EXAMENSARBETE INOM MEDICINSK TEKNIK,
AVANCERAD NIVÅ, 30 HP

STOCKHOLM, SVERIGE 2020

Extracting Adverse Drug Reactions from Product Labels using Deep Learning and Natural Language processing

SHACHI BISTA

KTH
SKOLAN FÖR KEMI, BIOTEKNOLOGI OCH HÄLSA

Extracting Adverse Drug Reactions from Product Labels using Deep Learning and Natural Language Processing

SHACHI BISTA

Master in Medical Technology
Date: June 29, 2020
Supervisor: Dr. Lucie Gattepaille
Examiner: Dr. Matilda Larsson
School of Engineering Sciences in Chemistry, Biotechnology and Health
Host company: Uppsala Monitoring Centre
Swedish title: Detektering av läkemedels biverkningar i bipacksedel med hjälp av maskininlärning

Abstract

Pharamacovigilance relates to activities involving drug safety monitoring in the post-marketing phase of the drug development life-cycle. Despite rigorous trials and experiments that drugs undergo before they are available in the market, they can still cause previously unobserved side-effects (also known as adverse events) due to drug–drug interaction, genetic, physiological or demographic reasons. The Uppsala Monitoring Centre (UMC) is the custodian of the global reporting system, VigiBase, for adverse drug reactions in collaboration with the World Health Organization (WHO). VigiBase houses over 20 million case reports of suspected adverse drug reactions from all around the world. However, not all case reports that the UMC receives pertains to adverse reactions that are novel in the safety profile of the drugs. In fact, many of the reported reactions found in the database are known adverse events for the reported drugs. With more than 3 million potential associations between all possible drugs and all possible adverse events present in the database, identifying associations that are likely to represent previously unknown safety concerns requires powerful statistical methods and knowledge of the known safety profiles of the drugs. Therefore, there is a need for a knowledge base with mappings of drugs to their known adverse reactions. To-date, such a knowledge base does not exist.

The purpose of this thesis is to develop a deep-learning model that learns to extract adverse reactions from product labels — regulatory documents providing the current state of knowledge of the safety profile of a given product — and map them to a standardized terminology with high precision. To achieve this, I propose a two-phase algorithm, with a first scanning phase aimed at finding regions of the text representing adverse reactions, and a second mapping phase aiming at normalizing the detected text fragments into Medical Dictionary for Regulatory Activities (MedDRA) terms, the terminology used at the UMC to represent adverse reactions. A previous dictionary-based algorithm developed at the UMC achieved a scanning F_1 of 0.42 (0.31 precision, 0.66 recall) and mapping macro-averaged F_1 of 0.43 (0.39 macro-averaged precision, 0.64 macro-averaged recall). State-of-the-art methods achieve F_1 above 0.8 and above 0.7 for the scanning and mapping problems respectively.

To develop algorithms for adverse reaction extraction, I use the 2019 ADE Evaluation Challenge data, a dataset made by the FDA with 100 product labels annotated for adverse events and their mappings to MedDRA. This thesis explores three architectures for the scanning problem: 1) a Bidirectional Long Short-Term Memory (BiLSTM) encoder followed by a softmax classifier, 2) a BiLSTM encoder with Conditional Random Field (CRF) classifier and finally, 3) a BiLSTM encoder with CRF classifier with Embeddings from Language Model (ELMo) embeddings. For the mapping problem, I explore Information Retrieval techniques using the search engines whoosh and Solr, as well as a *Learning to Rank* algorithm.

The BiLSTM encoder with CRF gave the highest performance on finding the adverse events in the texts, with an F_1 of 0.67 (0.75 precision, 0.61 recall), representing a 0.06 absolute increase in F_1 over the simpler BiLSTM encoder with softmax. Using the ELMo embeddings was proven detrimental and lowered the F_1 to 0.62. Error analysis revealed the adopted Inside, Beginning, Outside (IOB2) labelling scheme to be poorly adapted for denoting discontinuous and compound spans while introducing ambiguity in the training data. Based on the gold standard annotated mappings, I also evaluated the whoosh and Solr search engines, with and without *Learning to Rank*. The best performing search engine on this data was Solr, with a macro-averaged F_1 of 0.49 compared to the macro-averaged F_1 of 0.47 for the whoosh search engine. Adding a *Learning to Rank* algorithm on top of each engine did not improve mapping performance, as both macro-averaged F_1 dropped by over 0.1 when using the re-ranking approach. Finally, the best performing scanning and mapping algorithms beat the aforementioned dictionary-based baseline F_1 by 0.25 in the scanning phase and 0.06 in the mapping phase. A large source of error for the Solr search engine came from tokenisation issues, which had a detrimental impact on the performance of the entire pipeline.

In conclusion, modern Natural Language Processing (NLP) techniques can significantly improve the performance of adverse event detection from free-form text compared to dictionary-based approaches, especially in cases where context is important.

Sammanfattning

Farmakovigilans berör de aktiviteter som förbättrar förståelsen av biverkningar av läkemedel. Trots de stränga prövningar som behövs för läkemedelsutvecklingen finns ändå en del biverkningar som är okända p.g.a. genetik, fysiologiska eller demografiska faktorer. Uppsala Monitoring Centre (UMC), i samarbete med World Health Organization (WHO) är vårdnadshavare till den globala databasen av rapporter på medicinska biverkningar, VigiBase. VigiBase innehåller över 20 miljoner misstänkta rapporter från hela världen. Dock, en andel av dessa rapporter beskriver biverkningar som är redan kända. Egentligen finns det över 3 miljoner potentiella samband mellan alla läkemedel och biverkningar i databasen. Att hitta den riktiga och okända biverkningar behövs kraftfulla statistiska metoder samt kunskap om det kända säkerhetsprofil av läkemedlet. Det finns ett behöv för ett databas som kartlägger läkemedel med alla kända biverkningar men, inget sådant databas finns idag.

Syftet med detta examensarbete är att utveckla en djup-lärandemodell som kan läsa av texter på läkemedels etiketter — tillsynsdokument som beskriver säkerhetsprofil av läkemedel — och kartlägga dem till ett standardiserat terminologi med hög precision. Problemet kan brytas in i två fas, den första *scanning* och den andra *mapping*. *Scanning* handlar om att kartlägga position av text-fragmentet i etiketter. *Mapping* handlar om att kartlägga de detekterade text-fragmentet till Medical Dictionary for Regulatory Activities (MedDRA), den terminologi som används i UMC för biverkningar. Tidigare försök, s.k. *dictionary-based approach* på UMC uppnådde *scanning* F_1 i 0,42 (0,31 *precision*; 0,64 *recall*) och *mapping macro-averaged* F_1 i 0,43 (0,39 *macro-averaged precision*; 0,64 *macro-averaged recall*). De bästa systemen (s.k. state-of-the-art) uppnådde *scanning* F_1 över 0,8 och 0,7 för den *scanning* respektive *mapping* problemet.

Jag använder den *2019 ADE Evaluation Challenge* dataset att utveckla algoritmerna i projektet. Detta dataset innehåller 100 läkemedels etiketter annoterad med biverkningar och deras kartläggning i MedDRA. Denna avhandling utforskar tre arkitekturer till *scanning* problemet: 1) Bidirectional Long Short-Term Memory (BiLSTM) och *softmax* för klassificering, 2) BiLSTM med Conditional Random Field (CRF) klassificering och, till sist, 3) BiLSTM med CRF klassificering och Embeddings from Language Model (ELMo) *embeddings*. Med avseende till *mapping* problematiken utforskar jag metoder inom *Information Retrieval* genom användning av sökmotorerna whoosh och Solr. För att förbättra prestandan i *mapping* utforskar jag *Learning to Rank* metoder.

BiLSTM med CRF presterade bäst inom *scanning* problematiken med F_1 i 0,67 (0,75 *precision*; 0,61 *recall*) som är ett 0,06 absolut ökning över den BiLSTM *encoder* med *softmax* klassificering. Med ELMo försämrade F_1 till 0,62. Analys av felet visade att Inside, Beginning, Outside (IOB2) märkning som jag har valt att använda passar inte till att beteckna diskontinuerliga och sammansatta *spans*, och tillför betydande osäkerhet i träningsdata.

Med avseende till *mapping* problematiken har jag kollat på sökmotorn Solr och whoosh, med, och utan *Learning to Rank*. Solr visade sig som den bäst presterande sökmotorn med *macro-averaged* F_1 i 0,49 jämfört med whoosh som visade *macro-averaged* F_1 i 0,47. *Learning to Rank* algoritmerna försämrade F_1 med över 0,1 för båda sökmotorer. Den bäst presterande *scanning* och *mapping* algoritmer slog den *baseline* systemets F_1 med 0,25 i *scanning* faset, och 0,06 i *mapping* fasen. Ett stor källa av fel för den Solr sökmotorn har kommit från tokeniserings-fel, som hade en försämringseffekt i prestanda genom hela pipelinen.

I slutsats, moderna Natural Language Processing (NLP) tekniker kan kraftigt öka prestanda inom detektering av biverkningar från etiketter och texter, jämfört med gamla *dictionary* metoder, särskilt när kontexten är viktigt.

Acknowledgements

I would like to express my gratitude to my supervisor Dr. Lucie Gattepaille for the opportunity and guidance, as well as being so patient with me with multiple reviews of the thesis and the learning process. I would also like to thank UMC for providing me a space to work and resources to complete my thesis.

I would also like to thank Dr. Chunliang Wang for reviewing my thesis and providing important feedback.

Shachi Bista
Stockholm, Sweden

Contents

Chapter 1

Introduction

Over the last few decades, the number of drugs and medicines manufactured, and approved for management of health-related issues has been steadily increasing [1]. At the same time, the clinical trial phase and approval time has increased for almost all classes of drugs [2], implying that the time-to-market for new drugs is significantly higher with more rigorous trial and approval phase.

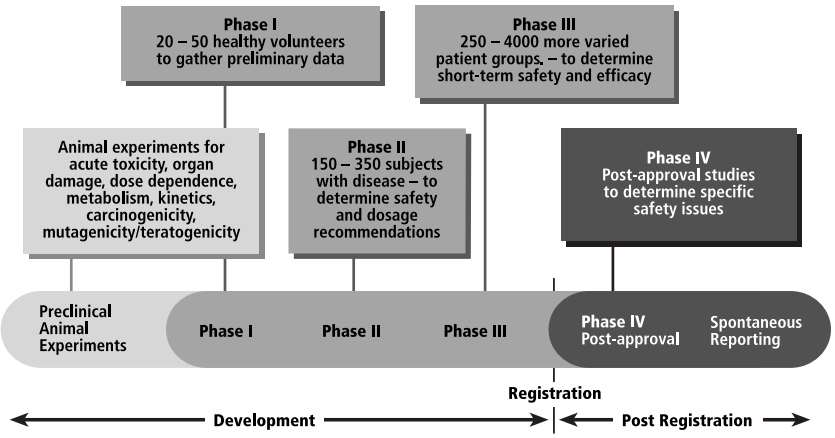

Figure 1.1: Phases for clinical development of medicines (Source: *Pharmacovigilance: Ensuring the Safe Use of Medicines* [3])

Ideally, side-effects (known as Adverse Event (AE) in the field of pharmacovigilance) caused by drugs are detected during the randomised control trial phases, however due to the number of participants and inclusion or exclusion criteria (such as demographics, age), combined with the trials being designed to test the efficacy of the drug in a *controlled* environment (e.g. untainted by

previous medical condition or drug–drug interaction) often fail to detect these unwanted effects in the pre-marketing stage [4]. These adverse events are not necessarily caused directly by the drugs themselves, but may be indirect effects of drug–drug interactions, medication errors or dosage issues[5].

This justifies the need for post-marketing surveillance programmes: the most prominent being spontaneous reporting. Spontaneous case reports can come from regulatory bodies, manufacturers, or if country regulations allow, directly from patients and healthcare professionals[6]. The Uppsala Monitoring Centre (UMC) maintains VigiBase, the global database of Individual Case Safety Reports (ICSRs) on behalf of the World Health Organization (WHO). As of early 2020, VigiBase contains over 20 million ICSRs which are reviewed on a case-by-case basis by experts in the National Centres of Pharmacovigilance from countries participating in the WHO Programme for International Drug Monitoring. In countries with mature pharmacovigilance systems, the sheer volume of reports makes the review of the cases a time– and resource–consuming process, if not unfeasible. Thus, there has been an increasing need for reliable statistical methods to detect patterns in the data that could be representative of a previously unknown safety concerns.

In VigiBase, there are more than 20,000 unique drugs and about as many different kinds of possible adverse events. Every possible drug–adverse event combination could be a signal of a novel safety concern to be communicated rapidly with all National Centres in the WHO Programme. However, with more than three million possible combinations, their manual review is unachievable. By comparing the observed number of reported cases to an expected number based on the drug and adverse event frequencies in the database, disproportionality analysis has demonstrated great value in deciding the priority of drug–adverse event combination for review, and a number of metrics have been proposed to facilitate the ranking of combinations[7, 8]. Among the top drug–adverse event combinations based on disproportionality, many of them describe known adverse events to the drugs[9]. While this is reassuring of the sensical nature of disproportionality analysis methods, it means that manual assessment of the top drug–adverse event combination will rarely lead to a signal of a novel safety concern. Having a reliable knowledge base linking every drug to all its known adverse events, based on the current knowledge of the safety of the drug, would greatly improve the ability to highlight the novel adverse events.

To date, determining whether a drug is known to cause a given adverse event is done manually by assessors, using product labels. Product labels are regulatory documents, usually managed by health authorities. Meant to be

read by healthcare professionals, product labels provide, in free text form the important information to be known about a given product (such as indication, dosage, posology, contra-indications) and in particular, the most definite and current reference for drug-related adverse reactions identified during the development phase and in the post-marketing phase (see fig. 1.1). In the United States (US), these documents, maintained by the Food and Drugs Adminstration (FDA), are provided by DailyMed[1]. Product Labels through DailyMed are publicly available for download as eXtended Markup Language (XML) files and are regularly updated as knowledge about drugs change. As a result, DailyMed provides a good basis for regulary applying text mining algorithms to extract the current state of knowledge on the safety of drugs approved for use in the US.

At UMC, a previous attempt has been made to build a knowledge base of known drug–adverse event combinations. The method employs a dictionary based algorithm to detect and match adverse events from various databases [10]. However, the algorithm can be too unspecific since it is matching items from a dictionary regardless of the context; it is also susceptible to missing adverse events that are not in the dictionary but may imply an adverse event due to context (e.g. *increase in blood pressure*, *elevation of liver associated enzymes*) or it may fail to identify that the phrase is referring to an exclusion criteria as in:

> *Fewer than 3% of adult patients without mycobacterial infections and fewer than 2% of pediatric patients without mycobacterial infections discontinued therapy because of drug-related side effects.*

In this phrase *myobacterial infections* is not an adverse event but is providing information about the drug trial, it is also negated by the term *without*. The lack of precision of this dictionary-based method made it unsuitable for automatic use in safety signal detection, as a false positive known drug–adverse event combination could mean wrongly discarding a combination that could in fact represent a true safety signal. Due to the complexities of natural language, a new approach is needed for reliably detecting adverse events from product labels that takes into account not just the words themselves but the context surrounding them. Recent developments in Natural Language Processing (NLP) augmented by deep-learning techniques now provide us with tools to perform such context-based classification.

The purpose of this project is to develop a deep-learning pipeline in order to read product labels, detect known adverse events from their text, map

[1] `https://dailymed.nlm.nih.gov/dailymed/`

the adverse event to a terminology and improve on the performance of the dictionary-based baseline algorithm. If demonstrated reliable, such a pipeline could then be applied regularly to DailyMed and extract the current knowledge of the safety of many drugs marketed in the US, in a format directly compatible with the analysis activities of the spontaneous reports database used by UMC. To develop and evaluate this pipeline, I used data from the ADE Evaluation Challenge (publication pending[2]), a Shared Task organized by FDA's Office of Surveillance and Epidemiology (OSE) based on manually annotated DailyMed data and designed to evaluate state-of-the-art (SOTA) systems on the extraction and normalization of adverse events from product labels.

1.1 Definitions

The terms Adverse Drug Reaction (ADR) and AE have very specific meanings within the field of Pharmacovigilance, here I define the terms that will be referred to throughout this document.

Adverse Drug Reaction (ADR) A response to a drug that is noxious and unintended and occurs at doses normally used in man for the prophylaxis, diagnosis or therapy of disease, or for modification of physiological function [11].

Adverse Event (AE) Any untoward medical occurrence in a patient or subject [in a clinical trial] administered a pharmaceutical product and which does not necessarily have a causal relationship with this treatment. An adverse event can therefore be any unfavourable and unintended sign (including an abnormal laboratory finding), symptom, or disease temporally associated with the use of a medicinal (investigational) product, whether or not related to the medicinal (investigational) product [12].

1.2 Problem Definition

The particular shared task challenge proposed by the FDA and of interest to UMC can be divided into two sub-tasks, hereby named as:

1. The Scan problem
2. The Map problem

[2] `https://sites.mitre.org/adeeval/`

These problems will be illustrated on the following phrase, taken from a product label description for the drug Belviq[3]:

Lorcaserin moderately elevates prolactin levels.

The Scan problem concerns the detection of ADRs; in the aforementioned example, the correct detection would be the sub-phrase *elevates prolactin levels*.

The Map problem relates to mapping the detected phrase *elevates prolactin levels* to Medical Dictionary for Regulatory Activities (MedDRA) which is the clinically-validated, standard terminology used by regulatory bodies and at the UMC. The detected phrase in the example above has two possible mappings in MedDRA:

Phrase	Mapped Term	Term ID
elevates prolactin levels	Blood prolactin increased	10005780
elevates prolactin levels	Hyperprolactinaemia	10020737

Where *Mapped Term* is the standard definition in the dictionary and *Term ID* is a unique identifier (across the dictionary) for the definition. A pairing of an ADR term and its mapping is called a *Mention*, in the example above, the tuple: *(elevates prolactin levels, Blood prolactin increased)* represents a single *Mention*.

[3] `https://dailymed.nlm.nih.gov/dailymed/drugInfo.cfm?setid=7cbbb12f-760d-487d-b789-ae2d52a3e01f`

Chapter 2

Methods

In order to develop and test the algorithms, I used the dataset and evaluation metrics provided by the ADE Evaluation Challenge. It is a Shared Task Challenge designed to evaluate state-of-the-art results for detection and normalization of ADR from product labels and is organized by the Office of Surveillance and Epidemiology (OSE) under FDA (see appendix A.4 for a more detailed description of the challenge).

I chose to pose the Scan problem as a Named Entity Recognition (NER) task with Inside, Beginning, Outside (IOB2) labels for classification [13]. For the Map problem I attempted to use Bidirectional Encoder Representations from Transformers (BERT) for re-ranking the normalized candidates from the search engines Solr[1] and whoosh[2].

2.1 Dataset

The dataset provided by the FDA contains 100 labelled and 2000 unlabelled Structured Product Label (SPL) documents. The FDA did not release annotations for the 2000 unlabelled documents which they use for internal evaluation. For development, the 100 annotated files are separated into: 80 training examples (fig. 2.1) and 20 test examples (fig. 2.2).

The annotated mentions in the labels are categorized into three classes:

OSE_Labelled_AE Category that the evaluation tools consider a valid ADR.

[1] `https://lucene.apache.org/solr/`

[2] `https://bitbucket.org/mchaput/whoosh/src/default/docs/source/intro.rst`

AFINITOR.xml	FOLOTYN.xml	LUMASON.xml	RITUXAN.xml
ANDROGEL.xml	GATTEX.xml	LUZU.xml	SABRIL.xml
ANORO.xml	GAZYVA.xml	LYNPARZA.xml	SAVELLA.xml
ARANESP.xml	GENOTROPIN_PRESERVATIVE_FREE.xml	MOVANTIK.xml	SENSIPAR.xml
ARCALYST.xml	HARVONI.xml	MYRBETRIQ.xml	SEROQUEL.xml
ASCLERA.xml	IMPAVIDO.xml	NATROBA.xml	SIGNIFOR.xml
AVASTIN.xml	INLYTA.xml	ONGLYZA.xml	SINGULAIR.xml
BELVIQ.xml	INVOKANA.xml	OPSUMIT.xml	STRIVERDI.xml
BENLYSTA.xml	ISTODAX.xml	PAXIL.xml	SURFAXIN.xml
BRILINTA.xml	JANUMET.xml	PLAVIX.xml	THALOMID.xml
CARBOPLATIN.xml	KADCYLA.xml	PREMARIN.xml	TOBI.xml
CELEBREX.xml	KERYDIN.xml	PREPOPIK.xml	TRADJENTA.xml
CYRAMZA.xml	KIT.xml	PROGRAF.xml	VICTOZA.xml
DAYTRANA.xml	KYPROLIS.xml	RAPIVAB.xml	VIIBRYD.xml
DEXAMETHASONE.xml	LASTACAFT.xml	RAXIBACUMAB.xml	VOTRIENT.xml
DOXIL_LIPOSOMAL_.xml	LATUDA.xml	RELISTOR.xml	VYVANSE.xml
EFFEXOR_XR.xml	LEVAQUIN.xml	REMERON.xml	XALKORI.xml
EFFIENT.xml	LEXAPRO.xml	REMICADE.xml	XELODA.xml
ENBREL.xml	LINZESS.xml	RENVELA.xml	XOFIGO.xml
ERBITUX.xml	LIPITOR.xml	RISPERDAL.xml	XTANDI.xml

Figure 2.1: Documents in the Training Set

AUBAGIO.xml	CYTOXAN.xml	HETLIOZ.xml	PENNSAID.xml	VIMPAT.xml
BELSOMRA.xml	EGRIFTA.xml	LIORESAL.xml	SIMPONI.xml	XARELTO.xml
BIAXIN.xml	ELLA.xml	LIVALO.xml	VALIUM.xml	XELJANZ.xml
CARAC.xml	FUROSEMIDE.xml	MEKINIST.xml	VIEKIRA.xml	ZALTRAP.xml

Figure 2.2: Documents in the Test Set

NonOSE_AE This is the category that may be an ADR but may result from unapproved use of the drug, occurs in the context of animal exposure, represents a sign/symptom of an OSE_Labelled_AE, or events resulting from a drug interaction. This category should not be detected.

Not_AE_Candidate Represent spans that appear to be ADR but are not; it can describe conditions unrelated to the ADR such as indications, contraindications or medical history. This category should not be detected.

Almost half (41%) of the annotated mentions should *not* be detected (Table table 2.1) by the system.

Category	Total Annotations	Disc. Annotations
OSE_Labelled_AE	36083 (58.63%)	2284 (3.71%)
NonOSE_AE	17787 (28.90%)	975 (1.58%)
Not_AE_Candidate	7678 (12.47%)	290 (0.47%)
	61548 (100%)	3549 (5.76%)

Table 2.1: Label statistics for the 100 annotated documents

2.2 Methods for the Scan Problem

In this section, I describe the different approaches I used on the Scan Problem, as well as the preprocessing steps. My experiments consisted of three approaches: a sequence-to-sequence model, a Bidirectional Long Short-Term Memory (BiLSTM) encoder architecture with a softmax classifier, a BiLSTM encoder with a Conditional Random Field (CRF) classifier and a BiLSTM encoder with CRF classifier and Embeddings from Language Model (ELMo) embeddings.

2.2.1 Preprocessing

For the Scan problem I performed very little preprocessing on the raw data. For every document in the training and validation set, I concatenated all the section text and passed it through the spaCy[3] tokeniser using the `en_core_web_sm` model. An intermediate representation is calculated for each token (algorithm 1) to handle out-of-vocabulary words and improve generalization. The original token text span index and length are recorded as metadata fields (since changing the token text changes the span start indices).

Algorithm 1: Token processor

Input: A spaCy segmented token
Result: A transformed representation of token
if *token is a drug name* **then**
  Result: `"< DRUG >"`
else if *token is a number* **then**
  Result: `"< NUMBER >"`
else if *token is one or more spaces* **then**
  Result: `"< SPACE >"`
else if *token is a linebreak* **then**
  Result: `"< BREAK >"`
else
  Result: input token

In order to tag the individual tokens according to the IOB2 scheme, I iterated through all of them and checked whether they fall within the span of a mention; if they do, I tagged them according to state transitions illustrated in fig. 2.3. The algorithm for generating these labels is listed in algorithm 2.

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

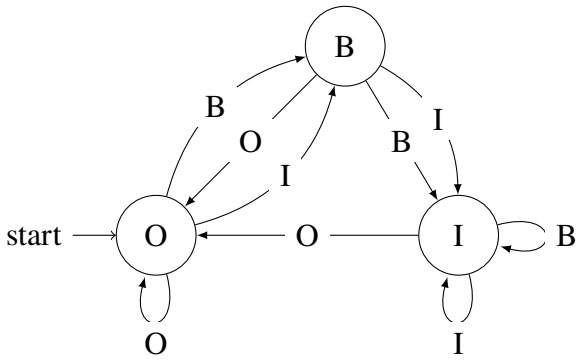

Figure 2.3: IOB2 state transitions

I also formulated an extended state transition table for the three mention categories table 2.2.

	O	B-PADR	I-PADR	B-NADR	I-NADR	B-NC	I-NC
O	O	B-ADR	B-ADR	B-NADR	B-NADR	B-NC	B-NC
B-PADR	O	I-ADR	I-ADR	B-NADR	O	B-NC	O
I-PADR	O	I-ADR	I-ADR	B-NADR	O	B-NC	O
B-NADR	O	B-ADR	O	I-NADR	I-NADR	B-NC	O
I-NADR	O	B-ADR	O	I-NADR	I-NADR	B-NC	O
B-NC	O	B-ADR	O	B-NADR	O	I-NC	I-NC
I-NC	O	B-ADR	O	B-NADR	O	I-NC	I-NC

Table 2.2: Transition table for extended labels

2.2.2 Embedding

I used pre-trained GloVe [14] and DailyMed token embeddings obtained as described below. The GloVe embeddings were downloaded from `http://nlp.stanford.edu/data/glove.6B.zip`.

I reused the DailyMed embeddings prepared by my supervisor Dr. Lucie Gattepaille. The embeddings were trained from the *Adverse event* and *warnings and precautions* sections of 24727 trade names from DailyMed using word2vec [15]. The embeddings have a dimension of 100 and were trained with a context size of 7, a minimum token count of 10 and downsampling parameter of 1×10^{-3} in the skip-gram mode. The library used to generate the embeddings was `gensim` and all other settings were kept at their default.

Algorithm 2: Procedure to generate IOB2 labels

Input: a list of tokens
Result: classifications

transitions ← (see fig. 2.3) ;

	O	B	I
O	O	B	B
B	O	I	I
I	O	I	I

classifications ← [];
last label ← O;
foreach *token in tokens* **do**
  current label ← O ;
  if *token is within a mention span* **then**
    if *first span* **then**
      current label ← B;
    else
      current label ← I;
  end
  current label ← transitions [last label][current label];
  last label ← current label
  classifications ← [classifications; current label];
end

Tokens that are not in the embedding are replaced with an `@@UNKNOWN@@` token.

2.2.3 Sequence to Sequence Model

My initial approach to the task was to use a sequence-to-sequence model and approach the task of AE detection as a Language Modelling problem. The input would be a list of tokens $w_0, w_1, \ldots, w_n$ and the outputs would be a sequence of elements from the two classes $c \in \{\text{O}, \text{ADR}\}$.

A sequence-to-sequence model consists of two pairs of BiLSTM layers, called encoder and decoder, respectively. The encoder hidden state $\mathbf{s}_0$ is initialised with zeros and accepts an embedded sequence $\{w_0, w_1, \cdots, w_n\} \mapsto \{\mathbf{e}_0, \mathbf{e}_1, \cdots, \mathbf{e}_n\} \in \mathbb{R}^d$, where d is the embedding dimension and passes the embedded feature vectors through a BiLSTM layer. The final hidden state of the encoder is called the *context vector*.

The decoder is another BiLSTM layer similar to the encoder but the initial hidden state $\mathbf{h}_0$ for the decoder is the context vector from the encoding layer

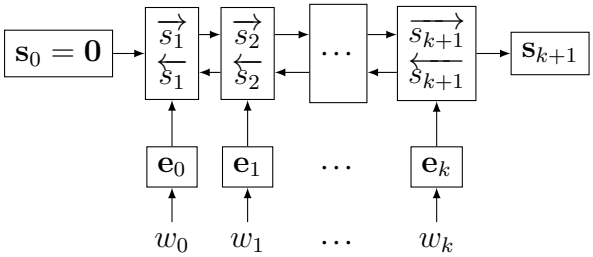

Figure 2.4: Schematic diagram of an encoder

$\mathbf{s}_{k+1}$ (fig. 2.5). Additionally, the input features to the decoder is the output from the previously sampled output of the decoder. Both the encoder and decoder introduce special `START` and `STOP` tokens to seed the decoding process and as a stop condition for decoding (since the length of the output sequence can be different than the length of the input sequence).

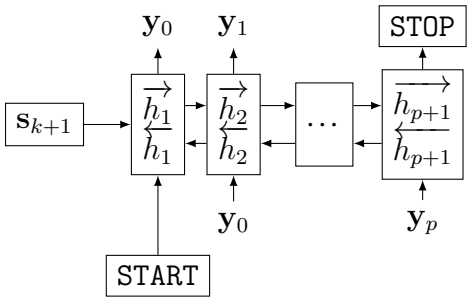

Figure 2.5: Schematic diagram of a decoder

The justification for attempting a sequence-to-sequence model was that, for every contextual sequence (e.g. a sentence), it would encode the context into a summary vector which could then be used to identify context-sensitive features for classifying ADR. The model would have information about all the tokens in the sentence before trying to find ADRs in the text. This approach has been applied well in the context of machine translation and augmented with special *attention layers* [16] which learn to "focus" on different parts of the text when decoding.

2.2.4 BiLSTM with Softmax

BiLSTM with the softmax classifier is a simple neural model for NER. Similar to the sequence-to-sequence model, it consists of an encoder that accepts word embeddings and outputs a concatenated bi-directional feature vector, unlike

Algorithm 3: Seq2Seq training loop

```
foreach epoch do
    foreach batch do
        last output = <START> ;
        foreach example do
            s_0 ← 0 ;
            contexts ← [] ;
            foreach embedded token do
                contexts ← [contexts; Encoder(embedded token,
                  s_0)] ;
            end
            last output ← <START> ;
            while last output is not <STOP> do
                y ← [y; Decoder(last output, contexts)] ;
            end
        end
    end
end
```

the sequence-to-sequence model the BiLSTM hidden states are ignored and its output features are passed through a fully connected layer followed by softmax. This is illustrated in fig. 2.6 where $\mathbf{f}_i = \mathbf{w}_i[\overrightarrow{s}_{i-1}; \overleftarrow{s}_{i-1}] + \mathbf{b}$ and σ is the softmax function.

One of the major issues with training BiLSTMs is the segmentation boundaries of the input text. Since BiLSTMs form a context vector by concatenating the state vectors from both directions in the text, with bad segmentation the BiLSTM only observes part of the context before the hidden state is reset for the next training example. To prevent this, I use spaCy sentence tokeniser to segment the section texts into smaller "sentence" segments.

In order to improve training speed the sentence segments are batched into mini-batches $b_m \in \mathbb{Z}^{m \times l}$, where m is the size of the mini-batch and l is the length of the longest sentence in the mini-batch and the integers in the b_m tensor are indices of the word vector in the embedding. Since sentences don't fit neatly into a rectangular matrix (due to varying lengths), sentences shorter than l are padded to the right with a special padding token (indexed as 0 in the word embedding). Passing this tensor into the word embedding returns a tensor of size $\mathbf{e} \in \mathbb{R}^{m \times l \times d}$ where d is the word embedding dimension.

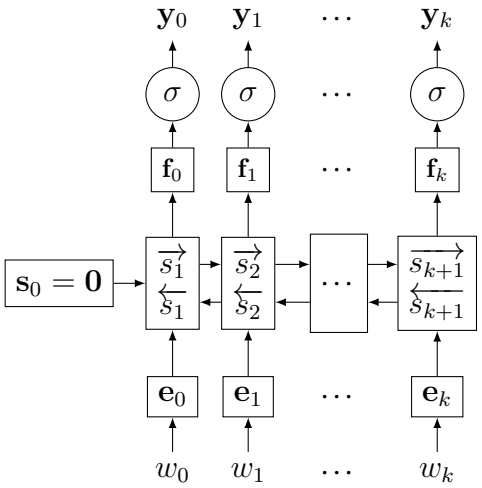

Figure 2.6: Simplest BiLSTM with softmax architecture

Special care must be taken when passing the embedded tensor into the BiLSTM: since we introduced padding tokens at the end of each mini-batch for sentences $< l$, we need to introduce a *mask* to the BiLSTM that indicates sentence boundaries. Without this, the backward Long Short-Term Memory (LSTM) will read zeros from the right until the sentence boundary. The output of the BiLSTM layer is of the dimension $\mathbf{s} \in \mathbb{R}^{m \times l \times 2r}$ where r is the size of the hidden state of the BiLSTM layer (these layers can be stacked), the factor of two arises from the concatenation operation of the forward and backward internal states (eq. (A.45)). A dropout layer that randomly zeros some cells of the hidden state tensor follows the BiLSTM stacks as a regularisation factor.

Finally, the fully connected layer reduces the dimension of the hidden states into the three classes $c \in \{\mathrm{O}, \mathrm{B}, \mathrm{I}\}$ resulting in tensors of size $\mathbf{f} \in \mathbb{R}^{m \times l \times 3}$. The output of the fully-connected layer are logits which have a large range $[-\infty, \infty]$, I use the softmax function to compress this range into $[0, 1]$. The result tensor is of size $\mathbf{y} \in \mathbb{R}^{m \times l \times 3}$.

$$\mathbf{y}_{ij} = \begin{bmatrix} p(\mathrm{O}|\mathbf{f}_i j) \\ p(\mathrm{B}|\mathbf{f}_i j) \\ p(\mathrm{I}|\mathbf{f}_i j) \end{bmatrix} \tag{2.1}$$

The predicted class is derived by performing an argmax over the vector $\mathbf{y}_i$.

All experiments were trained using the Adam optimizer with a learning rate of 3×10^{-3} for 30 epochs and a weight decay of 1×10^{-4}. When stacking the BiLSTM layers, I use a inter-layer dropout factor of 0.6 for each layer. Each mini-batches are sorted by the largest sequence length in the mini-batch.

2.2.5 BiLSTM with CRF

As mentioned earlier, the softmax classifier does not take into account labels that come before or after the current time-step. When using a sequential tagging scheme such as IOB2 the softmax classifier does not enforce constraints on the predicted labels. As a result a possible prediction might be O–O–I–B, which should not be possible. The idea with the CRF is that it would enforce transition probabilities from the previous estimations.

The BiLSTM–CRF architecture is very similar to the BiLSTM–Softmax architecture. The only difference between the two is the final classifier layer.

Since the output of the tokenisation process results in all tokens being lowercased, it is possible to lose abbreviations such as "ALT" in "ALT levels increases". In order to mitigate this the 100-dimensional DailyMed embeddings are concatenated with a bidirectional character RNN (charRNN) feature. The charRNN output dimension is 80, there are two layers with a 0.25 dropout; the final embedding dimension is $100 + 2 \times 80 = 260$.

The bidirectional state from the BiLSTM stack $\mathbf{s} \in \mathbb{R}^{m \times l \times 2r}$ is reduced by the fully connected layer which form the emission part of the CRF feature function, the CRF layer helps constrain the hidden states by predicting the most likely path given the observations using Viterbi decoding [17].

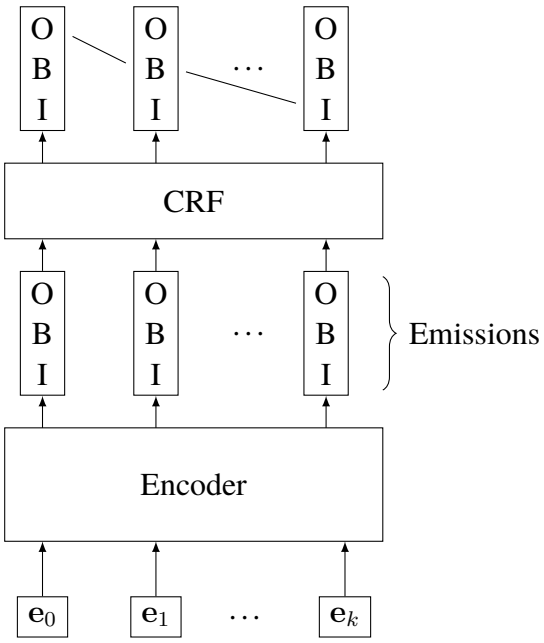

Figure 2.7: Schematic model of BiLSTM–CRF model

The BiLSTM–CRF model is also trained with the Adam optimizer with a learning rate of 1×10^{-3} for 10 epochs.

2.2.6 BiLSTM with CRF and ELMo embeddings

Finally, the BiLSTM–CRF architecture with ELMo embeddings is similar to the plain BiLSTM–CRF architecture. The only difference between these two architectures is the choice of the embedding layer. The DailyMed embeddings with charRNN features, are replaced a concatenation of DailyMed embeddings with ELMo embeddings. ELMo embeddings have a dimension of 1024 resulting in the final embedding dimension of $1024 + 100 = 1124$.

Terms in the product labels are intrinsically contextual, for e.g. the term *reactions* in the phrase *To report SUSPECTED ADVERSE REACTIONS, contact ...* is clearly not an adverse event, however the same term in the phrase *application site reaction* represents an adverse event. Using embeddings that do not take the context into account, both terms would be encoded into the same vector representation. Due to the contextual nature of ELMo, which is trained as a language model, the term *reactions* will be encoded into different vector representations in the above examples, which should help the system further differentiate between ADRs and non-ADRs phrases.

This architecture is trained with the same parameters as the BiLSTM–CRF architecture.

2.3 Methods for the Map Problem

This section describes the approach used for the Map Problem. I attempted to use a search engine followed by BERT based *Learning to Rank* algorithm to score the results. Some data preparation is necessary which is described in section 2.3.1.

2.3.1 Preprocessing

For the Map problem, I used a pre-indexed search index using MedDRA v20.0. Each document in the index contains a MedDRA Preferred Term (PT) code, PT term name and Lowest Level Term (LLT) name (see appendix A.2 for a more detailed description of the terminology and its hierarchical levels).

To train the ranking model with BERT, I extracted all mention texts (i.e. text fragments encoded by the spans) in the Gold Standard and their associated PT and LLT codes, as well as the human-readable names associated to the PT

codes: $(\mathbf{q}, \mathbf{pt}_{\text{name}}, \mathbf{pt}_{\text{code}}, \mathbf{llt}_{\text{code}})$. I combined this data with the MedNorm dataset [18] and created a mapping from free-text phrases to their PT or LLT terms. Then, for each query term $q \in \mathbf{q}$ in this mapping I perform a search against the indexed search engine for at most 10 documents. For each returned result, I calculated the relevance by checking the PT or LLT code and created a tuple $(q, m_i, r_i); i \in [1, 10]$ where m_i is the i-th search engine result (hereby called concept) and r_i is the relevance for the i-th concept, which is 1 if the PT or LLT code for the concept and query match, 0 otherwise. This query–result pair is then passed to a pre-trained BERT classifier with the relevance acting as a label.

2.3.2 Learning to Rank

The learning to rank model requires a pre-indexed search index which was adapted as described in section 2.3.1. The previously prepared query to concept mapping is then used to train a BERT classifier. The query and concept are concatenated with the special BERT tokens `[CLS] query [SEP] concept` for each concept query pair (q, m_i), the output of the BERT classifier is a softmax score which is interpreted as $p(r = 1|q, m_i)$ and can be used to re-rank the results from the search index. This model is adapted from [19] and illustrated in fig. 2.8.

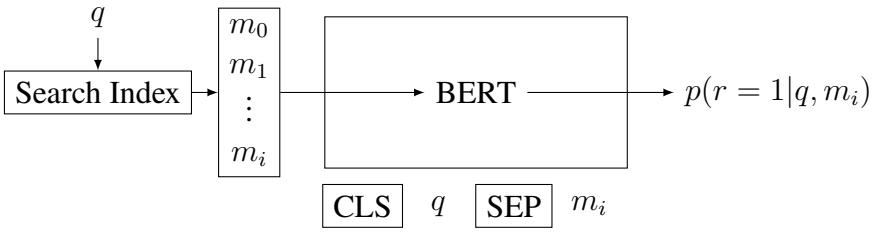

Figure 2.8: BERT based re-ranking model

2.4 Evaluation

Evaluation is performed in two phases: for each of the Scan and Map problems, during training, local evaluation metrics are used (see fig. 2.9). The full pipeline is then integrated and tested using System Evaluation metrics provided by the FDA which they use internally for evaluating the results of submissions.

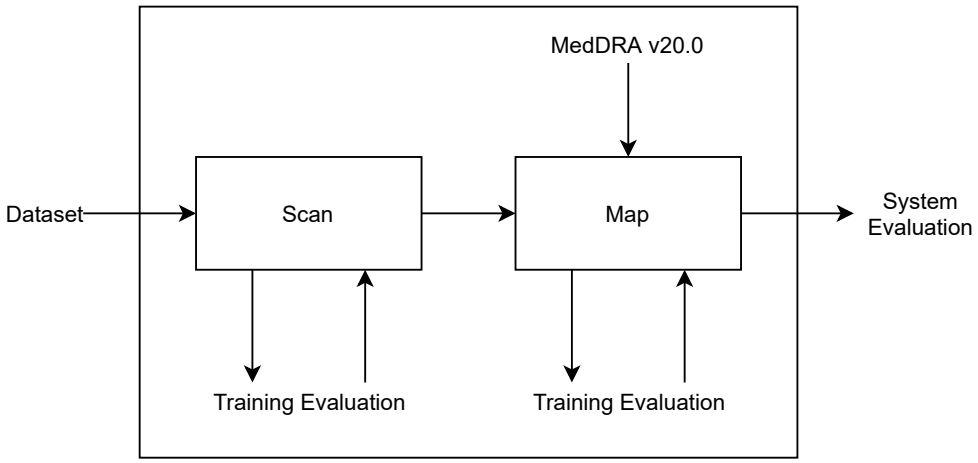

Figure 2.9: Overview of pipeline and evaluation checkpoints

2.4.1 Training Evaluation for Scan problem

During training of the Scan phase, I use the per-token label classification loss as the evaluation metric. Each token is labelled using the IOB2 labelling scheme.

This only evaluates whether the Scan system is able to find relevant tokens in the text. I select the model with the highest classification F_1 . Training performance is measured by performing a grid-search over parameter space and performing k-fold cross validation ($k = 5$).

2.4.2 Training Evaluation for Map problem

Training for the Map problem only concerns the Learning To Rank system, since the core system for the Map problem is a search engine and requires no training. For the Learning To Rank system, the training evaluation metric is the label classification loss, the labels represent the relevance of the search results (see section 2.3.2).

2.4.3 System Evaluation

The FDA provides a set of evaluation tools which they use internally for evaluating the results of submissions. During testing, I use the provided FDA evaluation tools to compare against state-of-the-art results and compare it to the baseline results.

For the Scan problem, the evaluation performance is defined by the Exact Mention Match metric, which performs a weighted scoring of the overlap between the detected character offsets (called *spans*, characterized by a start offset and length) and the spans in the gold label combined with the MedDRA match. Overlap is calculated by counting the number of characters that overlap between the gold and detected spans divided by the length of the detected span.

The *unweighted Exact Mention Match* metric assigns 80% of its weight to the span overlap and 20% to the MedDRA match (which is 1 if there is a match, 0 otherwise).

The *weighted Exact Mention Match* performs some FDA specific weighting determined by how difficult it is to add or remove a *Mention* by a human-in-the-loop (see appendix A.4.4 for more information on how the weights are defined). This is the primary metric used to compare the results of the Scan problem in this thesis to the state-of-the-art results[4]. The weights are defined in terms of four groups [20]:

exact match mention pairs, which are pairs of gold and submission mentions where the spans of the gold and submission mentions are identical, and the submission MedDRA PT code is among the gold mention MedDRA PT codes

inexact match mention pairs, which are pairs of gold and submission mentions which overlap in span, but which either don't match exactly in span, don't match in MedDRA PT code, or both

missing gold mentions, which have no submission counterpart

spurious submission mentions, which have no gold counterpart

For the Map problem, during testing, I use the *MedDRA coding* metric macro-averaged along the sections of the product labels, which defines the Precision/Recall/ F_1 for mapping the detected spans to their MedDRA entries.

As stated earlier in this chapter, the FDA has not released the 2000 test labels that they use for internal evaluation. The state-of-the-art results in their report [20] are tested on these 2000 labels. As a result, the results of this work are not directly comparable to the state-of-the-art results.

[4]This is mainly because the *unweighted Exact Mention Match* metrics are not reported for the state-of-the-art results.

Instead, in order to evaluate the results of this thesis directly, I compare with three previous attempts at UMC (see appendix A.8 for details) whose results are derived from the same test set as this project:

Baseline This is the baseline dictionary-based approach that the UMC currently uses, but it suffers from low precision due to its unspecific matching of ADR

Submission 1 This is an ensemble method developed by my supervisor Dr. Lucie Gattepaille; it combines the dictionary-based approach (see Baseline) with a BiLSTM–softmax classifier for the Scan problem.

Submission 2 This model is similar to Submission 1, but instead uses whoosh to perform the Map problem.

Chapter 3

Results

For the purpose of the UMC, not all metrics defined by the FDA are important. UMC is mostly interested in the *unweighted Exact mention match* metrics and MedDRA retrieval statistics. However the FDA defines the primary back-office metric to be the *weighted Exact mention match*. I have chosen to ignore the unweighted "Exact match (discontinuous)" metric because none of the models achieved any matches in the discontinuous case, resulting in the scores being 0 for all models.

Using the BERT-based ranking model resulted in MedDRA matches which were less performant than using traditional Information Retrieval (IR) techniques (table 3.4); as a result, I chose not to use it for evaluating the models. The MedDRA retrieval figures in table 3.3 are the results without re-ranking.

3.1 Parameter Selection

Parameter selection is done using a grid-search on the encoder hidden size $H \in \{8, 100, 300, 1024\}$ and number of LSTM layers $N \in \{1, 3, 5\}$. Batch size during training is 16 for all models.

3.1.1 BiLSTM with Softmax

The best validation F_1 of 0.752 in the BiLSTM–Softmax model was achieved with a hidden size $H = 100$ and with a single BiLSTM layer (fig. 3.1 and `id:fd2ff31c` in table B.1). $H = 1024$ is omitted because it was unable to complete due to memory issues on the GPU device.

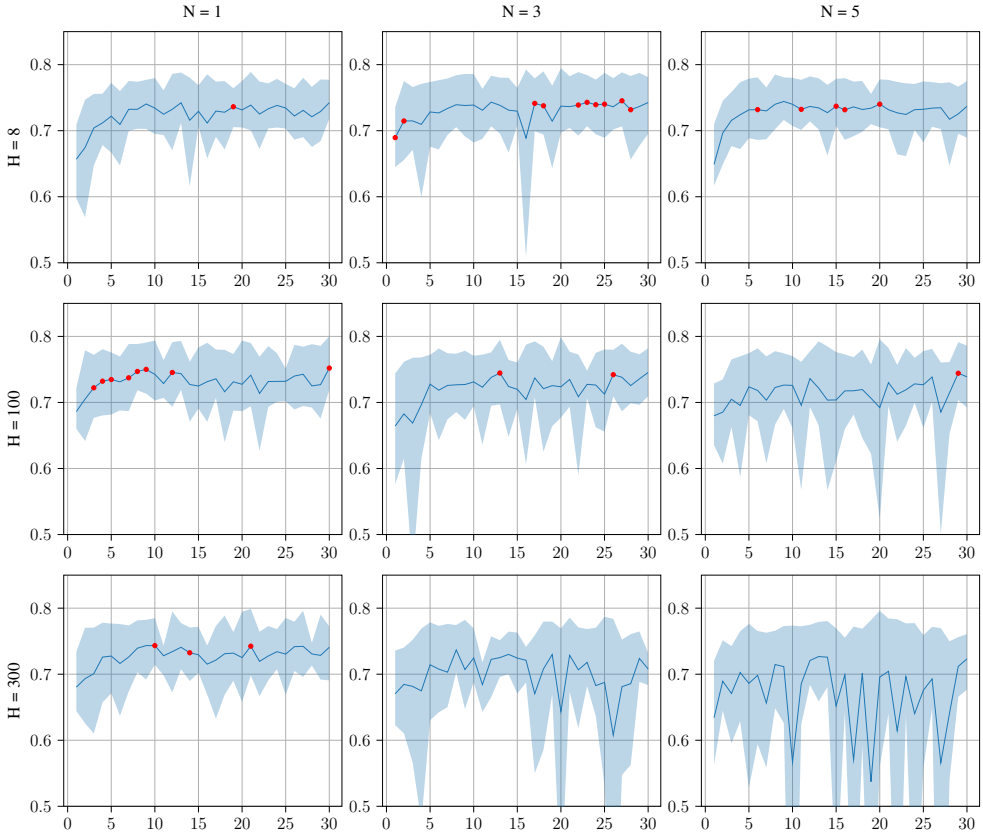

Figure 3.1: Validation F_1 (y-axis) per epoch (x-axis) across hidden size (by row) and number of layers (by column) of the BiLSTM–Softmax model. The shaded area represents the maximum and minimum F_1 per epoch across the 5-folds. The solid line represents the average F_1 across all folds. Red dots mark the run with the highest F_1 per epoch.

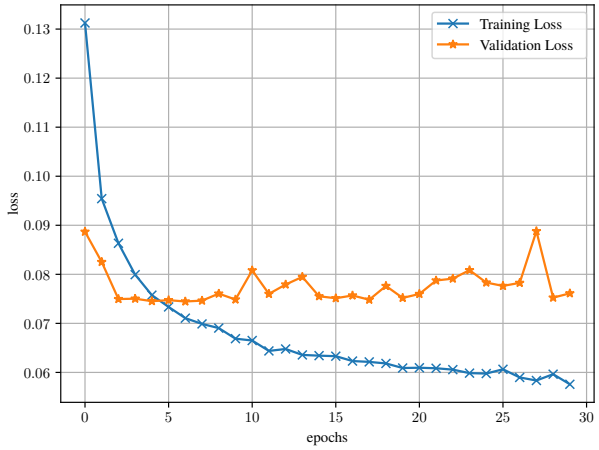

Figure 3.2: Training and validation loss for the best BiLSTM–Softmax model

3.1.2 BiLSTM with CRF

The best validation F_1 of 0.795 was achieved with a hidden size $H = 1024$ and a single BiLSTM layer (fig. 3.3 and `id:39c120be` in table B.1).

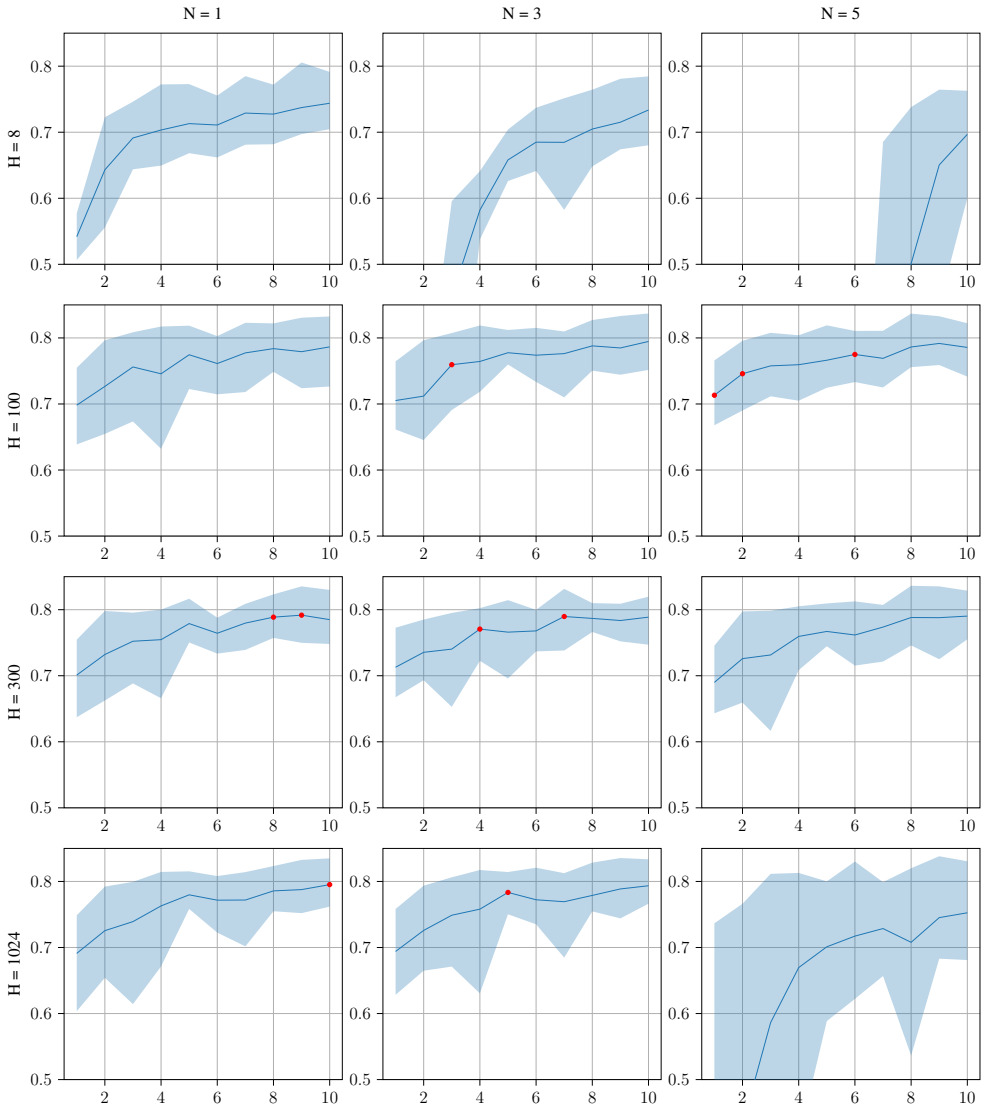

Figure 3.3: Validation F_1 (y-axis) per epoch (x-axis) across hidden size (by row) and number of layers (by column) of the BiLSTM–CRF model. The shaded area represents the maximum and minimum F_1 per epoch across the 5-folds. The solid line represents the average F_1 across all folds. Red dots mark the run with the highest F_1 per epoch.

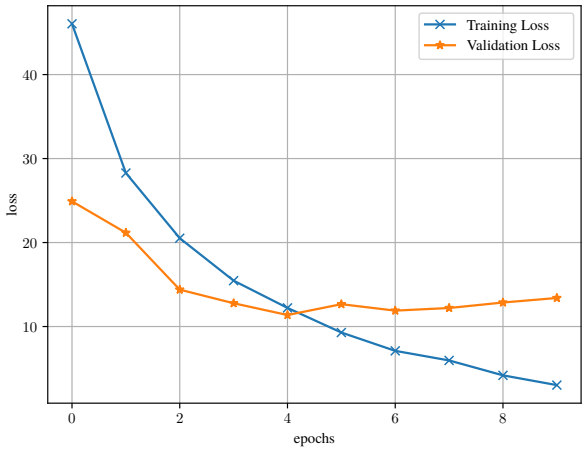

Figure 3.4: Training and validation loss for the best BiLSTM–CRF model

3.1.3 BiLSTM with CRF and ELMo

The best validation F_1 of 0.768 was achieved with $H = 1024$ and $N = 5$ (fig. 3.6 and `id:25e88b28` in table B.1).

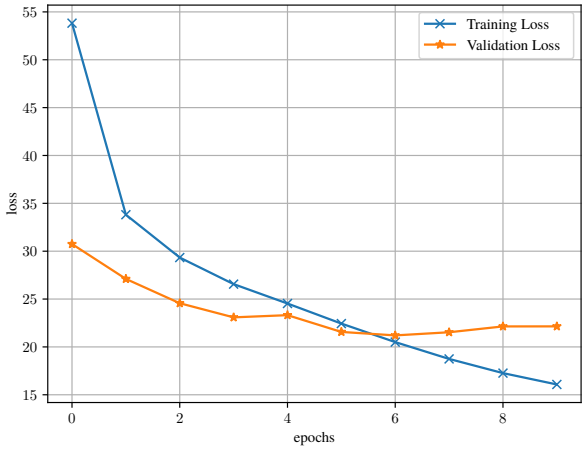

Figure 3.5: Training and validation loss for the best BiLSTM–CRF–ELMo model

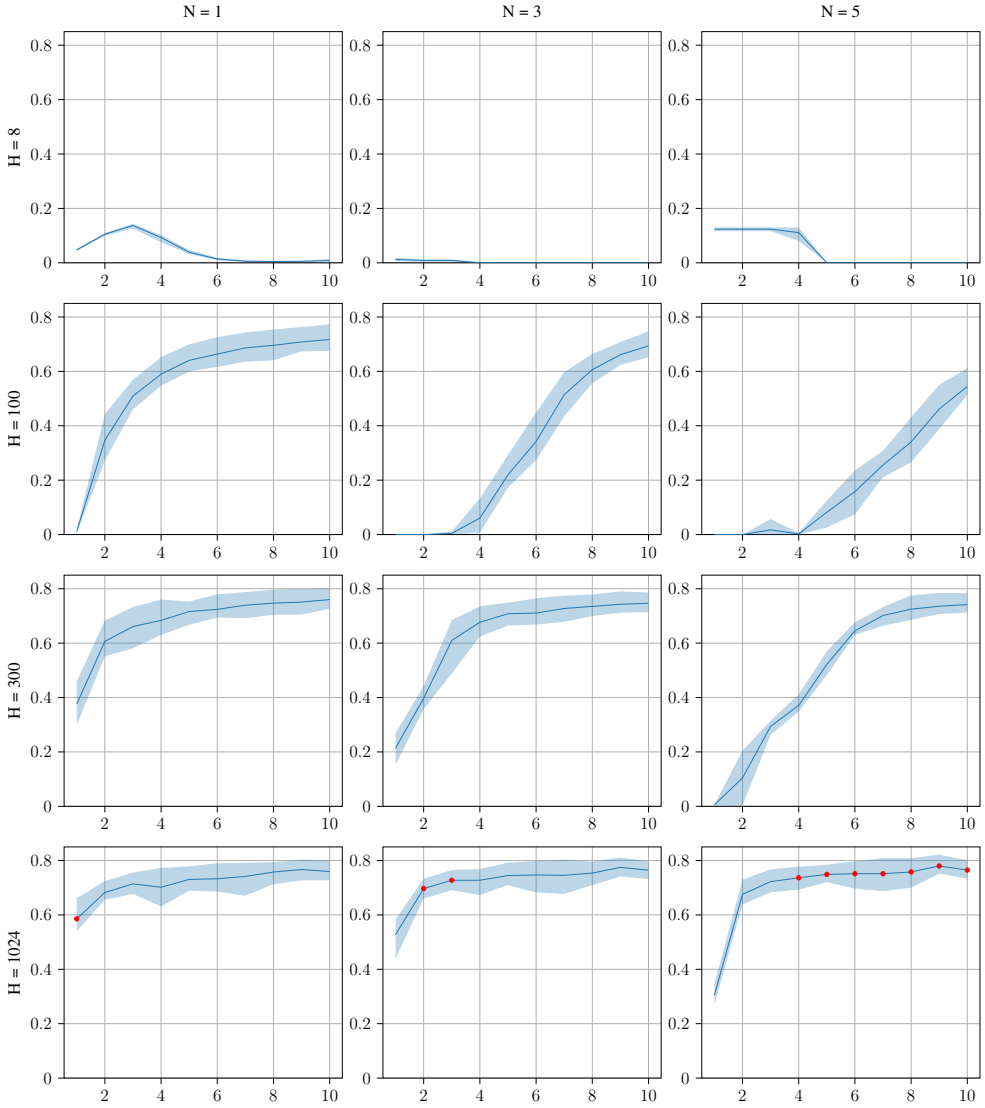

Figure 3.6: Validation F_1 (y-axis) per epoch (x-axis) across hidden size (by row) and number of layers (by column) of the BiLSTM–CRF model with ELMo embeddings. The shaded area represents the maximum and minimum F_1 per epoch across the 5-folds. The solid line represents the average F_1 across all folds. Red dots mark the run with the highest F_1 per epoch.

3.2 Selection of Embedding

For all three models, the DailyMed embeddings outperformed GloVe embeddings. However the differences are marginal and the choice of embedding does not seem to be the deciding factor in the system performance.

	BiLSTM+Softmax	BiLSTM+CRF	BiLSTM+ELMo+CRF
GloVE	0.743	0.775	0.765
DailyMed	0.752	0.795	0.768
Δ	0.009	0.020	0.003
$\Delta\%$	1.211	2.581	0.391

Table 3.1: Change in Validation F_1 with the two different embeddings

3.2.1 Effect of expanding the label space

As stated earlier, the annotated mentions fall into three distinct categories: OSE_Labeled_AE, NonOSE_AE and Not_AE_Candidate. For all the labelling tasks, we only consider the OSE_Labeled_AE category and ignore the others, but it seems to be a waste to ignore almost half of the annotations when trying to detect spans. In order to make use of most of the annotated data, I introduced extra classes to the best CRF classifier, namely B-, and I- tags for the three classes PADR (possible ADR), NADR (not ADR) and NC (not candidate).

	Exact match†			Exact match*					
	P	R	F_1	P	R	F_1	$C^\dagger/C^\star$	$M^\dagger/M^\star$	$S^\dagger/S^\star$
Limited	0.88	0.66	0.76	0.72	0.59	0.65	185/370	648/648	47/187
Extended	0.73	0.79	0.76	0.42	0.68	0.52	268/537	251/251	461/1843

†weighted
*unweighted

Table 3.2: Results of using the limited annotations and extended annotations with the best BiLSTM–CRF classifier

Table 3.2 shows the results of the expanding the label space, here C, M and S are the *inexact match*, *exact match* and *spurious* metrics, respectively. We can see precision drops significantly for both *weighted* and *unweighted Exact*

Mention Match metric. There is an increase in spurious matches (detected spans which are not in the gold labels) and a substantial drop in exact matches.

3.3 Scan Performance

Figure 3.7 presents the precision/recall for the *weighted Exact Mention Match* metric. The top state-of-the-art (SOTA) results achieve a F_1 of 0.89 (0.93 precision, 0.85 recall), which is significantly higher than our results. Submission 2 is the best performing system with an F_1 of 0.78 (0.87 precision, 0.70 recall), followed by the BiLSTM–CRF architecture with an F_1 of 0.76 (0.88 precision, 0.66 recall) marginally outperforming Submission 1 in the recall metric. All systems presented in this project and Submission 1 and 2 achieve significantly higher precision, compared to the baseline dictionary-based approach. The baseline method outperforms our approaches in the recall metric, which is expected as the systems presented and Submission 1 and 2 are more selective in their detection.

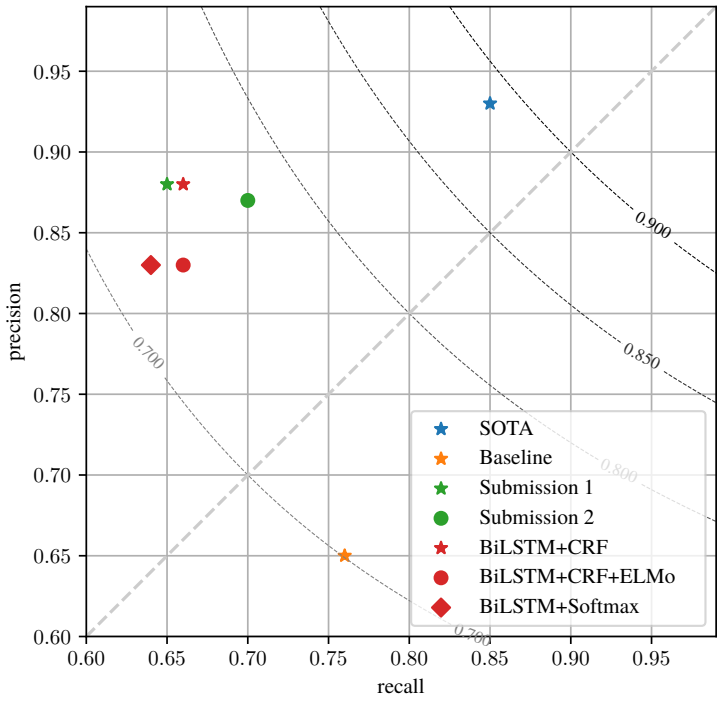

Figure 3.7: *weighted Exact Mention Match* for different architectures, curved lines show F_1 iso-curves, diagonal line denotes precision = recall

NB: The axes have been limited to $F_1 \geq 0.60$ to show the differences more clearly.

Figure 3.8 shows the precision/recall for the *unweighted Exact Mention Match* metric, where the BiLSTM–CRF architecture is the top performer with an F_1 of 0.67 (0.75 precision, 0.61 recall) followed closely by Submission 2 with an F_1 of 0.66 (0.70 precision, 0.64 recall).

Performance of the BiLSTM–softmax and BiLSTM–CRF–ELMo are similar in this metric, while BiLSTM–CRF–ELMo was slightly better in the *weighted Exact Mention Match* metric.

The baseline approach had the highest recall but has poor precision.

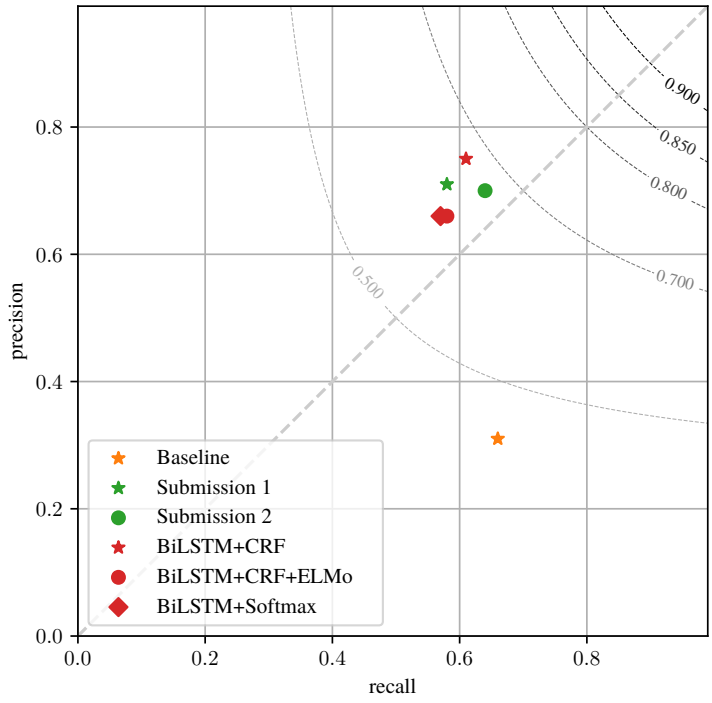

Figure 3.8: *unweighted Exact Mention Match* for different architectures, curved lines show F_1 iso-curves, diagonal line denotes precision = recall.

3.4 Map Performance

Almost all models performed better than the solutions proposed in this thesis in Map performance (fig. 3.9). The best performing state-of-the-art system is again, significantly higher than our models achieving an F_1 of 0.79 (0.83

precision, 0.79 recall). Followed by Submission 1 with an F_1 at 0.53 (0.71 precision, 0.50 recall).

Among the approaches implemented in this project, BiLSTM–CRF performed the best with an F_1 of 0.49 (0.61 precision, 0.48 recall) with an increase in F_1 of 0.06 over the baseline results. Again, similar to the Scan problem, the baseline method had a high recall compared to my methods.

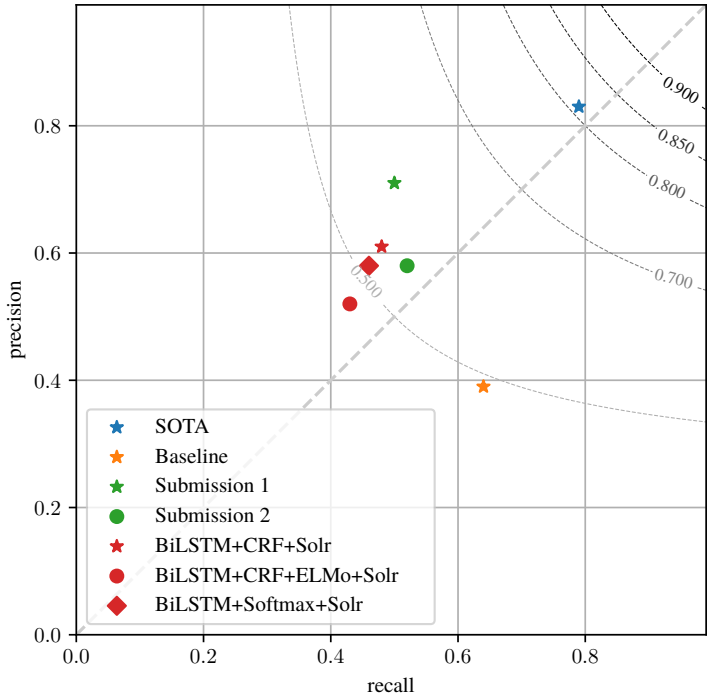

Figure 3.9: MedDRA coding performance, curved lines show F_1 iso-curves, diagonal line denotes precision = recall.

Model	Exact mention Match*			Exact Mention Match†			MedDRA coding‡			Quality§
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	
SOTA 1 (Team 2)	0.93	0.85	0.89	–	–	–	0.83	0.79	0.79	0.93
SOTA 2 (Team 11)	0.92	0.86	0.89	–	–	–	0.81	0.77	0.76	0.93
SOTA 3 (Team 1)	0.93	0.79	0.86	–	–	–	0.79	0.68	0.70	0.96
Baseline (Dictionary)	0.65	**0.76**	0.70	0.31	**0.66**	0.42	0.39	**0.64**	0.43	0.85
Submission 1	**0.88**	0.65	0.75	**0.71**	0.58	0.64	**0.71**	0.50	**0.53**	0.92
Submission 2	0.87	0.70	**0.78**	0.70	0.64	**0.66**	0.58	0.52	0.51	0.95
Best BiLSTM+Softmax	0.83	0.64	0.72	0.66	0.57	0.61	0.58	0.46	0.46	0.93
Best BiLSTM+CRF	**0.88**	0.66	0.76	**0.75**	0.61	**0.67**	0.61	0.48	0.49	**0.96**
ELMo+BiLSTM+CRF	0.83	0.66	0.73	0.66	0.58	0.62	0.52	0.43	0.43	0.93

*weighted

†unweighted

‡macro-averaged by section

§label scope

– unreported

Table 3.3: Evaluation metrics for the different architectures.

Engine	No re-ranking			Bert (bert-base-uncased)			BioBert		
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1
Whoosh	0.70	0.41	0.47	0.50	0.29	0.32	0.52	0.30	0.34
Solr	0.61	0.48	0.49	0.47	0.37	0.38	0.46	0.36	0.37

Table 3.4: Re-ranking performance for MedDRA retrieval macro-averaged per section on the best BiLSTM–CRF model

Chapter 4

Discussion

4.1 Model Selection

My initial approach to the Scan problem was to use a sequence-to-sequence model. Sequence-to-sequence model have shown great results in Neural Machine Translation (NMT) [21], however it is easy to see why this would fail in this particular formulation of the problem. In a language modelling problem, the source and target vocabulary have a large number of states, allowing the decoder to sample from a large probability distribution. In my formulation of the sequence-to-sequence model where there is a large set of input states and only two output states, it is not ideal since the neural model can be simplified into a state diagram with two states where the transitions are conditioned on the previous output and the hidden state (fig. 4.1).

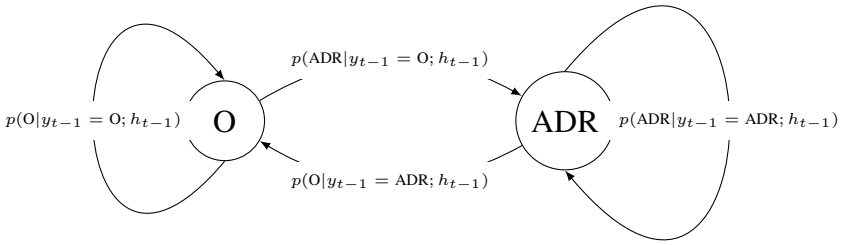

Figure 4.1: Simplified sequence to sequence model

The choice of BiLSTM–CRF model was justified by multiple instances of previous work where such an architecture was successfully used to perform NER and relation classification tasks [22–27].

The top performing systems in the ADE Evaluation Challenge [20] also

used a similar architecture with an additional rule-based system for special handling of tables and discontinuous mentions.

4.2 Evaluation Metrics

Even though the Scan and Map problems are described as two tasks in this project, the evaluation script provided by the FDA evaluate them jointly because the concept of a *Mention*, in this particular challenge is related both to the span and the MedDRA mapping.

The two-step validation of the models makes it difficult to evaluate the models easily. While the k-fold cross-validation hinted that the best model was the BiLSTM–CRF model with a hidden size $H = 1024$ and number of layers $N = 1$, evaluation on the test split reported that the best model was a BiLSTM–CRF with $H = 300$ and $N = 5$ (`id:0631ef1a` table B.1). However, this could be an artefact of the training–test split. I chose to keep the test split fixed because the previous attempts (“Dictionary”, “Submission 1” and “Submission 2”) used the same dataset as a test set and would be a good baseline to compare against.

Additionally, the metrics for the best performing systems in the ADE Eval challenge results are not directly comparable since the FDA has not released the annotations for the 2000 test labels that they use internally. So, while they are useful as an upper-bound reference, only the models “Dictionary”, “Submission 1”, “Submission 2” and the models presented in this thesis are directly comparable.

4.3 Scan Performance

As I mentioned earlier, I have chosen to ignore the “Exact match (discontinuous)” from the results. This is because the unweighted precision and recall was 0 for all models. This is not surprising for the BiLSTM–softmax classifier since it has no concept of what came before and after the current token. However, I would have expected the CRF classifier to perform better in this metric. Closer inspection reveals that this is possibly the result of the transition matrix formulated in algorithm 2. We see that an O token followed by an I token is normalized into a B token. Given a phrase such as: "The mean increases from baseline for total cholesterol " where *increases cholestrol* form a single mention, the correct labels would be: "O O B O O O O I "; instead the transition matrix encodes it as "O O B O O O O B ". This might explain the

similar performance of the CRF and softmax classifiers. This also affects the MedDRA coding since the two phrases *increases* and *cholesterol* form two separate queries and are not correctly resolved to their MedDRA concept IDs. Thus, correct label encoding could boost the classification scores greatly.

Of the three models, BiLSTM–CRF had the best overall performance, marginally beating Submission 2 in the unweighted Exact Mention Match metric.

Due to time constraints I was not able to evaluate parameters for the BiLSTM-CRF–ELMo model. Looking at the Validation F_1 across the different parameters for the BiLSTM–CRF–ELMo model (fig. 3.6) we can see that the model starts to perform well as $H \rightarrow 1024$, this makes sense since the base ELMo embedding vectors are of size 1024, as a result setting, $H < 1024$ results in loss of information that is output from the language model. For this particular model, it would possibly have been beneficial to vary the hidden sizes ≥ 1024 instead of the same range used for the other models. The variance in validation F_1 across folds is also significantly lower for the model with ELMo embeddings visible in fig. 3.6 compared to fig. 3.1 and fig. 3.3 suggesting that the introduction of a pre-trained language model adds some stability to the model performance across different datasets.

4.4 Map Performance

The MedDRA mapping of all three models is poor because it selects only the first (top-scoring) match to include in the mention, adding subsequent matches (e.g. the first two matches) could possibly bring it closer to Submission 2 scores. Referring back to the definition of *properly grounded* as "a correct MedDRA code is one which is realized in the gold standard by at least one mention which is paired with a submission mention with the same code" [28], we can exploit this to possibly boost the MedDRA retrieval scores.

The search engine Solr fails on queries involving ranges "neutrophil count < 1.5"; involving phrases that denote changes "ALT elevations greater than 3 * ULN", "elevations in lipid parameters", "Mean LDL cholesterol increased"; those involving acronyms "GI perforation events", "APA development" or a combination "decreases in absolute lymphocyte counts below 500"; this requires augmentation with a language model. A possible solution would be to use search engine index for simpler queries and use a language model for more complex queries (perhaps ELMo or BERT) since concepts such as "elevated", "increased" should ideally map to similar vectors and it would also be able to learn representations for ranges.

Another issue that contributed to poor mapping performance was bad tokenisation of the source text and generic matches. Since tokenisation is the first process in the pipeline, it has a domino effect within the model. The search engine failed to match cases where numbers and symbols were tokenised poorly such as: “pneumoniass]”, “= 150Maintenance”, “elevationss]”, “crampingt/s]”. It also failed on (possible) ADRs which were too ambiguous or were acronyms: “Malignancies”, “RVO”, “TMA”.

Chapter 5

Conclusion and Future Work

Despite the issues introduced by data preparation and tokenisation, modern NLP techniques show promising results in the detection of ADRs in free-form text. My method performed significantly better than the dictionary-based approach developed at UMC[10] on the dataset provided by the FDA, with an increase of 0.25 in F_1 for the Exact Mention Match evaluation metric and an increase of 0.06 in macro-averaged F_1 for the the MedDRA coding evaluation. Albeit demonstrating improvements over the UMC baseline, the best performing pipeline developed in this thesis is still below the state-of-the-art results (table 3.3) (data received in a personal communication, publication of these results is pending). The best-performing state-of-the-art system (SOTA 1 in table 3.3) in the ADE Evaluation Challenge was using an ensemble neural method augmented with a set of rules tailored to the task.

A big source of error in this project arises from the difficulty of labelling the spans when preprocessing the raw training data. The IOB2 scheme is limited when there are overlapping and discontinuous tokens. Indeed, almost all models proposed operate on a token level and disregard the concept of spans, this means that detection of nested and overlapping entities is not possible with these models; this is a disadvantage since the classification is being done in a single pass (each token can only belong to one mention) there are countless examples in the dataset where this is not true (i.e. the same token belongs to multiple spans). Recently, Eberts and Ulges proposed a span-based entity recognition system that uses a pre-trained BERT transformer, which could be a possible approach to mitigate this problem [29]. Alternatively, improved span representation might also lead to better results and would be worth investigating in future research.

Additional work is also possible within the architecture proposed in this

thesis. Recent development in attention-based models could be used to augment and weigh the feature vectors from the BiLSTM layer with a location based attention layer [30] to improve context retention of BiLSTM layers along long sequences. This is especially important since sentence tokenisation can sometimes fail and produce really long sequences. Furthermore, the Sequence-to-Sequence model I initially proposed could be reformulated to map to $w \mapsto w$ for non ADR tokens and $w \mapsto \{\mathrm{O}, \mathrm{B}, \mathrm{I}\}$ for ADR tokens.

In conclusion, extraction of ADRs from documents is a complex task due to the structureless-ness of free-form text. While traditional NER might be one approach to the Scan problem, it cannot solve it entirely due to additional complexities such as nested entities or shared tokens across mentions. To address the Scan problem fully, a system needs to have some idea of a language model to handle complex formulations (e.g. *Latent infections can be reactivated*, *serum phosphorus levels at least 0.3 mmol/L but less than 0.6 mmol/L*) that don't fit into the token-level Named Entity framework. Nonetheless, this work presents a clear improvement over the existing algorithm that was developed at UMC. Future research should be carried on evaluating how classification thresholds could be tailored to UMC's requirements in precision, instead of optimizing the algorithm on the more general F_1 metric, which gives an equal weight to precision and recall. A more stringent threshold could reduce the amount of false positives while still leading to substantial time improvements in the safety signal detection process. The ability to automatically and reliably detect, extract and map ADR from any text data source has a great role to play in the future of semi-automated pharmacovigilance and, considering the ever increasing amount of textual information available in the healthcare sector, will help develop comprehensive knowledge-bases on drugs and their use in the general population.

Bibliography

[1] Michael S. Kinch et al. “An Overview of FDA-Approved New Molecular Entities: 1827–2013”. en. In: *Drug Discovery Today* 19.8 (Aug. 2014), pp. 1033–1039. ISSN: 1359-6446. DOI: 10.1016/j.drudis.2014.03.018. URL: http://www.sciencedirect.com/science/article/pii/S1359644614001032 (visited on 10/28/2019).

[2] K. I. Kaitin and J. A. DiMasi. “Pharmaceutical Innovation in the 21st Century: New Drug Approvals in the First Decade, 2000–2009”. en. In: *Clinical Pharmacology & Therapeutics* 89.2 (2011), pp. 183–188. ISSN: 1532-6535. DOI: 10.1038/clpt.2010.286. URL: https://ascpt.onlinelibrary.wiley.com/doi/abs/10.1038/clpt.2010.286 (visited on 10/28/2019).

[3] World Health Organization et al. *Pharmacovigilance: Ensuring the Safe Use of Medicines*. Tech. rep. Geneva: World Health Organization, 2004.

[4] Abhyuday Jagannatha et al. “Overview of the First Natural Language Processing Challenge for Extracting Medication, Indication, and Adverse Drug Events from Electronic Health Record Notes (MADE 1.0)”. en. In: *Drug Safety* 42.1 (Jan. 2019), pp. 99–111. ISSN: 0114-5916, 1179-1942. DOI: 10.1007/s40264-018-0762-z. URL: http://link.springer.com/10.1007/s40264-018-0762-z (visited on 10/22/2019).

[5] Isabel Segura-Bedmar and Paloma Martínez. “Pharmacovigilance through the Development of Text Mining and Natural Language Processing Techniques”. en. In: *Journal of Biomedical Informatics* 58 (Dec. 2015), pp. 288–291. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2015.11.001. URL: http://www.sciencedirect.com/science/article/pii/S1532046415002385 (visited on 10/28/2019).

[6] Marie Lindquist. “VigiBase, the WHO Global ICSR Database System: Basic Facts”. en. In: *Drug Information Journal* 42.5 (Sept. 2008), pp. 409–419. ISSN: 0092-8615. DOI: 10.1177/009286150804200501. URL: https://doi.org/10.1177/009286150804200501 (visited on 10/28/2019).

[7] A. Bate and S. J. W. Evans. “Quantitative Signal Detection Using Spontaneous ADR Reporting”. en. In: *Pharmacoepidemiology and Drug Safety* 18.6 (2009), pp. 427–436. ISSN: 1099-1557. DOI: 10.1002/pds.1742. URL: http://onlinelibrary.wiley.com/doi/abs/10.1002/pds.1742 (visited on 05/19/2020).

[8] Ola Caster et al. “Improved Statistical Signal Detection in Pharmacovigilance by Combining Multiple Strength-of-Evidence Aspects in vigiRank: Retrospective Evaluation against Emerging Safety Signals”. en. In: *Drug Safety* 37.8 (Aug. 2014), pp. 617–628. ISSN: 0114-5916, 1179-1942. DOI: 10.1007/s40264-014-0204-5. URL: http://link.springer.com/10.1007/s40264-014-0204-5 (visited on 05/19/2020).

[9] Ola Caster et al. “vigiRank for Statistical Signal Detection in Pharmacovigilance: First Results from Prospective Real-World Use”. en. In: *Pharmacoepidemiology and Drug Safety* 26.8 (Aug. 2017), pp. 1006–1010. ISSN: 10538569. DOI: 10.1002/pds.4247. URL: http://doi.wiley.com/10.1002/pds.4247 (visited on 05/19/2020).

[10] Gunnar Dahlberg. *Implementation and Evaluation of a Text Extraction Tool for Adverse Drug Reaction Information*. eng. 2010. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-134063 (visited on 11/08/2019).

[11] Switzerland) WHO Meeting on International Drug Monitoring: the Role of National Centres (1971: Geneva and World Health Organization. *International Drug Monitoring : The Role of National Centres , Report of a WHO Meeting [Held in Geneva from 20 to 25 September 1971]*. en. World Health Organization, 1972. ISBN: 978-92-4-120498-9. URL: https://apps.who.int/iris/handle/10665/40968 (visited on 03/16/2020).

[12] ema. *ICH E6 (R2) Good Clinical Practice*. en. Text. Sept. 2018. URL: https://www.ema.europa.eu/en/ich-e6-r2-good-clinical-practice (visited on 03/16/2020).

[13] Erik F. Tjong Kim Sang and Jorn Veenstra. "Representing Text Chunks". en. In: *arXiv:cs/9907006* (July 1999). arXiv: cs/9907006. URL: http://arxiv.org/abs/cs/9907006 (visited on 01/03/2020).

[14] Matthew E. Peters et al. "Deep Contextualized Word Representations". In: *arXiv:1802.05365 [cs]* (Mar. 2018). arXiv: 1802.05365 [cs]. URL: http://arxiv.org/abs/1802.05365 (visited on 11/23/2019).

[15] Tomas Mikolov et al. "Efficient Estimation of Word Representations in Vector Space". In: *arXiv:1301.3781 [cs]* (Sept. 2013). arXiv: 1301.3781 [cs]. URL: http://arxiv.org/abs/1301.3781 (visited on 10/27/2019).

[16] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". In: (May 2016). arXiv: 1409.0473. URL: http://arxiv.org/abs/1409.0473 (visited on 11/25/2019).

[17] Dong Yu, Shizhen Wang, and Li Deng. "Sequential Labeling Using Deep-Structured Conditional Random Fields". en. In: *IEEE Journal of Selected Topics in Signal Processing* 4.6 (Dec. 2010), pp. 965–973. ISSN: 1932-4553, 1941-0484. DOI: 10.1109/JSTSP.2010.2075990. URL: http://ieeexplore.ieee.org/document/5570918/ (visited on 11/26/2019).

[18] Maksim Belousov, William G. Dixon, and Goran Nenadic. "MedNorm: A Corpus and Embeddings for Cross-Terminology Medical Concept Normalisation". In: *Proceedings of the Fourth Social Media Mining for Health Applications (#SMM4H) Workshop & Shared Task*. Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 31–39. DOI: 10.18653/v1/W19-3204. URL: https://www.aclweb.org/anthology/W19-3204 (visited on 11/28/2019).

[19] Zongcheng Ji, Qiang Wei, and Hua Xu. "BERT-Based Ranking for Biomedical Entity Normalization". In: *arXiv:1908.03548 [cs]* (Aug. 2019). arXiv: 1908.03548 [cs]. URL: http://arxiv.org/abs/1908.03548 (visited on 11/29/2019).

[20] Corporation MITRE. *For ADE Eval Performers: ADE Eval Final Report Version 8.0*. May 2019.

[21] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. "Sequence to Sequence Learning with Neural Networks". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., 2014, pp. 3104–3112. URL: http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf (visited on 11/04/2019).

[22] Alec B. Chapman et al. "Detecting Adverse Drug Events with Rapidly Trained Classification Models". en. In: *Drug Safety* 42.1 (Jan. 2019), pp. 147–156. ISSN: 0114-5916, 1179-1942. DOI: 10.1007/s40264-018-0763-y. URL: http://link.springer.com/10.1007/s40264-018-0763-y (visited on 10/11/2019).

[23] Bharath Dandala, Venkata Joopudi, and Murthy Devarakonda. "Adverse Drug Events Detection in Clinical Notes by Jointly Modeling Entities and Relations Using Neural Networks". en. In: *Drug Safety* 42.1 (Jan. 2019), pp. 135–146. ISSN: 1179-1942. DOI: 10.1007/s40264-018-0764-x. URL: https://doi.org/10.1007/s40264-018-0764-x (visited on 10/15/2019).

[24] Víctor Suárez-Paniagua et al. "A Two-Stage Deep Learning Approach for Extracting Entities and Relationships from Medical Texts". en. In: *Journal of Biomedical Informatics* 99 (Nov. 2019), p. 103285. ISSN: 15320464. DOI: 10.1016/j.jbi.2019.103285. URL: https://linkinghub.elsevier.com/retrieve/pii/S1532046419302047 (visited on 10/22/2019).

[25] Xuezhe Ma and Eduard Hovy. "End-to-End Sequence Labeling via Bi-Directional LSTM-CNNs-CRF". In: *arXiv:1603.01354 [cs, stat]* (May 2016). arXiv: 1603.01354 [cs, stat]. URL: http://arxiv.org/abs/1603.01354 (visited on 11/29/2019).

[26] Giannis Bekoulis et al. "Joint Entity Recognition and Relation Extraction as a Multi-Head Selection Problem". In: *Expert Systems with Applications* 114 (Dec. 2018), pp. 34–45. ISSN: 09574174. DOI: 10.1016/j.eswa.2018.07.032. arXiv: 1804.07847. URL: http://arxiv.org/abs/1804.07847 (visited on 12/09/2019).

[27] Ling Luo et al. "An Attention-Based BiLSTM-CRF Approach to Document-Level Chemical Named Entity Recognition". In: *Bioinformatics (Oxford, England)* 34 (Nov. 2017). DOI: 10.1093/bioinformatics/btx761.

[28] Corporation MITRE. *For ADE Eval Performers: Expanded ADE Eval Evaluation Metrics and Spreadsheets Version 5.0*. May 2019.

[29] Markus Eberts and Adrian Ulges. "Span-Based Joint Entity and Relation Extraction with Transformer Pre-Training". en. In: *arXiv:1909.07755 [cs]* (Nov. 2019). arXiv: 1909.07755 [cs]. URL: http://arxiv.org/abs/1909.07755 (visited on 11/25/2019).

[30] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. "Effective Approaches to Attention-Based Neural Machine Translation". en. In: *arXiv:1508.04025 [cs]* (Sept. 2015). arXiv: 1508.04025 [cs]. URL: http://arxiv.org/abs/1508.04025 (visited on 01/05/2020).

[31] Weltgesundheitsorganisation and Collaborating Centre for International Drug Monitoring, eds. *The Importance of Pharmacovigilance: Safety Monitoring of Medicinal Products*. en. Geneva: WHO [u.a.], 2002. ISBN: 978-92-4-159015-0.

[32] L. Härmark and A. C. van Grootheest. "Pharmacovigilance: Methods, Recent Developments and Future Perspectives". en. In: *European Journal of Clinical Pharmacology* 64.8 (Aug. 2008), pp. 743–752. ISSN: 1432-1041. DOI: 10.1007/s00228-008-0475-9. URL: https://doi.org/10.1007/s00228-008-0475-9 (visited on 10/28/2019).

[33] Patricia Mozzicato. "MedDRA: An Overview of the Medical Dictionary for Regulatory Activities". en. In: *Pharmaceutical Medicine* 23.2 (Apr. 2009), pp. 65–75. ISSN: 1178-2595, 1179-1993. DOI: 10.1007/BF03256752. URL: http://link.springer.com/10.1007/BF03256752 (visited on 10/15/2019).

[34] Thomas Ly et al. "Evaluation of Natural Language Processing (NLP) Systems to Annotate Drug Product Labeling with MedDRA Terminology". en. In: *Journal of Biomedical Informatics* 83 (July 2018), pp. 73–86. ISSN: 15320464. DOI: 10.1016/j.jbi.2018.05.019. URL: https://linkinghub.elsevier.com/retrieve/pii/S1532046418301060 (visited on 10/11/2019).

[35] Rave Harpaz et al. "Text Mining for Adverse Drug Events: The Promise, Challenges, and State of the Art". en. In: *Drug Safety* 37.10 (Oct. 2014), pp. 777–790. ISSN: 0114-5916, 1179-1942. DOI: 10.1007/s40264-014-0218-z. URL: http://link.springer.com/10.1007/s40264-014-0218-z (visited on 09/21/2019).

[36] Carol Friedman and Noémie Elhadad. "Natural Language Processing in Health Care and Biomedicine". en. In: *Biomedical Informatics: Computer Applications in Health Care and Biomedicine*. Ed. by Edward H. Shortliffe and James J. Cimino. London: Springer, 2014, pp. 255–284. ISBN: 978-1-4471-4474-8. DOI: 10.1007/978-1-4471-4474-8_8. URL: https://doi.org/10.1007/978-1-4471-4474-8_8 (visited on 10/23/2019).

[37] D. Terence Langendoen. "Review of Natural Language and Universal Grammar". In: *Language* 69.4 (1993), pp. 825–828. ISSN: 0097-8507. DOI: 10.2307/416893. URL: https://www.jstor.org/stable/416893 (visited on 10/31/2019).

[38] Yuan Luo et al. "Natural Language Processing for EHR-Based Pharmacovigilance: A Structured Review". en. In: *Drug Safety* 40.11 (Nov. 2017), pp. 1075–1089. ISSN: 1179-1942. DOI: 10.1007/s40264-017-0558-6. URL: https://doi.org/10.1007/s40264-017-0558-6 (visited on 10/23/2019).

[39] Wendy W. Chapman et al. "Overcoming Barriers to NLP for Clinical Text: The Role of Shared Tasks and the Need for Additional Creative Solutions". en. In: *Journal of the American Medical Informatics Association* 18.5 (Sept. 2011), pp. 540–543. ISSN: 1067-5027. DOI: 10.1136/amiajnl-2011-000465. URL: https://academic.oup.com/jamia/article/18/5/540/829390 (visited on 10/23/2019).

[40] Michael A. Nielsen. "Neural Networks and Deep Learning". en. In: (2015). URL: http://neuralnetworksanddeeplearning.com (visited on 01/04/2020).

[41] Keyuan Jiang et al. "Assessment of Word Embedding Techniques for Identification of Personal Experience Tweets Pertaining to Medication Uses". en. In: *Precision Health and Medicine*. Ed. by Arash Shaban-Nejad and Martin Michalowski. Vol. 843. Cham: Springer International Publishing, 2020, pp. 45–55. ISBN: 978-3-030-24408-8 978-3-030-24409-5. DOI: 10.1007/978-3-030-24409-5_5. URL: http://link.springer.com/10.1007/978-3-030-24409-5_5 (visited on 10/22/2019).

[42] Piotr Bojanowski et al. "Enriching Word Vectors with Subword Information". In: *arXiv:1607.04606 [cs]* (June 2017). arXiv: 1607.04606

[cs]. URL: http://arxiv.org/abs/1607.04606 (visited on 12/19/2019).

[43] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. "On the Difficulty of Training Recurrent Neural Networks". en. In: (2013), p. 9.

[44] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[45] Denny Britz et al. "Massive Exploration of Neural Machine Translation Architectures". en. In: *arXiv:1703.03906 [cs]* (Mar. 2017). arXiv: 1703.03906 [cs]. URL: http://arxiv.org/abs/1703.03906 (visited on 12/25/2019).

[46] John Lafferty, Andrew McCallum, and Fernando C N Pereira. "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data". en. In: (2001), p. 10.

[47] Charles Sutton. "An Introduction to Conditional Random Fields". en. In: *Foundations and Trends® in Machine Learning* 4.4 (2012), pp. 267–373. ISSN: 1935-8237, 1935-8245. DOI: 10.1561/2200000013. URL: http://www.nowpublishers.com/article/Details/MAL-013 (visited on 11/06/2019).

[48] Jacob Devlin et al. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding". In: *arXiv:1810.04805 [cs]* (May 2019). arXiv: 1810.04805 [cs]. URL: http://arxiv.org/abs/1810.04805 (visited on 11/28/2019).

[49] Ashish Vaswani et al. "Attention Is All You Need". In: *arXiv:1706.03762 [cs]* (Dec. 2017). arXiv: 1706.03762 [cs]. URL: http://arxiv.org/abs/1706.03762 (visited on 10/23/2019).

[50] Christopher Manning, Prabhakar Raghavan, and Hinrich Schuetze. "Introduction to Information Retrieval". en. In: (2009), p. 581.

[51] Stephen Robertson. "The Probabilistic Relevance Framework: BM25 and Beyond". en. In: *Foundations and Trends® in Information Retrieval* 3.4 (2010), pp. 333–389. ISSN: 1554-0669, 1554-0677. DOI: 10.1561/1500000019. URL: http://www.nowpublishers.com/product.aspx?product=INR%5C&doi=1500000019 (visited on 01/04/2020).

[52] *Apache/Lucene-Solr*. en. URL: https://github.com/apache/lucene-solr/blob/232d940c98429060576c9f2b6a0e3a49b019a688/lucene/core/src/java/org/apache/lucene/search/similarities/BM25Similarity.java%5C#L71 (visited on 03/13/2020).

[53] Edward Loper, Steven Bird, and Ewan Klein. *Natural Language Processing with Python*. en. ISBN: 978-0-596-51649-9. URL: http://shop.oreilly.com/product/9780596516499.do (visited on 10/15/2019).

[54] Jon D. Duke and Jeff Friedlin. "ADESSA: A Real-Time Decision Support Service for Delivery of Semantically Coded Adverse Drug Event Data". In: *AMIA Annual Symposium Proceedings* 2010 (2010), pp. 177–181. ISSN: 1942-597X. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3041415/ (visited on 10/15/2019).

[55] J. C. Smith et al. "Lessons Learned from Developing a Drug Evidence Base to Support Pharmacovigilance". en. In: *Applied Clinical Informatics* 04.4 (2013), pp. 596–617. ISSN: 1869-0327. DOI: 10.4338/ACI-2013-08-RA-0062. URL: http://www.thieme-connect.de/DOI/DOI?10.4338/ACI-2013-08-RA-0062 (visited on 10/15/2019).

[56] Ehtesham Iqbal et al. "ADEPt, a Semantically-Enriched Pipeline for Extracting Adverse Drug Events from Free-Text Electronic Health Records". In: *PLoS ONE* 12.11 (Nov. 2017). ISSN: 1932-6203. DOI: 10.1371/journal.pone.0187121. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5679515/ (visited on 10/11/2019).

[57] Xi Yang et al. "Identifying Relations of Medications with Adverse Drug Events Using Recurrent Convolutional Neural Networks and Gradient Boosting". en. In: *Journal of the American Medical Informatics Association* (Aug. 2019), ocz144. ISSN: 1527-974X. DOI: 10.1093/jamia/ocz144. URL: https://academic.oup.com/jamia/advance-article/doi/10.1093/jamia/ocz144/5555856 (visited on 10/11/2019).

[58] Susmitha Wunnava et al. "Adverse Drug Event Detection from Electronic Health Records Using Hierarchical Recurrent Neural Networks with Dual-Level Embedding". en. In: *Drug Safety* 42.1 (Jan. 2019), pp. 113–122. ISSN: 0114-5916, 1179-1942. DOI: 10.1007/s4026

4-018-0765-9. URL: http://link.springer.com/10.1007/s40264-018-0765-9 (visited on 10/11/2019).

[59] Mert Tiftikci et al. “Extracting Adverse Drug Reactions Using Deep Learning and Dictionary Based Approaches”. In: *TAC*. 2017.

Appendix A

State of the Art

In this chapter, I will further define Pharmacovigilance and its role in Drug Safety, followed by a brief description of how the results of this project fit into a larger context at Uppsala Monitoring Centre (UMC). I will then describe some core conceptual ideas related to this project such as the organization of the dictionary Medical Dictionary for Regulatory Activities (MedDRA), shared task challenges and the problems they solve, followed by the description of the challenge itself that this project is based on, as well as the extensive metrics that are part of the Shared Task Challenge toolkit.

Next, I will describe the granular technical context of this project: starting with the organization of the challenge, introducing the notion of Structured Product Label (SPL), its various components and the extensive evaluation metrics defined by the challenge organizers. I will then continue describing what Natural Language Processing (NLP) is, why it is a difficult problem and its evolution from Classical NLP to Machine Learning (ML) based NLP. For the latter, I will concretely describe the various ideas related to ML in the context of NLP. Finally, I will add a brief about information retrieval methods relevant to this project and conclude with a Literature Review of previous attempts on similar problems.

A.1 Pharmacovigilance

Adverse events caused by drugs and medicines may manifest themselves as minor side effects such as a skin rash or an irritation, but it can also cause long-term damage or death. One notable case is the Thalidomide crisis of 1961, where a drug administered to pregnant mothers resulted in congenitally deformed infants. This incident highlighted the importance of pharmacovigi-

lance and continual monitoring, even at the post-marketing phase. The World Health Organization (WHO) defines pharmacovigilance as [31]:

> *Pharmacovigilance is the science and activities relating to the detection, assessment, understanding and prevention of adverse effects or any other possible drug-related problems.*

The main methods used in pharmacovigilance to detect adverse events are pre-marketing clinical trials, reports from pharmaceutical industry, literature studies and spontaneous reporting [32].

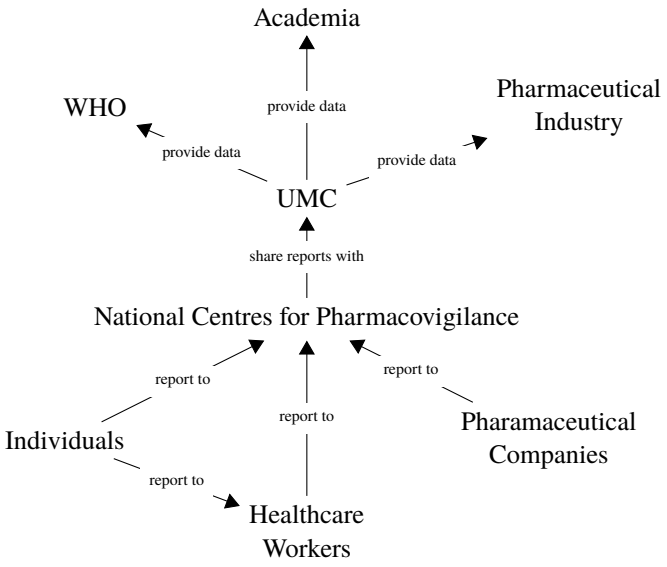

Figure A.1: Reporting structure

The WHO works with a number of partners (fig. A.1) in order to support international drug monitoring. Each partnering nation appoints a national pharmacovigilance centre for post-marketing surveillance. The UMC is an entity responsible for managing the international database of spontaneous Adverse Drug Reaction (ADR) reports [31].

A.2 Medical Dictionary for Regulatory Activities

MedDRA® the Medical Dictionary for Regulatory Activities terminology is the international medical terminology developed under the auspices of the In-

ternational Council for Harmonisation of Technical Requirements for Pharmaceuticals for Human Use (ICH). It is a clinically-validated, hierarchical medical terminology database used by regulatory bodies and bio-pharmaceutical industries [33]. Terms in the MedDRA dictionary are organized into five hierarchical levels, from general to specific:

1. System Organ Class (SOC, e.g. Cardiac disorders)
2. High Level Group Term (HLGT, e.g. Cardiac arrhythmias)
3. High Level Term (HLT, e.g. Rate and rhythm disorders NEC)
4. Preferred Term (PT, e.g. Arrhythmia)
5. Lowest Level Term (LLT, e.g. Arrhythmia NOS, Dysrhythmias)

The last class: Lowest Level Term (LLT) is a very fine-grained class that can contain synonyms, lexical variants and sub-elements of particular terms and form a many-to-one mapping onto the Preferred Term (PT) class.

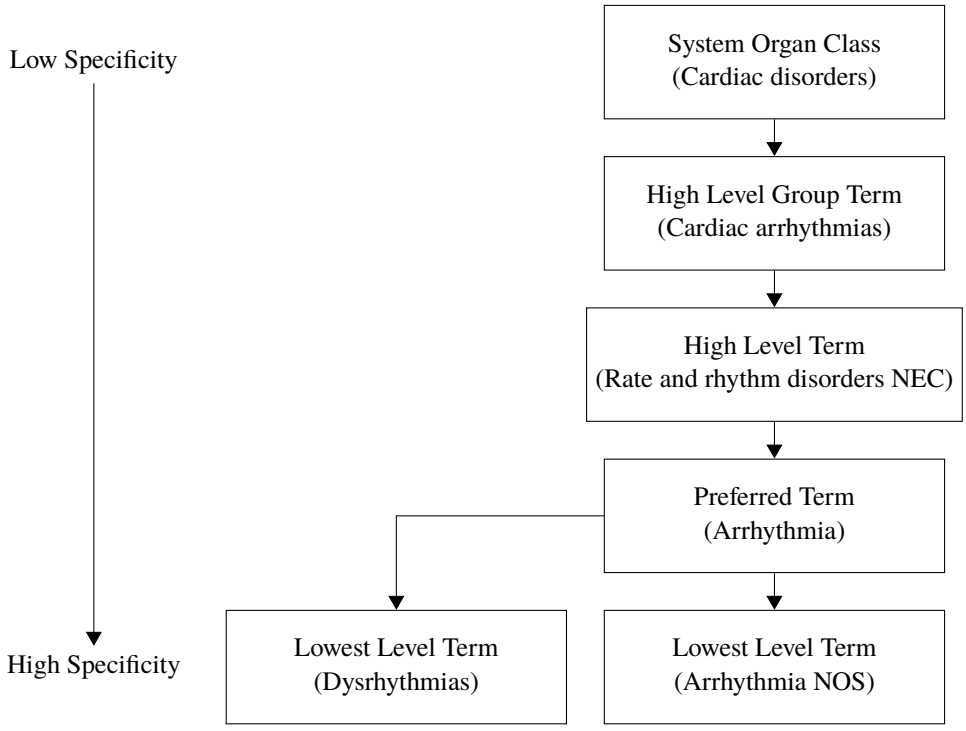

Figure A.2: MedDRA hierarchy

Despite the standardization, drug manufacturers are *not required* to use MedDRA terminology in product labels adding further challenges to applying automation to pharmacovigilance[34].

A.3 Shared Task Challenges

Shared Task Challenges are open invitations for researchers to solve a specific problem in various fields, including clinical NLP. They are often divided into *tasks* and researchers tend to focus on one or more of these tasks. Shared tasks provide researchers with a few important tools:

- An annotated and reliable Dataset, known as the Gold Standard (and additional test and validation datasets)
- Documentation on annotations, data format and evaluation criteria
- Evaluation metrics
- Additional tools for visualisation and data verification (optionally)
- A compilation of entries and their performance at the end of the challenge

A.4 FDA Challenge

The UMC is interested in an automated algorithm able to detect ADR mentions in product labels and map them to their relevant MedDRA PT codes to facilitate review and processing of Individual Case Safety Reports (ICSRs). The Office of Surveillance and Epidemiology (OSE) Food and Drugs Administration (FDA) challenge provides a standard dataset of drug labels (known as SPLs) for the same purpose, as described in the challenge homepage[1]:

> *OSE is interested in a tool that would enable pharmacovigilance safety evaluators to automate the identification of labelled Adverse Events (AEs) which could facilitate triage, review and processing of safety case reports.*

[1] `https://sites.mitre.org/adeeval/`

They provide 100 manually annotated *Gold Standard* SPL files for 100 drugs, 2000 unannotated test documents. Scripts for evaluating and cleaning the data files (for preparation of test and validation datasets) and for submission. A document describing the structure of the *Gold Standard* document, and a document describing the evaluation metrics.

The FDA defines two separate use cases for detection of ADR in product labels:

Front Office In this use-case the safety evaluators query the database with a known MedDRA code and the system should provide all product labels matching the MedDRA term and provide evidence. In this use-case the Map problem is the most important.

Back Office In this use-case the system is expected to automatically find and map the ADR detected in the product labels, possibly with a human evaluating the quality of the matches. In this use-case, the Span problem is the most important as any ambiguities can be resolved by a reviewer.

A.4.1 Structured Product Labels

SPL is an eXtended Markup Language (XML) document which is a downloadable electronic version of drug labels. They are provided in two formats: one targeted towards health professionals, and the other towards end-users, all files provided by the FDA challenge belong to the former category.

The *structured* in *Structured Product Labels*, refers only to a formatting structure of the document. The contents of each document is still free-form [35]. These documents are freely available for download from the DailyMed website maintained by the FDA[2]. The SPL files provided for the FDA challenge contains a subset of the XML files[3], specifically only the *Boxed Warning*, *Warnings*, *Precautions* and *Adverse Reactions* section, all of which contain free-text information. Figure A.3 shows an excerpt of the SPL file provided by the FDA challenge for the drug *Afinitor*.

[2] `https://dailymed.nlm.nih.gov/dailymed/`

[3] It should be noted that the challenge data represents a heavily pre-processed SPL file, where most of the XML structure is stripped away so that researchers can focus on the task instead of the details or parsing the complex and extensive SPL format.

```
<?xml version="1.0" encoding="UTF-8"?>
<GoldLabel drug="AFINITOR">
<Text>
<Section id="S1" name="adverse reactions">  6     ADVERSE REACTIONS

The following serious adverse reactions are discussed in greater detail in another section of
↪  the label  [see Warnings and Precautions (5)]  :

*  Non-infectious pneumonitis [see Warnings and Precautions (5.1)] .
*  Infections [see Warnings and Precautions (5.2)] .
*  Angioedema with concomitant use of ACE inhibitors [see Warnings and Precautions (5.3)] .
*  Stomatitis [see Warnings and Precautions (5.4)] .
*  Renal failure [see Warnings and Precautions (5.5)] .
*  Impaired wound healing [see Warnings and Precautions (5.6)] .
Because clinical trials are conducted under widely varying conditions, the adverse reaction
↪  rates observed cannot be directly compared to rates in other trials and may not reflect
↪  the rates observed in clinical practice.

 ... truncated ...

</Section>
</Text>
<IgnoredRegions>
<IgnoredRegion len="23" name="heading" section="S1" start="2" />
 ... truncated ...
<IgnoredRegion len="32" name="heading" section="S1" start="56491" />
</IgnoredRegions>
<Mentions>
<Mention id="M135" len="26" reason="from_drug_use" section="S1" start="177"
↪  type="OSE_Labeled_AE">
<Normalization meddra_llt="Pneumonitis" meddra_llt_id="10035742" meddra_pt="Pneumonitis"
↪  meddra_pt_id="10035742" />
</Mention>
<Mention id="M136" len="10" reason="from_drug_use" section="S1" start="248"
↪  type="OSE_Labeled_AE">
<Normalization meddra_llt="Infection"  meddra_llt_id="10021789"  meddra_pt="Infection"
↪  meddra_pt_id="10021789"  />
</Mention>
 ... truncated ...
</Mentions>
</GoldLabel>
```

Figure A.3: Structure of Gold Labelled SPL file provided by the FDA challenge

SPLs that are manually tagged for adverse events by domain experts are called *Gold Labels*, and will be used as a reference standard to train against and validate the performance of algorithms.

A SPL document consist of one or more sections of text (fig. A.4).

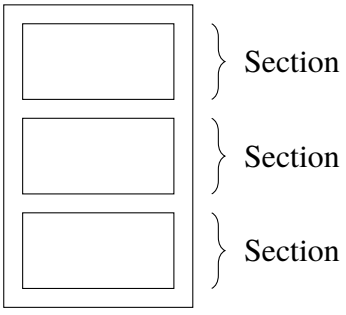

Figure A.4: General structure of an SPL document

A.4.2 Mention

A mention is an annotated instance of an ADR within the SPL section. It can contain one or more spans and association to one or more normalized MedDRA entry that is represented by the enclosed span. Spans can be disjoint but still represent a single mention.

Elevated SGPT (ALT) < 1%; SGOT (AST)
Normalization: Alanine aminotransferase increased (10001551)

Figure A.5: A mention with 2 spans (bordered)

A.4.3 Spans

A positively annotated ADR is called a *Mention*. Each section contains a number of *Mentions* specified by one or more *Spans* which are parametrized by a start index and length. The index is in reference to the first character of *each* section (fig. A.6), and newlines are counted as a character token.

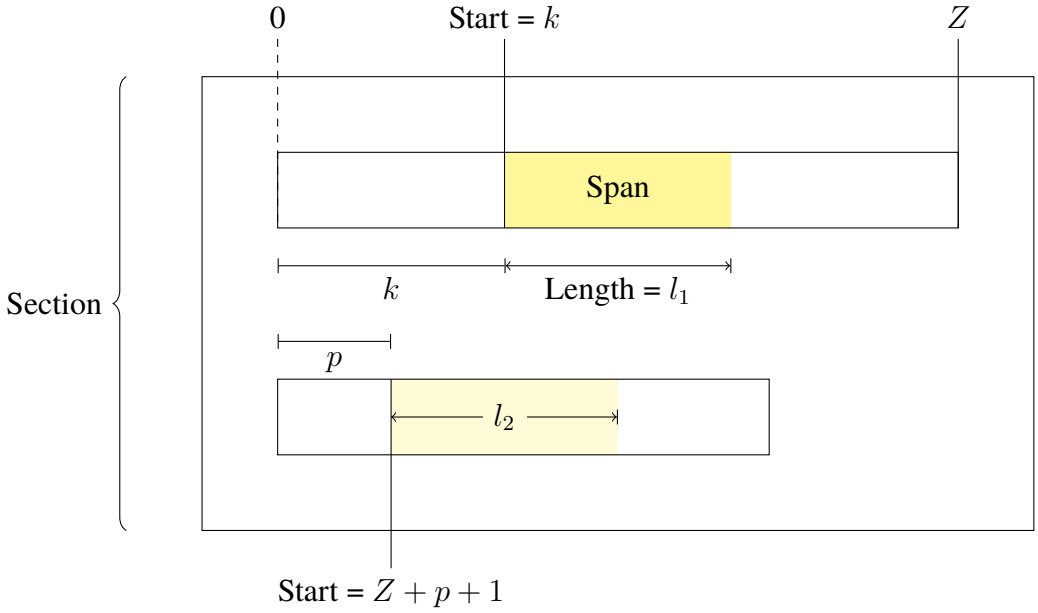

Figure A.6: Spans in Sections

Let $S_i = [s_i, e_i]$ represent a span with start index s_i, length l_i and end index $e_i = s_i + l_i$.

We define the span separation operation as:

$$\mathcal{D}(S_1, S_2) = \max(s_1, s_2) - \min(e_1, e_2) \tag{A.1}$$

And the span length operation as:

$$\|S_i\| = e_i - s_i = l_i \tag{A.2}$$

Adjacency

Two spans S_1 and S_2 are adjacent if $\mathcal{D}(S_1, S_2) = 0$.

Overlap

Two spans, S_1 and S_2 are overlapping if $\mathcal{D}(S_1, S_2) < 0$. There can be two variants of overlapping spans: partial overlaps where $|\mathcal{D}(S_1, S_2)| < \|S_2\|$, and complete overlaps where $|\mathcal{D}(S_1, S_2)| = \|S_2\|$

The SPL documents contain many such overlapping spans that represent distinct (possibly similar) ADR.

Cases of sensory or [sensorimotor [axonal polyneuropathy]]

Figure A.7: Examples of completely overlapping span

Most overlaps in the provided training data is in the form of complete overlaps.

Discontinuity

Discontinuity is the final case described by the separation operation and arises in cases where $\mathcal{D}(S_1, S_2) > 0$. This is very common in the provided annotated dataset, and the separation can span a few words to large sections.

Discontinuous spans come in two variants, the first are spans that represent a single ADR but are separated by one or more words. For example in fig. A.8 the correct ADR is *worsening BPH* but they are separated by a number of unrelated words. There are almost 3500 discontinuous annotations in the dataset [20].

Patients with BPH treated with androgens are at an increased risk for worsening of signs and symptoms of BPH.

Figure A.8: Example of discontinuous span

The second variant are spans that represent different ADRs but are linked by a common root as in fig. A.9. Here' the correct ADRs are: *fistulae tracheoesophageal*, *fistulae bronchopleural*, *fistulae biliary*, *fistulae vaginal*, *fistulae renal*, *fistulae bladder*.

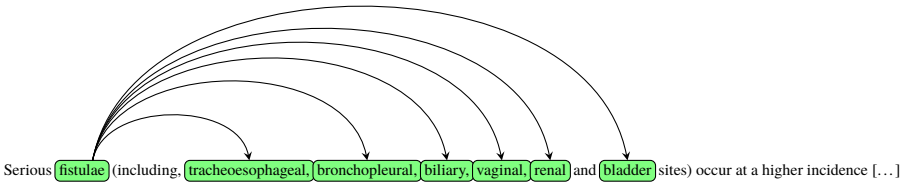

Figure A.9: Example of discontinuous spans with a common root

Alternatively, the connecting root can also follow the connected spans as in fig. A.10 where the correct ADRs are: *bacterial infections*, *fungal infections*, *viral infections* and *protozoal infections*.

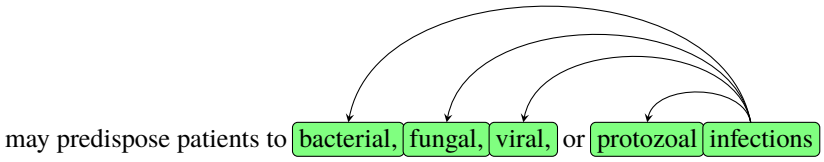

Figure A.10: Example of discontinuous span where the root follows the connected spans

A.4.4 Evaluation

The evaluation metrics provided by the FDA define multiple metrics for both the *Front Office* and *Back Office* use cases, this is illustrated in fig. A.11.

The evaluation is described in terms of two terms: *Gold Label* and *Submission Label*. *Gold Label* is the annotated label and *Submission Label* is the output from the ADR detection systems.

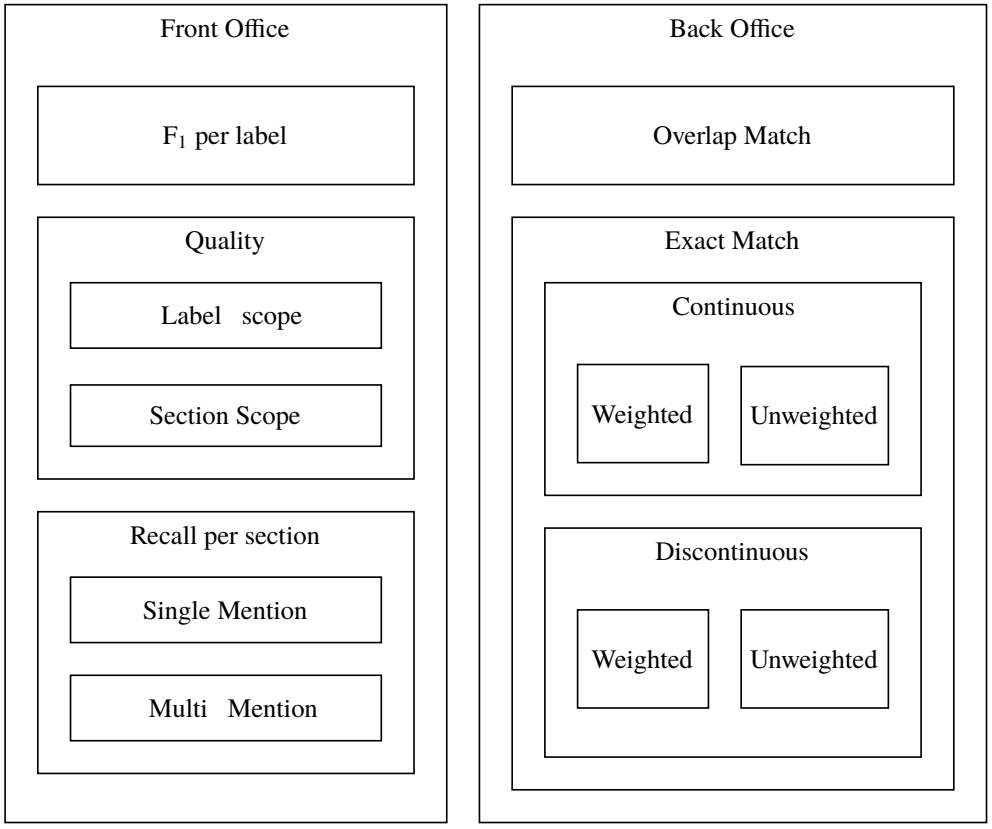

Figure A.11: Hierarchy of evaluation metrics

The front and back office metrics are based on the Precision/Recall/F_1 metrics defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{A.3}$$

$$\text{Recall} = \frac{\text{TP}}{\text{FP} + \text{FN}} \tag{A.4}$$

$$\text{F}_1 = 2 \cdot \left(\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \right) \tag{A.5}$$

Front Office Metrics

The front-office metrics consider the Map problem: evaluating the performance of mapping the detected ADR to their correct MedDRA dictionary entries. For this purpose, they define the term *Properly Grounded* which is

the instance where mapped MedDRA code in the Submission Label matches *at least one* MedDRA code in the Gold Label.

F_1 This is the MedDRA mapping metric in the Submission Label, macro averaged in the scope of SPL document.

Quality The Quality metric examines the quality of evidence for a given MedDRA match that is *properly grounded*. For each matching MedDRA code in the submission label, it looks at the proportion of matches in the gold label. The rationale for this metric is found in those cases where the Safety Evaluator actually refers to the text of the drug label, rather than the output of the analysis tool, to confirm the tool's analysis [20].

Recall This is the recall for mapped MedDRA matches in the Gold Label.

Back Office Metrics

Back-office metrics consider the Scan problem and defines the weighted and unweighted metrics as well as the concept of a *match*. It defines four variants of matches:

Exact Match The Submission Label contains a match that matches both the span and the mapped MedDRA code is properly grounded.

Overlap / Inexact Match The Submission Label contains a match that overlaps in span but either don't match the span exactly, does not match the MedDRA mapping, or both.

Missing Match The Submission Label does not contain a mention that is present in the Gold Label.

Spurious Match The Submission Label contains a mention that is not present in the Gold Label.

Since the Back Office use-case involves a human reviewing the output of the system, the weighted metrics take into account the difficulty of adding, correcting and removing ADR on the output of the system. The intuition based on the scoring is that adding a missing match is difficult since it requires reading through the product label and manually annotating the ADRs. Removing a spurious mention is fairly easy since all the context is provided by the system; and correcting an overlapping or inexact match is hard but not as hard as adding one (since some context is provided but it still requires manual work

looking up the correct MedDRA codes and/or fixing the spans). Taking this into account, given M exact matches, C overlap matches, N missing matches and S spurious matches:

$$\hat{N} = 1 \cdot N \tag{A.6}$$

$$\hat{S} = \frac{1}{4} \cdot S \tag{A.7}$$

$$\hat{C} = \frac{1}{2} \cdot C \tag{A.8}$$

$$\hat{M} = M + \hat{C} \tag{A.9}$$

The weighted precision and recall score for the back-office use-cased are defined in terms of these weighted metrics as [20]:

$$\text{Precision} = \frac{\hat{M}}{\hat{M} + \hat{C} + \hat{S}} \tag{A.10}$$

$$\text{Recall} = \frac{\hat{M}}{\hat{M} + \hat{C} + \hat{N}} \tag{A.11}$$

$$\tag{A.12}$$

A.5 Natural Language Processing

NLP is a multi-disciplinary field that combines linguistics, computer science, statistics and artificial intelligence in order to allow machines to understand and extract useful information from natural languages. Natural Language is the primary means of human communication, and the majority of clinical texts are recorded in free-form text narratives [36].

Natural languages are called *natural* because they evolve naturally through usage and time [37], in contrast to constructed languages (e.g. Esperanto). Due to the evolutionary nature of natural languages and since there is no central authority cataloguing and enforcing the formalisms of languages, they change over time, develop exceptions and separate into sublanguages and dialects. This evolutionary aspect gives natural languages efficiency since each evolution encodes many underlying assumptions intrinsic to the structure of the language, and is reliant on external knowledge. The process of language learning is an attempt to learn these intrinsic assumptions, grammar, rules and exceptions.

While natural languages provide efficiency in human communication, the efficiency is afforded at the cost of memory in the form of underlying assumptions both in the structure of the language, and the knowledge of the receiver[4]. This results in language learning to be both a complex and expensive task to perform algorithmically.

This section discusses briefly a general framework for understanding the various structures of language, the challenges, and possible solutions in language modelling and understanding.

A.5.1 Natural Language Structures

It is generally accepted that natural language understanding can be categorized into a hierarchical structure [36] as illustrated in fig. A.12. Each structure builds on the one at a lower level.

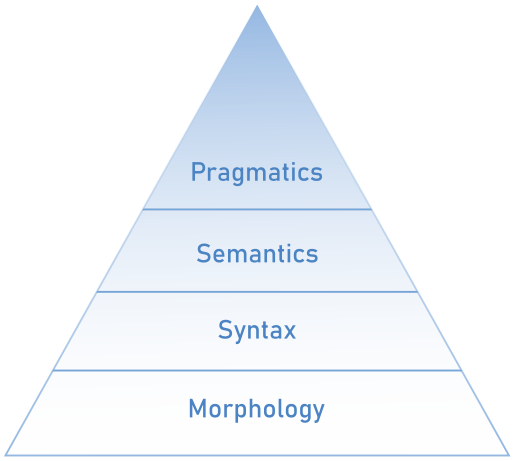

Figure A.12: Hierarchical structure of natural languages

Morphology is the most basic structure and informs how words are constructed. An example in English: the *root* word *work* can be morphed by the *-ing* suffix to form the *morpheme*: *working*.

Syntax is related to describing the structures of words and sentences. They often form the grammar rules of a language, for e.g. : the subject comes before the verb. Classifying words into *classes* such as *noun* and *verb* is also a part of syntax.

Semantics concerns itself with the understanding of meaning. A syntactically correct statement may have no (discernable) meaning. For e.g. the phrase

[4]Here, the *receiver* refers to the receiver of the information through speech or text

Colorless green ideas sleep furiously is syntactically correct, but semantically incomplete. While the phrases *colourless green* follow the adjective-noun syntax, it is not possible for something to be both *colourless* and *green*.

Pragmatics describes how the given text is understood given some contextual reference. For e.g. the term *mass* can be understood as *breast mass* in a mammography report, whereas it is understood as *mass in lung* in a radiological report of the chest. The meaning of the word *mass* changes based on the context. The contextual reference could also be time, for e.g. in a phrase containing the time reference *tomorrow* additional knowledge is required to know when the report was written [36].

Morphology and Syntax are often the *easy* parts of NLP, this is due to the fact that they require (relatively) small knowledge base and can mostly be parsed using dictionaries and rulesets, and require very little to no context. They are often solved using low-level text processing techniques and Finite State Machine (FSM) (e.g. a regular-expression engine) for matching and cleaning texts [36].

The main challenge of NLP is thus, provided by Semantics and Pragmatics, since they require significant knowledge accumulation and context and can result in linguistics structures that are often ambiguous or domain-dependent. For e.g. the phrase *The man put the rabbit in the hat. It jumped.*, the term *it* can refer to either the rabbit or the hat and it is not clear just by looking at the statement which subject *it* is referring to. However, with additional knowledge that hats don't jump, we can infer that the *rabbit* is most-likely jumping and not the hat (although, there is a possibility the hat could have jumped).

A.5.2 Classical NLP

The field of NLP can be partitioned into two eras: the classical NLP era, and the machine learning NLP era [36]. Machine-learning approaches have dominated the field post-2012 [38].

Classical NLP relies on a number of rule-based, dictionary-based and statistical approaches. The classical techniques focus on forming *pipelines* of operations that can be chained together [36], where each step in the pipeline produces some structure for the next item in the chain.

Harpaz et al. describe the following sub-tasks related to the extraction of adverse events using text-mining:

Segmentation Splitting the document into sections and sentence boundaries.

Tokenization Splitting sentences into words and punctuations.

Part of speech tagging Assigning grammatical labelling to the individual sentences (e.g. noun, verb, etc)

Parsing Inferring the grammatical structure from the labels and surrounding context

Named Entity Recognition (NER) Detecting and tagging structures of interest (e.g. AFINITOR is tagged as a DRUG)

Negation Detection Detecting negative phrases (e.g. "AFINITOR did not cause MYOCARDIAL INFRACTION")

Word Sense Disambiguation (WSD) Disambiguate similar words (e.g. "ankle strain" vs. "bacterial strain")

Temporal Inference Inferring temporal hierarchy (e.g. "adverse event occurred after prescription of drug")

Relation Detection Detecting if any of the entities are related to each other (e.g. "drug A treats disease B", "drug A induces disease B")

A.5.3 Clinical NLP

Clinical NLP is the application of NLP techniques to extract information in a clinical setting. There are a number of unique challenges in regard to clinical NLP [38, 39]:

- Unavailability of data due to privacy concerns or *data silos*
- Lack of reliable annotations because it is expensive
- Lack of standard evaluation metrics
- Lack of reproducibility

Shared task challenges solve some of these problems by providing researchers with anonymized, well annotated clinical data and a standard evaluation metrics. This gives researchers a way to develop state-of-the-art methods and compare their performance.

A.6 Machine Learning for NLP

Contrary to the classical approaches, machine learning based approaches to NLP attempt to have no assumptions about the underlying language model that are encoded by humans, instead it tries to learn these features through labelled and unlabelled data. It usually consists of two phases:

1. The training phase where the algorithm is shown a number of examples for learning.

2. The prediction phase where the algorithm is expected to predict the results for inputs it has not seen.

A.6.1 Neural Network

A neural network is a computation model inspired by the biology of the brain: a neural node collects one or more signals and triggers a signal if it exceeds a given threshold. It consists of a graph of nodes that pass information through the graph and attempts to minimize a loss function that models how well the inputs match the prediction.

The most basic unit of a neural network is a node.

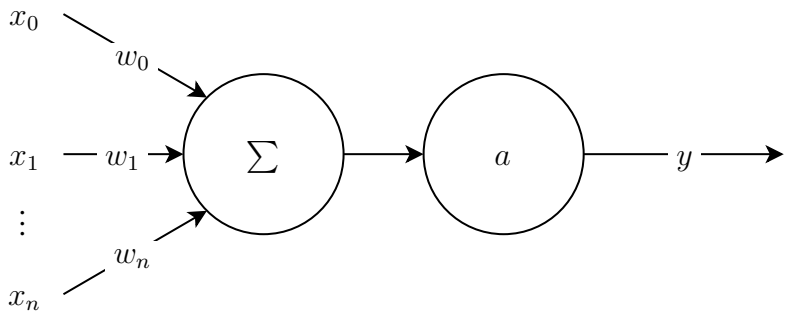

Figure A.13: Computational graph of a basic neuron model

The summation operation performs a weighted combination of the inputs $\mathbf{x} = (x_0, x_1, \cdots, x_n)$ and weights $\mathbf{w} = (w_0, w_1, \cdots, w_n)$. The function a models the activation of the neuron.

$$y = a(\mathbf{w} \cdot \mathbf{x}) \tag{A.13}$$

A simple node such as in fig. A.13 can perform an arbitrary linear classification task if we choose an appropriate activation function. The simplest

activation function can be a step function:

$$a(\Omega) = \begin{cases} 1 \text{ if } \Omega > 0 \\ 0 \text{ otherwise} \end{cases} \tag{A.14}$$

A neuron with such a model is called a *perceptron* and it is able to perform a simple binary classification when trained given input data $\mathbf{x} \in \mathbb{R}^d$ and expected output $\mathbf{y} \in \{0, 1\}$. After training, the weights $\mathbf{w}$ represent the normal to the hyperplane that separates the two classes. However, this hyperplane is often centered around the origin and cannot separate classes which lie far from the origin, in order to mitigate this issue we often introduce a bias term by setting $x_0 = 1$ and $w_0 = b$, thus is it more appropriate to write:

$$y = a(\hat{\mathbf{w}} \cdot \hat{\mathbf{x}} + b) \tag{A.15}$$

Where $\hat{\mathbf{x}} = (x_1, x_2, \cdots, x_n)$ and $\hat{\mathbf{w}} = (w_1, w_2, \cdots, w_n)$. However, for the sake of succinctness and generality, the introduction of the bias term is implied in the rest of this section.

These units can be stacked together to form complex networks that can perform more complex classification tasks, taking the output of one layer as the input to the next:

$$\mathbf{h} = a(\mathbf{w_0}^{\mathrm{T}} \cdot \mathbf{i}) \tag{A.16}$$

$$\mathbf{o} = a(\mathbf{w_1}^{\mathrm{T}} \cdot \mathbf{h}) \tag{A.17}$$

Where $\mathbf{i} \in \mathbb{R}^{d \times k}$ is the input to the network, $\mathbf{o} \in \mathbb{R}^{n \times k}$ is the output of the network, $\mathbf{w_0} \in \mathbb{R}^{d \times m}$ are the weights mapping from input to the intermediate layer $\mathbf{h}$[5], and $\mathbf{w_1} \in \mathbb{R}^{m \times n}$ are the weight mapping from the intermediate layer $\mathbf{h}$ to the output $\mathbf{y}$. m and n represent the number of units in each layer.

Equations (A.16) to (A.17) illustrate the importance of the activation function a being non-linear. Assume the activation is linear, i.e. it satisfies the additive and homogenous (of degree 1) property:

$$a(\Omega + \Delta) = a(\Omega) + a(\Delta) \tag{A.18}$$

$$a(\alpha \cdot \Omega) = \alpha \cdot a(\Omega) \quad \forall \alpha \tag{A.19}$$

We can simplify eqs. (A.16) to (A.17) as:

[5] Known as the *hidden* layer because the parameters and output are hidden and never "observed" directly

$$\mathbf{o} = a(\mathbf{w_1}^{\mathrm{T}} \cdot a(\mathbf{w_0}^{\mathrm{T}} \cdot \mathbf{i})) \tag{A.20}$$
$$= a(\mathbf{w_1}^{\mathrm{T}} \cdot \mathbf{w_0}^{\mathrm{T}} \cdot a(\mathbf{i})) \tag{A.21}$$
$$= \mathbf{w_1}^{\mathrm{T}} \cdot \mathbf{w_0}^{\mathrm{T}} \cdot a(a(\mathbf{i})) \tag{A.22}$$
$$= \mathbf{W} \cdot a(a(\mathbf{x})) \tag{A.23}$$

The resulting matrix $\mathbf{W}$ is a simple linear transformation over the mapped inputs.

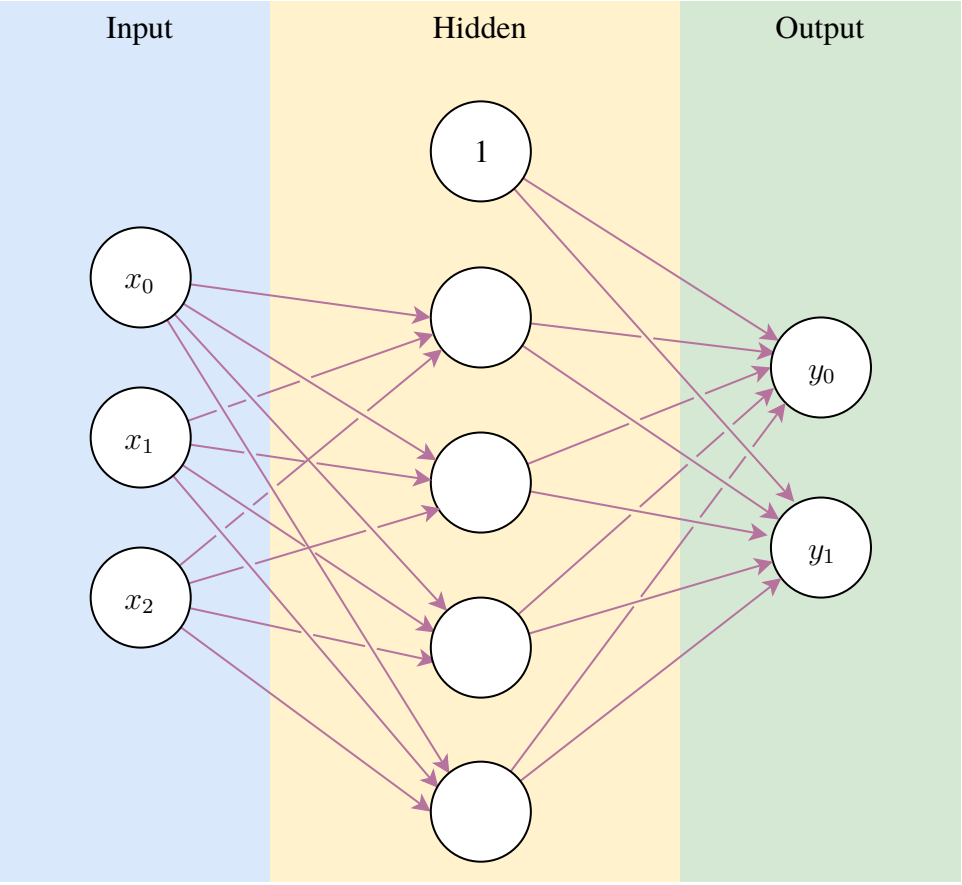

Figure A.14: Simple single-layer neural network with $k = 1$, $d = 2$, $m = 4$, $n = 2$ and $o = 1$

Networks of the type illustrated in fig. A.14 are known as feed-forward networks because the flow of information is forward through the network: Input $\rightarrow$ Process $\rightarrow$ Output. Such a network can approximate any function: $|a(\mathbf{w} \cdot \mathbf{x}) - f(\mathbf{x})| < \epsilon$ for any $\epsilon > 0$ and is known as the Universal Approxi-

mation Theorem [40]. We can approximate any function arbitrarily closely by adding more hidden units.

A single-layer neural network is easy to train because they are not susceptible to gradient vanishing/exploding issues that deep neural networks suffer from. Deeper networks (more than a few layers) require architectural changes in the neural network formulation that allow us to preserve gradients across multiple operations. Since single-layer neural networks allow us to approximate any function, they are still used for dimensionality reduction in the final layers of deeper architectures.

The network in fig. A.14 can be simplified into a schematic diagram that allows us to represent complex and deeper networks where the intricacies of the layers are hidden, this is illustrated in fig. A.15 where FCL represents a Fully Connected Layer representing a layer where all the inputs from the previous layer are connected to all the nodes of the next layer.

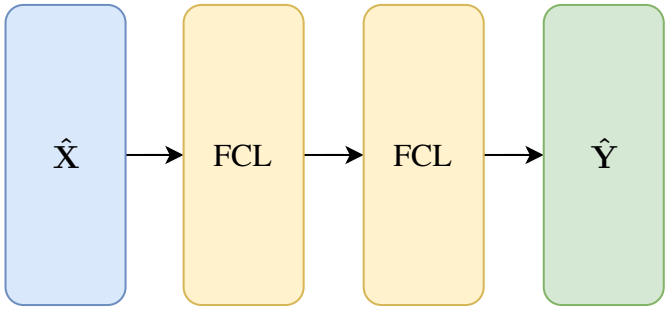

Figure A.15: Schematic diagram of a two-layer network

While feed-forward neural networks have been successfully used to solve a number of machine learning tasks, they are best suited to tasks where the input and outputs are of fixed dimensions. They are not naturally suited for *sequence modelling*. Textual information is often variable length and sequential (i.e. in left-to-right scripts, the contents on the left depend on the contents on the right of the current word).

A.6.2 Embeddings

In order for textual information to be useful for machine learning, we need some way of converting text into features that we can pass into the network. Textual data are made of characters, which are grouped into words, sentences and so on; the first decision is deciding the level of tokenisation.

Choosing to tokenize at the character level is often too fine-grained since it

does not provide us with any additional semantic information: the characters themselves have no meaning on their own.

One level higher, words often are the smallest unit of a language that still carry some meaning. As a result, many deep-learning techniques choose to encode words into some feature space for machine learning. The result of this encoding procedure is called an Embedding, and in its simplest form is just a lookup-table that maps a word in a vocabulary into some n-dimensional space: $v \in \mathcal{V} \mapsto \mathbb{R}^n$.

The simplest possible embedding is the one-hot encoding, where, for each $v_i \in \mathcal{V}$ there is an embedding matrix $\mathbb{E} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ such that:

$$\mathbb{E}_{ij} = \begin{cases} 1 \text{ if } i = j \\ 0 \text{ otherwise} \end{cases} \tag{A.24}$$

It is clear to see that the embedding matrix $\mathbb{E}$ will be a very sparse matrix and will scale quadratically with the vocabulary size $|\mathcal{V}|$. For each new word in the vocabulary, the one-hot encoding introduces a new dimension. The inefficiency of the one-hot can be illustrated with a simple example: Given the set $\mathcal{V} = \{\text{cat}, \text{dog}, \text{bird}\}$, a one-hot encoding would result in the embedding matrix:

$$\mathbb{E} = \left[\begin{array}{ccc|c} 1 & 0 & 0 & \text{cat} \\ 0 & 1 & 0 & \text{dog} \\ 0 & 0 & 1 & \text{bird} \end{array}\right] \tag{A.25}$$

We can easily find a more efficient and compact embedding:

$$\mathbb{E} = \left[\begin{array}{c|c} 0 & \text{cat} \\ 1 & \text{dog} \\ 2 & \text{bird} \end{array}\right] \tag{A.26}$$

However, this embedding has issues because it introduces an order between the concepts that have no basis in reality. In order to mitigate this, embeddings are often multi-dimensional where each dimension encodes some feature-space encoding the closeness of concepts in the scope of that particular feature (which may or may not have an interpretation).

word2vec

word2vec is an unsupervised learning algorithm that learns to align similar words together in a vector space. In its essence it is an auto-encoder which takes in a one-hot encoded feature vector and learns to embed it into a lower

dimension. The decoder is discarded and the embedded hidden state is used as the embedding matrix[15].

word2vec can be trained in one of two modes:

Continuous Bag of Words In this mode, the network learns to predict the current word given some context within a window.

Skip-gram In this mode, the network predicts the context given a word.

GloVe

GloVe stands for Global Vectors for Word Representations and is another unsupervised learning algorithm for mapping words to vectors. GloVe uses co-occurrence statistics to calculate the word vectors, the intuition is that words that are similar in context appear together across a large dataset. For example, the word *ice* is more likely to co-occur with the word *solid* than with the word *steam* or *fashion*.

GloVe has been shown to outperform word2vec in certain use cases[41].

fastText

fastText[6] is another embedding method from Facebook that uses morphological and sub-word features to handle out of vocabulary words. When encountering out-of-vocabulary words, it tries to break the word into smaller chunks and align the chunks, for e.g. given that the word ending in *-itis* such as *pancreatitis* is in the dictionary but not the word *stomatitis*, fastText will return vectors for *pancreatitis* and *stomatitis* that are fairly close [42].

ELMo

Embeddings from Language Model (ELMo) is another word embedding technique that uses a more complex model to learn word representations. The previous models do not take context into account and map each unique word in the vocabulary to the same vector representation. For e.g. , the phrase *bank* in the *river bank* is not the same as *bank deposit*, the words are the same but the semantic encoding is different based on context.

ELMo uses a two-layer bidirectional language model along with a character Convolutional Neural Network (CNN) whose final states are concatenated into a 1024 dimensional feature vector [14].

[6] `https://fasttext.cc/`

A.6.3 Recurrent Neural Network (RNN)

RNN are a form of sequential[7] neural networks that can pass information through layers and in the form of sequential input. As mentioned earlier in appendix A.5, understanding languages require a form of memory or context. RNNs allow this by maintaining a state vector that is propagated along with the inputs into the next sequence-step and used to calculate the new state [21].

$$h_t = \sigma(\mathbf{W}_{xh}x_t + \mathbf{W}_{hh}h_{t-1}) \tag{A.27}$$
$$y_t = \mathbf{W}_{hy}h_t \tag{A.28}$$

Recurrent networks can be illustrated as a loop in the network architecture.

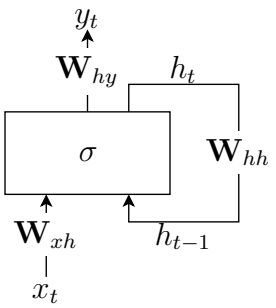

Figure A.16: RNN architecture

The loop can be *unrolled* to form a feed-forward network if we lay it out along the sequence dimension for all elements in the sequence (fig. A.17).

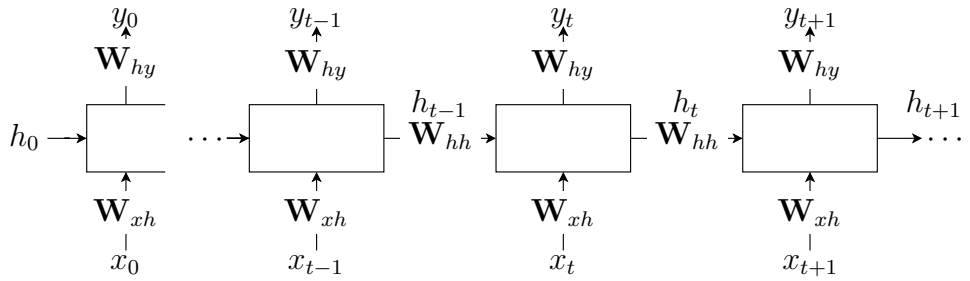

Figure A.17: RNN architecture unrolled

[7]They can also be seen as temporal networks due to the inherent sequential nature of time. Although most literature denote the sequence steps as a temporal variable t, this does not need to be the case. I adopt the same convention here.

We can now see the recurrent nature of the architecture: the basic unit of the network remains the same and we are continuously passing the inputs x_t at every sequence step into the network. The state is preserved in h_t for all steps t. It is easy to see why this is a sequential model, each h_t depends on the previous h_{t-1}:

$$h_t = \sigma(\mathbf{W}_{xh}x_t + \mathbf{W}_{hh}h_{t-1}) \tag{A.29}$$
$$h_{t-1} = \sigma(\mathbf{W}_{xh}x_{t-1} + \mathbf{W}_{hh}h_{t-2}) \tag{A.30}$$
$$h_{t-2} = \sigma(\mathbf{W}_{xh}x_{t-2} + \mathbf{W}_{hh}h_{t-3}) \tag{A.31}$$
$$h_{t-3} = \sigma(\mathbf{W}_{xh}x_{t-3} + \mathbf{W}_{hh}h_{t-4}) \tag{A.32}$$

Let $\hat{\mathbf{X}}_t = \mathbf{W}_{xh}x_t$, substituting into eq. (A.29) and expanding over the last few steps, we get:

$$\begin{aligned} h_t = \sigma(&\hat{\mathbf{X}}_t \\ &+ \mathbf{W}_{hh}\sigma(\hat{\mathbf{X}}_{t-1} \\ &+ \mathbf{W}_{hh}\sigma(\hat{\mathbf{X}}_{t-2} \\ &+ \mathbf{W}_{hh}\sigma(\hat{\mathbf{X}}_{t-3} \\ &+ \mathbf{W}_{hh}h_{t-4})))) \end{aligned} \tag{A.33}$$

We can see that we reuse the same matrix $\mathbf{W}_{hh}$ for the hidden layer for each step. If $\mathbf{W}_{hh} < 1$, we encounter the vanishing gradients and $\mathbf{W}_{hh} > 1$ results in the exploding gradients problem. This makes it extremely difficult to train RNN and retain past information [43].

A.6.4 Long Short-Term Memory (LSTM)

LSTMs are a type of RNN cell that attempts to reduce the vanishing/exploding gradients problem by introducing *gating* mechanisms. It introduces a memory mechanism called the *cell state* which can be manipulated through gates named as: *input*, *forget* and *output* gates [44]. The hidden state of the network is a function of its cell state.

$$i_t = \sigma(\mathbf{W}_{ii}x_t + \mathbf{b}_{ii} + \mathbf{W}_{hi}h_{t-1} + \mathbf{b}_{hi}) \tag{A.34}$$
$$f_t = \sigma(\mathbf{W}_{if}x_t + \mathbf{b}_{if} + \mathbf{W}_{hf}h_{t-1} + \mathbf{b}_{hf}) \tag{A.35}$$
$$o_t = \sigma(\mathbf{W}_{io}x_t + \mathbf{b}_{io} + \mathbf{W}_{hf}h_{t-1} + \mathbf{b}_{ho}) \tag{A.36}$$
$$g_t = \tanh(\mathbf{W}_{ig}x_t + \mathbf{b}_{ig} + \mathbf{W}_{gh}h_{t-1} + \mathbf{b}_{hg}) \tag{A.37}$$
$$c_t = f_t \circ c_{t-1} + i_t \cdot g_t \tag{A.38}$$
$$h_t = o_t \circ \tanh(c_t) \tag{A.39}$$

Where c_t is the cell state, x is the input to the cell, and h_t is the hidden state; $\circ$ represents the Hadamard (element-wise) product, $\sigma(\cdot) \in [0, 1]$ is the sigmoid function defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{A.40}$$

i_t, f_t and o_t represent the input, forget and output gates respectively and control the flow of the cell state onto the next step.

A.6.5 Gated Recurrent Unit (GRU)

GRUs is another type of RNN, that aims to solve the same problems that LSTMs solve, with a simpler model. The intuition behind a GRU unit is that the *forget* and *input* can be combined into a single *update* gate; and the hidden state can be simplified to not depend on an auxiliary cell state.

With these changes, a GRU cell can be defined as:

$$u_t = \sigma\mathbf{W}_{ux}x_t + \mathbf{W}_{uh}h_{t-1} + \mathbf{b}_u \tag{A.41}$$
$$r_t = \sigma\mathbf{W}_{rx}x_t + \mathbf{W}_{rh}h_{t-1} + \mathbf{b}_r \tag{A.42}$$
$$h_t = u_t \circ h_{t-1} \tag{A.43}$$
$$+ (1 - u_t) \circ \tanh(\mathbf{W}_{hx}x_t + \mathbf{W}_{hh}(r_t \circ h_{t-1}) + \mathbf{b}_h) \tag{A.44}$$

However, LSTM cells have been shown to consistently outperform GRU cells in language modelling tasks, despite being faster to train (due to the reduced number of parameters) [45].

A.6.6 Bidirectional Recurrent Neural Network (BiRNN)

A major issue of RNN based architectures is the fact that the state at each step t depends on the previous step $t-1$. For sequential modelling tasks, this means that the model only has information to the left of the current word. The intuition behind bidirectional RNN is that we can construct better predictions if we have information about both the upstream ($t-k$) and downstream ($t+k$) inputs.

This is achieved by running the input sequence first from $t=0$ to $t=T$ (assuming T is the length of the sequence), and running the input sequence from $t=T$ to $t=0$ on a separate RNN cell.

We call the forward RNN states $\overrightarrow{h_t}$ and the backward RNN states $\overleftarrow{h_t}$. These two vectors are then concatenated to form a bidirectional feature vector:

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \tag{A.45}$$

A.6.7 Conditional Random Field (CRF)

A Conditional Random Field (CRF) is an undirected probabilistic graphical model similar to Maximum Entropy Markov Model (MEMM) that was designed to solve the *label bias problem*, where the transitions from a given state depend only on the outgoing transitions instead of global transition (including all following transitions), this results in state with only a single transition transferring all their mass to the next state [46]. This results in state transitions that are the optimal locally but not globally in the scope of all transitions. CRFs attempt to solve this by normalizing the state transition probability by the global transition probabilities.

Given the sequence of random variables $\mathbf{X}$ conditioned on the hidden states $\mathbf{Y}$, CRF is defined as a graph $G=(V,E)$ where $\mathbf{Y}$ is indexed by the vertices of G: $\mathbf{Y}=(\mathbf{Y}_v)_{v\in V}$. The variables in $\mathbf{Y}$ follow the Markov property within the graph defined as: $p(\mathbf{Y}_v|\mathbf{X},\mathbf{Y}_w; w\neq v) = p(\mathbf{Y}_v|\mathbf{X},\mathbf{Y}_w; w\sim v)$ where $w\sim v$ represents neighbourhood in the state graph [46]. A graphical representation is presented in fig. A.18.

CRFs for sequence modelling are called Linear-chain CRF and are acyclic (fig. A.19), for the remainder of this thesis CRF refers to Linear-chain CRF. Note that the chain in fig. A.19 is similar to the field in fig. A.18 except for the edge between state X_4 and X_5. The Markov property maintains a linear dependency between the states in the chain.

Given an input sequence $\mathbf{x}=(\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_k)\in\mathcal{X}$ and its predictions

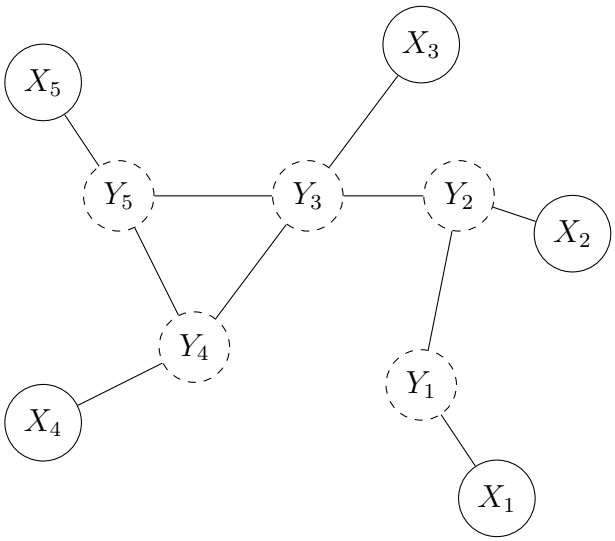

Figure A.18: Graphical Representation of CRF

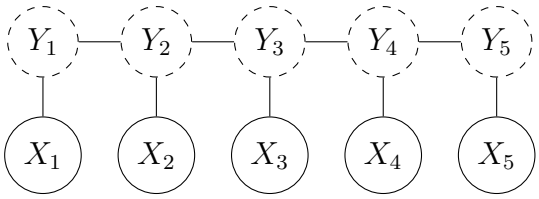

Figure A.19: Linear Chain CRF

$\mathbf{y} = (y_1, y_2, \ldots, y_k) \in \mathcal{Y}$, where $\mathcal{Y}$ is a finite set. We define the conditional probability $p(\mathbf{y}|\mathbf{x})$ with a log-linear model [47]:

$$p(\mathbf{y}|\mathbf{x}) = \frac{\exp\{\phi(\mathbf{x}, \mathbf{y})\}}{\sum_{\mathbf{y}'} \exp\{\phi(\mathbf{x}, \mathbf{y}')\}} \tag{A.46}$$

The function $\phi : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}^d$ is known as the *feature function*. For logistic regression, we can define $\phi(\mathbf{x}, \mathbf{y}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathbf{b}$. However, that would not give us any discriminative power constrained by the state transitions; we need to formulate some form of dependency between the state transitions. This is possible by introducing a state transition matrix $\mathbb{A} \in \mathbb{R}^{(m+2)\times(m+2)}$ where m is the number of hidden states (we introduce special states `START` which no

state can *transition to* and `STOP` which cannot *transition to* any other state).

$$\mathbb{A} = \begin{array}{c|cccccc} & \texttt{START} & S_0 & S_1 & \dots & S_m & \texttt{STOP} \\ \hline \texttt{START} & -\infty & w_{1,2} & w_{1,3} & \dots & w_{1,m} & -\infty \\ S_0 & -\infty & w_{2,2} & w_{2,3} & \dots & w_{2,m} & w_{2,m+1} \\ S_1 & -\infty & w_{3,2} & w_{3,3} & \dots & w_{3,m} & w_{3,m+1} \\ \vdots & -\infty & \vdots & \vdots & \ddots & \vdots & \vdots \\ S_m & -\infty & w_{m,2} & w_{m,3} & \dots & w_{m,m} & w_{m,m+1} \\ \texttt{STOP} & -\infty & -\infty & -\infty & -\infty & -\infty & -\infty \end{array} \tag{A.47}$$

We can now define a feature function for linear-chain CRF with dependencies between the previous and next state:

$$\phi(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{k} U(\mathbf{x}_i, y_i) + \underbrace{\sum_{i=0}^{k} \mathbb{A}(y_{i-1} \to y_i)}_{\text{Linear Dependency}} \tag{A.48}$$

Where $U(\mathbf{x}_i, y_i)$ is the emission score estimate that measures how likely is the prediction given the input $\mathbf{x}_i$. However, this estimate does not take into account valid transitions, e.g. `STOP` $\to$ `START`. The transition matrix helps penalize bad transitions: for e.g. $\mathbb{A}(\texttt{STOP} \to \texttt{START}) = -\infty$ adds a large negative weight to the estimation.

During training, the algorithm is learning the weights of the transition matrix that best minimizes the negative log-likelihood of $p(\mathbf{y}|\mathbf{x})$.

Finally, what sets CRF apart from MEMM is the denominator of the log-linear model, known as the *partition function*. For MEMM, this is simply an estimate over all states $y \in \mathcal{Y}$, however CRF define it to be an estimate over all states, over the entire sequence $s \in \mathcal{Y}^m$ (the global normalization property alluded to earlier).

A.6.8 BERT

Bidirectional Encoder Representations from Transformers (BERT) [48] is a state-of-the-art language model that is, as its name suggests built on top of the Transformer model [49]. One of the issues of LSTMs and GRUs is that, while they do maintain some sort of memory, its size is fixed. As a result, performance of RNNs across longer sequences drops drastically [16]. As a remedy to this issue, Bahdanau, Cho, and Bengio proposed the *attention* mechanism, which introduces an additional layer, called the *attention layer* which learns to

selectively weigh and focus on specific parts of the sequence. Bahdanau, Cho, and Bengio formulated attention in the form of an encoder-decoder model for Neural Machine Translation (NMT) and is often referred to as *additive attention*, it requires both the encoder and decoder state to calculate an attention score.

However, it was quickly realized that attention mechanism could also benefit in language modelling. Luong, Pham, and Manning formulate additional attention scoring functions, including *dot product attention* (as well as many other varieties), which only required the states of the encoder. This meant that attention could also be used within a language modelling scenario without a decoder [30].

The other issue of LSTMs and GRUs is that they are sequential and each state is dependent on some previous state (see appendix A.6.3), this makes it difficult to train the models in parallel. Vaswani et al. proposed the Transformer model with an idea called *Multi-head Self Attention*. Here, self-attention refers to the attention model applied on the input sequence itself (that measures how similar a given concept is to every other), multi-head refers to a parallel processing of all sequences (at the expense of a larger memory footprint). Figure A.20 illustrates a schematic diagram of the Transformer encoder block, the original model proposed by Vaswani et al. used 6 of these stacked blocks. Positional encodings are necessary because the parallel nature of the training loses the sequential information about the sequence.

The standard BERT pre-trained model is provided in two sizes, the smaller one of which (bert–base) stacks 12 transformer encoders (thus 12×6 encoder blocks) with 12 heads each. It is trained in two modes: the masked language mode and next sentence prediction mode, both of which are unsupervised. In the masked-language mode 15% of the words in the training data are masked with a special `[MASK]` token and the model learns to predict those words. In the next sentence prediction mode, the model is provided two candidate sentences which may or may not follow each other and the network learns to predict whether sentence B follows sentence A [48]. BERT contains a few special tokens: every classification task starts with the special token `[CLS]` and each sentence fragment is separated by `[SEP]`. The feature vector emitted by BERT (per token) can then be used for downstream tasks (e.g. classification using a dense layer) in a process called *fine-tuning*.

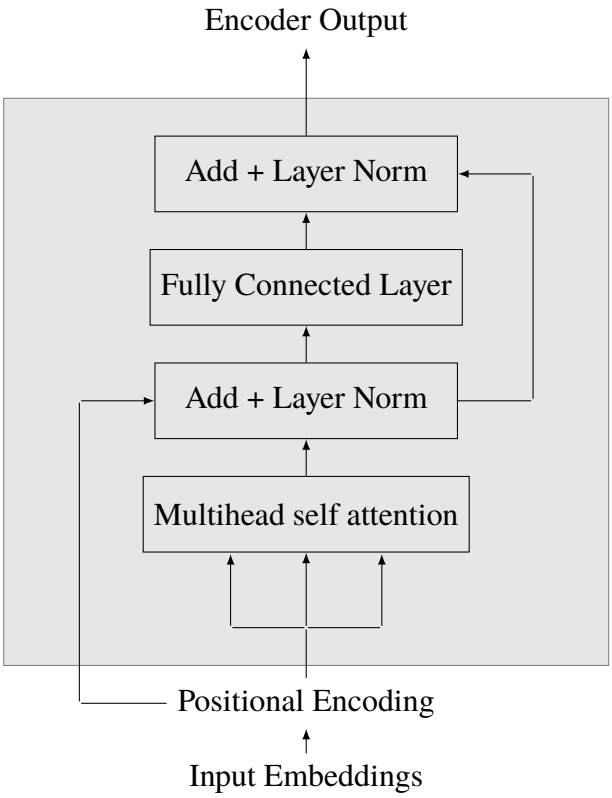

Figure A.20: Transformer Encoder Architecture

A.7 Information Retrieval

Information Retrieval (IR) is the process of finding and scoring documents by relevancy. The object to be retrieved is called a *document* and the intent is called a *query* [50]. The simplest information retrieval process is to scan all documents with the query, however it not scalable as the number of documents grow, and it is not efficient to scan through all documents for every query. In order to mitigate this issue, IR systems build *indexes* which are precomputed statistics, heuristics and data structure that allow quick matching of the query to the document.

IR systems often use NLP techniques to build their indexes. It is common for IR systems to remove stop words from queries and documents, perform stemming, lemmatization and n-gram tokenisation into a tree data structure. When performing a query the IR system traverses this tree to find the most likely candidates [50].

However, searching is only part of the responsibilities of an IR system. The

other half is ranking the results. Ranking involves boosting the most likely candidates to the top of the result list, and many models have been developed for it.

One of the simplest models is Term Frequency — Inverse Document Frequency (TF–IDF). It relies on the assumption that the number of query–related words in a document is directly proportional to its relevancy to the query, while at the same time words that are common across many documents (e.g. "a", "the") are not very discriminative and should not reflect in the ranking. It contains two terms, the tf_t term which counts the number of times a term t occurs across a particular document $d \in \mathcal{D}$. The second term is the inverse document frequency and defined as:

$$\text{idf}_t = \log \frac{N}{\text{df}_t} \tag{A.49}$$

Where N is the number of documents in $\mathcal{D}$ and df_t is the number of documents containing the term t. TF–IDF is defined as a product of these two terms:

$$\text{tfidf}_t = \text{tf}_t(d \in \mathcal{D}) \cdot \text{idf}_t(\mathcal{D}) \tag{A.50}$$

It is easy to see that, for common words like "the", $N \approx \text{df}_t$ resulting in $\text{idf}_t \approx 0$.

Another more recent probabilistic ranking model is knows as BM25, where BM stands for *Best Match*. It derives from the Probability Ranking Principle [51] which states that if we rank documents based on $p(\mathbf{r}|\mathbf{q}, \mathbf{d})$ where $\mathbf{r} \in \{0, 1\}$ is a binary variable representing whether the documents $\mathbf{d}$ are relevant to the queries $\mathbf{q}$ or not; then the ranking is the best that can be obtained.

The BM25 model has the general form of the TF–IDF model, but it redefines how the TF and IDF terms are calculated, based on probabilistic assumptions [51]. The particular implementation used in this project redefines the inverse document frequency as [52]:

$$\hat{\text{idf}}_t = \log \left(1 + \frac{N - \text{df}_t + 0.5}{\text{df}_t + 0.5} \right) \tag{A.51}$$

The constant 1 is added to the argument of the logarithm to guarantee that $\hat{\text{idf}}_t > 0 \; \forall \; \text{df}_t$, without which $\hat{\text{idf}}_t$ would be negative if a term appears in more than half of the documents [50].

For the TF term, BM25 introduces tunable parameters k and b. The traditional term frequency used in TF–IDF is unbounded, the score is proportional

to the number of times it appears in the document. BM25 aims to introduce the parameter k that bounds this score:

$$\hat{\text{tf}}_t(d \in \mathcal{D}) = \frac{(k+1) \cdot \text{tf}_t(d)}{k + \text{tf}_t(d)} \tag{A.52}$$

We can see that $\lim_{\text{tf}_t \to \infty} \hat{\text{tf}}_t = k + 1$; the term frequency score asymptotically approaches $k+1$ and the contribution of the term t decreases as the term frequency increases.

The second parameter b introduces document length normalization, a document about "cardiac arrest" has less relevance if it is 10 words longs than if it is 2 words long (the two words being "cardiac arrest").

$$\hat{\text{tf}}_t(d \in \mathcal{D}) = \frac{(k+1) \cdot \text{tf}_t(d)}{k \cdot \left((1-b) + b \cdot \|d\| \cdot \|\bar{\mathcal{D}}\|^{-1}\right) + \text{tf}_t(d)} \tag{A.53}$$

Where $\|\bar{\mathcal{D}}\|$ is the average document length $\forall d \in \mathcal{D}$ and $\|d\|$ is the document length of the current document. We can see that if we set $b = 0$, the document length normalization is disabled. Setting $b = 1$ enables the document length normalization fully.

Finally, the BM25 score is calculated as a product of these scores[8]:

$$\text{BM25} = \hat{\text{idf}}_t \cdot \hat{\text{tf}}_t \tag{A.54}$$

Typical values for these parameters are $k \in [1.2, 2]$ and $b = 0.75$ [50].

A.7.1 Learning to Rank

As stated earlier, the probability ranking principle states that for any given query q_i, if we can sort $p(\mathbf{r}_i | q_i, \mathbf{d}) \forall d \in \mathcal{D}$ in decreasing order, then the system performance is the best that can be obtained. Due to the probabilistic formulation of ranking, we can use machine learning to learn to rank matches.

To train such a system, the system first performs traditional IR techniques to to retrieve at most k documents from the index for a query in our training set q_i. We can now generate a relevancy matrix $\mathbf{r} \in \{0, 1\}^{k \times 1}$ using the correct labels for the true (q_i, d_i) pairs in our training set. A machine learning system is trained on these datasets that optimally fits all training pair datasets (q_i, d_i) which can be used at testing time to re-rank entries.

[8]There is an additional term: the query length normalization [50] parameter, but the implementation used in this project does not use it.

A.8 Related work

Traditional approaches to NLP has been mostly rule-based or use statistical techniques [53]. However, recently Deep Neural Networks and new techniques such as word embeddings [15] have steered NLP into data-driven pastures and deeper understanding of language models. For the purpose of this document, the former (rule-based and statistical technique) will be called the text-mining approach, and the latter will be called data-driven NLP.

Harpaz et al. [35] note that most applications of biomedical text mining for Named Entity Recognition (NER) rely on a dictionary based lookup, but this requires customizing the dictionary for maximum effectiveness. A rule-based approach might capture phrases that match the pattern "`[DRUG]` induces `[DISEASE]`" where `[DRUG]` and `[DISEASE]` are labels tagged by the NER step. A more sophisticated approach might use Part Of Speech (POS) tags, for e.g. matching only proper nouns and correlating it with a disease database.

Duke and Friedlin [54] use text mining techniques to extract AEs from SPLs using an application written in Java called Structured Product Label Information Coder and ExtractoR (SPLICER). It consists of three modules:

1. SPL Parser: Removes XML tags and deconstructs paragraphs
2. AE Extractor: Extracts all AEs using punctuation and table information
3. MedDRA Mapper: Maps matched AEs to MedDRA dictionary

Similarly, Smith et al. [55] use regular expression string matching against various data sources, and integrating them to match drugs against clinical manifestations in order to build a drug evidence database.

Iqbal et al. [56] develop a "semantically-enriched pipeline" for extracting ADRs from Electronic Health Record (EHR) that uses rule-based NLP procedure to detect ADRs in psychiatric health records. However, their approach is limited because they only consider 19 ADRs specific to antipsychotic and antidepressant medications.

More recently, approaches based on Deep Neural Networks have been developed by Dandala, Joopudi, and Devarakonda, Yang et al. [23, 57, 58]. A key difference between the traditional NLP approach (Segmentation, Tokenisation, etc.) and the data-driven approach is that the data-driven approach requires much less pre-processing of data (such as stop-word removal, part of speech tagging) and the feature engineering phase is often unsupervised (i.e. it learns the features of the language independently) at the cost of an increase in complexity. All three [23, 57, 58] approaches use a Bidirectional Long

Short-Term Memory (BiLSTM) coupled with a CRF. The free-form text to be parsed is passed through the BiLSTM layer both forward and backward. The CRF layer is an undirected graphical model that jointly models the probability of relationship between the output of the BiLSTM layer through sequential inputs. This allows efficient association of inter-term dependencies. All three approaches operate on clinical notes instead of SPLs to handle the noisy format (due to patient history, family history, differential diagnosis) of clinical notes. Wunnava et al. [58] use an additional rule-based layer that rely on traditional NLP techniques.

Most previous work on entity normalization fall into three categories: Information Retrieval/dictionary-based methods, neural network models and hybrid approaches. IR methods generally make use of a full-text search engine with some form of text processing [10, 59]. Hybrid approaches use Information Retrieval with a neural ranking model [19].

Previous attempts

UMC uses a dictionary and rule based system for detection of ADRs developed as a part of a Master's Thesis project [10] whose performance was evaluated on the FDA provided dataset. Its results are summarized in table A.1 labelled *Dictionary*. The dictionary based approach performs stop word removal (such as *a*, *the*, *in*), synonym expansion (e.g. *decrease* $\rightarrow$ *lower*, *convulsion* $\rightarrow$ *seizure*), stems words (e.g. *hepatitis* $\rightarrow$ *hepatit*) and performs word permutations on multi word phrases in order to expand its vocabulary.

Dr. Lucie Gattepaille, my supervisor and Data Scientist at UMC participated in the original challenge and made two submissions, whose scores are also stated in table A.1 labelled *Submission 1* and *Submission 2*.

Submission 1 uses the dictionary based approach to scan and map all ADR in the dataset, then uses a trained BiLSTM with a sigmoid classifier with a predetermined threshold to filter whether the ADR detected by the rule-based system is a valid ADR or not. The embedding is a pre-trained embedding extracted from the *Adverse event* and *warnings and precautions* sections of 24727 trade names from DailyMed (not part of the provided data). The embedding with dimension of 100, a context size of 7, a minimum token count of 10 and downsampling parameter of 1×10^{-3} is generated using the python package `gensim` with the word2vec model in the skip-gram mode. The BiLSTM hidden size is 128 with a dropout of 0.7 and is followed by a fully connected layer with a sigmoid activation function.

Submission 2 uses the same BiLSTM model but on the label text, and does

not use the dictionary method. For the Map problem, it uses a full-text search engine with n-gram and stemming indices with a custom scoring function.

It is noteworthy that the dictionary based approach leads in recall, while Submission 1 leads in precision and Submission 2 has higher F_1 across the different metrics. It is not surprising that the dictionary based approach has a high recall and low precision, as its formulation makes it an unspecific classifier since it tries to match as many ADR are possible.

Submission 1 uses the BiLSTM classifier to filter out many false positive matches from the results of the dictionary based algorithm, this reduction in false positive rate increases the precision.

Submission 2 seems the most balanced (as visible with the highest F_1 across many metrics) with precision similar to Submission 1 (only a few points lower) in the Scan problem. Its performance is comparable to Submission 1 for the Map problem, this is possibly because Submission 1 depends on the dictionary-based algorithm for mapping which is essentially a mini search engine (since it performs many functions that search-engines perform such as stop word removal, stemming and edit-word / Levenstein distance calculations as well as soundex analysis). Submission 2 also has the highest front-office quality which measures how good the evidence for a particular MedDRA query is, the search engine with the n-gram, stemming and basic filters provide a lot of features in the search space which is not possible with the dictionary approach which does not consider sub-word matches.

	Dictionary			Submission 1			Submission 2		
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1
Exact mention match - unweighted	0.31	**0.66**	0.42	**0.71**	0.58	0.64	0.70	0.64	**0.66**
Exact mention match (discontinuous) - unweighted	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Overlap mention match - unweighted	0.34	**0.74**	0.47	**0.78**	0.64	**0.70**	0.72	**0.66**	0.69
Exact mention match - weighted	0.65	**0.76**	0.70	**0.88**	0.65	0.75	0.87	0.70	**0.78**
Exact mention match (discontinuous) - weighted	**0.50**	0.28	0.36	**0.50**	0.23	0.32	**0.50**	**0.36**	**0.42**
MedDRA Retrieval (macro-averaged Label scope)	0.45	**0.71**	0.54	**0.78**	0.63	**0.69**	0.71	0.69	**0.69**
MedDRA Retrieval (macro-averaged Section scope)	0.39	**0.64**	0.43	**0.71**	0.50	**0.53**	0.58	0.52	0.51
	Label	Section		Label	Section		Label	Section	
Front Office Quality	0.85	0.78		0.92	0.90		**0.95**	**0.93**	

Table A.1: Results of Dictionary based matching, the first and second submissions; bold figures highlight highest precision, recall, F_1 across all models

Appendix B

Trained Model Parameters

ID	Git hash	H	N
bilstm-crf-char-dailymed			
0631ef1a-644e-11ea-9072-b0c090be5a88	e2169d8	300	5
300beb22-6321-11ea-a9b9-b0c090be5a88	18f2c3f	100	3
39c120be-646e-11ea-a9b9-b0c090be5a88	e2169d8	1024	1
51aeee1e-6303-11ea-a9b9-b0c090be5a88	18f2c3f	100	1
545ef81c-62c1-11ea-a9b9-b0c090be5a88	18f2c3f	8	3
98fb22d8-6380-11ea-a9b9-b0c090be5a88	18f2c3f	300	3
9aa41bd8-62e1-11ea-a9b9-b0c090be5a88	18f2c3f	8	5
a469649c-6362-11ea-a9b9-b0c090be5a88	18f2c3f	300	1
c8a31b54-64b2-11ea-a9b9-b0c090be5a88	e2169d8	1024	5
e3f8e078-62a2-11ea-9a91-b0c090be5a88	18f2c3f	8	1
f8604c76-6340-11ea-a9b9-b0c090be5a88	18f2c3f	100	5
ff93f21c-648b-11ea-a9b9-b0c090be5a88	e2169d8	1024	3
elmo-dailymed			
0824b028-66d6-11ea-9465-b0c090be5a88	c8841fa	8	1
09862386-68fd-11ea-8eef-b0c090be5a88	c8841fa	1024	3
09961b5c-6778-11ea-8eef-b0c090be5a88	c8841fa	100	1
25e88b28-68c4-11ea-8eef-b0c090be5a88	c8841fa	1024	1
5c6fefac-67ac-11ea-8eef-b0c090be5a88	c8841fa	100	3
618046e8-670a-11ea-8eef-b0c090be5a88	c8841fa	8	3
6bb06d7a-681a-11ea-8eef-b0c090be5a88	c8841fa	300	1
7cb9beda-6740-11ea-8eef-b0c090be5a88	c8841fa	8	5
91d8e1f8-67e2-11ea-8eef-b0c090be5a88	c8841fa	100	5

ID	Git hash	H	N
98713f38-684f-11ea-8eef-b0c090be5a88	c8841fa	300	3
a8e8d840-6888-11ea-8eef-b0c090be5a88	c8841fa	300	5
cb7e6200-693e-11ea-8eef-b0c090be5a88	c8841fa	1024	5
bilstm-softmax-dailymed			
09a3c6be-61f4-11ea-925c-b0c090be5a88	ced1e4b	300	1
2a530b26-61a7-11ea-925c-b0c090be5a88	ced1e4b	8	5
2bf4ad52-61ca-11ea-925c-b0c090be5a88	ced1e4b	100	3
5a7cd5ba-61ff-11ea-925c-b0c090be5a88	ced1e4b	300	3
72b4c35e-618a-11ea-ba89-b0c090be5a88	ced1e4b	8	1
954dfac2-6195-11ea-925c-b0c090be5a88	ced1e4b	8	3
d1f233e2-6211-11ea-925c-b0c090be5a88	ced1e4b	300	5
eca14014-61db-11ea-925c-b0c090be5a88	ced1e4b	100	5
fd2ff31c-61be-11ea-925c-b0c090be5a88	ced1e4b	100	1
crf-char-glove			
0a38e7f6-59a8-11ea-a9b9-b0c090be5a88	ff2b991	300	3
2e303fde-59c7-11ea-a9b9-b0c090be5a88	ff2b991	300	5
3d55151c-58ef-11ea-a9b9-b0c090be5a88	ff2b991	8	3
41f04398-5a07-11ea-a9b9-b0c090be5a88	ff2b991	1024	3
4f49e0ee-58d2-11ea-84d8-b0c090be5a88	ff2b991	8	1
5c1c6e78-596a-11ea-a9b9-b0c090be5a88	ff2b991	100	5
6c4cff68-59e8-11ea-a9b9-b0c090be5a88	ff2b991	1024	1
867e5676-594b-11ea-a9b9-b0c090be5a88	ff2b991	100	3
87c61638-592e-11ea-a9b9-b0c090be5a88	ff2b991	100	1
b3e2793e-5a2e-11ea-a9b9-b0c090be5a88	ff2b991	1024	5
fb727878-598a-11ea-a9b9-b0c090be5a88	ff2b991	300	1
fbe307b4-590d-11ea-a9b9-b0c090be5a88	ff2b991	8	5
elmo-glove			
0f5a879e-5e69-11ea-8eef-b0c090be5a88	1f24153	1024	1
14333bca-5db9-11ea-8eef-b0c090be5a88	1f24153	300	1
3e6a13b0-5e2c-11ea-8eef-b0c090be5a88	1f24153	300	5
48afb358-5ee6-11ea-8eef-b0c090be5a88	1f24153	1024	5
58639546-5ea3-11ea-8eef-b0c090be5a88	1f24153	1024	3
5b7f9738-5d43-11ea-8eef-b0c090be5a88	1f24153	100	3
9461dd4a-5c97-11ea-8eef-b0c090be5a88	1f24153	8	3
951a089c-5c60-11ea-bae5-b0c090be5a88	1f24153	8	1

ID	Git hash	H	N
9bd94f8c-5df1-11ea-8eef-b0c090be5a88	1f24153	300	3
ea698cdc-5d7c-11ea-8eef-b0c090be5a88	1f24153	100	5
f2a064be-5d0b-11ea-8eef-b0c090be5a88	1f24153	100	1
f3dec11e-5cd0-11ea-8eef-b0c090be5a88	1f24153	8	5
softmax-glove			
1327147a-5b3c-11ea-925c-b0c090be5a88	445fd62	100	1
453d801a-5b6d-11ea-925c-b0c090be5a88	445fd62	300	1
52e5dc22-5b46-11ea-925c-b0c090be5a88	445fd62	100	3
9388cf06-5b77-11ea-925c-b0c090be5a88	445fd62	300	3
97f911d2-5b0b-11ea-8ba6-b0c090be5a88	445fd62	8	1
ab946594-5b56-11ea-925c-b0c090be5a88	445fd62	100	5
b5dd7712-5b15-11ea-925c-b0c090be5a88	445fd62	8	3
b97f6400-5b88-11ea-925c-b0c090be5a88	445fd62	300	5
d89c6d2e-5b25-11ea-925c-b0c090be5a88	445fd62	8	5

Acronyms

WHO World Health Organization

UMC Uppsala Monitoring Centre

ICSR Individual Case Safety Report

NLP Natural Language Processing

EHR Electronic Health Records

SRS Spontaneous Reporting System

FDA Food and Drugs Adminstration

OSE Office of Surveillance and Epidemiology

SPL Structured Product Label

XML eXtended Markup Language

MedDRA Medical Dictionary for Regulatory Activities

ICH International Council for Harmonisation of Technical Requirements for Pharmaceuticals for Human Use

AE Adverse Event

ADE Adverse Drug Event

ADR Adverse Drug Reaction

NER Named Entity Recognition

RI Relation Inference

FCL Fully Connected Layer

FFNN Feed-Forward Neural Network

WSD Word Sense Disambiguation

POS Part Of Speech

SPLICER Structured Product Label Information Coder and ExtractoR

EHR Electronic Health Record

SOC System Organ Class

HLGT High Level Group Term

HLT High Level Term

PT Preferred Term

LLT Lowest Level Term

CNN Convolutional Neural Network

RNN Recurrent Neural Network

LSTM Long Short-Term Memory

GRU Gated Recurrent Unit

BiLSTM Bidirectional Long Short-Term Memory

BiRNN Bidirectional Recurrent Neural Network

CRF Conditional Random Field

HMM Hidden Markov Model

MEMM Maximum Entropy Markov Model

FSM Finite State Machine

IOB2 Inside, Beginning, Outside

NMT Neural Machine Translation

IR Information Retrieval

ELMo Embeddings from Language Model

BERT Bidirectional Encoder Representations from Transformers

NMT Neural Machine Translation

charRNN character RNN

TF–IDF Term Frequency — Inverse Document Frequency

ML Machine Learning

US United States

TRITA CBH-GRU-2020:014