Preclinical Risk of Bias Assessment and PICO Extraction using Natural Language Processing

Qianying Wang

Doctor of Philosophy

Centre for Clinical Brain Sciences

The University of Edinburgh

2021

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

Qianying Wang

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor Prof Malcolm Macleod for all your guidance, patience and continuous support throughout the years. Thank you for providing the opportunity and always believing in me. I would also like to thank my other supervisor Prof Mirella Lapata for all the helpful advice on my work, and Dr Jing Liao for your technical and mental support. I am grateful to do research at the University of Edinburgh and I would like to thank the China Scholarship Council for funding my PhD.

I feel honoured to do my PhD in the CAMRADES group, and I would like to thank Emily and Gill for their comments on my presentations; Zsanett and Kaitlyn, for all the lunchtime chats, Scottish culture experiences and support in the conferences; Can and Chris, for solving my boring technique issues on pomegranate; Emma, Alex and Jing, for proofreading my thesis. And finally, Ezgi, it's lucky to work with you and thank you for always understanding my ups and downs in the journey.

Away from work, special thanks to my dearest friend Yuwei for always listening to my struggles and helping me relieved from anxiety. It is a tough journey but we will find a way. Yaqiu, thank you for pushing me to do exercise and comforting me whenever I feel upset. I will miss the hilarious dancing and cooking classes. Xiaomin, for always being available for a fancy but non-alcohol drink. Xueke, for pushing me for the application and helping me learn to accept myself. Xuerui and Bingyu, for your accompany and I will miss all the funny times we had together. Thanks Shen, Buyi, Jun, Yuxi, Yang and Zhan who bring me so much happiness during hard times. I could not have overcome all the difficulties without your help and wish you all can achieve the things you love.

Finally, I want to express my deepest love to my parents who are always proud of me and support me no matter what decision I make. I dedicate the thesis to my loving family.

Abstract

Drug development starts with preclinical studies which test the efficacy and toxicology of potential candidates in living animals, before proceeding to clinical trials examined on human subjects. Many drugs shown to be effective in preclinical animal studies fail in clinical trials, indicating the potential reproducibility issues and translation failure. To obtain less biased research findings, systematic reviews are performed to collate all relevant evidence from publications. However, systematic reviews are time-consuming and researchers have advocated the use of automation techniques to speed the process and reduce human efforts. Good progress has been made in implementing automation tools into reviews for clinical trials while the tools developed for preclinical systematic reviews are scarce. Tools for preclinical systematic reviews should be designed specifically because preclinical experiments differ from clinical trials. In this thesis, I explore natural language processing models for facilitating two stages in preclinical systematic reviews: risk of bias assessment and PICO extraction.

There are a range of measures used to reduce bias in animal experiments and many checklist criteria require the reporting of those measures in publications. In the first part of the thesis, I implement several binary classification models to indicate the reporting of random allocation to groups, blinded assessment of outcome, conflict of interests, compliance of animal welfare regulations, and statement of animal exclusions in preclinical publications. I compare traditional machine learning classifiers with several text representation methods, convolutional/recurrent/hierarchical neural networks, and propose two strategies to adapt BERT models to long documents. My findings indicate that neural networks and BERT-based models achieve better performance than traditional classifiers and rule-based approaches. The attention mechanism and hierarchical architecture in neural networks do not improve performance but are useful for extracting relevant words or sentences from publications to inform users' judgement. The advantages of the transformer structure are hindered when documents are long and computing resources are limited.

In literature retrieval and citation screening of published evidence, the key elements of interest are Population, Intervention, Comparator and Outcome, which compose the framework of PICO. In the second part of the thesis, I first apply several question answering models based on attention flows and transformers to extract phrases describing intervention or method of induction of disease models from clinical abstracts and preclinical full texts. For preclinical datasets describing multiple interventions or induction methods in the full texts, I apply additional unsupervised information retrieval methods to extract relevant sentences. The question answering models achieve good performance when the text is at abstract-level and contains only one intervention or induction method, while for truncated documents with multiple PICO mentions, the performance is less satisfactory. Considering this limitation, I then collect preclinical abstracts with finer-grained PICO annotations and develop named entity recognition models for extraction of preclinical PICO elements including Species, Strain, Induction, Intervention, Comparator and Outcome. I decompose PICO extraction into two independent tasks: 1) PICO sentences classification, and 2) PICO elements detection. For PICO extraction, BERT-based models pre-trained from biomedical corpus outperform recurrent networks and the conditional probabilistic module only shows advantages in recurrent networks. Self-training strategy applied to enlarge training set from unlabelled abstracts yields better performance for PICO elements which lack enough amount of instances.

Experimental results demonstrate the possibilities of facilitating preclinical risk of bias assessment and PICO extraction by natural language processing.

Lay Summary

In drug development, drug efficacy must be tested in preclinical animal experiments before the candidates proceed to clinical trials. Many drugs proven to be effective in living animals fail in clinical trials, suggesting the translation issue between preclinical to clinical research. Many factors affect the validity and credibility of preclinical animal studies and scientists obtain less biased research findings by conducting systematic reviews to collate all relevant evidence from publications. Systematic reviews are time-consuming, and automation techniques can be used to speed up the process and reduce human efforts, which have achieved good progress in clinical systematic reviews. Tools for preclinical systematic reviews are scarce and should be designed separately because preclinical experiments differ from clinical trials. The development of natural language processing (NLP) which uses computer programming to process, analyse and understand human language, enables the exploiting of such tools for preclinical systematic reviews. In this thesis, I explore NLP techniques to facilitate two stages for preclinical systematic reviews, risk of bias assessment and PICO extraction.

There are a range of measures used to reduce bias in animal experiments and several checklist criteria require the reporting of these measures in publications. In the first past of the thesis, I focus on using text classification models to automatically indicate if random allocation to groups, blinded assessment of outcome, conflict of interests, compliance of animal welfare regulations, and animal exclusions are reported in preclinical publications. I compare traditional machine learning classifiers with several text representation methods, three neural networks, and propose two strategies to adapt transformer models to long documents. My findings suggest that neural networks and transformer models achieve better performance than traditional classifiers and rule-based approaches but some complicated structures do not show advantages for long documents.

When retrieving publications in the database and determining whether the studies should be included or excluded in a systematic review for a biomedical research question, the key elements of interest are Population, Intervention, Comparator and Outcome, which compose the framework of PICO. In the second part of the thesis, I focus on extracting PICO phrases from publications by NLP methods, which is further categorised into two types of tasks: question answering and named entity recognition, depending on the annotation format of datasets. The principle of the two tasks is to classify each word in the article into pre-defined PICO categories, and find the most possible combination of the starting and end words of a PICO phrase, which are then extracted from the text. I first apply question answering models to extract phrases describing intervention or method of induction of disease models from clinical abstracts and preclinical full texts. For preclinical datasets which have multiple interventions or induction methods described in the full texts, I apply additional unsupervised methods to extract relevant sentences. The question answering models achieve good performance when texts are at abstract-level and contain only one intervention or induction method, while for truncated documents with multiple PICO mentions, the performance is less satisfactory. Considering this limitation, I then collect preclinical abstracts with finer-grained PICO annotations and develop PICO recognition models for extraction of preclinical PICO elements including Species, Strain, Induction, Intervention, Comparator and Outcome. My findings suggest transformer models pre-trained from biomedical corpus outperform other neural networks and a probabilistic module which proved to be beneficial in neural networks does not show advantages in transformer models. A semi-supervised strategy is applied to enlarge the training set from unlabelled abstracts, which yields better performance for PICO elements lacking enough amount of instances.

Experiments results demonstrate the possibilities of facilitating preclinical risk of bias assessment and PICO extraction by natural language processing.

Contents

List of Abbreviations

AE: Animal Exclusions

Attn: Attention mechanism

BAO: Blinded Assessment of Outcomes

BERT: Bidirectional Encoder Representations from Transformers

BERT-DCP: BERT with Document Chunk Pooling

BERT-SE: BERT with Sentence Extraction

Bi: Bidirectional

BiDAF: Bidirectional Attention Flow

BM25: Okapi BM25

bow: Bag-of-Words

CAWR: Compliance of Animal Welfare Regulations

CI: Conflict of Interests

CNN: Convolutional Neural Network

CRF: Conditional Random Field

d2v: doc2vec

DBOW: Distributed Bag-of-Words

DM: Distributed Memory

EM: Exact Match

GRU: Gated Recurrent Unit

HAN: Hierarchical Attention Network

LogReg: Logistic Regression

LSTM: Long Short-Term Memory

MAP: Mean Average Precision

MMR: Mean Match Ratio

NER: Named Entity Recognition

QA: Question Answering

RA: Random Allocation

RF: Random Forest

rMAP: 'ratio-mode' Mean Average Precision

rMMR: 'ratio-mode' Mean Match Ratio

RNN: Recurrent Neural Network

RoB: Risk of Bias

SBERT: Sentence BERT, or Sentence Transformers (Sentence Embeddings using Siamese BERT-Networks)

sMAP: 'strict-mode' Mean Average Precision

sMMR: 'strict-mode' Mean Match Ratio

SVM: Support Vector Machine

TF-IDF: Term Frequency - Inverse Document Frequency

w2v: word2vec

List of Tables

List of Figures

Chapter 1 Introduction

1.1 Preclinical Research

Developing a new drug is time-consuming and costly. Scientists first conduct the basic research to understand the mechanism of a disease and discover potential compounds as drug candidates. This is followed by preclinical studies to test the efficacy and toxicology of drug candidates inside and outside of the living animals. Finally, clinical trials are conducted to test if the drug has similar effects on human subjects over five phases (Honek, 2017). It can take up to 12 years to go from discovery to approval to reaching target patients, with an average cost of over 2 billion dollars (Wouters *et al.*, 2020).

Preclinical research is an important interlink between drug discovery and clinical testing. Researchers conduct preclinical studies in living animals (in vivo) and microorganisms, cells, or biological molecules (in vitro) to obtain information for how to treat the diseases by drug candidates, and whether the candidates are effective and safe. Regulatory agencies including Food and Drug Administration (FDA) and European Medicines Agency (EMA) require drug candidates to undergo a series of investigations including pharmacodynamics (biochemical and physiological effects of drugs on the living body), pharmacokinetics (uptake, distribution, metabolites and elimination of drugs in the living body) (Nordberg *et al.*, 2004), and toxicology testing. These tests are used to determine the formulation, route, frequency, dosage, and duration of exposure of drugs, and must be accurate to guarantee the efficacy and safety. Candidates that do not pass the tests at the preclinical stage must be abandoned and cannot proceed into the clinical trials.

Reproducibility Crisis and Translation Failure

More than one hundred million animals are used for biomedical experiments every year and the number of preclinical studies is rapidly increasing (Taylor *et al.*, 2008). Guidelines including the Good Laboratory Practice (GLP) and Good Scientific Practices (GSP) have been established to regulate the implementation of animal experiments to ensure their reliability and reproducibility (Shegokar, 2020). However, evidence has uncovered a reproducibility crisis of preclinical studies and difficulties in translating findings from preclinical to clinical research. For example, when a pharmaceutical company investigated their 67 in-house projects in the fields of oncology, woman's health and cardiovascular, they were able to reproduce only 21% of the results (Prinz *et al.*, 2011). Scott *et al* analysed 70 drugs on their abilitiy to extending life in a murine model of amyotrophic lateral sclerosis, tested 18,000 mice across 221 studies, but none of the significantly positive or negative results could be replicated (Scott *et al.*, 2008). In experimental stroke research, 1026 treatments were tested in preclinical experiments and 374 interventions were tested effective in in vivo experiments, but only one intervention was proved to be effective in clinical trials (O'Collins *et al.*, 2006). The low success rate indicates that the preclinical studies are not robust as expected and the usability is limited.

1.2 Systematic Review

Traditionally, biomedical scientists make decisions on experimental design or clinical practice based on individual expertise, intuition, previous experience, or combined with a narrative review of relevant published work. This decision-making process can produce rapid solutions but often lacks a clear definition of the research question, a systematic search and selection criteria of all relevant studies, and rigorous summarising and evaluation of results (Pae, 2015). Findings from narrative reviews are subjective, biased, and can be inconsistent (Cipriani and Geddes, 2003). Systematic review is developed and advocated for in the field of evidence-based medicine which aims to inform the best decision making using all available research findings relevant to a pre-specified research question (Sackett *et al.*, 1996; Masic *et al.*, 2008; Higgins *et al.*, 2011).

To conduct a systematic review, investigators first formulate the research question, and define the inclusion and exclusion criteria for studies (Uman, 2011). A search strategy is then developed based on a comprehensive list of Medical Subject Headings (MeSH) terms (Bodenreider, 2004), which are used to retrieve articles from databases such as PubMed, Embase and Web of Science. Next, two independent investigators perform citation screening to manually check if retrieved study should be included or excluded. They then extract data from the included full-text publications and organise the information for later analysis. Any disagreement during the process is solved by a senior expert. The validity of studies varies and quality assessment is applied to inform bias of the quantitative results from meta-analysis (Julian P. T. Higgins *et al.*, 2011). Those stages are transparent and reproducible, and the conclusions from a systematic review are less biased.

Preclinical vs. Clinical

Systematic review of clinical studies has reshaped the evidenced-based medicine (Sackett *et al.*, 1996; Szajewska, 2018) and contributed greatly to the clinical practice (Macleod *et al.*, 2004) while the development of methods and standards in systematic review for laboratory animal experiments is relatively backward (de Vries *et al.*, 2014). Preclinical systematic review is distinguished from clinical systematic review because the characteristics of animal experiments and clinical trials are different in many aspects (Muhlhausler *et al.*, 2013). Preclinical animal studies aims to explore new hypotheses for treatment development and test the safety of interventions, while clinical trials mainly aim to examine the treatment efficacy on human patients (Hooijmans *et al.*, 2014). Clinical trials enrol patients with naturally presenting disease, whilst preclinical studies rely on induction of disease models by human researchers. The sample size in preclinical experiments is relatively small due to concerns regarding the cost and ethics of using animals (Sena *et al.*, 2014), and different species can be involved. Various lab outcomes including post-mortem outcomes such as histopathology of organs can be investigated in animal experiments, while outcomes in clinical trials are directly relevant to the patient. These factors induce more variations to outcomes of preclinical studies, thus the methods of interpreting the variations and exploring the source of heterogeneity are different from that in the clinical systematic review (Hooijmans *et al.*, 2018). The proper implementation of a preclinical systematic review can provide a reliable summary of research findings, inform future experimental design and animal model selection (Higgins *et al.*, 2011; Hooijmans *et al.*, 2014; Hooijmans *et al.*, 2018), indicate if an intervention can proceed into clinical trials (Pound and Ritskes-Hoitinga, 2020), and reduce research waste (Macleod *et al.*, 2014; Ioannidis *et al.*, 2014; Moher *et al.*, 2016).

Risk of Bias

Systematic review is not bias free and many potential factors affect the validity and credibility of preclinical animal studies (Sena *et al.*, 2014). The validity of an individual animal study affects its usability, generalisation and the evaluation of summarised findings from all relevant animal studies in the same topic domain, and further affects the transition to clinical trials. Four categories of validity are considered in preclinical animal studies. These are internal validity (how well the experiments were performed), external validity (whether the results can be generalised to experiments conducted in other conditions, population, time points, etc), construct validity (whether the animal models can represent the methodology behind), and reporting bias (non-significant results tend to take longer to be published) (van der Staay *et al.*, 2009; Worp *et al.*, 2010). Particularly, internal validity is influenced by a range of biases including selection bias, performance bias, detection bias, attrition bias and other biases (Hooijmans *et al.*, 2014). In animal experiments, these biases can be reduced by measures such as randomly allocating animals to groups, blinding scientists who perform the experiments and analyse the outcomes so they do not know which intervention an animal received, and specifying any exclusion of animals and outcome data (Macleod *et al.*, 2015).

Synthesis of results from experiments lacking these measures can lead to the efficacy of a treatment or intervention being overestimated or underestimated, which can be partially reflected by some reporting issues. For instance, in a preclinical systematic review of focal cerebral ischaemia, the intervention was found 30% more effective in the studies that did not report blinded assessment of outcome than the studies that reported blinding (MacLeod *et al.*, 2008); and the treatment effect of acute ischaemic stroke was 10% more effective in the studies that did not report randomisation than the studies that reported randomisation (Van Der Worp *et al.*, 2007). Poor reporting does not directly mean the experiments were conducted in a biased manner, as sometimes investigators may not report the bias-related measures of a rigorous experimental study in the publication, for a certain reason. Once journals reach a consensus and widely require the reporting of a series of designs, measures and implementations in the publications, there is no doubt that the results in the publications which do not report those measures should be questioned. To raise the awareness of the importance of reporting, and enable the assessment process transparent and objective, standard checklists for risk of bias assessment in animal intervention studies have been developed, such as CAMARADES ten-item quality checklist (Macleod *et al.*, 2004), ARRIVE guidelines (Kilkenny *et al.*, 2010), Landis checklist (Landis *et al.*, 2012) and SYRCLE's risk of bias tool (Hooijmans *et al.*, 2014). Although the specific criteria of risk of bias items may vary depending on the study, some checklist items are widely applicable across experiments with different treatments or interventions (Macleod *et al.*, 2015), such as random allocation of animals to groups, blinded assessment of outcome, compliance of animal welfare regulations, potential conflict of interests, animal exclusions from the study, etc. Clear reporting of the risk of bias items provides more options for exploring the source of heterogeneity and discussing the effect of relevant measures on the overall efficacy in meta-analysis, hence reduce the risk of an intervention being overestimated or underestimated.

PICO

Literature search, citation screening and data extraction are the critical steps in systematic reviews. The key elements of interest in these procedures are the Population/Problem, Intervention, Comparator and Outcome, which compose the framework of PICO (Richardson *et al.*, 1995). The PICO frame has been used as the basis for retrieval, inclusion and classification of published evidence, and is beneficial to clinical evidence based medicine (Jin and Szolovits, 2018a). Empirical studies have shown the usage of PICO elements in literature retrieval facilitates more complex search strategies and yields more precise search results for systematic reviews or answering clinical questions (Huang et al., 2006; Schardt *et al.*, 2007; Boudin *et al.*, 2010; Chabou and Iglewski, 2018; Booth *et al.*, 2000). During citation screening, investigators screen abstracts to determine the inclusion or exclusion of studies. Abstracts that are pre-structured according to the PICO frame or combine demonstration with PICO phrases enable faster judgement of study relevance for each PICO element (Jin and Szolovits, 2018b; Tsafnat *et al.*, 2018; Brockmeier *et al.*, 2019). Pre-structured PICO information also allows investigators to locate relevant descriptions from full-text articles which may speed up the data extraction process (Wallace *et al.*, 2016).

The application PICO framework is less common in the preclinical field. Considering the leading clinical research and its difference compared to the preclinical experiments, the SYRCLE group have derived the definition of preclinical PICO from the clinical PICO frame, where "Population" does not represent patients but instead represents the animal species, strain and method of induction of disease model, and outcomes can involve survival, behavioural, histological and biochemical outcomes which are not directly relevant to clinical situations (Hooijmans *et al.*, 2018).

1.3 Automation for Systematic Review

Both the number of biomedical publications and the number of systematic reviews are rapidly increasing (Bastian et al., 2010; Fontelo and Liu, 2018). In an investigation of 195 reviews registered in PROSPERO, the number of studies entered into a systematic literature search ranged from 27 to 92,020, and it took more than 2.5 years on average to complete a systematic review and publish the results (Borah et al., 2017). The large number of publications require many human efforts across all stages in a systematic review, and the lengthy duration to reach publication means the findings of reviews may go out of date (Shojania *et al.*, 2007).

The evidence synthesis community has advocated for the use of automation techniques to assist systematic reviews (Elliott *et al.*, 2017; Thomas et al., 2017). Recent developments in machine learning and natural language processing have allowed the training and building of automation tools which can aid systematic reviews. The main tasks involved in systematic reviews can be formulated into text classification and data extraction (Marshall and Wallace, 2019).

Text Classification

Text classification is a categorisation task which aims to group documents into classess of interests. More formally, given a document $x \in X$ and a set of classes $C = \{c_1, c_2, \dots\}$, where x is the numeric representation of the document and X is the document space, a classification function or model f is learned to map the representations of documents into classes, i.e. $f\colon X \to C$ (Manning et al., 2008). For example, citation screening can be modelled to automatically classify abstracts or full-text articles into two classes, such as studies that describe in vivo experiments and those that do not, or studies which investigate a certain disease and those that do not. Similarly, risk of bias assessment can be modelled to classify the representations of full texts or their sentences into two classess including ‘reported’ and ‘unreported’, for each checklist item.

Data Extraction

Data extraction, including PICO extraction, aims to identify numeric values or precise words and then tabularise the information so it can be used directly in the following meta-analysis. However, in practice, considering the difficulties of identifying structured information, PICO extraction is often converted to identify sentences, snippets or phrases describing PICO elements from full texts. Particularly, identifying snippets or phrases requires the classification of words (or tokens) in the given text, so the consecutive tokens with high probabilities can be then concatenated together to form the phrases. In the field of natural language processing, extracting words or phrases from context is the word or token classification task, which can be further categorised into question answering or named entity recognition (Manning et al., 2008), depending on the format of annotations in the labelled training data.

More specifically, in the question answering architecture, given a question sequence such as "What is the intervention?" and its relevant context, i.e. the corresponding publication, a model is learned to find the position of answer strings from the publication context, by classifying the representation vector of each context token into two classes to indicate if the token belongs to the intervention answer strings or not respectively. While in the named entity recognition architecture, a model is learned to classify the representation of each token into pre-defined PICO entity types such as "Intervention" or "Outcome". Both tasks belongs to the multi-class token classification problem: given an input text sequence $X = \{x_1, x_2, \dots\}$, where x_i is the i-th token (word) in the sequence, the goal is to assign a label for each token and generate an output label sequence $Y = \{y_1, y_2, \dots\}$. For each individual publication, the question answering architecture requires the annotation of the position of exact answer strings, which is more applicable for the situation where only one or two precise answer strings exist for the question, while the named entity recognition architecture requires the annotation of every mentioned PICO phrases, which allows multiple mentions of a same PICO phrase.

The basis of solving those tasks is first converting text into numeric representations of individual words, sentences or whole documents, and then training classifiers to learn patterns, relations and parameters across different dimensions and positions of text representations, which can then be used to score new articles for prediction. Both text representation methods and classifiers involve statistical methods and traditional machine learning models, and the development of neural network methods (Goldberg, 2016) enriches the field, especially the transformer-based models (Vaswani *et al.*, 2017) which achieve state-of-the-art performance on a wide range of tasks (Devlin *et al.*, 2018).

Automation tools developed using machine learning and natural language processing algorithms have achieved great success in clinical fields, such as EPPI reviewer (Shemilt *et al.*, 2016) for citation screening, ExaCT (Kiritchenko *et al.*, 2010) for data extraction and RobotReviewer (Marshall *et al.*, 2016) for bias assessment. However, tools for preclinical systematic reviews are scarce. Citation screening tools for automatically classifying in vivo abstracts (Liao *et al.*, 2018) is well established, and a preclinical risk of bias assessment tool exists (Bahor *et al.*, 2016) but it is built on a rule-based method which generates only binary labels and is limited to a certain disease. Tools for risk of bias assessment and PICO extraction using recent natural language processing techniques are waiting to be explored.

1.4 Thesis overview

Preclinical research is critical to decide which drugs to move forward to clinical trials but there are reproducibility crises and translation issues which limit clinical success. Scientists conduct systematic reviews on preclinical animal experiments to obtain less biased summarises of findings and explore the reliability of treatment efficacy. Procedures in systematic reviews including quality assessment and data extraction are time-consuming. Automation tools designed for preclinical systematic reviews are necessary to speed up the process and reduce human efforts, which may further contribute to the translation from preclinical to clinical research.

The overall aim of the thesis is to investigate and apply natural language processing models for risk of bias assessment and PICO elements extraction in preclinical publications. More specifically, I formulate risk of bias assessment as a long document classification task, and PICO elements extraction as a token classification task, which is further categorised into question answering and named entity recognition, depending on the specific format of annotations in the datasets. The remainder of this thesis is structured as follows.

In Chapter 2, I perform a series of binary classification models for full-text preclinical publications, to automatically indicate if they reported five risk of bias items including random allocation to groups, blinded assessment of outcomes, conflict of interests, compliance of animal welfare regulations, and statement of animal exclusions. I perform traditional machine learning classifiers with several text representation methods as baselines. I illustrate the mechanism of the convolutional, recurrent and hierarchical neural networks, and how they are used to construct the classification model. I present critical elements in transformer models and propose two strategies to adapt BERT models to long documents. After implementation, training and parameters tuning procedures, I compare the validation performance of above models, and select the best model for each risk of bias item separately. I discuss whether the attention mechanism in the recurrent and hierarchical structure can be used to extract the most important words and relevant sentences for risk of bias assessment. I use the optimal models to build the preclinical risk of bias tool and compare the test performance with an existing text mining approach.

In Chapter 3, I present question answering models and demonstrate the possibilities to extract phrases describing the intervention used or the method of induction of disease models from clinical abstracts and preclinical full texts. For clinical abstracts which focus on one intervention measure, I implement the classic question answering models based on attention flows and transformers to extract the exact phrases. For preclinical datasets which have multiple interventions or induction methods described in the full texts, I explore several information retrieval methods to extract relevant sentences which are concatenated into new passages, and implement the question answering models to extract and filter out a list of candidates for interventions or induction methods from the shortened passages. I compare the performance of information retrieval methods and question answering models, and demonstrate the prototype of an interface developed using the best model trained from the preclinical datasets.

In Chapter 4, I present the annotation of six PICO elements (Species, Strain, Induction, Intervention, Comparator and Outcome) in the collected preclinical abstracts, and use them to develop models for PICO phrases extraction. I decompose PICO extraction into two independent tasks: 1) PICO sentences classification, where I apply BERT to classify sentences from abstracts and remove non-PICO sentences to obtain the shortened texts; 2) PICO elements detection, where I implement recurrent networks and BERT-based models with or without a conditional probabilistic model, on the truncated abstracts to extract precise PICO phrases. I apply the self-training strategy to enlarge the training datasets and discuss whether it can enhance the test performance.

In Chapter 5, I summarize the conclusions from previous chapters, and discuss directions for future work.

Chapter 2 Assessing Risk of Bias Reporting in Preclinical Literature: Text Classification

Systematic review is believed to be the least biased method of collating evidence from research publications. It is not bias free and quality assessment is performed to evaluate the internal validity of individual studies (van der Staay *et al.*, 2009; Worp *et al.*, 2010). Flaws in experimental design reflect from some reporting issues in publications and studies have shown that publications lacking clear reporting of risk of bias leads to the treatment efficacy being overestimated or underestimated (Van Der Worp *et al.*, 2007; MacLeod *et al.*, 2008; Sena *et al.*, 2014), which further affect the translation to clinical research. To reduce risk of bias, regulate experimental designs and raise the awareness of reporting issues in preclinical systematic reviews, several risk of bias tools have been established, such as CAMARADES ten-item quality checklist (Macleod *et al.*, 2004), ARRIVE guidelines (Kilkenny *et al.*, 2010), Landis checklist (Landis *et al.*, 2012) and SYRCLE's risk of bias tool (Hooijmans *et al.*, 2014). In the field of evidence synthesis, the risk of bias 'tools' often refer to the guidelines or checklist forms with a series of criteria of items related to experimental designs such as randomisation, blinding and sample size calculation. The assessment process is conducted by at least two independent investigators to check every included full-text publication separately, extract the relevant descriptions and then make the judgement if the publication has low, high, or unknown risk for a specific bias item. A senior adjudicator is also required to resolve any disagreements between the two investigators. The reviewing work is repetitive and workload is rapidly increasing due to the fast-growing number of preclinical publications. To speed up the workflow and systematically evaluate the risk of bias, it is critical to develop tools to automatically appraise risk of bias reporting. Improving this workflow enables the processing and evaluation of large amounts of publications to guide researches by institutions, journals and funders, which may improve research activities and provide more precise findings to inform clinical trials.

2.1 Related Work

Systematic reviewers have advocated the use of automated approaches to assist risk of bias assessment, using human effort and machine automation in mutually reinforcing ways (Elliott *et al.*, 2017). The development of machine learning and natural language processing, including neural models and transfer learning, provides opportunities to create robust tools for risk of bias assessment.

For clinical trials, Marshall et al have applied support vector machine with bag-of-word representations at sentence/document/joint levels, using 2,200 full-text clinical reports with annotated document labels ('low risk' and 'unknown/high risk') and sentence quotes which indicate the justification. They found joint models using associated sentence-level annotations improve the predictions by between 1% and 9% for six bias domains, and the optimal model achieves F1 of 72% for random sequence generation and 67% for blinded assessment of outcome (Marshall *et al.*, 2015). In a later development of RobotReviewer (Marshall *et al.*, 2016), they use distant supervision (Mintz *et al.*, 2009) to automatically derive pseudo document and sentence level annotations from unlabelled clinical reports, based on the 1,400 unique strings of bias domains from the Cochrane Database of Systematic Reviews. By training the support vector machine separately for each bias domain or jointly for all domains (multi-task learning (Ruder, 2017)) on 6,610 algorithmically annotated full texts, they obtain the accuracy of 76% for random sequence generation and 67% for blinded assessment of outcome. Experiments have shown their automation tool is quicker than pure manual assessment (Soboczenski *et al.*, 2019). A risk of bias assessment function is also developed in Trialstreamer (Marshall *et al.*, 2020), by training a logistic regression model on 13,463 abstracts of clinical trials which achieves a recall of 46% and precision of 44%. Similarly, Millard et al apply logistic regression models on 1,467 full-text clinical reports for sentence and document classification separately, which achieves an area under the ROC curve over 72% for randomisation sequence generation, allocation concealment and blinding (Millard *et al.*, 2016).

With the same datasets used in the development of RobotReviewer, Zhang et al consider the supported sentence annotations of bias domains as 'rationales' and use them to train convolutional neural networks (Kim, 2014) to classify sentences into three classes (positive, negative and neutral) for a bias item in each document (Zhang *et al.*, 2016). They explore different approaches to summarise sentence vectors to obtain the document representation, by simply summarising, summarising by class weights, or summarising by sentence weights which are jointly learned in the convolutional neural network with an 'attention' mechanism (Yang *et al.*, 2016). The accuracies of four bias items are improved by 5% using the rationale-augmented convolutional neural networks, compared to their baseline support vector machines.

Menke et al have reported the performance of a proprietary tool 'SciScore' (Menke *et al.*, 2020) which trains conditional random fields (Sutton and McCallum, 2011) on 250 research articles manually labelled with entity mentions for random allocation and blinding. The training corpus is randomly selected from the PubMed Open Access articles, and the portion of clinical or preclinical publications is not clear.

Compared with clinical trials, animal studies are conducted in relatively small teams, are reported in a different style, have been shown to have lower reporting of strategies to reduce risk of bias (Macleod *et al.*, 2015), and are susceptible to different risk of bias items (Hooijmans *et al.*, 2014). Hence, separate tools for RoB assessment in preclinical literature are necessary. Bahor et al. have previously reported the use of regular expressions based on rule-based string matching to recognise phrases related to bias reporting in experimental animal studies, which requires many hand-crafted term selections (Bahor *et al.*, 2017).

2.2 Dataset

I use a collection of full-text publications which have been annotated for risk of bias items in preclinical systematic reviews in three domains: 2,453 records of a focal ischaemic stroke study (McCann *et al.*, 2016), 1,626 records of a chemotherapy-induced peripheral neuropathy study (Currie *et al.*, 2019), 2,404 records of a psychotic disorder study (Bahor *et al.*, 2016); and from two studies assessing the effectiveness of interventions to improve reporting quality across in vivo research: 760 records of NPQIP (Nature Publication Quality Improvement Project) (The NPQIP Collaborative Group, 2019) and 665 records from the IICARus project (Intervention to Improve Compliance with the ARRIVE guidelines) (Hair *et al.*, 2019). The risk of bias labels are at the document level (1 for reported, 0 for not reported/unknown) which are derived from the annotations of two independent investigators followed by an internal validation process.

I consider five risk of bias items: (1) Random Allocation (RA): animals are randomly allocated to treatment or control groups at the start of the experiment; (2) Blinded Assessment of Outcome (BAO): group identity is concealed from the scientist measuring the outcome; (3) Conflict of Interests (CI): the authors report any relationship which might be perceived to introduce a potential conflict of interests, or the absence of such relationships; (4) Compliance with Animal Welfare Regulations (CAWR): the researchers report that they complied with animal welfare regulations; (5) Animal Exclusions (AE): a statement of whether or not all animals, all data and all outcomes measured are accounted for and presented in the final analysis. Some example sentences indicating the reporting of each risk of bias item are displayed in Table 2.1.

Publications are initially in PDF format and I convert them to plain texts using Xpdf (www.xpdfreader.com). I convert all text to lower case and use regular expression to remove references, citations, URLs, digits, non-ASCII characters and text which precedes the “Introduction” section, because they are unlikely to report risk of bias. I use Stanford CoreNLP (Manning et al., 2015) for word and sentence tokenisation. After removing invalid records (for instance where text conversion fails), 7,840 full-text publications have annotations for random allocation, blinded assessment and animal exclusions, and 7,089 of them have annotations for animal welfare regulations and conflict of interests. Records are shuffled and randomly split into training, validation and test set (80%/10%/10%). Some summary statistics of the dataset are shown in Table 2.2.

Risk of bias item	**Positive example**
Random allocation	…a randomisation code is used to allocate animals to treatment group…
Blinded assessment of outcome	…the midbrain sections from each animal were screened for … by a person unaware of the treatment condition of the animals…
Conflict of interests	The authors declare that they have no competing interests.
Compliance of animal welfare regulations	…experiments were performed in accordance with protocols by the Institutional Animal Care and Use Committee at…
Animal exclusions	... cases in which the lesion was assessed to involve less than <50% of the dopamine neurons, the animal was excluded from...

Table 2.1: Example sentences from a full-text publication indicating the risk of bias reporting.

	Samples for RA, BAO, AE			Samples for CAWR, CI		
	Train	Valid	Test	Train	Valid	Test
No. documents	6272	784	784	5671	708	710
Avg no. tokens per document	4977	5112	5077	4947	5057	4964
Avg no. sentences per document	180	186	184	178	182	178
Avg no. tokens per sentence	28	28	28	28	28	28
Ratio of records reported RA	27%	25%	30%	--	--	--
Ratio of records reported BAO	30%	29%	33%	--	--	--
Ratio of records reported AE	12%	14%	11%	--	--	--
Ratio of records reported CAWR	--	--	--	78%	77%	75%
Ratio of records reported CI	--	--	--	32%	28%	31%

Table 2.2: Dataset statistics. Samples for random allocation, blinded assessment of outcomes and animal exclusions consist of 7,840 records; samples for animal welfare regulations and conflict of interests consist of 7,089 records.

2.3 Methods

Risk of bias assessment can be cast as a typical text classification task: the input is the full-text publication, and the output is the binary risk of bias label (reported/unreported) for each item. A classification model cannot be trained from the plain text directly and I need to convert text information into analysable data. The core concept is to map each document to a matrix consisting of fixed-dimension word vectors or word embeddings (Daniel and Martin, 2020), before training a classification model which learns to map the numeric text representations to the binary output label. For text representation methods, I explore bag-of-words (Goldberg, 2017), word2vec (Mikolov, Chen, *et al.*, 2013), doc2vec (Le and Mikolov, 2014) and embeddings from BERT (Devlin *et al.*, 2018). For classification models, I implement three baseline models (support vector machine, logistic regression and random forest), neural models (convolutional neural network, recurrent neural network with attention and hierarchical neural network) and BERT models using two strategies (document chunk pooling and sentence extraction). An overview of the methods is demonstrated in Figure 2.1, and details are described in the following sections.

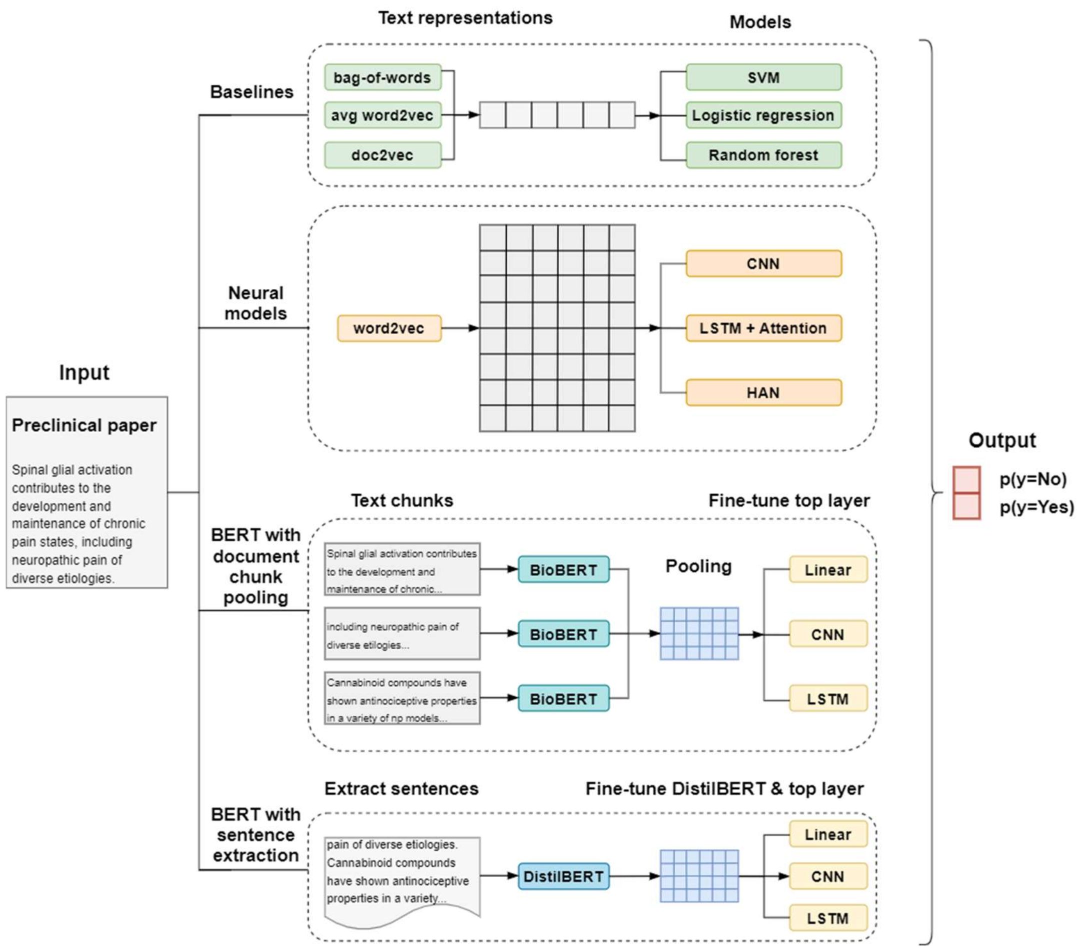

Text representations
Models
Baselines
bag-of-words
avg word2vec
doc2vec
SVM
Logistic regression
Random forest
Neural models
word2vec
CNN
LSTM + Attention
HAN
Input
Preclinical paper
Spinal glial activation contributes to the development and maintenance of chronic pain states, including neuropathic pain of diverse etiologies.
Output
p(y=No)
p(y=Yes)
Text chunks
Fine-tune top layer
BERT with document chunk pooling
Spinal glial activation contributes to the development and maintenance of chronic...
including neuropathic pain of diverse etilogies...
Cannabinoid compounds have shown antinociceptive properties in a variety of np models...
BioBERT
Pooling
Linear
CNN
LSTM
Extract sentences
Fine-tune DistilBERT & top layer
BERT with sentence extraction
pain of diverse etiologies. Cannabinoid compounds have shown antinociceptive properties in a variety...
DistilBERT

Figure 2.1: An overview of text representation methods and classification models being explored.

2.3.1 Text representations

Bag-of-words (bow) uses the word frequency within a document to represent the word importance. In this method, a document with s words is converted to a s-dimensional vector where the i-th element is the i-th word frequency. Stop words such as ‘the’ and ‘a’ should be assigned lower weights or removed as they usually have high counts but are less helpful to understand texts. Some words related to the topic may appear more common than other words among the documents and should be assigned higher weights. The common solution is using TF-IDF (Term Frequency-Inverse Document Frequency) weighting

(Goldberg, 2017). In a collection of full-text publications, the TF-IDF weight of the i-th word w_i in the j-th document d_j is:

$$\mathrm{tfidf}\left(w_i, d_j, D\right) = \frac{f\left(w_i, d_j\right)}{l\left(d_j\right)} \times \ln \frac{N(d)}{n(w_i)} \tag{2. 1}$$

where $f(w_i, d_j)$ is the frequency of word w_i in the document d_j, $l(d_j)$ is the total number of words in the document d_j, i.e. document length, $N(d)$ is the total number of documents in the collection of full-text publications (corpus) D, and $n(w_i)$ is the number of documents containing the word w_i. Words can be counted by unigram (each word is counted alone as one token), bigram (every two adjacent words are counted as one token) and combination of unigram and bigram (Goldberg, 2016).

Word2vec is a type of language modelling which learns to map words to continuous vectors which can preserve the semantic relationship among words. For example, the cosine distance of vector('King') and vector('Man') is similar to the distance of vector('Queen') and vector('Woman'). These word vectors can be generated from the learning process jointly within the neural network and one commonly used method is the skip-gram model, which predicts surrounding context words given a centre word (Mikolov, Chen, *et al.*, 2013). Initially, from a large corpus with vocabulary size v, every single word is represented by a v-dimensional one-hot vector and is fed into a simple neural network. As Figure 2.1 shows, the input layer receives the one-hot vector of a centre word ('treatment'), and the output layer generates the probability vectors $\hat{y}$ of the predicted context words ('effects' and 'therapy'). The neural network is trained to update the parameter matrix $W \in \mathbb{R}^{v \times h}$ and $W' \in \mathbb{R}^{h \times v}$ to get a low cross-entropy loss between the probability vectors $\hat{y}$ and the actual one-hot vectors y, where h is the size of the hidden layer. The parameter matrix W obtained through this training process is the representation matrix consisting of v word vectors or embeddings, which can be used for other natural language processing tasks. Word vectors are pre-trained on a large amount of corpora like GoogleNews and Wikipedia, and are then applied as the input of new tasks, either fixed or jointly updated with other new parameters. Large vocabulary size v increases computing expenses of updating weight matrix W, and negative sampling (Mikolov, Sutskever, *et al.*, 2013) is introduced to solve this issue. In the negative sampling method, instead of updating all $h \times v$ parameters in W, a small amount of 'negative' words are selected, for example 10 words, by uniform distribution, and parameters of 'positive' words ('effects' and 'therapy' in Figure 2.2) and 10 'negative' words (other words excluding 'effects' or 'therapy' in Figure 2.2) are selectively updated. The computation complexity is then reduced from $O(h \times v)$ to $O(h \times 10)$, where v is often on billion-level.

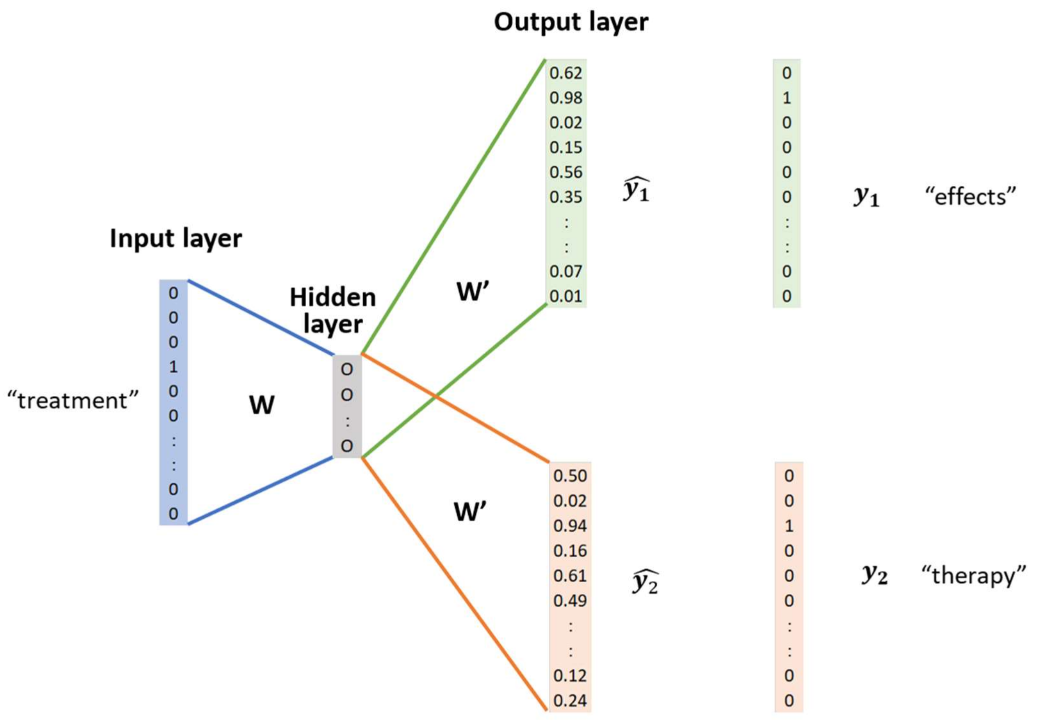


Figure 2.2: An example of a skip-gram model with window size 1 (considering one adjacent word of the central word, from both directions)

Doc2vec is an unsupervised method which learns to represent a document by a dense vector (Le and Mikolov, 2014). For baseline classification models which only receive a vector representation as input, a document's matrix representation from word2vec is needed to be summarised or averaged to reduce dimension. Representation from doc2vec can be fed to those baseline classifiers directly, which often outperforms simply summarised or averaged word vectors. There are two methods to generate the document vector: Distributed Memory (DM) and Distributed Bag-of-Words (DBOW), see Figure 2.3. In the distributed memory method, the i-th document is represented by a unique vector d_i, and a matrix consisting of word vectors $\{v_{i,1}, v_{i,2}, \ldots\}$ in sequential order, where the vector of each unique word is the same across all documents in the training corpus. d_i can be considered as a special word vector which summarises the topic of the document and contains information missed by individual word vectors. The document vector d_i and word vectors $\{v_{i,j}\}$ are concatenated or averaged, and are then trained by a linear classifier to predict the next word in a context window. Distributed bag-of-words is similar to the skip-gram model described in the word2vec section. Instead of using the one-hot vector of a centre word as input, DBOW uses an initialised document vector as input, and updates the vector by training a linear projection network to classify whether a randomly sampled word from the document belongs to a randomly sampled text window cut from the document. DBOW consumes less memory than DM because it does not require to store and update word vectors. From previous studies, DM alone works well for multiple tasks, but its combination with DBOW is more consistent across several tasks (Le and Mikolov, 2014).

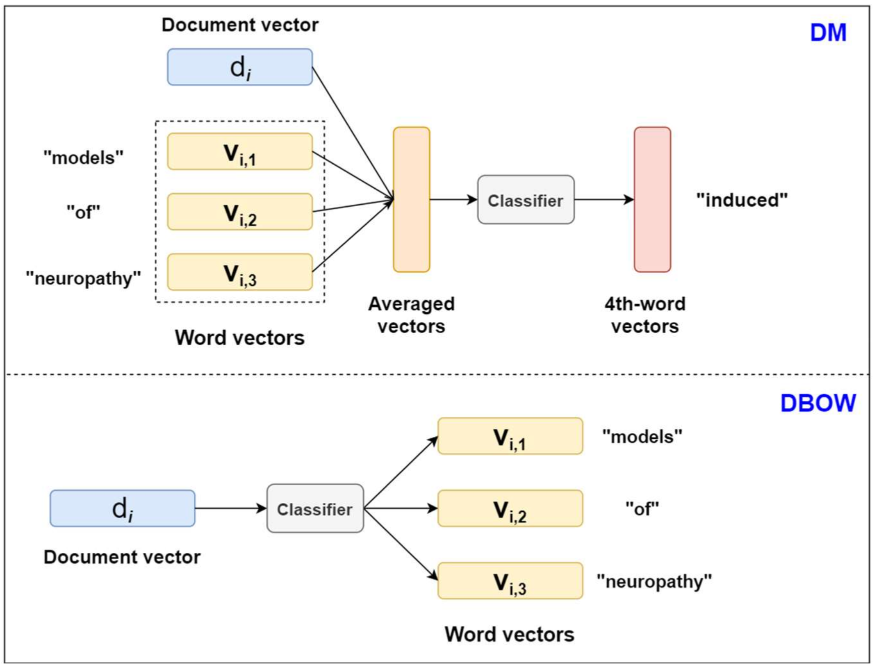


Figure 2.3: An example of distributed memory and distributed bag-of-word model (with window size 3). The classifier is a linear neural network.

All three text representation methods are explored for baseline classification models, and word2vec is used for neural models. Text representation for BERT models is described later in Section 2.3.4.1.

2.3.2 Baseline Models

I explore three baseline models for the classification task: Support Vector Machine (SVM), Logistic Regression (LogReg) and Random Forest (RF). I define the representation vector of the i-th document in the training collection of publications as $X_i \in \mathbb{R}^d$, where d is the vocabulary size when using the bag-of-words approach, or pre-defined dimension of the continuous vector induced from word2vec or doc2vec. Support vector machine and logistic regression learn to map document vector X_i to the target label Y_i by $Y_i = WX_i$, where the weight matrix W is trained to minimise the objective function:

$$\sum_{i=1}^{N} L(Y_i, WX_i) + \lambda R(W) \tag{2.2}$$

N is the number of document, L is the loss function, λ is a constant and R is a regularisation term defined as $R(W) = \|W\|^2/2$, which serves as a penalty for a large number of features (Hastie, Trevor, Tibshirani, Robert, Friedman, 2009).

For support vector machine, L is hinge-loss and the loss of i-th document is calculated by

$$L(Y_i, WX_i) = \begin{cases} max(0, 1 - WX_i), & Y_i = 1\ (reported\ RoB) \\ 1, & Y_i = 0\ (not\ reported\ RoB) \end{cases} \tag{2.3}$$

Stochastic gradient decent algorithm (Goldberg, 2016) is used to update parameter matrix W. At one iteration step, the gradient of the loss is estimated by one record randomly sampled from the training data, and W is then updated following the rule:

$$g = \frac{\partial\left[L\left(Y_j, W^T X_j\right) + \lambda R(W)\right]}{\partial W} \tag{2.4}$$

$$\eta = \frac{1}{\lambda(t_0 + t)} \tag{2.5}$$

$$W \leftarrow W - \eta \times g \tag{2.6}$$

where j is the index of one randomly sampled record from training data, η is the decreasing learning rate by iteration step t, g is gradient, and t_0 is a heuristic constant (Bottou, 2012). It is faster and less memory-consuming, compared to the gradient descent method (Lemaréchal and Lemaréchal, 2012) which updates parameters using average gradients of all training data in one iteration step.

For logistic regression, L is log-loss and the loss of i-th document is calculated by

$$L\left(Y_i, W^T X_i\right) = \begin{cases} \log\left(1 + e^{-WX_i}\right), & Y_i = 1\ (reported\ RoB) \\ \log\left(1 + e^{WX_i}\right) & Y_i = -1\ (not\ reported\ RoB) \end{cases} \tag{2.7}$$

Limited-memory BFGS (Nocedal, 1980) is applied to update parameter matrix W, which uses a dimension-reduced approximation of the Hessian matrix of the loss function and is proved to be the best choice for log-linear models with L2 regularisation (Andrew and Gao, 2007).

Random forest is an ensemble-based method which combines several decision trees trained on sub-samples and then average results of multiple trees to avoid over-fitting (Breiman, 2001). Decision tree is a non-parametric method and the basic tree structure is displayed in Figure 2.4. For the text classification task, a decision tree first splits documents into two nodes (leaves) based on their representation vectors. Then within each node, it repeats the splitting process and forms its own branch, until a pre-defined maximum depth of trees, maximum number of leaf nodes, or the minimum number of samples required to form a leaf node, is reached.

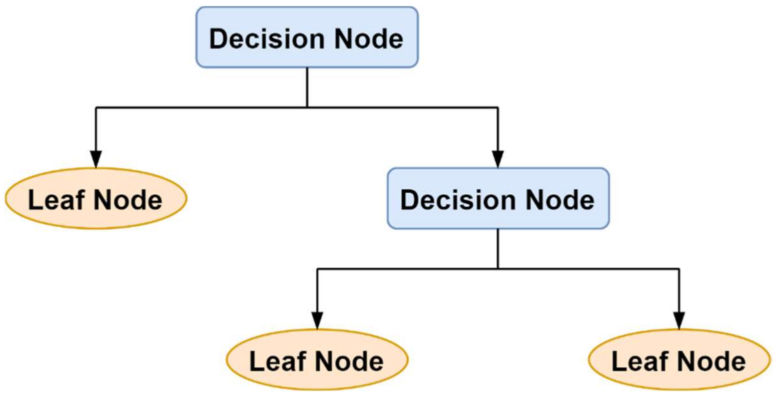


Figure 2.4: A decision tree structure.

2.3.3 Neural Networks

I explore three neural models: Convolutional Neural Network (CNN), a powerful model for text classification (Kim, 2014); Recurrent Neural Network (RNN) which is good at modelling sequential text data (Hochreiter and Schmidhuber, 1997); and Hierarchical Attention Network (HAN) which takes the hierarchical structure among word, sentence and document into consideration (Yang *et al.*, 2016).

2.3.3.1 Convolutional Neural Network

An individual document can be considered as a long-text sentence and I use the one-layer convolutional neural network structure for sentence classification

(Kim, 2014). Critical elements of this architecture are described below and shown in Figure 2.5.

Embedding layer. The CNN architecture begins with mapping each document into a representative matrix $x \in \mathbb{R}^{l \times \mathrm{d}}$ by a pre-trained word embedding $E \in \mathbb{R}^{\mathrm{v} \times \mathrm{d}}$, where each row is a word vector representing one word, l is the number of words in the document, d is the dimension of the pre-trained embedding and v is the vocabulary size which depends on the specific pre-training corpus. The shape of the weight matrices trained in CNN are fixed, but length l varies among different documents. Hence, I define a threshold s for document length, and cut the document when its length l is larger than s or pad the document with zeros when its length l is smaller than s. This guarantees the embedding matrix of every document has the same dimension ($\mathbb{R}^{s \times \mathrm{d}}$).

Convolution layer. The main characteristic of a convolution layer is the multiple filters (2D matrices) with different sizes. Let $x[i\colon j]$ denote the matrix extracted from row i to row j in the document matrix. For one document matrix $x \in \mathbb{R}^{s \times \mathrm{d}}$ and one filter $f \in \mathbb{R}^{h \times \mathrm{d}}$, where h is the filter size, the convolution layer sequentially extracts a sub-matrix which has the same dimension as filter f, and then summarise the element-wise productions between $x[i\colon i-h+1] \in \mathbb{R}^{h \times \mathrm{d}}$ and f:

$$c_i = \sum_{j=1}^{h} \sum_{k=1}^{d} x[i\colon i-h+1] \odot f \tag{2.8}$$

where $i = 1, \dots, s-h+1, c_i \in \mathbb{R}$. This generates a summarised feature vector $c = [c_1, c_2, \dots, c_{s-h+1}] \in \mathbb{R}^{s-h+1}$, by one filter f with size h. For each filter size, multiple filters are used to extract multiple features. As the example shown in Figure 2.5, a document with 8 words is mapped to a matrix by a 5-dimension word embedding, i.e. s=8, d=5. Four filters are used, where two of them have size 3 (h=3) and another two have size 4 (h=4). After the convolution operation, four feature vectors are generated, where two with 8-3+1=6 elements and another two with 8-4+1=5 elements, for each filter size respectively.

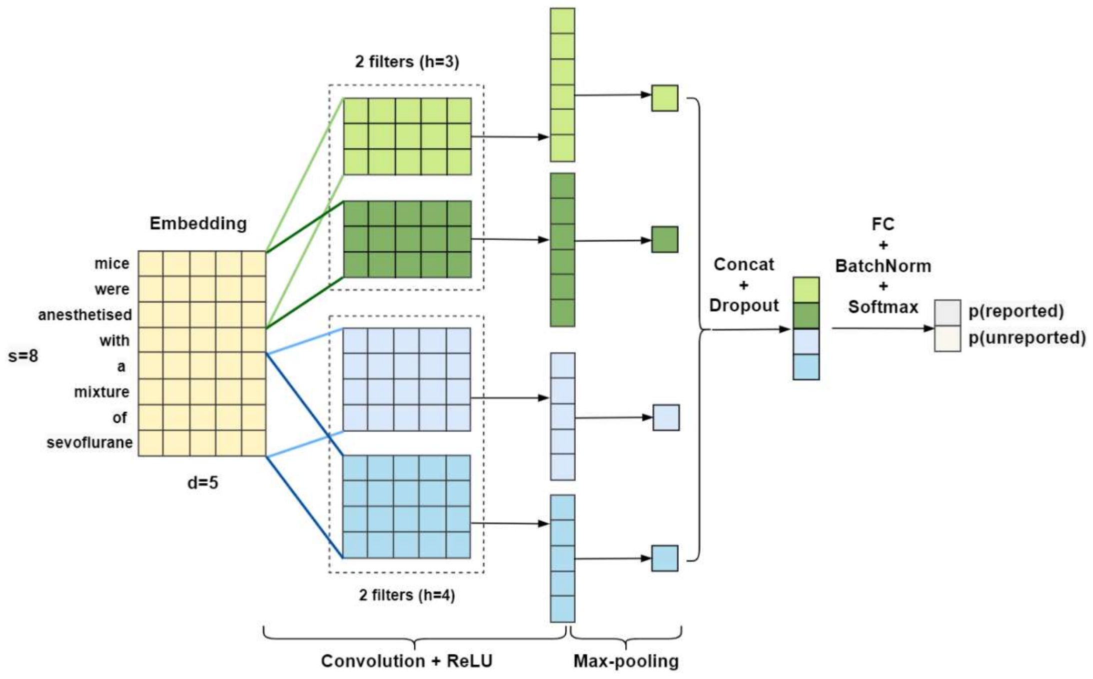


Figure 2.5: An example of convolutional neural network for document classification. A document with 8 words is mapped by a 5-dimension word embedding. Two filters size [3, 4] are used, and each of them has two filters.

These feature vectors are then passed through an activation function to induce more non-linearity to the network, and the rectified linear activation function (ReLU) is a preferred default in convolutional neural networks (Krizhevsky *et al.*, 2012), which is defined as $\mathrm{ReLU}(c) = \max(0, c) \in \mathbb{R}^{s-h+1}$.

Max-pooling layer. Max-pooling (Dumoulin and Visin, 2016) is applied to take the maximum value of each feature vector c, which represents the most important feature extracted by the corresponding filter. This layer generates q maximum scalars which are concatenated into one vector, where $q =$ total number of filters = number of different filter sizes $*$ number of filters for each filter size.

Dropout layer. Dropout serves as a regularisation strategy to prevent over-fitting (Srivastava *et al.*, 2014). It randomly ignores some 'neurons' by setting some elements in the feature vector to zeros during the training process. The dropout rate refers to the proportion of neurons being ignored.

Fully connected layer. Fully connected (FC) layer performs a linear transformation to map the m-dimensional vector from the max-pooling layer into a 2-dimensional vector $[p^1, p^0]$, which refers to the score of 'reported' and 'unreported' respectively.

Batch normalisation layer. Batch normalisation (Ioffe and Szegedy, 2015) is performed before the final output layer. 'Batch' refers to the sub-sample split from the whole training set. In one training iteration (epoch), neural networks use a batch training strategy (Goodfellow *et al.*, 2016), which splits the whole training sample into several subsets, i.e. 'mini-batches', and updates parameters after the calculation of gradients within every mini-batch. This strikes a balance between the gradient descent algorithm and stochastic gradient descent algorithm for memory, computing efficiency and convergence robustness. Batch normalisation performs normalisation within each mini-batch to reduce the dependency of parameters' distribution changes among layers, where high dependency makes it more difficult to distinguish output values between classes. For a mini-batch B including n documents, the output from the fully connected layer is $\{[p_i^1, p_i^0]\} \in \mathbb{R}^{n\times 2}$, $i = 1, \dots, n$. The mean and variance of score of each class (reported/unreported) across the mini-batch are calculated by:

$$\mu^1 = \frac{1}{n}\sum\nolimits_{i=1}^{n} p_i^1, \qquad (\sigma^1)^2 = \frac{1}{n}\sum\nolimits_{i=1}^{n}(p_i^1 - \mu^1)^2 \tag{2.9}$$

$$\mu^0 = \frac{1}{n}\sum\nolimits_{i=1}^{n} p_i^0, \qquad (\sigma^0)^2 = \frac{1}{n}\sum\nolimits_{i=1}^{n}(p_i^0 - \mu^0)^2 \tag{2.10}$$

Scores are then normalised, scaled and shifted by:

$$\widetilde{p_i^1} = \gamma\frac{p_i^1 - \mu^1}{\sqrt{(\sigma^1)^2 + \epsilon}} + \beta, \qquad \widetilde{p_i^0} = \gamma\frac{p_i^0 - \mu^0}{\sqrt{(\sigma^0)^2 + \epsilon}} + \beta \tag{2.11}$$

where γ and β are the scaled and shifted parameters to be trained.

A softmax function is then applied to convert scores to probabilities ranging from 0 to 1:

$$p_{reported} = \frac{\exp\left(\widetilde{p^1}\right)}{\exp\left(\widetilde{p^1}\right) + \exp\left(\widetilde{p^0}\right)}, \qquad p_{unreported} = \frac{\exp\left(\widetilde{p^0}\right)}{\exp\left(\widetilde{p^1}\right) + \exp\left(\widetilde{p^0}\right)} \tag{2. 12}$$

Parameters in the network are trained and updated using Adam algorithm (Kingma and Ba, 2015) to minimise the cross-entropy loss for binary classification, calculated by:

$$Loss = -[y \times log\left(p_{reported}\right) + (1-y) \times log\left(1 - p_{unreported}\right)]$$

$$= \begin{cases} -log\left(p_{reported}\right), & y = 1\ (reported\ RoB) \\ -log\left(1 - p_{unreported}\right), & y = 0\ (not\ reported\ RoB) \end{cases} \tag{2. 13}$$

Adam is a type of adaptive optimisation method, which updates different parameters with different learning rate based on the gradient values of the parameters. Parameters showing small gradient updates from previous iterations will be updated faster with a larger learning rate, and parameters with large gradient updates will be updated slower. More specifically, at an iteration time step t, it first updates two estimates m and v, which are the mean and uncentred variance of gradients g, by:

$$g \leftarrow \frac{\partial Loss}{\partial W} + \lambda \mathrm{W} \tag{2. 14}$$

$$m \leftarrow \beta_1 m + (1 - \beta_1) g \tag{2. 15}$$

$$v \leftarrow \beta_2 v + (1 - \beta_2) \mathrm{g}^2 \tag{2. 16}$$

where W represents weight parameters, λ is the regularisation constant, β_1 and β_2 are constants to control the decay rate of m and v separately. The two biased estimates are then rescaled by

$$m' = \frac{m}{1 - \beta_1^t}, \qquad v' = \frac{v}{1 - \beta_2^t} \tag{2. 17}$$

and the training parameters are updated by

$$W \leftarrow W - \frac{\alpha m'}{\sqrt{v'} + \epsilon} \tag{2. 18}$$

When gradients change slowly, the gradient variance v approximately equals to the quadratic gradient mean m'^2, so the decay value of W is close to the learning rate α; when gradients change rapidly, m'^2 is much smaller than v', so the decay value is much smaller than α, which means the parameters would be updated slower (Kingma and Ba, 2015).

2.3.3.2 Recurrent Neural Network

Recurrent neural network (RNN) is a type of neural network which builds connections over time steps (Abiodun *et al.*, 2018). It is good at modelling sequential data and capturing information from earlier or later time steps, where here a time step refers to a word or a token in the text sequence. Figure 2.6 demonstrates a basic one-layer bidirectional RNN (BiRNN) for text classification. ‘Bidirectional’ means words are modelled in both positive and reverse order, so the model can learn from the text before and after the current word.

Similar to the convolutional neural network, a document is first mapped into an embedding matrix $x \in \mathbb{R}^{s \times d}$, and vector $x_t \in \mathbb{R}^d$ is the embedding of the word at position t, where $t = 1, \ldots, \mathrm{s}$, d is the embedding dimension, and s is the document length after padding or cutting. Then in the RNN layer, the RNN cell at position t combines information from the corresponding input state x_t and the previous hidden state $h_{t-1} \in \mathbb{R}^h$, where h is the hidden dimension. The simplest structure of the RNN cell is the linear operation followed by an activation function (for one direction):

$$h_t = \sigma(W_{hh}h_{t-1} + W_{xh}x_t + b_h) \tag{2.19}$$

where $W_{xh} \in \mathbb{R}^{d \times h}$ is the weight between each pair of the input state and the hidden state, $W_{hh} \in \mathbb{R}^{h \times h}$ is the weight between each pair of the previous hidden state and the current hidden state, and σ is the activation function such as the hyperbolic tangent function, $\mathrm{Tanh}(a) = \frac{\exp(a) - \exp(-a)}{\exp(a) + \exp(-a)}$. The same operation is applied in a reversed order to obtain another hidden state h_t', and the output of the RNN cell at position t is the concatenation of two hidden states $[h_t, h_t'] \in$

$\mathbb{R}^{2h}$. This generates a hidden matrix $\in \mathbb{R}^{s \times 2h}$ for a sequence with s words, and to reduce dimension, the output can be obtained from the last hidden state $[h_s, h'_1] \in \mathbb{R}^{2h}$, or element-wise maximum or mean of all hidden states $[\overline{h_t}, \overline{h'_t}]$. The rest of the operations to obtain probabilities for two classes (reported/unreported) are similar to what has been described in the convolutional neural network (see Section 02.3.3.1). RNN can have multiple hidden layers, which is supposed to capture more complicated text information.

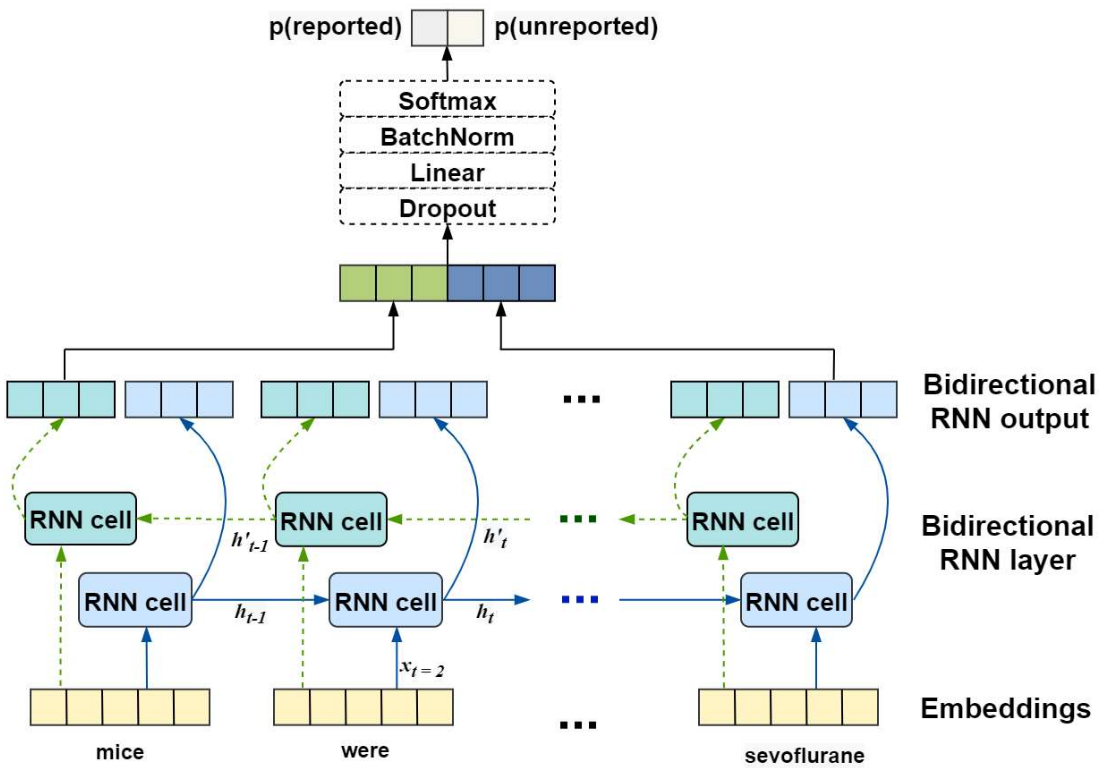


Figure 2.6: The architecture of bidirectional recurrent neural network for document classification.

RNN can handle any-length texts and it is supposed to catch the memory of previous or future steps. However, if the sequence is very long, it is difficult to keep the information from very earlier steps to later steps because of the exploding or vanishing gradient problem (Pascanu *et al.*, 2012). Following the logic in RNN, the weight matrices are multiplied by themselves multiple times which depends on the sequence length, so the gradients of the loss for a long sequence would expand to very large values (exploding gradient) or approach to zero (vanishing gradient), which stops the network learning efficiently. To solve this ‘short-memory’ problem, two variants of RNN cell, Long Short-Term Memory (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (Chung *et al.*, 2014) are developed. They are supposed to be used in the same chain structure in Figure 2.6, but the module for each time step is different.

Long Short-Term Memory (LSTM)

The basic RNN cell passes the hidden state and the current input state through the simple linear and Tanh operation, while LSTM has a cell state c_t for time step t and three gates (forget gate, input gate and output gate) to control information which should be flowed straight, forgotten, stored and updated to the next step (Hochreiter and Schmidhuber, 1997), see the grey area in Figure 2.7. Both the forget gate (block in green dash line) and input gate (block in red dash line block) check the information from the previous hidden state h_{t-1} and current input state x_t, and generate a value between 0 and 1 using the sigmoid function $\text{Sigmoid}(a) = \frac{1}{1+\exp(-a)}$. The forget gate decides how much information should be forgotten in the previous cell state by

$$f_t = \text{Sigmoid}\left(W_{fh}h_{t-1} + W_{fi}x_t + b_f\right) \tag{2.20}$$

where less information will be forgotten when the value is closer to 1. The input gate decides how much new information should be stored in the new cell state by

$$i_t = \text{Sigmoid}(W_{ih}h_{t-1} + W_{ii}x_t + b_i) \tag{2.21}$$

Then the cell state is updated by combining the information kept from the forget gate and new information selected from the input gate:

$$c_t = f_t * c_{t-1} + i_t * \mathrm{c}'_t \tag{2.22}$$

where $\mathrm{c}'_t = \text{Tanh}(W_{ch}h_{t-1} + W_{ci}x_t + b_c)$, which decides what new information will be updated to the cell state. The output gate (block in brown dash line) decides the portion of information for output:

$$o_t = \sigma(W_{th}h_{t-1} + W_{ti}x_t + b_o) \tag{2.23}$$

The new hidden state is then updated by multiplying the portion from the output gate and current cell state through the Tanh operation:

$$h_t = o_t * \text{Tanh}\,(c_t) \tag{2.24}$$

W and b in the above equations are parameters trained in the network.

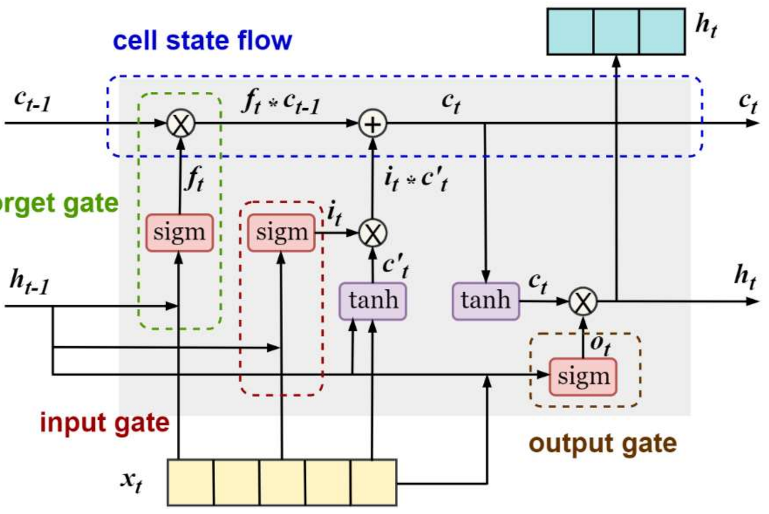


Figure 2.7: The architecture of a long short-term memory block (one direction).

Gated Recurrent Unit (GRU)

GRU (Chung et al., 2014) is a simplified variation of LSTM which converts the forget gate and input gate into a reset gate r_t and update gate z_t, and merges the output gate and cell state together, see Figure 2.8.

The reset gate r_t decides how important the previous hidden state h_{t-1} is to the next step hidden state h_t by:

$$r_t = \text{Sigmoid}(W_{rh}h_{t-1} + W_{ri}x_t + b_r) \tag{2.25}$$

A temporary hidden state h'_t is calculated by combining the current input state and the important portion from the previous hidden state by:

$$h'_t = \text{Tanh}(r_t * W_{hh}h_{t-1} + W_{hi}x_t + b_h) \tag{2.26}$$

The update gate decides what should be kept from the previous hidden state h_{t-1} and what should be updated from the new hidden state h_t':

$$z_t = \text{Sigmoid}(W_{zh}h_{t-1} + W_{zi}x_t + b_z) \tag{2.27}$$

Then the hidden state is updated by combining the information from the previous hidden state h_{t-1} and the temporary hidden state h_t'. When the output value z_t from update gate is 1, the hidden state is entirely copied from the previous hidden state; when z_t is 0, the hidden state is entirely updated from the temporary hidden state:

$$h_t = z_t * h_{t-1} + (1 - z_t) * h_t' \tag{2.28}$$

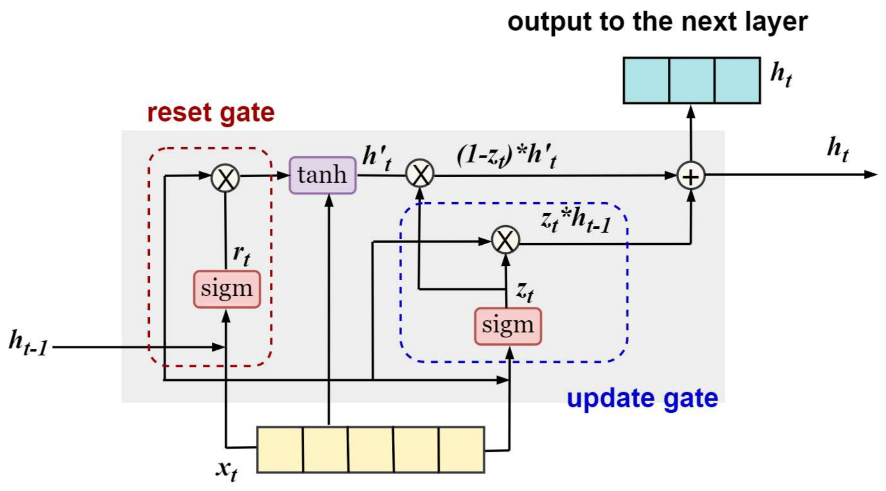


Figure 2.8: The architecture of a gated recurrent unit block (one direction).

RNN with Attention

In the general RNN structure, the output from the hidden layer is obtained by simply taking the hidden state of the last RNN cell (one cell in the unidirectional RNN or two cells in the bidirectional RNN), which lose some information from other RNN cells; or averaging hidden states of all RNN cells, which treats words at different positions equally. However, words contribute differently and the same word in different positions also plays a different role in the decision of the classification. As the example shown in Figure 2.9, ‘committee’ is the most important word in the text piece, and the word ‘animals’ in the fifth line and seventh line contributes differently to the decision of the reporting of compliance of animal welfare regulations. Therefore, in my implementation, I add an attention module after the RNN hidden layer, which is analogous to the word-level weights used in the memory networks for machine translation (Bahdanau et al., 2015). It should be noted that the term ‘attention’ here is different from the ‘self-attention’ (Vaswani *et al.*, 2017) in transformer models (see Section 0). The architecture of RNN with attention for document classification is displayed in Figure 2.10.

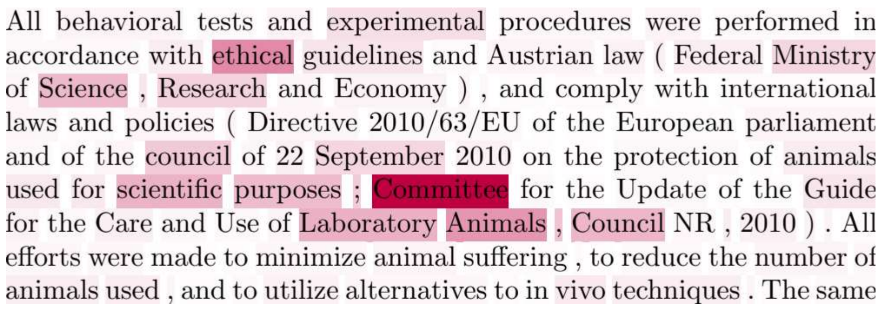

All behavioral tests and experimental procedures were performed in accordance with ethical guidelines and Austrian law (Federal Ministry of Science , Research and Economy) , and comply with international laws and policies (Directive 2010/63/EU of the European parliament and of the council of 22 September 2010 on the protection of animals used for scientific purposes ; Committee for the Update of the Guide for the Care and Use of Laboratory Animals , Council NR , 2010) . All efforts were made to minimize animal suffering , to reduce the number of animals used , and to utilize alternatives to in vivo techniques . The same

Figure 2.9: A example text piece from a publication which reported compliance of animal welfare regulations. The colour demonstrates the importance of the word in the classification decision and deeper colour means the word has a larger contribution to the decision.

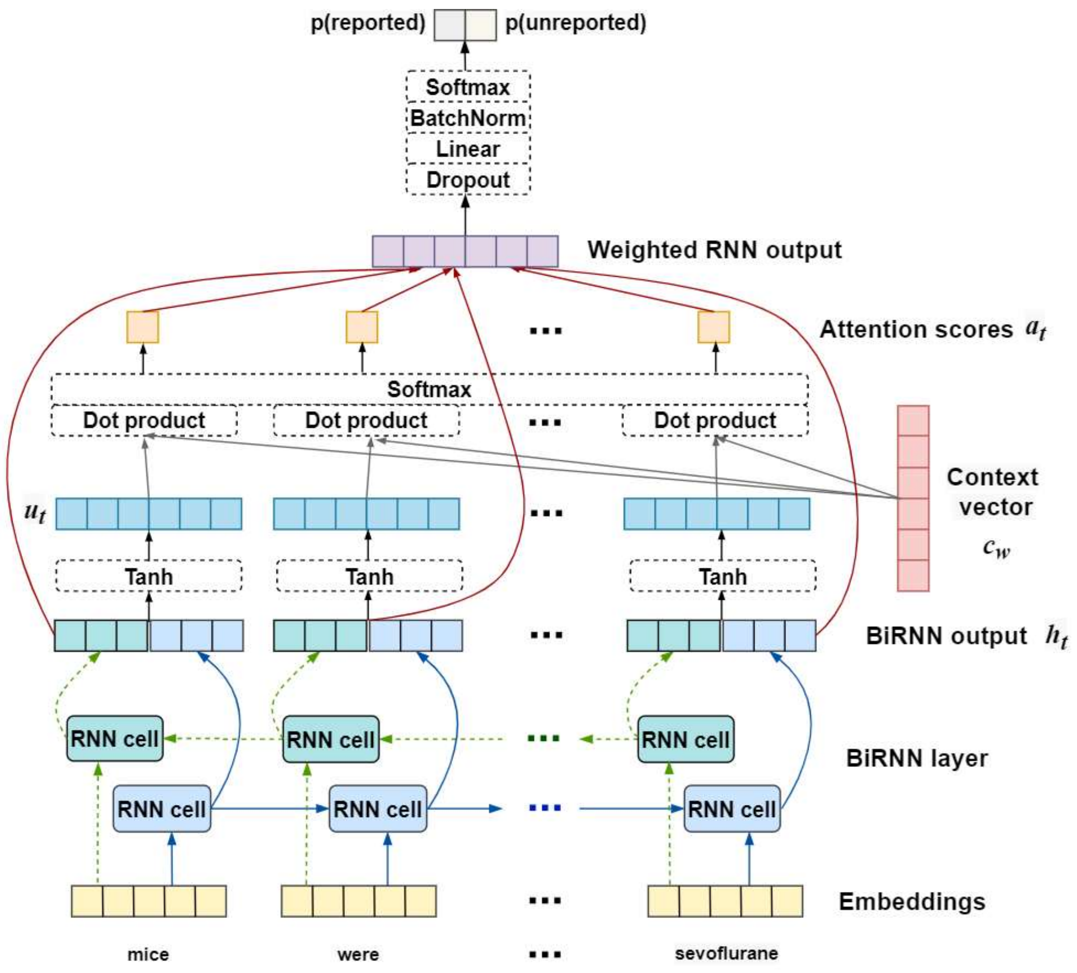


Figure 2.10: The architecture of bidirectional recurrent neural network with attention mechanism for document classification.

Considering a document with s words, after the embedding layer and the bidirectional RNN hidden layer, the output of word at position t is the hidden state $h_t \in \mathbb{R}^{2h}$, which is passed to the Tanh function to obtain a new hidden representation u_t:

$$u_t = \mathrm{Tanh}(W_w h_t + b_w), \quad u_t \in \mathbb{R}^{2h},\ t = \{1, \dots, \mathrm{s}\} \tag{2.29}$$

A global word context vector $c_w \in \mathbb{R}^{2h}$ is initialised to represent the whole document. The word attention score is calculated by multiplying the new hidden state u_t and the context vector c_w to measure the importance of the word at position t in the whole document, and then normalised by the Softmax function:

$$a_t = \mathrm{Softmax}(u_t \cdot c_w), \qquad a_t \in [0,1] \tag{2.30}$$

The summarisation of the weighted hidden states $\sum_{t=1}^{s} h_t \cdot a_t \in \mathbb{R}^{2h}$ is then sent to the output layers. The training and optimisation strategies follow the same rule described in the convolutional neural network (Section 2.3.3.1).

2.3.3.3 Hierarchical Attention Network

In the document classification task, words contribute differently to an individual sentence and sentences contribute differently in the whole document. Hierarchical Attention Network (HAN) is proposed to imitate the hierarchical structure of documents, which has two levels of attention mechanisms applied at the word-level and sentence-level (Yang *et al.*, 2016). The model architecture is shown in Figure 2.11.

I assume a document has L sentences, and the i-th sentence has T words (the value of T varies in different sentences). For the i-th sentence, T word hidden states $h_{it} \in \mathbb{R}^{2h\_w}, t \in [1, T]$ are obtained from the bidirectional GRU layer, where h_w is the dimension of word hidden states for one direction.

In the word-level attention module, which is similar to the attention illustrated in Figure 2.10, a new hidden representation u_{it} for the t-th word in the i-th sentence is obtained by

$$u_{it} = Tan\,h(W_w h_{it} + b_w), \;\; u_{it} \in \mathbb{R}^{2h_w}, \;\; t \in [1, T] \tag{2.31}$$

A local word context vector $c_w \in \mathbb{R}^{2h\_w}$ is initialised to represent the i-th sentence. The word attention scores in the i-th sentence are calculated by multiplying u_{it} and c_w to measure the importance of the t-th word in the i-th sentence, and are then normalised by the Softmax function:

$$a_{it} = \mathrm{Softmax}(u_{it} \cdot c_w), \quad a_{it} \in [0,1], \;\; t \in [1, T] \tag{2.32}$$

In the sentence-level attention module, the sentence representations are first calculated by summarising the weighted word attention scores for each sentence:

$$s_i = \sum\nolimits_{t=1}^{T} h_{it} \cdot a_{it}, \qquad s_i \in \mathbb{R}^{2h_w}, i \in [1, L] \tag{2.33}$$

Similar to the word-level attention module, sentence hidden states $h_i \in \mathbb{R}^{2h\_s}, i \in [1, L]$ are obtained from the bidirectional GRU layer, where h_s is the dimension of sentence hidden states for one direction. Then the new sentence-level hidden representations are calculated from the Tanh operation:

$$u_i = \mathrm{Tanh}(W_s h_i + b_s), \qquad u_i \in \mathbb{R}^{2h_s}, i \in [1, L] \tag{2.34}$$

A global sentence context vector $c_s \in \mathbb{R}^{2h\_s \times 2h\_s}$ is initialised to represent the whole document and multiplied by the new sentence-level hidden representation to measure the importance of each sentence in the whole document:

$$a_i = \mathrm{Softmax}(u_i \cdot c_s), \quad a_i \in [0, 1], \quad i \in [1, L] \tag{2.35}$$

Then the document representations are calculated and sent to the output layers, by:

$$d = \sum\nolimits_{t=1}^{L} h_i \cdot a_i, \qquad s_i \in \mathbb{R}^{2h_s}, i \in [1, L] \tag{2.36}$$

The training and optimisation strategies follow the same rule described in the convolutional neural network (section 2.3.3.1).

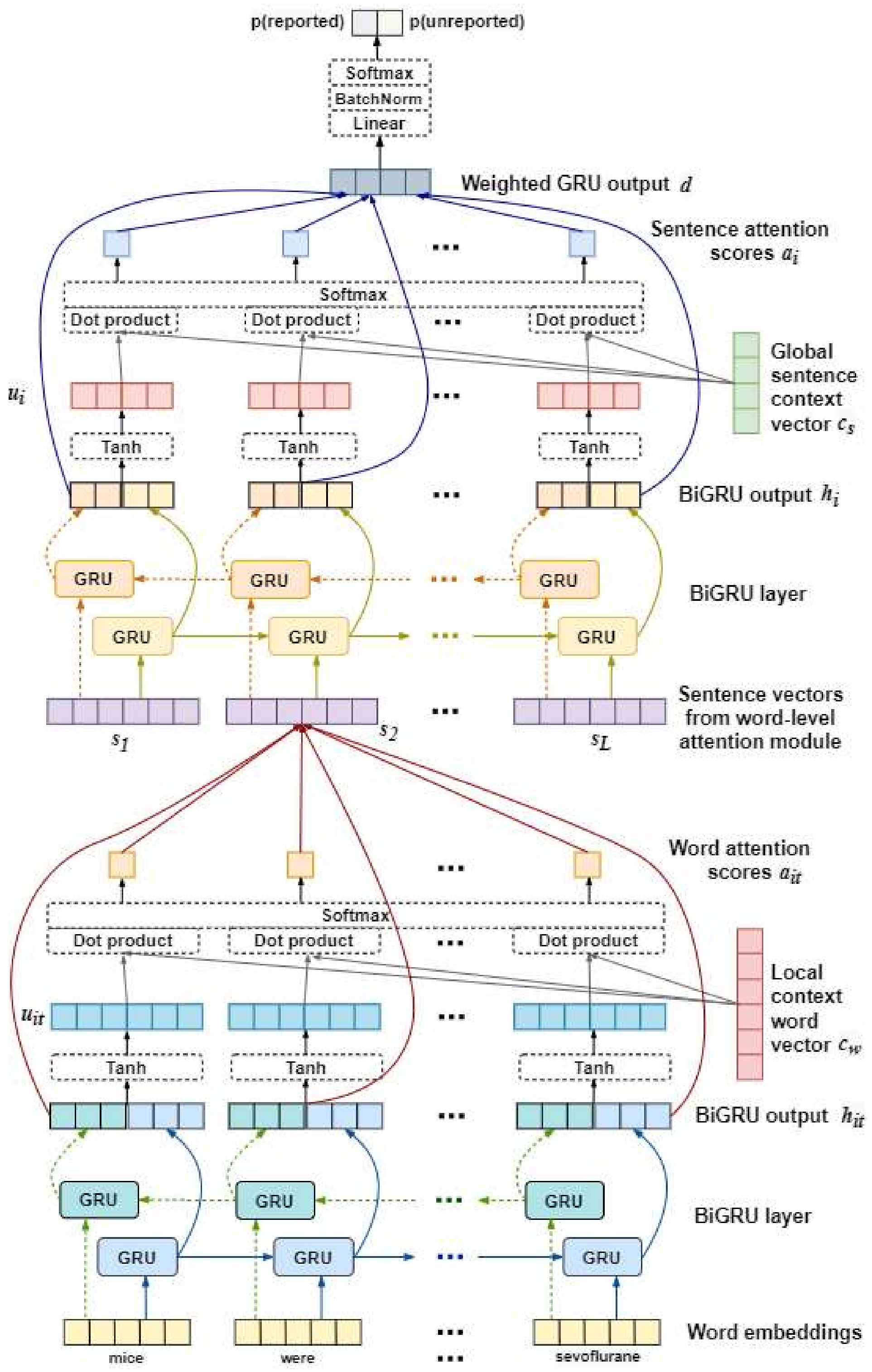


Figure 2.11: The architecture of hierarchical attention network for document classification. i is the sentence index of the document and t is the word index of the i-th sentence. In the word-level attention module, the figure displays the part for the second sentence only ($i = 2$).

2.3.4 BERT with Two Strategies

One limitation of word embeddings like word2vec is that the representation vector of the same word is fixed and independent, no matter what the context words are. To solve this issue, contextualised word representation models like ELMo (Peters *et al.*, 2018) and BERT (Devlin *et al.*, 2018) are proposed. ELMo generates the contextual word embeddings by training a bidirectional LSTM to predict the next word in the sequence; while BERT extracts the contextualised embeddings by training the bidirectional encoders from the Transformer (Vaswani *et al.*, 2017) to finish two language modelling tasks: randomly masked words prediction and next sentence prediction. Transformers can learn long-range dependency better than LSTM models (Vaswani *et al.*, 2017) and many state-of-the-art models for multiple NLP tasks are created based on transformers. The next section explains the structure of transformer encoders used in BERT and how I use BERT for risk of bias classification in long documents.

2.3.4.1 Premise: Transformer Encoder and BERT

Transformer encoders in BERT mainly consist of twelve identical blocks and each block contains two sub-layers: a multi-head self-attention layer and a feed-forward neural network layer. Each sub-layer has a residual connection around and is followed by the layer normalisation operation (Vaswani *et al.*, 2017), see Figure 2.12. Similar to the previous neural models, considering a document with s tokens (after padding/cutting), each input token is mapped to an embedding vector $x_i \in \mathbb{R}^{d_{embed}}, i \in [0, s-1]$. Positional encoding vectors are added to the input vectors to inject the order information in the sequence:

$$x_i \leftarrow x_i + \sum_{j=0}^{d_{embed}-1} PE_{i,j} \tag{2.37}$$

$$PE_{i,j} = \begin{cases} \sin\left(\frac{i}{10000^{j/d_{embed}}}\right), & j = 0,2,4,\dots \\ \cos\left(\frac{i}{10000^{j-1/d_{embed}}}\right), & j = 1,3,5,\dots \end{cases} \tag{2.38}$$

where j is the element index in the embedding vector.

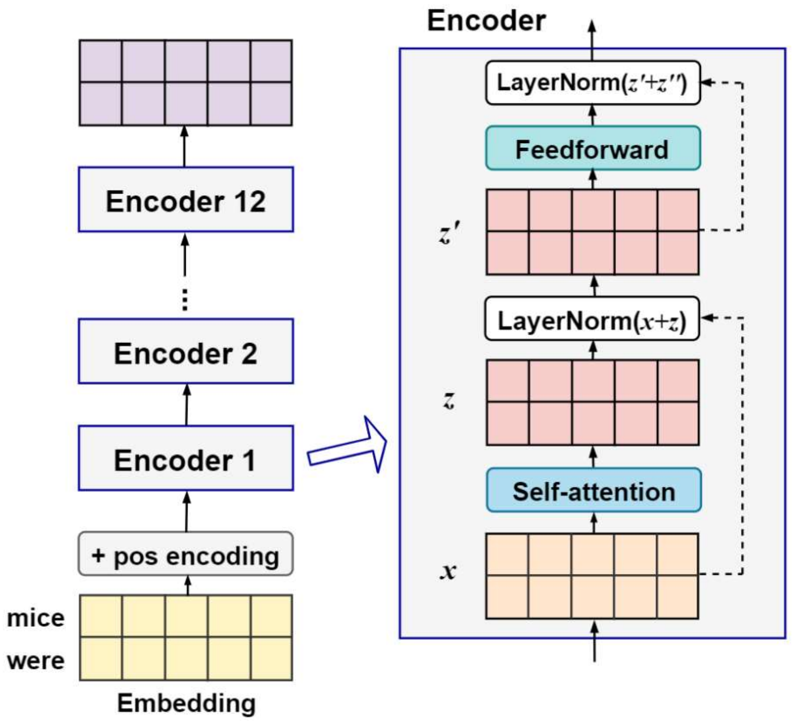


Figure 2.12: The architecture of transformer encoders. Dash lines represent the residual connection.

The self-attention calculation is illustrated in Figure 2.13. For a document with input representation $X \in \mathbb{R}^{s \times d_{embed}}$, three matrices Q, K and V are calculated by

$$Q = XW^Q \in \mathbb{R}^{s \times d_k}, \qquad W^Q \in \mathbb{R}^{d_{embed} \times d_k} \tag{2.39}$$

$$K = XW^K \in \mathbb{R}^{s \times d_k}, \qquad W^K \in \mathbb{R}^{d_{embed} \times d_k} \tag{2.40}$$

$$V = XW^V \in \mathbb{R}^{s \times d_v}, \qquad W^V \in \mathbb{R}^{d_{embed} \times d_v} \tag{2.41}$$

where the i-th row of Q, K and V refers to the query vector, key vector and value vector of the i-th token respectively; d_k and d_v are the dimension of the key vector and value vector; W^Q, W^K and W^V are initialised weight matrices. The single-head attention matrix is then computed by

$$Z_{single} = \mathrm{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V \in \mathbb{R}^{s \times d_v} \tag{2.42}$$

Vaswani et al pointed out that repeating the attention calculation with different linear projections multiple times, i.e. multi-head attention, improves the performance (Vaswani *et al.*, 2017). After the self-attention layer with h heads, the attention matrices are concatenated from different heads and are then multiplied by another initialised weight matrix $W \in \mathbb{R}^{hd_v \times d_{embed}}$, to generate the output matrix $Z \in \mathbb{R}^{s \times d_{embed}}$. The followed layer normalisation is performed over the summation of X and Z, and the output are fed to the next sub-layer, i.e. a position-wise feedforward network for each token:

$$Z' = \text{LayerNorm}(X + Z) \in \mathbb{R}^{s \times d_{embed}} \tag{2.43}$$

$$Z'' = \text{Feedforward}(Z') = \text{Linear}(\text{ReLU}\ (\text{Linear}(Z'))) \in \mathbb{R}^{s \times d_{embed}} \tag{2.44}$$

Then the output after the first encoder block is $\text{LayerNorm}(Z' + Z'') \in \mathbb{R}^{s \times d_{embed}}$, which will be sent to the next identical encoder block. After repeating the procedure with the rest eleven encoder blocks, the new representation vectors of each token are generated as the final output.

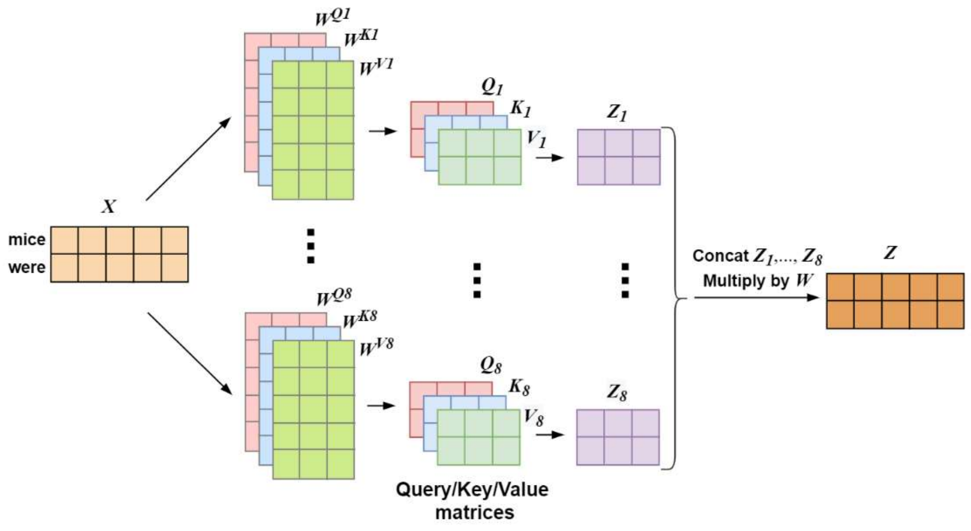
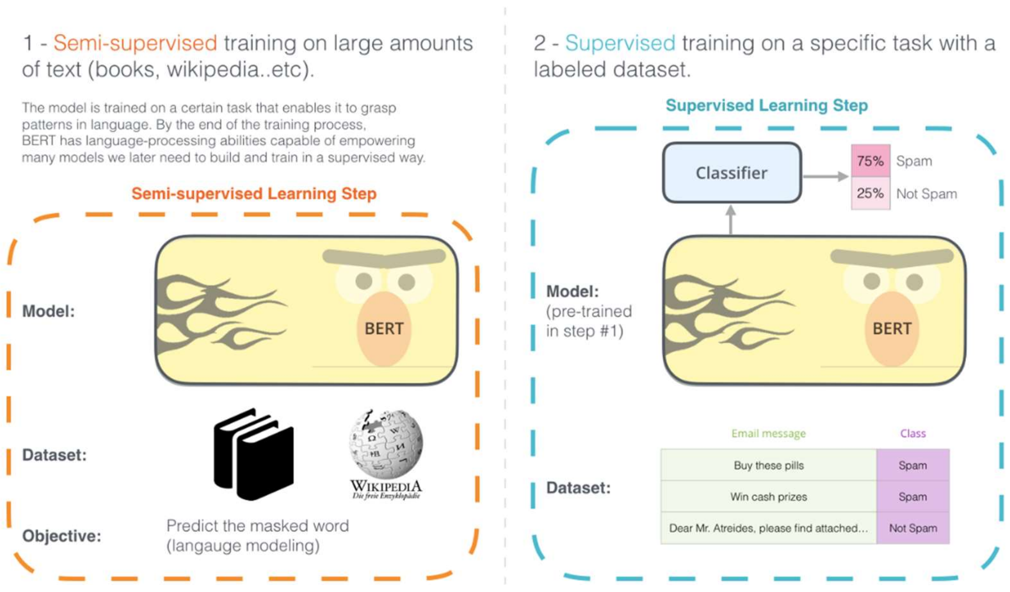


Figure 2.13: The illustration of calculations in the multi-head self-attention layer. In this example, the document length is 2, embedding dimension is 5, dimension of key and value vectors is 3, and the number of heads is 8.

BERT is developed with two stages: the pre-training stage and fine-tuning stage, as Figure 2.14 shows. In the pre-training stage, BERT trains transformer encoders on the collection of BooksCorpus (0.8 billion words) and English Wikipedia (2.5 billion words). In the fine-tuning stage, the same encoder architecture is used and initialised with the weights from the pre-training stage, and all the weights are then fine-tuned on the downstream task dataset (Devlin *et al.*, 2018). The pre-training weights are expensive to train but they are made freely available to the public, so I can download them and only need to fine-tune models on the custom dataset.

Previous work indicates the domain corpus used for pre-training affects the performance of the downstream task (Beltagy *et al.*, 2019). Since the risk of bias assessment task focus on the biomedical publications, I use BioBERT to initialise models, which applies the same architecture in BERT and is pre-trained on the combinations of text corpora including BookCorpus, English Wikipedia, PubMed abstracts (4.5 billion words) and PubMed Central full-text articles (13.5 billion words) (Lee *et al.*, 2019).



1 - Semi-supervised training on large amounts of text (books, wikipedia..etc).
The model is trained on a certain task that enables it to grasp patterns in language. By the end of the training process, BERT has language-processing abilities capable of empowering many models we later need to build and train in a supervised way.
Semi-supervised Learning Step
Model:
BERT
Dataset:
Wikipedia
Objective:
Predict the masked word (langauge modeling)
2 - Supervised training on a specific task with a labeled dataset.
Supervised Learning Step
Classifier
75% Spam
25% Not Spam
Model: (pre-trained in step #1)
BERT
Dataset:
Email message
Class
Buy these pills
Spam
Win cash prizes
Spam
Dear Mr. Atreides, please find attached…
Not Spam

Figure 2.14: Two development stages of BERT (Alammar, 2018)

The input embeddings of BERT are the sum of three embeddings: token embeddings, segment embeddings and position embeddings. For token embeddings, BERT uses WordPiece (Wu et al., 2016) with a 30,000-token vocabulary for tokenisation. It is a subword tokenisation algorithm which handles rare words better than 'pure' word embeddings and more efficiently than character embeddings. For example, 'randomisation' can be decomposed into two subwords 'random' and 'isation' which appear more frequently than 'randomisation', while keeping the meaning of 'randomisation' at the same time. In BERT's tokenizer, tokens of the two subwords are marked as ['random', '##isation'], where '##' means the rest characters 'isation' should be concatenated to the previous token 'random' when converting subword tokens back to the complete word in the decoding or prediction stage. Segment embeddings work for a special segmentation token '[SEP]', which are designed for the 'two-sequences' tasks to indicate the first and the second portions of the sequence pair. For example, in a question answering task, the segment embeddings of the question tokens are marked as [A] while the segment embeddings of the context tokens are marked as [B]. In a text classification task, segment embeddings do not have any influence on the model result, but they are kept as a default input formatting of BERT. In addition, the first token of every sequence is a special classification token '[CLS]' and the corresponding embedding can be used to generate the decision of the sequence classification, see the embedding layer in Figure 2.15. Position embeddings provide the order information of the tokens in the sequence as described in Section 2.3.4.1

2.3.4.2 Applying BERT with Two Strategies

One drawback of BERT is that it can only accept embeddings of maximum 512 tokens as input, which limits the usage for tasks with long documents. There are other transformer models designed for long documents, such as Transformer XL (Dai *et al.*, 2019) which has no limitation for sequence length and Longformer (Beltagy *et al.*, 2020) which can process a maximum of 4096 tokens. However, they are still computationally expensive and very memory-consuming from my initial experiments on full-text publications containing 5000 tokens on average. To solve this issue, I propose two strategies as described below.

BERT with Document Chunk Pooling (BERT-DCP). I split documents into text chunks, apply BioBERT to each chunk, and pool the hidden states from different chunks using multiple strategies. This is similar to the structure applied in the classification of clinical notes for patient smoking status (Mulyar *et al.*, 2019), with some modifications (see Figure 2.15). Module in the red/black/blue box refers to the linear head, convolution head and LSTM head separately. After the WordPiece tokenisation, a document with s tokens is split into $m = \lceil s/(512 - 2) \rceil = \lceil s/510 \rceil$ chunks (excluding the classification token '[CLS]' and the segmentation token '[SEP]'). The input representation of the document is $\mathrm{X} \in \mathbb{R}^{m \times 512 \times h}$, where h is the hidden dimension throughout the embedding layer and encoder layers in BioBERT. Instead of taking the hidden states from the last encoder layer, I perform the average pooling operation over several encoder layers to obtain the output (see the 1st pooling layer in Figure 2.15). I then summarise across tokens within each chunk using five different options: 1) max-pooling, 2) average-pooling, 3) concatenate output from max-pooling and average-pooling, 4) use hidden states of the '[CLS]' token, and 5) concatenate hidden states of all tokens (see the 2nd pooling layer in Figure 2.15). The output dimension from the second pooling layer is $[m, 1, h]$ for option 1), 2) and 4), $[m, 2, h]$ for option 3), and $[m, 512, h]$ for option 5). After two pooling layers, I add a head layer for the downstream classification task, and I explore three different heads (linear/convolution/LSTM). The convolution head and LSTM head use the same architecture as described in the previous sections (see Section 2.3.3.1 and 2.3.3.2 respectively). Unlike the convolution head or LSTM head, the linear head cannot handle sequences with different lengths, so I add another pooling layer to obtain the fixed-dimension output. The pooling methods use the same options applied in the second pooling layer, with the exclusion of 'concatenate hidden states of all tokens', because it does not generate a fixed-dimension output.

BERT with Sentence Extraction (BERT-SE). Instead of using the full-text documents as input, I extract the most relevant sentences to the risk of bias description. I first use scispaCy (Neumann *et al.*, 2019) to split a document into sentences, and then apply SentenceTransformers (Reimers and Gurevych, 2019) to obtain a representation vector for each individual sentence. I also feed the description sentence of each risk of bias item to SentenceTransformers to obtain the corresponding ‘RoB item vector’. The RoB description sentences are same across different documents. For example, the description sentence for random allocation is “animals are randomly allocated to treatment or control groups at the start of the experiment” (see description sentences for other items in section 0). Therefore, the five ‘RoB item vectors’ are also same across documents. Next, for each individual document, I calculate the cosine similarity score between each sentence vector and the RoB vector of one item. I take the first k sentences with the highest similarity scores, i.e. the most k relevant sentences, to form a new shorter passage. I then fine-tune DistilBERT (a smaller, faster and lighter version of BERT) (Sanh *et al.*, 2019) with a linear/convolution/LSTM head on the new passage, to generate the probabilities of risk of bias reporting. The sentence extraction process is unsupervised and is independent of the actual training process.

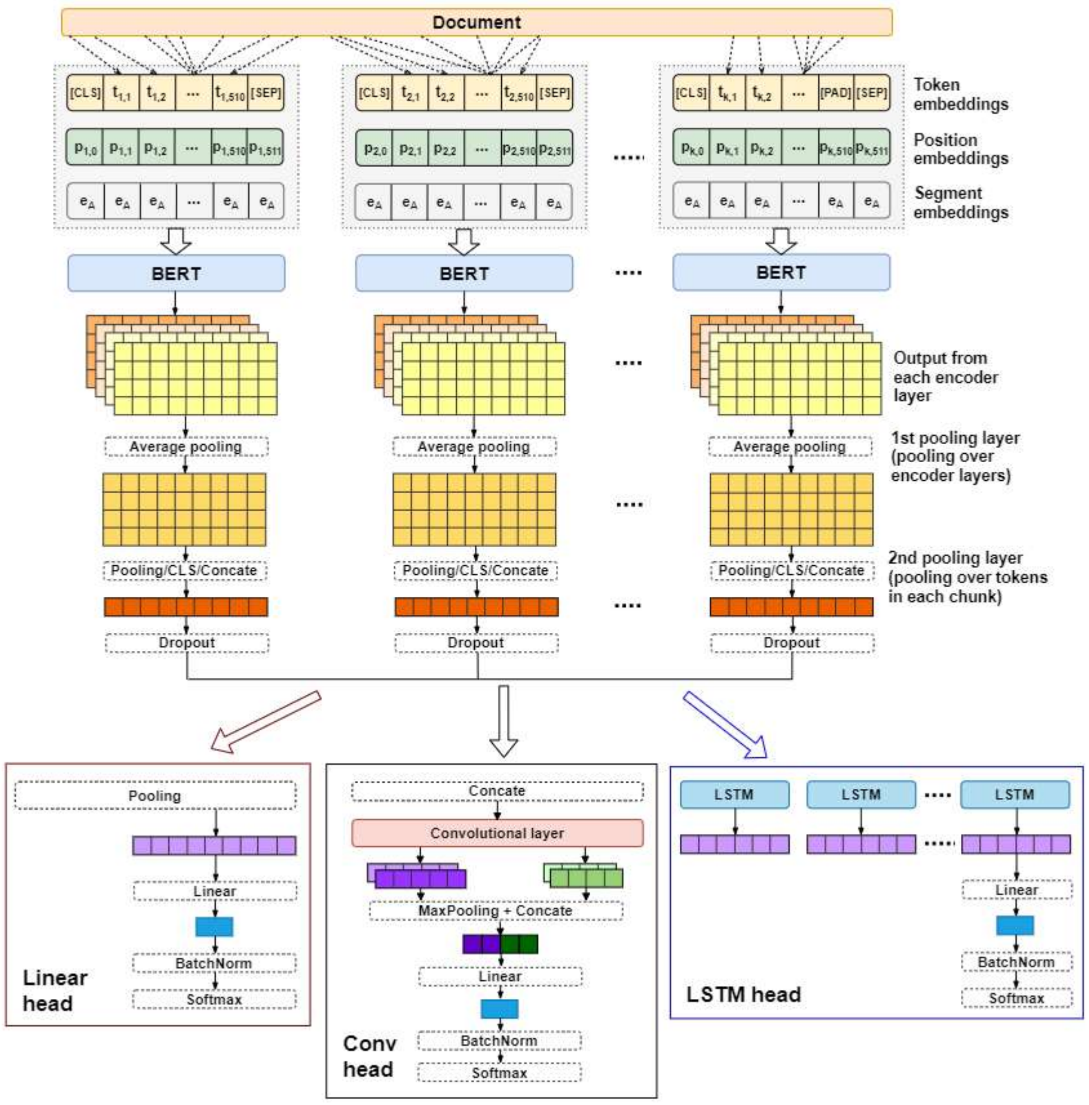

Document
[CLS] $t_{1,1}$ $t_{1,2}$ ... $t_{1,510}$ [SEP]
$p_{1,0}$ $p_{1,1}$ $p_{1,2}$ ... $p_{1,510}$ $p_{1,511}$
e_A
[CLS] $t_{2,1}$ $t_{2,2}$ ... $t_{2,510}$ [SEP]
$p_{2,0}$ $p_{2,1}$ $p_{2,2}$ ... $p_{2,510}$ $p_{2,511}$
[CLS] $t_{k,1}$ $t_{k,2}$ ... [PAD] [SEP]
$p_{k,0}$ $p_{k,1}$ $p_{k,2}$ ... $p_{k,510}$ $p_{k,511}$
Token embeddings
Position embeddings
Segment embeddings
BERT
Output from each encoder layer
Average pooling
1st pooling layer (pooling over encoder layers)
Pooling/CLS/Concate
2nd pooling layer (pooling over tokens in each chunk)
Dropout
Pooling
Linear
BatchNorm
Softmax
Linear head
Concate
Convolutional layer
MaxPooling + Concate
Linear
BatchNorm
Softmax
Conv head
LSTM
Linear
BatchNorm
Softmax
LSTM head

Figure 2.15: The architecture of BERT with document chunk pooling strategy for long documents classification. The red/black/blue block on the bottom refers to the option of the linear/convolution/LSTM head on top respectively.

In the training procedure, I apply gradient clipping with a maximum norm 0.1 to rescale gradients, where all gradients are concatenated into a single vector and normalised, then the elements in the vector larger than 0.1 are replaced by 0.1. This can prevent vanishing/exploding gradients issues and accelerate training (Zhang et al., 2019). I use gradient accumulation every 16 steps (mini-batches) to reduce memory consumption, where the model parameters are not updated from gradients after every mini-batch, but from gradients accumulated after every 16 mini-batches. Parameters are trained to minimise the cross-entropy loss using AdamW algorithm (Loshchilov and Hutter, 2017). I use the slanted triangular learning rate scheduler (Howard and Ruder, 2018), where the learning rate is linearly increased from zero to a threshold value over the warm-up training steps, then linearly drops to zero in the following steps.

2.3.5 Evaluation metrics

For all classification models described above, four metrics are reported for performance evaluation, which are calculated by:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \tag{2.45}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \tag{2.46}$$

$$\text{F1} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}} \tag{2.47}$$

$$\text{Specificity} = \frac{\text{True Negative}}{\text{True Negative} + \text{False Positive}} \tag{2.48}$$

where ‘True Positive’ is the number of records which report the risk of bias item and are predicted as reported; ‘True Negative’ is the number of records which do not report the risk of bias item and are predicted as unreported; ‘False Positive’ is the number of records which do not report the risk of bias item but are predicted as reported; and ‘False Negative’ is the number of records which report the risk of bias item but are predicted as unreported. Recall (or sensitivity) measures the portion of records which are identified as reported among all truly reported records. Precision measures the portion of truly reported records among all records identified reported. Specificity measures the portion of records identified as non-reported among all non-reported records. F1 is the harmonic mean of recall and precision, and I use it as the main metric for hyperparameter and model selection.

For evaluation in batch training for neural models and BERT models, loss and four metric scores are calculated within every mini-batch, and the final training loss, F1 and other scores are averaged by the number of mini-batches respectively.

2.3.6 Comparison to Regular Expression

A regular expression (Regex) tool developed for preclinical risk of bias assessment (Bahor et al., 2016) is evaluated on my test set, and the performance are compared with that of each item's best NLP model selected from experiments of the previously described models. The Regex tool is built based on the rule-based string matching method to recognise phrases associated with the reporting of random allocation, blinded assessment of outcome, conflict of interest and animal welfare regulations in experimental animal studies. The McNemar's test (Raschka, 2018) is performed to compare how the two methods perform differently, which includes two scenarios: 1) Regex tool predicts correctly but NLP model predicts wrong, and 2) Regex tool predicts wrong but NLP model predict correctly, as shown in Table 2.3.

		NLP model	
		Correct	Wrong
Regex	Correct	N^{++}	N^{+-}
	Wrong	N^{-+}	N^{--}

Table 2.3: The confusion matrix between the Regex tool and NLP model in the McNemar's test. N represents the number of cases - its first superscript represents the Regex tool predicts correctly (+) or wrong (-), and the second superscript represents the NLP model predict correctly (+) or wrong (-).

A Chi-square statistic is computed as below:

$$\chi^2 = \frac{(|B - C| - 1)^2}{B + C} \tag{2.49}$$

to test the null hypothesis that the two methods do not perform differently.

2.4 Experiments

This section describes details of training, implementation, parameters tuning and sensitivity analysis. Processed datasets are available in the Preclinical RoB Assessment repository (osf.io/fjwx6).

2.4.1 Experiments of Baseline Models

I roughly tune parameters for three baselined models with three text representation methods, which induces nine combinations. For bag-of-words, I tune word n-gram size among {(1, 1), (1, 2), (1, 3), (2, 2), (2, 3), (3, 3)} where (p, q) means every p, p+1, …, q adjacent words are counted as features, with or without TF-IDF weighting. For word2vec, document representations are generated by averaging their word vectors as all baseline models require a single dimension vector as input. For doc2vec, I set the initial learning rate to 0.01 and it drops linearly to 0.001 during 20 training epochs. I draw 5 'negative' words in negative sampling. I average the context vectors to generate the document vector representation. I tune the dimension of feature vectors among {100, 200, 300, 400, 500}, and doc2vec training models among {DM, DBOW, concatenation of DM and DBOW}. For bag-of-words and word2vec, I tune minimum document frequency (words with a document frequency lower than this threshold are ignored) among {10, 20, 50, 100}, maximum vocabulary size (when building the corpus, only the top words ordered by term frequency are considered) among {500, 1000, 2000, 5000}. For doc2vec, I set the minimum document frequency to 10 and only tune the maximum vocabulary size among {5000, 10000} as these two parameters did not affect performance much from my initial experiments.

I assign class weights for all three classification models to solve the data imbalance issue, calculated by

$$\left[\frac{\text{number of samples}}{2 * \text{number of reported samples}}, \quad \frac{\text{number of samples}}{2 * \text{number of unreported samples}}\right] \qquad (2.50)$$

for the positive and negative class respectively. For support vector machine, I tune α which is used to multiply the $l2$ regularisation term and compute the learning rate (Pedregosa *et al.*, 2011), from 5e-10 to 1e-2. 10% of the training samples were used for early stopping to terminate training. For logistic regression, I tune the inversed value of regularisation strength among {1, 100, 1000}, which is a multiplier for loss function L and large value means less strong regularization penalty. For random forest, I set the maximum depth of an individual tree to 2. I tune the number of trees in the forest among {100, 500, 1000}, and the maximum number of features considered in the splitting process, which is selected from the total number of features (vocabulary size for bag-of-words, or dimension of document vector for wor2vec and doc2vec), $sqrt$(total number of features), or log_2(total number of features).

Bag-of-words and averaged word2vec are implemented using function CountVectorizer and TfidfTransformer from Python library scikit-learn (Pedregosa *et al.*, 2011). Doc2vec is implemented via Doc2Vec from library Gensim (Rehurek and Sojka, 2010). Support vector machine, logistic regression and random forest are implemented using SGDClassifier, LogisticRegression and RandomForestClassifier in scikit-learn respectively. The hyperparameters in all the baseline experiments are tuned in the grid search manner. All baseline experiments are conducted using a CPU with 16 cores and codes are available at github.com/qianyingw/rob-chia.

Results

After tuning parameters for each risk of bias item, I select the model with the highest F1 score on the validation set, and then evaluate performance on the test set, as reported in Table 2.4. Logistic regression with doc2vec works well for blinded assessment of outcomes and conflict of interests, with F1 around 60% and 68% respectively. Support vector machine with word2vec shows strong performance for animal welfare regulations, with F1 of 90%. Random forest with bag-of-words achieves the best performance on the validation set for random allocation and animal exclusions, but proves to be over-fitting as the large difference of results between validation and test set shows. The optimal setting for each risk of bias items are: {maximum vocabulary size 2000, minimum document frequency 10, 1-gram, applying TF-IDF weighting; 500 trees in the forest, 2000 features considered for splitting trees} for random allocation, {maximum vocabulary size 10000, minimum document frequency 10, using DBOW, document vector size 300; inverse regularization constant 1} for blinded assessment of outcome, {maximum vocabulary size 10000, minimum document frequency 10, using DBOW, document vector size 400, inverse regularization constant 1} for conflict of interests, {α of 5e-10, maximum vocabulary size 10000, minimum document frequency 10} for animal welfare regulations, {maximum vocabulary size 5000, minimum document frequency 20, 2-grams and 3-grams, not using TF-IDF; 100 trees in the forest, 5000 features considered for splitting trees} for animal exclusions.

RoB item	**Model**	**Valid**				**Test**			
		F1	**R**	**P**	**S**	**F1**	**R**	**P**	**S**
RA	RF + bow	67.2	79.9	58.1	81.0	10.1	6.3	26.3	92.3
BAO	LogReg + d2v	60.0	69.1	53.0	74.5	58.4	62.3	54.9	74.6
CI	LogReg + d2v	68.8	76.1	62.8	82.6	67.9	74.9	62.1	79.6
CAWR	SVM + w2v	90.1	96.3	84.6	42.8	89.6	94.0	85.6	51.1
AE	RF + bow	48.8	44.6	53.8	93.6	15.9	22.6	12.3	80.6

Table 2.4: Performance of the best model for each risk of bias item on the validation set and final performance on the test set. 'R', 'P' and 'S' refer to recall, precision and specificity respectively.

Within each baseline model, I compare the performance of three text representation methods on the validation set, as shown in Table 2.5, Table 2.6 and Table 2.7. For support vector machine and logistic regression, doc2vec and word2vec outperform bag-of-words in general, and doc2vec performs slightly better than word2vec, with F1 increased by from 1% to 5%. For random forest, doc2vec and bag-of-words work better than word2vec, with the improvement of F1 ranging from 1% to 24%, but bag-of-words tends to be over-fitting easily. Doc2vec yields the best result in almost all baseline models, which is reasonable because doc2vec is generated and updated from the training sample itself, which is closer to the preclinical domain, while the pre-trained word vectors are induced from the more general biomedical corpus.

RoB item	**Feature**	**F1**	**Recall**	**Precision**	**Specificity**
Random allocation	bow	44.1	72.7	31.6	48.3
	w2v	51.0	58.2	45.4	76.9
	d2v	**51.9**	72.2	40.5	65.1
Blinded assessment of outcome	bow	47.4	94.3	31.6	15.3
	w2v	53.4	74.8	41.5	56.3
	d2v	**59.3**	67.8	52.7	74.7
Conflict of interests	bow	52.4	83.2	38.2	48.1
	w2v	65.7	70.6	61.5	83.0
	d2v	**67.1**	79.7	57.9	77.7
Compliance of animal welfare regulations	bow	87.3	97.4	79.0	15.7
	w2v	**90.1**	96.3	84.6	42.8
	d2v	86.7	82.7	91.2	74.1
Animal exclusions	bow	29.4	81.3	17.9	38.1
	w2v	38.7	49.1	32.0	82.6
	d2v	**39.0**	64.3	28.0	72.5

Table 2.5: Performance of support vector machine with three text representation methods for five risk of bias items on the validation set.

RoB item	Feature	F1	Recall	Precision	Specificity
Random allocation	bow	42.1	59.3	32.7	59.8
	w2v	**56.3**	75.3	44.9	69.7
	d2v	51.0	65.5	41.8	70.0
Blinded assessment of outcome	bow	46.6	78.3	33.1	34.5
	w2v	56.3	68.3	47.9	69.1
	d2v	**60.0**	69.1	53.0	74.5
Conflict of interests	bow	50.7	83.2	36.4	44.0
	w2v	66.1	74.6	59.3	80.2
	d2v	**68.8**	76.1	62.8	82.6
Compliance of animal welfare regulations	bow	86.9	100.0	76.9	1.8
	w2v	84.1	78.4	90.6	73.5
	d2v	**87.6**	85.4	89.9	68.7
Animal exclusions	bow	29.5	62.5	19.3	56.5
	w2v	36.8	65.2	25.6	68.5
	d2v	**41.4**	62.5	31.0	76.8

Table 2.6: Performance of logistic regression with three text representation methods for five risk of bias items on the validation set.

RoB item	Feature	F1	Recall	Precision	Specificity
Random allocation	bow	**67.2**	79.9	58.1	81.0
	w2v	43.4	60.3	33.9	61.4
	d2v	50.4	59.8	43.6	74.6
Blinded assessment of outcome	bow	50.7	57.8	45.1	70.8
	w2v	49.3	73.0	37.3	48.9
	d2v	**57.8**	68.3	50.2	71.8
Conflict of interests	bow	52.8	72.6	41.4	60.5
	w2v	55.1	60.4	50.6	77.3
	d2v	**65.1**	68.5	61.9	83.8
Compliance of animal welfare regulations	bow	87.7	92.1	83.7	41.6
	w2v	85.3	86.5	84.1	46.4
	d2v	**88.8**	89.7	88.0	60.2
Animal exclusions	bow	**48.8**	44.6	53.8	93.6
	w2v	35.0	64.3	24.1	66.2
	d2v	40.9	61.6	30.7	76.8

Table 2.7: Performance of random forest with three text representation methods for five risk of bias items on the validation set.

Keeping other parameters constant, I compare methods used for generating document vectors in doc2vec, which includes DM, DBOW, and concatenation of DM and DBOW. The original study suggests using DM or concatenation of DM and DBOW (Le and Mikolov, 2014), but my experiments demonstrate that DBOW alone achieves the best performance on almost all cases of three baseline models for each risk of bias item, with exception of random forest for random allocation and animal exclusions, where the concatenation of DM and DBOW works better, as shown in Figure 2.16 and Appendix Table 1. The optimal dimension of document vectors generated from doc2vec depends on the specific risk of bias item and classification model, as shown in Figure 2.17 and Appendix Table 2. Different dimensions of document vectors can induce 10% changes to F1 for animal exclusions. For the other four items, logistic regression is less sensitive to the dimension of document vectors, with changes of F1 less than 2%, while support vector machine and random forests can change F1 by 6%.

In practice, among baseline models, I recommend support vector machine or logistic regression with doc2vec which gives general good performance and is less prone to be over-fitting, compared to random forest or other combinations.

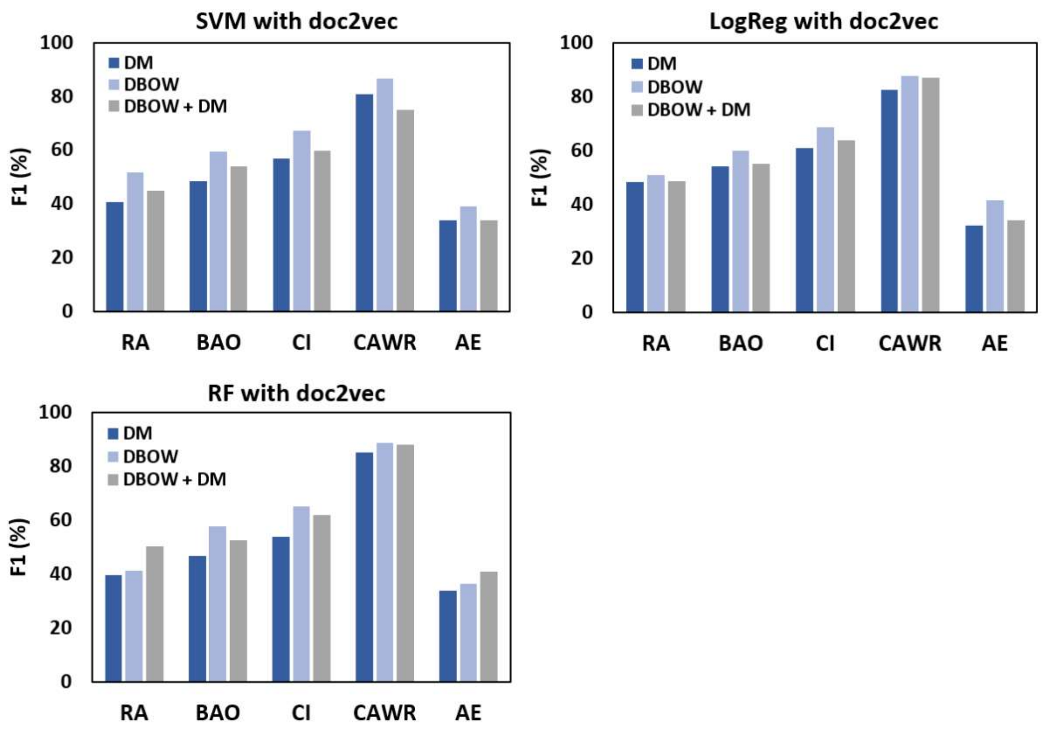


Figure 2.16: Effect of the method for generating document vectors in doc2vec.

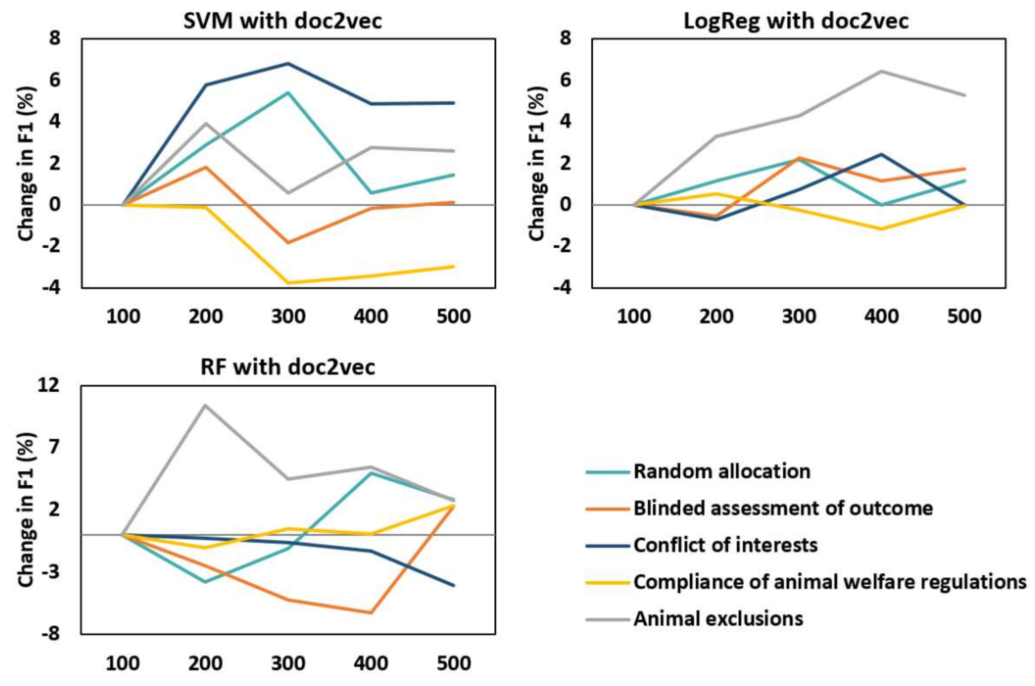


Figure 2.17: Effect of the dimension of document vectors generated from doc2vec.

2.4.2 Experiments of Neural Networks

This section describes experiments and results of three neural network models including convolutional neural network, recurrent neural network and hierarchical attention network. For all three models, the pre-trained word vectors used in the embedding layer are induced on a combination of PubMed and PMC texts with texts extracted from a recent English Wikipedia dump (5.5 billion words), using the skip-gram model with a window size of 5 (Pyysalo et al., 2013). Tokens with document frequency less than 10 are excluded when building the vocabulary. I set the number of samples included in every mini-batch (batch size) to 32 according to the suggestion from previous studies (Masters and Luschi, 2018). In the Adam algorithm, I set the learning rate to 1e-4, and coefficients for the mean and variance of gradients (β_1, β_2) to 0.9 and 0.999. I set dropout rate to 0.5. In default, I assign class weight [1/number of reported training samples, 1/number of unreported training samples] to records belonging to positive and negative class respectively and apply batch normalisation before the final softmax function. I use early stopping to terminate training and the stopping criteria are set based on the validation loss and F1 score. For one epoch, if (current validation loss - minimum validation loss of previous epochs) < $c1$ and (maximum validation F1 of previous epochs - current validation F1) > $c2$, this epoch is marked as 'stopping'. When the number of 'stopping' epochs reaches a value '$patience$', training stops. The values of $c1$, $c2$ and $patience$ are determined for different models and risk of bias items separately, based on the learning curves. Other configurations are described separately for each model in the following sections. Hyperparameters of neural networks are tuned sequentially, i.e. tuning one hyperparameter each time and selecting the optimal value, and then tuning the next hyperparameter.

Neural network models are implemented using PyTorch (Paszke *et al.*, 2019) and all the experiments are conducted on a GTX 1080 GPU with 12GB memory. Codes are available at github.com/qianyingw/pre-rob.

2.4.2.1 Experiments of Convolutional Neural Networks

As a starting configuration, I use three filter sizes [3,4,5] and apply 100 filters for each of the filter sizes. Documents are padded or cut when the number of words is less or more than 5000 respectively, and I consider 5000 most frequent words as features from the training corpus to build the vocabulary.

With this setting unchanged, I tune maximum document length among {5000, 10000, 15000}, number of filters for each filter size from 20 to 300, filter sizes among {[3,4,5], [4,5,6], …, [11,12,13]}, maximum number of features among {5000, 7000, 9000}, assigning class weight or not, applying batch normalisation or not, freeze embedding or not. I explore the undersampling strategy (Fernández *et al.*, 2018) for animal exclusions particularly because it has the most imbalanced classes (reported/unreported=12%) compared to the other four items in the dataset. More specifically, I randomly remove some unreported records from the training set, to increase the ratio of reported to unreported records to 100%, 50% or 25%. Based on the results and learning curves from some initial experiments, the number of training epochs and early stopping criteria are set to {$c1$=5e-3, $c2$=0.01, $patience$=9, 20 epochs} for random allocation, {$c1$=6e-3, $c2$=0.01, $patience$=1, 20 epochs} for blinded assessment of outcome, {$c1$=0.01, $c2$=0.01, $patience$=2, 20 epochs} for conflict of interests, and {$c1$=0.01, $c2$=0.01, $patience$=3, 40 epochs} for animal welfare regulations and animal exclusions.

Results

After parameters tuning, I select configurations which achieve the highest F1 score on the validation set, and evaluate performance on the final test set, as reported in Table 2.8. Compared to baseline models (Table 2.4), CNN models improve the F1 score by 10% to 20% for four items, with exception of compliance of animal welfare regulations, where the validation F1 decreases by 3%. CNN models are also more robust than baseline models in general, as the difference of F1 between the validation set and test set is smaller, except for animal welfare regulations, where the difference of F1 is 13% higher than that of baseline models. The optimal setting for each risk of bias items are: {maximum 10000 tokens in each document, 120 filters for each filter size of [10,11,12], applying class weight balancing, no batch normalization, 7000 most frequent tokens considered for building vocabulary} for random allocation, {maximum 14000 tokens in each document, 140 filters for each filter size of [9,10,11], applying class weight balancing, no batch normalization, 5000 most frequent tokens considered for building vocabulary} for blinded assessment of outcomes, {maximum 14000 tokens in each document, 100 filters for each filter size of [3,4,5], applying class weight balancing, no batch normalisation, 5000 most frequent tokens considered for building vocabulary} for conflict of interests, {maximum 10000 tokens in each document, 280 filters for each filter size of [3,4,5], no class weight balancing, applying batch normalization, 5000 most frequent tokens considered for building vocabulary} for compliance of animal welfare regulations, and {no limitation for maximum document length, 50 filters for each filter size of [3,4,5], applying class weight balancing, no batch normalisation, 9000 features considered for building vocabulary} for animal exclusions.

RoB item	**Valid**				**Test**			
	F1	**R**	**P**	**S**	**F1**	**R**	**P**	**S**
RA	86.4	93.2	81.8	92.8	84.4	88.1	82.0	91.5
BAO	82.4	88.5	77.8	89.4	81.6	85.9	79.7	89.0
CI	84.5	86.8	84.1	93.8	82.7	80.6	86.2	93.9
CAWR	86.9	83.3	92.4	97.4	72.8	71.3	76.9	93.8
AE	60.2	73.6	54.2	89.7	46.6	56.5	45.0	91.1

Table 2.8: Performance of the convolutional neural network model with the optimal configuration for each risk of bias item on the validation set, and final performance on the test set. ‘R’, ‘P’ and ‘S’ refer to recall, precision and specificity respectively.

With other configurations constant, I explore the effect of the number of filters for each filter size, and the changes of F1 score on the validation set are displayed in Figure 2.18 and Appendix Table 3. For random allocation, blinded assessment of outcome and conflict of interests, the performance does not vary much with the different number of filters (fluctuation of F1 is less than 2%), but the optimal number is between 100 to 140. For animal welfare regulations, there is a clearly ascending trend in F1, with 10% improvement when the number of filters rises from 40 to 300. For animal exclusions, the F1 score first rises then becomes relatively stable when the number of filters is larger than 60, with fluctuation less than 5%. Consider the performance improvement and computation cost, it is unlikely worth trying a larger number of filters. For the effect of the filter size, [3,4,5] is a good choice for all risk of bias items, as shown in Figure 2.19 and Appendix Table 4. For conflict of interests, animal welfare regulations and animal exclusions, increasing filter size harms the performance, with a reduction of F1 by 2%, 4% and 20% respectively, while larger filter size like [9,10,11] or [10,11,12] provides the possibility of better performance for random allocation and blinded assessment of outcomes. Table 2.9 shows that freezing embeddings harms the performance by 0.2% to 15%, which indicates the embeddings should be jointly trainable within the neural network in this task.

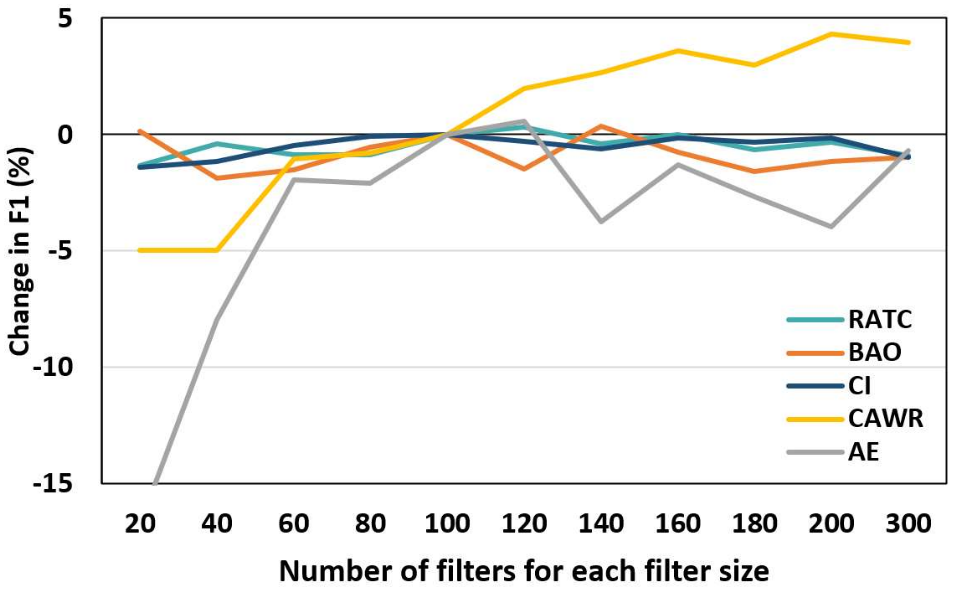


Figure 2.18: Effect of the number of filters for each filter size in the convolutional neural network for risk of bias classification.

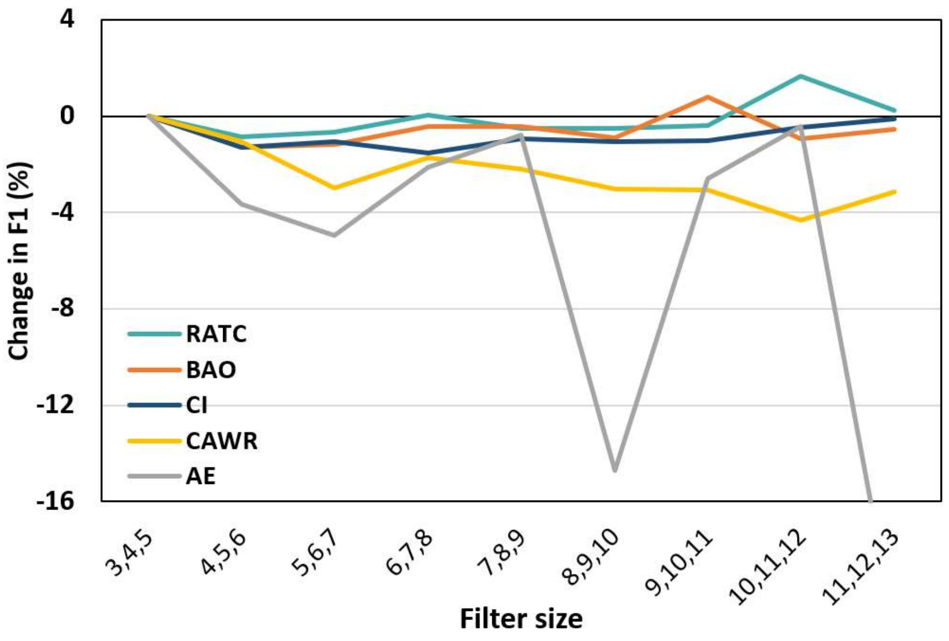


Figure 2.19: Effect of filter size in the convolutional neural network for risk of bias classification.

RoB item	Embedding	F1	Recall	Precision	Specificity
RA	freeze	86.2	91.2	82.7	92.6
	trainable	86.4	93.2	81.8	92.9
BAO	freeze	79.9	87.2	75.7	87.6
	trainable	82.4	88.5	77.8	89.1
CI	freeze	81.9	83.5	82.0	90.5
	trainable	84.5	86.8	84.1	91.7
CAWR	freeze	82.9	80.1	86.5	93.3
	trainable	86.9	83.3	92.4	93.3
AE	freeze	45.6	76.1	36.2	76.3
	trainable	60.2	73.6	54.2	86.8

Table 2.9: Effect of freezing embedding in the convolutional neural network for risk of bias classification.

2.4.2.2 Experiments of Recurrent Neural Network with Attention

As a starting configuration, I use LSTM as the RNN cell in the one-layer bidirectional structure and set the dimension of hidden states to 50. I set the maximum document length to 10000, and consider 5000 most frequent words as features from the training corpus to building the vocabulary. With this setting unchanged, I explore the effect of the different structure of RNN cell (LSTM or GRU). I tune the dimension of hidden states from 2 to 100, maximum document length from 5000 to 20000, the maximum number of features considered for building vocabulary from 3000 to 20000, using bidirectional structure or not, assigning class weight or not, applying batch normalisation or not, and freezing embedding or not. For the number of hidden layers, I only explore models with one or two layers as my initial experiments showed that models with deeper layers tended to be over-fitting. Similar to the CNN experiments, I also explore an undersampling strategy for animal exclusions. For the context vector in the attention module, I use the Kaiming initialisation method which helps to prevent the gradients from growing or shrinking (He *et al.*, 2015). It samples the values of weight W following a uniform distribution $W \sim U[-\sqrt{\frac{2}{h}}, \sqrt{\frac{2}{h}}]$, where h is the dimension of inputs to the hidden layer. Based on the results and learning curves from some initial experiments, the number of training epochs and early stopping criteria are set to {$c1$=0.01, $c2$=0.008, $patience$=2, 20 epochs} for random allocation and conflict of interests, {$c1$=0.02, $c2$=0.01, $patience$=6, 20 epochs} for blinded assessment of outcome, {$c1$=0.01, $c2$=0.01, $patience$=3, 40 epochs} for compliance of animal welfare regulations, and {$c1$=0.01, $c2$=0.01, $patience$=3, 20 epochs} for animal exclusions.

Results

After parameters tuning, I select configurations which achieve the highest F1 score on the validation set, and evaluate performance on the final test set, as reported in Table 2.10. RNN with attention also outperforms baseline models (Table 2.4) with an improvement of F1 by 10% to 24% for four items, while the validation F1 of animal welfare regulations decreases by 14%. Compared to CNN models (Table 2.8), RNN with attention does not show many advantages, as the validation F1 increases by less than 1% for random allocation and blinded assessment of outcome, but declines by 2% for conflict of interests and animal exclusions, and by 11% for animal welfare regulations. Similar to CNN models, the performance of RNN with attention is also less robust than that of the baseline model for animal welfare regulations, as the difference of F1 between the validation set and the test set is as high as 12%.

The optimal setting for each risk of bias items are: {one-layer bidirectional LSTM with hidden dimension 14, applying class weight balancing, no batch normalisation} for random allocation, {two-layers bidirectional GRU with hidden dimension 100, applying class weight balancing, no batch normalisation} for blinded assessment of outcomes, {two-layers bidirectional GRU with hidden dimension 20, applying class weight balancing, no batch normalisation} for conflict of interests, {one-layer bidirectional GRU with hidden dimension 4, no class weight balancing, applying batch normalisation} for compliance of animal welfare regulations, and {one-layer bidirectional LSTM with hidden dimension 6, applying class weight balancing and batch normalization, no under sampling} for animal exclusions. The optimal maximum document length is 10000 and feature numbers for building vocabulary is 5000 for all items.

RoB item	Valid				Test			
	F1	R	P	S	F1	R	P	S
RA	87.2	92.4	83.7	93.7	82.0	86.8	79.5	89.6
BAO	83.0	91.1	77.2	88.5	81.6	87.8	78.2	88.4
CI	82.9	85.4	82.0	92.9	81.5	81.6	82.6	92.2
CAWR	76.3	77.6	78.3	93.5	76.3	77.6	78.3	93.5
AE	58.0	68.3	54.3	90.0	42.3	50.6	38.7	90.9

Table 2.10: Performance of the recurrent neural network with attention using the optimal configuration for each risk of bias item on the validation set, and final performance on the test set. ‘R’, ‘P’ and ‘S’ refer to recall, precision and specificity respectively.

For each risk of bias item, I use its optimal RNN model to output the attention scores of tokens in every individual papers which reported the item, thus I can extract the most important words in the decision of risk of bias classification. The five most important words are {"randomly", "induced", "supported", "randomized", "increase"} for random allocation, {"blind", "by", "observer", "experimenter", "investigator"} for blinded assessment of outcome, {"interest", "of", "no", "authors", "statement"} for conflict of interests, {"animal", "care", "procedures", "figure", "committee"} for animal welfare regulations, and {"excluded", "were", "from", "included", "died"} for animal exclusions, as shown in Figure 2.20. This may inform future rule-based approaches development.

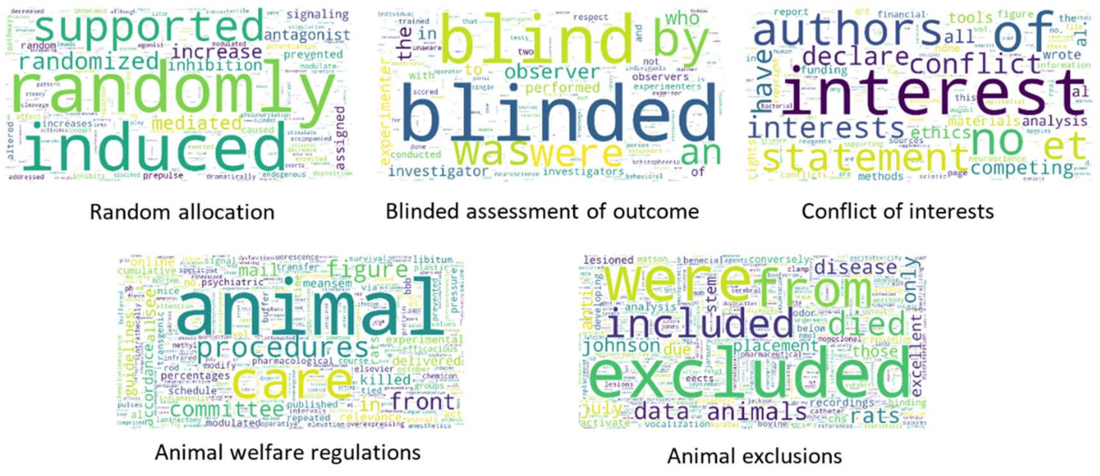


Figure 2.20: Most important words in the decision of classification for each risk of bias item, based on the average attention scores from RNN output across all positive samples.

The effect of the hidden dimension on validation F1 are displayed in Figure 2.21 and Appendix Table 6. For random allocation and blinded assessment of outcome, the performance is relatively stable and the optimal hidden dimension is around 20. For conflict of interests, a hidden dimension larger than 30 harms the performance by 20% and the model tends to be over-fitting. For animal welfare regulations, F1 reduces by 16% when the hidden dimension rises from 2 to 100. For animal exclusions, the changes of F1 fluctuate and the model with 6-dimension hidden states achieves the best performance. Considering the performance reduction, memory consumption and running cost, it is unnecessary to explore hidden dimensions larger than 20 in RNN models for risk of bias classification.

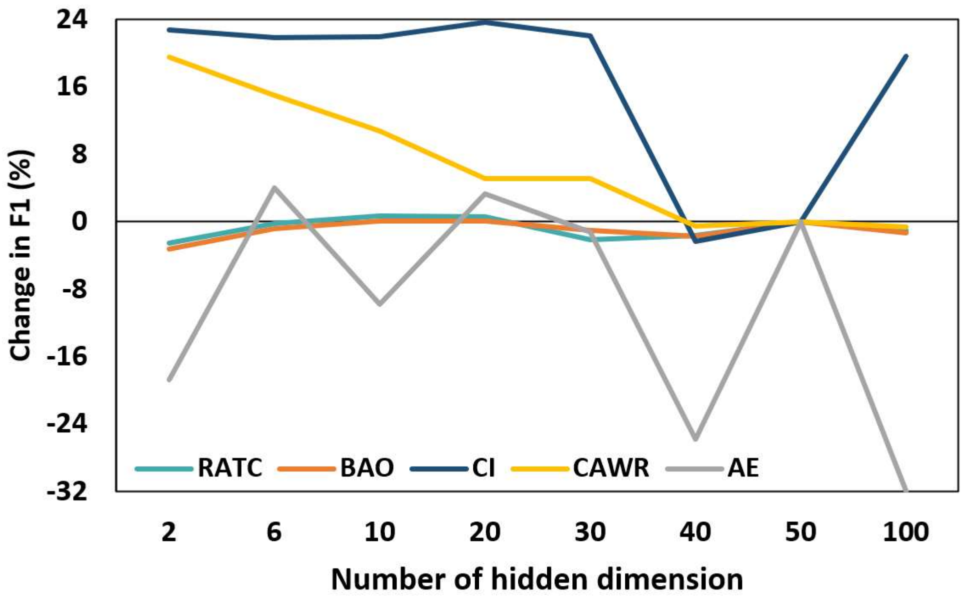


Figure 2.21: Effect of the number of hidden dimension in the recurrent neural network models for risk of bias classification.

The effect of cell structure in RNN models on the validation performance is reported in Table 2.11. Bidirectional structure performs better than unidirectional structure, with the improvement of F1 ranging from 0.5% to 20% for four items, except for conflict of interests, where the validation F1 of the bidirectional GRU is 7% lower than that of the unidirectional GRU. The selection of RNN cell structure between LSTM and GRU depends on the specific risk of bias item and other model configurations. GRU performs slightly better than LSTM for random allocation, while LSTM is optimal for blinded assessment of outcome, although the difference is almost negligible. The conclusions for the other three items are different when the RNN cell is applied in the unidirectional or bidirectional manner.

RoB item	RNN cell	Bidirectional				Unidirectional			
		F1	R	P	S	F1	R	P	S
RATC	LSTM	85.0	89.1	82.6	93.9	83.6	92.5	77.6	91.5
	GRU	**85.2**	91.6	80.7	92.9	84.9	89.7	81.7	93.6
BAO	LSTM	**81.6**	88.4	76.4	88.1	81.0	87.5	77.2	88.5
	GRU	81.2	88.4	76.1	88.4	79.6	87.4	74.7	87.7
CI	LSTM	73.4	72.3	76.2	91.6	53.5	50.0	59.1	87.4
	GRU	73.1	76.8	72.4	89.1	**80.1**	80.0	81.0	93.2
CAWR	LSTM	73.6	69.8	81.3	95.2	73.4	71.8	76.7	93.3
	GRU	**76.3**	77.6	78.3	93.5	71.4	75.9	72.1	90.9
AE	LSTM	**58.0**	68.3	54.3	90.0	46.4	73.1	36.4	79.1
	GRU	51.6	60.8	46.7	89.3	48.7	58.9	44.3	88.4

Table 2.11: Effect of different cell structure in the recurrent neural network models. 'R', 'P' and 'S' refer to recall, precision and specificity respectively.

2.4.2.3 Experiments of Hierarchical Attention Network

As a starting configuration, each document is padded or cut when it has less or more than 500 sentences, and each sentence is padded or cut when it has less or more than 100 words. I set the dimension of word hidden states and sentence hidden states to 50. I consider the 5000 most frequent words as features when building the vocabulary. With the setting unchanged and considering the memory limitation of our GPU, I tune the maximum number of sentences in each document from 200 to 1000, the maximum number of words in each sentence from 10 to 100, dimension of word hidden states and sentence hidden states from 5 to 80, the maximum number of features from 3000 to 9000, assigning class weight or not and applying batch normalisation or not. I also explore an undersampling strategy for animal exclusions like previous neural network experiments. I use the Kaiming initialisation method (He *et al.*, 2015) to initialise the local word context vector and the global sentence context vector. Based on the results and learning curves from some initial experiments, the number of training epochs and early stopping criteria is set to {*c1*=0.01, *c2*=0.008, *patience*=2, 20 epochs} for random allocation, blinded assessment of outcome and conflict of interests, and {*c1*=0.01, *c2*=0.01, *patience*=5, 30 epochs} for compliance of animal welfare regulations and animal exclusions.

Results

After parameters tuning, I select configurations which achieves the highest validation F1 score for each item and evaluate performance on the final test set, see Table 2.12. Similar to CNN and RNN with attention models, HAN outperforms baseline models (Table 2.4) with an improvement of F1 by 8% to 21% for four items, while the validation F1 of animal welfare regulations decreases by 11%. Compared to CNN models (Table 2.8), HAN models do not show any advantages as the validation F1 decreases for all five items by 1% to 8%. Compared to RNN models (Table 2.10), the validation F1 of HAN models declines by around 1% for random allocation, 2% for blinded assessment of outcome and 5% for animal exclusions, but increases by 0.3% and 3% for conflict of interests and animal welfare regulations respectively. The difference of HAN performance between the validation and test set is also less robust than that of baseline models for animal welfare regulations, as the difference of F1 increased by 13%.

RoB item	**Valid**				**Test**			
	F1	**R**	**P**	**S**	**F1**	**R**	**P**	**S**
RA	86.2	91.3	83.1	93.7	83.2	86.8	80.9	91.0
BAO	81.3	86.4	77.5	89.1	80.2	83.4	78.4	89.0
CI	83.2	84.7	82.8	93.2	81.0	79.7	84.1	92.9
CAWR	79.3	77.8	84.5	94.9	73.5	73.1	75.3	93.1
AE	53.4	58.4	54.0	88.9	42.4	53.3	38.2	88.9

Table 2.12: Performance of the hierarchical attention network with the optimal configuration for each risk of bias item on the validation set, and final performance on the test set. ‘R’, ‘P’ and ‘S’ refer to recall, precision and specificity respectively.

The optimal setting for each risk of bias items are: {maximum 500 sentences in each document, maximum 40 words in each sentence, word/sentence hidden states with dimension 50, applying class weight balancing, no batch normalisation} for random allocation and conflict of interests, {maximum 500 sentences in each document, maximum 50 words in each sentence, word/sentence hidden states with dimension 50, applying class weight balancing, no batch normalisation} for blinded assessment of outcomes, {maximum 600 sentences in each document, maximum 50 words in each sentence, word/sentence hidden states with dimension 50, not assigning class weight, applying batch normalisation} for compliance of animal welfare regulations, and {maximum 900 sentences in each document, maximum 100 words in each sentence, word/sentence hidden states with dimension 20, applying class weight balancing, no batch normalisation, no undersampling} for animal exclusions. The optimal feature number for building vocabulary is 5000 for all items.

Table 2.13 demonstrates the prediction probability and relevant sentences extracted from the optimal HAN model for each risk of bias item on an example full-text publication. Unlike the previous regular expression approaches which generate the yes/no label only, HAN can be used to extract the most relevant sentences from full text, which can enhance the judgment from the prediction probabilities, or provide signals whether users need to re-check the full texts. As the example shows, sentences extracted for blinded assessment of outcome and animal exclusions indicate the clear relation with the items and positive evidence for the prediction probabilities, while sentences extracted for conflict of interests do not show any relation with the items, which proves the prediction in a different direction. For the other two items, the conclusions from the prediction probability and the extracted sentences are opposite, where users can find enough positive evidence from the extracted sentences for random allocation to reject the prediction, but may need to re-check full text for animal welfare regulations.

RoB item	True	Predicted	High scored sentences
Random allocation	Yes	0.1712	*video records of randomly selected animals were recoded by an observer blind to the experimental conditions*
			in the stress groups animals were presented with ten sec db white noise tones that coterminated with a sec ma footshock measured according to the method of sananes and davis one tonefootshock pair was presented randomly within each of consecutive min intervals
Blinded assessment of outcome	Yes	1.0000	*video records of randomly selected animals were recoded by an observer blind to the experimental conditions*
			animals were coded initially by an observer not blind to the experimental conditions
Conflict of interests	No	2e-7	*schematic depictions of the regions dissected for neurochemical analysis are presented in figure tissue da and its metabolite dihydroxyphenylacetic acid dopac and ht and its metabolite hydroxy indoleacetic acid hiaa were isolated according to a modification of the procedure of reinhard et al*
			*** statistically significant difference versus the sham no stress group p statistically significant difference versus the sham no stress group p …*
Compliance of animal welfare regulations	No	0.9998	*the journal of neuroscience august role of the amygdala in the coordination of behavioral neuroendocrine and prefrontal cortical monoamine responses to psychological stress in the rat …*
			the present study used a conditioned stress model in which rats were trained to fear a substartle threshold tone paired previously with footshock and assessed for behavioral neuroendocrine and neurochemical stress responses
Animal exclusions	Yes	0.9989	*in of the original lesioned animals in the pretraining experiment the lesions were judged incomplete by the criteria above and were excluded from the data analyses*
			animals were subjected to bilateral sham sham or excitotoxic lesions of the amygdala amyg d before conditioning

Table 2.13: An example of prediction and relevant sentences extracted from the hierarchical attention network for risk of bias items on a full-text publication (PMCID: PMC6579011). Sentences are not exactly same as the original text because I remove digits and some punctuations in the text pre-processing step.

For the configurations related to padding and cutting, I explore the effect of the maximum number of sentences in each document (Figure 2.22 and Appendix

Table 7) and the effect of the maximum number of words in each sentence (Figure 2.23 and Appendix Table 8). Although the average number of sentences in a document is 180 (Table 2.2), the optimal threshold for sentence numbers is around 600 for animal welfare regulations and 500 for the other four items because of some long publications. Sentence counts larger than those values cause some reductions on performance. For the maximum word count in each sentence, the optimal threshold is around 40 for random allocation and conflict of interests, 50 for blinded assessment of outcomes, and 60 for animal exclusions. When the word count in each sentence is larger than those optimal values, the validation F1 first drops sharply and then goes up back to the peal value for four items, excluding random allocation where the performance remains stable after 60. There might be an improvement when the maximum word number in each sentence is larger than 100, but I am not able to continue the experiments with the limited GPU memory.

The effect of the dimension of word and sentence hidden states is displayed in Figure 2.24 and Appendix Table 9. The trend of the validation F1 for random allocation, blinded assessment of outcome and conflict of interests is relatively stable as the hidden dimension changes. For animal welfare regulations, the validation F1 declines by 22% when the word/sentence hidden dimension increases from 5 to 40. It rises up to a peak value when the dimension is 50 and then decreases again. For animal exclusions, the validation F1 fluctuates as the hidden dimension changes and there is no clear trend.

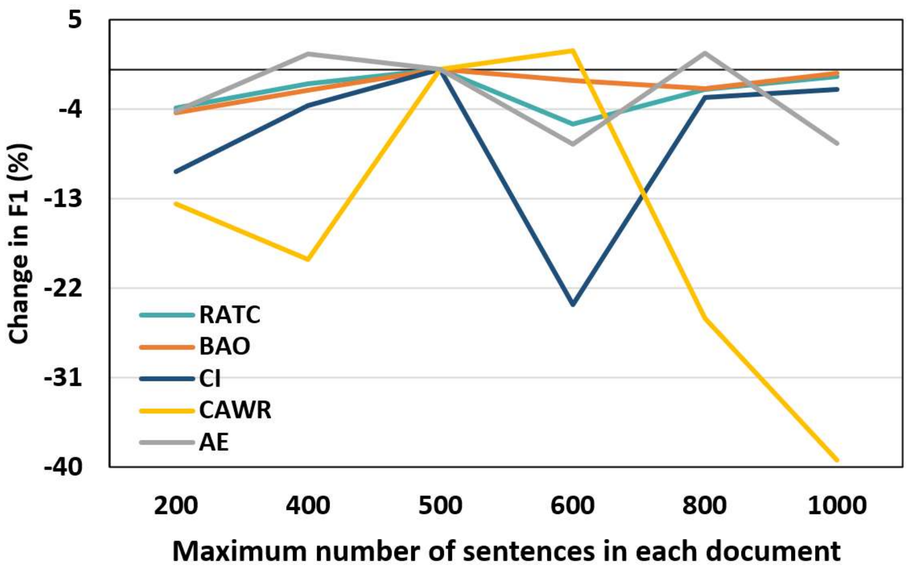


Figure 2.22: Effect of the maximum number of sentences in each document in the hierarchical neural network models for risk of bias classification.

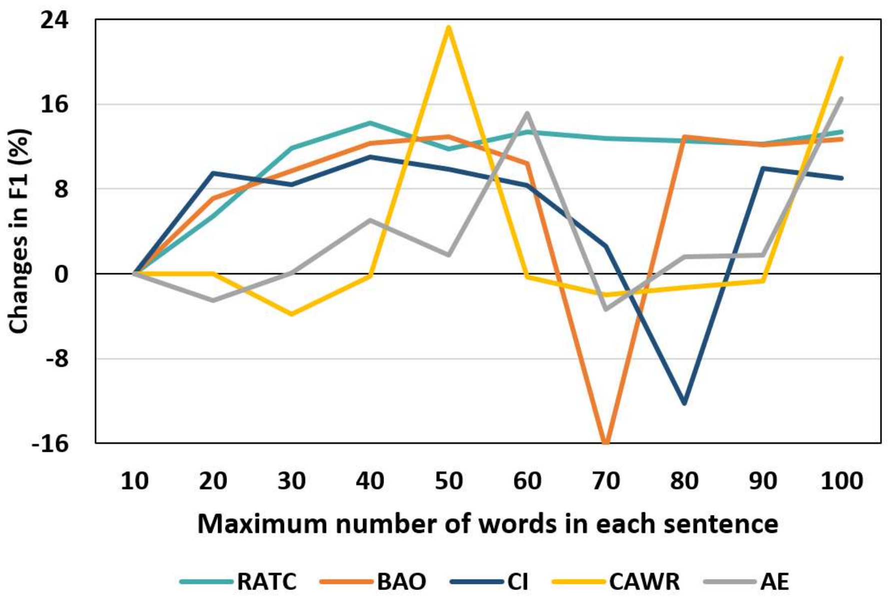


Figure 2.23: Effect of the maximum number of words in each sentence in the hierarchical neural network models for risk of bias classification.

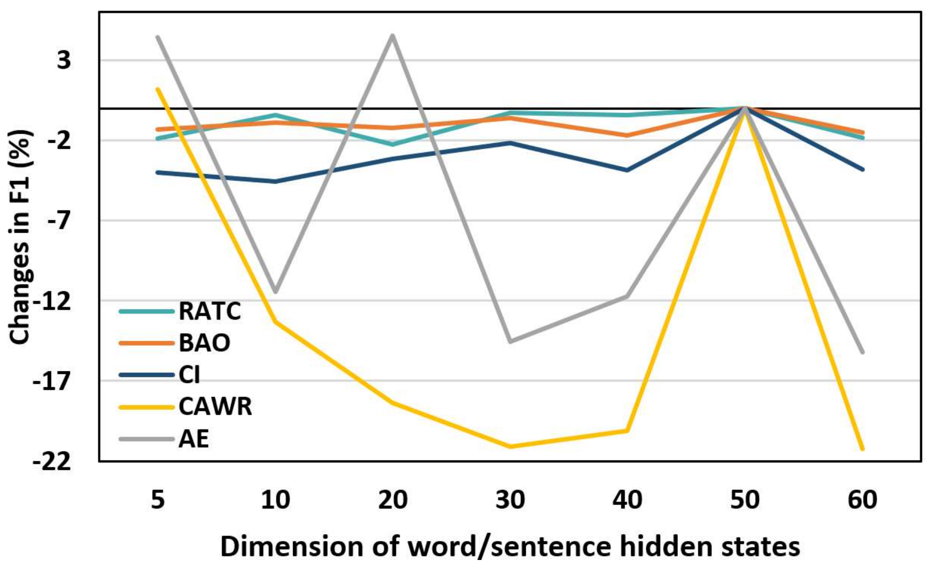


Figure 2.24: Effect of the dimension of word/sentence hidden states in the hierarchical neural network models for risk of bias classification.

2.4.3 Experiments of BERT with Two Strategies

BERT models are implemented using PyTorch (Paszke et al., 2019) and HuggingFace Transformers (Wolf *et al.*, 2019). All the experiments are conducted on a GTX 1080 GPU with 12GB memory. Codes are available at github.com/qianyingw/pre-rob.

2.4.3.1 Experiments of BERT-Document Chunk Pooling

After tokenisation, each document is split into chunks with 512 tokens (including the two special tokens '[CLS]' and '[SEP]') and the maximum number of chunks is set to 20 because of the memory limitation. Therefore, the model can process maximum 10,200 (510x20) tokens for each document, and documents with tokens more than 10,200 are cut. The last chunk with tokens less than 512 is padded when the total number of tokens is less than 10,200. I use the default configuration of the BERT-Base-Uncased module, i.e. 12 encoder layers, 768 units in hidden states, 12 attention heads (Devlin *et al.*, 2018). As a starting setting, I perform average pooling over the last 10 encoder layers for the first pooling layer; I concatenate hidden states of all the tokens within every chunk for the second pooling layer, i.e. no pooling. I set batch size to 32 for models with the linear/convolution head and 16 for models with the LSTM head, and assign class weights to solve the data imbalance issue. Considering the memory issue, I freeze all the BERT layers and only fine-tune the linear/LSTM/convolution head for the classification task. For BERT with linear head, I use the average pooling method for the third pooling layer. For BERT with convolution head, I set three filter sizes to [3,4,5] and apply 100 filters for each filter size. For BERT with LSTM head, I use one-layer unidirectional LSTM, and the hidden dimension is set to 768, which is same as the dimension in the BERT encoder layers. The threshold learning rate is 1e-3, the weight decay rate is 1e-2, and coefficients of controlling decay rate of gradients mean and variance are 0.9 and 0.999 respectively. With these settings, I test models for 30 epochs for random allocation, and the learning curves are shown in Figure 2.25. Models with the linear head or LSTM head are not trained properly and do not show promising performance, with the highest validation F1 around 50%. Considering the running cost and performance gain, I continue experiments of models with the convolution head only.

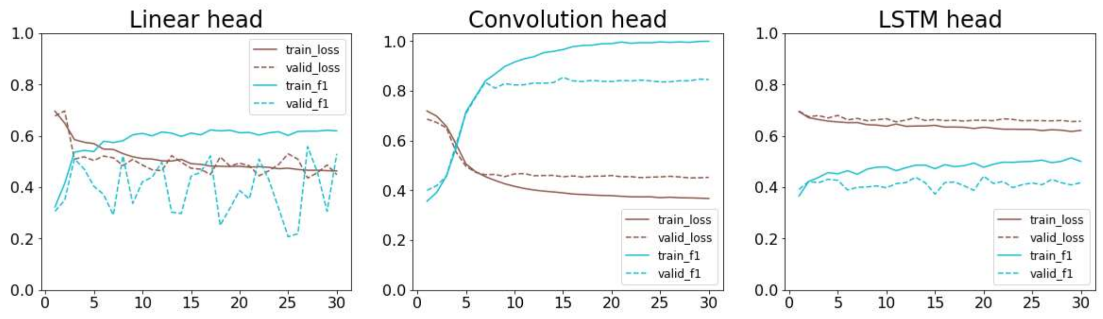


Figure 2.25: Learning curves of BERT-document chunk pooling models (with linear/convolution/LSTM head) for random allocation. Solid and dash lines refer to training and validation curves respectively; brown and cyan lines refer to loss and F1 respectively.

I tune three hyperparameters sequentially for BERT-DCP models with the convolution head. For the first pooling layer, I explore the different number of encoder layers being averaged from the last 10 to 1 encoder layers from

BERT. For the second pooling layer, I explore five pooling methods described in section 2.3.4.2. I freeze all BERT layers in default, which hinders the advantages of BERT architecture. With the available GPU, I can only fine-tune the last encoder layer but need to reduce batch size to 4, otherwise it causes the memory issue. Results reported in Table 2.14 indicate the number of encoder layers averaged at the first pooling layer does not have much effect on the performance. At the second pooling layer, pooling the output over tokens within each chunk with different methods harms the performance by 40%, compared with concatenating the hidden states of all the tokens within each chunk. Fine-tuning the last encoder layer from BERT harms the performance by 20%. Considering the running cost and reduction of performance, I use the optimal default settings for the other four risk of bias items and do not continue tuning parameters for them.

BERT-DCP	Settings	F1	R	P	S
Default	10 layers, no pooling, freeze all	**85.4**	**92.7**	80.1	92.2
1) Encoder layers averaged in the 1st pooling layer	Last 8 layers	84.5	92.0	79.2	91.6
	Last 6 layers	84.8	92.6	79.5	92.3
	Last 4 layers	85.0	91.4	**81.1**	**93.7**
	Last 2 layers	85.3	92.3	80.2	91.9
	Last 1 layer	84.7	92.1	79.4	92.3
2) Method in the 2nd pooling layer	Average-pooling	50.6	83.5	37.6	54.5
	Max-pooing	42.9	73.9	30.9	46.9
	Average- and max-pooling	48.1	82.5	35.3	52.2
	Hidden states from [CLS]	45.8	70.0	35.1	57.0
3) Unfreeze	Fine-tune the last encoder	62.0	65.8	61.1	91.0

Table 2.14: Effect of three hyperparameters on validation performance of BERT-document chunk pooling models with convolution head, for random allocation. The default settings are: 1) averaging output from the last 10 encoder layers at the first pooling layer; 2) concatenating hidden states of all tokens within each chunk at the second pooling layer; 3) freezing all BERT layers. ‘R’, ‘P’ and ‘S’ refer to recall, precision and specificity respectively.

2.4.3.2 Experiments of BERT-Sentence Extraction

As a default setting, the maximum number of sentences extracted from full-text publications is set to 30. The new passage concatenated from the most 30 relevant sentences with the number of tokens larger than 512 is truncated. I use the default configuration of the DistilBERT-Base-Uncased module: 6 encoder layers, 768 hidden units, 12 attention heads (Sanh *et al.*, 2019). I set batch size to 16, and assign class weights to solve the data imbalance issue. The threshold learning rate is 2e-5, the weight decay rate is 1e-2, and coefficients of controlling decay rate of gradients mean and variance are 0.9 and 0.999 respectively. As the new passages are much shorter compared to the full-text publications, I do not need to freeze any encoder layer like BERT-DCP models and can fine-tune the whole DistilBERT with the additional linear/convolution/LSTM head for the classification task. For models with the linear head, I use the hidden states of the '[CLS]' token from the last encoder layer as the output vectors. For models with the convolution head, I set three filter sizes to [3,4,5] and apply 100 filters for each filter size. For models with the LSTM head, I use a one-layer unidirectional LSTM and hidden dimension 768. With these settings, I run each model for 30 epochs with the different number of sentences extracted, see Table 2.15. The performance of models with different head or number of sentences extracted does not show much difference. Considering the running cost and experience from previous experiments, I use model with the convolution head with default setting as the final choice, and do not continue tuning parameters for the other four risk of bias items.

BERT-SE	No.sents	F1	Recall	Precision	Specificity
Linear head	10	78.1	79.7	81.0	92.6
	20	80.1	83.6	81.6	92.9
	30	80.5	82.2	83.3	94.1
	40	80.5	82.2	83.3	94.1
Convolution head	10	77.9	78.7	80.6	92.5
	20	79.6	84.8	79.5	90.8
	30	**80.6**	82.0	82.0	92.7
	40	78.8	81.7	79.7	92.7
LSTM head	10	78.0	79.6	82.4	93.5
	20	79.3	83.6	80.6	92.3
	30	77.9	84.1	77.8	91.1
	40	78.7	82.1	78.9	92.6

Table 2.15: Effect of the number of sentences extracted from full-text publications on validation performance of BERT-sentence extraction models with linear/convolution/LSTM head, for random allocation.

BERT models using two strategies do not outperform neural models, except for animal welfare regulations, where the validation F1 is increased by 3% to 4%, see the final performance in Table 2.16. This is reasonable because, in the document chunk pooling strategy, I do not take any advantage of BERT architecture by freezing all the encoder layers, and multiple pooling strategies help little to address this limitation; while in the sentence extraction strategy, although I can fine-tune all the encoder layers in DistilBERT, I still lose some information by using shorter texts extracted from full publications.

RoB item	BERT-DCP				BERT-SE			
	F1	R	P	S	F1	R	P	S
RA	85.4	92.7	80.1	92.2	80.6	82.0	82.0	92.7
BAO	83.1	91.8	77.0	87.7	79.9	84.7	79.8	89.9
CI	79.5	84.6	76.8	90.1	64.0	64.3	70.9	88.3
CAWR	93.8	92.1	95.8	87.7	94.0	94.6	93.8	75.1
AE	56.2	77.0	46.8	84.7	34.4	46.5	30.5	79.5

Table 2.16: Performance of the BERT-document chunk pooling and BERT-sentence extraction models using the optimal configuration for each risk of bias item on the validation set. ‘R’, ‘P’ and ‘S’ refer to recall, precision and specificity respectively.

2.4.3.3 BERT embeddings in baseline models

In the optimal BERT-DCP models with the convolution head where all encoder layers are untrainable, the essence of the architecture is analogous to the convolutional neural network, while the text representations are BERT embeddings instead of word2vec vectors. The inconspicuous difference of performance between BERT-DCP models with convolution head and CNN models indicate that BERT embeddings are not superior to word2vec vectors in neural network models for risk of bias classification, from another perspective. I then conduct experiments to test if BERT embeddings outperform word2vec embeddings in baseline models. For models using BERT embeddings, each document is split into sentences by scispaCy (Neumann *et al.*, 2019) and each sentence is then encoded into an embedding by SentenceTransformer (Reimers and Gurevych, 2019) with the pre-trained weights from BioBERT (Lee *et al.*, 2019). The text representation of the document is the averaged sentence embeddings. Other details of model implementation and parameters tuning are same as the models using word2vec which are described in Section 2.3.2. The validation results of three baseline models using averaged word embeddings from biomedical word2vec and averaged sentence embeddings from BioBERT are reported in Table 2.17. In support vector machines, F1 of BERT embeddings declines by 2% to 6% for four items; in logistic regression models, F1 reduces by 8% for random allocation, but increase by 2% for animal welfare regulations; in random forests, BERT embeddings improve F1 by 2% to 4% for blinded assessment of outcome and conflict of interests, but harm performance by 3% for animal exclusions, compared to word2vec embeddings. Other improvements or reductions are less than 1%. Although the selection of the embeddings in baseline models depends on the specific classifier and risk of bias item, none of the models using BERT embeddings outperform the best baseline model for each risk of bias item. Model architecture still plays a more important role than text representation methods on the risk of bias classification.

RoB item	Feature	SVM	LogReg	RF
RA	bert	44.7	47.9	43.9
	w2v	51.0	56.3	43.4
BAO	bert	54.0	56.0	53.2
	w2v	53.4	56.3	49.3
CI	bert	64.2	65.6	57.4
	w2v	65.7	66.1	55.1
CAWR	bert	86.6	86.4	85.3
	w2v	90.1	84.1	85.3
AE	bert	35.8	37.6	31.6
	w2v	38.7	36.8	35.0

Table 2.17: Validation F1 of three baseline models with averaged word embeddings from biomedical word2vec or averaged sentence embeddings from BioBERT for risk of bias classification.

2.4.4 Comparison to Regular Expression

With the best model and its optimal setting for each risk of bias item, I evaluate and compare the performance with the regular expression approach on the test set. Note that I select RNN with attention as the optimal model for blinded assessment of outcome rather than BERT with document chunk pooling strategy, considering the negligible improvement (0.1%) and complexity of pre-processing by the latter approach. From Table 2.18, NLP models improve performance by between 13% and 36% for four RoB items tested, and these improvements are significant with p < 0.05 according to McNemar's test (Raschka, 2018).

RoB item	Method	Test performance				McNemar table		
		F1	R	P	S		regex+	regex-
RA	RNN+Attn	82.0	86.8	79.5	89.7	nlp+	548	156
	Regex	68.8	96.4	53.6	62.7	nlp-	22	58
BAO	RNN+Attn	81.6	87.8	78.2	88.4	nlp+	601	87
	Regex	68.3	59.8	79.6	92.1	nlp-	28	68
CI	CNN	82.7	80.6	86.2	93.9	nlp+	533	103
	Regex	48.7	33.8	87.1	97.8	nlp-	21	54
CAWR	BERT-SE	91.5	91.4	92.0	70.9	nlp+	327	293
	Regex	55.2	40.9	85.2	78.2	nlp-	28	63
AE	CNN	60.2	73.6	54.2	89.7	nlp+	--	--
	Regex	--	--	--	--	nlp-	--	--

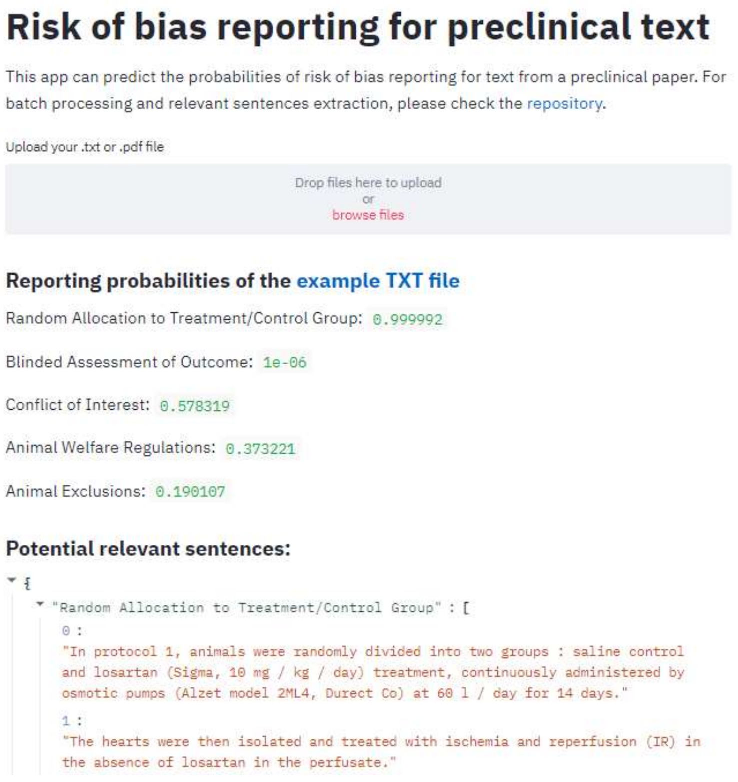
Table 2.18: Performance of the best NLP model and regular expression approach for each risk of bias item on the test set, and the corresponding table of McNemar test. '+' ('-') refers to records correctly (incorrectly) predicted by the approach. 'R', 'P' and 'S' means recall, precision and specificity respectively. A regular expression approach has not been developed for animal exclusions so the performance cannot be compared.

2.5 Discussion

I have shown that different models are optimal for the detection of reporting of different risks of bias. CNN is the best choice for conflict of interests and RNN with attention works well for random allocation and blinded assessment of outcome. For compliance with animal welfare regulations, models using BERT with sentence extraction strategy achieve the best performance. For animal exclusions, CNN achieves the best performance on the validation set, but no models provide reliable performance on the test set.

2.5.1 Tool and Streamlit interface

I make the codes and weights of optimal models for five risk of bias items available at github.com/qianyingw/pre-rob. A Streamlit web application is developed to demonstrate the function of the risk of bias tool, as shown in Figure 2.26. After a single TXT file is uploaded by the user, the app processes the text and runs the optimal model for each item in the background. It then generates probabilities and the most three relevant sentences which can assist judgement. However, it is often slow and memory-consuming to use the app for a single prediction because the models involve large pre-trained weights loading and heavy computations for neural networks and BERT models. For batch processing, I recommend to clone the source codes and keep pre-trained modules locally. More specifically, users need to have miniconda3 installed under the Linux environment. The tool receives the path of a CSV file as input, where the CSV contains the absolute paths of individual full-text TXT files. The plain texts of preclinical publications can be obtained by 1) converting from PDFs using tools like Xpdf, 2) parsing from PubMed NXML files using tools like PubMed Parser (Achakulvisut *et al.*, 2020), or 3) copy and paste. After activation of the virtual environment, installation of relevant python packages, downloading of pre-trained modules and configurations, users can obtain the prediction probabilities or relevant sentences (if the number of sentences is specified in the command) in an output CSV file by launching the python command, see Figure 2.27.



Risk of bias reporting for preclinical text

This app can predict the probabilities of risk of bias reporting for text from a preclinical paper. For batch processing and relevant sentences extraction, please check the repository.

Upload your .txt or .pdf file

Reporting probabilities of the example TXT file

Random Allocation to Treatment/Control Group: `0.999992`

Blinded Assessment of Outcome: `1e-06`

Conflict of Interest: `0.578319`

Animal Welfare Regulations: `0.373221`

Animal Exclusions: `0.190107`

Potential relevant sentences:

```
{
  "Random Allocation to Treatment/Control Group" : [
    0 :
    "In protocol 1, animals were randomly divided into two groups : saline control
    and losartan (Sigma, 10 mg / kg / day) treatment, continuously administered by
    osmotic pumps (Alzet model 2ML4, Durect Co) at 60 l / day for 14 days."
    1 :
    "The hearts were then isolated and treated with ischemia and reperfusion (IR) in
    the absence of losartan in the perfusate."
```

Figure 2.26: A demonstration Streamlit app for preclinical risk of bias assessment.

```
# =====================
# Clone source code
# =====================
git clone https://github.com/qianyingw/rob-pome.git

# =====================
# Settings
# =====================
# Install miniconda3
wget https://repo.continuum.io/miniconda/Miniconda3-latest-Linux-x86_64.sh
bash Miniconda3-latest-Linux-x86_64.sh

# Create and activate virtual environment
cd rob-pome/rob-app
virtualenv -p python3 rob
source rob/bin/activate

# Install packages
pip install -r requirements.txt

# Download module & pre-trained weights
sh setup.sh

# =====================
# Run
# =====================
# Obtain prediction probability of five items
python rob.py -p .../input.csv  # absolutae path of input.csv
# Extract two relevant sentences for each item
python rob.py -p .../input.csv -s 2
```

Figure 2.27: Usage commands of the preclinical risk of bias tool.

2.5.2 Error Analysis

Among the incorrect records, my models are more likely to conclude that publications report random allocation, blinded assessment of outcome and animal exclusions (false positive greater than false negative), and less likely to predict that publications report conflict of interests and animal welfare regulations (false negative greater than false positive), see Figure 2.28. To analyse sources of error I randomly select 10 incorrect records for each item from the test set. My models do not recognise phrases like ‘unaware’ for blinded assessment but consider that ‘animals are randomly selected for testing’ indicated random allocation to the experimental group. It may be that most records in the training set describe random allocation based on the presence of the word ‘random’ and blinded assessment based on the word ‘blind’, and that the training corpus does not have sufficient examples of alternative valid descriptions for these to be learned. I also find two records where a conflict of interest is given before the ‘Introduction’ section or after the ‘Reference’ section, where I have removed the relevant text in the text processing stage.

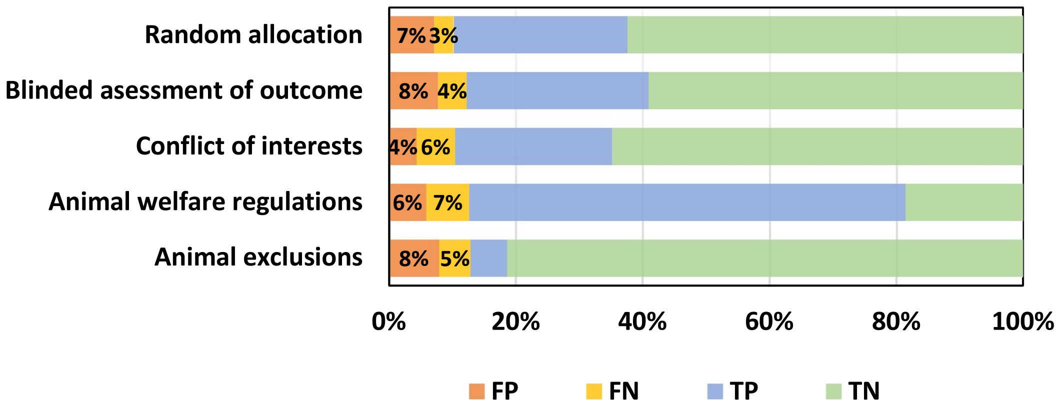


Figure 2.28: Percentages of the false positive, false negative, true positive and true negative of the optimal model for each risk of bias item on the test set.

2.5.3 Limitation and Future Work

This work has several limitations. First, the training dataset includes publications drawn from three datasets focusing on specific disease models (focal ischaemic stroke, chemotherapy-induced peripheral neuropathy, psychotic disorders), as well as two datasets from unselected preclinical studies published in PLOS One and Nature. This may influence the generalisability of the findings. Second, PDF to text conversion loses document structure and I cannot identify the main sections of publications. This introduces some noise (for instance text from figures and tables) to the training corpus. Tools like GROBID (github.com/kermitt2/grobid) can convert PDFs to structured XML but it highly depends on the quality of PDF, and in my experience it does not work well for some preclinical publications. However, enhanced approaches to PDF conversion, and increased availability of publications in XML format, means that this approach may become feasible in the future.

In future work I will seek to improve performance further, using datasets involving more journals and a wider range of preclinical experiments (both disease modelling and mechanistic studies), and will exploit diseases and texts from structured PubMed XMLs, which may yield better performance. I will continue improving the attribution of animal exclusions to achieve more reliable performance and I will develop approaches for the other risk of bias items including sample size calculation and allocation concealment. I will also develop a user-friendly function embedded in the preclinical systematic review facility SyRF (http://syrf.org.uk/) and a standalone API, enabling usage to others.

2.6 Summary

In this chapter, I present multiple text classification models for risk of bias assessment in preclinical publications, including baselines (support vector machine, logistic regression, random forest), neural networks (convolutional neural network, recurrent neural network with attention, hierarchical neural network) and BERT models using two strategies (document chunk pooling and sentence extraction); and text representation methods, including bag-of-words, word2vec, doc2vec, BERT embeddings.

Support vector machine and logistic regression are robust choices among baseline models, while neural models and BERT models have substantial improvements on performance compared to baselines, for four risk of bias items (random allocation, blinded assessment of outcome, conflict of interests and animal exclusions). For animal welfare regulations, baselines and BERT models have competitive performance and they both outperform neural models. Among three neural models, the difference on performance between CNN and RNN is not obvious, and the hierarchical structure in HAN does not show advantages. However, the attention mechanism can be used to develop the sentence extraction function to provide potentially relevant snippets as clues for users making judgements. Between the two BERT models, the document chunk pooling strategy works better than the sentence extraction strategy in general. Representations containing semantic information like word2vec or contextual embeddings yield better performance than the simple use of weighted frequency representations. If computational limitations require the implementation of a single tool, I recommend convolutional neural networks.

Compared to the previous regular expression approach, the performance of the best NLP models is significantly improved for four risk of bias items. I encourage the use of NLP techniques to assist the risk of bias assessment and reduce workflow for the preclinical systematic review. The performance of these tools is such that they could be deployed in automated approaches to monitor risks of bias reporting as part of institutional research improvement activities.

Chapter 3 Identifying Intervention and Method of Induction of Disease Model: Question Answering

Identifying information from publications can be cast as Question Answering (QA), which is defined as answering a question in natural language, by automatically copying or summarising a piece of text from available context or document(s). Answering biomedical questions varies from answering general-domain questions because of the different data sources, knowledge taxonomy and more complicated constraints for answer candidates (Niu *et al.*, 2003). Competitions such as the BioASQ challenge (Krallinger *et al.*, 2020) are held annually which aims to solve the information retrieval and question answering particularly for biomedical research, and several biomedical QA systems have been developed, including MedQA (Lee *et al.*, 2006), AskHERMES (Cao *et al.*, 2011), Olelo (Neves *et al.*, 2017), and SemBioNLQA (Sarrouti and Ouatik El Alaoui, 2020), etc. The architecture of these systems is end-to-end which contains four main stages: 1) analysing and parsing a question, 2) retrieving and ranking documents from large databases or corpora, 3) ranking and extracting passages/snippets from the top relevant documents, and 4) extracting answers by copying or summarising pieces of text from the top-ranking snippets (Prager, 2006). For PICO extraction in systematic reviews, the first two stages are not in my consideration because 1) the questions for PICO elements are generalised and fixed, such as 'What are the interventions?', and I cannot ask a more informative question for a specific disease or species without knowing other three PICO elements; 2) answers are extracted from each publication separately, where the publication is obtained by searching from biomedical database like PubMed or Embase via the MeSH (Medical Subject Headings) terms pre-defined by investigators. The publications are further checked by researchers or screened by some text mining algorithms to exclude irrelevant publications, so all the remaining documents are supposed to be relevant to the research topic of a systematic review. Therefore, I focus on retrieving PICO snippet from each document and extracting short precise answers from those snippets.

3.1 Related Work

For snippet retrieval, early works apply keywords or medical concepts matching by TF-IDF and BM25 (Robertson and Zaragoza, 2009). More complicated methods are also explored to retrieve and re-rank snippets, including training neural networks like LSTM (Wang and Nyberg, 2015) and CNN based models (Pappas *et al.*, 2020), and fine-tuning transformer-based model BERT (Nogueira and Cho, 2019). The advanced models demonstrate some improved performance but the improvement is limited, compared to the unsupervised baseline methods (Kazaryan *et al.*, 2020). In addition, those re-ranking models are supervised and require annotations of the relevance of snippets in the dataset, such as the Text REtrieval Conference (TREC) Data (Voorhees, 2001). For more precise answer extraction, similarly, hand-crafted matching patterns have been explored, while neural models like BiDAF (Seo *et al.*, 2016), QANet (Yu *et al.*, 2018) and more recent BERT models achieve promising performance. I describe these models in greater detail in the method section.

Among a few question answering approaches designed specifically for PICO extraction, Demner-Fushman et al use 275 clinical abstracts with PICO annotations to manually derive rule-based patterns for population, intervention, and comparator, assisted by the software MetaMap (Aronson and Lang, 2010) which can identify the Unified Medical Language System (UMLS) concepts (Bodenreider, 2004). While for outcomes, they do not develop a similar rule-based snippet-level extractor, but train a Naïve Bayes classifier to identify sentences containing outcome descriptions (Demner-Fushman and Lin, 2005). More recent work has applied the transformer-based models to extract PICO elements (Schmidt *et al.*, 2020), using a dataset converted from the 'EBM-NLP' corpus (Nye *et al.*, 2018) which contains 5000 clinical abstracts annotated with exact positions of PICO spans and is originally designed for

PICO extraction via named entity recognition models. To adapt the dataset to the question answering task, Schmidt et al split each abstract into sentences, convert the sentence-question-answer triplets into the SQuAD (Stanford Question Answering Dataset) format (Rajpurkar *et al.*, 2016). They then inject additional records from the SQuAD dataset and fine-tune the BertForQuestionAnswering module from the Hugging Face Transformers library (Wolf *et al.*, 2019), which achieves good performance (F1 over 75% for each PICO category) and outperforms the named entity recognition models on the EBM-NLP leaderboard (ebm-nlp.herokuapp.com/#Leaderboard). These works all focus on the clinical PICO extraction and no question answering approaches have been applied to the preclinical PICO extraction.

3.2 Dataset

The available datasets are from three systematic review projects: one focuses on clinical motor neuron disease, and the other two focus on preclinical psychotic disorders and chemotherapy-induced peripheral neuropathy. PICO annotations of each record contain only one answer phrase (factoid type) or a list of answer phrases (list type) and no position information is provided. The cleaning process and some summary statistics of the datasets are described below.

3.2.1 Clinical dataset: MND

MND dataset is obtained from the ReLiSyR-MND (Repurposing Living Systematic Review for Motor Neuron Disease) project, which aims to collect the clinical and preclinical evidence of motor neuron disease (MND) and other neurodegenerative diseases to inform the selection of drugs for repurposing in MND clinical trials (Macleod and Wong, 2019). 8,566 records with title and abstract from 3,122 papers annotated by multiple investigators were collected on 28th July 2020. I remove records without valid abstracts, where in most cases only titles can be found, and records without investigator information so I cannot trace back for data validation when necessary. The dataset includes two parent questions: ‘what is the intervention measure used?’ and ‘what is the disease investigated?'. As the whole dataset focuses on MND disease, the number of unique answer strings of the disease question is limited to 6 and I decide to focus on the intervention question. After removing duplicate records and records with inconsistent annotations from multiple investigators, I obtain 2,476 records with 1,812 unique answers in total. Among those, only 1,983 records have annotations which can be exactly matched in the title and abstract, as demonstrated in Figure 3.1. On average, there are 12 sentences and 289 tokens in an abstract, 25 tokens in a sentence and 2 tokens in an answer string (Table 3.1). The distributions of these characteristics are shown in Figure 3.2.

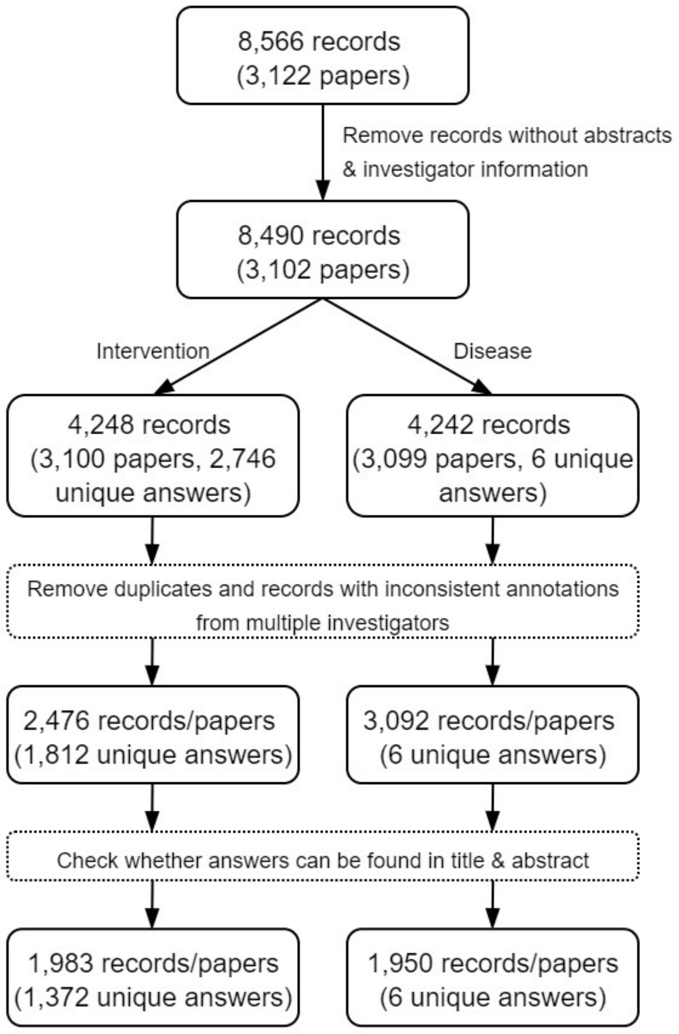

8,566 records (3,122 papers)
Remove records without abstracts & investigator information
8,490 records (3,102 papers)
Intervention
Disease
4,248 records (3,100 papers, 2,746 unique answers)
4,242 records (3,099 papers, 6 unique answers)
Remove duplicates and records with inconsistent annotations from multiple investigators
2,476 records/papers (1,812 unique answers)
3,092 records/papers (6 unique answers)
Check whether answers can be found in title & abstract
1,983 records/papers (1,372 unique answers)
1,950 records/papers (6 unique answers)

Figure 3.1: Preparation process of the clinical MND dataset.

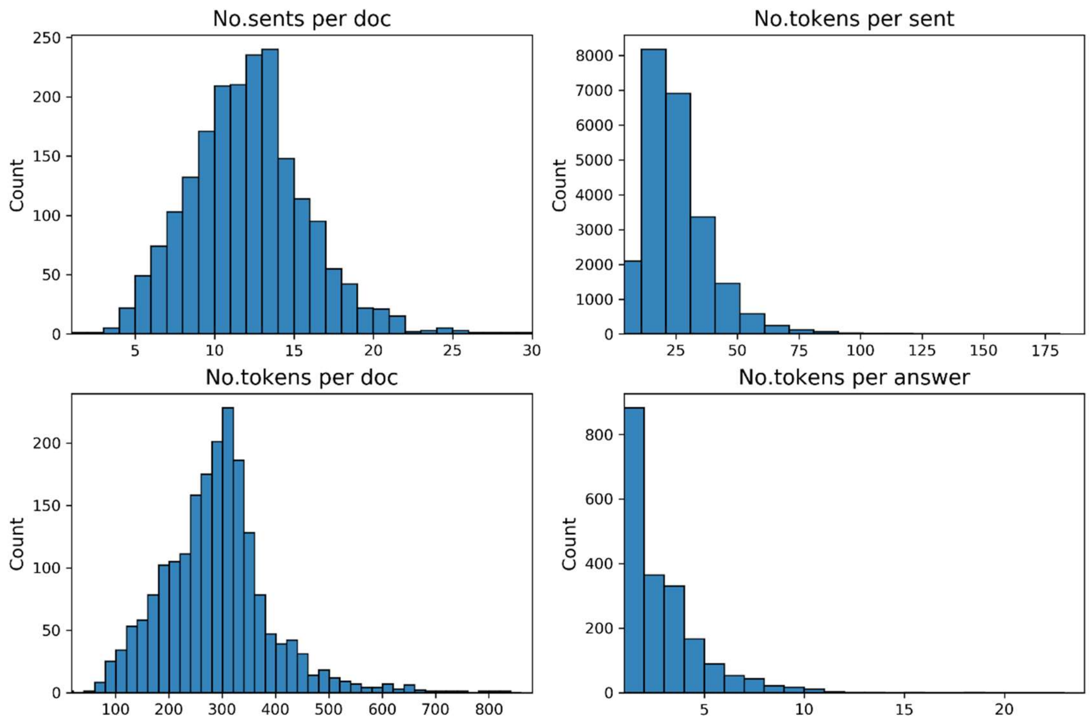


Figure 3.2: Distributions of text characteristics in the clinical MND dataset.

3.2.2 Preclinical Dataset: Psycho-CIPN

Psycho-CIPN dataset is concatenated from two systematic reviews which aim to collect the preclinical evidence of psychotic disorders (Psychosis) (Bahor *et al.*, 2016) and chemotherapy-induced peripheral neuropathy (CIPN) (Currie *et al.*, 2019) respectively. In the Psycho dataset, 12,440 papers with abstracts were obtained on 29th March 2018 and 3,024 records (two records for each of the 1,512 papers) with annotations of “treatments tested” (interventions) and “method of model induction” were filtered out. The annotations of each record consist of one or more than one answer candidates delimitated by a comma and whitespace. In the CIPN dataset, 1,397 records were obtained from 694 papers on 7th Sep 2020. Each record refers to a cohort in an individual study and some grouping process are required to generate the answer lists of intervention/model induction for each paper. 33% of records do not have answer matches (case insensitive) in titles and abstracts so I decided to work on full texts for preclinical intervention and induction identifications. I removed records without full texts and missing answer matches, and obtained 1,225 question-answer-context triplets from 906 papers, as demonstrated in Figure 3.3.

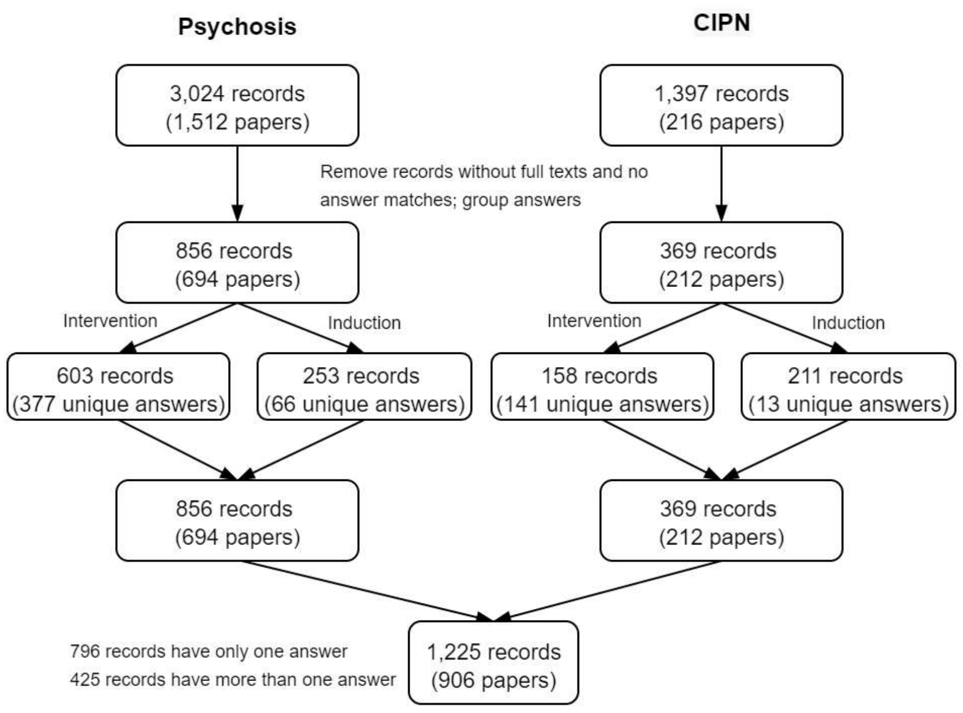

Psychosis
CIPN
3,024 records (1,512 papers)
1,397 records (216 papers)
Remove records without full texts and no answer matches; group answers
856 records (694 papers)
369 records (212 papers)
Intervention
Induction
Intervention
Induction
603 records (377 unique answers)
253 records (66 unique answers)
158 records (141 unique answers)
211 records (13 unique answers)
856 records (694 papers)
369 records (212 papers)
796 records have only one answer
425 records have more than one answer
1,225 records (906 papers)

Figure 3.3: The preparation process of the preclinical Psycho-CIPN dataset.

Psycho-CIPN-factoid & Psycho-CIPN-list

Among the final 1,225 records, 796 of them have only one answer and 429 of them have more than one answer candidates. On average, there are 203 sentences and 6,157 tokens in a full-text publication, 30 tokens in a sentence, and 1.1 tokens in an answer candidate (Table 3.1). The distributions of these characteristics are shown in Figure 3.4. In the actual experiments, I use the Psycho-CIPN data in two ways: 796 records with only one answer as Psycho-CIPN-factoid dataset, and records with one or more than one answers as Psycho-CIPN-list dataset. Additional processing is implemented for Psycho-CIPN-list records because it is not applicable to compare multiple positions of true answer candidates and predicted answer candidates, and compute loss during the training procedure. Therefore, a parent record with more than one answer candidates is split into separate sub-records for each answer candidate, which share the same id, context and question strings. This generates 1,984 sub-records as Psycho-CIPN-list dataset, which can be trained like the factoid-type question answering task. After training, the sub-records will be merged by ids for evaluation.

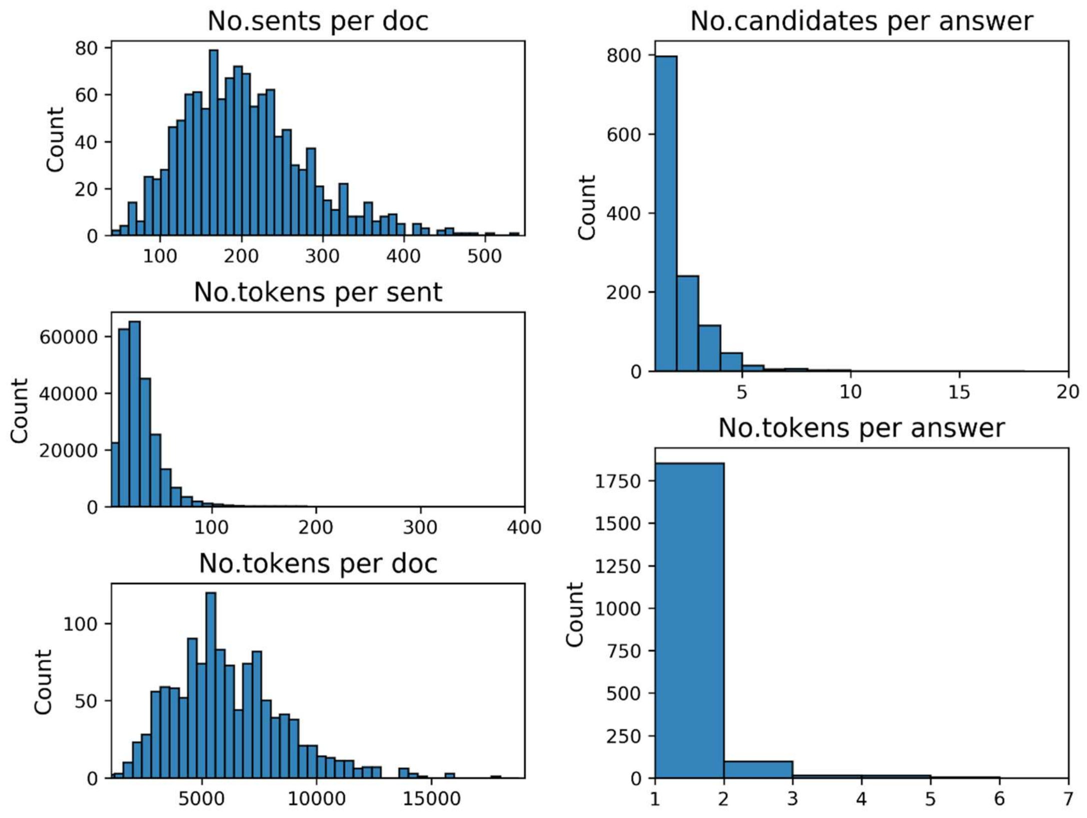


Figure 3.4: The distributions of text characteristics in the Psycho-CIPN dataset.

	MND	Psycho-CIPN
Avg no. sents per doc	12	203
Avg no. tokens per sent	25	30
Avg no. tokens per doc	289	6157
Avg no. tokens per answer	2.5	1.1
Avg no. answer candidates	1	1.6
Number of records	1983	1225 (796 factoid, 1984 list)

Table 3.1: Summary statistics of clinical MND dataset and preclinical Psycho-CIPN dataset. '1225' is the number of preclinical records before splitting them by number of answer candidates.

3.3 Methods

Identifying intervention/method of induction in biomedical literature can be formulated as a question answering (QA) task: given a question (“what is the intervention” or “what is the method of model induction of disease model”) and its corresponding context (an abstract or a full-text publication), I need to build a model to automatically extract answers (pieces of text describing interventions/induction) from the context, as demonstrated in Figure 3.5. I thus convert the datasets into the format in the standard reading comprehension challenge of Stanford Question Answering Dataset (SQuAD) (Rajpurkar *et al.*, 2016), as shown in Figure 3.6. The inputs of a standard QA model are the numeric representations of the tokenised question and context, and the true starting/ending positions (indices) of the answer strings, which are then compared to the predicted answer positions for training/evaluation, or converted back to answer tokens for prediction. However, the datasets do not contain any position information of the answer strings and I cannot compute the training loss for a QA model without the exact answer positions. As a compromise, I use the position of the first answer match in the context as the plausible position, inspired by a pre-processing strategy applied in a biomedical question answering challenge (Yoon *et al.*, 2019).

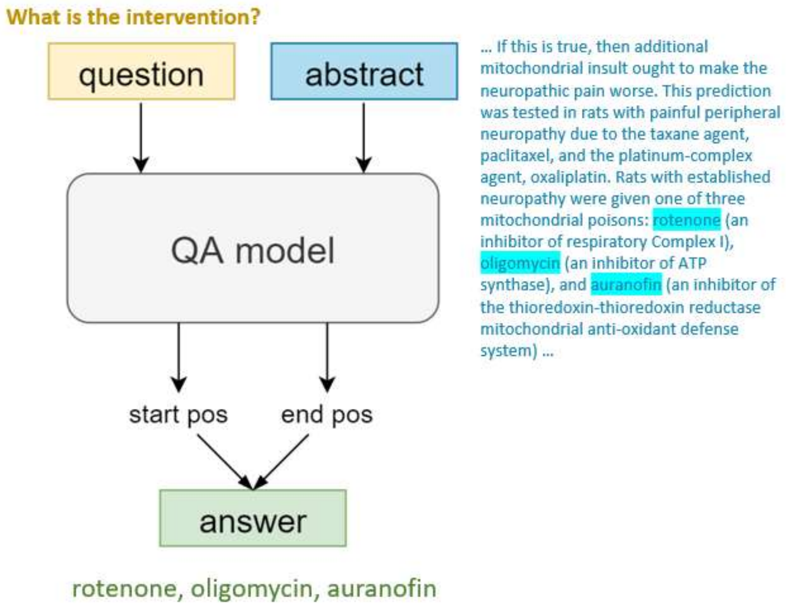
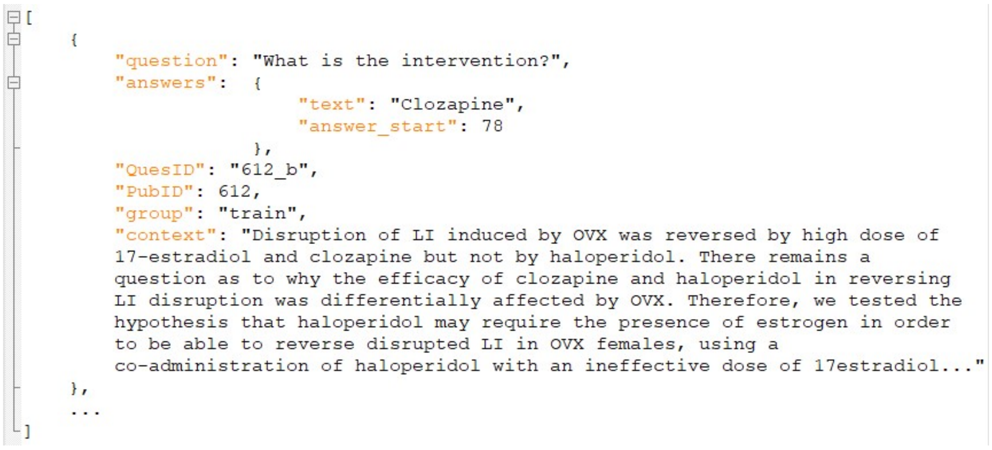

What is the intervention?
question
abstract
... If this is true, then additional mitochondrial insult ought to make the neuropathic pain worse. This prediction was tested in rats with painful peripheral neuropathy due to the taxane agent, paclitaxel, and the platinum-complex agent, oxaliplatin. Rats with established neuropathy were given one of three mitochondrial poisons: rotenone (an inhibitor of respiratory Complex I), oligomycin (an inhibitor of ATP synthase), and auranofin (an inhibitor of the thioredoxin-thioredoxin reductase mitochondrial anti-oxidant defense system) ...
QA model
start pos
end pos
answer
rotenone, oligomycin, auranofin

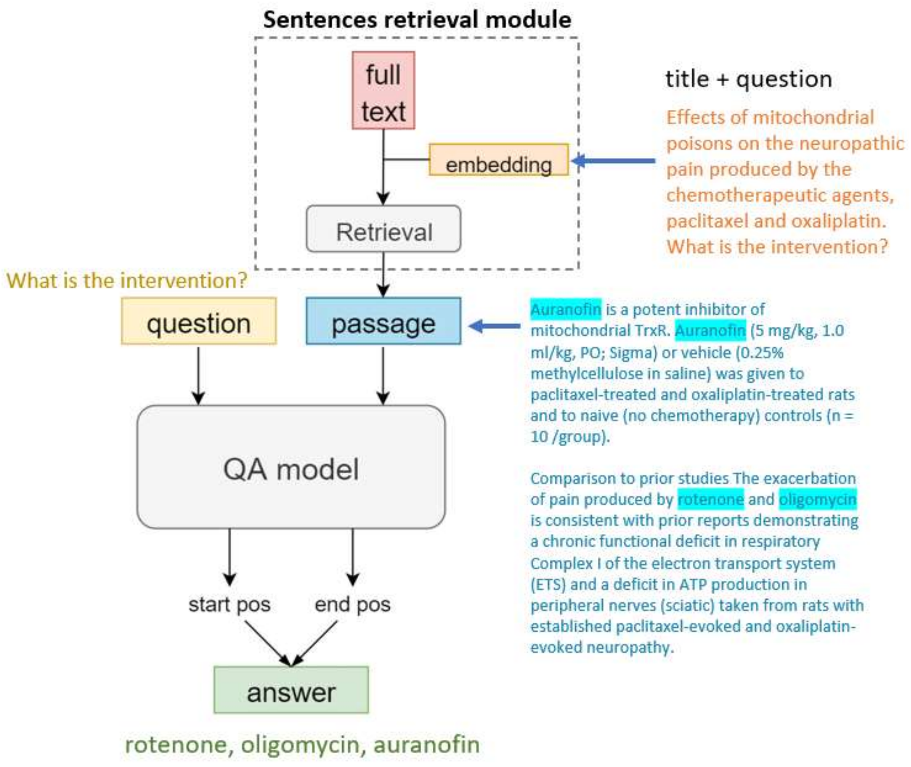
Figure 3.5: Task formulation for interventions identification of MND data.



```
[
    {
        "question": "What is the intervention?",
        "answers":  {
                        "text": "Clozapine",
                        "answer_start": 78
                    },
        "QuesID": "612_b",
        "PubID": 612,
        "group": "train",
        "context": "Disruption of LI induced by OVX was reversed by high dose of
        17-estradiol and clozapine but not by haloperidol. There remains a
        question as to why the efficacy of clozapine and haloperidol in reversing
        LI disruption was differentially affected by OVX. Therefore, we tested the
        hypothesis that haloperidol may require the presence of estrogen in order
        to be able to reverse disrupted LI in OVX females, using a
        co-administration of haloperidol with an ineffective dose of 17estradiol..."
    },
    ...
]
```

Figure 3.6: An example question answering record of intervention/induction identification in SQuAD format.

The average context length of the Psycho-CIPN dataset is much longer than that of the MND dataset, which includes 203 sentences and 12 sentences respectively. The ‘true’ index of answer strings is obtained from the first answer match in the context and a longer context can make the ‘true’ answer index more plausible. In a real case, the first answer match in the abstract could be the real correct answer, while the first answer match in the full-text paper is normally not. In a full-text publication, the answer is also mentioned in the introduction or background sections, which may not describe the actual intervention measures in the experiments. Context with full-text length also increases the training time and is likely to cause memory issues. To solve this, I add a sub-module for Psycho-CIPN datasets to retrieve relevant sentences which are then concatenated as new shorter passages, before training the question answering models, see Figure 3.7.



Sentences retrieval module
full text
embedding
Retrieval
title + question
Effects of mitochondrial poisons on the neuropathic pain produced by the chemotherapeutic agents, paclitaxel and oxaliplatin. What is the intervention?
What is the intervention?
question
passage
Auranofin is a potent inhibitor of mitochondrial TrxR. Auranofin (5 mg/kg, 1.0 ml/kg, PO; Sigma) or vehicle (0.25% methylcellulose in saline) was given to paclitaxel-treated and oxaliplatin-treated rats and to naive (no chemotherapy) controls (n = 10 /group).
Comparison to prior studies The exacerbation of pain produced by rotenone and oligomycin is consistent with prior reports demonstrating a chronic functional deficit in respiratory Complex I of the electron transport system (ETS) and a deficit in ATP production in peripheral nerves (sciatic) taken from rats with established paclitaxel-evoked and oxaliplatin-evoked neuropathy.
QA model
start pos
end pos
answer
rotenone, oligomycin, auranofin

Figure 3.7: Task formulation for interventions identification of Psycho-CIPN data.

The following sections describe the sentence retrieval module and classical QA models including BiDAF (Seo *et al.*, 2016), QANet (Yu *et al.*, 2018), fine-tuned DistilBERT (Sanh *et al.*, 2019) and BERT (Devlin *et al.*, 2018) with different pre-trained weights.

3.3.1 Sentence Retrieval

The sentence retrieval module is designed analogically to the document retriever (Chen *et al.*, 2017) for the open-domain question answering task. The open-domain question answering system takes a question as the only input and the answers are obtained through several processes including documents retrieval from the open-source knowledge base such as Wikipedia, passages/snippets retrieval, re-ranking from the top relevant documents, and answer span extraction from the top relevant snippets. Many retrieval modules use unsupervised approaches to extract the most relevant passages/snippets based on the ranked similarity scores between the question and each passage/snippet (Semnani and Pandey, 2020). Analogously, I can calculate the similarity scores between the question sentence and each sentence in the full-text publication to obtain the most relevant sentences. However, questions like ‘what is the intervention’ and ‘what is the method of model induction’ are not informative enough to retrieve sentences, so I concatenate the original question and the title of the publication to form a ‘new question’ for similarity calculation, as demonstrated in Figure 3.7.

To calculate similarities, sentences should be converted to numerical vectors and here I explore three representation methods: bag-of-words with TF-IDF (Goldberg, 2017), bag-of-words with BM25 (Robertson and Zaragoza, 2009), and Sentence-Transformers or Sentence-BERT (SBERT) (Reimers and Gurevych, 2019). Bag-of-words with TF-IDF method uses word frequency with normalisation of term and inverse document frequency to represent word importance, as described in Section 2.3.1. The difference is now the whole corpus is one full-text publication and the document is an individual sentence in the publication. The TF-IDF weight of the word w_i in the sentence s_j is:

$$\mathrm{tfidf}\left(w_i, s_j, S\right) = \frac{f\left(w_i, s_j\right)}{l\left(s_j\right)} \times \ln \frac{N(s)}{n(w_i)} \tag{3.1}$$

where $f(w_i, s_j)$ is the frequency of word w_i in the sentence s_j, $l(s_j)$ is the total number of words in the sentence s_j, $N(s)$ is the total number of sentences in the full-text publication (corpus) S, and $n(w_i)$ is the number of sentences containing the word w_i. Similar to the risk of bias project, words can be counted by unigram, bigram, or the combination of unigram and bigram.

BM25 is a modified version of TF-IDF weighting which takes the frequency saturation and average sentence length into consideration. The BM25 weight of the word w_i in the sentence s_j is calculated by:

$$\begin{aligned}&\mathrm{bm25}\left(w_i, s_j, S\right)\\ &= \frac{(k_1+1)f\left(w_i, s_j\right)}{f\left(w_i, s_j\right) + k_1\left(1 - b + b\frac{l\left(s_j\right)}{L}\right)} \times ln\left(\frac{N(s) - n(w_i) + 0.5}{n(w_i) + 0.5} + 1\right) \qquad (3.2)\end{aligned}$$

where constant $k_1 \in [1.2, 2]$ controls how much a single word can affect the weight of a sentence, constant $b \in [0.5, 0.8]$ controls the effect of relative sentence length compared to the average length (sentence length normalisation), and L is the average sentence length over corpus S (Robertson and Zaragoza, 2009).

Sentence-BERT (SBERT) takes advantage of the powerful pre-trained transformer-based model, BERT (Devlin *et al.*, 2018), to derive semantically meaningful sentence embeddings by averaging BERT output vectors of all tokens in the sentence (Reimers and Gurevych, 2019). I explore six pre-trained modules including 1) DistilBERT-NLI-STSb: DistilBERT (Sanh *et al.*, 2019) trained on the combination of the multi-genre Natural Language Inference (NLI) dataset (Bowman *et al.*, 2015), the Stanford Natural Language Inference (SNLI) dataset (Williams *et al.*, 2017) and Semantic Textual Similarity benchmark (STSb) dataset (Cer *et al.*, 2017); 2) DistilBERT-Marco: DistilBERT trained on a large scale MAchine Reading Comprehension (MS MARCO) dataset (Bajaj *et al.*, 2016); 3) BERT-Base: the original BERT trained on the combination of BookCorpus, and English Wikipedia; 4) BioBERT (Lee *et al.*, 2019): BERT trained on the combination of BookCorpus, English Wikipedia, PubMed abstracts and PubMed Central full-text articles (mixed-domain pretraining); 5) PubMedBERT-Abs: BERT trained on PubMed abstracts only, and 6) PubMedBERT-Abs-Full on a combination of PubMed abstracts and PubMed Central full-text articles (domain-specific pretraining) (Gu *et al.*, 2020).

With the numerical sentence vectors, the similarity score of the question vector q and the i-th sentence vector in the full-text publication is calculated by

$\text{similarity}(q, s_i) = \frac{q \cdot s_i}{\|q\| \times \|s_i\|}$. Then the top k sentences with the highest similarity scores are concatenated to form a new passage for each context in the training of QA models. Some open-domain QA systems also train a document or passage re-ranker (Nogueira and Cho, 2019) to obtain more precise passages. I do not apply this strategy here for two reasons. First, when the answer consists of several intervention measures, one sentence may not contain all the answer candidates. Some candidates are mentioned more frequently than other candidates throughout the full-text publication, and sentences containing the same most mentioned candidates may be ranked higher than sentences containing the rest but less mentioned candidates, which may leave some truly relevant sentences out when constructing the new passage. Second, to train a sentence re-ranker, I need to know the true relevance label of each sentence, which is not applicable in my datasets. The relevance label could be generated by judging if the sentence contains the answer strings but this strategy may introduce extra errors.

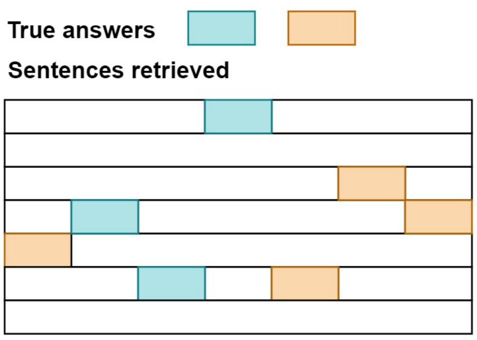
Metrics for Sentence Retrieval

A widely used metric of the general document retrieval tasks is the Mean Average Precision (MAP). For one question (information need), average precision (AP) is obtained from precisions of the top k relevant documents (k is the pre-defined number of documents required to be retrieved in total). Then MAP is obtained by taking the mean value of the average precisions of multiple questions (information needs) (Manning *et al.*, 2008). In the document retrieval task, the dataset used for training usually provides the relevance/irrelevance labels, which is different from my question answering datasets which do not have any relevance annotations at the sentence level.

Therefore, I introduce and illustrate two modified metric scores based on different definitions of 'relevance': sMAP (strict MAP) and rMAP (ratio MAP). I explain these scores with an example demonstrated in Figure 3.8, where seven sentences are retrieved for a question and the cyan/orange rectangle shapes represent two answer candidates. The precision of the i-th sentence is defined as

$$\text{Precision}_i = \frac{\text{accumulated number of relevant sentences}}{\text{accumulated number of retrieved sentences}} \tag{3.3}$$

In the strict mode, a sentence is considered relevant only when it contains all answer candidates (mark as 1). In the ratio mode, a sentence is considered relevant when it contains at least one answer candidate, but the accumulated relevance number is marked as the ratio of number of answer candidates included to the total number of answers in the sentence .



Acc sents retireved	Acc relevant (strict)	Acc relevant (ratio)
1	0	0.5
2	0	0.5
3	0	1
4	1	2
5	1	2.5
6	2	3.5
7	2	3.5

Figure 3.8: An illustrated example for metric (mean average precision) calculation in sentence retrieval. 'Acc sents retrieved' refers to the accumulated number of retrieved sentences; 'Acc relevant' refers to the accumulated number of relevant sentences.

Only precisions of relevant sentences are counted in the calculation of average precisions for one single record:

$$\text{AP} = \frac{\sum_{i=1}^{K} \text{Precision}_i \times \text{Relevance}_i}{\text{accumulated number of relevant sentences}} \tag{3.4}$$

$$\text{Relevance}_i = \begin{cases} 1, & \text{ith sentence is relevant} \\ 0, & \text{else} \end{cases} \tag{3.5}$$

Therefore in the example, the strict-mode AP is $\frac{\frac{0}{1}\times 0 + \frac{0}{2}\times 0 + \frac{0}{3}\times 0 + \frac{1}{4}\times 1 + \frac{1}{5}\times 0 + \frac{2}{6}\times 0 + \frac{2}{7}\times 0}{2} = 29.2\%$, and the ratio-mode AP is $\frac{\frac{0.5}{1}\times 1 + \frac{0.5}{2}\times 0 + \frac{1}{3}\times 1 + \frac{2}{4}\times 1 + \frac{2.5}{5}\times 1 + \frac{3.5}{6}\times 1 + \frac{3.5}{7}\times 0}{3.5} = 69.0\%$. Then sMAP and rMAP are calculated by averaging the strict and ratio APs across all records, respectively.

MAP scores measure the retrieval performance at the sentence level which is a little difficult to interpret for my task and datasets. Hence, I introduce another metric, Mean Match Ratio (MMR), to measure the retrieval performance on the entire passage (constructed from the relevant sentences). For one single record, in the strict-mode, the match ratio (MR) is set to 1 if the passage includes all the answer candidates, else zero; while in the ratio-mode, the match ratio is equal to the percentage between the number of answer candidates which can be found in the passage and the number of all true answer candidates. Then MMR is calculated by averaging match ratios across all records.

3.3.2 Question Answering Models

In this section, I present three question answering models: 1) Bidirectional Attention Flow (BiDAF) (Seo *et al.*, 2016), an influential milestone in the question answering field, 2) QANet (Yu *et al.*, 2018), a faster and the last superior model before BERT (Devlin *et al.*, 2018), and 3) fine-tuning BERT with different pre-trained modules for the downstream question answering task. As the input of QA models, questions and contexts are converted to numeric representations. For BiDAF and QANet, I use the biomedical word2vec (Pyysalo *et al.*, 2013) to generate text embeddings, as described in Chapter 2 (see Section 2.3.1), and WordPiece tokenisation (Wu *et al.*, 2016) for BERT models.

3.3.2.1 BiDAF

I implement BiDAF as the baseline QA model. The original BiDAF architecture includes three embedding layers for character embeddings, word embeddings and contextual embeddings, an attention flow layer, a modelling layer, and an output layer which are described in greater detail below. From the model ablation analysis in the original study, the semantics information is mainly represented by word-level embeddings and the absence of character-level embeddings harms the performance by about 2% (Seo *et al.*, 2016). Considering the little improvement and the training size of my dataset, I do not use character-level embeddings in the experiments and follow a simplified architecture described in the Stanford CS 224N project handout (Francois Chaubard, Michael Fang, Guillaume Genthial, Rohit Mundra, 2019), as shown in Figure 3.9.

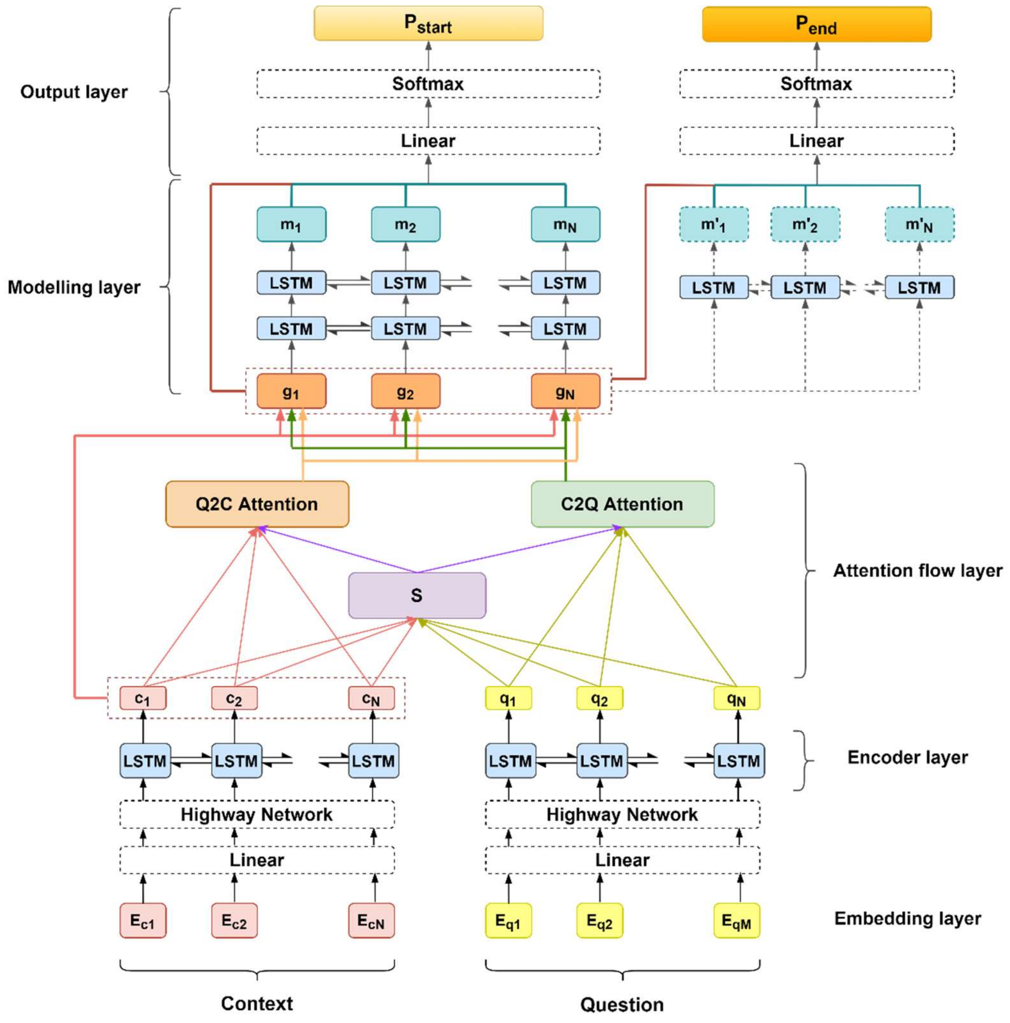


Figure 3.9: The architecture of the bidirectional attention flow (BiDAF) model.

Word embedding layer

The word embedding layer maps individual words in the context and question into fixed dimension vectors, by the pre-trained biomedical word vectors (Pyysalo *et al.*, 2013). It generates a context representation $C_E \in \mathbb{R}^{clen \times d}$ and a question representation $Q_E \in \mathbb{R}^{qlen \times d}$, where $clen$ is the number of tokens in the context, $qlen$ is the number of tokens in the question, and d is the embedding dimension. Each row of C_E and Q_E represents a word token in the text. A linear projection is applied to change the dimension of representation vectors from d to h:

$$C_E \in \mathbb{R}^{clen \times d} \rightarrow \overline{C_E} \in \mathbb{R}^{clen \times h} \tag{3.6}$$

$$Q_E \in \mathbb{R}^{qlen \times d} \rightarrow \overline{Q_E} \in \mathbb{R}^{qlen \times h} \tag{3.7}$$

where h is the universal hidden dimension across the model.

Highway network layer

A two-layer highway network is then applied to refine context and question representations. Highway network uses gates to control information flow, which is designed to ease gradient-based training of very deep networks (Srivastava et al., 2015). Highway network feeds the input x to a transform gate f_H and a carry gate f_C, which decides how much the information of input is transformed or carried to generate the output:

$$y = f_H(x) \bullet f_T(x) + f_C(x) \bullet x \tag{3.8}$$

where f_H and f_T are functions consist of a linear projection followed by a non-linear activation function:

$$f_H(x) = \mathrm{ReLU}(W_H x + b_H) \tag{3.9}$$

$$f_T(x) = \mathrm{Sigmoid}(W_T x + b_T) \tag{3.10}$$

and the carry gate f_C is set as $1 - f_T(x)$. After the highway network layer, the context matrix and question matrix are converted from $\overline{C_E}$ and $\overline{Q_E}$ to $\overline{\overline{C_E}} \in \mathbb{R}^{clen \times h}$ and $\overline{\overline{Q_E}} \in \mathbb{R}^{qlen \times h}$, respectively.

Encoder layer

The encoder layer uses a bidirectional LSTM to model the dependency between words at different positions, and generate hidden states for the context and question.

$$C = BiLSTM\left(\bar{\bar{C_E}}\right) \in \mathbb{R}^{clen \times 2h} \tag{3.11}$$

$$Q = BiLSTM\left(\bar{\bar{Q_E}}\right) \in \mathbb{R}^{qlen \times 2h} \tag{3.12}$$

Attention flow layer

The attention flow layer is designed to connect and combine information between context and question from both directions, by calculating the Context-to-Question (C2Q) attention and Question-to-Context (Q2C) attention. Both attentions are calculated based on a similarity matrix $S \in \mathbb{R}^{clen \times qlen}$. The similarity score of the i-th token in the context and the j-th token in the question is computed by:

$$S_{ij} = W_s\left[C_{i:}; Q_{:j}; C_{i:} \circ Q_{:j}\right] \in \mathbb{R} \tag{3.13}$$

where $C_{i:} \in \mathbb{R}^{2h}$ is the i-th row of the context hidden states C from the encoder layer output, $Q_{:j} \in \mathbb{R}^{2h}$ is the j-th column of the question hidden states Q from the encoder layer output, and $W_s \in \mathbb{R}^{6h}$ is the weight vector.

To calculate the C2Q attention, we first take the row-wise softmax of the similarity matrix S to obtain $\bar{S} \in \mathbb{R}^{clen \times qlen}$, where the i-th row of $\bar{S}$, i.e. $\overline{S_{i:}} \in \mathbb{R}^{qlen}$, indicating the importance of each question token to the i-th context token. Then the C2Q attention output A is generated by:

$$A = \bar{S}Q \in \mathbb{R}^{clen \times 2h} \tag{3.14}$$

where each row refers to the context token representation weighted by the question attentions.

Similarly, to calculate the Q2C attention, we take the column-wise softmax of the similarity matrix S to obtain $\bar{\bar{S}} \in \mathbb{R}^{clen \times qlen}$, where the j-th column of $\bar{\bar{S}}$, i.e.

$\bar{\bar{S}}_{:j} \in \mathbb{R}^{clen}$, indicating the importance of each context token to the j-th question token. Then the Q2C attention output B is generated by:

$$B = \bar{S}\bar{\bar{S}}^T C \in \mathbb{R}^{clen \times 2h} \tag{3.15}$$

Finally, the context hidden states C, the C2Q attention output A and the Q2C attention output B is combined to generate the output of the bidirectional attention flow layer:

$$G = [C; A; C \circ A; C \circ B] \in \mathbb{R}^{clen \times 8h} \tag{3.16}$$

Modelling layer

The modelling layer uses a two-layer bidirectional LSTM to capture the dependencies between context tokens weighted by the question attentions and question tokens weighted by the context attentions. The structure is similar to that of the encoder layer, and the difference is the context hidden states are no longer independent with question hidden states. The hidden size in the modelling layer is set to h and the modelling layer output is calculated by:

$$M = \text{BiLSTM}(G) \in \mathbb{R}^{clen \times 2h} \tag{3.17}$$

Output layer

To extract the answer from the context, I need to locate the relative answer position, which is obtained by generating distributions across context tokens for the answer start token and the answer end token separately. The output layer produces the probabilities of the start token and end token by

$$p_{start} = \text{Softmax}(W_{start}[G; M]) \in \mathbb{R}^{clen} \tag{3.18}$$

$$p_{end} = \text{Softmax}(W_{end}\left[G; M'\right]) \in \mathbb{R}^{clen} \tag{3.19}$$

where $M' \in \mathbb{R}^{clen \times 2h}$ is the output from another bidirectional LSTM module. The model is trained to minimise the sum of the cross-entropy loss for the answer start and end indices:

$$Loss = \text{CrossEntropy}(p_{start}, y_{start}) + \text{CrossEntropy}(p_{end}, y_{end})$$

$$= \frac{-1}{clen} \sum_{i=1}^{clen} \begin{bmatrix} y_{start} \log(p_{start}) + (1 - y_{start}) \log(1 - p_{start}) \\ +y_{end} \log(p_{end}) + (1 - y_{end}) \log(1 - p_{end}) \end{bmatrix} \quad (3.20)$$

where y_{start}, $y_{end} \in \mathbb{R}^{clen}$ are the one-hot vectors indicating the true positions of answer start token and answer end token. For evaluation, I obtain the answer start index idx_{start} and the answer end index idx_{end} from the pairwise elements in $(p_{i,start}, p_{i,end})$ which has the maximum joint probability $p_{i,start} \times p_{i,end}$, $i = 1, \dots, clen$, subject to the constraints $\mathrm{idx}_{start} < \mathrm{idx}_{end}$ and $\mathrm{idx}_{end} - \mathrm{idx}_{start} <$ maximum answer length. I then convert idx_{start} and idx_{end} back to the actual text tokens and compare it with the true answer text. The calculation details of the evaluation metrics are described in Section 3.3.2.4.

3.3.2.2 QANet

QANet (Yu *et al.*, 2018) is another classical question answering model which has a similar structure to BiDAF but replaces the recurrent modules (LSTM blocks) by the depthwise separable convolution (Kaiser *et al.*, 2017) and self-attention mechanism (Vaswani *et al.*, 2017). The motivation to remove recurrent blocks is to speed up the training process and provide the possibility to inject more attention modules. From the architecture demonstrated in Figure 3.10, the input embedding layer, attention flow layer and output layer are same as that of BiDAF, while the embedding encoder layer and modelling encoder layer are different which are described in greater detail below.

The main elements in the embedding encoder layer and modelling encoder layer are the stacked encoder blocks which consist of repeated separable depthwise convolution layers, a self-attention layer and a feedforward network followed with layer normalisation each (see the blue block in the right top of Figure 3.10). The positional encoding input, multi-head self-attention and feedforward network are same as described in the transformer architecture (see Section 2.3.4.1).

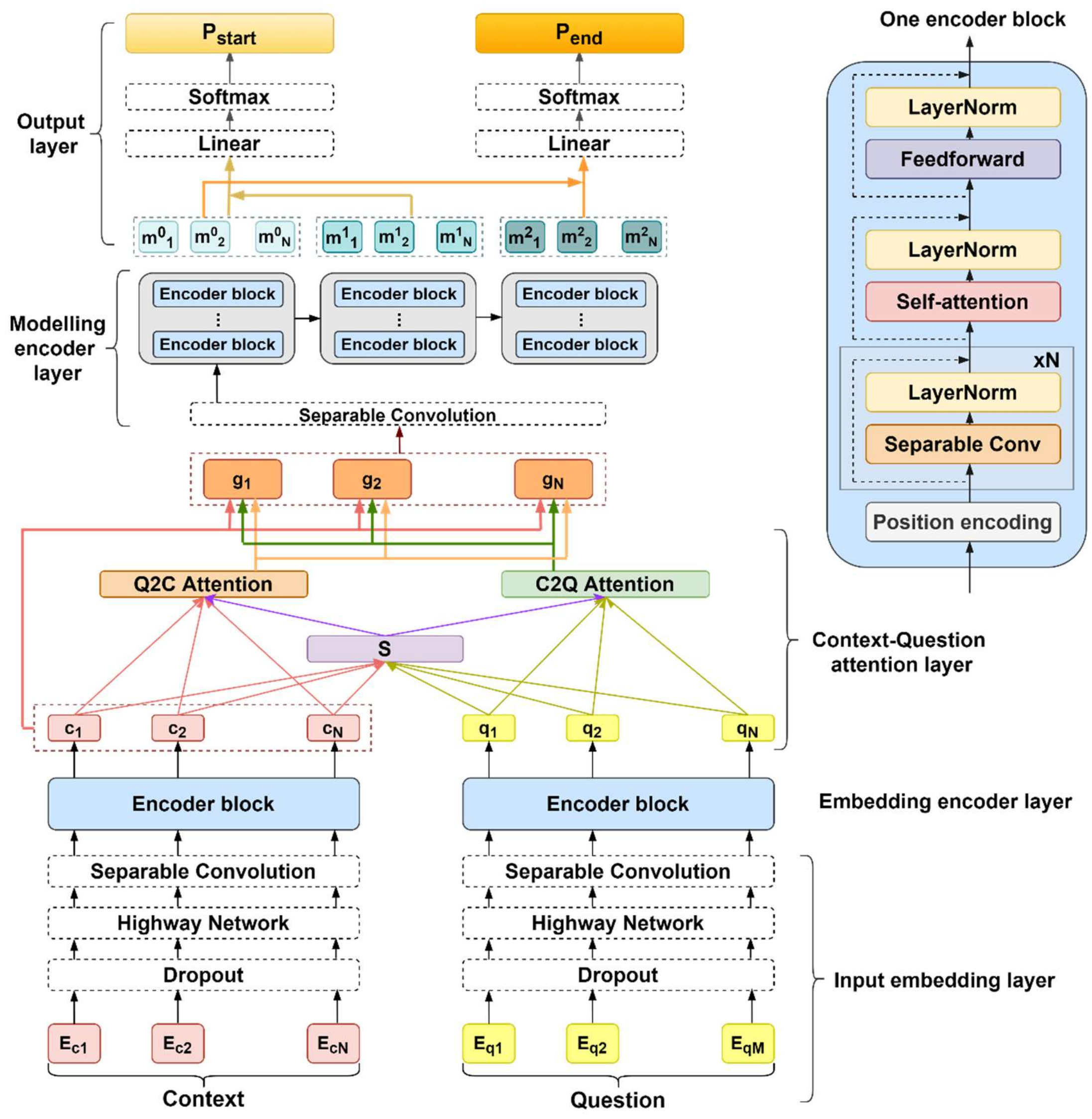


Figure 3.10: The architecture of QANet model.

The depthwise separable convolution is proposed to reduce the number of operations for more efficient training (Chollet, 2017). I demonstrate the advantage of depthwise separable convolution over normal convolution by an example displayed in Figure 3.11. The sequence in the example consists of 7 tokens and each token is represented by a 4-dimensional vector, so the sequence is encoded by a matrix $\in \mathbb{R}^{7\times 4}$. After padding, the normal convolution uses two filters $\in \mathbb{R}^{3\times 6}$ to generate a new representation matrix $\in \mathbb{R}^{7\times 2}$ for the sequence, which involves $3\times 6\times (9-3+1)\times 2 = 252$ multiplications. In the depthwise separate convolution, the padded input matrix is first passed to the depthwise convolution, using four separate filters $\in \mathbb{R}^{3\times1}$ to move vertically through the matrix and generate an intermediate representation matrix $\in \mathbb{R}^{7\times4}$; then to the pointwise convolution, using two filters $\in \mathbb{R}^{3\times1}$ to generate the final output matrix $\in \mathbb{R}^{7\times2}$ of the whole convolution operation. The two separate convolutions involve $4 \times 3 \times (9 - 3 + 1) + 7 \times 4 \times 2 = 140$ multiplications, which are almost half less than the normal convolution. Applying depthwise separable convolution is more beneficial when the sequence length, hidden dimension or the number of filters is large. In my implementation, for dimension consistency, the depthwise separable convolution is also applied as the connection between the input embedding layer and embedding encoder layer, and between the context-question attention layer and modelling encoder layer, which is not explicitly described in the original work of QANet (Yu *et al.*, 2018).

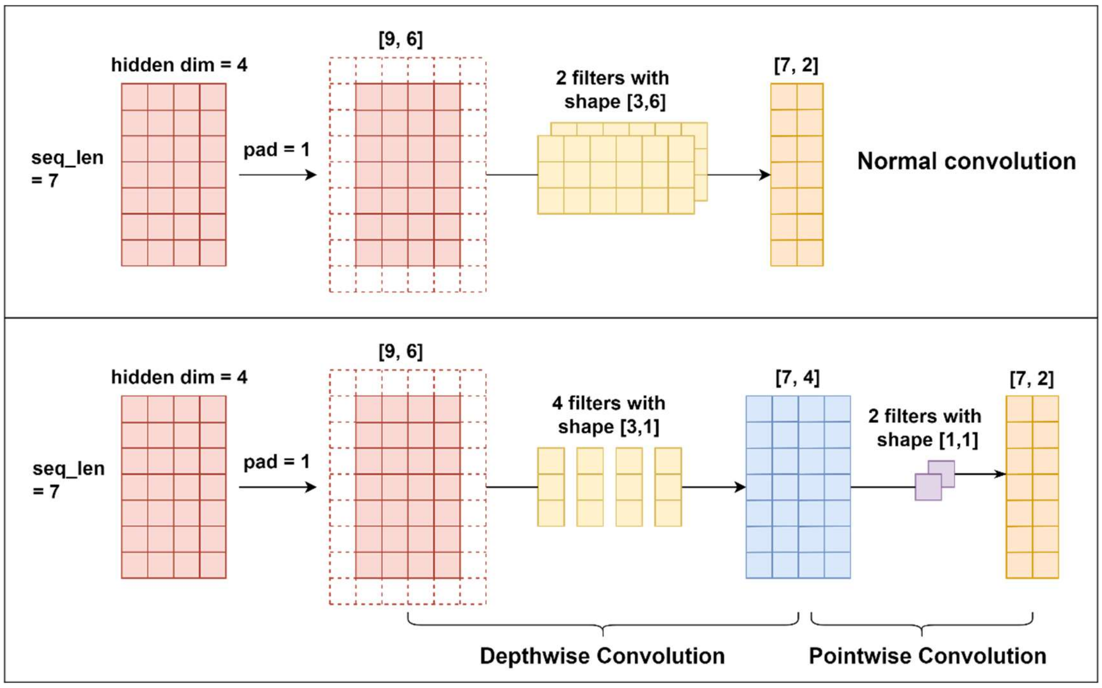


Figure 3.11: A comparison of the normal convolution and depthwise separable convolution which includes the depthwise convolution and pointwise convolution.

Another difference between QANet and BiDAF is how to generate the output matrix M of the modelling layer from the output G of the context-question layer, which will be used to calculate probabilities of answer start token and answer end token.

BiDAF uses two different encoders (LSTMs) to generate separate M and M', while QANet uses three identical encoders (stacked encoder blocks described earlier) which share the same weights to generate M^0, M^1 and M^2 sequentially:

$$M^0 = \text{ModelEncoder}(G) \tag{3.21}$$

$$M^1 = \text{ModelEncoder}\left(M^0\right) \tag{3.22}$$

$$M^2 = \text{ModelEncoder}\left(M^1\right) \tag{3.23}$$

Then the probabilities of the answer start tokenand answer end token from the output layer are computed by

$$p_{start} = \text{Softmax}(W_{start}\left[M^0; M^1\right]) \tag{3.24}$$

$$p_{end} = \text{Softmax}(W_{end}\left[M^0; M^2\right] \tag{3.25}$$

The training loss calculation and evaluation methods from these probabilities are same as that of BiDAF (see Section 3.3.2.1).

3.3.2.3 Fine-tuning BERT

Transformer models like BERT (Devlin *et al.*, 2018) or DistilBERT (Sanh *et al.*, 2019) can be also trained for question answering tasks. As Figure 3.12 shows, words in each context-question pair are concatenated, tokenised by WordPiece (Wu *et al.*, 2016) and then encoded into embeddings as input, where '[CLS]' refers to the conventional classification token and '[SEP]' represents the segmentation of the context and question sequences. BERT/DistilBERT combined with a linear layer is then fine-tuned to generate log probabilities for each token, and the indexes of answer tokens are inferred from the maximum joint probabilities ($p_{start}{*}p_{end}$), subject to $\text{idx}_{start} < \text{idx}_{end}$ and $\text{idx}_{end} - \text{idx}_{start} <$ maximum answer length. For pre-trained modules, I explore similar modules as described in Section 3.3.1, including DistilBERT-Base

(Sanh *et al.*, 2019), BERT-base (Devlin *et al.*, 2018), BioBERT-Base (Lee *et al.*, 2019), PubMedBERT-Abs, and PubMedBERT-Abs-Full (Gu *et al.*, 2020).

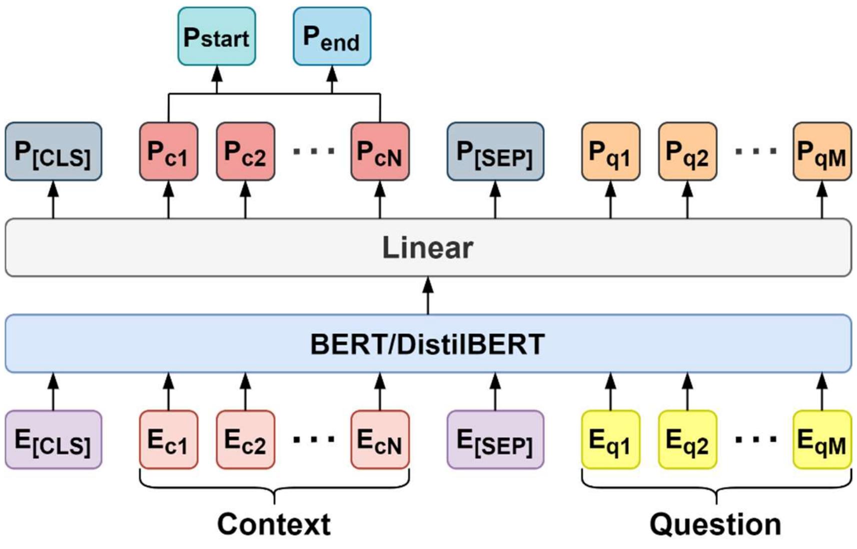


Figure 3.12: Fine-tuning BERT/DistilBERT with a linear layer for question answering. ‘E’ and ‘P’ refers to the embedding and probability of each token in the sequence, respectively.

3.3.2.4 Evaluation Metrics

Metrics for factoid-type dataset

For the MND dataset and Psycho-CIPN-factoid dataset which have only one answer for each question, i.e. factoid-type datasets, the performance of a QA model is assessed by the Exact Match (EM) score and F1 score. For each context-question record, EM is marked as 1 if the predicted answer string is exactly same as the true answer string; otherwise 0. To calculate F1 score, I first compute the individual precision and recall for each record by:

$$p_i^{\text{factoid}} = \frac{\text{number of predicted tokens which belong to the true tokens}}{\text{number of predicted tokens}} \tag{3.26}$$

$$r_i^{\text{factoid}} = \frac{\text{number of predicted tokens which belong to the true tokens}}{\text{number of true tokens}} \tag{3.27}$$

For example, in one record, if the true answer is "timed light therapy", and the predicted answer from the model is "light therapy", then the precision score is

2/2 = 100% (because the predicted answer tokens "light" and "therapy" all belong to the true answer tokens), and the recall is 2/3 = 66.7% (because the model predicts two tokens out of the true answer tokens). Then the individual F1 score is the harmonic mean of precision and recall: $F1_i^{\text{factoid}} = 2 \times p_i^{\text{factoid}} \times r_i^{\text{factoid}} \big/ (p_i^{\text{factoid}} + r_i^{\text{factoid}})$. The overall EM and F1 scores are then obtained by averaging all individual EM and F1 scores across the entire evaluation set.

Metrics for list-type dataset

For the Psycho-CIPN-list dataset, each sub-record is learned to generate fixed-number answer candidates, which are then concatenated into an answer list for the parent record. A threshold is used to filter answer candidates in the list: candidates with the joint probabilities higher than the threshold form the final answer list for the parent record. Although the training process is conducted on the sub-records, the evaluation should be performed on the parent record.

As the parent record may have more than one answer candidate and each candidate may consist of several tokens, I first calculate precision and recall for every pair of true and predicted answer candidates:

$$p_i^{\text{list}} = \frac{1}{TK} \sum_{t=1}^{T} \sum_{k=1}^{K} p_{t,k}^{\text{factoid}} \tag{3.28}$$

$$r_i^{\text{list}} = \frac{1}{TK} \sum_{t=1}^{T} \sum_{k=1}^{K} r_{t,k}^{\text{factoid}} \tag{3.29}$$

where T is the number of true answer candidates, K is the number of predicted answer candidates; $p_{t,k}^{\text{factoid}}$ ($r_{t,k}^{\text{factoid}}$) refers to the precision (recall) of the t-th candidate in the true answer list and the k-th candidate in the predicted answer list, as how the precision (recall) for each factoid-type record is calculated. Then F1 score of the i-th record is $F1_i^{\text{list}} = 2 \times p_i^{\text{list}} \times r_i^{\text{list}} \big/ (p_i^{\text{list}} + r_i^{\text{list}})$. For example, in one record, the true answer candidates are [“isolation rearing”, “DNA dopamine aptamer”, “clozapine”] and the predicted answer candidates are [“clozapine”, “dopamine aptamer”]. After calculating the precision and recall of each pairwise candidates (Table 3.2), the precision of this record is (1+2/2)/6 = 33%, and recall is (1+2/3)/6 = 28%, so the F1 score is 2*33%*28%/(33%+28%) = 30%.

Pred / **True**	clozapine	dopamine, aptamer

isolation, rearing	p = 0
	r = 0
		p = 0
		r = 0

DNA, dopamine, aptamer	p = 0
	r = 0
		p = 2/2
		r = 2/3

clozapine	p = 1
	r = 1
		p = 0
		r = 0

Table 3.2: An example of precision and recall calculation for each pair of true and predicted answer candidates in one list-type record with multiple answer candidates.

The overall F1 score is then obtained by averaging all the individual F1 scores across the evaluation set.

3.4 Experiments

Processed datasets are available in the Intervention/Induction Identification by Question Answering repository (osf.io/wr4xa) and codes of experiments are available at github.com/qianyingw/bioqa.

3.4.1 Experiments of Sentence Retrieval

I apply three sentence retrieval methods for the overall 1,225 Psycho-CIPN records (pre-processed Psycho-CIPN-list dataset before splitting into sub-records) and 796 factoid-type records. For TF-IDF and BM25 methods, stop words and punctuations are removed, and tokenisation is implemented via scispaCy (Neumann *et al.*, 2019). SBERT uses WordPiece tokenisation (Wu *et al.*, 2016) like most transformer models. Sentence retrieval and ranking are implemented using scikit-learn (Pedregosa *et al.*, 2011), Gensim (Rehurek and Sojka, 2010) and Sentence-Transformers (Reimers and Gurevych, 2019) for three retrieval methods respectively. As a default configuration, I use unigram features in TF-IDF and ‘DistilBERT-NLI-STSb’ for the pre-trained weights in SBERT.

I explore the effect of number of sentences on the strict mean match ratio (sMMR) and strict mean average precision (sMAP) among three retrieval methods. For the overall 1,225 Psycho-CIPN records, around 65% of passages concatenated from the top 5 relevant sentences contain all answer candidates, and achieving 90% of the strict mean match ratio requires over 30 sentences to be extracted. For the 796 factoid records, around 75% of passages concatenated from the top 5 relevant sentences contain all answer candidates and achieving 90% of the strict mean match ratio requires over 20 sentences to be extracted. In the overall dataset, SBERT shows slightly better performance than TF-IDF and BM25, with an average 1.8% and 2.8% improvement respectively; while in the factoid dataset, the improvements are less obvious (2.1% and 0.9% respectively), as demonstrated in Figure 3.13 and Appendix Table 10-11. The strict mean average precision shows a decreasing trend when the number of extracted sentences increases, for all three retrieval methods in two datasets. This is because compared to the increased relevance of answers included, the number of sentences retrieved has a larger impact on the average precisions. SBERT outperforms other two methods with strict mean average precision improved by between 2% and 7%.

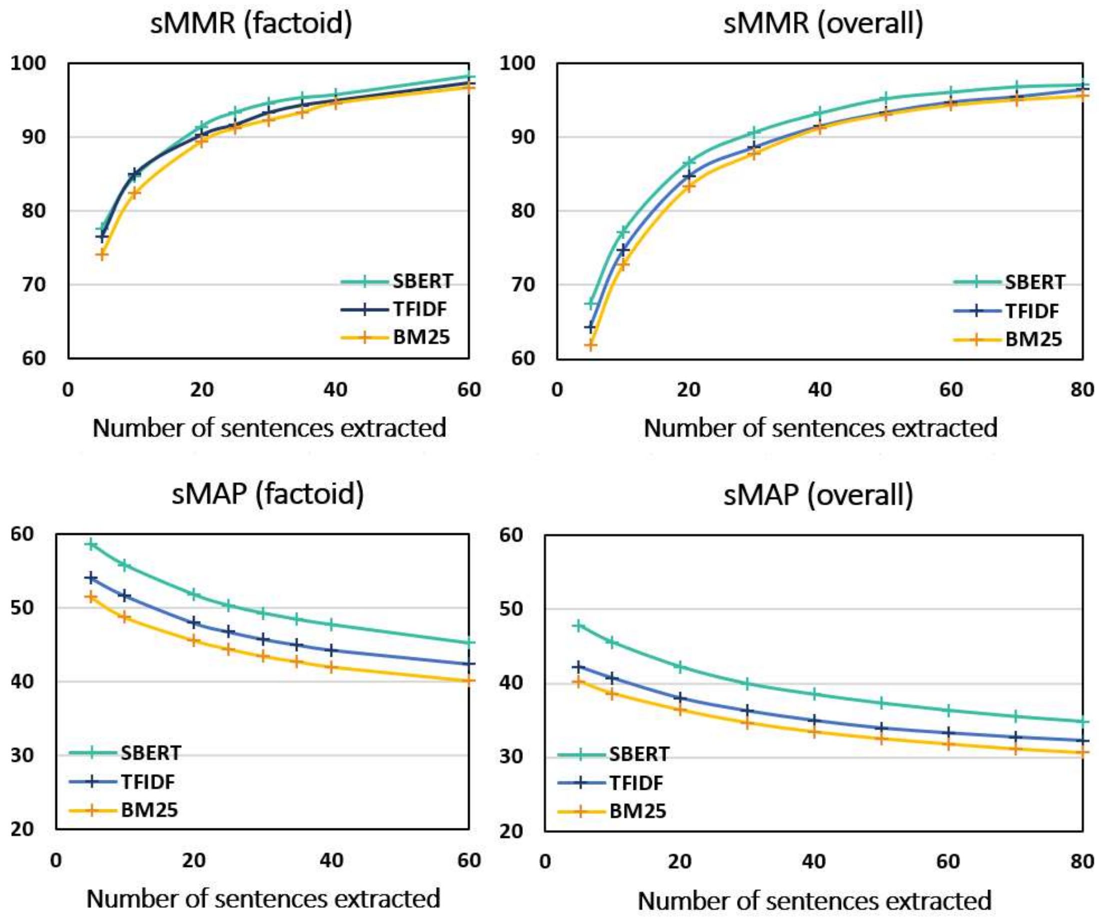


Figure 3.13: Effect of number of sentences extracted on strict mean match ratio (sMMR) and strict mean average precision (sMAP) for three sentence retrieval methods. Charts on the left are from the Psycho-CIPN-factoid dataset; charts on the right are from the overall Psycho-CIPN dataset (Psycho-CIPN-list dataset before splitting into sub-records).

I also explore the effect of n-grams in TF-IDF and the effect of pre-trained modules in SBERT, as shown in Table 3.3. TF-IDF with unigram performs better than that with bigram (2.4% improvement of strict mean match ratio) or its combination with bigram (0.6% improvement). For SBERT method, different pre-trained modules do not show much difference, but 'DistilBERT-NLI-STSb' gives slightly better scores and also spends at least twice less time than other modules. Therefore, I use SBERT with 'DistilBERT-NLI-STSb' as the final choice for the retrieval module to extract sentences and construct the input of the next question answering stage. For the Psycho-CIPN-factoid dataset, I use the passages concatenated from the top 20 sentences retrieved from the full texts, with sMMR of 91.5%; while for the Psycho-CIPN-list dataset, I use the passages are concatenated from the top 30 sentences, with sMMR of 90.7%.

Method	Setting	sMMR	rMMR	sMAP	rMAP
TF-IDF	Unigram	**88.7**	**91.5**	**36.3**	**46.8**
	Bigram	86.3	89.5	33.6	42.9
	Uni- & Bi-gram	88.1	91.2	35.6	45.9
SBERT	DistilBERT-NLI-STSb	**90.7**	**93.4**	40.0	51.6
	DistilBERT-Marco	90.0	92.7	41.1	53.7
	BERT-Base	90.1	93.0	41.5	54.4
	BioBERT	89.4	92.5	**43.2**	**55.6**
	PubMedBERT-Abs	89.1	91.7	37.9	50.0
	PubMedBERT-Abs-Full	87.7	91.0	38.3	50.5

Table 3.3: Effect of n-grams for TF-IDF method and pre-trained modules for SBERT on sentence retrieval performance. sMMR (rMMR) refers to strict (ratio) mean match ratio, and sMAP (rMAP) refers to strict (ratio) mean average precision. Here sMMR is the most important metric as it directly indicates whether the new passage contains the exact answer tokens.

3.4.2 Experiments of Question Answering Models

For all three models (BiDAF, QANet and fine-tuning BERT), parameters are trained to minimise the sum of the cross-entropy loss of answer start and answer end tokens using AdamW algorithm (Loshchilov and Hutter, 2017). I apply gradient clipping with the threshold norm 0.1 to rescale gradients and use gradient accumulation every 4 mini-batches to reduce memory consumption. The warm-up fraction in the slanted triangular learning rate scheduler (Howard and Ruder, 2018) is set to 0.1. Dropout rate is set to 0.1 for all models. The maximum number of tokens in an answer string is 15. All three datasets (MND, Psycho-CIPN-factoid, and Psycho-CIPN-list) are split into train (80%), validation (10%) and test (10%) set separately. Scores from the epoch with the minimum validation loss are recorded for comparison.

3.4.2.1 Experiments of BiDAF

For both two factoid datasets (MND and Psycho-CIPN-factoid), batch size is set to 32. Models are trained for 20 epochs. I explore the threshold learning rate in the slanted triangular scheduler and set it to 1e-3 in the end. The similarity matrix in the attention flow layer is initialised using Kaiming initialisation (He *et al.*, 2015). For the Psycho-CIPN-list dataset, I set the maximum number of answers extracted from each sub-records to 5 and the threshold learning rate to 1e-4. Models are trained for 40 epochs. I tune the threshold for filtering answer candidates and use 0.1 as the final choice which yields the highest F1 score of 13.9%, as demonstrated in Figure 3.14.

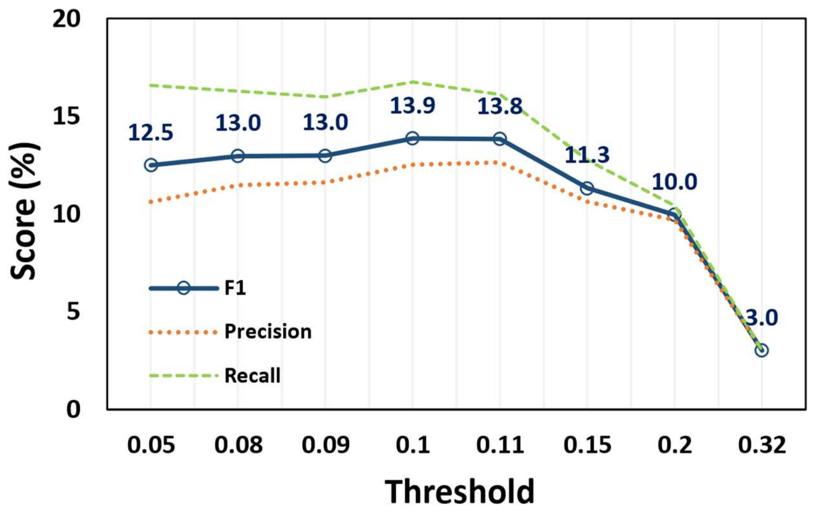


Figure 3.14: Validation performance of BiDAF model with different answer thresholds on the preclinical Psycho-CIPN-list dataset.

With other configurations unchanged, I tune the universal hidden dimension in the model among {32, 64, 128, 256, 512} and the validation performance are reported in Table 3.4. I do not explore a larger dimension because of the over-fitting issue. The BiDAF model achieves good performance on MND data, with the highest exact match around 60% and F1 around 80%. When the hidden dimension increases from 64 to 512, the validation F1 changes by between 2% and 6%. For the Psycho-CIPN-factoid dataset, the best performance is achieved when the hidden dimension is 512, with F1 around 40%. Models achieve poor performance on the Psycho-CIPN-list data, with F1 of only 13%, and the effect of the hidden dimension is trivial when the dimension changes from 128 to 512, with the fluctuation of F1 within 2%.

Dataset	Dimension	EM	F1	Recall	Precision
MND	32	35.2	59.8	61.5	71.5
	64	58.8	76.6	76.8	82.6
	128	56.8	76.1	75.5	84.7
	256	**61.3**	**79.0**	**79.6**	**85.0**
	512	54.8	76.9	78.9	85.0
Psycho-CIPN -factoid	32	26.6	27.2	27.0	27.8
	64	31.6	32.6	33.3	33.1
	128	38.0	38.0	38.0	38.0
	256	39.2	39.2	39.2	39.2
	512	**41.8**	**41.8**	**41.8**	**41.8**
Psycho-CIPN -list	32*	--	5.9	7.3	5.4
	64*	--	7.6	10.1	6.5
	128	--	12.0	14.2	11.1
	256	--	13.8	16.1	12.7
	512	--	12.4	14.5	11.5

Table 3.4: Validation performance of BiDAF model with different hidden dimensions on the clinical MND dataset and preclinical Psycho-CIPN-factoid/list dataset. * For the Psycho-CIPN-list dataset, I increase the threshold learning rate to 1e-3 when the hidden dimension equals 32 or 64, because models with learning rate 1e-4 do not converge after 40 epochs.

3.4.2.2 Experiments of QANet

I use the same setting as described in the original QANet model (Yu *et al.*, 2018): in the embedding encoder layer, the number of separable convolution layers is 4, filter size is 7 and the number of blocks in the encoder is 1; in the modelling encoder layer, the number of separable convolution layers is 2, filter size is 5 and number of blocks in the encoder is 7. Models are trained for 20 epochs. The similarity matrix in the attention flow layer is initialised using

Kaiming initialisation (He *et al.*, 2015) with the values of weights W following a normal distribution $W \sim N[0, \frac{2}{h}]$. I try different threshold learning rates in the slanted triangular scheduler and set the value to 1e-4 for MND and Psycho-CIPN-list data, and 1e-3 for Psycho-CIPN-factoid data. I also explore different maximum context length (number of tokens in the context) and set the value to 512 for MND and 768 for Psycho-CIPN data. Batch size is set to 16. For Psycho-CIPN-list dataset, the number of answers extracted from each sub-records is 5. I tune the threshold for filtering answer candidates and value 0.13 achieves the highest F1 score of 14.8%, as demonstrated in Figure 3.15.

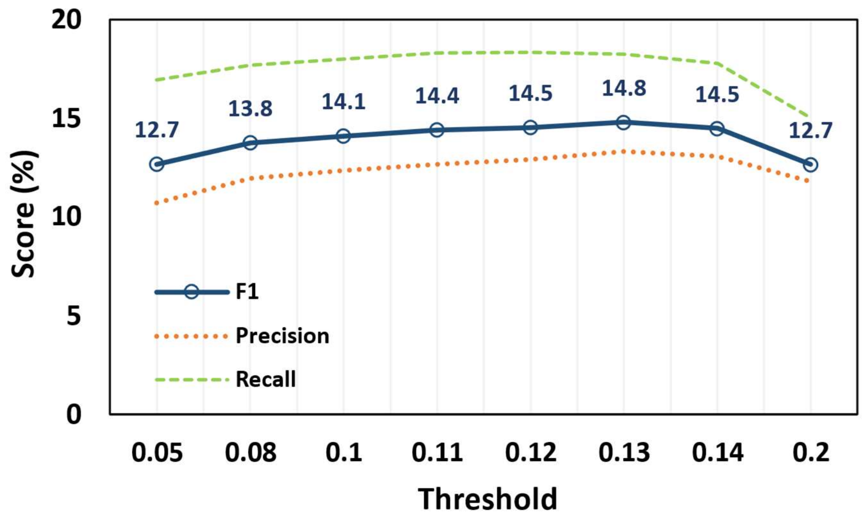


Figure 3.15: Validation performance of QANet model with different answer threshold on the Psycho-CIPN-list dataset.

I tune the number of encoder blocks in the model encoder layer and the performance is reported in Table 3.5. For MND data, the highest F1 (79.9%) is achieved by 7 blocks in the model encoder; while for Psycho-CIPN-factoid data, the highest F1 (45.6%) is achieved by 5 blocks. The deficiency in the performance of models with a small number of blocks is larger in Psycho-CIPN-factoid data than that in MND data, with 6% and 12% reduction of F1 respectively. However, for Psycho-CIPN-list data, the model using one block in the model encoder yields the best performance. Compared to BiDAF (Table

3.4), the validation F1 is slightly improved by 0.9%, 3.8% and 1% for MND, Psycho-CIPN-factoid and Psycho-CIPN-list datasets respectively.

Dataset	No. blocks	EM	F1	Recall	Precision
MND	7	**65.8**	**79.9**	**83.2**	**84.3**
	5	62.8	78.6	84.3	81.5
	3	62.3	79.2	84.0	83.8
	1	59.3	75.8	80.2	80.9
Psycho-CIPN -factoid	7	44.3	44.5	45.6	44.4
	5	**45.6**	**45.6**	**45.6**	**45.6**
	3	31.6	31.6	31.6	31.6
	1	32.9	32.9	32.9	32.9
Psycho-CIPN -list	7	--	12.3	15.7	10.7
	5	--	14.1	17.2	12.6
	3	--	14.1	18.1	12.2
	1	--	**14.8**	**18.2**	**13.3**

Table 3.5: Validation performance of QANet model with different number of blocks in the model encoder layer on the clinical MND dataset and preclinical Psycho-CIPN-factoid/list dataset.

3.4.2.3 Experiments of Fine-tuning BERT

For all three datasets, bath size is set to 32 and the threshold learning rate in the slanted triangular scheduler is set to 5e-5. For the MND dataset, models are trained for 20 epochs for all pre-trained modules; for Psycho-CIPN-factoid/list datasets, models are trained for 30 epochs for the DistilBERT-Base and BERT-Base modules, 20 epochs for BioBERT-Base, and 40 epochs for the PubMedBERT-Abs and PubMedBERT-Abs-Full modules. These are determined based on the learning curves from some initial experiments. For the Psycho-CIPN-list dataset, I set the number of answers extracted from each sub-records to 5. I also explore extracting 3 or 7 answers, but the changes in performance are negligible. I tune the threshold for filtering answer candidates in the BERT/DistilBERT model and use 0.32 as the final value as it achieves the highest F1 score of 30.8%, as demonstrated in Figure 3.16.

Table 3.6 demonstrates the validation performance of fine-tuning BERT/DistilBERT with different pre-trained modules on three datasets. For the MND data, all five modules achieve good performance with F1 over 84%, and BioBERT and PubMedBERT pre-trained on abstracts give the highest F1 score of 88%. For the Psycho-CIPN-factoid data, fine-tuning PubMedBERT pre-trained only on the abstracts yields the highest F1 score of 92%, while fine-tuning the module pre-trained on the combination of abstracts and full texts reduces the validation F1 by 3%. The other three pre-trained modules harm the performance severely, where F1 declines by around 20%, particularly for that of BioBERT, which has the poorest F1 of 66%. The performance on the Psycho-CIPN-list dataset is much lower than that of factoid-type datasets, as it is more difficult to filter the accurate number of answer candidates and extract strings. The best performances are achieved by the two PubMedBERT modules, with F1 around 44%. The performance of BioBERT is slightly lower (1.5% reduction), while that of DistilBERT and BERT-Base are 15% lower, compared to the best module.

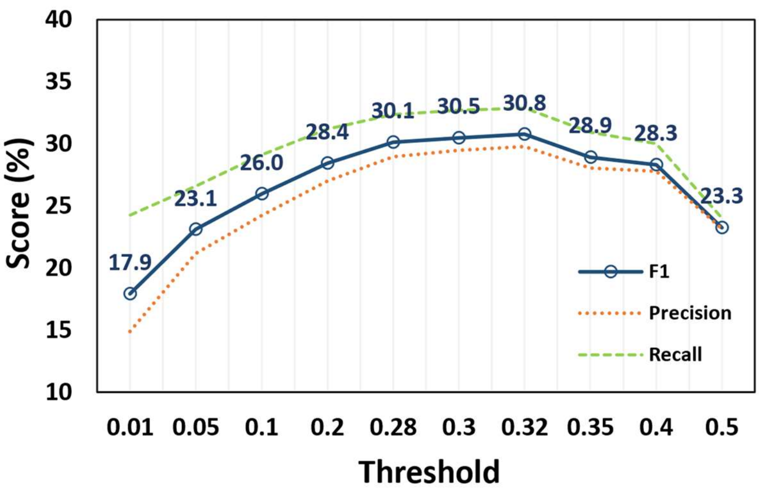


Figure 3.16: Validation performance of DistilBERT model with different answer thresholds on the Psycho-CIPN-list dataset.

Dataset	Pre-trained module	EM	F1	Recall	Precision
MND	DistilBERT-Base	76.9	87.4	88.1	89.7
	BERT-Base	73.9	84.1	85.0	86.0
	BioBERT-Base	78.4	88.0	88.0	91.0
	PubMedBERT-Abs	78.4	88.0	88.4	90.3
	PubMedBERT-Abs-Full	76.4	86.6	88.3	88.3
Psycho-CIPN -factoid	DistilBERT-Base	68.4	68.4	68.4	68.4
	BERT-Base	70.9	70.9	70.9	70.9
	BioBERT-Base	65.8	65.8	65.8	65.8
	PubMedBERT-Abs	91.1	92.4	92.8	92.8
	PubMedBERT-Abs-Full	89.9	89.9	89.9	89.9
Psycho-CIPN -list	DistilBERT-Base	--	30.8	32.9	29.8
	BERT-Base	--	28.9	30.8	28.0
	BioBERT-Base	--	42.9	44.6	42.1
	PubMedBERT-Abs	--	44.4	46.5	43.3
	PubMedBERT-Abs-Full	--	43.7	45.6	42.9

Table 3.6: Performance of fine-tuning BERT/DistilBERT with different pre-trained modules on the clinical MND dataset and preclinical Psycho-CIPN-factoid/list dataset.

3.4.3 Overall Performance

Table 3.7 demonstrates the overall validation performance of BiDAF, QANet and BERT with the corresponding optimal settings on the MND and Psycho-CIPN-factoid/list datasets. Compared to BiDAF and QANet, BERT with the optimal pre-trained module improves the performance significantly, particularly for the preclinical datasets, by 50% for the factoid-type data and 30% for the list-type data. Fine-tuning BERT also requires less parameter tuning and achieves lower loss easily without overfitting, as shown in the learning curves in Figure 3.17.

Dataset	Model	EM	F1	Recall	Precision
MND	BiDAF	61.3	79.0	79.6	85.0
	QANet	65.8	79.9	83.2	84.3
	BioBERT-Base	78.4	88.0	88.0	91.0
Psycho-CIPN -factoid	BiDAF	41.8	41.8	41.8	41.8
	QANet	45.6	45.6	45.6	45.6
	PubMedBERT-Abs	91.1	92.4	92.8	92.8
Psycho-CIPN -list	BiDAF	--	13.9	16.7	12.5
	QANet	--	14.8	18.2	13.3
	PubMedBERT-Abs	--	44.4	46.5	43.3

Table 3.7: Overall validation performance of BiDAF, QANet and BERT with their optimal settings on the clinical MND, preclinical Psycho-CIPN-factoid/list datasets.

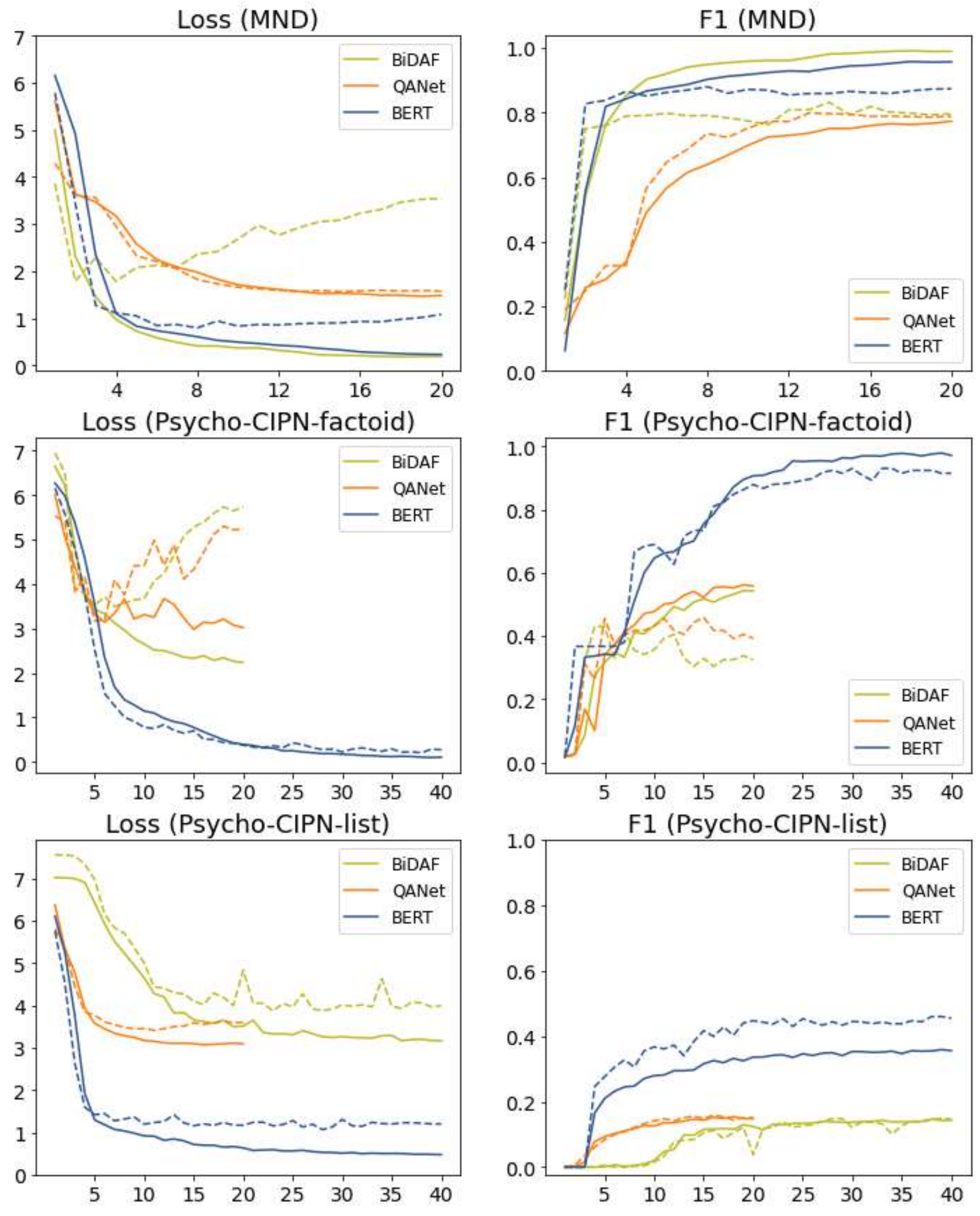
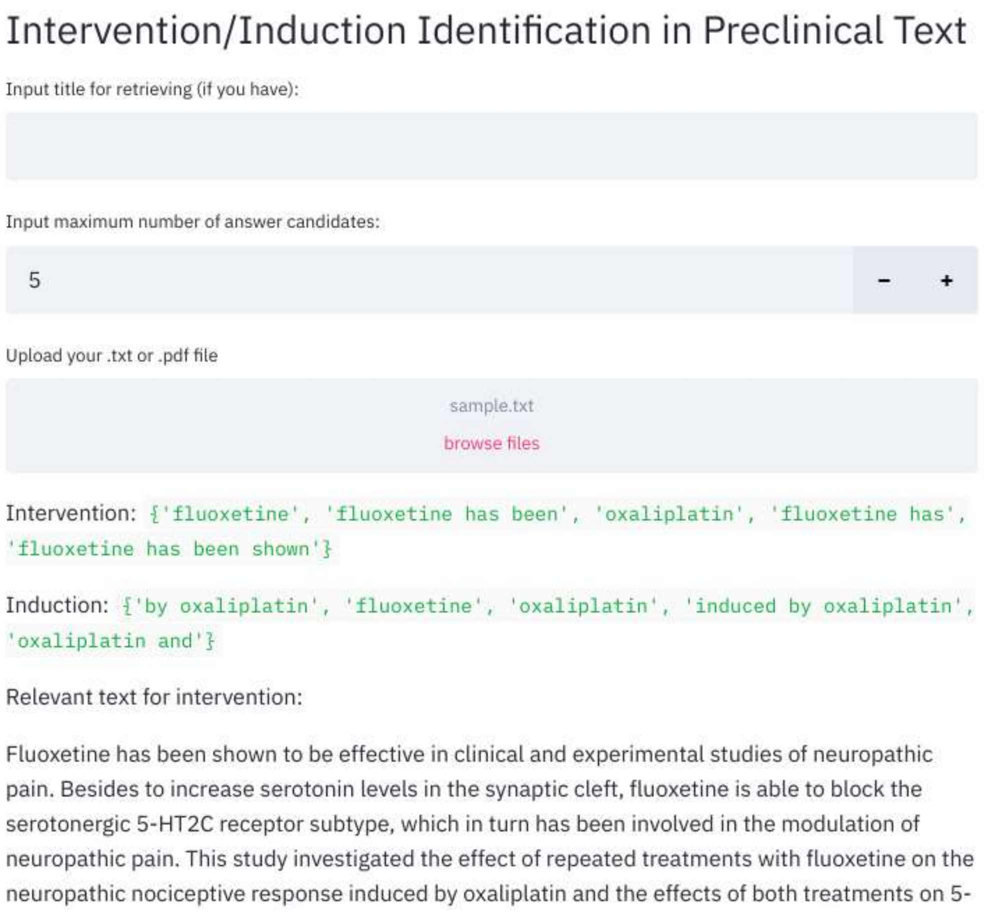


Figure 3.17: Learning curves of BiDAF, QANet and BERT models with their optimal settings on three datasets. Solid and dash lines represent training curves and validation curves separately.

3.5 Discussion

3.5.1 Streamlit Interface

For a potential application, I develop an answer extraction module for identifying intervention and method of model induction in preclinical abstracts or full texts. In real-world cases, we cannot know how many interventions or methods of induction are described in an abstract or a full-text paper, so models developed for MND and Psycho-CIPN-factoid datasets aiming to extract only one answer are not suitable. Hence, I build the answer extraction module using the best model trained for the Psycho-CIPN-list dataset, which can extract a list of answer candidates (test performance: 48.3% of F1, 51.6% of recall, 46.6% of precision; see validation performance in Table 3.7). If the input text is an abstract or a paragraph with abstract length, answers can be extracted by the question answering model directly, as demonstrated in the interface developed via Streamlit (Figure 3.18). If the input text is a full-text paper converted from PDF, the module first cleans the text including removing references and sections before ‘Introduction’, lines with digits/punctuations/line character only, and non-ASCII characters. Users are required to input the title for the sentence retrieval function constructed on SBERT with the ‘DistilBERT-NLI-STSb’ pre-trained module. 30 sentences extracted from the cleaned text are then concatenated as the final context for the question answering model. The maximum number of answer candidates are determined by users (5 candidates in default), and candidates consist of digits, punctuations and stop words only are removed from the final answer list. Relevant contexts for intervention and method of induction are also displayed to provide clues for users.


Intervention/Induction Identification in Preclinical Text

Input title for retrieving (if you have):

Input maximum number of answer candidates:

5

Upload your .txt or .pdf file

sample.txt

Intervention: {'fluoxetine', 'fluoxetine has been', 'oxaliplatin', 'fluoxetine has', 'fluoxetine has been shown'}

Induction: {'by oxaliplatin', 'fluoxetine', 'oxaliplatin', 'induced by oxaliplatin', 'oxaliplatin and'}

Relevant text for intervention:

Fluoxetine has been shown to be effective in clinical and experimental studies of neuropathic pain. Besides to increase serotonin levels in the synaptic cleft, fluoxetine is able to block the serotonergic 5-HT2C receptor subtype, which in turn has been involved in the modulation of neuropathic pain. This study investigated the effect of repeated treatments with fluoxetine on the neuropathic nociceptive response induced by oxaliplatin and the effects of both treatments on 5-

Figure 3.18: A demonstration Streamlit interface for intervention/induction identification in preclinical text.

3.5.2 Error Analysis

I randomly select 10 preclinical publications to inspect the answer extraction function. For each publication, I obtain the predicted answer candidates in two ways: 1) upload only abstract, and obtain 10 candidates from the abstract; 2) upload the full text converted from PDF, input title, get relevant sentences and obtain 10 candidates from the text concatenated from those sentences. I compare the two candidate lists and find that for some records, answers extracted from the abstract are more accurate than answers extracted from the 'shortened' full text; while for other records, vice versa. It may depend on whether the abstract mentions intervention or induction, and whether the sentence retrieval module can detect the true relevant sentences. The module sometimes ranks sentences from the introduction and discussion sections higher than sentences from the method or result sections, which are unlikely to be the best answer positions and may affect the accuracy of the answer extraction.

The main error is the module does not handle well with the boundary of answer strings. Some of the difficulties are from the punctuations in some chemical names. For example, the true interventions of a publication is ‘WIN 55,212-2’ and ‘minocycline’, while the top 5 answer candidates predicted from my module are {'WIN', 'WIN 55, 212', 'WIN 55, 212-', '55, 212-2', '212-2', 'WIN 55', ', 212-2', '-2', 'in the present', 'in the present study'}. It does not identify the full string of ‘WIN 55,212-2’, and ranks several pieces of the string as the top candidates. This may make other potential candidates do not appear at the top of the list. Another error is the module does not capture the complete name of abbreviations. For instance, a method of model induction is ‘Stable Tubule Only Peptide’ which is abbreviated as ‘STOP’. 5 out of 10 answer candidates contain ‘STOP’ but none of the candidates contains the full name. For users who do not check context may not know the meaning of the abbreviation defined by authors.

3.5.3 Limitation and Future Work

The work has several limitations. First, the preclinical dataset is limited to two diseases (psychosis disorder and chemotherapy-induced peripheral neuropathy). Although there are 518 unique interventions and 79 unique induction methods in the overall 1,225 records, the optimal models, experiments and their performance are still task-dependent, and may not generalise enough to be applied to other diseases. Second, I use the first answer match in the context concatenated from retrieved sentences to obtain the plausible answer position, which induces extra noises. In some cases, the module does extract the true answer string, but most sentences in the context are from the introduction and discussion sections, which may not be the most appropriate places. Last, in the sentence retrieval module, I use the combination of question and title to extract relevant sentences, while I believe there should be better retrieval strategies. For example, using only biomedical entities in the title to retrieve sentences would change the performance. Similar to the project of risk of bias assessment, all texts are converted from PDFs by Xpdf. The datasets do not contain PMIDs or PMCIDs of publications, which can be used to search the PubMed XML files to extract the method and result sections.

In future work which aims to extract PICO elements from preclinical publications using question answering methods, datasets with more precise annotations are necessary. The annotations can be the exact string position in the full texts, passages, or complete text snippets/sentences which can be located or retrieved easily. For cases where it is difficult to determine the most 'appropriate' position, separate annotation of relevant sentences can be provided. This can provide more flexible strategies for information retrieval and answer extraction modules. To develop more generalisable modules, datasets should also involve a wider range of preclinical studies, include more diverse disease models and mechanistic experiments.

3.6 Summary

In this chapter, I demonstrate three question answering models for identifying intervention and induction measures from clinical abstracts and preclinical full-text publications, including BiDAF, QANet and fine-tuning BERT with different pre-trained weights. For preclinical full texts, to obtain shorter passages, I also explore three information retrieval methods (TF-IDF, BM25 and biomedical BERT embeddings) to extract relevant sentences, and the best retrieval method (biomedical BERT embeddings) require 30 sentences to be extracted to contain the answer strings. All three question answering models achieve good performance for clinical abstracts focusing on MND disease (F1 around 80%), while fine-tuning biomedical BERT significantly outperforms BiDAF and QANet on preclinical publications focusing on psychosis disorder and chemotherapy-induced peripheral neuropathy, for both factoid and list type datasets. With some pre-processing and post-processing procedures, the question answering module can extract multiple answer candidates for one publication, although the prediction module tends to put string candidates containing the same keyword on top of the answer list rather than other potential intervention/induction methods. Datasets focusing on a wider range of disease or mechanisms, and more granular sentence annotations to locate answers are necessary to develop question answering models for PICO extraction.

Chapter 4 Extracting PICO Elements in Preclinical Text: Named Entity Recognition

In the previous chapter, I apply question answering models to extract interventions or methods of induction of disease model from clinical abstracts and preclinical full texts, which achieves satisfactory performance for records with only one answer candidate. The performance of records with multiple answer candidates is poor because of the difficulties of predicting the number of candidates and handling the span boundary of each candidate properly. In most cases, PICO elements are mentioned multiple times throughout an abstract or a full-text article, and it is difficult to determine the most appropriate or exact position of a candidate. In this scenario, the usability of question answering models is limited, and named entity recognition (Manning *et al.*, 2008) is more appropriate for PICO elements extraction. Named entity recognition seeks to classify each word in the text into pre-defined categories so adjacent words belonging to the same category can be extracted to formulate answer strings and repeated mentions are allowed. In this chapter, I annotate preclinical abstracts with PICO mentions and apply named entity recognition models to extract PICO elements.

4.1 Related Work

While extracting PICO elements from clinical reports is relatively well-explored, no method has been developed or evaluated for preclinical animal literature. Here I discuss related work of PICO extraction for clinical trials.

Most of the previous work casts PICO extraction as a sentence classification task, which aims to identify sentences containing at least one PICO concept. The training datasets are often small-size abstracts annotated by human reviewers, or large-size data derived from structured PubMed abstracts which contain explicit keywords in subheadings like ‘Patients’, ‘Interventions’ and

‘Outcomes’, so sentences that belong to the corresponding sections can be annotated automatically (Boudin *et al.*, 2010; Jin and Szolovits, 2018a; Jin and Szolovits, 2018b). Some early studies represent sentences by numeric vectors based on a series of text features, including sentence position, sentence length, number of words, digits, and UMLS (Unified Medical Language System) concepts (Bodenreider, 2004), etc. Classification models are then trained on the sentence representations to classify PICO sentences, including conditional random fields (Chabou and Iglewski, 2018), support vector machine, random forest and Naïve Bayes (Boudin *et al.*, 2010). Differently, Wallace et al use a collection of free-text PICO summaries from the Cochrane Database of Systematic Reviews (www.cochranelibrary.com/cdsr/about-cdsr) to automatically derive PICO annotations by checking if each sentence shares at least 4 common words with the summary descriptions of a review. Using the pseudo labels, they trained logistic regression models assisted by distant supervision for PICO sentence classification (Wallace *et al.*, 2016). More recent studies have used recent neural network models for PICO sentence classification such as the bidirectional LSTM (Jin and Szolovits, 2018a) with some variations (Jin and Szolovits, 2018b) and texts are mainly represented by word vectors, which requires less feature engineering.

More precise PICO phrases extraction is formulated as a named-entity recognition task, and LSTM or BERT models with a conditional random field layer are common approaches (Brockmeier *et al.*, 2019; Nye *et al.*, 2018). The training datasets involved contain fine-grained PICO span annotations (DeYoung *et al.*, 2020). In a less common approach, Zhang et al trains a graph learning model (Perozzi *et al.*, 2014) on the UMLS concepts to obtain graph representations for PICO entities, which achieves small marginal improvements for disease identification in sentences describing population and outcomes (Zhang *et al.*, 2020).

There are also some work aiming to predict the result of a clinical treatment given partial of PICO elements (Lehman *et al.*, 2019) and additional trial proposal (Jin *et al.*, 2020), or extract relations among PICO entities (Nye *et al.*, 2020), which are beyond the discussion here due to the limited format of preclinical datasets I can obtain.

4.2 Dataset

2,207,654 articles from PubMed Central Open Access Subset database (www.ncbi.nlm.nih.gov/pmc/tools/openftlist) published from 2010 to 2019 are downloaded, and a citation screening filter is used to identify in vivo research from title and abstract (developed by EPPI-Centre, UCL, (Liao *et al.*, 2018)). I choose an inclusion cut-point with a precision of 99% and obtain 50,653 abstracts describing in vivo animal experiments. I randomly select 400 abstracts for the annotation task and another 10,000 for the self-training experiments.

I use the online platform tagtog (https://www.tagtog.net) for PICO phrases annotation. In addition to Intervention, Comparator and Outcome, I divide the Population category into three sub-entities: the Species, the Strain, and the method of Induction of the disease model. After the initial annotation process and discussion with a senior clinician, I propose with some general rules for the annotation task:

- Only PICO spans describing in vivo experiments are annotated, i.e. interventions or treatments should be conducted within an entire, living organism. Interventions applied to tissues derived from an animal or in cell culture (ex vivo or in vitro experiments) should not be annotated;

- Texts describing the introduction, conclusion or objectives should not be annotated in most cases because these might relate to work other than that described in the publication. They should be annotated only when the remaining text lacks a clear description of the method, or where the text gives the meaning of abbreviations;

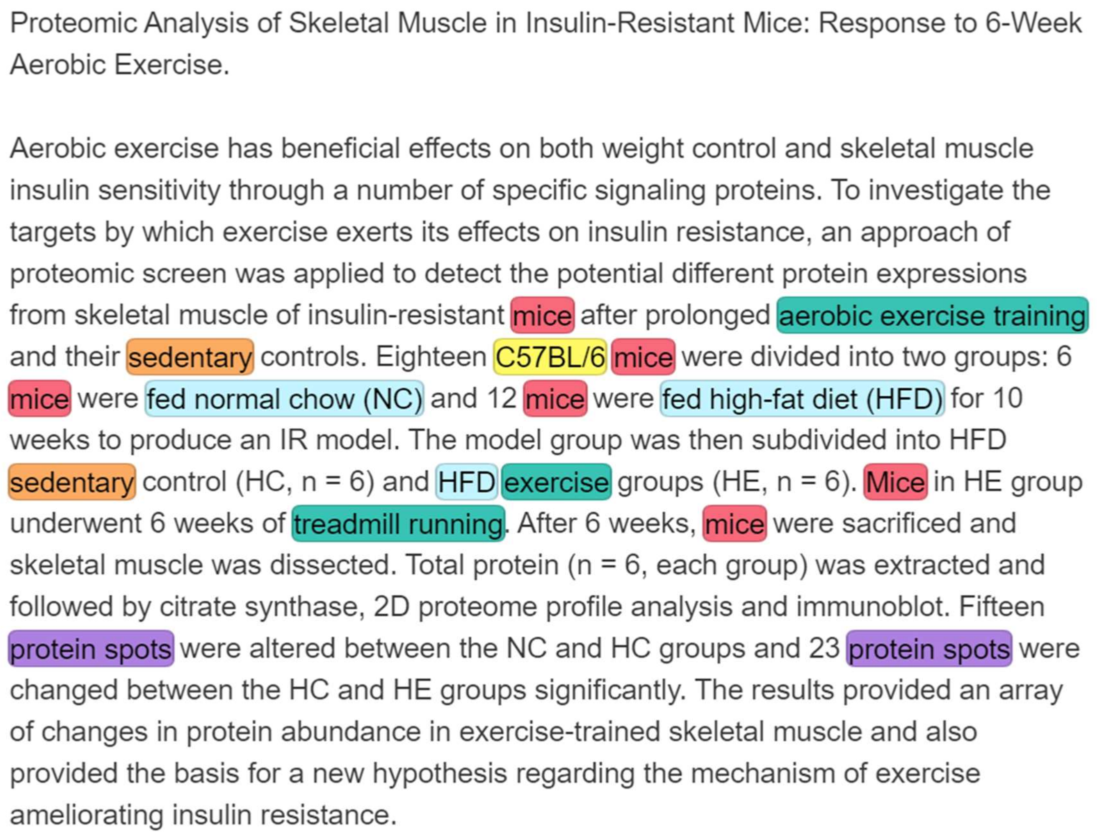
- The first occurrence of abbreviation should be annotated together with the parent text. For example, "vascular endothelial growth factor (VEGF)" should be tagged as one entity for its first occurrence; in the remainder of the text, "VEGF" or "vascular endothelial growth factor" could be annotated separately if they are not mentioned together;

- Uninformative phrases like "control" or "wild-type" should not be annotated;

- Any extra punctuations between phrases (such as commas) should not be annotated. However, if the entity appears only one time in the text, punctuations can be included in a long span of text which consists of several phrases;

- Entity spans cannot be overlapped. Annotations in tagtog are output in EntitiesTsv format which resembles the tsv output in the Stanford NER tool (Finkel *et al.*, 2005), and this does not support overlapped entities.

Figure 4.1 demonstrates an example of annotated abstract (PMC3541238) using tagtog. After excluding the title, introduction sentence, first part of the objective sentence and the conclusion sentence which do not explicitly describe experimental elements, PICO entities are extracted from the rest of the sentences: 1) Species: mice; 2) Strain: C57BL/6; 3) Induction: fed normal chow (NC), fed high-fat diet (HFD); 4) Intervention: aerobic exercise training, exercise, treadmill running; 5) Comparator: sedentary; and 6) Outcome: protein spots.


Proteomic Analysis of Skeletal Muscle in Insulin-Resistant Mice: Response to 6-Week Aerobic Exercise.

Aerobic exercise has beneficial effects on both weight control and skeletal muscle insulin sensitivity through a number of specific signaling proteins. To investigate the targets by which exercise exerts its effects on insulin resistance, an approach of proteomic screen was applied to detect the potential different protein expressions from skeletal muscle of insulin-resistant mice after prolonged aerobic exercise training and their sedentary controls. Eighteen C57BL/6 mice were divided into two groups: 6 mice were fed normal chow (NC) and 12 mice were fed high-fat diet (HFD) for 10 weeks to produce an IR model. The model group was then subdivided into HFD sedentary control (HC, n = 6) and HFD exercise groups (HE, n = 6). Mice in HE group underwent 6 weeks of treadmill running. After 6 weeks, mice were sacrificed and skeletal muscle was dissected. Total protein (n = 6, each group) was extracted and followed by citrate synthase, 2D proteome profile analysis and immunoblot. Fifteen protein spots were altered between the NC and HC groups and 23 protein spots were changed between the HC and HE groups significantly. The results provided an array of changes in protein abundance in exercise-trained skeletal muscle and also provided the basis for a new hypothesis regarding the mechanism of exercise ameliorating insulin resistance.

Figure 4.1: Example of PICO phrases annotation for a preclinical abstract. Screenshot from tagtog.

On average, there are 11 sentences in an abstract, and 5 of them describe PICO elements. Among 400 abstracts, 6,837 entities are annotated in total, where around 41% of entities belong to outcomes, 24% to interventions, 20% to species, 11% to the method of induction of disease, 3% to strain and less than 2% belong to comparators, as shown in Table 4.1. Almost all abstracts describe outcomes and species, 90% of documents describe interventions, 60% describe the method of induction of disease, 30% describe animal strains, and 16% describe comparators. Each abstract mentions around 7 entities for outcomes, 4 entities for interventions, 3 entities for species, 1 entity for the method of induction of disease and less than 1 entity for comparator and animal strain.

	Distribution of PICO entities across all abstracts	**Percentage of abstracts annotated with PICO entity**	**Avg number. of PICO entities in each abstract**
Intervention	24.1%	92.5%	4.1
Comparator	1.8%	16.8%	0.3
Outcome	40.6%	99.8%	6.9
Induction	10.6%	64.0%	1.8
Species	19.6%	98.3%	3.4
Strain	3.3%	30.5%	0.6
Total	100%	100%	17.1

Table 4.1: Statistics of PICO entities in the 400 annotated preclinical abstracts.

To check the ambiguity of PICO entities, I convert all the 6,873 PICO phrases into text embeddings and apply the t-distribution stochastic neighbouring embedding (t-SNE) algorithm (Van Der Maaten and Hinton, 2008) to project the high-dimensional embeddings into the 2-dimensional space. I use five embeddings for phrases conversion: biomedical word2vec (Pyysalo *et al.*, 2013) and BERT embeddings with four pre-trained modules including 1) BERT-Base, the original BERT trained on the combination of BookCorpus, and English Wikipedia (Devlin *et al.*, 2018); 2) BioBERT-Base, which trains BERT on the combination of BookCorpus, English Wikipedia, PubMed abstracts and PubMed Central full-text articles (Lee *et al.*, 2019); 3) PubMedBERT-Abs, which trains BERT on PubMed abstracts only, and 4) PubMedBERT-Abs-Full on a combination of PubMed abstracts and PubMed Central full-text articles (Gu *et al.*, 2020). For word2vec embeddings, each PICO phrase is split into words by scispaCy (Neumann *et al.*, 2019) and the phrase embedding is obtained by averaging its word embeddings (dimension 200); while for BERT embeddings, each PICO phrase is tokenised by WordPiece (Wu *et al.*, 2016), and the phrase embeddings are generated by averaging its token embeddings (dimension 768). The visualisation of PICO entities by t-SNE is displayed in Figure 4.2. The iteration step is 1000 and I compare the scatter charts using different perplexity from 2 to 50, and set the final perplexity as 5. For all five representation approaches, points of Species are clearly separated from points of other entities. The boundary between points of Intervention and Outcome is less clear, especially for embeddings from biomedical word2vec and BERT-Base. Points of Induction and Intervention are overlapped, which implies the ambiguity of these two PICO types and the difficulties of distinguishing those two elements in the following entity recognition task.

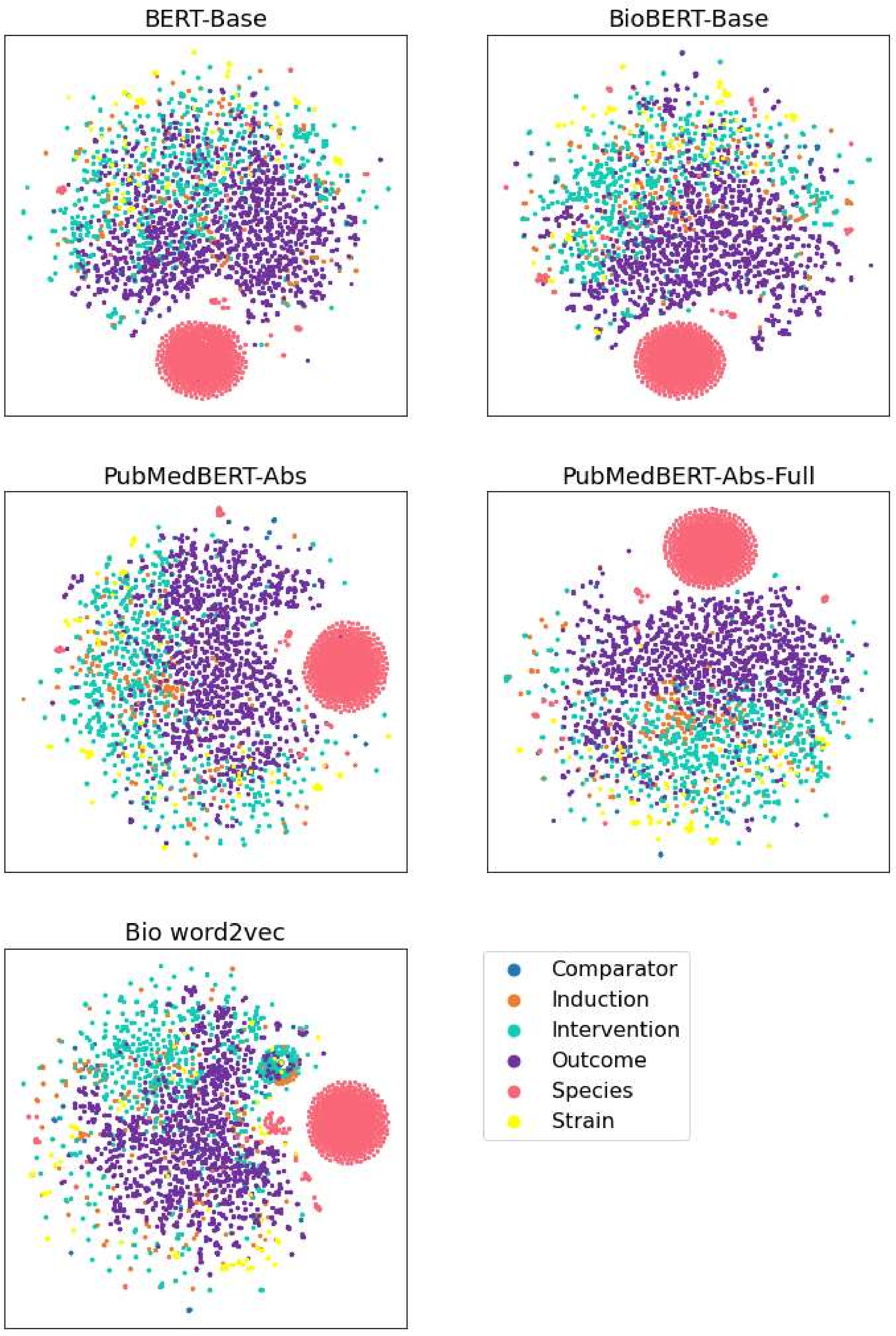


Figure 4.2: Visualisation of 6,837 PICO phrases represented by vectors using biomedical word2vec and BERT embeddings from four pre-trained modules. Axes refer to the two most important dimensions selected by the t-SNE algorithm.

4.3 Methods

From the last section, less than 50% of sentences in each abstract contain PICO phrases and using the entire abstracts to train an entity recognition model is inefficient. Therefore, I decompose the PICO phrases extraction task into two subtasks: 1) PICO sentence classification, and 2) PICO entity recognition, as the workflow demonstrated in Figure 4.3. The training materials of the first task are individual sentences from the annotated abstracts, where each sentence is labelled as yes/no if it contains or does not contain any PICO phrases. The training materials for the second task are the truncated abstracts consisting of only PICO sentences, where non-PICO sentences (sentences do not contain any PICO phrases) are removed automatically. For PICO entity prediction in the future application, sentences in an individual abstract are classified by the best PICO sentence classifier from the first task, and the non-PICO sentences are then removed automatically based on the sentence classification labels. The two tasks are trained independently, which guarantees the quality of training samples for the PICO entity recogniser and the efficiency of training programs.

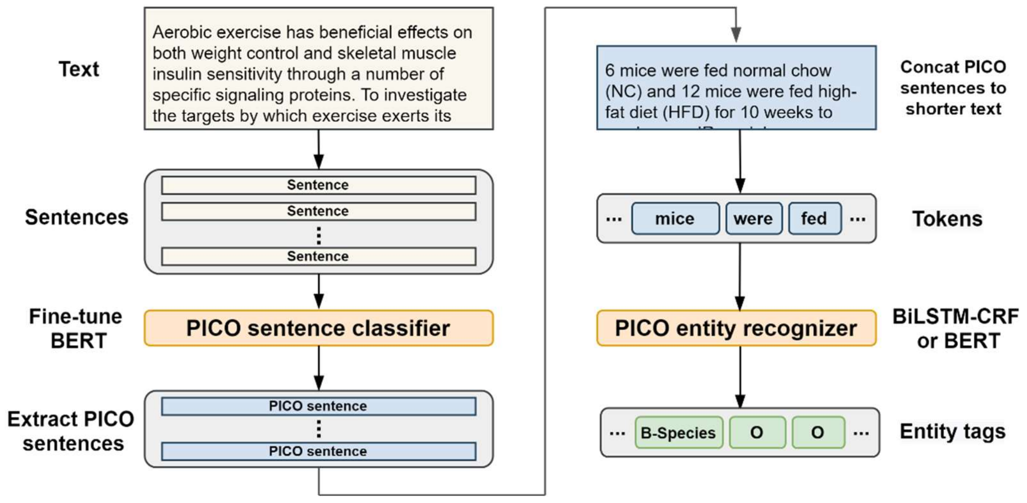

Text
Aerobic exercise has beneficial effects on both weight control and skeletal muscle insulin sensitivity through a number of specific signaling proteins. To investigate the targets by which exercise exerts its
Sentences
Sentence
Sentence
Sentence
Fine-tune BERT
PICO sentence classifier
Extract PICO sentences
PICO sentence
PICO sentence
6 mice were fed normal chow (NC) and 12 mice were fed high-fat diet (HFD) for 10 weeks to
Concat PICO sentences to shorter text
mice
were
fed
Tokens
PICO entity recognizer
BiLSTM-CRF or BERT
B-Species
O
O
Entity tags

Figure 4.3: The workflow of PICO extraction.

4.3.1 PICO Sentence Classification

Text from 400 abstracts are split into 4,247 sentences by scispaCy (Neumann *et al.*, 2019) and sentences containing at least one PICO entity are labelled as ‘yes’ for PICO sentence. Sentences are shuffled and split into training, validation and test set (80%/10%/10%). For the sentence-level classification task, I fine-tune BERT (Vaswani *et al.*, 2017) with four pre-trained weights (BERT-Base, BioBERT-Base, PubMedBERT-Abs and PubMedBERT-Abs-Full). The validation metrics including F1, recall, precision and specificity are same as what has been described in Chapter 2 (see Section 2.3.5).

4.3.2 PICO Entity Recognition

Identifying specific PICO phrases from the remaining texts in each abstract is a standard named entity recognition (NER) task, and NER models aim to assign an entity tag for each word/token. As a PICO phrase may consist of several tokens, labelling all those tokens with the same entity tag is ambiguous. For example, if the tokens of an intervention phrase are [‘aerobic’, ‘exercise’, ‘training’] and they are tagged as [‘Intervention’, ‘Intervention’, ‘Intervention’], we cannot tell if these tokens are three different interventions or they are part of one intervention. To solve this, the BIO (Beginning-Inside-Outside) tagging format (Ramshaw and Marcus, 1995) is proposed where the beginning token of an entity phrase is labelled as ‘B-XX’, and the remaining tokens of the phrase are labelled as ‘I-XX’ (‘XX’ refers to one unique entity type). All other tokens which do not belong to any entities are tagged as ‘O’. In the example described above, the entity tags in BIO format are [‘B-Intervention’, ‘I-Intervention’, ‘I-Intervention’], which clearly indicates the three tokens belong to one PICO phrase. This tagging format generates 13 unique tags for 6 PICO entities (two tags for each entity type, plus tag ‘O’).

The following sections describe PICO entity recognition models including conditional random field (CRF) (Sutton and McCallum, 2011), the bidirectional LSTM with a CRF layer on top (BiLSTM-CRF) (Lample *et al.*, 2016), which is a classic NER model, fine-tuning BERT model with CRF/LSTM layer on top, and self-training strategy (van Engelen and Hoos, 2020) which aims to solve the data-hungry issue.

4.3.2.1 CRF, BiLSTM and BiLSTM-CRF

Conditional random field (CRF) is a discriminative probabilistic model (Sutton and McCallum, 2011) which aims to find the optimal path (predicted tag sequence) achieving the maximum joint probabilities and learn the transition constraints among predicted tags. One example of the transition constraint is: if the tag of a word in the sequence is 'I-Outcome', the tag of the previous word can only be 'B-Outcome' or 'I-Outcome', and impossible to be 'I-Intervention', 'O' or other tags. CRF is supposed to reduce the transition errors among tags and has been proved beneficial when it is added on top of other layers (Lample *et al.*, 2016).

Formally, in the scenario of a linear-chain CRF applied in named entity recognition, given a text sequence with T tokens $\vec{x} = \{x_1,\ x_2, \dots, x_T\}$, the goal is to assign an entity tag for each word token, which forms the corresponding tag (state) sequence $\vec{y} = \{y_1,\ y_2, \dots, y_T\}$. We need to build a model to estimate the conditional probability

$$p(\vec{y}|\vec{x};\vec{w}) = p(y_1, \dots, y_T|x_1, \dots, x_T;\vec{w}) \tag{4.1}$$

where $\vec{w} \in \mathbb{R}^d$ refer to the parameters learned from the training process. According to the chain rule of conditional probabilities and the Markov assumption (Collins, 2011) - the tag of the token at position t only depends on the tag of its previous token at position t-1, the conditional probability can be written as

$$p(\vec{y}|\vec{x};\vec{w}) = p(y_1, \dots, y_T|x_1, \dots, x_T;\vec{w})$$

$$= p(y_1|\vec{x};\vec{w})p(y_2|y_1,\vec{x};\vec{w}) \dots p(y_T|y_1, \dots ,\ y_{T-1},\vec{x};\vec{w})$$

$$= \prod_{t=1}^{T} p(y_t|y_1, \dots, y_{t-1}, \vec{x}\ ;\vec{w})$$

$$= \prod_{t=1}^{T} p(y_t|y_{t-1}, \vec{x}\ ;\vec{w}) \tag{4.2}$$

The term $p(y_t|y_{t-1}, \vec{x}\,; \vec{w})$ can be modelled by a log-linear model as

$$p(y_t|y_{t-1}, \vec{x}\,; \vec{w}) = \frac{\exp\left[\vec{w} \bullet \phi(\vec{x}, t, y_{t-1}, y_t)\right]}{\sum_{y'_t \epsilon \Upsilon} \exp\left[\vec{w} \bullet \phi\left(\vec{x}, t, y_{t-1}, y'_t\right)\right]} \tag{4.3}$$

where ϕ is defined as a feature function constructed by all tokens in the text sequence $(x_1, \ldots, x_T)$, the position of the token being tagged (t), the previous tag (y_{t-1}) and the new tag (y_t); y'_t is the tag of $t\text{-}th$ token and can take any value in the set Υ of all possible entity tags. Therefore,

$$p(\vec{y}|\vec{x}; \vec{w}) = \prod_{t=1}^{T} \frac{exp\left[\vec{w} \bullet \phi(\vec{x}, t, y_{t-1}, y_t)\right]}{\sum_{y'_t \epsilon \Upsilon} exp\left[\vec{w} \bullet \phi\left(\vec{x}, t, y_{t-1}, y'_t\right)\right]}$$

$$= \frac{\exp\left[\sum_{t=1}^{T} \vec{w} \bullet \phi(\vec{x}, t, y_{t-1}, y_t)\right]}{\prod_{t=1}^{T} \sum_{y'_t \epsilon \Upsilon} exp\left[\vec{w} \bullet \phi\left(\vec{x}, t, y_{t-1}, y'_t\right)\right]} \tag{4.4}$$

We need to find the most likely tag sequence for the given text sequence:

$$\underset{\vec{y}}{argmax}\; p(\vec{y}|\vec{x}; \vec{w}) = \underset{y_1,\ldots,y_T}{argmax} \frac{exp\left[\sum_{t=1}^{T} \vec{w} \bullet \phi(\vec{x}, t, y_{t-1}, y_t)\right]}{\prod_{t=1}^{T} \sum_{y'_t \epsilon \Upsilon} exp\left[\vec{w} \bullet \phi\left(\vec{x}, t, y_{t-1}, y'_t\right)\right]}$$

$$= \underset{y_1,\ldots,y_T}{argmax}\; exp\left[\sum_{t=1}^{T} \vec{w} \bullet \phi(\vec{x}, t, y_{t-1}, y_t)\right]$$

$$= \underset{y_1,\ldots,y_T}{argmax} \sum_{t=1}^{T} \vec{w} \bullet \phi(\vec{x}, t, y_{t-1}, y_t) \tag{4.5}$$

The most straightforward solution is to traverse all possible paths to find the optimal tag combination which has the maximum joint conditional probability. However, if the total number of entity types is K ($K = 13$ for my dataset) and the sequence length is T, there are K^T possible tag combinations, which is inefficient and memory-consuming especially when T is large. Viterbi algorithm is applied to find the optimal path recursively (Forney, 1973), which reduces the computing complexity from $O(K^T)$ to $O(TK^2)$. Specifically, δ_{t,y_t} is defined as the maximum probability of any tag sequence which ends with tag y_t at position t. Then for sequence with T words,

$$\delta_{1,y_1} = \vec{w} \bullet \phi(\vec{x}, 1, y_0, y_1), \qquad y_1 \in \Upsilon \tag{4.6}$$

$$\delta_{t,y_t} = \max_{y_{t-1}\in\Upsilon} \left[\delta_{t-1,y_{t-1}} + \vec{w} \bullet \phi(\vec{x}, t, y_{t-1}, y_t)\right], t = 2, \ldots, T, y_t \in \Upsilon \tag{4.7}$$

By calculating y_t forward from the first word ($t = 1$) to the final word ($t = T$), the highest probability can be obtained by

$$\max_{y_1,\ldots,y_T} p(\vec{y}|\vec{x}; \vec{w}) = \max_{y_1,\ldots,y_T} \sum_{t=1}^{T} \vec{w} \bullet \phi(\vec{x}, t, y_{t-1}, y_t) = \max(\delta_{T,y_T}) \tag{4.8}$$

and the optimal tag sequence can be obtained by backpointers (Collins, 2011). The term $\vec{w} \bullet \phi(\vec{x}, t, y_{t-1}, y_t)$ can be decomposed as

$$\vec{w} \bullet \phi(\vec{x}, t, y_{t-1}, y_t) = a(y_t, y_{t-1}, \vec{x}, t\,) + b(y_t, \vec{x}, t) \tag{4.9}$$

where $a(y_t, y_{t-1}, \vec{x}, t\,)$ denotes the transition score of the tag at step t switched from the previous step t-1, and $b(y_t, \vec{x}, t)$ represents the emission score of the tag at step t given the text sequence. In the actual training, parameters including the transition matrix $A \in \mathbb{R}^{K\times K}$ between every unique pair of entity types are updated to minimise the negative log-likelihood loss function

$$L(\vec{w}) = -\frac{1}{n}\left[\sum_{i=1}^{n} \log p\left(\vec{y^i}\middle|\vec{x^i}; \vec{w}\right) - \frac{\lambda\|\vec{w}\|^2}{2}\right] \tag{4.10}$$

where $\lambda\|\vec{w}\|^2/2$ is the regularisation term to penalise large parameter values.

The architecture of the CRF model for PICO entity recognition is shown in Figure 4.4. Each word in the text sequence $\vec{x} = \{x_1,\ x_2, \ldots, x_T\}$ is first mapped into embeddings $e_{t=\{1,\ldots,T\}} \in \mathbb{R}^d$ by the pre-trained biomedical word vectors (Pyysalo *et al.*, 2013), where d is the dimension of word vectors. A dropout layer is then applied for regularisation, and a linear layer is used to convert the dimension of word embedding from d to K (the output of the linear layer corresponds to the emission scores $B \in \mathbb{R}^{T\times K}$). Finally, the CRF layer is applied to learn the transition matrix among each unique pair of tags, and update the probability of the $t\text{-}th$ word belonging to the $k\text{-}th$ entity, $p_{t,k}$, by summarising the emission score and the transition score, where $t = 1, \ldots, T$ and $k = 1, \ldots, K$. The entity with the highest probability is the prediction tag of the token. The architecture of the BiLSTM-CRF model is demonstrated in Figure 4.5. The only difference between the BiLSTM-CRF and CRF model is the bidirectional LSTM layer before the linear layer, which generates a new hidden representation for each token (dimension switched from d to $2h$, where h is the hidden dimension in LSTM). I do not show the architecture of BiLSTM model separately because the main components are same to that of BiLSTM-CRF, but without the CRF layer on top.

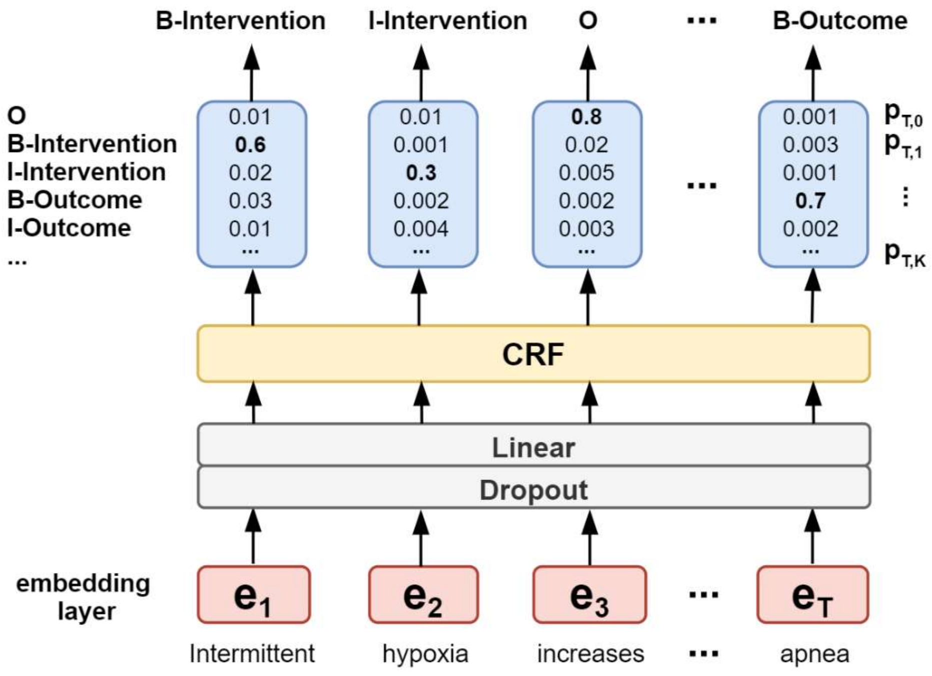


Figure 4.4: The architecture of CRF model for PICO entity recognition.

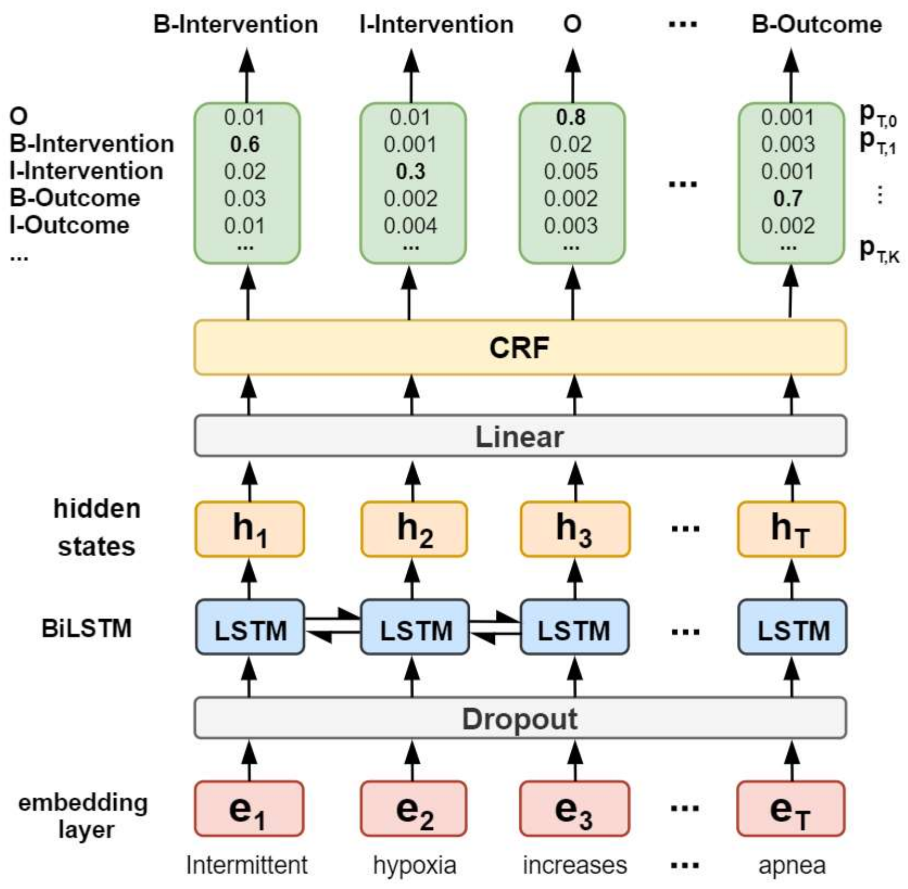


Figure 4.5: The architecture of BiLSTM-CRF model for PICO entity recognition.

4.3.2.2 BERT Models

Similar to the PICO sentence classification, I fine-tune BERT with different pre-trained weights for the entity recognition task, using BertForTokenClassification module from Hugging Face Transformers library (Wolf *et al.*, 2019). I also explore the effect of adding CRF and LSTM layers on top of BERT, by the BERT-CRF and BERT-LSTM-CRF models. The architecture of the BERT-LSTM-CRF model for PICO entity recognition is displayed in Figure 4.6, and the BERT-CRF model just removes the LSTM layer. There are some extra pre- and post- processing steps for BERT models because of the WordPiece (Wu *et al.*, 2016) tokeniser. Although the input sequences are individual words pre-tokenised by sciSpacy (Neumann *et al.*, 2019), BERT's tokeniser further splits each word into sub-tokens which do not have any tags assigned. To solve this, at the training stage, each sub-token is assigned the tag of the word they come from (same as the tag of the first sub-token of the word). However, this increases the number of entity instances which makes the evaluation results not comparable to the previous CRF or LSTM models. For example, an intervention phrase 'aerobic treadmill training' is tokenised as ['aero', '##bic', 'tread', '##mill', 'training'] and the corresponding tags are ['B-Intervention', 'B-Intervention', 'I-Intervention', 'I-Intervention', 'I-Intervention'], which is misleading because the tag sequence indicates two separate interventions and the performance may be ‘improved’ due to the increasing number of instances. To make the evaluations comparable, I record the word ids (sub-tokens from the same word share the same word id), and at the evaluation stage, the tags of sub-tokens with the same word ids are merged as one entity tag so the number of instances is same as that of previous models.

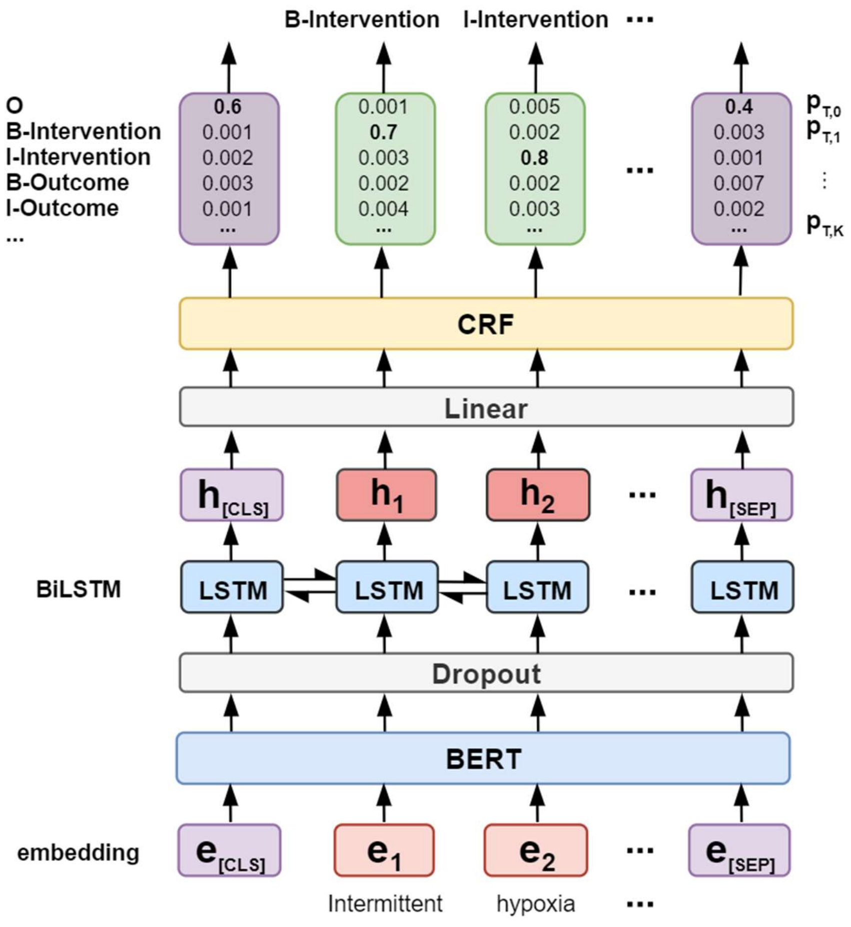


Figure 4.6: The architecture of BERT-LSTM-CRF model for PICO entity recognition.

4.3.2.3 Self-Training

One limitation of this project is the small amount of training data so I explore a semi-supervised learning strategy, self-training, which utilises the unlabelled corpus to generate pseudo labels for training (Ruder and Plank, 2018). I use 400 annotated abstracts as gold data, and 10,000 unlabelled abstracts randomly selected from 50,653 in vivo animal records as silver data. Non-PICO sentences are removed from the unlabelled abstracts by the best PICO sentence classification model, and the truncated abstracts are used for self-training. As Figure 4.7 shows, I first use the fine-tuned PICO entity recogniser from the gold set (80% of 400 labelled records for training, 10% for validation) to predict the tag of each token in the silver set. For each abstract in the silver set, I calculate the average prediction probabilities of all tokens within that abstract. Silver records with average probabilities larger than a threshold are then combined with the original gold training/validation set; the enlarged new dataset is used to fine-tune a newly initialised PICO entity recogniser. Then I repeat the prediction, pseudo data generation, data selection and supervised fine-tuning procedures, until no more unlabelled records with average prediction probabilities larger than the threshold can be added. In every data enlarging step, newly included silver records are split into the training set (80%) and validation set (20%), then combined with the gold training records (320 records) and gold validation records (40 records) separately. This guarantees the gold validation set is only ever used for validation. The original gold test set is only used for final evaluation.

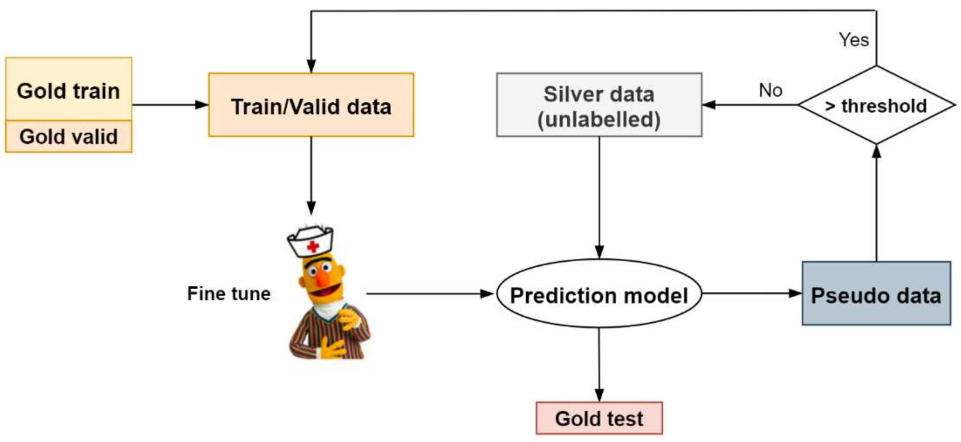


Figure 4.7: The workflow of self-training experiments.

4.3.2.4 Evaluation Metrics

For evaluation, the true tags and predicted tags of all records construct separate lists and the entity-level metrics (Hiroki Nakayama, 2018) including precision, recall and F1 are calculated for each of the 6 PICO entity types by:

$$\text{Precision} = \frac{\text{number of correctly predicted entities}}{\text{number of predicted entities}} \tag{4.11}$$

$$\text{Recall} = \frac{\text{number of correctly predicted entities}}{\text{number of true entities}} \tag{4.12}$$

$$\text{F1} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.13}$$

The calculation for an example sample containing two records is demonstrated in Table 4.2. After calculation for each entity types, the overall metrics are micro-average scores across all entity types, which are (1+1) / (1+2) = 2/3 of precision, (1+1) / (1+1) = 1 for recall, and 0.8 for F1 in the example. The micro-average F1 of validation samples is used for model and parameters selection.

	Record 1		Record 2
True tags	[‘O’, ‘O’, ‘O’, ‘B-Intervention’, ‘I-Intervention’, ‘O’]		[‘O’, ‘O’, ‘B-Outcome’, ‘I-Outcome’]
Predicted tags	[‘O’, ‘B-Intervention’, ‘I-Intervention’, ‘B-Intervention’, ‘I-Intervention’, ‘O’]		[‘O’, ‘O’, ‘B-Outcome’, ‘I-Outcome’]
No. true entities	1 for Intervention, 0 for Outcome		0 for Intervention, 1 for Outcome
No. predicted entities	2 for Intervention, 0 for Outcome		0 for Intervention, 1 for Outcome
No. correctly predicted entities	1 for Intervention, 0 for Outcome		0 for Intervention, 1 for Outcome
	Precision	**Recall**	**F1**
Intervention	1/2	1/1	0.67
Outcome	1/1	1/1	1

Table 4.2: The calculation of entity-level metrics for an example sample containing two records.

4.4 Experiments

Annotation project is available at www.tagtog.net/qwang/pre-pico/pool; processed datasets are available in the Preclinical PICO extraction repository (osf.io/2dqcg); and codes of experiments are available at github.com/qianyingw/pre-pico.

4.4.1 Experiments of PICO Sentence Classification

For PICO sentence classification, I fine-tune BERT with different pre-trained modules as described earlier. Parameters are trained to minimise the cross-entropy loss using the AdamW algorithm (Loshchilov and Hutter, 2017). I use the slanted triangular learning rate scheduler (Howard and Ruder, 2018) with a threshold learning rate 5e-5. I apply gradient clipping (Zhang et al., 2019) with a threshold norm 0.1 to rescale gradients and use gradient accumulation every 4 mini-batches to reduce memory consumption. Batch size is set to 16. All models are trained for 10 epochs and scores from the epoch with the minimum loss on the validation set are recorded. From Table 4.3, the performance of PICO sentence classifier does not show much difference among different pre-trained modules in BERT, with F1 scores all over 80%. BERT pre-trained on the biomedical corpus (BioBERT and two PubMedBERT) can identify more PICO sentences, as recalls are 2%~4% higher than that of the general-domain BERT. BERT pre-trained on PubMed abstracts achieves the highest validation F1 of 89%, which is selected to detect PICO sentences from abstracts for the prediction module.

	Valid				**Test**			
Pre-trained modules	**F**	**R**	**P**	**S**	**F**	**R**	**P**	**S**
BERT-Base	86.6	87.7	87.2	91.0	80.6	81.4	82.1	86.8
BioBERT-Base	87.7	89.6	88.1	92.4	84.3	81.0	90.0	92.6
PubMedBERT-Abs	89.3	91.3	89.1	91.3	85.4	88.4	85.0	88.5
PubMedBERT-Abs-Full	85.8	89.3	84.6	88.5	84.2	87.1	83.8	87.7

Table 4.3: Performance of PICO sentence classification by fine-tuning BERT with different pre-trained modules on the validation and test set. ‘F’, ‘R’, ‘P’, ‘S’ represents F1, recall, precision and specificity separately.

4.4.2 Experiments of PICO Entity Recognition

4.4.2.1 Experiments of CRF, BiLSTM and BiLSTM-CRF

For all entity recognition models, batch size is set to 16. I compare Adam and AdamW optimizers with the constant or slanted triangular learning rate scheduler and explore the different threshold learning rates. I also compare the effect of freezing word embeddings or not. Models are trained for 20 epochs. The optimal settings for CRF, BiLSTM and BiLSTM-CRF are using Adam optimizer with a constant learning rate (1e-2, 1e-3 and 5e-3 for CRF, BiLSTM and BiLSTM-CRF respectively). Freezing word embeddings achieves better performance for CRF and BiLSTM-CRF, while for BiLSTM, updating word embeddings jointly with the entity recognition model yields better results, see Appendix Table 12. The best overall validation performance and the corresponding test performance are shown in Table 4.4. The results of CRF and BiLSTM are competitive, where the validation F1 of CRF is 4% higher than that of BiLSTM, but test F1 is 2% lower. The CRF layer added to the BiLSTM model enhances the performance, as the overall F1 score is increased by 14% on the test set.

Model	**Valid**			**Test**		
	F	**R**	**P**	**F**	**R**	**P**
CRF	46.2	40.7	53.4	41.9	33.7	55.3
BiLSTM	41.7	44.2	39.5	43.5	38.1	50.6
BiLSTM-CRF	**58.8**	56.9	61.0	**57.9**	54.7	61.6

Table 4.4: Overall performance of CRF, BiLSTM and BiLSTM-CRF for PICO entity recognition on validation and test set. ‘F’, ‘R’, ‘P’ represents F1, recall and precision separately.

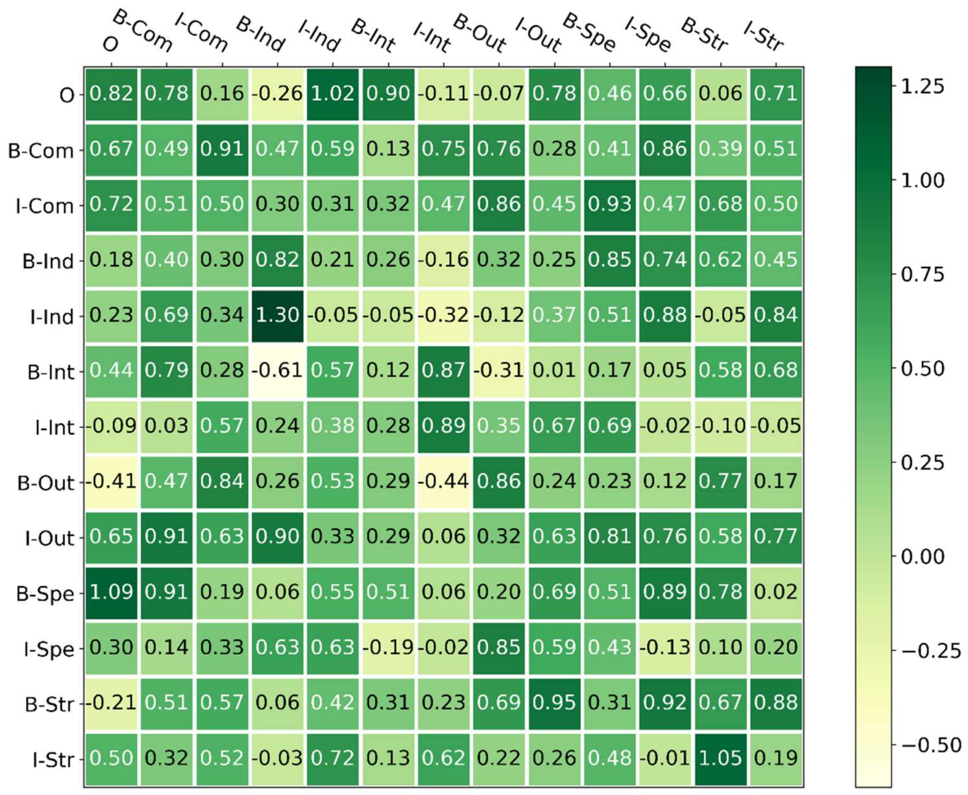
The entity-level performance of three models is demonstrated in Table 4.5. All three models achieve good performance for Species and Strain, with validation F1 around 95% and 80% respectively, because these two entities are limited to certain types in preclinical animal studies (like mice, fish for species; Wistar, Sprague Dawley for strain), so their identification is not complicated. The performance does not differ greatly among three models for Species (less than 2% difference of validation F1), while BiLSTM-CRF has F1 10% higher than that of BiLSTM for Strain. For Intervention, Outcome and Induction, the performance is less satisfactory, with F1 around 45%, 52% and 35% respectively, which is remained to be improved. The advantages of adding the CRF layer on top of the BiLSTM model are more obvious for Intervention and Outcome, as F1 improved by 30% and 16% respectively. The test F1 is around 20%~30% for Comparator, but the scores are zeros on the validation set. The poor and inconsistent performance may be caused by the lack of Comparator instances in the training corpus, and unclear boundary of the definition of comparator and intervention in some experiments which includes multiple pairs of comparator and intervention.

Entity	Model	Valid			Test		
		F	R	P	F	R	P
Induction	**CRF**	14.6	10.2	26.1	7.2	31.6	25.7
	BiLSTM	16.9	20.3	14.5	**20.0**	16.9	24.6
	BiLSTM-CRF	**35.2**	32.2	38.8	**20.0**	14.5	32.4
Species	**CRF**	**96.4**	100.0	93.1	**97.0**	95.5	95.7
	BiLSTM	94.7	99.1	90.7	95.9	97.0	94.8
	BiLSTM-CRF	96.0	100.0	92.3	96.6	97.0	96.2
Strain	**CRF**	80.0	96.0	68.6	72.2	92.9	86.7
	BiLSTM	69.0	80.0	60.6	75.0	66.7	85.7
	BiLSTM-CRF	**81.4**	96.0	70.6	**81.3**	72.2	92.9
Intervention	**CRF**	32.7	27.0	41.2	11.6	18.8	12.5
	BiLSTM	17.0	15.5	18.9	7.8	5.1	16.7
	BiLSTM-CRF	**44.7**	42.6	47.0	**45.0**	52.2	39.6
Comparator	**CRF**	0.0	0.0	0.0	20.0	100.0	44.4
	BiLSTM	0.0	0.0	0.0	25.0	15.0	75.0
	BiLSTM-CRF	0.0	0.0	0.0	**33.3**	20.0	100.0
Outcome	**CRF**	28.4	24.4	34.1	20.2	37.3	33.3
	BiLSTM	35.8	39.7	32.5	33.9	32.4	35.4
	BiLSTM-CRF	**52.2**	48.7	56.2	**54.2**	48.9	61.0

Table 4.5: Entity-level performance of CRF, BiLSTM and BiLSTM-CRF for PICO entity recognition on validation and test set. ‘F’, ‘R’, ‘P’ represents F1, recall and precision separately.

Figure 4.8 demonstrates an example of the transition matrix of PICO entity tags trained from the best BiLSTM-CRF model, where each element refers to the transition score between each pair of tags. Note the high transition score does not mean the final sequential tags of the text sequence will follow this transition because the final probabilities of the tag sequence consist of the transition scores and emission scores. However, it still reflects some patterns. For example, the induction method, comparator, and intervention are likely to be mentioned before the species: B-Induction → B-Species (score 0.85), I-

Comparator → B-Species (score 0.93), I-Intervention → B-Species (score 0.69). Species often consist of one word: B-Species → O (score 1.09); and interventions consist of several words: B-Intervention → I-Intervention (score 0.87), I-Intervention → I-Intervention (score 0.89).



	O	B-Com	I-Com	B-Ind	I-Ind	B-Int	I-Int	B-Out	I-Out	B-Spe	I-Spe	B-Str	I-Str
O	0.82	0.78	0.16	-0.26	1.02	0.90	-0.11	-0.07	0.78	0.46	0.66	0.06	0.71
B-Com	0.67	0.49	0.91	0.47	0.59	0.13	0.75	0.76	0.28	0.41	0.86	0.39	0.51
I-Com	0.72	0.51	0.50	0.30	0.31	0.32	0.47	0.86	0.45	0.93	0.47	0.68	0.50
B-Ind	0.18	0.40	0.30	0.82	0.21	0.26	-0.16	0.32	0.25	0.85	0.74	0.62	0.45
I-Ind	0.23	0.69	0.34	1.30	-0.05	-0.05	-0.32	-0.12	0.37	0.51	0.88	-0.05	0.84
B-Int	0.44	0.79	0.28	-0.61	0.57	0.12	0.87	-0.31	0.01	0.17	0.05	0.58	0.68
I-Int	-0.09	0.03	0.57	0.24	0.38	0.28	0.89	0.35	0.67	0.69	-0.02	-0.10	-0.05
B-Out	-0.41	0.47	0.84	0.26	0.53	0.29	-0.44	0.86	0.24	0.23	0.12	0.77	0.17
I-Out	0.65	0.91	0.63	0.90	0.33	0.29	0.06	0.32	0.63	0.81	0.76	0.58	0.77
B-Spe	1.09	0.91	0.19	0.06	0.55	0.51	0.06	0.20	0.69	0.51	0.89	0.78	0.02
I-Spe	0.30	0.14	0.33	0.63	0.63	-0.19	-0.02	0.85	0.59	0.43	-0.13	0.10	0.20
B-Str	-0.21	0.51	0.57	0.06	0.42	0.31	0.23	0.69	0.95	0.31	0.92	0.67	0.88
I-Str	0.50	0.32	0.52	-0.03	0.72	0.13	0.62	0.22	0.26	0.48	-0.01	1.05	0.19

Figure 4.8: Transition matrix of PICO entity tags from the best BiLSTM-CRF model. Rows refer to the tag of the previous token, and columns refer to the tag of the current token.

For BiLSTM models with or without the CRF layer on top, I also explore the effect of hidden dimension in LSTM modules. The optimal dimension for BiLSTM is around 64 to 256, while the impact of dimension is negligible for BiLSTM-CRF, as shown in Table 4.6.

	Hidden dim	F1	Recall	Precision
BiLSTM	32	34.8	32.8	37.2
	64	**41.7**	43.5	**40.0**
	128	40.8	42.1	39.6
	256	**41.7**	**44.2**	39.5
	512	37.3	38.7	36.1
BiLSTM-CRF	32	57.0	56.7	57.3
	64	58.5	**57.2**	59.8
	128	57.7	53.7	**62.3**
	256	**58.8**	56.9	61.0
	512	57.1	55.3	59.1

Table 4.6: Validation performance of BiLSTM models with different hidden dimension.

4.4.2.2 Experiments of BERT models

For BERT-based models, I fine-tune BERT for 30 epochs, BERT-CRF for 40 epochs and BERT-LSTM-CRF for 60 epochs. I use the slanted learning rate scheduler with the threshold learning rate 1e-4 but remove warm-up steps. The hidden dimension in BERT-LSTM-CRF is set to 64. Other settings are similar to that of the PICO sentence classification task. These are determined by checking overfitting or convergence issues from their learning curves.

The overall performance of three BERT-based models with different pre-trained weights is shown in Table 4.7. Compared with the previous best model BiLSTM-CRF, BERT and BERT-CRF pre-trained from the general-domain knowledge do not outperform BiLSTM-CRF, as the validation F1 are 3% and 1% lower respectively, while BERT-LSTM-CRF slightly improves F1 by 2%. Other BERT models pre-trained from the biomedical-domain knowledge all achieve better performance than BiLSTM-CRF, with F1 improved by 5% to 9%. Within three biomedical pre-trained weights, models with domain-specific pre-trained modules (two PubMedBERT) yield better performance than models with the mixed-domain pre-trained module (BioBERT). The advantage of the

CRF layer is trivial compared to the benefit of the large-scale pre-trained domain knowledge as there is not much difference among the performance of BERT, BERT-CRF and BERT-LSTM-CRF.

Model	Pre-trained weights	Valid			Test		
		F	R	P	F	R	P
BERT	BERT-Base	56.0	62.7	50.6	61.3	66.3	57.1
	BioBERT-Base	64.2	69.8	59.4	65.4	69.8	61.5
	PubMed-Abs	65.0	70.5	60.2	**70.1**	73.2	67.3
	PubMed-Abs-Full	**68.1**	73.0	63.8	69.9	73.4	66.7
BERT-CRF	BERT-Base	57.7	62.9	53.3	62.1	67.2	57.8
	BioBERT-Base	65.1	70.0	60.9	66.5	70.1	63.3
	PubMed-Abs	65.5	70.9	60.9	**68.0**	71.5	64.9
	PubMed-Abs-Full	**68.0**	72.8	63.7	67.5	70.9	64.5
BERT -BiLSTM -CRF	BERT-Base	60.8	66.4	56.1	64.6	69.5	60.3
	BioBERT-Base	66.0	70.0	62.5	68.3	71.2	65.6
	PubMed-Abs	**68.1**	73.3	63.5	67.2	70.8	64.0
	PubMed-Abs-Full	68.0	72.8	63.8	**68.5**	72.6	64.8

Table 4.7: Overall performance of BERT, BERT-CRF and BERT-LSTM-CRF with different pre-trained weights on validation and test set. ‘F’, ‘R’, ‘P’ represents F1, recall and precision separately.

The entity-level performance of three BERT models is demonstrated in Table 4.8. The performance of Species is as good as previous CRF/LSTM based models, with F1 scores over 95%. The highest validation F1 of Strain is increased by 4% but the test F1 is reduced by between 4% and 18%. The performance of Intervention, Outcome and Induction is improved a lot, as the validation F1 of the best BERT-based model is increased by 22%, 12% and 12% respectively, compared to the best LSTM/CRF based model. For Comparator, BERT-CRF has robust but still poor performance, with F1 around 30%. Considering the consistency of performance between the validation and test set, the best model is BERT-CRF for Comparator and Outcome, BERT-

LSTM-CRF for Induction and Species, BiLSTM-CRF for Strain, and BERT for Intervention.

Entity	Model	Valid			Test		
		F	R	P	F	R	P
	BERT	46.2	50.9	42.3	49.1	50.6	47.7
Induction	**BERT-CRF**	45.7	49.2	42.7	41.2	41.0	41.5
	BERT-LSTM-CRF	**47.5**	55.9	41.3	42.2	41.0	43.6
	BERT	96.4	99.1	93.9	98.1	100.0	96.4
Species	**BERT-CRF**	96.4	100.0	93.1	98.1	100.0	96.4
	BERT-LSTM-CRF	**97.3**	99.1	95.5	98.1	100.0	96.4
	BERT	80.0	80.0	80.0	63.4	72.2	56.5
Strain	**BERT-CRF**	75.5	80.0	71.4	65.0	72.2	59.1
	BERT-LSTM-CRF	**85.2**	92.0	79.3	**68.3**	77.8	60.9
	BERT	**67.3**	69.6	65.2	70.2	76.1	65.2
Intervention	**BERT-CRF**	63.9	66.9	61.1	69.6	75.4	64.6
	BERT-LSTM-CRF	63.2	65.5	61.0	**71.1**	78.3	65.1
	BERT	**33.3**	66.7	22.2	16.0	10.0	40.0
Comparator	**BERT-CRF**	30.8	66.7	20.0	**43.8**	35.0	58.3
	BERT-LSTM-CRF	0.0	0.0	0.0	0.0	0.0	0.0
	BERT	61.5	68.0	56.2	**65.4**	70.6	60.9
Outcome	**BERT-CRF**	**63.7**	69.2	58.9	61.1	66.0	56.9
	BERT-LSTM-CRF	63.3	69.7	58.0	60.5	66.4	55.6

Table 4.8: Entity-level performance of BERT, BERT-CRF and BERT-LSTM-CRF with the optimal pre-trained weight on validation and test set. ‘F’, ‘R’, ‘P’ represents F1, recall and precision separately.

4.4.2.3 Experiments of Self-Training

In self-training experiments, the best PICO sentence classifier (BERT pre-trained on PubMed abstracts) is used to remove non-PICO sentences for unlabelled data, and the best PICO entity recogniser (BERT pre-trained on PubMed abstracts and full texts) is used to identify PICO phrases and calculate prediction scores across all tokens in each individual text. I explore two thresholds (0.95, 0.99) for records selection, and the results are shown in Figure 4.9. When the threshold is 0.99, no more silver records are included in the training set beyond the first iteration, and self-training does not improve performance. When the threshold is 0.95, the performance fluctuates and the best F1 score is improved by 5% and 1% on the gold validation set and test set respectively, achieved at the sixth iteration step. I terminated the training program after 15 iterations because the training size tends to saturate and the improvement of performance is very limited. For specific PICO entities, the test performance of Comparator and Strain is improved by over 30% and 6% respectively. The enlarged dataset does not help much for other entity types because the changes of test F1 scores are less than 2%, as shown in Table 4.9.

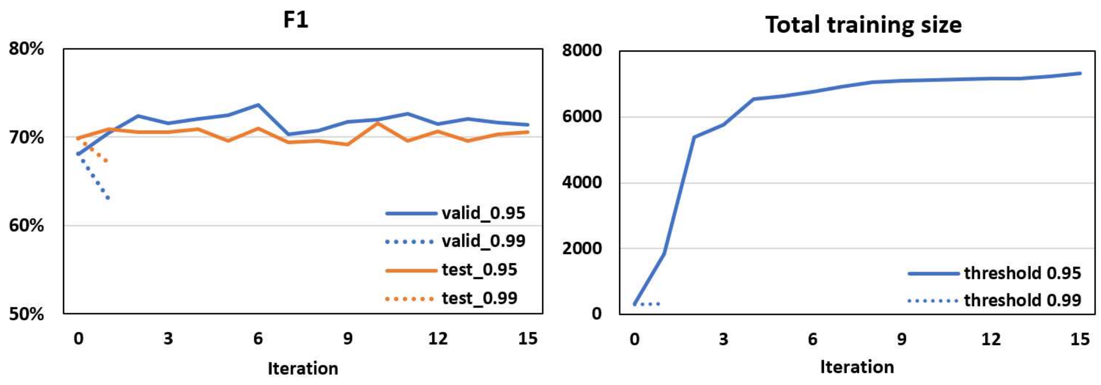


Figure 4.9: Performance of BERT (pre-trained on PubMed abstracts and full texts) for PICO entity recognition using self-training. 'F', 'R', 'P' represents F1, recall and precision separately.

	Gold valid			Gold test		
	F1	**R**	**P**	**F1**	**R**	**P**
Comparator	+46.7	0.0	+77.8	+32.5	+30.0	+21.5
Induction	-0.4	-10.2	+9.9	-1.2	-1.2	-1.1
Intervention	+2.3	+5.4	-0.3	-0.4	-1.5	+0.4
Outcome	+8.4	+5.6	+10.5	+1.5	0.0	+2.7
Species	0.0	0.0	0.0	0.0	0.0	0.0
Strain	+10.9	+20.0	+3.3	+6.6	+5.6	+7.1

Table 4.9: Improvement of entity-level performance of PubMedBERT from the best self-training iteration compared to that of PubMedBERT without self-training on the gold validation and test set (threshold = 0.95).

4.4.3 Overall Performance

In this work, I demonstrate PICO sentence classification models and PICO entity recognition models for PICO extraction in abstracts describing preclinical animal studies. For sentence classification, BERT models with different pre-trained weights have generally good performance (F1 over 80%), and biomedical BERT (BioBERT or PubMedBERT) have slightly better performance than general BERT. For PICO entity recognition, all BERT-based models with biomedical pre-trained weights outperform BiLSTM with or without CRF layer, with an improvement of F1 by between 5% and 9%. It is unnecessary to use more complicated structures based on BERT, as the results of BERT, BERT-BiLSTM and BERT-BiLSTM-CRF do not have much difference, but the latter two require longer training time and resources. Within LSTM based models, adding a CRF layer is beneficial, where recall is increased by 16% and precision is increased by 9%. The training time of LSTM based models is much shorter than fine-tuning BERT, and this could be a quick alternative solution when computing resources are limited. F1 scores are generally good for identifying Species and Strain (over 95% and 70% respectively), satisfactory for Intervention and Outcome (around 60%), acceptable for Induction (around 45%), but very low for Comparator (only 30%) which may need a larger collection of instances and clearer definition distinguished from Intervention. The self-training method helps to identify more comparators and strains which lack an adequate amount of instances in the original training set, although the improvement of overall performance is limited. The enhanced performance of the best PICO entity recogniser is shown in Table 4.10, achieved by fine-tuning BERT pre-trained on PubMed abstracts and full texts and applying self-training.

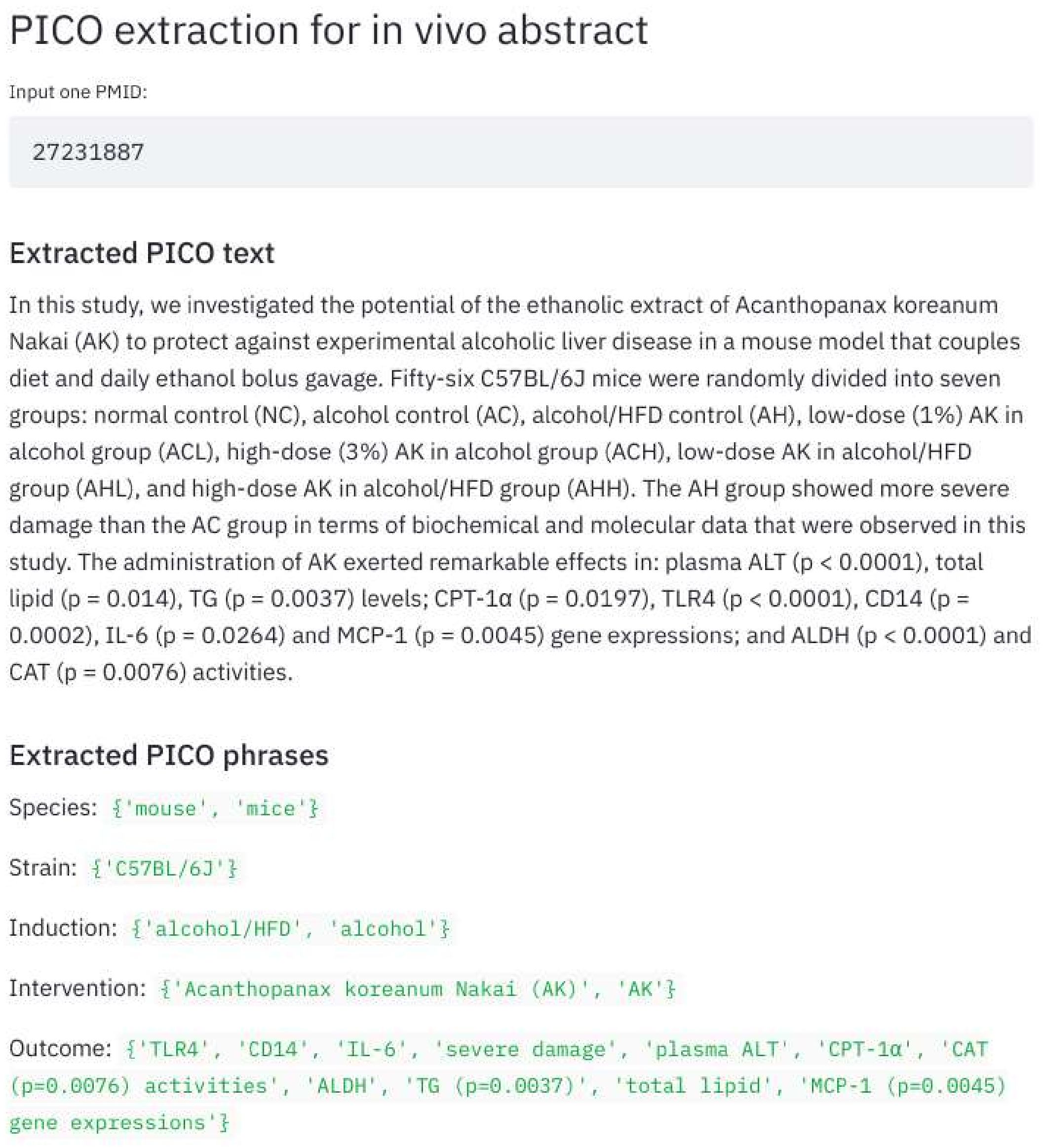
	Validation			Test		
	F1	**R**	**P**	**F1**	**R**	**P**
Overall	73.6	76.4	71.0	71.0	74.0	68.2
Comparator	80.0	66.7	100.0	48.5	40.0	61.5
Induction	45.7	40.7	52.2	48.0	49.4	46.6
Intervention	69.6	75.0	64.9	69.8	74.6	65.6
Outcome	69.9	73.5	66.7	66.9	70.6	63.6
Species	96.4	99.1	93.9	98.1	100.0	96.4
Strain	90.9	100.0	83.3	70.0	77.8	63.6

Table 4.10: Performance of the best PICO entity recognition model (after self-training applied) on the validation and test set. ‘F’, ‘R’, ‘P’ represents F1, recall and precision separately.

4.5 Discussion

4.5.1 Streamlit Interface

I develop an interactive app via Streamlit for potential use which can give a quick overview of PICO elements of an experimental study (see Figure 4.10; codes are available at github.com/qianyingw/pre-pico/tree/master/app). When the user inputs the PMID, the app will call the PubMed Parser (Achakulvisut *et al.*, 2020) to return its title and abstract if the corresponding record is available from the PubMed Open Access Subset database. The background sentence model (pre-trained on PubMedBERT abstracts and fine-tuned on our preclinical PICO sentences) classifies and removes non-PICO sentences. The shortened text consisting of the remaining PICO sentences are then sent into the best PICO entity recogniser (pre-trained on PubMedBERT abstracts and full texts, fine-tuned on the gold preclinical PICO texts and self-trained on the silver preclinical PICO texts) to identify PICO phrases. Some post-processing steps are added to generate the list of candidates for each type of the PICO entity. First, the pre-trained module from PubMedBERT is uncased and its tokeniser converts all words into lower cases, so the extracted PICO strings which are converted directly from the tokens are also in lower cases. The lower-cased PICO strings may be misleading especially for some chemical names. To solve this, I use scispaCy to tokenise the PICO text, where each word is not split into sub-tokens or converted to lower cases without specification. Although BERT's tokeniser further splits those word tokens into sub-word pieces, by recording the word ids of sub-tokens, the sub-word pieces which share the same word id can be merged back to an entire word which can be matched in the list of cased tokens from scispaCy. The entity tag is set to the tag of the first sub-word; other sub-words from the same word may have different entity tags but they are ignored in this processing strategy. Second, the previous training and validation procedures are at token or entity level and the model outputs are a sequence of token tags in BIO format which are not intuitionistic to users, so I also add a function to merge tokens and tags from the same entity, and generate a list of candidates for each entity type. Finally, the duplicate candidates, incomplete brackets and single characters are removed from each string inside each candidate list.



PICO extraction for in vivo abstract

Input one PMID:

27231887

Extracted PICO text

In this study, we investigated the potential of the ethanolic extract of Acanthopanax koreanum Nakai (AK) to protect against experimental alcoholic liver disease in a mouse model that couples diet and daily ethanol bolus gavage. Fifty-six C57BL/6J mice were randomly divided into seven groups: normal control (NC), alcohol control (AC), alcohol/HFD control (AH), low-dose (1%) AK in alcohol group (ACL), high-dose (3%) AK in alcohol group (ACH), low-dose AK in alcohol/HFD group (AHL), and high-dose AK in alcohol/HFD group (AHH). The AH group showed more severe damage than the AC group in terms of biochemical and molecular data that were observed in this study. The administration of AK exerted remarkable effects in: plasma ALT (p < 0.0001), total lipid (p = 0.014), TG (p = 0.0037) levels; CPT-1α (p = 0.0197), TLR4 (p < 0.0001), CD14 (p = 0.0002), IL-6 (p = 0.0264) and MCP-1 (p = 0.0045) gene expressions; and ALDH (p < 0.0001) and CAT (p = 0.0076) activities.

Extracted PICO phrases

Species: `{'mouse', 'mice'}`

Strain: `{'C57BL/6J'}`

Induction: `{'alcohol/HFD', 'alcohol'}`

Intervention: `{'Acanthopanax koreanum Nakai (AK)', 'AK'}`

Outcome: `{'TLR4', 'CD14', 'IL-6', 'severe damage', 'plasma ALT', 'CPT-1α', 'CAT (p=0.0076) activities', 'ALDH', 'TG (p=0.0037)', 'total lipid', 'MCP-1 (p=0.0045) gene expressions'}`

Figure 4.10: A demonstration Streamlit interface for PICO extraction in preclinical abstracts.

4.5.2 Error Analysis

I randomly select 10 abstracts from the test set to investigate the modules of PICO sentence classification and PICO entity recognition.

The PICO sentence classifier works well in most cases as the performance demonstrates. The main error comes from the judgements of the definition of PICO sentences in the annotation process. In some cases, the first introduction sentence explains a PICO phrase and its abbreviation, and the following texts mention only the abbreviation word. I annotate that sentence as PICO sentence because my original purpose is to enable the model to extract the full name which indicates the meaning of the abbreviation word. However, my model does not recognise it as a PICO sentence because most general introduction sentences in an abstract do not describe the actual experimental procedures. In other cases, the model extracts some sentences describing the purpose of the study, explaining the research findings, or discussing the background mechanism as PICO sentences. Those sentences are often placed before the method sentences or after the result sentences, and some of them mention PICO phrases but do not explicitly describe the experimental procedures or the specific outcomes and interventions. Considering the functionality and relative position of those sentences in the entire abstract, I do not annotate them as PICO sentences, but it is ambiguous in the model training.

For PICO entity recognition, one issue is the boundary of PICO phrases. For example, an outcome phrase is ‘level of plasma corticosterone’ but my model extracts ‘plasma corticosterone’. In another example, the outcome annotations are ‘VEGF mRNA’ and ‘VEGF protein’ but my model combines two text spans into one phrase ‘VEGF mRNA and VEGF protein’, which reduces the scores calculated in the validation process but does not affect users to obtain information from the output. The second issue is I did not annotate summarised or indirect phrases but my model extracts those types of outcomes. For example, in the sentence ‘Met-knockdown reduced tumour burden correlating with decreased cell survival and tumour angiogenesis, with minimal effect on cell growth’, my annotation of Outcome includes ‘cell survival’, ‘tumour angiogenesis’ and ‘cell growth’ but excludes ‘tumour burden’ which is extracted by the model. The third issue is the model cannot distinguish induction methods and interventions in some cases. Broadly speaking, the method of induction of disease models belongs to interventions, but it is not the intervention being investigated in the study. The pattern of sentences describing induction methods and interventions are similar, such as ‘applied

…' or 'after … treatment', which increases the difficulties to distinguish the two types of PICO entity.

4.5.3 Limitation and Future Work

One limitation of my work is that the training corpus is at the abstract level but PICO elements are often not described in the abstract particularly for preclinical animal studies. This limits the usage of my applications and I cannot transfer it to full-text identification without further evaluation. Of note, this same limitation applies to manual approaches to identifying PICO elements based on the abstract alone. In a related literature search, for instance, that manual screening for inclusion based on title and abstract has substantially lower sensitivity than manual screening of full texts (https://osf.io/nhjeg). Another limitation is the amount of training/validation/test data is not adequate. Although my best models do not show very inconsistent results between validation and test set (excluding Comparator), the conclusions may still be biased using a small dataset. Previous studies show that self-training can propagate both knowledge and error from high confidence predictions on unlabelled samples (Gao *et al.*, 2021), training from a larger annotated corpus may reduce the error propagation and boost performance. Large datasets also provide possibilities for exploring more complicated models which are proved effective in other tasks.

Future work can evaluate the PICO sentence classification and entity recognition models on some full-text publications to observe any heuristic implications. Existing clinical PICO extraction tools can also be evaluated on preclinical text to identify interventions and outcomes because these two categories may be more similar in preclinical and clinical studies than other PICO elements. The annotated corpus for clinical PICO is relatively larger and more standard and training using combined corpus may yield better performance.

4.6 Summary

I demonstrate a workflow of PICO extraction in preclinical animal text using CRF, LSTM and BERT based models. Without feature engineering, BERT pre-trained on PubMed abstracts is optimal for preclinical PICO sentence classification, and PubMedBERT pre-trained on the combination of PubMed abstracts and full texts achieves the best performance for preclinical PICO entity recognition. PICO entities including Intervention, Outcome, Species, Strain and Strain have acceptable precisions and recalls, while Comparator has a low recall due to the lack of the instances in the training corpus which can be partially solved by the self-training approach.

Chapter 5 Conclusions and Future Work

5.1 Conclusions

In this thesis, I apply natural language processing techniques on two tasks in preclinical systematic reviews - risk of bias assessment and PICO element extraction, and demonstrate the prototype of the automation interface for each task. According to the annotation format of the training samples, the risk of bias assessment is cast as long document classification (Chapter 2), and PICO element extraction is cast as question answering (Chapter 3) and named entity recognition (Chapter 4). The principle of these natural language processing tasks is to first convert a word, sentence or document into numerical representations, and then train classifiers, either traditional machine learning models or recent neural networks, to map the text representations to pre-defined finite categories, such as classifying a full-text publication into the reported/unreported group, or classifying words in publications into different PICO elements.

For automatic risk of bias assessment in preclinical publications (Chapter 2), I implement a range of text representation methods and text classification models for each of five risk of bias items including random allocation, blinded assessment of outcome, conflict of interests, compliance of animal welfare regulations, and animal exclusions. Text representation methods including bag-of-words, word2vec, doc2vec and BERT embeddings are mainly explored in three baseline classifiers. Among text representation methods, word2vec and doc2vec outperform bag-of-words because bag-of-words represent words based on the frequency and do not contain any semantic information. Doc2vec performs better than word2vec in most cases because document vectors from doc2vec are trained from the preclinical training set, while the document vectors from word2vec are averaged fixed word vectors pre-trained on a general PubMed corpus, which may not be as close as to the preclinical domain and the averaging operation across words weakens the information contained. When using the doc2vec method to generate representation vectors, the distributed bag-of-words method outperforms distributed memory or the combination of two methods, which differs from the suggestion in the original study (Le and Mikolov, 2014). Similarly, document vectors generated from the averaged BioBERT embeddings do not show any advantages and perform worse than the averaged biomedical word2vec. Among three baseline models, support vector machine and logistic regression generally achieve better performance than random forest, and are less likely to be over-fitting. Neural network models including convolutional neural network, recurrent neural network with attention mechanism and hierarchical neural network yields better performance than baseline models, increasing F1 by 10%-24%, for all items except compliance of animal welfare regulations. Neural network models enable the word embeddings jointly trainable within the network and unfreezing embedding achieves better results. The performance of RNN with attention does not differ much from that of CNN, and the selection of RNN cell structure (LSTM or GRU) depends on the specific item and whether the bidirectional structure is applied. The hierarchical attention structure among words and sentences in HAN does not yield a better performance compared to CNN and RNN, but the attention weights can be used to score sentences and extract the most relevant sentence for each risk of bias item in a publication. BERT has proved to be the state-of-the-art method in a wide range of NLP tasks (Devlin et al., 2018), but its usage is limited to short documents. To meet the requirement of document length, I propose two strategies of adapting BERT for long documents: document chunk pooling (DCP) and sentence extraction (SE). These two strategies only demonstrate advantages for compliance of animal welfare regulations and results for other items are competitive with neural network models. The best pooling method in the DCP strategy is concatenating hidden states from the output of all BERT chunks. The number of sentences extracted in the SE strategy does not have a big impact on the performance. I also explore adding an additional linear/convolutional/LSTM layer on top of BERT layers, and find the convolutional layer is the optimal selection in the DCP strategy, while three layers do not show much difference in the SE strategy. Considering the unpromising performance gain from BERT-based models and limited computing resources, CNN or RNN with attention are recommended for risk of bias classification in general. The best model for each risk of bias item improves performance significantly by 13%-36%, compared to the previous regular expression approaches.

When question answering approaches are applied to extract PICO elements from biomedical publications (Chapter 3), some extra processing steps are required especially when the context is at full-text level or the publication reports multiple different PICO elements. For the preclinical dataset where I extract the method of intervention/induction from full texts, I explore three unsupervised sentence retrieval methods including bag-of-words with TF-IDF, BM25 and Sentence-BERT, to extract relevant sentences related to intervention or induction method. Around 20 to 30 sentences should be extracted from full texts so 90% of the shorter passages concatenated from the extracted sentences can contain the answer strings. Among three retrieval methods, Sentence-BERT performs slightly better than TF-IDF and BM25 with an improvement of performance around 2%, and the difference of performance between TF-IDF and BM25 is small especially when the number of extracted sentences is high. After sentence retrieval, I implement three question answering models to extract precise intervention/induction answer candidates on three datasets. Baseline question answering models including BiDAF and QANet achieve satisfactory performance on the clinical factoid-type data (F1 over 60%), while the performance for two preclinical data is low (F1 around 40% and 14% respectively). An explanation for this might be that the clinical abstracts are relatively well-written, while preclinical texts often contain multiple interventions which are less explicit. QANet slightly improves performance by 1% to 4% compared to that of BiDAF, and a high number of blocks in the modelling encoder layer does not guarantee better performance. The best question answering approach is fine-tuning BERT, which improves the answer extraction performance by 30% to 50% on preclinical data, and 20% on clinical data. The selection of pre-trained weights in BERT does not have a significant effect on clinical data, but different choices can induce 20%

difference of F1 score on preclinical data. The weight induced from pre-training BERT on only PubMed abstracts is the best choice across all three datasets. The poor performance on the preclinical list-type data indicates the difficulties of extracting the correct number of answer candidates and handling the boundary of an intervention/induction phrase properly.

Considering the limitations of extracting PICO elements by question answering methods with the existing datasets, I annotate 400 preclinical abstracts and implement named entity recognition models to identify six types of PICO elements (Chapter 4). Less than half of the sentences in an abstract describe PICO elements so I fine-tune BERT to classify PICO sentences in the abstracts. All four pre-trained weights in BERT show good performance and the module pre-trained on PubMed abstracts achieves the highest F1 of 89%, for identifying PICO sentences. Then I explore several named entity recognition models based on the conditional random fields, bidirectional LSTM, BERT and their combinations, and train them on the truncated abstracts where all non-PICO sentences are removed. For overall performance, CRF and BiLSTM alone achieve competitive results, and BiLSTM-CRF outperforms those two models with F1 improved by 14%. BiLSTM-CRF is also less sensitive to hidden dimension than CRF and BiLSTM. Fine-tuning BERT pre-trained on the biomedical corpus (BioBERT and two PubMedBERT) improves the overall F1 by 10% compared to BiLSTM-CRF, while the result from BERT pre-trained on the general domain corpus is even worse than BiLSTM-CRF. I also implement BERT-LSTM and BERT-LSTM-CRF with different pre-trained modules, but the additional LSTM or CRF layer does not show any advantages. By PICO entity level, all models achieve good performance for Species and Strain, with F1 over 95% and 70% separately, because the phrases of those two elements are limited to certain types so the task is relatively easy. The performance for Intervention, Outcome and Induction is less satisfactory by BiLSTM or CRF based models, but is improved greatly by BERT models, with F1 around 70%, 70% and 50% respectively. The performance for Comparator is poor and the highest F1 is only 30%, which is caused by the limited number of comparator instances. To solve the data-hungry issue, I apply a self-training approach to obtain silver data from unlabelled abstracts, and the model trained on the enlarged dataset improved the test performance for Comparator and Strain, with F1 increased by 30% and 7% respectively.

Across all three projects, BERT models with biomedical pre-trained module are good selections for the automation of risk of bias assessment and PICO extraction, which require less feature engineering and hyperparameters tuning procedures, but always achieve better performance than baseline machine learning models or neural models in general. BERT models often have a large number of training parameters and require a long training time, which is prone to be over-fitting on the small biomedical training samples. These attributes may be also the reason that models with more complicated architecture based on BERT do not help with performance much. When the training documents are long or the computing resources (GPU, memory space) are limited, the advantages of the powerful transformer structure and pre-trained domain knowledge associated with BERT are hindered, and extra procedures are needed such as splitting documents into chunks or extracting relevant sentences are needed. Some simple neural networks such as CNN, LSTM or probabilistic models such as CRF based classifiers may be good substitutions to achieve relatively robust performance.

5.2 Future Work

There are remaining challenges for automatic preclinical systematic reviews. I discuss some directions for future work as follows.

More datasets

One direction is to obtain more datasets for developing automation tools for preclinical systematic reviews. Large training samples provide possibilities for developing deep neural networks or other complicated models such as BERT which often achieve the state-of-the-art performance in many NLP tasks. The large training samples also guarantee a sufficient amount and the variation of types of certain instances. With more training samples, in risk of bias assessment, models would have more opportunities to learn different description patterns of risk of bias reporting in publications; in PICO entity recognition, it is possible to develop less biased models for comparators. Large datasets would reduce the risk of models which tend to be over-fitting due to the small training size, hence the evaluation performance on validation or test samples would be more reliable.

There are also concerns about the generalisability. The PICO sentence classifier and entity recogniser are developed on a small number of preclinical abstracts (Chapter 4). Both the text classification (Chapter 2) and question answering (Chapter 3) tasks use the preclinical datasets which are limited to two or three cardiovascular or neurological diseases. It may affect the generalisability of model performance and evaluations of publications focusing on other preclinical studies are necessary. Therefore, the collection of datasets from a variety of preclinical research topics is needed to enable the evaluations and the development of general models.

In addition, the preclinical datasets should be easily accessible. The number of preclinical systematic reviews is rapidly increasing but the open-access datasets or structured databases are scarce, compared with datasets of clinical systematic reviews, such as the Cochrane Database of Systematic Reviews. Many preclinical protocols or publications do not cover the details and statistics of data collection, data curation, data structures and validation of annotations. Those information could be published in a separate report or publication to enable a wider application of the datasets, which would also benefit the NLP or computer science field.

Better annotations

Another direction is the collection of fine-grained annotations. First, preclinical texts of better quality should be obtained. In most of my datasets, plain full texts are converted from PDFs by Xpdf, which contains a lot of noisy information and requires some cleaning process. Although structured PubMed

XMLs are widely available for many recent publications, only PDFs exist for some publications included in preclinical systematic reviews not limited by date. For those publications, potential solutions include retrieving databases by title, PMID or PMCID to obtain XMLs, or parsing PDFs to XMLs via tools such as Grobid, which may need further evaluation.

Second, the annotation strategies and tools could be improved to obtain precise and informative annotations. Many preclinical systematic reviewers annotate experimental characteristics from publications by taking notes in Word or Excel; some may complete the process assisted by a systematic review platform such as SyRF (Bahor *et al.*, 2021). In those cases, typos might be generated when they record PICO phrases or sentences from a large amount of collection of publications, which brings difficulties to locate the exact information from the whole publication or validate annotations among different reviewers. In an ideal scenario, the preclinical texts are demonstrated in a structured HTML webpage or PDF reviewer (when the XMLs of publications are unavailable), and systematic reviewers can just highlight the sentences of each risk of bias item or PICO phrases of each experiment, with different colours to indicate different items or experimental group information. The background program of the annotation tool would record the annotated texts, the corresponding labels and their positions, and all the information would be automatically stored in a database with pre-defined structures. Annotation tools for general NLP usages exist such as tagtog (https://www.tagtog.net), but annotations tools designed for preclinical systematic reviews remained to be developed, which should contain some specific functions to reduce the effort of data conversion and curation for the following systematic review procedures.

Another concern is the granularity of annotations. For risk of bias assessment, I have demonstrated the good performance of a range of classification models for four risk of bias items. However, almost all models are trained on full texts, while the risk of bias descriptions often consist of only one or two sentences. This makes the training inefficient, and the error analysis indicates that my models do not correctly locate and recognise the risk of bias sentences in some cases. Models for other risk of bias items including sample size calculation and allocation concealment remain to be developed and if future annotators can record the complete sentences which indicate the reporting of a risk of bias item, more flexible and efficient approaches can be designed. For example, the collected risk of bias sentences can be analysed to develop the generalised preclinical risk of bias embeddings for unsupervised strategies, as the reporting should follow certain patterns. For PICO extraction, I do not recommend using question answering models because there are always multiple mentions for one PICO element and it is difficult to predict the number of mentions and the most appropriate candidate. However, it might be promising to develop better sentence retrieval methods or classifiers if the top relevant sentences or summary descriptions of PICO elements are available. In addition, annotations among different reviewers should be evaluated by some statistical methods such as Dawid-Skene model which estimates the reliability of reviewers and Cohen's Kappa statistic to measure the agreement among multiple reviewers (Nye et al., 2018). These evaluation functions could also be embedded in annotation platforms so reviewers could adjust their annotation behaviours when necessary.

Extrinsic evaluation

With a large number of datasets and fine-grained annotations, some automation tools based on NLP techniques could be developed for preclinical systematic review tasks. Before applying tools to the practical projects, reviewers need to perform extrinsic evaluation on some of their own datasets to estimate the applicability of tools.

For risk of bias assessment, if researchers have annotated publications from previous projects, they could apply the tools to those annotated publications to obtain a prediction probability for each risk of bias item in each individual publication. If the annotations contain only the decision of a risk of bias item, such as “yes”, “no” or “unclear”, reviewers could compare it with the prediction label generated from the prediction probability by setting thresholds. For example, when the probability is smaller than 0.4, the prediction of the risk of bias is high; when the probability is larger than 0.6, the prediction of the risk of bias is low; and when the probability is between 0.4 and 0.6, the prediction of the risk of bias is unclear, which may need further investigation of the full texts. The ranges can be changed to <0.1, >0.9, and 0.1~0.9 to obtain more confident predictions. The tools can calculate general evaluation scores including recall, precision and F1, and they can also provide the elements of the confusion matrix, i.e. true positive, true negative, false positive and false negative, so users could calculate other evaluate scores when needed. When sentence-level annotations are available, such as text snippets or a whole sentences indicating the risk of bias reporting, reviewers could compare them with the relevant sentences extracted by the hierarchical neural networks (Section 2.4.2.3) or other sentence extraction methods, where the extracted sentences can be output as a list or shown in the original publications embedded in the PDF/HTML reviewer with highlight colours, so users can check the context sentences. Based on the comparison above, reviewers could have a general idea of whether the tools can be applied to their field or how confident the prediction results are.

For PICO extraction, where it might be difficult to obtain a large amount of annotated datasets in short times, reviewers can randomly select 10 to 20 samples from the included publications and annotate PICO phrases or sentences manually. Then they need to compare the machine-extracted and human extracted PICO phrases, where the performance can be evaluated by reviewers' own experiences or understandings, the holistic metrics such as recall and F1 scores, or a more interpretable evaluation methodology which considers multiple entity attributes at the same time, including entity length, entity/token frequency, label consistency of entity/token, etc (Fu *et al.*, 2020). The extracted PICO phrases can be demonstrated in the original publications embedded in the PDF/HTML reviewer with multiple highlight colours for different PICO entity types, so users can easily check the location of the PICO phrases and the context sentences. With the general/interpretable evaluation scores and reviewers' own understandings, reviewers can decide if they want to apply the tools to extract one or multiple types of PICO entities from all their publications.

Cross-field evaluation

Although there are some differences between preclinical and clinical systematic reviews, some reporting strategies and characteristics of experiments or trials are common, and some automation tools developed for clinical systematic reviews could be evaluated on preclinical publications. For example, RobotReviewer (Marshall et al., 2016), the clinical risk of bias assessment tool, could be used to identify the reporting of blinded assessment of outcomes in preclinical publications; and Trialstreamer (Marshall et al., 2020) could be used to identify interventions and outcomes in preclinical experiments.

Easier, quicker and lighter tools

Automation tools for risk of bias assessment, PICO extraction and other preclinical systematic review tasks should be easier to use, without the need for specialist operating systems, complicated installation or other preparation steps. Users do not need to have any NLP or computer science knowledge, and ideally, all the actions just include uploading files, selecting models/settings and obtaining results in some user-defined formats which are convenient to process for further analyses. The tools can also include some interactive functions, such as modifying the annotation and recording the modification in the background so models can be re-trained and updated to adapt to users' datasets from different fields. Cloud platforms sometimes are better than locally deployed tools, where the former allow collaborations and evaluations among multiple reviewers; if there is a demand for processing a large amount of publications, the computations can be done in the background modules and the platform can send a notification to users when results are produced, so they do not need to check progress frequently. The reaction speed is also important, especially for small evaluations which may affect users' intention to continue using it for processing a large amount of publications. BERT-based models often achieve decent performance but they are associated with large trained modules, which are heavy and slow to load and conduct the computations when GPU is unavailable. Developing lighter and quicker tools is necessary to promote usability.

Appendix

Appendix Table 1: Validation performance of baseline models using different doc2vec methods to generate document vectors for risk of bias classification.

Classifier	RoB item	d2v method	F1	Recall	Precision	Specificity
Support vector machine	Random allocation	DM	40.7	57.2	31.5	59.2
		DBOW	51.9	72.2	40.5	65.1
		DM+DBOW	44.9	62.4	35.1	62.0
	Blinded assessment of outcome	DM	48.5	60.4	40.5	63.2
		DBOW	59.3	67.8	52.7	74.7
		DM+DBOW	54.0	79.1	41.0	52.7
	Conflict of interests	DM	57.0	67.5	49.3	73.2
		DBOW	67.1	79.7	57.9	77.7
		DM+DBOW	59.6	60.4	58.9	83.8
	Compliance of animal welfare regulations	DM	80.8	74.0	88.9	69.9
		DBOW	86.7	82.7	91.2	74.1
		DM+DBOW	74.9	62.9	92.7	83.7
	Animal exclusions	DM	34.1	49.1	26.1	76.8
		DBOW	39.0	64.3	28.0	72.5
		DM+DBOW	34.0	48.2	26.2	77.4
Logistic regression	Random allocation	DM	48.2	62.9	39.1	67.8
		DBOW	51.0	65.5	41.8	70.0
		DM+DBOW	48.8	61.3	40.5	70.3
	Blinded assessment of outcome	DM	54.1	61.7	48.1	72.4
		DBOW	60.0	69.1	53.0	74.5
		DM+DBOW	54.9	61.7	49.5	73.8
	Conflict of interests	DM	60.8	65.0	57.1	81.2
		DBOW	68.8	76.1	62.8	82.6
		DM+DBOW	63.9	69.5	59.1	81.4
	Compliance of animal welfare regulations	DM	82.5	76.9	88.9	68.7
		DBOW	87.6	85.4	89.9	68.7
		DM+DBOW	87.0	84.9	89.1	66.3
	Animal exclusions	DM	32.1	44.6	25.0	77.7
		DBOW	41.4	62.5	31.0	76.8

		DM+DBOW	34.0	42.9	28.2	81.8
Random forest	Random allocation	DM	39.7	46.9	34.5	70.7
		DBOW	41.4	51.5	34.6	68.0
		DM+DBOW	50.4	59.8	43.6	74.6
	Blinded assessment of outcome	DM	46.8	55.2	40.6	66.4
		DBOW	57.8	68.3	50.2	71.8
		DM+DBOW	52.7	61.7	46.0	69.9
	Conflict of interests	DM	53.9	58.4	50.0	77.5
		DBOW	65.1	68.5	61.9	83.8
		DM+DBOW	61.8	64.5	59.3	83.0
	Compliance of animal welfare regulations	DM	85.2	86.2	84.3	47.6
		DBOW	88.8	89.7	88.0	60.2
		DM+DBOW	88.1	89.5	86.8	55.4
	Animal exclusions	DM	33.8	54.5	24.5	72.0
		DBOW	36.4	50.0	28.6	79.2
		DM+DBOW	40.9	61.6	30.7	76.8

Appendix Table 2: Validation performance of baseline models with different dimensions of document vectors generated from doc2vec in the risk of bias classification task.

Classifier	RoB item	vector dim	F1	Recall	Precision	Specificity
Support vector machine	Random allocation	100	46.5	59.3	38.2	68.5
		200	49.4	69.1	38.4	63.6
		300	51.9	72.2	40.5	65.1
		400	47.0	65.5	36.7	62.9
		500	47.9	79.9	34.2	49.5
	Blinded assessment of outcome	100	57.5	68.3	49.7	71.3
		200	59.3	67.8	52.7	74.7
		300	55.7	76.5	43.8	59.2
		400	57.3	70.4	48.4	68.8
		500	57.6	67.4	50.3	72.4
	Conflict of interests	100	60.3	84.8	46.8	62.8
		200	66.1	76.6	58.1	78.7
		500	67.1	79.7	57.9	77.7
		400	65.2	66.5	63.9	85.5
		300	65.2	74.1	58.2	79.5
	Compliance of animal welfare regulations	100	86.7	82.7	91.2	74.1
		200	86.6	82.8	90.7	72.3
		300	83.0	76.9	90.1	72.3
		400	83.3	78.2	89.1	68.7
		500	83.7	78.4	89.9	71.1
	Animal exclusions	100	35.1	55.4	25.7	73.4
		200	39.0	64.3	28.0	72.5
		300	35.7	56.3	26.1	73.5
		400	37.9	58.0	28.1	75.3
		500	37.7	50.0	30.3	80.8
Logistic regression	Random allocation	100	48.8	64.9	39.1	66.8
		200	50.0	64.9	40.6	68.8
		300	51.0	65.5	41.8	70.0
		400	48.8	63.9	39.5	67.8
		500	50.0	63.4	41.3	70.3
		100	57.7	65.7	51.5	74.4

	Blinded assessment of outcome	200	57.2	65.7	50.7	73.5
		300	60.0	69.1	53.0	74.5
		400	58.9	68.3	51.8	73.6
		500	59.5	69.6	51.9	73.3
	Conflict of interests	100	66.4	75.1	59.4	80.2
		200	65.6	75.6	58.0	78.9
		300	67.1	76.1	60.0	80.4
		400	68.8	76.1	62.8	82.6
		500	66.4	73.6	60.4	81.4
	Compliance of animal welfare regulations	100	87.1	83.9	90.5	71.1
		200	87.6	85.4	89.9	68.7
		300	86.8	83.8	90.1	69.9
		400	85.9	82.8	89.3	67.5
		500	87.0	84.9	89.3	66.9
	Animal exclusions	100	35.0	53.6	26.0	74.6
		200	38.3	56.3	29.0	77.1
		300	39.3	57.1	29.9	77.7
		400	41.4	62.5	31.0	76.8
		500	40.3	57.1	31.1	78.9
Random forest	Random allocation	100	45.5	57.2	37.8	69.0
		200	41.7	51.0	35.2	69.2
		300	44.4	54.6	37.3	69.8
		400	50.4	59.8	43.6	74.6
		500	48.3	58.8	41.0	72.2
	Blinded assessment of outcome	100	55.6	67.4	47.3	68.8
		200	53.0	64.3	45.1	67.5
		300	50.3	58.3	44.2	69.5
		400	49.3	57.8	42.9	68.1
		500	57.8	68.3	50.2	71.8
	Conflict of interests	100	65.1	68.5	61.9	83.8
		200	64.8	69.0	61.0	83.0
		500	64.4	63.5	65.4	87.1
		400	63.8	67.0	60.8	83.4
		300	61.0	61.9	60.1	84.1
		100	86.5	86.0	87.1	58.4

	Compliance of animal welfare regulations	200	85.5	85.4	85.6	53.0
		300	87.0	86.9	87.1	57.8
		400	86.6	86.2	87.0	57.8
		500	88.8	89.7	88.0	60.2
	Animal exclusions	100	30.6	50.9	21.8	69.6
		200	40.9	61.6	30.7	76.8
		300	35.0	50.0	26.9	77.4
		400	36.0	51.8	27.6	77.4
		500	33.3	45.5	26.3	78.7

Appendix Table 3: Validation performance of the convolutional neural network with different number of filters for each filter size in the risk of bias classification task.

	No. filters	F1	Recall	Precision	Specificity
Random allocation	20	82.4	88.8	77.9	91.6
	40	83.3	91.3	77.7	91.3
	60	82.8	86.7	81.1	93.0
	80	82.9	88.5	79.1	92.6
	100	83.7	90.0	79.2	91.5
	120	84.0	89.9	80.6	92.3
	140	83.3	88.7	79.3	92.6
	160	83.7	89.9	79.7	91.8
	180	83.1	89.4	78.5	92.0
	200	83.4	90.8	78.1	91.7
	300	82.8	90.9	77.8	91.2
Blinded assessment of outcome	20	81.4	88.6	76.0	88.3
	40	79.4	89.2	73.0	86.6
	60	79.7	90.5	71.8	85.0
	80	80.7	89.8	74.0	86.6
	100	81.2	92.6	73.8	86.2
	120	79.8	89.7	73.3	86.6
	140	81.6	89.2	76.0	87.5
	160	80.4	90.1	74.3	87.1
	180	79.6	89.6	72.7	85.6
	200	80.1	88.4	73.9	86.6
	300	80.2	89.0	74.0	87.3
Conflict of interests	20	83.1	86.9	81.5	92.2
	40	83.3	85.3	82.7	93.0
	60	84.0	86.5	83.5	93.4
	80	84.4	87.6	83.1	93.2
	100	84.5	86.8	84.1	93.8
	120	84.2	87.2	83.3	93.4
	140	83.9	86.6	82.4	92.2
	160	84.4	87.6	83.0	93.5
	180	84.2	87.6	82.8	93.0
	200	84.4	87.6	83.1	93.5

	300	83.5	83.9	85.3	94.6
Compliance of animal welfare regulations	20	76.2	70.9	84.0	96.3
	40	76.2	71.0	86.0	96.2
	60	80.1	77.0	87.9	96.5
	80	80.3	76.1	89.0	97.0
	100	81.1	77.3	89.1	97.2
	120	83.1	82.1	86.0	95.8
	140	83.8	82.3	88.7	96.9
	160	84.8	82.9	88.4	96.7
	180	84.1	81.8	88.1	96.5
	200	85.5	82.5	90.5	96.9
	300	85.1	82.5	89.5	97.1
Animal exclusions	20	38.7	64.7	29.7	76.0
	40	47.5	56.6	43.6	87.9
	60	53.5	70.8	45.1	84.1
	80	53.4	67.5	46.3	85.6
	100	55.5	66.4	51.7	88.7
	120	56.0	68.2	49.4	87.7
	140	51.7	75.1	43.7	85.0
	160	54.2	71.0	48.2	87.1
	180	52.8	65.0	49.3	90.2
	200	51.5	69.9	46.1	87.3
	300	54.8	68.0	50.0	88.9

Appendix Table 4: Validation performance of the convolutional neural network with different filter sizes for risk of bias classification.

	filter size	F1	Recall	Precision	Specificity
Random allocation	3,4,5	84.0	89.9	80.6	92.3
	4,5,6	83.2	89.8	79.5	91.8
	5,6,7	83.4	91.6	77.8	90.5
	6,7,8	84.1	90.6	79.5	92.4
	7,8,9	83.5	90.7	78.3	91.3
	8,9,10	83.5	90.2	79.0	91.6
	9,10,11	83.6	92.8	77.1	91.1
	10,11,12	85.7	90.6	82.7	93.1
	11,12,13	84.3	90.6	79.8	92.5
Blinded assessment of outcome	3,4,5	81.6	89.2	76.0	87.5
	4,5,6	80.3	88.5	74.6	87.9
	5,6,7	80.4	90.8	73.8	87.1
	6,7,8	81.2	89.4	74.9	87.2
	7,8,9	81.1	88.5	75.5	87.8
	8,9,10	80.7	90.1	74.5	86.3
	9,10,11	82.4	88.5	77.8	89.4
	10,11,12	80.6	89.6	74.8	87.7
	11,12,13	81.0	90.1	75.1	87.6
Conflict of interests	3,4,5	84.5	86.8	84.1	93.8
	4,5,6	83.2	84.7	84.1	94.0
	5,6,7	83.5	83.9	85.1	94.5
	6,7,8	83.0	84.9	83.1	93.6
	7,8,9	83.6	85.6	83.7	93.8
	8,9,10	83.5	85.6	83.4	93.6
	9,10,11	83.5	85.7	83.6	93.6
	10,11,12	84.0	85.2	85.1	94.3
	11,12,13	84.4	85.7	85.2	94.3
Compliance of animal welfare regulations	3,4,5	86.9	83.3	92.4	97.4
	4,5,6	85.8	84.0	88.6	96.8
	5,6,7	83.9	79.1	92.3	97.7
	6,7,8	85.2	81.8	90.7	96.6
	7,8,9	84.7	84.1	86.4	95.2

	8,9,10	83.9	82.2	87.6	96.3
	9,10,11	83.8	82.4	86.8	96.0
	10,11,12	82.5	80.2	87.4	96.3
	11,12,13	83.7	81.9	87.4	96.2
Animal exclusions	3,4,5	56.7	67.6	53.7	90.8
	4,5,6	53.1	66.1	46.3	85.8
	5,6,7	51.8	65.2	46.1	85.3
	6,7,8	54.6	66.8	52.2	90.3
	7,8,9	56.0	70.9	51.3	89.5
	8,9,10	42.0	50.0	45.8	90.0
	9,10,11	54.1	60.1	53.7	90.8
	10,11,12	56.3	68.0	51.6	87.9
	11,12,13	35.4	50.1	29.1	79.5

Appendix Table 5: Validation performance of the convolutional neural network with different maximum vocabulary sizes for risk of bias classification.

	vocab size	F1	Recall	Precision	Specificity
Random allocation	3000	78.6	83.4	75.6	91.1
	5000	85.7	90.6	82.7	93.1
	7000	86.4	93.2	81.8	92.8
	9000	84.7	90.8	80.7	93.3
Blinded assessment of outcome	5000	82.4	88.5	77.8	89.4
	7000	81.6	89.7	75.6	87.7
	9000	81.2	92.5	73.8	85.4
Conflict of interests	5000	84.5	86.8	84.1	93.8
	7000	83.7	86.8	83.3	93.5
	9000	84.1	89.1	80.8	91.7
Compliance of animal welfare regulations	5000	86.9	83.3	92.4	97.4
	7000	84.3	81.1	89.6	97.0
	9000	85.0	84.7	87.2	96.2
Animal exclusions	5000	56.7	67.6	53.7	90.8
	7000	54.1	60.1	53.7	90.8
	9000	60.2	73.6	54.2	89.7

Appendix Table 6: Validation performance of the recurrent neural network with different hidden dimension for risk of bias classification.

	hidden dim	F1	Recall	Precision	Specificity
Random allocation	2	83.1	88.2	80.2	92.8
	6	85.4	89.6	82.9	93.5
	10	86.3	86.5	86.9	95.7
	20	86.3	89.5	84.7	94.7
	30	83.5	93.4	76.6	90.7
	40	84.0	89.6	81.4	93.0
	50	85.7	92.7	80.9	92.2
	100	84.6	89.9	81.2	92.9
Blinded assessment of outcome	2	78.7	85.6	74.3	88.2
	6	81.2	87.8	76.1	88.5
	10	82.1	86.6	79.6	89.8
	20	82.1	90.6	75.8	87.9
	30	80.9	87.2	76.6	89.0
	40	80.2	86.2	75.7	88.5
	50	82.0	87.8	78.3	90.2
	100	80.7	89.9	74.6	87.4
Conflict of interests	2	78.6	82.4	77.4	90.8
	6	77.7	78.0	79.1	91.7
	10	77.9	81.9	76.7	90.9
	20	79.6	83.0	77.8	91.3
	30	78.0	78.9	79.0	91.8
	40	53.5	66.9	46.1	70.5
	50	55.9	56.6	57.3	83.7
	100	75.6	73.1	80.2	93.7
Compliance of animal welfare regulations	2	74.8	72.9	81.9	94.0
	6	70.3	70.4	75.5	90.7
	10	66.1	59.9	78.2	94.6
	20	60.4	53.6	76.3	94.8
	30	60.5	55.0	75.2	93.9
	40	54.8	47.4	68.0	93.8
	50	55.3	47.2	72.9	94.2
	100	54.7	43.5	83.0	96.7

Animal exclusions	2	35.3	54.5	30.6	79.5
	6	58.0	68.3	54.3	90.0
	10	44.3	58.4	37.7	83.4
	20	57.3	66.7	55.5	90.9
	30	52.8	66.1	46.2	87.2
	40	28.2	68.0	18.4	46.3
	50	54.1	74.0	44.1	83.9
	100	22.1	48.2	15.1	54.6

Appendix Table 7: Validation performance of the hierarchical neural network with different sentence length limitation for risk of bias classification.

	Sentence length	F1	Recall	Precision	Specificity
Random allocation	10	72.0	67.0	82.5	95.1
	20	77.4	80.6	76.0	91.5
	30	83.8	89.5	80.1	92.0
	40	86.2	91.3	83.1	93.7
	50	83.8	91.3	79.4	91.8
	60	85.4	90.6	81.8	93.0
	70	84.8	94.1	78.8	91.3
	80	84.5	88.7	82.0	93.8
	90	84.2	91.6	79.6	91.9
	100	85.4	90.3	82.1	93.2
Blinded assessment of outcome	10	68.3	75.7	64.3	82.3
	20	75.4	80.3	73.5	87.7
	30	78.0	81.9	76.5	89.2
	40	80.6	86.1	77.7	90.0
	50	81.3	86.4	77.5	89.1
	60	78.7	86.3	73.4	87.3
	70	51.9	56.2	50.6	78.5
	80	81.2	88.3	76.6	88.6
	90	80.4	84.5	77.4	89.4
	100	81.0	87.8	75.9	88.7
Conflict of interests	10	72.1	75.3	71.4	89.1
	20	81.6	81.0	83.6	93.9
	30	80.5	83.1	79.6	91.8
	40	83.2	84.7	82.8	93.2
	50	82.0	85.7	80.0	92.1
	60	80.5	80.1	83.6	94.2
	70	74.7	77.7	73.8	91.4
	80	59.9	60.4	63.4	85.4
	90	82.1	82.3	82.7	93.2
	100	81.2	82.9	80.4	92.2
	10	54.2	49.6	63.6	91.7

Compliance of animal welfare regulations	20	54.2	49.6	63.6	91.7
	30	50.3	39.5	77.4	97.2
	40	53.9	49.6	62.7	91.3
	50	77.4	78.1	79.7	93.2
	60	53.8	45.2	74.9	96.1
	70	52.1	46.1	68.6	93.6
	80	52.8	49.9	64.4	92.6
	90	53.5	46.0	68.7	94.5
	100	74.5	69.9	85.2	97.3
Animal exclusions	10	30.2	59.7	21.6	60.3
	20	27.7	55.8	19.9	63.7
	30	30.2	59.6	21.1	60.4
	40	35.2	76.7	23.8	55.0
	50	31.9	72.4	21.0	53.9
	60	45.3	58.9	38.5	83.2
	70	26.9	63.2	18.1	55.6
	80	31.8	82.3	21.1	50.3
	90	32.0	69.2	21.2	54.8
	100	46.7	67.8	37.8	81.6

Appendix Table 8: Validation performance of the hierarchical neural network with different document length limitation for risk of bias classification.

	Document length	F1	Recall	Precision	Specificity
Random allocation	200	82.4	88.5	78.7	91.7
	400	84.8	92.2	80.0	92.1
	500	86.2	91.3	83.1	93.7
	600	80.7	92.7	73.5	89.4
	800	84.2	88.7	81.5	93.4
	1000	85.5	91.7	81.3	93.3
Blinded assessment of outcome	200	76.9	83.4	72.7	87.2
	400	79.2	85.6	75.3	88.3
	500	81.3	86.4	77.5	89.1
	600	80.1	85.5	76.7	89.1
	800	79.3	83.9	76.6	89.7
	1000	80.9	87.5	76.7	89.3
Conflict of interests	200	72.9	74.3	74.1	89.8
	400	79.5	79.6	82.2	93.5
	500	83.2	84.7	82.8	93.2
	600	59.5	60.2	60.7	85.5
	800	80.3	80.2	82.6	93.6
	1000	81.2	82.7	81.1	92.4
Compliance of animal welfare regulations	200	63.8	54.9	83.3	96.9
	400	58.3	49.9	79.2	95.4
	500	77.4	78.1	79.7	93.2
	600	79.3	77.8	84.5	94.9
	800	52.4	43.0	73.5	95.0
	1000	38.1	27.5	74.9	98.6
Animal exclusions	200	47.0	56.6	45.4	88.6
	400	52.8	65.3	46.1	87.3
	500	51.2	62.8	46.1	86.9
	600	43.7	68.9	34.5	79.0
	800	52.8	61.9	49.7	90.2
	1000	43.8	67.2	35.2	79.8

Appendix Table 9: Validation performance of the hierarchical neural network with different hidden dimension of word/sentence hidden states in the risk of bias classification task.

	Hidden dim	F1	Recall	Precision	Specificity
Random allocation	5	84.3	89.5	81.7	92.8
	10	85.8	84.9	88.0	95.9
	20	84.0	85.4	84.3	95.1
	30	86.0	90.6	83.2	94.1
	40	85.8	90.8	83.3	94.1
	50	86.2	91.3	83.1	93.7
	60	84.4	91.8	78.9	91.9
	80	85.0	91.5	80.6	92.9
Blinded assessment of outcome	5	80.0	87.7	74.9	87.9
	10	80.4	87.9	76.3	87.4
	20	80.1	86.0	76.4	89.4
	30	80.7	86.9	77.0	89.1
	40	79.6	86.3	74.7	88.3
	50	81.3	86.4	77.5	89.1
	60	79.8	86.9	74.2	87.0
	80	80.9	87.6	76.7	89.3
Conflict of interests	5	79.2	81.2	78.8	91.1
	10	78.6	82.4	77.2	90.9
	20	80.0	84.3	78.5	91.5
	30	81.0	80.6	82.7	93.4
	40	79.3	79.1	81.6	92.9
	50	83.2	84.7	82.8	93.2
	60	79.4	77.8	83.5	94.1
	80	75.3	73.4	79.1	94.3
Compliance of animal welfare regulations	5	78.6	79.5	80.9	94.3
	10	64.0	64.6	67.0	89.2
	20	59.0	51.9	74.7	95.4
	30	56.3	50.7	70.5	94.1
	40	57.3	59.4	58.0	87.0
	50	77.4	78.1	79.7	93.2

	60	56.1	48.8	70.7	94.8
	80	0.0	0.0	0.0	99.8
Animal exclusions	5	51.1	59.0	49.8	91.3
	10	35.2	60.6	27.1	74.1
	20	51.2	62.8	46.1	86.9
	30	32.1	60.1	22.7	66.6
	40	34.9	61.6	25.6	69.3
	50	46.7	67.8	37.8	81.6
	60	31.5	43.3	25.9	78.1
	80	--	--	--	--

Appendix Table 10: Effect of number of sentences extracted on sentence retrieval performance for the Psycho-CIPN-factoid dataset. sMMR (rMMR) refers to strict (ratio) mean match ratio, and sMAP (rMAP) refers to strict (ratio) mean average precision.

Method	No. sents	sMAP	rMAP	sMMR	rMMR
TF-IDF	5	54.1	54.1	76.5	76.5
	10	51.7	51.7	85.1	85.1
	20	48.0	48.0	90.3	90.3
	25	46.8	46.8	91.7	91.7
	30	45.8	45.8	93.3	93.3
	35	45.1	45.1	94.4	94.4
	40	44.4	44.4	95.0	95.0
	60	42.5	42.5	97.4	97.4
BM25	5	51.6	51.6	74.1	74.1
	10	48.8	48.8	82.4	82.4
	20	45.7	45.7	89.5	89.5
	25	44.5	44.5	91.2	91.2
	30	43.5	43.5	92.3	92.3
	35	42.8	42.8	93.3	93.3
	40	42.1	42.1	94.6	94.6
	60	40.2	40.2	96.7	96.7
SBERT	5	58.7	58.7	77.6	77.6
	10	55.9	55.9	84.7	84.7
	20	51.9	51.9	91.5	91.5
	25	50.4	50.4	93.3	93.3
	30	49.4	49.4	94.6	94.6
	35	48.5	48.5	95.4	95.4
	40	47.8	47.8	95.7	95.7
	60	45.3	45.3	98.2	98.2

Appendix Table 11: Effect of number of sentences extracted on sentence retrieval performance for the overall Psycho-CIPN dataset (Psycho-CIPN-list dataset before splitting into sub-records). sMMR (rMMR) refers to strict (ratio) mean match ratio, and sMAP (rMAP) refers to strict (ratio) mean average precision.

Method	No. sents	sMAP	rMAP	sMMR	rMMR
TF-IDF	5	42.2	54.5	64.3	70.3
	10	40.7	52.4	74.7	80.4
	20	38.0	48.8	84.7	88.1
	30	36.3	46.8	88.7	91.5
	40	35.0	45.5	91.5	93.8
	50	33.9	44.4	93.4	95.3
	60	33.3	43.7	94.8	96.6
	70	32.7	43.1	95.5	97.0
	80	32.2	42.6	96.5	97.6
BM25	5	40.3	51.9	61.9	68.0
	10	38.6	49.6	72.7	78.4
	20	36.4	46.7	83.4	87.2
	30	34.7	44.8	87.8	90.7
	40	33.5	43.4	91.3	93.5
	50	32.5	42.4	93.1	95.1
	60	31.8	41.7	94.4	96.1
	70	31.1	41.0	95.1	96.6
	80	30.6	40.5	95.6	96.9
SBERT	5	47.8	60.5	67.5	73.3
	10	45.5	57.7	77.2	81.9
	20	42.3	54.0	86.6	89.9
	30	40.0	51.6	90.7	93.4
	40	38.5	50.1	93.3	95.2
	50	37.3	48.8	95.3	96.7
	60	36.3	47.7	96.2	97.5
	70	35.5	46.8	96.9	98.1
	80	34.8	46.1	97.1	98.3

Appendix Table 12: Effect of optimizer, learning rate scheduler and freezing embedding on overall PICO entity recognition performance of CRF, BiLSTM and BiLSTM-CRF.

Model	Optimizer & Scheduler	F1	Recall	Precision
CRF	Adam	43.0	38.8	48.2
	Adam + Freeze	46.2	40.7	53.4
	Adam + STLR	42.5	40.2	45.1
	Adam + STLR + Freeze	44.7	38.8	52.6
	AdamW	41.6	39.9	43.6
	AdamW + Freeze	39.4	31.4	53.1
	AdamW + STLR	40.5	40.6	40.5
	AdamW + STLR + Freeze	37.3	28.4	54.1
BiLSTM	Adam	41.7	44.2	39.5
	Adam + Freeze	40.8	41.9	39.8
	Adam + STLR	39.4	40.4	38.4
	Adam + STLR + Freeze	36.7	38.1	35.4
	AdamW	29.6	26.2	34.2
	AdamW + Freeze	26.8	22.7	32.8
	AdamW + STLR	26.1	24.4	28.0
	AdamW + STLR + Freeze	23.9	19.4	31.2
BiLSTM-CRF	Adam	42.9	42.5	43.3
	Adam + Freeze	57.7	53.7	62.3
	Adam + STLR	46.6	46.1	47.1
	Adam + STLR + Freeze	51.1	48.5	54.0
	AdamW	36.4	35.9	36.8
	AdamW + Freeze	50.8	46.3	56.2
	AdamW + STLR	45.3	44.2	46.5
	AdamW + STLR + Freeze	50.0	48.5	51.7

Appendix Table 13: Performance of BERT (pre-trained on PubMed abstracts and full texts) for PICO entity recognition using self-training with thresholds 0.99 and 0.95. Scores at iteration 0 refer to the performance of the original model before using self-training.

Threshold	Iteration	gold valid			gold test			
		f1	rec	prec	f1	rec	prec	train size
	0	68.1	73.0	63.8	69.9	73.4	66.7	320
0.99	1	63.1	68.6	58.3	67.2	70.3	64.3	324
0.95	1	70.5	75.4	66.2	70.9	75.3	66.9	1845
	2	72.4	75.7	69.3	70.5	74.6	66.9	5385
	3	71.5	75.0	68.3	70.5	72.9	68.3	5760
	4	72.1	74.9	69.5	70.9	72.9	69.0	6533
	5	72.4	75.6	69.5	69.5	72.1	67.1	6626
	6	73.6	76.4	71.0	71.0	74.0	68.2	6757
	7	70.3	74.2	66.9	69.4	72.6	66.4	6921
	8	70.7	74.7	67.1	69.6	74.0	65.6	7053
	9	71.7	75.6	68.2	69.1	73.4	65.4	7089
	10	72.0	74.5	69.6	71.5	74.1	69.1	7115
	11	72.6	75.6	69.9	69.6	72.7	66.6	7139
	12	71.5	75.6	67.8	70.7	74.3	67.4	7163
	13	72.1	74.9	69.5	69.6	72.1	67.2	7167
	14	71.7	75.6	68.1	70.3	73.8	67.1	7227
	15	71.4	75.0	68.1	70.6	74.0	67.5	7328

Bibliography

Abiodun,O.I. *et al.* (2018) State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4, e00938.

Achakulvisut,T. *et al.* (2020) Pubmed Parser: A Python Parser for PubMed Open-Access XML Subset and MEDLINE XML Dataset XML Dataset. *J. Open Source Softw.*, 5, 1979.

Alammar,J. (2018) The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning) – Jay Alammar – Visualizing machine learning one concept at a time.

Andrew,G. and Gao,J. (2007) Scalable training of L1-regularized log-linear models. In, *ACM International Conference Proceeding Series*., pp. 33–40.

Aronson,A.R. and Lang,F.M. (2010) An overview of MetaMap: Historical perspective and recent advances. *J. Am. Med. Informatics Assoc.*, 17, 229–236.

Bahdanau,D. *et al.* (2015) Neural machine translation by jointly learning to align and translate. In, *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR.

Bahor,Z. *et al.* (2016) Improving our understanding of the in vivo modelling of psychotic disorders: A protocol for a systematic review and meta-analysis . *Evidence-based Preclin. Med.*, 3, e00022.

Bahor,Z. *et al.* (2017) Risk of bias reporting in the recent animal focal cerebral ischaemia literature. *Clin. Sci.*, 131, 2525 LP – 2532.

Bahor,Z. *et al.* (2021) Development and uptake of an online systematic review platform: The early years of the CAMARADES Systematic Review Facility (SyRF). *BMJ Open Sci.*, **5**, 100103.

Bajaj,P. *et al.* (2016) MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *CEUR Workshop Proc.*, 1773.

Bastian,H. *et al.* (2010) Seventy-five trials and eleven systematic reviews a day: How will we ever keep up? *PLoS Med.*, 7.

Beltagy,I. *et al.* (2020) Longformer: The Long-Document Transformer.

Beltagy,I. *et al.* (2019) SCIBERT: A Pretrained Language Model for Scientific Text.

Bodenreider,O. (2004) The Unified Medical Language System (UMLS): Integrating biomedical terminology. *Nucleic Acids Res.*, 32.

Booth,A. *et al.* (2000) Structuring the pre-search reference interview: A useful technique for handling clinical questions. *Bull. Med. Libr. Assoc.*, 88, 239–247.

Borah,R. *et al.* (2017) Analysis of the time and workers needed to conduct systematic reviews of medical interventions using data from the PROSPERO registry. *BMJ Open*, 7, e012545.

Bottou,L. (2012) Stochastic gradient descent tricks. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, 7700 LECTU, 421–436.

Boudin,F. *et al.* (2010) Improving medical information retrieval with PICO element detection. In, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer Verlag, pp. 50–61.

Bowman,S.R. *et al.* (2015) A large annotated corpus for learning natural language inference. *Conf. Proc. - EMNLP 2015 Conf. Empir. Methods Nat. Lang. Process.*, 632–642.

Breiman,L. (2001) Random forests. *Mach. Learn.*, 45, 5–32.

Brockmeier,A.J. *et al.* (2019) Improving reference prioritisation with PICO recognition. *BMC Med. Inform. Decis. Mak.*, 19, 256.

Cao,Y.G. *et al.* (2011) AskHERMES: An online question answering system for complex clinical questions. *J. Biomed. Inform.*, 44, 277–288.

Cer,D. *et al.* (2017) SemEval-2017 Task 1: Semantic Textual Similarity - Multilingual and Cross-lingual Focused Evaluation. *arXiv*.

Chabou,S. and Iglewski,M. (2018) Combination of conditional random field with a rule based method in the extraction of PICO elements. *BMC Med. Inform. Decis. Mak.*, 18, 128.

Chen,D. *et al.* (2017) Reading Wikipedia to Answer Open-Domain Questions. *ACL 2017 - 55th Annu. Meet. Assoc. Comput. Linguist. Proc. Conf. (Long Pap.*, 1, 1870–1879.

Chollet,F. (2017) Xception: Deep learning with depthwise separable convolutions. In, *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*. Institute of Electrical and Electronics Engineers Inc., pp. 1800–1807.

Chung,J. *et al.* (2014) Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling.

Cipriani,A. and Geddes,J. (2003) Comparison of systematic and narrative reviews: the example of the atypical antipsychotics. *Epidemiol. Psychiatr. Sci.*, 12, 146–153.

Collins,M. (2011) Log-Linear Models , MEMMs , and CRFs.

Currie,G.L. *et al.* (2019) Animal models of chemotherapy-induced peripheral neuropathy: A machine-assisted systematic review and meta-analysis. *PLOS Biol.*, 17, e3000243.

Dai,Z. *et al.* (2019) Transformer-XL: Attentive Language Models Beyond a

Fixed-Length Context. 2978–2988.

Daniel,J. and Martin,J.H. (2020) Speech and Language Processing: Vector Semantics and Embeddings. In, *Speech and Language Processing*.

Demner-Fushman,D. and Lin,J. (2005) Knowledge Extraction for Clinical Question Answering: Preliminary results. In, *AAAI Workshop - Technical Report*., pp. 1–9.

Devlin,J. *et al.* (2018) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*.

DeYoung,J. *et al.* (2020) Evidence Inference 2.0: More Data, Better Models.

Dumoulin,V. and Visin,F. (2016) A guide to convolution arithmetic for deep learning.

Elliott,J.H. *et al.* (2017) Living systematic review: 1. Introduction—the why, what, when, and how. *J. Clin. Epidemiol.*, 91, 23–30.

van Engelen,J.E. and Hoos,H.H. (2020) A survey on semi-supervised learning. *Mach. Learn.*, 109, 373–440.

Fernández,A. *et al.* (2018) Learning from Imbalanced Data Sets Springer International Publishing.

Finkel,J.R. *et al.* (2005) Incorporating non-local information into information extraction systems by Gibbs sampling. In, *ACL-05 - 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*. Association for Computational Linguistics (ACL), pp. 363–370.

Fontelo,P. and Liu,F. (2018) A review of recent publication trends from top publishing countries. *Syst. Rev. 2018 71*, 7, 1–9.

Forney,G.D. (1973) The Viterbi Algorithm. *Proc. IEEE*, 61, 268–278.

Francois Chaubard, Michael Fang, Guillaume Genthial, Rohit Mundra,R.S. (2019) CS224n: Natural Language Processing with Deep Learning, Lecture Notes: Part I.

Fu,J. *et al.* (2020) Interpretable multi-dataset evaluation for named entity recognition. *EMNLP 2020 - 2020 Conf. Empir. Methods Nat. Lang. Process. Proc. Conf.*, 6058–6069.

Gao,S. *et al.* (2021) A pre-training and self-training approach for biomedical named entity recognition. *PLoS One*, 16.

Goldberg,Y. (2016) A primer on neural network models for natural language processing. *J. Artif. Intell. Res.*, 57, 345–420.

Goldberg,Y. (2017) Neural Network Methods for Natural Language Processing. *Synth. Lect. Hum. Lang. Technol.*, 10, 1–311.

Goodfellow,I. *et al.* (2016) Deep Learning MIT Press.

Gu,Y. *et al.* (2020) Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing.

Hair,K. *et al.* (2019) A randomised controlled trial of an Intervention to Improve Compliance with the ARRIVE guidelines (IICARus). *Res. Integr. Peer Rev.*, 4, 12.

Hastie, Trevor, Tibshirani, Robert, Friedman,J. (2009) The Elements of Statistical Learning Springer New York, New York, NY.

He,K. *et al.* (2015) Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In, *Proceedings of the IEEE International Conference on Computer Vision*., pp. 1026–1034.

Higgins,Julian P. T. *et al.* (2011) Cochrane Handbook for Systematic Reviews of Interventions Version 5.1.0 [updated March 2011].

Higgins,Julian P.T. *et al.* (2011) The Cochrane Collaboration’s tool for assessing risk of bias in randomised trials. *BMJ*, 343.

Hiroki Nakayama (2018) seqeval: A Python framework for sequence labeling evaluation.

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

Honek,J. (2017) Preclinical Research in Drug Development. *Med. Writ.*, 26, 5–8.

Hooijmans,C.R. *et al.* (2018) Facilitating healthcare decisions by assessing the certainty in the evidence from preclinical animal studies. *PLoS One*, 13.

Hooijmans,C.R. *et al.* (2014) SYRCLE's risk of bias tool for animal studies. *BMC Med. Res. Methodol.*, 14, 43.

Howard,J. and Ruder,S. (2018) Universal language model fine-tuning for text classification. In, *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers).*, pp. 328–339.

Huang,X. *et al.* (2006) Evaluation of PICO as a knowledge representation for clinical questions. *AMIA Annu. Symp. Proc.*, 2006, 359–363.

Ioannidis,J.P.A. *et al.* (2014) Increasing value and reducing waste in research design, conduct, and analysis. *Lancet*, 383, 166–175.

Ioffe,S. and Szegedy,C. (2015) Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

Jin,D. and Szolovits,P. (2018a) Advancing PICO Element Detection in Biomedical Text via Deep Neural Networks. *Bioinformatics*, 36, 3856–3862.

Jin,D. and Szolovits,P. (2018b) PICO Element Detection in Medical Text via Long Short-Term Memory Neural Networks. In, *Proceedings of the*

BioNLP 2018 workshop. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 67–75.

Jin,Q. *et al.* (2020) Predicting Clinical Trial Results by Implicit Evidence Integration. *arXiv*.

Kaiser,L. *et al.* (2017) Depthwise Separable Convolutions for Neural Machine Translation. *arXiv*.

Kazaryan,A. *et al.* (2020) Transformer-Based Open Domain Biomedical Question Answering at BioASQ8 Challenge. *CLEF 2020 Work. Notes*, 22–25.

Kilkenny,C. *et al.* (2010) Improving Bioscience Research Reporting: The ARRIVE Guidelines for Reporting Animal Research. *PLOS Biol.*, 8, e1000412.

Kim,Y. (2014) Convolutional neural networks for sentence classification. In, *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*.

Kingma,D.P. and Ba,J.L. (2015) Adam: A method for stochastic optimization. In, *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR.

Kiritchenko,S. *et al.* (2010) ExaCT: Automatic extraction of clinical trial characteristics from journal publications. *BMC Med. Inform. Decis. Mak.*, 10.

Krallinger,M. *et al.* (2020) BioASQ at CLEF2020: Large-scale biomedical semantic indexing and question answering. In, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer, pp. 550–556.

Krizhevsky,A. *et al.* (2012) ImageNet Classification with Deep Convolutional

Neural Networks.

Lample,G. *et al.* (2016) Neural Architectures for Named Entity Recognition. *2016 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. NAACL HLT 2016 - Proc. Conf.*, 260–270.

Landis,S.C. *et al.* (2012) A call for transparent reporting to optimize the predictive value of preclinical research. *Nature*, 490, 187–191.

Le,Q. and Mikolov,T. (2014) Distributed representations of sentences and documents. In, *31st International Conference on Machine Learning, ICML 2014.*, pp. 2931–2939.

Lee,J. *et al.* (2019) BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Lee,M. *et al.* (2006) Beyond information retrieval--medical question answering. *AMIA Annu. Symp. Proc.*, 2006, 469–473.

Lehman,E. *et al.* (2019) Inferring which medical treatments work from reports of clinical trials. In, *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*. Association for Computational Linguistics (ACL), pp. 3705–3717.

Lemaréchal,C. and Lemaréchal,C. (2012) Cauchy and the Gradient Method. *Doc. MATH.* .

Liao,J. *et al.* (2018) Automation of citation screening in pre-clinical systematic reviews. *bioRxiv*, 280131.

Loshchilov,I. and Hutter,F. (2017) Decoupled Weight Decay Regularization. *7th Int. Conf. Learn. Represent. ICLR 2019*.

Van Der Maaten,L. and Hinton,G. (2008) Visualizing Data using t-SNE.

Macleod,M. and Wong,C. (2019) ReLiSyR MND.

Macleod,M.R. *et al.* (2014) Biomedical research: increasing value, reducing waste. *Lancet*, 383, 101–104.

Macleod,M.R. *et al.* (2004) Pooling of animal experimental data reveals influence of study design and publication bias. *Stroke*, 35, 1203–1208.

Macleod,M.R. *et al.* (2015) Risk of Bias in Reports of In Vivo Research: A Focus for Improvement. *PLOS Biol.*, 13, e1002273.

MacLeod,M.R. *et al.* (2008) Evidence for the efficacy of NXY-059 in experimental focal cerebral ischaemia is confounded by study quality. *Stroke*, 39, 2824–2829.

Manning,C.D. *et al.* (2008) Introduction to Information Retrieval.

Marshall,I.J. *et al.* (2015) Automating Risk of Bias Assessment for Clinical Trials. *IEEE J. Biomed. Heal. Informatics*, 19, 1406–1412.

Marshall,I.J. *et al.* (2016) RobotReviewer: evaluation of a system for automatically assessing bias in clinical trials. *J. Am. Med. Informatics Assoc.*, 23, 193–201.

Marshall,I.J. *et al.* (2020) Trialstreamer: A living, automatically updated database of clinical trial reports. *J. Am. Med. Inform. Assoc.*, 00, 1–10.

Marshall,I.J. and Wallace,B.C. (2019) Toward systematic review automation: A practical guide to using machine learning tools in research synthesis. *Syst. Rev.*, 8, 163.

Masic,I. *et al.* (2008) Evidence Based Medicine – New Approaches and Challenges. *Acta Inform. Medica*, 16, 219.

Masters,D. and Luschi,C. (2018) Revisiting Small Batch Training for Deep Neural Networks. *arXiv*.

McCann,S.K. *et al.* (2016) Systematic Review and Meta-Analysis of the Efficacy of Interleukin-1 Receptor Antagonist in Animal Models of Stroke:

an Update. *Transl. Stroke Res.*, 7, 395–406.

Menke,J. *et al.* (2020) The Rigor and Transparency Index Quality Metric for Assessing Biological and Medical Science Methods. *iScience*, 23.

Mikolov,T., Sutskever,I., *et al.* (2013) Distributed Representations of Words and Phrases and their Compositionality. *Adv. Neural Inf. Process. Syst.*

Mikolov,T., Chen,K., *et al.* (2013) Efficient Estimation of Word Representations in Vector Space.

Millard,L.A. *et al.* (2016) Machine learning to assist risk-of-bias assessments in systematic reviews. *Int. J. Epidemiol.*, 45, 266–277.

Mintz,M. *et al.* (2009) Distant Supervision for Relation Extraction Without Labeled Data. In, *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 1003–1011.

Moher,D. *et al.* (2016) Increasing value and reducing waste in biomedical research: Who's listening? *Lancet*, 387, 1573–1586.

Muhlhausler,B.S. *et al.* (2013) Whole Animal Experiments Should Be More Like Human Randomized Controlled Trials. *PLoS Biol.*, 11.

Mulyar,A. *et al.* (2019) Phenotyping of Clinical Notes with Improved Document Classification Models Using Contextualized Neural Language Models. *arXiv*.

Neumann,M. *et al.* (2019) ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. Association for Computational Linguistics (ACL), pp. 319–327.

Neves,M. *et al.* (2017) Olelo: A Question Answering Application for

Biomedicine. 61–66.

Niu,Y. *et al.* (2003) Answering clinical questions with role identification. Association for Computational Linguistics (ACL), pp. 73–80.

Nocedal,J. (1980) Updating Quasi-Newton Matrices with Limited Storage. *Math. Comput.*, 35, 773.

Nogueira,R. and Cho,K. (2019) Passage Re-ranking with BERT.

Nordberg,M. *et al.* (2004) Glossary of terms used in toxicokinetics (IUPAC Recommendations 2003). *Pure Appl. Chem.*, 76, 1033–1082.

Nye,B. *et al.* (2018) A corpus with multi-level annotations of patients, interventions and outcomes to support language processing for medical literature. In, *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers).*, pp. 197–207.

Nye,B.E. *et al.* (2020) Understanding Clinical Trial Reports: Extracting Medical Entities and Their Relations.

O'Collins,V.E. *et al.* (2006) 1,026 Experimental treatments in acute stroke. *Ann. Neurol.*, 59, 467–477.

Pae,C.-U. (2015) Why Systematic Review rather than Narrative Review? *Psychiatry Investig.*, 12, 417.

Pappas,D. *et al.* (2020) AUEB at BioASQ 7: Document and Snippet Retrieval. In, *Communications in Computer and Information Science*. Springer, pp. 607–623.

Pascanu,R. *et al.* (2012) On the difficulty of training Recurrent Neural Networks. *30th Int. Conf. Mach. Learn. ICML 2013*, 2347–2355.

Paszke,A. *et al.* (2019) PyTorch: An Imperative Style, High-Performance Deep Learning Library. *arXiv*.

Pedregosa,F. *et al.* (2011) Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.*, 12, 2825–2830.

Perozzi,B. *et al.* (2014) DeepWalk: Online Learning of Social Representations. *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 701–710.

Peters,M. *et al.* (2018) Deep Contextualized Word Representations., pp. 2227–2237.

Pound,P. and Ritskes-Hoitinga,M. (2020) Can prospective systematic reviews of animal studies improve clinical translation? *J. Transl. Med. 2020 181*, 18, 1–6.

Prager,J. (2006) Open-domain question-answering. *Found. Trends Inf. Retr.*, 1, 91–233.

Prinz,F. *et al.* (2011) Believe it or not: how much can we rely on published data on potential drug targets? *Nat. Rev. Drug Discov. 2011 109*, 10, 712–712.

Pyysalo,S. *et al.* (2013) Distributional Semantics Resources for Biomedical Text Processing. *Proc. 5th Lang. Biol. Med. Conf. (LBM 2013)*, 39–44.

Rajpurkar,P. *et al.* (2016) SQuAD: 100,000+ Questions for Machine Comprehension of Text.

Ramshaw,L.A. and Marcus,M.P. (1995) Text Chunking using Transformation-Based Learning. 157–176.

Raschka,S. (2018) Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning. *arXiv*.

Rehurek,R. and Sojka,P. (2010) Software Framework for Topic Modelling with Large Corpora. *undefined*.

Reimers,N. and Gurevych,I. (2019) Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *EMNLP-IJCNLP 2019 - 2019 Conf. Empir. Methods Nat. Lang. Process. 9th Int. Jt. Conf. Nat. Lang. Process.*

Proc. Conf., 3982–3992.

Richardson,W.S. *et al.* (1995) The well-built clinical question: a key to evidence-based decisions. *ACP J. Club*, 123.

Robertson,S. and Zaragoza,H. (2009) The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3, 333–389.

Ruder,S. (2017) An Overview of Multi-Task Learning in Deep Neural Networks.

Ruder,S. and Plank,B. (2018) Strong Baselines for Neural Semi-supervised Learning under Domain Shift. *ACL 2018 - 56th Annu. Meet. Assoc. Comput. Linguist. Proc. Conf. (Long Pap.*, 1, 1044–1054.

Sackett,D.L. *et al.* (1996) Evidence based medicine: what it is and what it isn't. 1996. *BMJ*, 312.

Sanh,V. *et al.* (2019) DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.

Sarrouti,M. and Ouatik El Alaoui,S. (2020) SemBioNLQA: A semantic biomedical question answering system for retrieving exact and ideal answers to natural language questions. *Artif. Intell. Med.*, 102, 101767.

Schardt,C. *et al.* (2007) Utilization of the PICO framework to improve searching PubMed for clinical questions. *BMC Med. Inform. Decis. Mak.*, 7, 1–6.

Schmidt,L. *et al.* (2020) Data Mining in Clinical Trial Text: Transformers for Classification and Question Answering Tasks. *Heal. 2020 - 13th Int. Conf. Heal. Informatics, Proceedings; Part 13th Int. Jt. Conf. Biomed. Eng. Syst. Technol. BIOSTEC 2020*, 83–94.

Scott,S. *et al.* (2008) Design, power, and interpretation of studies in the standard murine model of ALS. *Amyotroph. Lateral Scler.*, 9, 4–15.

Semnani,S.J. and Pandey,M. (2020) Revisiting the open-domain question answering pipeline. In, *arXiv*.

Sena,E.S. *et al.* (2014) Systematic reviews and meta-analysis of preclinical studies: Why perform them and how to appraise them critically. *J. Cereb. Blood Flow Metab.*, 34, 737–742.

Seo,M. *et al.* (2016) Bidirectional Attention Flow for Machine Comprehension.

Shegokar,R. (2020) Preclinical testing—Understanding the basics first. In, *Drug Delivery Aspects*. Elsevier, pp. 19–32.

Shemilt,I. *et al.* (2016) Use of cost-effectiveness analysis to compare the efficiency of study identification methods in systematic reviews. *Syst. Rev. 2016 51*, 5, 1–13.

Shojania,K.G. *et al.* (2007) How quickly do systematic reviews go out of date? A survival analysis. *Ann. Intern. Med.*, 147, 224–233.

Soboczenski,F. *et al.* (2019) Machine learning to help researchers evaluate biases in clinical trials: A prospective, randomized user study. *BMC Med. Inform. Decis. Mak.*, 19, 96.

Srivastava,N. *et al.* (2014) Dropout: A Simple Way to Prevent Neural Networks from Overfitting.

van der Staay,F.J. *et al.* (2009) Evaluation of animal models of neurobehavioral disorders. *Behav. Brain Funct.*, 5, 1–23.

Sutton,C. and McCallum,A. (2011) An introduction to conditional random fields. *Found. Trends Mach. Learn.*, 4, 267–373.

Szajewska,H. (2018) Evidence-Based Medicine and Clinical Research: Both Are Needed, Neither Is Perfect. *Ann. Nutr. Metab.*, 72, 13–23.

Taylor,K. *et al.* (2008) Estimates for worldwide laboratory animal use in 2005. *ATLA Altern. to Lab. Anim.*, 36, 327–342.

The NPQIP Collaborative Group (2019) Did a change in Nature journals' editorial policy for life sciences research improve reporting? *BMJ Open*

Sci., 3, e000035.

Thomas,J. *et al.* (2017) Living systematic reviews: 2. Combining human and machine effort. *J. Clin. Epidemiol.*, 91, 31–37.

Tsafnat,G. *et al.* (2018) Automated screening of research studies for systematic reviews using study characteristics. *Syst. Rev. 2018 71*, 7, 1–9.

Uman,L.S. (2011) Systematic Reviews and Meta-Analyses. *J. Can. Acad. Child Adolesc. Psychiatry*, 20, 57.

Vaswani,A. *et al.* (2017) Attention is all you need. In, *Advances in Neural Information Processing Systems*., pp. 5999–6009.

Voorhees,E.M. (2001) Question answering in TREC. In, *International Conference on Information and Knowledge Management, Proceedings*. Association for Computing Machinery (ACM), pp. 535–537.

de Vries,R.B.M. *et al.* (2014) The usefulness of systematic reviews of animal experiments for the design of preclinical and clinical studies. *ILAR J.*, 55, 427–437.

Wallace,B.C. *et al.* (2016) Extracting PICO Sentences from Clinical Trial Reports using Supervised Distant Supervision. *J. Mach. Learn. Res.*, 17.

Wang,D. and Nyberg,E. (2015) A long short-term memory model for answer sentence selection in question answering. In, *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference*. Association for Computational Linguistics (ACL), pp. 707–712.

Williams,A. *et al.* (2017) A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. *NAACL HLT 2018 - 2018 Conf. North*

Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf., 1, 1112–1122.

Wolf,T. *et al.* (2019) HuggingFace's Transformers: State-of-the-art Natural Language Processing.

Worp,H.B. van der *et al.* (2010) Can Animal Models of Disease Reliably Inform Human Studies? *PLOS Med.*, 7, e1000245.

Van Der Worp,H.B. *et al.* (2007) Hypothermia in animal models of acute ischaemic stroke: A systematic review and meta-analysis. *Brain*, 130, 3063–3074.

Wouters,O.J. *et al.* (2020) Estimated Research and Development Investment Needed to Bring a New Medicine to Market, 2009-2018. *JAMA*, 323, 844–853.

Wu,Y. *et al.* (2016) Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation.

Yang,Z. *et al.* (2016) Hierarchical Attention Networks for Document Classification. In, *HLT-NAACL*.

Yoon,W. *et al.* (2019) Pre-trained Language Model for Biomedical Question Answering. *Commun. Comput. Inf. Sci.*, 1168 CCIS, 727–740.

Yu,A.W. *et al.* (2018) QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. *6th Int. Conf. Learn. Represent. ICLR 2018 - Conf. Track Proc.*

Zhang,J. *et al.* (2019) Why gradient clipping accelerates training: A theoretical justification for adaptivity.

Zhang,T. *et al.* (2020) Unlocking the Power of Deep PICO Extraction: Step-wise Medical NER Identification. *arXiv*.

Zhang,Y. *et al.* (2016) Rationale-Augmented Convolutional Neural Networks for Text Classification. *CoRR*, abs/1605.0.