages to deal with biomedical event extraction. Namely, first identifying trigger mentions, and then evaluating all possible combinations for each trigger to all entities or triggers in order to detect event arguments (Li et al., 2019; Björne & Salakoski, 2018). They come with the shortcomings of traditional classifier pipeline methods, particularly the error propagation between different modules, such as trigger and edge (i.e., argument) detection. In this work we instead take an alternative direction modeling both triggers and event arguments jointly. Further, many studies use dependency parsers to obtain features or for guidance of Tree-LSTMs (Li et al., 2019; Björne & Salakoski, 2018), being dependent on predicted information by other tools to tackle the task. We instead dispense external resources relying on contextualized word representations only.

Earlier work framed biomedical event extraction as syntactic and semantic tree- or graph-parsing (McClosky et al., 2011; Rao et al., 2017). In particular, McClosky et al. (2011) do dependency parsing, followed by a second-stage parse reranker model for event extraction, and Rao et al. (2017) cast the problem as subgraph identification problem. Recent work in syntactic parsing has shown that reducing parsing to sequence labeling is a viable alternative for both constituent and dependency parsing (Spoustová & Spousta, 2010; Gómez-Rodríguez & Vilares, 2018; Strzyz et al., 2019), which we took as inspiration in this work. To the best of our knowledge, we are the first to cast biomedical event extraction as a sequence labeling task.

Joint learning paradigms for biomedical event extraction were explored in early work (Riedel & McCallum, 2011; Riedel et al., 2011; Venugopal et al., 2014; Vlachos & Craven, 2012). Remarkably, the system by Riedel et al. (2011)

is still amongst the top-scoring models (cf. Figure 5.1). Contemporary to our work, a very recent study proposes oneIE, a joint learning model for event extraction in the newswire domain (Lin et al., 2020). It proposes a single end-to-end model for event extraction using four stages, paired with a beam search, obtaining good results on ACE data (Walker et al., 2006). However, event structures are flat in the newswire ACE data, while they can be nested in the biomedical GENIA data thus being more challenging (Miwa et al., 2014).

To handle the frequent occurrence of tokens that are argument of multiple events, we devise a dedicated multi-label decoder. Processing multiple labels per token has previously been done for relation extraction using multi-head selection (Bekoulis et al., 2018a, 2018b), and sequence labeling has been employed for joint entity and relation classification (Dai et al., 2019) with inter-token attention. We instead employ multi-label decoding at the token-level for multi-label aware sequence labeling.

5.3 Methods

In this section we revise what event structures are and provide details on how we encode them to a representation suitable for sequence labeling (Section 5.3.1). Afterwards, we present our biomedical event extraction model for learning the resulting linearized representation in an end-to-end fashion, by exploiting an encoder pre-trained on biomedical texts, different multi-task learning strategies, and multi-label decoding (Section 5.3.2).

5.3.1 Encoding Event Structures

Events are structured representations which comprise multiple information units (Figure 5.2, top). An event is anchored to a *trigger*, a text span which indicates the presence of an event (Figure 5.2, rounded boxes). Each event has one or more arguments, namely *entities* or other events (Figure 5.2, end of arrows), which are assigned a *role* in the event (Figure 5.2, labels on arrows). For example, an Expression event is indicated in Figure 5.2 at "production" involving the Protein "IL-10" as its argument. Nested structures are possible and frequent in biomedical event extraction. For instance, the +Regulation

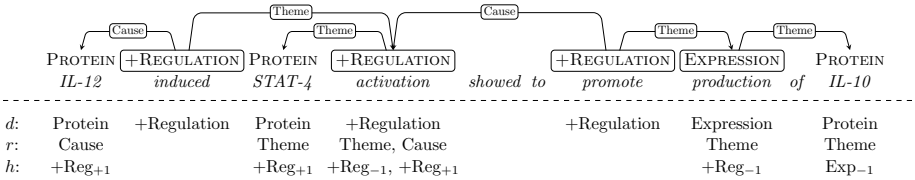


Figure 5.2: Top: a text excerpt with four biomedical events. Above the text (*italicized*), mentions (triggers inside rounded boxes, and entities without rounded boxes) and argument roles (labels on arrows) are indicated. Bottom: our proposed encoding, where d, r and h represent the label parts for dependents, relations, and heads, respectively.

event centered on "activation" is both argument of the "induced"-anchored event as well as the "promote"-anchored event.

Given $[x_1, ..., x_n]$ a sequence of n tokens, we encode event structures as token-level labels $[y_1, ..., y_n]$, to reduce the task to a sequence labeling problem. Adopting the dependency parsing terminology (Jurafsky & Martin, 2020), we encode the label y_i for each token x_i as a tuple $\langle d, r, h \rangle$, where:

- d (*dependent*): a label indicating the mention type of the token (either a trigger, entity, or nothing);
- r (*relation*): a label indicating the argument role type of the token, with respect to the event it is participating in;
- h (*head*): a label indicating the actual event (trigger) type of which the token has an argument role in.

The labels are illustrated in Figure 5.2 (bottom). In more detail, to discriminate heads with the same type in text, we encode the heads h as *relative head mention position*.[2] For instance, $h = +\text{REG}_{+1}$ means the head is the first +REGULATION on the right of d in the relative surface order, whereas $h = +\text{REG}_{-2}$ means it is the second +REGULATION on the left. In Figure 5.2 the

[2]In preliminary experiments we found this mitigates the label sparsity problem of other positional encodings, e.g., relative positional encoding (Strzyz et al., 2019). We additionally found relative head mention positions ≥ 2 are rare in our data.

label for "production" is ⟨Expression, Theme, +Reg$_{-1}$⟩, denoting the token is an Expression trigger, Theme of the first +Regulation event on the left.

As opposed to dependency parsing, tokens may have zero or multiple roots, and thus multiple heads and relations. This poses additional challenges. For instance, the "activation"-anchored event (Figure 5.2) is both Theme and Cause of "induced"- and "promote"-anchored event heads, respectively. As a result, both r and h are multi-label, and the label for "activation" is encoded as ⟨+Regulation, [Theme, Cause], [+Reg$_{-1}$, +Reg$_{+1}$]⟩, where the order of r and h items is preserved. We handle multiple labels per token via multi-label decoding (Section 5.3.2.2).

5.3.2 Event Extraction as Sequence Labeling

Formally, we aim to learn a function $f : X \mapsto Y$ that assigns each token x_i a structured label y_i, i.e., $\langle d, r, h \rangle$. A straightforward solution is to predict the label y_i as an atomic entity (i.e., a single label resulting from the concatenation of d, r and h) in a single-task model. For BeeSL, we instead propose to use multi-task learning (MTL) which allows to learn interdependencies while cutting down the label space, paired with multi-label prediction.

An overview of BeeSL is shown in Figure 5.3. We use BERT (Devlin et al., 2019) as encoder, pre-trained on biomedical texts (Section 5.4), masking entity spans for better generalization (Alt et al., 2019). The first WordPiece (Schuster & Nakajima, 2012) of each token x_i is used for prediction, where the contextual hidden representation e_i of the token x_i is encoded with layer-wise attention over the BERT layers, similarly to (Peters et al., 2018; Kondratyuk & Straka, 2019). As decoders, we use standard softmax with a cross entropy loss unless otherwise specified (Figure 5.3, upper left), and introduce a multi-label decoder (Section 5.3.2.2) (Figure 5.3, upper right). Variants of the model depicted in Figure 5.3 are thoroughly discussed in Section 5.3.2.1 and Section 5.3.2.2.

We empirically evaluate both single-task and multi-task setups, including several MTL encoding alternatives, discussing their limitations and benefits. In the following, we first introduce the multi-task setups (Section 5.3.2.1), and then multi-label decoding (Section 5.3.2.2).

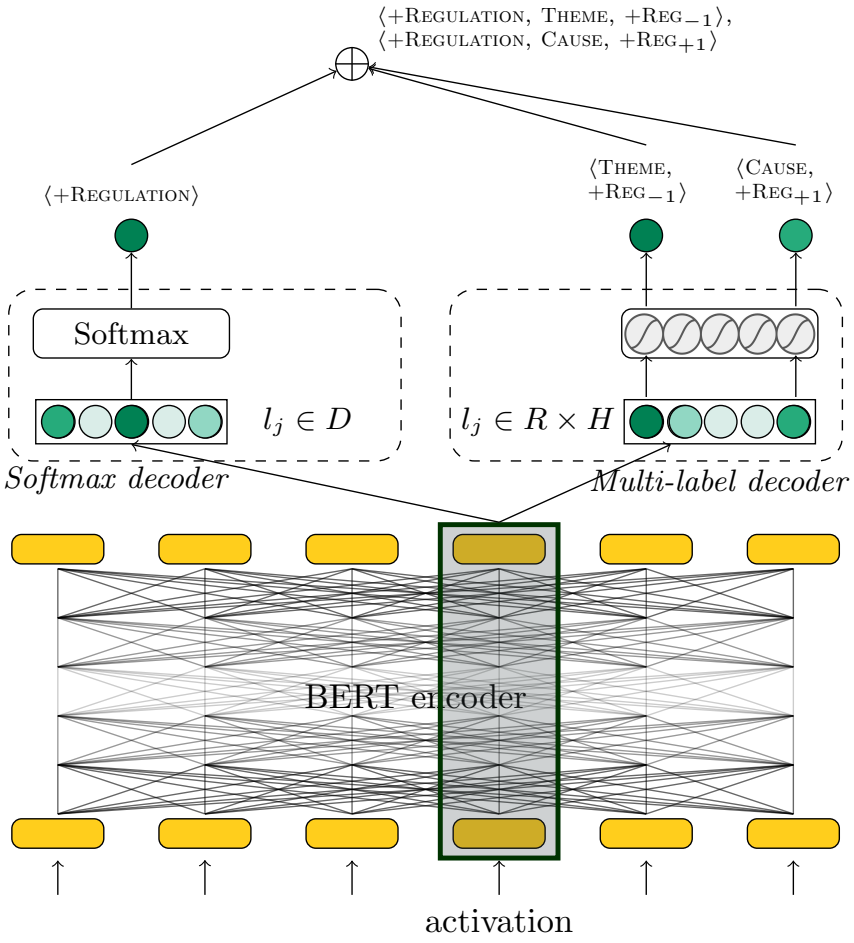


Figure 5.3: BeeSL uses a multi-task multi-label model, using a BERT encoder with layer attention, and dedicated decoders for predicting the labels for each label sub-space, which are trivially merged. In this figure the prediction of labels for the token "activation" is illustrated.

5.3.2.1 Multi-Task Strategies

We denote the label spaces for each component of the labels as $d_i \in D$, $r_i \in R$, and $h_i \in H$. Further, we use $\mathcal{L}$ to refer to the maximum label space size.

Single-task A single-task (ST) setup is used as a baseline. It predicts a single label $y_i = \langle d, r, h \rangle$ for each input token x_i, thus using a single decoder. The label space is up to $\mathcal{L} = |D| \times |R| \times |H|$.

Multi-task The label y_i for each token x_i is decomposed into parts (hereafter, sub-labels), each treated as a prediction task. The decomposition of the label space allows each sub-label space to be framed as a different task with its own private decoder, mitigating the output space sparsity (Vilares et al., 2019). Depending on the decomposition of the label $y_i = \langle d, r, h \rangle$, we have four multi-task learning options (pairs of tasks, or each subpart as a task, respectively) with the following properties:

Option 1. $\langle d \rangle, \langle r, h \rangle$: up to $\mathcal{L} = |D| + |R| \times |H|$;

Option 2. $\langle d, r \rangle, \langle h \rangle$: up to $\mathcal{L} = |D| \times |R| + |H|$;

Option 3. $\langle d, h \rangle, \langle r \rangle$: up to $\mathcal{L} = |D| \times |H| + |R|$;

Option 4. $\langle d \rangle, \langle r \rangle, \langle h \rangle$: up to $\mathcal{L} = |D| + |R| + |H|$.

Option 4 encodes each subpart as its own task. While this leads to the smallest label space, it decouples the problem into 3 separate tasks. Options 1-3 are pair-wise task setups. We hypothesize that BEESL benefits from disentangling mention detection from head labeling (option 1, depicted in Figure 5.3).

As illustrated in Figure 5.3, BEESL uses the predicted sub-labels to form the complete label tuple $\hat{y}_i = \langle \hat{d}, \hat{r}, \hat{h} \rangle$. In case r and h belong to different sub-label spaces (as is possible in options 2-4), we require that both predictions $\hat{r}$ and $\hat{h}$ are present (non-empty) to ensure well-formedness. This is a downside of these alternative options 2-4, as we will see empirically (Section 5.5).

During training, the MTL loss is computed as $L = \sum_t \lambda_t L_t$, where L_t is the loss for each task t, given by the respective decoder (see also Section 5.3.2.2),

with λ_t a task-specific weighting parameter. In our experiments we kept $\lambda = 1.0$ for all, since preliminary experiments showed weighting sub-tasks differently was not beneficial. In the single-task setup, the loss reduces to $L = L_t$.

5.3.2.2 Multi-Label Decoder

The multi-label decoder is designed to handle multiple labels per token, thus being suitable for predicting relations and heads, which may be more than one as introduced in Section 5.3.1. Given a task with $l_j \in L$ labels, the multi-label decoder models $P(l_j|e_i)$ for each label l_j. Differently from the single-label decoder (i.e., softmax), each label is predicted with a sigmoid, where all contribute equally to the loss. Thus, given the probabilities $P(l_j|e_i)$ for the $l_j \in L$ labels and a threshold τ, the token x_i is assigned all the labels l_j with probability $P(l_j|e_i) \geq \tau$. If no $P(l_j|e_i) \geq \tau$ is found, we take the highest scoring label l_j (which may also be empty) as a fallback.[3] We employ a binary cross-entropy loss, averaged across all batches.

5.4 Experimental Setup

We evaluate BeeSL on the Genia 2011 benchmark (J.-D. Kim et al., 2011), which is the largest dataset to date that comprises both the linguistic varieties of abstracts and full-texts, also including a large fraction of nested events (Björne & Salakoski, 2011). The corpus consists of annotations for Protein entities and 9 fine-grained event types. The Genia event extraction task expects both texts and entities as input, and complete events need to be predicted. Statistics on the dataset are shown in Table 5.1. Further details on data creation and annotation are in Appendix D.

Event types can be categorized into simple, binding and complex events, related to the number and types of arguments. *Simple events* require a Theme only, *binding events* require one or more Theme arguments, while *complex events* take both Theme and Cause arguments, where both can in turn be

[3] In case $\tau = 0 \vee \tau = 1$, we adopt the same strategy, since all or no labels would be potentially predicted, respectively.

Table 5.1: Statistics of the Genia 2011 event dataset.

Item	Train	Dev	Test
Documents	908	259	347
Sentences	8,664	2,888	3,363
Tokens	230,737	74,334	90,091
Entities	11,625	4,690	5,301
Events	10,310	3,250	4,487

Table 5.2: Formal definition of biomedical events. P: PROTEIN entity, E: any event type, +: 1 or more arguments, ?: 0 or 1 arguments.

Event type	Arguments
Simple events	
Gene expression	Theme(P)
Transcription	Theme(P)
Protein catabolism	Theme(P)
Phosphorylation	Theme(P)
Localization	Theme(P)
Binding	Theme(P)+
Complex events	
Regulation	Theme(P/E), Cause(P/E)?
Positive regulation	Theme(P/E), Cause(P/E)?
Negative regulation	Theme(P/E), Cause(P/E)?

other events, resulting in nested structures. Björne and Salakoski (2011) estimated that 37.2% of the events in the data are nested. A formal specification of biomedical events is detailed in Table 5.2, whereas the full data can be downloaded freely from the official GENIA portal.[4]

BEESL is based on MaChAmp (van der Goot et al., 2021), a toolkit for multi-task learning and fine-tuning of BERT-like models, which in turn is based on the PyTorch-based (Paszke et al., 2019) AllenNLP library v0.9.0 (Gardner et al., 2018). We extend MaChAmp to also handle multi-label sequence labeling. We experiment with BEESL in single- and different multi-task setups.

[4] `http://bionlp-st.dbcls.jp/GE/2011/downloads/`

After sequence labeling, token-level labels are converted into the official BioNLP-ST standoff format for evaluation (J.-D. Kim et al., 2011). We simply split the event arguments based on their formal definition, producing complete structures (e.g., an EXPRESSION event with k THEME arguments is split into k EXPRESSION events, with one THEME each). Similarly to previous work, we focus on sentence-level events.[5] We used BioBERT-Base 1.1 as our BERT model for experiments, since it provides state-of-the-art performance across multiple biomedical information extraction tasks (Lee et al., 2020).

Hyper-parameter values and tuning details The list of hyper-parameter values and the search space are presented in Table 5.3, whereas the number of trainable parameters in BEESL is $\approx 110M$. For tuning, we started from the values reported in previous works on multi-task learning for NLP evaluation benchmarks, e.g., UDify (Kondratyuk & Straka, 2019), and perform a minimal search to avoid overfitting. We performed 32 search trials via grid search, in which the "batch size" and the "base learning rate" have been coupled due to their strong interdependency – $(32, 1e^{-}3)$ and $(64, 1e^{-}2)$. Additional search trials have been performed for threshold τ selection (yielding the threshold for multi-task $\tau_{MT} = 0.5$ and for single-task $\tau_{ST} = 0.7$). We used the official approximate recursive span matching based F1 score for model selection, whereas the sum of span-based F1 scores of the tasks was employed to determine early stopping of the training process.

Evaluation In line with previous work, we evaluate BEESL in terms of precision (P), recall (R), and F1 score according to the approximate recursive span matching criterion (J.-D. Kim et al., 2011) using the official BioNLP online evaluation service.[6]

[5] Events crossing sentence boundaries account for 6.0% of the total events in GENIA. Sentence-level processing theoretically limits the maximum performance that can be achieved by a system; however, dealing with multi-sentence inputs requires specifically tailored methods and is likely to drastically reduce the efficiency of a system in terms of training and inference time and memory, thus undermining its practical usage. Moreover, we hypothesize that such a small fraction of inter-sentence events is likely to introduce a high number of false positive cross-sentence events. This is in line with findings by Lever and Jones (2016).

[6] `http://bionlp-st.dbcls.jp/GE/2011/eval-test/`

Table 5.3: Hyper-parameter values and search space.

Hyper-parameter	Value	Space
Optimizer	Adam	
β_1, β_2	0.9,0.99	
Weight decay	0.01	
Gradient clipping	10	
Dropout	0.5	0.1, 0.3, 0.5
BERT dropout	0.1	0.1, 0.2
Mask probability	0.1	0.1, 0.15, 0.2
Layer dropout	0.1	
Batch size	64	32, 64
Base learning rate	$1e^{-}2$	$1e^{-}3$, $1e^{-}2$
BERT learning rate	$5e^{-}5$	
Epochs	50	
Patience	5	
Multi-label threshold	0.5	0.1, 0.2, ..., 1.0

No gold entities In biomedical event extraction, entities are typically given in advance. To evaluate BeeSL in a setup with *predicted* entities (Section 5.6.3), we firstly employ our model as single-task sequence labeler for recognizing entity mentions using default settings and a standard Conditional Random Field (CRF) decoder (Gardner et al., 2018). Note that for comparison purposes in all other experiments we assume entity mentions are given. Then, we evaluate BeeSL with raw texts and *predicted* entities as input, thus indirectly penalizing events that take over-predicted entities or that miss entities since they are under-predicted.

5.5 Empirical Results

First, we evaluate the MTL strategies and multi-label decoding on the development set to determine the best setup (see "Multi-task settings" and "Adding the multi-label decoder"). Then, we compare BeeSL to the results obtained by the top performing systems on the official test set (see "Comparison to the state of the art"). Finally, we gauge its speed (see "Speed comparison").

Table 5.4: Performance of diverse settings for BeeSL (multi-task and multi-label) on the development set.

Multi-task	P	R	F1
$\langle d \rangle, \langle r, h \rangle$	71.28	55.44	**62.37**
$\langle d, r \rangle, \langle h \rangle$	72.35	51.31	60.04
$\langle d, h \rangle, \langle r \rangle$	73.51	49.49	59.16
$\langle d \rangle, \langle r \rangle, \langle h \rangle$	73.05	51.34	60.30
Multi-label	P	R	F1
BeeSL$_{ST}$	73.30	52.42	61.13
with multi-label	71.74	56.71	63.34
BeeSL$_{MT}$	71.28	55.44	62.37
with multi-label	71.84	59.42	**65.04**

Multi-task settings Table 5.4 (top) summarizes the main results for the MTL experiments. They confirm our hypothesis that $\langle d \rangle, \langle r, h \rangle$ (option 1) is the most viable representation; it leads to the highest F1 score, largely outperforming the other MTL options, particularly in recall. These results show that a multi-task setup with separate tasks for mention detection, and relation and head labeling, respectively, is the most useful. Option 1, i.e., $\langle d \rangle, \langle r, h \rangle$ defaults to the multi-task option for BeeSL (Figure 5.3) used in the following experiments.

Adding the multi-label decoder We evaluate the multi-label decoder for both single-task (BeeSL$_{ST}$) and multi-task (BeeSL$_{MT}$) setups (Table 5.4, bottom). Multi-label decoding is beneficial, as the data contains many multi-headed tokens (i.e., with multiple incoming edges), and modeling them improves both setups. Single task performance increases substantially, from 61.13 to 63.34 F1 score. Similar significant performance gains are observed for multi-task learning, from 62.37 to 65.04 F1 score. Regardless of the multi-label modeling, the multi-task setup provides the highest overall performance.

Comparison to the state of the art We now compare the multi-task multi-label BeeSL (hereafter, simply BeeSL) to the top performing systems.

Table 5.5: Performance comparison on the test set of the standard BioNLP Genia 2011 benchmark. *indicates that the system was trained on training plus part of development data. BeeSL uses the official training portion only. Top: traditional ML systems; Middle: state-of-the-art neural systems; Bottom: proposed multi-task multi-label sequence labeling system (BeeSL).

Method	P	R	F1
Riedel et al. (2011) *FAUST – Model combination (joint+parsing)*	64.75	49.41	56.04
Miwa et al. (2012) *EventMine – SVM pipeline (+coref)*	63.48	53.35	57.98
Venugopal et al. (2014) *BioMLN – SVM pipeline*	63.61	53.42	58.07
Majumder et al. (2016) *Stacked generalization*	66.46	48.96	56.38
Björne and Salakoski (2018) *TEES – CNN pipeline (single model)*	64.86	50.53	56.80
Björne and Salakoski (2018) *TEES – CNN pipeline (5x ensemble)*	68.76	49.97	57.87
Björne and Salakoski (2018)* *TEES – CNN pipeline (mixed 5x ensemble)*	69.45	49.94	58.10
Li et al. (2019) *BiLSTM pipeline*	62.18	48.44	54.46
Li et al. (2019) *Tree-LSTM pipeline*	64.56	50.28	56.53
Li et al. (2019) *KB-driven Tree-LSTM pipeline*	67.01	52.14	58.65
BeeSL	69.72	53.00	**60.22**

As shown in Table 5.5, BeeSL outperforms the state-of-the-art by a large margin, i.e., an absolute improvement of 1.57 points in F1 score over the KB-Tree LSTM model (Li et al., 2019) (hereafter, KBTL). It improves over both precision and recall, and yields a new state of the art with an F1 score of 60.22%, yet being conceptually simple.

Table 5.6: Detailed per-event performance of BeeSL and KBTL (KB-driven TreeLSTM) on the test set. Best scores are in bold.

	BeeSL			KBTL		
Event type	P	R	F1	P	R	F1
Simple events	84.17	74.98	**79.31**	85.95	72.62	78.73
Gene expression	84.55	77.54	**80.90**	87.24	74.35	80.28
Transcription	72.50	66.67	69.46	82.31	69.54	**75.39**
Protein catabolism	83.33	66.67	**74.07**	87.50	46.67	60.87
Phosphorylation	94.05	85.41	**89.52**	87.28	81.62	84.36
Localization	83.21	59.69	**69.51**	80.28	59.69	68.47
Binding	65.36	40.73	**50.19**	53.16	37.68	44.10
Complex events	58.54	41.14	**48.32**	55.73	41.73	47.72
Regulation	62.22	36.36	**45.90**	53.61	36.62	43.52
Positive regulation	60.14	41.93	**49.41**	57.90	41.37	48.26
Negative regulation	53.19	42.38	47.17	52.39	46.06	**49.02**
All events	69.72	53.00	**60.22**	67.01	52.14	58.65

Table 5.6 compares the scores of BeeSL to the previous best model on a per-event level. BeeSL outperforms the KBTL approach (Li et al., 2019) overall on 7 out of the 9 event types. From a coarse-grained perspective, BeeSL outperforms KBTL on all simple, binding, and complex event categories. Particularly, improvements over KBTL on simple events are as large as +13% F1 score. Furthermore, noticeable are also the improvements for binding and nested, complex events, for which our model achieves 50.19% and 48.32% F1 score. From a closer look, the recall of BeeSL on simple events is substantially higher than KBTL, which ease a correct identification of complex events. The only exceptions are for -Regulation and Transcription event types, for which BeeSL's F1 scores are lower that KBTL ones. While the precision for -Regulation events is higher than KBTL, recall is substantially lower. We hypothesize that this behaviour is due to the use of external knowledge information by KBTL, which allows to retrieve more instances of this type, at the cost of precision. Moreover, we speculate that Transcription errors are caused by the ambiguity of Expression and Transcription triggers. We

Table 5.7: Speed comparison to TEES single and ensemble models at inference time. Results are sents/min, averaged over 5 runs.

	sents/min
TEES (*single*)	$255_{\pm 1}$
TEES (*ensemble*)	$101_{\pm 1}$
BeeSL	$499_{\pm 3}$

refer to Section 5.6.4 for more details.

Next, we look at performance per text type (i.e., abstract and full-text subsets). BeeSL achieves 62.14% F1 score on abstracts-only documents, and 55.59% F1 score on full-texts. This confirms that full-texts are harder to process than abstracts, due to the known differences in structural and content aspects (Cohen et al., 2010).

To sum up, BeeSL handles events well, and unlike most prior work, does not use knowledge bases or dependency parsers as pre-processing step to obtain features. BeeSL uses multi-task learning with a contextual encoder and both single- and multi-label aware decoding, herewith bringing progress to the biomedical event extraction task as illustrated in Figure 5.1.

Speed comparison We compare BeeSL to TEES, the Turku Event Extraction System (Björne & Salakoski, 2018) to compare their speed at inference time on commodity hardware. TEES is the 2nd top-performing system (Figure 5.1), and its code is freely available. To the best of our knowledge, the source code of (Li et al., 2019) is not yet available.

Results in Table 5.7 show that BeeSL is ∼2x faster and ∼5x faster on a consumer grade CPU[7] than TEES single and ensemble system, respectively. In terms of sentences per minute, BeeSL processes ∼500 sents/min compared to 255 sents/min and 101 sents/min in TEES single (3.42% lower F1) and ensemble (2.12% lower F1), respectively.

[7]Intel Core i5-6360U (2 cores).

Table 5.8: Ablation study on BeeSL when removing the multi-task capability (i.e., replacing MTL with independent classifiers) and the multi-label handling.

Setting	P	R	F1
BeeSL	71.84	59.42	**65.04**
– multi-task	71.66	56.95	63.47
– multi-label	74.28	52.39	61.44

5.6 Analysis and Discussion

To gain insights about BeeSL, we shed more light on several aspects. Firstly, we analyze how much BeeSL gains from multi-task learning, compared to using a powerful contextualized BERT encoder alone in a single-task learning setup and a formulation with two independent classifiers (Section 5.6.1). Then, we quantify the stability of the threshold τ of the multi-label decoder (Section 5.6.2). We also aim to get deeper insight on model performance in the case entities are not explicitly available (Section 5.6.3), and qualitatively study the sources of prediction errors of BeeSL (Section 5.6.4).

5.6.1 Importance of Multi-Task Learning

As opposed to running one single model which models $\langle d \rangle$ and $\langle r, h \rangle$ jointly in a multi-task setup (i.e., our best setup), we also experiment with a formulation of two single-task classifiers which predict $\langle d \rangle$ and $\langle r, h \rangle$ separately. This allows us to gauge the effectiveness of the multi-task learning approach compared to local classifiers which use strong BERT-based encoding, and compared to predicting an atomic label in single-task learning.

Results in Table 5.8 confirm that leveraging a shared encoder and multi-task learning for both $\langle d \rangle$ and $\langle r, h \rangle$ is crucial. Without multi-task learning and multi-label decoding, the F1 score drops to 61.44 in the case of independent classifiers (Table 5.8), and to 61.13 in the case of single-task learning (BeeSL$_{ST}$ in Table 5.4). Adding the multi-label decoding capability helps, as expected. Independent classifiers achieve 63.47% F1 score (Table 5.8), whereas single-task learning reaches 63.34% F1 score (BeeSL$_{ST}$ with multi-label in Table 5.4).

Table 5.9: Ablation study on the threshold τ of the multi-label decoder ("with best-only predicion": $\tau = 1.0$).

Setting	F1	Δ
BEESL$_{ST}$ (multi-label)	63.34	
with best-only prediction	62.87	-0.47
BEESL$_{MT}$ (multi-label)	**65.04**	
with best-only prediction	64.54	-0.50

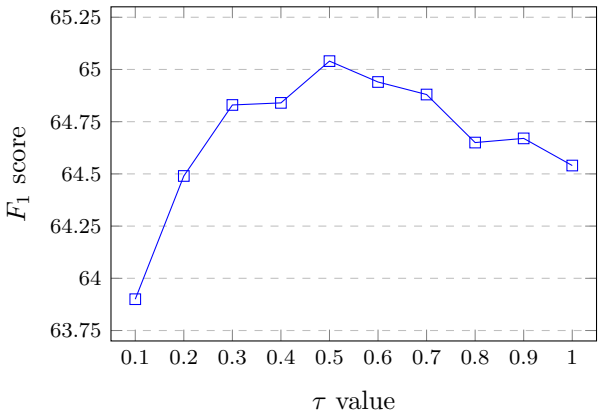


Figure 5.4: Stability of the threshold τ. Values in the range 0.3-0.7 only minimally alter BEESL scores.

However, the full power of BEESL is only achieved by using *both* the multi-task and the multi-label approach, which leads to the novel state of the art.

5.6.2 Stability of the Multi-Label Decoder

As shown in Table 5.4, using a multi-label decoder largely increases the performance of a system with a single-label decoder (from 62.37 to 65.04 F1 score). However, what is left is how much the threshold τ impacts the performance. To get insights on it, we firstly performed an ablation study setting $\tau = 1.0$. As introduced in Section 5.3.2.2, this reduces to predicting the highest scoring label only – however, in a reduced label space induced by the multi-label decoder

Table 5.10: Performance of BeeSL with *no* gold entities.

	P	R	F1
BeeSL	71.84	59.42	**65.04**
– gold entities	66.15	54.09	59.51

(meaning that k labels per token are treated separately, i.e., $y_i = \{y_i^1, ..., y_i^k\}$ instead of being considered an atomic concatenated unit, i.e., $y_i = y_i^1 \oplus ... \oplus y_i^k$). We positively found that only part of the improvement given by the multi-label decoding is due to the threshold τ in both multi-task and single-task settings (+0.50% and +0.47%, respectively) (Table 5.9).

Moreover, we evaluated BeeSL with different τ values. As shown in Figure 5.4, a threshold in the range 0.3-0.7 only marginally alters the results, which are still better than predicting the highest scoring label only ($\tau = 1.0$). This confirms the importance of using a multi-label aware decoder as well as the stability of the threshold in predicting the output labels.

5.6.3 Impact of Gold Entity Information

The standard in biomedical event extraction is to evaluate the performance of a system on raw texts paired with gold entity annotations (i.e., entities that are given in advance). This makes sense to isolate the task and to assess the actual event extraction performance, as well as to be able to compare results to previous work in the field. However, in real-world situations it is unlikely that the data is annotated for entities. We believe it is important to estimate the system performance in case of predicted, non-gold entities (hereafter, *silver entities*). The performance of the entity mention prediction (cf. Section 5.4) on the development set is 87.95 span-based F1 score.

The results on the event extraction task using silver entities are shown in Table 5.10. The overall drop in F1 amounts to around 5%, and it is well-balanced across precision and recall. This shows that BeeSL's performance is clearly affected, but that the system is relatively robust to noisy, non-gold *silver entities*. We believe that this performance gap can be further minimized by using jackknifing (Agić & Schluter, 2017) to reduce data mismatch, however,

Table 5.11: Error analysis on a random sample of 30 documents from the development set.

Error type	Fraction
Trigger	
Under-prediction	31.43%
Over-prediction	28.57%
Wrong type	10.00%
Argument	
Under-prediction	22.86%
Over-prediction	7.14%
Wrong type	0.00%

this requires to align the predicted entities with the existing events in the training data, which is non-trivial, and we leave this for future work.

5.6.4 Error Analysis

We randomly sampled a total of 30 documents (comprising 168 gold events) from the development set for a manual scrutiny for sources of errors. We classified errors into two broad categories, namely trigger and argument errors. Further, we classified them in fine-grained categories based on the type of error, namely under-prediction, over-prediction, and wrong type. Table 5.11 summarizes the results.

We notice the largest fraction of errors is due to trigger errors. From a closer look, under-predicted triggers account for 31.43% of the total, whereas over-predicted triggers for 28.57%. We investigated the reason for these errors, finding that over-predicted triggers are often due to generic words used very frequently to indicate specific trigger types. For instance, BeeSL identifies the +Regulation event anchored at "activated" in the following sentence:

> *Tax co-transfected with reporter constructs into Jurkat cells maximally* ***activated*** *HTLV-I-LTR-CAT and kappa B-fos-CA.*
>
> (Sentence ID: PMID-7505113)

albeit the gold standard does not contain the event in this instance. However, from a semantic point of view we believe these errors are acceptable. Other cases include the words such as "detected" and "influences", which are often used as EXPRESSION and REGULATION event triggers, respectively.

Under-prediction of triggers is instead due to a variety of reasons. Both rare words (e.g., a +REGULATION event centered on "co-transfected") and uncertain events account for a large fraction of this error type. An example of uncertain event is presented in the following sentence:

> *The observation that a mutated LT-kappa B construct (M1-CAT) was inactive in C81-66-45, confirmed the* ***importance*** *of NF-kappa B in LT gene expression.* (Sentence ID: PMID-7505113)

where the +REGULATION trigger "importance" is not recognized by BEESL.

Wrongly typed triggers represent only 10% of the errors. An example is represented by ambiguous trigger types. In the sentence:

> *There is a report describing that mitogenic stimulation of T cells upregulates A3G mRNA* ***levels****.*
> (Sentence ID: PMC-1920263-01-INTRODUCTION)

BEESL classifies "levels" as an EXPRESSION trigger, while the gold annotation indicates it is a TRANSCRIPTION trigger. By a closer look, we found some triggers in the corpora are annotated as EXPRESSION and TRANSCRIPTION types interchangeably. This is due to the fact that a TRANSCRIPTION, from a biological point of view, is actually a kind of gene EXPRESSION.

Regarding the identification of arguments, over-predictions are quite uncommon. If erroneous, the main error we found may benefit from syntactic information, which we aim to integrate in a multi-task setup in future work. We found no misclassification of arguments in our document samples. Lastly, under-prediction of arguments is instead mostly due to under-predicted events.

5.7 Summary and Conclusions

In this chapter we described BeeSL, a novel end-to-end biomedical event extraction system which is both efficient and accurate. BeeSL is broadly applicable to event extraction and other tasks that can be recast as sequence labeling. The system's strength comes from the joint multi-task modeling paired with multi-label decoding, which aids interdependencies between the tasks and is superior to alternative decoders based on strong contextualized BERT embeddings. BeeSL is fast, and achieves state-of-the-art performance on the standard Genia 2011 event extraction benchmark without the need of external tools for features and resources such as parsers and knowledge bases. Our analysis shows that BeeSL works very well across event types. Future work include the experimentation using BeeSL in low-resource data setups, and in the context of other publicly available datasets with different definitions of events. We also release the code freely, to foster research on using BeeSL for other central NLP tasks as well, such as enhanced dependency parsing, fine-grained named entity recognition, and semantic parsing. General conclusions are provided in the next chapter (Chapter 6).

Chapter 6

Summary and Conclusions

In this thesis, we focused on designing effective means for extracting semantic relational information from biomedical literature texts, in order to assist field experts in keeping pace with the growing volume of biomedical knowledge being published. After providing background information on natural language processing and introducing symbolic approaches and relevant deep and transfer learning methods (Chapter 2), we focused on the central tasks for biomedical information extraction, namely biomedical relation and event extraction, and particularly on specific challenges towards a reliable and smooth adoption of knowledge extraction solutions in real-world scenarios. We summarize our findings that aim to answer the research questions outlined in Chapter 1 in the following synopsis, then focusing on future research directions in the field.

Specifically, in Chapter 3 we designed a symbolic approach for biomedical relation extraction which leverages syntactic dependency trees and surface linguistic information such as part-of-speech tags by means of carefully designed patterns and rules. The aim of this study was to investigate if symbolic methods for relation extraction may reduce the time-consuming manual scrutiny of false positive relation instances, a typical desideratum of biomedical practitioners to readily exploit the automatically extracted evidence for further biologically-driven analyses. Empirical results are encouraging, showing that our symbolic method achieves the highest precision score – while maintaining

a comparable or higher F_1 score – compared to both previous rule-based and machine learning alternatives, as well as to more recent transfer learning approaches we specifically fine-tuned to the purpose (which however show very high overall performance on the task). Therefore, the answer to the research question RQ_1 is positive, since manual curation is drastically reduced by our symbolic approach, particularly when annotated data is scarce. In future work we aim to investigate if the advantage of our approach over machine learning ones also holds in high-resource regimes. Despite the good results, a detailed error analysis highlighted that there are cases that rules still do not capture due to very complex linguistic constructs, and that the expressivity of binary relations is somehow limited in modeling fine-grained, high-order relations (i.e., relations that are argument of other relations). These facts provided us the motivation to focus on the more expressive biomedical event extraction task, and to employ deep and transfer learning methods.

Building on top of the aforementioned findings, in Chapter 4 we investigated the cross-domain robustness of deep learning solutions typically employed for biomedical event extraction. Specifically, we focused on the most challenging biomedical event extraction sub-task, namely edge detection, whose evaluation is not typically reported in literature. We employed convolutional neural networks since they have been shown to be more robust across linguistic variations compared to previous machine learning alternatives (T. H. Nguyen & Grishman, 2015), enriching it with syntactic and semantic input embeddings and setting it as strong baseline for the study. To quantify the drop in performance when dealing with out-of-distribution data, we trained multiple models on in-domain corpora, and we tested their robustness when applied on all other corpora. Concretely, we answered the research question RQ_2 by finding a drop in performance up to 32.97 and 38.25 points in micro and macro F_1 score, respectively, which highlighted the importance of accounting and tackling linguistic varieties in future work to ensure a high out-of-distribution robustness of methods. We thus released standardized benchmark data to encourage future research in cross-domain generalization of edge detection in biomedical event extraction pipelines. Future avenues for research in this direction include the design of robust edge detection models based on transformer-based

architectures, and domain adaptation techniques to handle domain variation.

The previous contribution to the BioNLP community posed the motivation to design an end-to-end solution for biomedical event extraction, in order to overcome the limitations of traditional classifier pipelines, namely the error cascading issues and the lack of information sharing between sub-tasks. Also, the recent outstanding achievements of transfer learning methods in NLP (A. Wang, Pruksachatkun, et al., 2019), and the substantially increased out-of-distribution performance of these methods (Hendrycks et al., 2020), motivated us to blend these ideas together. Specifically, in Chapter 5 we investigated if the highly complex biomedical event extraction task may be tackled in an end-to-end fashion with both sequential transfer learning ideas such as BERT (Devlin et al., 2019) and a multi-task learning paradigm, while improving both speed and accuracy compared to previously proposed solutions. To the goal, we proposed Biomedical Event Extraction as Sequence Labeling (BeeSL), an end-to-end approach that transforms event structures into token-level labels, thus modeling the sub-tasks of trigger and edge detection jointly while implicitly leveraging inter-dependencies between them and mitigating error cascading. Instead of evaluating pairwise associations multiple times, BeeSL reads the input sequence only once, and also dispenses external resources and pre-processing tools for obtaining features. A BERT encoder pre-trained on biomedical literature (Lee et al., 2020) is fine-tuned to produce domain-relevant contextualized representations for the task, and specifically purposed decoders such as a multi-label decoder are devised on top of the encoder and trained in a multi-task setup. Our end-to-end approach to biomedical event extraction successfully answered the research question RQ_3, achieving state-the-art results on the standard biomedical event extraction GENIA benchmark while showing an increase up to $5\times$ in speed efficiency at inference time compared to the previous best solution, making it a suitable approach for large-scale real-world applications. Additional insights such as the importance of multi-task learning and multi-label decoding and the impact of gold entity information complemented the work, opening interesting future directions to the broader NLP community, such that the repurposing of the linearization approach to other structured tasks such as enhanced dependency parsing, fine-

grained named entity recognition, and semantic parsing.

In summary, in this thesis we have made several contributions to advance the BioNLP field, by tackling both the tasks of biomedical relation extraction (Chapter 3) and biomedical event extraction (Chapters 4 and 5), and providing novel solutions as well as relevant insights for real-world adoption of knowledge extraction in the biomedical field, including (i) a high-precision relation extraction approach to reduce the time-consuming manual curation efforts of the extracted relation instances (Chapter 3), (ii) a thorough cross-domain study to quantify the drop in performance of biomedical edge detection methods when applied to potentially different linguistic varieties (Chapter 4), and (iii) a fast and accurate solution for biomedical event extraction that makes it a viable approach to be used in real-world large-scale applications (Chapter 5). Concretely, we believe end users might benefit from these insights in a practical setting. The work in Chapter 3 makes a step towards allowing biomedical practitioners to focus more on downstream analyses, rather than curation. The work in Chapter 4 might instead sensitize the research community on model robustness in BioNLP, which in turn will be of practical usefulness to biomedical experts to apply models in a variety of sub-domains. Finally, the event extraction system proposed in Chapter 5 makes the extraction of events substantially faster, making it an attractive solution when large-scale extraction is needed or computational resources are limited. We have thus explored complementary views of "goodness" of BioNLP models, including robustness and efficiency, issues that are central for real-world adoption of BioNLP systems.

We believe both the tasks of relation and event extraction have their own strengths for extracting relational knowledge in biomedical literature. Although limited in representational power, biomedical relation extraction is a simpler formalism that can be employed in an early stage by field practitioners to explore the biomedical literature and find dense clusters of relevant information to focus on (e.g., by means of downstream applications such as knowledge graphs). On the other hand, biomedical event extraction is a fine-grained alternative to relation extraction that can better inform researchers on specific details about novel biomedical findings, such as nested reactions and interactions, thus being highly effective when a specific research question is set.

Besides representational strengths and weaknesses, we believe the reporting of human performance on the corpora would be extremely valuable for assessing the progress in both the tasks, and eventually determine when benchmarks saturate. Although this is not central to this work, we found that not all the corpora are accompanied by an inter-annotator agreement score (Appendix D). On the basis of the agreement reported for HPRD50 (81% F_1, Appendix D), we speculate our relation extraction method (79.5% F_1 on HPRD50) is close to an empirical upper bound. On the contrary, an average estimate of the inter-annotator agreement reported for event extraction corpora ($\sim$70–75% F_1, Appendix D) suggests there is still room for improvement in the field.

We identified some potential avenues for future research that we outline in the following. As common practice in relation and event extraction and their standard benchmarks for evaluation, in this thesis we assume entity mentions are given. However, devising unified information extraction systems that predict both entities and their relations (or events) may be beneficial. While for relation extraction this has been explored in recent times (Bekoulis et al., 2018b), for event extraction this integration is not straightforward, mainly due to the fact that event extraction itself comprises multiple inter-related tasks. As a step towards this goal, in this thesis we provide the first results on biomedical event extraction without assuming gold entities (Chapter 5). An interesting future direction includes learning entity mentions and events together following a multi-task paradigm. We envision an increasing number of such joint methods in the near future.

Another interesting direction is about reliably applying current methods to specific sub-domains of biomedicine in which annotations are not abundant or readily available. This translates to drastically reducing the need for annotated training data. With the recent rise of sequential transfer learning approaches such as BERT (Devlin et al., 2019), we positively observed impressive performance gains in a wide array of NLP tasks (A. Wang et al., 2018), as well as a reduced need for annotated target data for reaching the same performance on a target task compared to training a model from scratch (Howard & Ruder, 2018). However, learning in extreme scenarios where annotated data is very scarce or absent, namely low-resource regimes, is still an open problem and

should be extensively tackled in the near future (Hedderich et al., 2020). An interesting research idea for future work is to simulate a low-resource scenario for relational information extraction tasks, in order to assess the amount of labeled data that is needed to reach a sufficient accuracy on a target task, i.e., by maximizing the annotation effort–performance tradeoff.

Another research perspective includes the design of neuro-symbolic methods for NLP. As presented in the previous chapters, we employed symbolic and deep transfer learning approaches, each of which has its own advantages. Symbolic methods are transparent but are brittle and exhibit a rigid behaviour, whereas deep transfer learning methods are opaque but are highly performant solutions which are able to capture hidden patterns from data automatically. Blending these two lines together is seeing an increasing interest in the broader machine learning community (d'Avila Garcez & Lamb, 2020), and we envision a rapid development of these techniques in the next decade. Additionally, softening the rigid behaviour of binary decisions may help in reducing errors caused by misclassification of early assessments in a processing pipeline. An interesting research direction is to employ a probabilistic assessment of the conditions of the relation router presented in Figure 3.2 (Chapter 3).

A last challenge is about going towards multilingual biomedical knowledge extraction. In this thesis we focused on *English* biomedical literature, since it is the common language used to communicate science. However, some research studies are only published in other languages, mainly those pertaining to the Chinese language family. Accounting for the over 7,000 languages in the world is a long-term goal of current NLP (Joshi et al., 2020). While large pre-trained multilingual models already exist for the general domain, e.g., multilingual BERT (Devlin et al., 2019), to the best of our knowledge no transformer-based model has been yet developed for biomedical texts in multiple languages, in order to be ultimately used as backbone for complex information extraction systems. We believe extracting information of biomedical relevance from texts in multiple languages will be one of the next drivers of progress in BioNLP. In the short term, we envision a shift from monolingual to bilingual or multilingual relation and event extraction systems to support biomedicine.

References

Abacha, A. B., Chowdhury, M. F. M., Karanasiou, A., Mrabet, Y., Lavelli, A., & Zweigenbaum, P. (2015). Text Mining for Pharmacovigilance: Using Machine Learning for Drug Name Recognition and Drug–Drug Interaction Extraction and Classification. *Journal of Biomedical Informatics*, *58*, 122–132.

Agić, Ž., & Schluter, N. (2017). How (Not) to Train a Dependency Parser: The Curious Case of Jackknifing Part-of-Speech Taggers. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics: Volume 2, Short Papers (ACL)* (pp. 679–684). Vancouver, Canada: Association for Computational Linguistics.

Ahmed, M., Islam, J., Samee, M. R., & Mercer, R. E. (2019). Identifying Protein-Protein Interaction using Tree LSTM and Structured Attention. In *2019 IEEE 13th International Conference on Semantic Computing (ICSC)* (pp. 224–231). Newport Beach, CA, USA: IEEE.

Alt, C., Hübner, M., & Hennig, L. (2019). Improving Relation Extraction by Pre-trained Language Representations. In *Proceedings of Automated Knowledge Base Construction (AKBC) 2019.* Amherst, MA, USA.

Ananiadou, S., Pyysalo, S., Tsujii, J., & Kell, D. B. (2010). Event Extraction for Systems Biology by Text Mining the Literature. *Trends in Biotechnology*, *28*(7), 381–390.

Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer Normalization. *arXiv preprint arXiv:1607.06450*.

Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. In *3rd International Conference*

on Learning Representations (ICLR), Conference Track Proceedings. San Diego, CA, USA.

Baldwin, T., Cook, P., Lui, M., MacKinlay, A., & Wang, L. (2013). How Noisy Social Media Text, How Diffrnt Social Media Sources? In *Proceedings of the Sixth International Joint Conference on Natural Language Processing (IJCNLP)* (pp. 356–364). Nagoya, Japan: Asian Federation of Natural Language Processing.

Bekoulis, G., Deleu, J., Demeester, T., & Develder, C. (2018a). Adversarial Training for Multi-context Joint Entity and Relation Extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 2830–2836). Brussels, Belgium: Association for Computational Linguistics.

Bekoulis, G., Deleu, J., Demeester, T., & Develder, C. (2018b, 04). Joint Entity Recognition and Relation Extraction as a Multi-head Selection Problem. *Expert Systems with Applications*, *114*, 34-45.

Beltagy, I., Lo, K., & Cohan, A. (2019). SciBERT: A Pretrained Language Model for Scientific Text. 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)* (pp. 3615–3620). Hong Kong, China: Association for Computational Linguistics.

Bender, E. M., & Koller, A. (2020). Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 5185–5198). Online: Association for Computational Linguistics.

Bengio, Y. (2019). *Deep Learning for AI.* Retrieved 2020-10-07, from `http://www.iro.umontreal.ca/~bengioy/HLF-Turing-23sept2019.pdf` (Turing Award Lecture, Heidelberg Laureate Forum)

Bhasuran, B., & Natarajan, J. (2018). Automatic Extraction of Gene-Disease Associations from Literature using Joint Ensemble Learning. *PloS one*, *13*(7), e0200699.

Binder, J. X., Pletscher-Frankild, S., Tsafou, K., Stolte, C., O'Donoghue, S. I., Schneider, R., & Jensen, L. J. (2014). COMPARTMENTS: Unification

and Visualization of Protein Subcellular Localization Evidence. *Database*, *2014*.

Bingel, J., & Søgaard, A. (2017). Identifying Beneficial Task Relations for Multi-task Learning in Deep Neural Networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers (EACL)* (pp. 164–169). Valencia, Spain: Association for Computational Linguistics.

Björne, J. (2014). Biomedical Event Extraction with Machine Learning. Ph.D. Thesis. University of Turku, Finland: TUCS Dissertations.

Björne, J., & Salakoski, T. (2011). Generalizing Biomedical Event Extraction. In *Proceedings of BioNLP Shared Task 2011 Workshop* (pp. 183–191). Portland, Oregon, USA: Association for Computational Linguistics.

Björne, J., & Salakoski, T. (2015). TEES 2.2: Biomedical Event Extraction for Diverse Corpora. *BMC Bioinformatics*, *16*(S16), S4.

Björne, J., & Salakoski, T. (2018). Biomedical Event Extraction Using Convolutional Neural Networks and Dependency Parsing. In *Proceedings of the BioNLP 2018 Workshop* (pp. 98–108). Melbourne, Australia: Association for Computational Linguistics.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, *3*(Jan), 993–1022.

Brown, P. F., Della Pietra, V. J., deSouza, P. V., Lai, J. C., & Mercer, R. L. (1992). Class-based n-gram Models of Natural Language. *Computational Linguistics*, *18*(4), 467–480.

Bundschus, M., Dejori, M., Stetter, M., Tresp, V., & Kriegel, H.-P. (2008). Extraction of Semantic Biomedical Relations from Text Using Conditional Random Fields. *BMC Bioinformatics*, *9*(1), 207.

Canese, K., & Weis, S. (2013). PubMed: The Bibliographic Database. In *The NCBI Handbook [Internet]. 2nd edition.* National Center for Biotechnology Information (US).

Caruana, R. (1993). Multitask Learning: A Knowledge-Based Source of Inductive Bias. In *Proceedings of the Tenth International Conference on Machine Learning (ICML)* (pp. 41–48). Amherst, MA, USA: Morgan Kaufmann.

Caruana, R. (1997). Multitask Learning. *Machine Learning*, *28*(1), 41–75.

Chang, Y.-C., Chu, C.-H., Su, Y.-C., Chen, C. C., & Hsu, W.-L. (2016). PIPE: A Protein–Protein Interaction Passage Extraction Module for BioCreative Challenge. *Database*, *2016*.

Changpinyo, S., Hu, H., & Sha, F. (2018). Multi-Task Learning for Sequence Tagging: An Empirical Study. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING)* (pp. 2965–2977). Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Cheng, D., Knox, C., Young, N., Stothard, P., Damaraju, S., & Wishart, D. S. (2008, 05). 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.

Chiu, B., Crichton, G., Korhonen, A., & Pyysalo, S. (2016). How to Train good Word Embeddings for Biomedical NLP. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing* (pp. 166–174). Berlin, Germany: Association for Computational Linguistics.

Cohen, K. B., & Demner-Fushman, D. (2014). *Biomedical Natural Language Processing*. John Benjamins.

Cohen, K. B., Johnson, H. L., Verspoor, K., Roeder, C., & Hunter, L. E. (2010). The Structural and Content Aspects of Abstracts Versus Bodies of Full Text Journal Articles are Different. *BMC Bioinformatics*, *11*(1), 492.

Collier, N., Park, H. S., Ogata, N., Tateishi, Y., Nobata, C., Ohta, T., ... Tsujii, J.-i. (1999). The GENIA Project: Corpus-based Knowledge Acquisition and Information Extraction from Genome Research Papers. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics (EACL)* (pp. 271–272). Bergen, Norway: Association for Computational Linguistics.

Collobert, R., & Weston, J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML)* (pp. 160–167). Helsinki, Finland.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural Language Processing (Almost) from Scratch. *Journal*

of Machine Learning Research, *12*, 2493–2537.

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, *20*(3), 273–297.

Dai, D., Xiao, X., Lyu, Y., Dou, S., She, Q., & Wang, H. (2019). Joint Extraction of Entities and Overlapping Relations Using Position-Attentive Sequence Labeling. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence* (pp. 6300–6308). Honolulu, HI, USA.

Daumé III, H. (2007). Frustratingly Easy Domain Adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL)* (pp. 256–263). Prague, Czech Republic: Association for Computational Linguistics.

d'Avila Garcez, A., & Lamb, L. C. (2020). Neurosymbolic AI: The 3rd Wave. *arXiv preprint arXiv:2012.05876*.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies: Volume 1, Long and Short Papers (NAACL-HLT)* (pp. 4171–4186). Minneapolis, Minnesota, USA: Association for Computational Linguistics.

Ding, J., Berleant, D., Nettleton, D., & Wurtele, E. (2001). Mining MEDLINE: Abstracts, Sentences, or Phrases? In *Proceedings of the Pacific Symposium on Biocomputing 2002 (PSC)* (pp. 326–337). Kauai, HI, USA: World Scientific.

Dos Santos, C., & Zadrozny, B. (2014). Learning Character-level Representations for Part-of-Speech Tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML)* (pp. 1818–1826). Beijing, China.

Eisenstein, J. (2013). What to do About Bad Language on the Internet. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies (NAACL-HLT)* (pp. 359–369). Atlanta, Georgia, USA: Association for Computational Linguistics.

Eisenstein, J. (2019). *Introduction to Natural Language Processing*. MIT Press.

Firth, J. R. (1957). A synopsis of Linguistic Theory, 1930-1955. *Studies in Linguistic Analysis*.

Foster, J., Cetinoglu, O., Wagner, J., Le Roux, J., Hogan, S., Nivre, J., ... Van Genabith, J. (2011). #hardtoparse: POS Tagging and Parsing the Twitterverse. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence: Analyzing Microtext.* San Francisco, CA, USA.

Francis, W. N., & Kucera, H. (1979). *Brown Corpus Manual* (Tech. Rep.). Department of Linguistics, Brown University, Providence, Rhode Island, US.

Fundel, K., Küffner, R., & Zimmer, R. (2007). RelEx–Relation Extraction Using Dependency Parse Trees. *Bioinformatics*, *23*(3), 365–371.

Gardner, M., Grus, J., Neumann, M., Tafjord, O., Dasigi, P., Liu, N. F., ... Zettlemoyer, L. (2018). AllenNLP: A Deep Semantic Natural Language Processing Platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)* (pp. 1–6). Melbourne, Australia: Association for Computational Linguistics.

Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., ... Smith, N. A. (2011). Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics – Human Language Technologies (ACL-HLT)* (pp. 42–47). Portland, Oregon, USA: Association for Computational Linguistics.

Goldberg, Y. (2017). Neural Network Methods for Natural Language Processing. *Synthesis Lectures on Human Language Technologies*, *10*(1), 1–309.

Gómez-Rodríguez, C., & Vilares, D. (2018). Constituent Parsing as Sequence Labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1314–1324). Brussels, Belgium: Association for Computational Linguistics.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* MIT Press.

Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., & Smith, N. A. (2020). Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 8342–8360).

Online: Association for Computational Linguistics.

Harris, Z. S. (1954). Distributional Structure. *Word*, *10*(2-3), 146–162.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778). Las Vegas, Nevada, USA.

Hearst, M. A. (1992). Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proocedings of the 15th International Conference on Computational Linguistics: Volume 2 (COLING).* Nantes, France.

Hedderich, M. A., Lange, L., Adel, H., Strötgen, J., & Klakow, D. (2020). A Survey on Recent Approaches for Natural Language Processing in Low-Resource Scenarios. *arXiv preprint arXiv:2010.12309*.

Henderson, J. (2020). The Unstoppable Rise of Computational Linguistics in Deep Learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 6294–6306). Online: Association for Computational Linguistics.

Hendrycks, D., Liu, X., Wallace, E., Dziedzic, A., Krishnan, R., & Song, D. (2020). Pretrained Transformers Improve Out-of-Distribution Robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 2744–2751). Online: Association for Computational Linguistics.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780.

Hornik, K., Stinchcombe, M., White, H., et al. (1989). Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, *2*(5), 359–366.

Howard, J., & Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics: Volume 1: Long Papers (ACL)* (pp. 328–339). Melbourne, Australia: Association for Computational Linguistics.

Huang, C.-C., & Lu, Z. (2016). Community Challenges in Biomedical Text Mining Over 10 Years: Success, Failure and the Future. *Briefings in Bioinformatics*, *17*(1), 132–144.

Joshi, P., Santy, S., Budhiraja, A., Bali, K., & Choudhury, M. (2020). The State and Fate of Linguistic Diversity and Inclusion in the NLP World. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 6282–6293). Online: Association for Computational Linguistics.

Jurafsky, D., & Martin, J. H. (2020). *Speech and Language Processing (3rd Edition)*. Pearson Prentice Hall.

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A Convolutional Neural Network for Modelling Sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: Volume 1, Long Papers (ACL)* (pp. 655–665). Baltimore, Maryland: Association for Computational Linguistics.

Kemper, B., Matsuzaki, T., Matsuoka, Y., Tsuruoka, Y., Kitano, H., Ananiadou, S., & Tsujii, J. (2010). PathText: A Text Mining Integrator for Biological Pathway Visualizations. *Bioinformatics*, *26*(12), i374–i381.

Kim, J.-D., Ohta, T., & Tsujii, J. (2008). Corpus Annotation for Mining Biomedical Events from Literature. *BMC Bioinformatics*, *9*(1), 1–25.

Kim, J.-D., Wang, Y., Takagi, T., & Yonezawa, A. (2011). Overview of Genia Event Task in BioNLP Shared Task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop* (pp. 7–15). Portland, Oregon, USA: Association for Computational Linguistics.

Kim, S., Yoon, J., Yang, J., & Park, S. (2010). Walk-Weighted Subsequence Kernels for Protein-Protein Interaction Extraction. *BMC Bioinformatics*, *11*(1), 107.

Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1746–1751). Doha, Qatar: Association for Computational Linguistics.

Kondratyuk, D., & Straka, M. (2019). 75 Languages, 1 Model: Parsing Universal Dependencies Universally. 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)* (pp. 2779–2795). Hong Kong, China: Association for Computational Linguis-

tics.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet Classification with Deep Convolutional Neural Networks. *Communications of the ACM*, *60*(6), 84–90.

Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An Introduction to Latent Semantic Analysis. *Discourse Processes*, *25*(2-3), 259–284.

Le, Q. V., Jaitly, N., & Hinton, G. E. (2015). A Simple Way to Initialize Recurrent Networks of Rectified Linear Units. *arXiv preprint arXiv:1504.00941*.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep Learning. *Nature*, *521*(7553), 436–444.

LeCun, Y., Bengio, Y., et al. (1995). Convolutional Networks for Images, Speech, and Time Series. *The Handbook of Brain Theory and Neural Networks*, *3361*(10), 1995.

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based Learning Applied to Document Recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., & Kang, J. (2020). BioBERT: A Pre-trained Biomedical Language Representation Model for Biomedical Text Mining. *Bioinformatics*, *36*(4), 1234–1240.

Lever, J., & Jones, S. J. (2016). VERSE: Event and Relation Extraction in the BioNLP 2016 Shared Task. In *Proceedings of the 4th BioNLP Shared Task Workshop* (pp. 42–49). Berlin, Germany: Association for Computational Linguistics.

Li, D., Huang, L., Ji, H., & Han, J. (2019). Biomedical Event Extraction Based on Knowledge-driven Tree-LSTM. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies: Volume 1, Long and Short Papers (NAACL-HLT)* (pp. 1421–1430). Minneapolis, Minnesota, USA: Association for Computational Linguistics.

Lin, Y., Ji, H., Huang, F., & Wu, L. (2020). A Joint Neural Model for Information Extraction with Global Features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 7999–8009). Online: Association for Computational Linguistics.

Lippincott, T., Ó Séaghdha, D., Sun, L., & Korhonen, A. (2010). Exploring Variation Across Biomedical Subdomains. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)* (pp. 689–697). Beijing, China: COLING 2010 Organizing Committee.

Lippincott, T., Séaghdha, D. Ó., & Korhonen, A. (2011). Exploring Subdomain Variation in Biomedical Language. *BMC Bioinformatics*, *12*(1), 212.

Majumder, A., Ekbal, A., & Naskar, S. K. (2016). Biomolecular Event Extraction Using a Stacked Generalization based Classifier. In *Proceedings of the 13th International Conference on Natural Language Processing (ICON)* (pp. 55–64). Varanasi, India: NLP Association of India.

Maloney, C., Sequeira, E., Kelly, C., Orris, R., & Beck, J. (2013). PubMed Central. In *The NCBI Handbook [Internet]. 2nd edition.* National Center for Biotechnology Information (US).

Manning, C. (2015). Last Words: Computational Linguistics and Deep Learning. *Computational Linguistics*, *41*(4), 701–707.

Manning, C., & Schutze, H. (1999). *Foundations of Statistical Natural Language Processing.* MIT Press.

Marcus, M. P., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, *19*(2), 313–330.

McClosky, D., Surdeanu, M., & Manning, C. (2011). Event Extraction as Dependency Parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics – Human Language Technologies (ACL-HLT)* (pp. 1626–1635). Portland, Oregon, USA: Association for Computational Linguistics.

McCulloch, W. S., & Pitts, W. (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. In *International Conference on Learning Representations (ICLR), Conference Track Proceedings.* Scottsdale, Arizona, USA.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Dis-

tributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems (NeurIPS)* (Vol. 26, pp. 3111–3119). Lake Tahoe, Nevada, USA: Curran Associates, Inc.

Mitchell, T. M. (1997). *Machine Learning*. New York: McGraw-Hill.

Miwa, M., & Ananiadou, S. (2015). Adaptable, High Recall, Event Extraction System with Minimal Configuration. *BMC Bioinformatics*, *16*(S10), S7.

Miwa, M., Pyysalo, S., Ohta, T., & Ananiadou, S. (2013). Wide Coverage Biomedical Event Extraction Using Multiple Partially Overlapping Corpora. *BMC Bioinformatics*, *14*(1), 175.

Miwa, M., Sætre, R., Miyao, Y., & Tsujii, J. (2009). Protein–Protein Interaction Extraction by Leveraging Multiple Kernels and Parsers. *International Journal of Medical Informatics*, *78*(12), e39–e46.

Miwa, M., Thompson, P., & Ananiadou, S. (2012). Boosting Automatic Event Extraction from the Literature Using Domain Adaptation and Coreference Resolution. *Bioinformatics*, *28*(13), 1759–1765.

Miwa, M., Thompson, P., Korkontzelos, I., & Ananiadou, S. (2014). Comparable Study of Event Extraction in Newswire and Biomedical Domains. In *Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers (COLING)* (pp. 2270–2279). Dublin, Ireland: Dublin City University and Association for Computational Linguistics.

Murugesan, G., Abdulkadhar, S., & Natarajan, J. (2017). Distributed Smoothed Tree Kernel for Protein-Protein Interaction Extraction from the Biomedical Literature. *PLoS One*, *12*(11), e0187379.

Nédellec, C. (2005). Learning Language in Logic – Genic Interaction Extraction Challenge. In *Learning Language in Logic Workshop (LLL05)*. Bonn, Germany.

Nédellec, C., Vetah, M. O. A., & Bessieres, P. (2001). Sentence Filtering for Information Extraction in Genomics, a Classification Problem. In *European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)* (pp. 326–337). Freiburg, Germany.

Neumann, M., King, D., Beltagy, I., & Ammar, W. (2019). ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *Pro-*

ceedings of the 18th BioNLP Workshop and Shared Task (pp. 319–327). Florence, Italy: Association for Computational Linguistics.

Nguyen, D. Q., & Verspoor, K. (2019). From POS Tagging to Dependency Parsing for Biomedical Event Extraction. *BMC Bioinformatics*, *20*(1), 72.

Nguyen, T. H., & 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 (ACL-IJCNLP)* (pp. 365–371). Beijing, China: Association for Computational Linguistics.

Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajič, J., Manning, C. D., ... Zeman, D. (2016). Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC)* (pp. 1659–1666). Portorož, Slovenia: European Language Resources Association (ELRA).

Ohta, T., Pyysalo, S., Miwa, M., Kim, J.-D., & Tsujii, J. (2010). Event Extraction for Post-translational Modifications. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing* (pp. 19–27). Uppsala, Sweden: Association for Computational Linguistics.

Ohta, T., Pyysalo, S., Rak, R., Rowley, A., Chun, H.-W., Jung, S.-J., ... Tsujii, J. (2013). Overview of the Pathway Curation (PC) Task of BioNLP Shared Task 2013. In *Proceedings of the BioNLP Shared Task 2013 Workshop* (pp. 67–75). Sofia, Bulgaria: Association for Computational Linguistics.

Ohta, T., Pyysalo, S., & 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* (pp. 16–25). Portland, Oregon, USA: Association for Computational Linguistics.

Ono, T., Hishigaki, H., Tanigami, A., & Takagi, T. (2001). Automated Extraction of Information on Protein-Protein Interactions from the Biological Literature. *Bioinformatics*, *17*(2), 155-161.

Pan, S. J., & Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345-1359.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... others (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing systems (NeurIPS)* (pp. 8026–8037). Vancouver, Canada.

Peri, S., Navarro, J. D., Kristiansen, T. Z., Amanchy, R., Surendranath, V., Muthusamy, B., ... others (2004). Human Protein Reference Database as a Discovery Resource for Proteomics. *Nucleic Acids Research*, *32*(suppl_1), D497–D501.

Perkins, D. N., Salomon, G., et al. (1992). Transfer of Learning. *International Encyclopedia of Education*, *2*, 6452–6457.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies: Volume 1, Long Papers (NAACL-HLT)* (pp. 2227–2237). New Orleans, Louisiana, USA: Association for Computational Linguistics.

Phan, T. T. T., & Ohkawa, T. (2016). Protein-Protein Interaction Extraction with Feature Selection by Evaluating Contribution Levels of Groups Consisting of Related Features. *BMC Bioinformatics*, *17*(7), 246.

Piñero, J., Queralt-Rosinach, N., Bravo, A., Deu-Pons, J., Bauer-Mehren, A., Baron, M., ... Furlong, L. I. (2015). DisGeNET: A Discovery Platform for the Dynamical Exploration of Human Diseases and their Genes. *Database*, *2015*.

Plank, B. (2011). Domain Adaptation for Parsing. Ph.D. Thesis. University of Groningen, Netherlands.

Plank, B. (2016). What to do About Non-Standard (or Non-Canonical) Language in NLP. In *Proceedings of the 13th Conference on Natural Language Processing (KONVENS).* Bochum, Germany.

Plank, B., & van Noord, G. (2011). Effective Measures of Domain Similarity for Parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics – Human Language Technologies (ACL-HLT)* (pp. 1566–1576). Portland, Oregon, USA: Association for Computational Linguistics.

Pyysalo, S., Ginter, F., Moen, H., Salakoski, T., & Ananiadou, S. (2013). Distributional Semantics Resources for Biomedical Text Processing. In *Proceedings of the 5th International Symposium on Languages in Biology and Medicine (LBM)* (pp. 39–44). Tokyo, Japan.

Pyysalo, S., Ohta, T., Miwa, M., Cho, H.-C., Tsujii, J., & Ananiadou, S. (2012). Event Extraction Across Multiple Levels of Biological Organization. *Bioinformatics*, *28*(18), i575–i581.

Pyysalo, S., Ohta, T., Rak, R., Sullivan, D., Mao, C., Wang, C., ... Ananiadou, S. (2011). Overview of the Infectious Diseases (ID) Task of BioNLP Shared Task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop* (pp. 26–35). Portland, Oregon, USA: Association for Computational Linguistics.

Pyysalo, S., Sætre, R., & Salakoski, T. (2008). Why Biomedical Relation Extraction Results are Incomparable and What to do About it. In *Proceedings of the 3rd International Symposium on Semantic Mining in Biomedicine (SMBM).* Turku, Finland.

Ramponi, A., Giampiccolo, S., Tomasoni, D., Priami, C., & Lombardo, R. (2020). High-Precision Biomedical Relation Extraction for Reducing Human Curation Efforts in Industrial Applications. *IEEE Access*, *8*, 150999-151011. doi: doi: 10.1109/ACCESS.2020.3014862 © 2020 IEEE.

Ramponi, A., & Plank, B. (2020). Neural Unsupervised Domain Adaptation in NLP—A Survey. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)* (pp. 6838–6855). Barcelona, Spain (Online): International Committee on Computational Linguistics.

Ramponi, A., Plank, B., & Lombardo, R. (2020). Cross-Domain Evaluation of Edge Detection for Biomedical Event Extraction. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)* (pp. 1982–1989). Marseille, France: European Language Resources Association.

Ramponi, A., van der Goot, R., Lombardo, R., & Plank, B. (2020). Biomedical Event Extraction as Sequence Labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 5357–5367). Online: Association for Computational Linguistics.

Rao, S., Marcu, D., Knight, K., & Daumé III, H. (2017). Biomedical Event Extraction Using Abstract Meaning Representation. In *Proceedings of*

BioNLP 2017 Workshop (pp. 126–135). Vancouver, Canada: Association for Computational Linguistics.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 779–788). Las Vegas, Nevada, USA.

Riedel, S., & McCallum, A. (2011). Robust Biomedical Event Extraction with Dual Decomposition and Minimal Domain Adaptation. In *Proceedings of BioNLP Shared Task 2011 Workshop* (pp. 46–50). Portland, Oregon, USA: Association for Computational Linguistics.

Riedel, S., McClosky, D., Surdeanu, M., McCallum, A., & Manning, C. D. (2011). Model Combination for Event Extraction in BioNLP 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop* (pp. 51–55). Portland, Oregon, USA: Association for Computational Linguistics.

Riedl, M. (2020). AI Democratization in the Era of GPT-3. *The Gradient*. (https://thegradient.pub/ai-democratization-in-the-era-of-gpt-3/)

Rosenblatt, F. (1958). The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, *65*(6), 386.

Ruder, S. (2016). An Overview of Gradient Descent Optimization Algorithms. *arXiv preprint arXiv:1609.04747*.

Ruder, S. (2017). An Overview of Multi-Task Learning in Deep Neural Networks. *arXiv preprint arXiv:1706.05098*.

Ruder, S. (2019). Neural Transfer Learning for Natural Language Processing. Ph.D. Thesis. National University of Ireland, Galway, Ireland.

Ruder, S., Peters, M. E., Swayamdipta, S., & Wolf, T. (2019). Transfer Learning in Natural Language Processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies: Tutorials (NAACL-HLT)* (pp. 15–18). Minneapolis, Minnesota, USA: Association for Computational Linguistics.

Saik, O. V., Ivanisenko, T. V., Demenkov, P. S., & Ivanisenko, V. A. (2016). Interactome of the Hepatitis C Virus: Literature Mining with ANDSystem. *Virus Research*, *218*, 40–48.

Schuster, M., & Nakajima, K. (2012). Japanese and Korean Voice Search. In *Proceedings of the 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 5149–5152). Kyoto, Japan.

Smith, N. A. (2020). Contextual Word Representations: Putting Words into Computers. *Communications of the ACM*, *63*(6), 66–74.

Spoustová, D., & Spousta, M. (2010). Dependency Parsing as a Sequence Labeling Task. *The Prague Bulletin of Mathematical Linguistics*, *94*(1), 7–14.

Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and Policy Considerations for Deep Learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (pp. 3645–3650). Florence, Italy: Association for Computational Linguistics.

Strzyz, M., Vilares, D., & Gómez-Rodríguez, C. (2019, June). Viable Dependency Parsing as Sequence Labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies: Volume 1, Long and Short Papers (NAACL-HLT)* (pp. 717–723). Minneapolis, Minnesota, USA: Association for Computational Linguistics.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)* (pp. 3104–3112). Montreal, Canada.

Szklarczyk, D., Morris, J. H., Cook, H., Kuhn, M., Wyder, S., Simonovic, M., ... others (2016). The STRING Database in 2017: Quality-Controlled Protein–Protein Association Networks, made Broadly Accessible. *Nucleic Acids Research*, *45*, D362–D368.

Tafti, A. P., Badger, J., LaRose, E., Shirzadi, E., Mahnke, A., Mayer, J., ... Peissig, P. (2017). Adverse Drug Event Discovery Using Biomedical Literature: A Big Data Neural Network Adventure. *JMIR Medical Informatics*, *5*(4), e51.

Tenney, I., Das, D., & Pavlick, E. (2019). BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 4593–4601). Florence, Italy: Association for Computational Linguistics.

Turing, A. M. (1950). Computing Machinery and Intelligence. *Mind*, *59*(236), 433–460.

van der Goot, R., Üstün, A., Ramponi, A., Sharaf, I., & Plank, B. (2021). Massive Choice, Ample Tasks (MaChAmp): A Toolkit for Multi-task Learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations (EACL)* (pp. 176–197). Online: Association for Computational Linguistics.

Vanegas, J. A., Matos, S., González, F., & Oliveira, J. L. (2015). An Overview of Biomolecular Event Extraction from Scientific Documents. *Computational and Mathematical Methods in Medicine*, *2015*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is All you Need. In *Advances in Neural Information Processing Systems (NeurIPS)* (pp. 5998–6008). Long Beach, California, USA: Curran Associates, Inc.

Venugopal, D., Chen, C., Gogate, V., & Ng, V. (2014). Relieving the Computational Bottleneck: Joint Inference for Event Extraction with High-Dimensional Features. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 831–843). Doha, Qatar: Association for Computational Linguistics.

Vilares, D., Abdou, M., & Søgaard, A. (2019). Better, Faster, Stronger Sequence Tagging Constituent Parsers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies: Volume 1, Long and Short Papers (NAACL-HLT)* (pp. 3372–3383). Minneapolis, Minnesota, USA: Association for Computational Linguistics.

Vlachos, A., & Craven, M. (2012). Biomedical Event Extraction from Abstracts and Full Papers Using Search-based Structured Prediction. *BMC Bioinformatics*, *13*(11), 1–11.

Walker, C., Strassel, S., Medero, J., & Maeda, K. (2006). *ACE 2005 Multilingual Training Corpus.* (https://catalog.ldc.upenn.edu/LDC2006T06)

Wang, A., Hula, J., Xia, P., Pappagari, R., McCoy, R. T., Patel, R., ... Bowman, S. R. (2019). Can You Tell Me How to Get Past Sesame Street?

Sentence-Level Pretraining Beyond Language Modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 4465–4476). Florence, Italy: Association for Computational Linguistics.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., ... Bowman, S. (2019). SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Advances in Neural Information Processing Systems (NeurIPS)* (Vol. 32, pp. 3266–3280). Vancouver, Canada: Curran Associates, Inc.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. (2018). GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP* (pp. 353–355). Brussels, Belgium: Association for Computational Linguistics.

Wang, X., McKendrick, I., Barrett, I., Dix, I., French, T., Tsujii, J., & Ananiadou, S. (2011). Automatic Extraction of Angiogenesis Bioprocess from Text. *Bioinformatics*, *27*(19), 2730–2737.

Warikoo, N., Chang, Y.-C., & Hsu, W.-L. (2018). LPTK: A Linguistic Pattern-aware Dependency Tree Kernel Approach for the BioCreative VI CHEMPROT Task. *Database*, *2018*.

Webber, B. (2009). Genre Distinctions for Discourse in the Penn TreeBank. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP)* (pp. 674–682). Suntec, Singapore: Association for Computational Linguistics.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., ... others (2016). Google's Neural Machine Translation System: Bridging the Gap Between Human and Machine Translation. *arXiv preprint arXiv:1609.08144*.

Yadav, S., Ekbal, A., Saha, S., Kumar, A., & Bhattacharyya, P. (2019). Feature Assisted Stacked Attentive Shortest Dependency Path based Bi-LSTM Model for Protein–Protein Interaction. *Knowledge-Based Systems*, *166*,

18–29.

Yu, K., Lung, P.-Y., Zhao, T., Zhao, P., Tseng, Y.-Y., & Zhang, J. (2018). Automatic Extraction of Protein-Protein Interactions using Grammatical Relationship Graph. *BMC Medical Informatics and Decision Making*, *18*(2), 42.

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

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

Zhang, Y., Lin, H., Yang, Z., Wang, J., Zhang, S., Sun, Y., & Yang, L. (2018). A Hybrid Model Based on Neural Networks for Biomedical Relation Extraction. *Journal of Biomedical Informatics*, *81*, 83–92.

Zhang, Y., & Weiss, D. (2016). Stack-Propagation: Improved Representation Learning for Syntax. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics: Volume 1, Long Papers (ACL)* (pp. 1557–1566). Berlin, Germany: Association for Computational Linguistics.

Zhao, Z., Yang, Z., Lin, H., Wang, J., & Gao, S. (2016). A Protein-Protein Interaction Extraction Approach Based on Deep Neural Network. *International Journal of Data Mining and Bioinformatics*, *15*(2), 145–164.

Zhou, J., & Fu, B.-q. (2018). The Research on Gene-Disease Association Based on Text-Mining of PubMed. *BMC Bioinformatics*, *19*(1), 37.

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., & Fidler, S. (2015). Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (pp. 19–27). Las Condes, Cile.

Appendix A

Part-of-Speech Tags

A list of the part-of-speech labels is provided in the following. Particularly, in Table A.1 the *coarse-grained* Universal POS tags (Nivre et al., 2016) are presented,[1] whereas in Table A.2 is the *fine-grained* PENN Treebank POS tag set (Marcus et al., 1993), OntoNotes 5 version.[2] For further information about the tags, their design methods and motivation, we refer the reader to the original research publications (Nivre et al., 2016; Marcus et al., 1993).

Table A.1: Universal POS tags.

Tag	Description	Tag	Description
ADJ	Adjective	PART	Particle
ADP	Adposition	PRON	Pronoun
ADV	Adverb	PROPN	Proper noun
AUX	Auxiliary	PUNCT	Punctuation
CCONJ	Coordinating conjunction	SCONJ	Subordinating conjunction
DET	Determiner	SYM	Symbol
INTJ	Interjection	VERB	Verb
NOUN	Noun	X	Other
NUM	Numeral		

[1] `https://universaldependencies.org/u/pos/`
[2] `https://catalog.ldc.upenn.edu/LDC2013T19`

Table A.2: Penn Treebank POS tags.

Tag	Description	Tag	Description
$	Symbol, currency	NNP	Noun, proper singular
“	Opening quotation mark	NNPS	Noun, proper plural
”	Closing quotation mark	NNS	Noun, plural
,	Punctuation mark, comma	PDT	Predeterminer
-LRB-	Left round bracket	POS	Possessive ending
-RRB-	Reft round bracket	PRP	Pronoun, personal
.	Punctuation mark, sentence closer	PRP$	Pronoun, possessive
:	Punctuation mark, colon or ellipsis	RB	Adverb
ADD	Email	RBR	Adverb, comparative
AFX	Affix	RBS	Adverb, superlative
CC	Coordinating conjunction	RP	Adverb, particle
CD	Cardinal number	SP	Space
DT	Determiner	SYM	Symbol
EX	Existential there	TO	Infinitival “to”
FW	Foreign word	UH	Interjection
GW	Add. word in multi-word expression	VB	Verb, base form
HYPH	Punctuation mark, hyphen	VBD	Verb, past tense
IN	Preposition / subordinating conjunction	VBG	Verb, gerund or present participle
JJ	Adjective	VBN	Verb, past participle
JJR	Adjective, comparative	VBP	Verb, non-3rd person singular present
JJS	Adjective, superlative	VBZ	Verb, 3rd person singular present
LS	List item marker	WDT	Wh-determiner
MD	Verb, modal auxiliary	WP	Wh-pronoun, personal
NFP	Superfluous punctuation	WP$	Wh-pronoun, possessive
NIL	Missing tag	WRB	Wh-adverb
NN	Noun, singular or mass	XX	Unknown

Appendix B

Grammatical Dependency Tags

A list of grammatical dependency tags (drawn from the ClearNLP tag set[1]) together with the linguistic descriptions are provided in Table B.1.

Table B.1: ClearNLP Dependency Labels.

Tag	Description
acl	Clausal modifier of noun
acomp	Adjectival complement
advcl	Adverbial clause modifier
advmod	Adverbial modifier
agent	Agent
amod	Adjectival modifier
appos	Appositional modifier
attr	Attribute
aux	Auxiliary
auxpass	Auxiliary, passive
case	Case marking
cc	Coordinating conjunction
ccomp	Clausal complement

(continued on the next page)

[1] `https://github.com/clir/clearnlp-guidelines/blob/master/md/specifications/dependency_labels.md`

(continued from the previous page)

Tag	Description
compound	Compound
conj	Conjunct
csubj	Clausal subject
csubjpass	Clausal subject, passive
dative	Dative
dep	Unclassified dependent
det	Determiner
dobj	Direct object
expl	Expletive
intj	Interjection
mark	Marker
meta	Meta modifier
neg	Negation modifier
nounmod	Modifier of nominal
npmod	Noun phrase as adverbial modifier
nsubj	Nominal subject
nsubjpass	Nominal subject, passive
nummod	Numeric modifier
parataxis	Parataxis
oprd	Object predicate
pcomp	Complement of preposition
pobj	Object of preposition
poss	Possession modifier
preconj	Pre-correlative conjunction
predet	Pre-determiner
prep	Prepositional modifier
prt	Particle
punct	Punctuation
quantmod	Modifier of quantifier
relcl	Relative clause modifier
root	Root
xcomp	Open clausal complement

Appendix C

Lexical Resources

The lexicons used for the study in Chapter 3 are provided in the following:

- **Purpose words**. {assay, characterize, conjugate, elucidate, investigate, measure, propose, publish, suggest, test};
- **Negation verbs**. {dissociate, unaffect};
- **Negation adverbs**. {almost, barely, diffusely, hardly, inadequately, infrequently, insignificantly, insufficiently, irregularly, marginally, meagerly, merely, minimally, narrowly, negatively, negligibly, neither, nowhere, rarely, scantily, scarcely, seldom, skimpily, slightly, sparely, sparsely, sporadically, uncommonly, unlikely, unusually, weakly};
- **Negation nouns**. {blockade, lack};
- **Negation adjectives**. {absent, different, independent, insignificant, irrelevant, negative, unaffected, unrelated}.

Appendix D

Corpora Details

This appendix provides additional information on the data collection and annotation process that has been followed for creating the corpora used throughout this thesis, as reported in the original publications. Relevant data statistics are instead presented in the main body when the corpora are introduced.

GE11 A standard corpus annotated for biomedical events used to measure the progress in the field, which has originated from a long-standing project effort (Collier et al., 1999; J.-D. Kim et al., 2008, 2011). The subject domain of GE11 (hereafter, GENIA) is about reactions concerning transcription factors in human blood cells. The corpus is the largest event extraction dataset to date and comprises both abstracts and full-texts. Data collection has been originally performed by querying the MEDLINE database[1] for `"Humans" [MeSH] AND "Blood Cells" [MeSH] AND "Transcription Factors" [MeSH]`,[2] then randomly sampling 2,000 abstracts for annotation (J.-D. Kim et al., 2008). GE11 abstracts derive from the effort described in J.-D. Kim et al. (2008), from which irrelevant annotations have been filtered out and new ones have been added to make the corpus more suitable for a shared task (J.-D. Kim et al., 2011). Additionally, fourteen full-text documents have also been included in GE11. Annotation has been performed by three graduates in molecular biol-

[1] `https://www.nlm.nih.gov/medline/medline_overview.html`

[2] MeSH refers to a controlled vocabulary thesaurus used for PubMed article indexing.

ogy, coordinated by a domain expert, and annotation quality has been ensured by frequent discussions and weekly meetings to inspect problematic annotation cases (J.-D. Kim et al., 2008). We were unable to find precise inter-annotator agreement scores for event annotations; however, the whole process has been coordinated by well-defined principles of annotations and very detailed guidelines – subsequently used as the basis for the annotation of the other event extraction datasets – accompanied by regular discussions and revisions (principles and guidelines can be found in J.-D. Kim et al. (2008)).

ID11 A corpus about mechanisms of infectious diseases in two-component regulatory systems, introduced in Pyysalo et al. (2011). Data has been collected from full-text publications within the PMC[3] open access subset. The selection has been performed by infectious disease experts from Virginia Tech as a representative sample of publications on two-component regulatory systems, published from 2006 onwards. Data annotation has been done by two teams (from the University of Tokyo and Virginia Tech, respectively), following annotation guidelines based on GENIA event annotations, which have been refined throughout the process. The annotation effort has been coordinated by an annotator with previous experience in event annotation. Based on the consistency of annotations produced by the two groups, the authors have estimated no lower than 75% F_1 score as inter-annotator consistency (Pyysalo et al., 2011) according to the standard evaluation criterion for event extraction described in Section 2.1.2, paragraph "Evaluation metrics".

EPI11 A corpus about epigenetic change events and post-translational modifications (Ohta et al., 2011). PubMed abstracts having `DNA methylation` or one of the prominent post-translational modifications identified in Ohta et al. (2010) (i.e., `glycosylation`, `hydroxylation`, `methylation`, `acetylation`) as MeSH term have been selected. Abstracts with fewer than five entities (identified with a named entity tagger) have been discarded, then the remaining abstracts have been manually filtered to ensure relevance to the intended topics. Data annotation has been performed by three annotators with a molecular

[3] `https://www.ncbi.nlm.nih.gov/pmc/`

biology background, following the GENIA event annotation guidelines. An event annotation expert has been in charge of supervising the entire process. An independent annotation has been performed on a random sample of 10% of the documents to ensure annotation consistency (each document has been annotated twice). Based on the standard evaluation criterion for event extraction described in Section 2.1.2, paragraph "Evaluation metrics", the agreement in terms of F_1 score has been reported to be 89% (Ohta et al., 2011).

PC13 A corpus of event-annotated documents relevant to pathway reactions, introduced in Ohta et al. (2013). Data selection has been performed based on the relevance of documents to specific pathway reactions. To the goal, a sample of literature references from the most annotated pathways in BioModels[4] has been collected (e.g., mTOR, yeast cell cycle), and manual filtering has been performed to a subset of abstracts concerning relevant reactions. Then, a random set of reactions from a selection of PantherDB[5] models has been used to retrieve documents relevant to those reactions by an automated system (Kemper et al., 2010). A manual check has then been performed to select the most relevant abstracts. Data annotation has been done by a team of three biologists, and a random sample of 20% of documents has been annotated by all annotators to discuss the most complex cases, refining the guidelines, and measuring the inter-annotator agreement. The inter-annotator agreement in terms of F_1 score, based on the standard evaluation criterion for event extraction described in Section 2.1.2, paragraph "Evaluation metrics", has been reported to be 61% after the initial annotation. For redundantly labeled documents, at the end of the annotation process a coordinator has evaluated and chosen the best documents to include in the final corpus (Ohta et al., 2013).

MLEE A corpus about angiogenesis (i.e., the development of new blood vessels from pre-existing ones) with event annotations, introduced in Pyysalo et al. (2012). This sub-domain concerns processes that are relevant to cancer and pathologies at the organism level. A corpus of 262 PubMed abstracts collected

[4] `https://www.ebi.ac.uk/biomodels/`
[5] `http://www.pantherdb.org/pathway/`

in a previous study (X. Wang et al., 2011) has been annotated for structured events. Data selection has been performed by randomly sampling MEDLINE abstracts from October 2009 onwards which match manually-crafted angiogenesis and event patterns (refer to X. Wang et al. (2011) for a full list). The annotation has been performed by a Ph.D. biologist with extensive experience in annotation and a Ph.D. computer scientist with experience in event extraction. Annotation guidelines have been based on the GENIA ones, and have been refined during the labeling process to ensure consistency for ambiguous cases. The inter-annotator agreement has been calculated in five rounds of annotations (ten abstracts each), followed by a revision to the annotations and the guidelines. The agreement has been calculated using F_1 score based on the standard evaluation criterion for event extraction described in Section 2.1.2, paragraph "Evaluation metrics". The agreement scores have been reported to be 56.8%, 71.6%, 70.1%, 76.1%, and 72.2%, showing that the disagreement has been reduced right after the first round of annotation (Pyysalo et al., 2012).

LLL A relation-annotated corpus that was created for the LLL05 (Learning Language in Logic) shared task (Nédellec, 2005), focused on gene transcription and sporulation of the model bacterium *Bacillus subtilis*. Abstracts have been selected by querying for `"Bacillus subtilis transcription"` on MEDLINE (Nédellec et al., 2001). Sentences containing at least two entities from a curated list of relevant genes and proteins have been retained (Nédellec et al., 2001; Nédellec, 2005). Annotation was performed by the MIG-INRA lab; however, we could not find mention on the inter-annotator agreement for the corpus.

IEPA A corpus annotated for relations, focused on interactions between a set of biochemicals (e.g., insulin, oxytoxin, leptin, etc.) (Ding et al., 2001). Data collection has been performed by querying MEDLINE through the PubMed interface using ten queries consisting of pairs of biochemical entities which represent diverse biological areas (e.g., `insulin AND oxytoxin`, `flavonoid AND cholesterol`; refer to Ding et al. (2001) for a full list). Sentences from titles and abstracts having at least two co-occurring biochemicals have been

retained. We were unable to find details on the manual annotation process. The F_1 score for information retrieval at the sentence level has been reported to be 72.9% (Ding et al., 2001).

HPRD50 A corpus about regulatory relations, direct physical interactions, and modifications, introduced in Fundel et al. (2007). Fifty PubMed abstracts referenced by the Human Protein Reference Database (Peri et al., 2004) have been randomly selected. Sentences from those abstracts have been annotated for relations by two annotators with a biochemical background, reaching an inter-annotator agreement[6] of 81%. As reported in Fundel et al. (2007), by considering one of the annotators as the ground truth this corresponds to 89% F_1 score for the other annotator.

[6]The intersection of relations that have been annotated over all the relations.