

**Structuring the Unstructured:
Unlocking pharmacokinetic data from
journals with Natural Language
Processing**

by

Ferran Gonzalez Hernandez

A dissertation submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

University College London

September 2022

Supervision: Joseph Standing, Frank Klopogge, Watjana
Lilaonitkul, Juha Iso-Sipilä

I, Ferran Gonzalez Hernandez , confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

The development of a new drug is an increasingly expensive and inefficient process. Many drug candidates are discarded due to pharmacokinetic (PK) complications detected at clinical phases. It is critical to accurately estimate the PK parameters of new drugs before being tested in humans since they will determine their efficacy and safety outcomes. Preclinical predictions of PK parameters are largely based on prior knowledge from other compounds, but much of this potentially valuable data is currently locked in the format of scientific papers. With an ever-increasing amount of scientific literature, automated systems are essential to exploit this resource efficiently. Developing text mining systems that can structure PK literature is critical to improving the drug development pipeline.

This thesis studied the development and application of text mining resources to accelerate the curation of PK databases. Specifically, the development of novel corpora and suitable natural language processing architectures in the PK domain were addressed. The work presented focused on machine learning approaches that can model the high diversity of PK studies, parameter mentions, numerical measurements, units, and contextual information reported across the literature. Additionally, architectures and training approaches that could efficiently deal with the scarcity of annotated examples were explored. The chapters of this thesis tackle the development of suitable models and corpora to (1) retrieve PK documents, (2) recognise PK parameter mentions, (3) link PK entities to a knowledge base and (4) extract relations between parameter mentions, estimated measurements, units and other contextual information. Finally, the last chapter of this thesis studied the feasibility of the whole extraction pipeline to accelerate tasks in drug development research.

The results from this thesis exhibited the potential of text mining approaches to automatically generate PK databases that can aid researchers in the field and ultimately accelerate the drug development pipeline. Additionally, the thesis presented contributions to biomedical natural language processing by developing suitable architectures and corpora for multiple tasks, tackling novel entities and relations within the PK domain.

Impact Statement

This PhD thesis presents research in the intersection of drug development and Natural Language Processing (NLP), and it aims to enhance the development of text mining systems that can accelerate the curation of datasets used across the drug development pipeline. Learnings and findings from this PhD have a potential impact on developing new medicines and systems that structure the biomedical text.

Systems that can automatically extract evidence across the scientific literature become increasingly important as the amount of biomedical literature expands. Such systems can aid biomedical researchers in finding relevant information and accelerate the discovery of new associations. Specifically, this thesis contributed to developing new text mining systems that can centralise pharmacokinetic (PK) measurements reported across the literature to automatise the construction of chemical databases critical for drug development. In addition, the thesis developed novel corpora and NLP architectures in the PK domain.

To put the findings of this thesis to beneficial use, they were presented to the academic and pharmaceutical communities in different forms. The first chapter of this thesis was published as an article at Wellcome Open Research, describing an algorithm to accelerate the identification of publications reporting pharmacokinetic (PK) parameters. Additionally, a search engine was deployed to facilitate the finding of PK information by researchers in academia and industry. Finally, the code and data developed were publicly released, and they could be used to evaluate and develop novel text mining systems in this domain. Results from the following chapters of this thesis were disseminated as oral and poster presentations in the UK Quantitative Systems Pharmacology Network 2019 meeting, the Pharmacokinetics UK 2019 conference and the World Conference on Pharmacometrics 2020 and 2022.

A database with a large amount of pharmacokinetic evidence was generated in the last chapter of this thesis. In addition, a search interface has been built to aid researchers in preclinical and clinical drug development perform better predictions of novel compounds. The information extraction pipeline and database built in this work can enhance reproducible research practices in PK and provide a structured and centralised resource that could enable machine learning approaches to have a more significant role in drug development.

Acknowledgements

First and foremost, I am deeply thankful to my supervisory team. To Joe, for introducing me to this project, for his continuous support, trust, guidance, and for being a role model throughout this journey. To Frank, for the unconditional support, for helping me with endless doubts, introducing me to many people, believing and helping so much in this project, and for all the fun together. To Waty, for encouraging me to always aim higher, for the support in each step, for all the hours spent in discussion, planning, coding, for her energy, for going far beyond what was required, for being such an inspiration throughout these years. Finally, to Juha, for getting involved in this project in the first place, for all the meetings together, for the technical advice, the opportunities, and for being the best example of a team leader and engineer. I feel incredibly fortunate to have had each of you as supervisors.

I will be forever grateful to the Alan Turing Institute for the Enrichment Scheme. During twelve months, I had the privilege to meet and work with unique colleagues and friends who brought me much inspiration and from whom I have learned a lot: Bea, James, Pedro, Sara, Tuğçe, Danielle, Victoria and Alessandro. I am also thankful to Mark and George for introducing me to the Defence and Security programme and collaborating on a fantastic project. Finally, I would like to thank Nigel for being my fellow supervisor and introducing me to the NLP group in Cambridge.

This project would not have been the same without my time at BenevolentAI. I consider myself lucky to have met and worked alongside many brilliant researchers and engineers. I would like to especially thank Juha, Mikko, Maciek, Felix, Jonas, Roxana, Sia, and the rest of the knowledge team colleagues for so much inspiration, fun, and teamwork. This was an inflexion point for my development as a researcher and professional. Finally, I would like to extend my gratitude to Jane, Abhishek and the rest of the DMPK researchers who made the internship project possible through annotations and insightful discussions on the potential applications of this project in drug development.

I would like to thank my thesis examiners, Sophia and Lewis for an engaging discussion during the exam and all suggestions and feedback provided. I am thankful to Pontus and Mario for their insightful discussions and engagement during the upgrade examination of this thesis and the posterior feedback and direction for the project. I am grateful to my previous advisors from Universitat Politècnica de Catalunya, Dani and Clara, for teaching me and preparing me for this research. Finally, I am thankful to the CoMPLEX centre at UCL for providing me with a wealth of training and research opportunities at the intersection of mathematics, computer science and biomedicine. I would like to especially thank my colleagues Dionna, Raquel, Hassan, Kate and Valeria.

This project was only possible thanks to extensive collaborations with pharmacometricians who engaged and believed in this project. First, I would like to thank Simon for a year of insightful discussions, feedback and endless annotations. Silke, for the multiple annotations and discussions on potential applications of this project. Paul, for making the project grow and expand across the industry. Ahmed, for voluntarily getting involved in this project and helping its development. Gill, for the insightful feedback and encouragement on the potential applications of this project within DMPK teams and the annotations provided. Palang and Thanaporn for voluntarily collaborating on the project and all the time spent on it. Finally, I am deeply thankful to the researchers in pharmacometrics from Universitat Ramon Llull, who made the relation extraction project possible. Especial thanks to Jose Antonio, Maria Rosa, Mario and Albert for all the annotations, the discussions, and the great time collaborating on this project.

The PhD would not have been as fun without the colleagues at ICH. I am especially thankful to Soumya for sharing the journey since the first day and supporting each other along the way. I am grateful to Lily and Kerry for all the fun during the first years. To Carlota and Angel for becoming the best bubble during the lockdown period and sharing so much fun time. I am incredibly thankful to Vicky for sharing so many hours on google meets, coding together, discussing algorithms, climbing and becoming a fantastic colleague. I am very thankful to have met Quang for all the insightful discussions on NLP and CV approaches, the immense help implementing NER algorithms and the fantastic Vietnamese coffee!

I am deeply thankful to the Graduate Research Scholarship from UCL, which allowed me to come all the way from Barcelona and provided me with financial support during the master's and PhD, making all this possible.

I want to thank Liam for all the time spent together, for keeping me sane throughout this PhD journey and for becoming my family away from home. Jina, thank you for all the energy, support and inspiration along the way - I feel very fortunate to have met you during my PhD. Jacob and Bella, for sharing so much time in Streatham and making the lockdown period much more fun. I am very grateful to have met Laurence, for all the coffees, evenings playing chess and dancing around Streatham.

I want to express my deepest gratitude to my parents and family for their unconditional support and encouragement throughout this PhD journey, and without whom, this would not have been possible. Finally, I would like to thank Júlia for helping me in each stage of this process, encouraging me throughout the journey, doing all this together, and being the best teammate I could have imagined.

Dedicat al meu avi, Francesc

Contents

Abstract	ii
Impact Statement	iii
Acknowledgements	v
1 Introduction and Background	12
1.1 Thesis motivation	12
1.2 Quantitative pharmacology	13
1.2.1 Modelling approaches	14
1.2.2 Drug development	14
1.2.3 Pharmacokinetic/Pharmacodynamic modelling	16
1.2.3.1 Pharmacokinetic parameters	16
1.2.4 Pharmacokinetic preclinical prediction	20
1.3 Text mining	22
1.3.1 Underlying techniques	22
1.3.2 Biomedical text mining	24
1.3.2.1 Resources	24
1.3.2.2 Tasks, supporting tools and evaluation metrics	27
1.3.3 Related work and research gap	32
1.3.3.1 Research gap	34
1.4 Research aim, questions and thesis outline	36
2 Models in Natural Language Processing	38
2.1 Context-independent	38
2.1.1 Bag-of-Words	38
2.1.2 TF-IDF	39
2.2 Distributional	40
2.2.1 Count-based	40
Co-occurrence Counts	40
Positive Pointwise Mutual Information (PPMI)	40
Latent Semantic Analysis (LSA)	41
2.2.2 Prediction-based	41
2.2.2.1 Feed-forward neural networks	41
2.2.2.2 Word2Vec	43
2.2.2.3 Recurrent models	45
2.2.2.4 Transformers	47
2.2.2.5 BERT	48
2.3 Non-neural models	50
3 Pharmacokinetic Document Retrieval	55
3.1 Introduction	55
3.2 Methods	57
3.2.1 Corpus development	57

3.2.1.1	Source	57
3.2.1.2	Size and criteria	58
3.2.1.3	Annotation	58
3.2.2	Pipeline development	59
3.2.2.1	Parsing	59
3.2.2.2	Evaluation	60
3.2.2.3	Classifier comparison	62
	Extreme Gradient Boosting	62
3.2.2.4	Field-selection	63
3.2.2.5	N-grams	65
3.2.2.6	Distributed representations	65
	BioBERT	66
	SPECTER	66
3.2.2.7	Final pipeline	67
3.2.3	Large-scale application	67
3.3	Results and Discussion	68
3.3.1	Feature visualisation	68
3.3.2	Inter-annotator agreement	68
3.3.3	Classifier comparison	69
3.3.4	Field selection	70
3.3.5	N-grams	72
3.3.6	Distributed representations	74
3.3.7	Final Pipeline	75
3.3.8	Large-scale application	77
3.4	Conclusion	77
3.5	Future work	78
4	Named Entity Recognition of Pharmacokinetic Parameters	79
4.1	Introduction	79
4.2	Methods	82
4.2.1	NER models	82
4.2.1.1	Evaluation	82
4.2.1.2	Rule-based	84
4.2.1.3	BERT	84
4.2.1.4	SpaCy	86
4.2.2	Corpus construction	88
4.2.2.1	Source	88
4.2.2.2	Annotation interface and Guidelines	89
4.2.2.3	Annotation process	91
4.2.2.4	External dataset validation	94
4.3	Results and Discussion	96
4.3.1	Corpus statistics	96
4.3.2	Model Comparison	97
4.3.3	Active Learning protocol	99
4.3.4	External corpus	101
4.4	Conclusions and Future work	104

5	Few-shot Entity Linking of Pharmacokinetic parameters	105
5.1	Introduction	105
5.2	Methods	107
5.2.1	Corpus construction	107
5.2.1.1	PK Knowledge Base	108
5.2.1.2	Data source and annotation	108
5.2.2	EL architectures	109
5.2.2.1	Bi-encoder	110
5.2.2.2	Baseline	113
5.2.2.3	Evaluation metrics	113
5.3	Results and Discussion	114
5.3.1	Corpus statistics	114
5.3.2	Preliminary analyses	116
	Model initialisation	116
	Sharing bi-encoder parameters	116
	Negative samples	117
5.3.3	Baseline comparison	119
5.3.4	Few-shot performance	120
5.3.5	Error analysis	121
5.3.6	Zero-shot parameters	122
5.4	Conclusions and Future work	123
6	End-to-End Relation Extraction of Pharmacokinetic estimates	125
6.1	Introduction	125
6.2	Methods	129
6.2.1	Corpus construction	129
6.2.1.1	Entities, relations and guidelines	129
6.2.1.2	Source	131
6.2.1.3	Annotation	132
6.2.2	Model development	134
6.2.2.1	Task definition and evaluation	134
6.2.2.2	Architecture	136
	Comparison to previous work	140
6.2.3	Data augmentation	141
6.3	Results and Discussion	144
6.3.1	Corpus	144
6.3.2	Multi-Task solution	146
6.3.3	Effect of encoder and context	148
6.3.4	Augmentation	150
6.3.5	Error analysis	153
6.4	Conclusions and Future work	156
7	Pharmacokinetics database	158
7.1	Database	159
7.1.1	Construction	159
7.1.2	Statistics	160
7.1.3	Quality assessment	161

7.1.4	Potential use	165
7.2	Case study	167
7.2.1	Document search	167
7.2.2	Extraction of estimates	168
8	Conclusions	175
8.1	Summary of research findings	175
8.2	Open Problems and Future work	177
	 Bibliography	 179
	 Appendices	 204
	Appendix A: Guidelines Named Entity Recognition	205
	Appendix B: Named Entity Recognition patterns	225
	Appendix C: Guidelines Relation Extraction	228
	Appendix D: Unit Dictionaries	241

Abbreviations

ADME administration, distribution, metabolism and elimination

BERT Bidirectional Encoder Representations from Transformers

BoW bag-of-words

CBOW Continuous Bag of Words

CRF conditional random field

DDI drug-drug interaction

EHR electric health records

EL Entity Linking

GRU Gated Recurrent Unit

IE Information Extraction

IR Information Retrieval

KB Knowledge Base

LM Language Modelling

LSTM long short-term memory

MeSH Medical Subject Headings

ML Machine Learning

MLM “masked language model”

MLP Multilayer Perceptron

NCA Non-Compartmental Analyses

NER Named Entity Recognition

NLM National Library of Medicine

NLP Natural Language Processing

PBPK physiologically-based pharmacokinetic

PD pharmacodynamic

PICO Population/Problem, Intervention, Comparator, and Outcome

PK pharmacokinetic

PK/PD pharmacokinetic/pharmacodynamic

PMC PubMed Central

POS part-of-speech

QSAR Quantitative Structure-Activity Relationship

ReLU Rectified Linear Activation

RNN Recurrent Neural Network

ROC Receiver Operating Characteristics

SMILES Simplified Molecular-Input Line-Entry System

Tanh Hyperbolic Tangent

UMLS Unified Medical Language System

XGBoost Extreme Gradient Boosting

List of Figures

1.1	Example of the drug concentration vs time profile after a single oral administration and graphical representation of the area under the curve (AUC), maximum concentration (C_{max}), time to C_{max} (t_{max}) and half-life ($t_{1/2}$).	19
1.2	Structural representation of a 2-compartment PK model. The subscripts c and p stand for central and peripheral compartments, A represents the amount of drug, V the volume of distribution and CL and Q the elimination and intercompartmental clearance, respectively.	19
2.1	Graphical representation of a neural network with 4 layers. The input layer is represented in orange and composed by 4 neurons, whereas the output layer is represented by two blue neurons. Two hidden layers are exemplified in grey.	42
2.2	Illustrative example of the sliding widow used in <i>Word2Vec</i> . Taken from Voita 2020 [1].	44
2.3	Schematic representation of the <i>Skip-Gram</i> (a) and Continuous Bag of Words (CBOW) (b) architectures from <i>Word2Vec</i>	44
2.4	Schematic illustration of an RNN where θ represents the network parameters, and x_i , s_i and y_i represent the input, cell state, and output vectors at a particular time step.	46
2.5	Graphical representation of BERT-base accepting a sequence of tokens that go through twelve encoder blocks and return a sequence of vectors. Each Encoder represents a Trasformer Encoder block displayed on the right panel. [CLS] is a special token added at the beginning of each sequence and [SEP] was originally used to separate sentences [2].	49
2.6	(a) Decision tree splitting two features X_1 and X_2 at different points t . A 2D visualisation of the regions generated by these splits is observed in the right hand-panel (b)	52
2.7	Example of a decision tree of depth 3.	53
3.1	MeSH indexing of the term “Pharmacokinetics” and its related sub-terms.	55
3.2	Example of XML tags for title and abstract in PubMed files.	60

3.3	A) Bootstrap procedure to compare the effect of different features during field selection, n-grams and distributed representations analyses. B) The best-performing features from previous analyses were selected to compare different hyperparameter combinations with 5-fold cross validation. Finally, the best-performing features and hyperparameters were used to apply the pipeline to the final test set.	61
3.4	Example of the approaches used to generate distributed representations for an input title after BioBERT encoding. The same procedure was applied for tokens in the abstract.	66
3.5	Comparison word cloud with the unigrams and bigrams obtained after preprocessing each field for bag-of-words encoding. Word size is proportional to the frequency of a token in the vocabulary, whereas the position in the vertical axis refers to the proportion of labels in which that token appears. Note that tokens that are not from the abstract are displayed with a field identifier (e.g. T for tokens from the title or M for mesh terms).	69
3.6	Distribution of F_1 scores for the different features used in the <i>classifier comparison</i> analysis after 200 bootstrap iterations.	70
3.7	Distribution of F_1 scores for the different features used in the <i>field selection</i> analysis after 200 bootstrap iterations. The fields Chemicals, Journal, Authors, Keywords, Affiliations, MeSH terms and Publication Type were encoded together with the title and abstract tokens. The Optimal Fields include the title, abstract, MeSH terms and Publication Type.	72
3.8	Distribution of F_1 scores for the <i>n-grams</i> analysis after 200 bootstrap iterations.	73
3.9	Distribution of F_1 scores for the <i>distributed</i> analysis after 200 bootstrap iterations.	75
3.10	F_1 score distributions for the pipelines using unigrams together with BioBERT embeddings.	76
4.1	Illustration of the fine-tuning strategy for NER with BERT-based models. Blue boxes represent the input embeddings (E_i) for each token. Green boxes represent the output embeddings (T_i) from BERT which go through a feed-forward layer (Token Classifier) and are mapped to token-level BIO labels.	85

4.2	Distribution of number of BERT tokens in each sentence of the training set. The red line determines 256, which was the maximum length established during the experiments.	87
4.3	Flow diagram showing the main processes involved to generate a pool of candidate sentences for NER labelling.	88
4.4	Screenshot of the interface used to annotate PK entities from scientific text. The example displays a single sentence after being annotated.	90
4.5	Schematic representation of the approach used to label instances for the training set by using and updating scispaCy NER model in the loop.	94
4.6	Distribution of F_1 , Recall and Precision scores for the <i>Active Learning</i> and <i>Random Sampling</i> datasets (n=500 sentences) after 10 runs with different random seeds. The left and right panels display the scores considering strict and partial matching of entities, respectively.	101
5.1	Schematic description of the bi-encoder approach to represent, encode, and link PK parameter mentions in sentences to Knowledge Base (KB) entries. T_B represents the language model used to encode mentions and KB entries into single-vector representations in an embedding space. Each KB entity representation was scored against the mention representation using the dot-product, and the entity with the maximum dot product was selected as the model prediction.	110
5.2	Performance of the bi-encoder on the development set when starting the model with BERT-base or BioBERT parameters. Experiments were performed over 20 epochs, and one epoch = 125 training steps. Shaded lines represent the raw values, which were smoothed using an exponential average with a weight of 0.1 (non-shaded lines).	116
5.3	Performance of the bi-encoder on the development set when sharing or not sharing parameters. The left- and hand-side panels show micro-accuracy and mean rank, respectively. Experiments were performed over 20 epochs, and one epoch = 125 training steps. Shaded lines represent the raw values, which were smoothed using an exponential average with a weight of 0.1 (non-shaded lines).	117
5.4	Performance of the bi-encoder on the development set when changing the number of negative samples used for training. The left- and hand-side panels show micro-accuracy and mean rank, respectively. Experiments were performed over 20 epochs, and one epoch = 125 training steps. Shaded lines represent the raw values, which were smoothed using an exponential average with a weight of 0.1 (non-shaded lines).	118

5.5	Performance when using the bi-encoder or the softmax classifier on the development set. The left- and hand-side panels show micro-accuracy and mean rank, respectively. Experiments were performed over 20 epochs, and one epoch = 125 training steps. Shaded lines represent the raw values, which were smoothed using an exponential average with a weight of 0.1 (non-shaded lines).	119
5.6	Box-plot showing the accuracy stratified by class of the bi-encoder and softmax classifier on the test set. The accuracy was computed by class (i.e. KB entity) on the test set, and classes were grouped depending on how many mentions of that class they had on the training set.	121
5.7	Confusion matrix of the bi-encoder on the test set showing the predicted vs true labels per class. The scale is calculated as the proportion of samples predicted as one class (Predicted) that belonged to an annotated (Actual) class. Empty columns correspond to entities that were never predicted in the test set.	122
6.1	The top panel shows a sentence with the key entities to extract PK measurements highlighted. The bottom panel shows the sentence in structured format, with the key columns to build PK databases. The sentence was adapted from Teng et al. [3] for illustration purposes.	126
6.2	Example of a sentence after all entities had been annotated.	130
6.3	The top panel shows a sentence where all entities and relations had been annotated. The bottom panel shows how the annotated entities and relations can be mapped into a tabular format that can be integrated into a database of PK measurements.	131
6.4	Screenshot of the interface used to annotate entities and relations from scientific text. The example displays a single sentence after entities and relations were annotated.	133
6.5	Illustration of the approach used for joint entity and relation extraction of PK measurements. The model first receives a sequence of token embeddings (blue boxes, E_i) and goes through the encoder layers to generate a sequence of contextual token embeddings (green boxes, T_i) which are shared in both tasks. Then, (A) contextual token embeddings go through the token classifier (feed-forward layer) to output BIO labels that will allow recognising entities. (B) Entities and contexts (span between two entities) are represented by max-pooling their contextual token embeddings. Finally, pairs of entities are concatenated with their context representation and passed through the relation classifier (feed-forward layer).	137

6.6	Illustration of labelled sentences with (A) multiple parameter mentions and their respective values, and (B) a measurement that does not refer to the parameter mentioned.	141
6.7	Example of the augmentation process to generate a synthetic sentence (bottom) given its original one (top).	142
6.8	Illustration of the process implemented to covert <i>Units</i> to their dimensions. (A) Input mentions annotated as <i>Units</i> go through a series of rule-based steps to be converted into a (B) Normalised form. Finally, normalised forms are tokenised based on multiplication and division symbols, and each token is mapped to its dimension (C).	143
6.9	Box-plot showing the F_1 scores on the test set for each relation class when performing augmentation of sentences in the training set. Experiments were performed augmenting each annotated sentence once (x2), twice (x3) or three times (x4) or not performing augmentation at all. Ten runs with different seeds were performed for each experiment.	152
6.10	Box-plot showing the NER micro- and macro-averaged F_1 scores when performing augmentation of sentences in the training set. Experiments were performed augmenting each annotated sentence once (x2), twice (x3) or three times (x4). Ten runs with different seeds were performed for each experiment.	152
6.11	Box-plot showing the F_1 scores for each entity type when performing augmentation of sentences in the training set. Experiments were performed augmenting each annotated sentence once (x2), twice (x3) or three times (x4). Ten runs with different seeds were performed for each experiment.	153
6.12	Examples from the test set of main causes of model error at the NER level. One example is presented for over-predicted, missed, partial match, and wrongly annotated sentences.	154
6.13	Example of a complex annotated case where the rule-based algorithm to associate consecutive <i>PK</i> parameters and their respective <i>Values</i> would not work.	155
7.1	Pie chart displaying the frequency of PK estimates in PKUnlocked stratified by the type of parameter. The legend shows the main entity name together with the knowledge base identifier, sorted by their frequency from top to bottom. The top-15 most frequent parameter types were included and the rest were grouped into the <i>Others</i> category. The chart shows the total number of estimates for a given entity followed by their relative % over all estimates in the PKUnlocked.	162

7.2	Example of the evaluation interface built for PKUnlocked. The abstract is displayed with the title, and the pipeline predictions were highlighted at the entity level. The extracted rows for a given abstract were displayed at the bottom in tabular format. From this visualisation, true/false positives and false negatives were derived.	163
7.3	Example of two sentences where the extraction pipeline successfully predicted all entities and relations for each estimate but did not link some of the parameter mentions to the correct knowledge base identifier (i.e. <i>distribution</i> in example A and <i>central</i> in example B).	165
7.4	Example of a search interface to filter estimates from PKUnlocked.	166
7.5	Box-plots displaying the distribution of clearance estimates extracted from the scientific literature and stored in PKUnlocked for Amoxicillin, Clavulanic, Piperacillin, Tazobactam and Meropenem. Individual estimates were represented next to each box-plot. The top panel (A) shows the values before being inspected by a pharmacometrician and the bottom panel (B) shows the values after inspection/removal of values not associated to that drug or measured in non-human studies.	172
7.6	Box-plots displaying the distribution of volume of distribution estimates extracted from the scientific literature and stored in PKUnlocked for Amoxicillin, Clavulanic, Piperacillin, Tazobactam and Meropenem. Individual estimates were represented next to each box-plot. The top panel (A) shows the values before being inspected by a pharmacometrician and the bottom panel (B) shows the values after inspection/removal of values not associated to that drug or measured in non-human studies.	173
7.7	Box-plots displaying the half-life estimates extracted from the scientific literature and stored in PKUnlocked for Amoxicillin, Clavulanic, Piperacillin, Tazobactam, Meropenem and Vancomycin. Individual estimates were represented next to each box-plot. Values are shown after the drug was searched in PKUnlocked without being filtered/inspected by a pharmacometrician. The graph only displays values up to 45h but a few estimates for vancomycin reached up to 180h.	174

List of Tables

1.1	Examples of terminological resources for genes, proteins chemical compounds and enzymes. This table was adapted from Piliouras [4].	25
1.2	Summary of main datasets annotated for biomedical Natural Language Processing (NLP) tasks	27
3.1	Summary statistics reporting the percentage of documents in which a particular field was available, and the proportion of papers labeled as Relevant and Not Relevant . The statistics are reported for both training and final test sets.	64
3.2	Hyperparameters tuned during cross-validation and their default values. The range represents the different values tested for each hyperparameter in the grid-search procedure. The step size refers to the increase between the starting and stop values.	67
3.3	Summary table with performance metrics reported as median (95% CI) and F_1 interquartile variance (IQV) after 200 bootstrap iterations. The performance metrics are compared across pipelines using different classifiers. 70	
3.4	Summary table with performance metrics reported as median (95% CI) and F_1 interquartile variance (IQV) after 200 bootstrap iterations. The performance metrics are compared across pipelines using different fields from PubMed entries.	71
3.5	Summary table with performance metrics reported as median (95% CI) and F_1 interquartile variance (IQV) after 200 bootstrap iterations. The performance metrics are compared across pipelines using different n-grams from the optimal fields.	73
3.6	Summary table with performance metrics reported as median (95% CI) and F_1 interquartile variance (IQV) after 200 bootstrap iterations. The performance metrics are compared across pipelines using different distributed document representations.	74
3.7	Summary table with performance metrics reported as median (95% CI) and F_1 interquartile variance (IQV) after 200 bootstrap iterations. The performance metrics are compared across pipelines using BoW together with distributed representations.	75

3.8	Performance metrics of the final pipeline on the <i>test set</i>	77
4.1	Example of system prediction and true labels in PK NER. Where “O” stands for out of an entity, “B-PK” beginning of a PK entity and “I-PK” for inside a PK entity.	83
4.2	Corpus statistics of the PK-Ontology-corpus stratified by the training and test sets.	95
4.3	Corpus statistics of the PK-NER-corpus stratified by the training, development and test sets.	96
4.4	Results on the test set for different NER models. Metrics are reported at the entity level using strict and partial match.	98
4.5	Common errors from the BioBERT model on the test set. green [*] = annotated entity, blue [*] = model prediction	99
4.6	Summary table with performance metrics comparing random sampling against active learning protocols. The <i>Active Learning</i> dataset was obtained by randomly sampling 500 sentences from the training set. The <i>Random Sampling</i> refers to the development set (n=500 sentences) with different model initializations in each run. Metrics were obtained in the test set after training the pipelines for five epochs. Metrics are reported as median values after 10 runs.	100
4.7	Results obtained on the external validation corpus test set. Metrics from the PK-NER-corpus were obtained by training models on PK-NER-corpus and applying them to the PK-Ontology-corpus. Their own training set was used for the PK-Ontology-corpus to fit the NER model.	102
4.8	Predictions vs annotations in the PK-Ontology-corpus. green [*] = annotated entities, blue [*] = predictions of model trained on the PK-NER-corpus.	103
5.1	Corpus statistics stratified by the training, development and test sets. . .	115
5.2	Summary table with performance metrics on the development and test sets comparing the softmax classifier against the bi-encoder.	119
5.3	Examples of mentions classified as fraction of the drug absorbed (f_a). . . .	123
5.4	Examples of mentions classified as free:brain plasma ratio (k_{pu}).	123
6.1	Corpus statistics summarising the sentences, entities and relations in the dataset stratified by the training, development and test sets.	144

6.2	Named Entity Recognition results on the test set for the model using multi-task learning (MT), NER + RE, against a model only optimising for NER (no-MT). The metrics reported consider strict matching over entity mentions. Results are displayed as the median over ten runs with their interquartile variance in subscript.	147
6.3	End-to-end relation extraction results on the test set for the MT model configuration. Results are displayed as the median over ten runs with their interquartile variance in subscript.	148
6.4	Results on the test set when using different encoder models. Results are displayed as the median over ten runs with their interquartile variance in subscript. NER metrics are the micro- and macro- averaged F_1 scores over all entities, and RE metrics are the F_1 scores for each relation class.	149
6.5	Results on the test set when using different representations as input to the relation classifier. Local context is the max pooling of all tokens strictly between two entities. No context only used the concatenation of each entity representation in a specific relation. Results are displayed as the median over ten runs with their interquartile variance in subscript. NER metrics are the micro- and macro- averaged F_1 scores over all entities and RE metrics are the F_1 scores for each relation class.	150
6.6	Results on the test set when performing augmentation on the training set. Results are displayed as the median over ten runs with their interquartile variance in subscript. NER metrics are the micro- and macro- averaged F_1 scores over all entities, and RE metrics are the F_1 scores for each relation class.	151
7.1	Statistics of the construction process of PKUnlocked.	160

Chapter 1

Introduction and Background

1.1 Thesis motivation

Recent research estimates the R&D costs of bringing a new chemical compound to the market between \$161m and \$4.5bn [5]. Meanwhile, over 90% of drug candidates fail after entering phase I clinical trials [6, 7]. Accurate predictions of candidate drug properties at an early stage are critical for improving the efficiency of this process.

For candidate drugs to elicit the desired effect, they must reach a specific concentration over a certain period of time at the target site of the body [8]. Predicting whether candidate drugs will reach the desired concentration over a certain period at the target site requires understanding the processes of absorption, distribution, metabolism and excretion (ADME) of drugs from the human body. Pharmacokinetic (PK) parameters quantify the ADME processes of chemical compounds through numerical estimates. Therefore, accurate estimation of drugs' PK parameters is crucial for drug development research [8]. Mechanistic models have been widely used to predict the PK parameters of candidate drugs before they are tested in humans. However, a significant proportion of those candidates still fail due to PK complications found during the clinical phases [9]. Hence, improving PK predictions of candidate compounds before they are given to humans is crucial for assessing candidate prospects and optimising the drug development pipeline.

One of the main limitations to improving PK predictions of chemical compounds is the lack of comprehensive and standardised PK repositories [10]. Existing open-access databases collate information ranging from chemical structure to a summary of PK publications, but only very sparse PK information is explicitly reported [10, 11]. As a result, researchers need to search and curate PK estimates from the scientific literature before preclinical predictions can be performed [10, 12]. Unfortunately, the extensive amount of PK information locked in scientific articles and the vast and in-coming number of PK publications limits our ability to efficiently find and curate comprehensive datasets manually [11]. Hence, despite the potential PK data in scientific articles, efficiently exploiting this resource still represents a significant challenge in drug development.

Text mining can help researchers find and extract information from the scientific literature more efficiently using automated approaches. For this reason, this thesis investigates

the application of text mining and NLP to structure PK information reported in the scientific literature. Developing automated systems that can process PK articles effectively has the potential of aiding the curation of more comprehensive datasets, improving pre-clinical predictions of novel compounds, PK meta-analyses and providing informative priors for model building.

Introduction outline The following sections present background and literature review on drug development, pharmacokinetic modelling and limitations in preclinical PK predictions. Then, resources and related work on text mining are reviewed. Finally, the research gap is discussed in detail, and the aims and research questions are presented together with a thesis outline.

1.2 Quantitative pharmacology

The development cycle of a new successful drug can be divided into drug discovery, and drug development [13].

Drug Discovery The discovery phase aims to understand the role of different molecules in a specific disease to detect potential biological targets and design new therapeutic compounds that successfully bind to that target. During drug discovery, *in vitro* and *in silico* studies are performed. *In vitro* studies are performed in a controlled environment where drug properties are studied outside a living organism (e.g. test tubes, flasks, Petri dishes) while *in silico* refers to biological experiments performed via computer simulations. Thousands of compounds may be considered potential drug candidates, but only a small percentage moves into the drug development phase after early testing.

Drug Development During drug development, *in vivo* studies are also performed, which involve those experiments where a drug is administered to a living organism (e.g. animal and human studies). Hence, preclinical and clinical trials are performed to assess the likelihood of that compound becoming a usable medicine.

Bringing a new chemical into the market is an extremely long and complicated process that suffers from a high degree of uncertainty at each stage. For this reason, the pharmaceutical industry has relied on quantitative models to inform and improve the decision-making process at each phase [14–16].

1.2.1 Modelling approaches

Drug Discovery Since *in vitro* and *in silico* experiments are often cheaper to perform than experiments *in vivo*, the drug discovery phase has more often relied on data-driven models. For instance, Quantitative Structure-Activity Relationship (QSAR) approaches use machine learning models to empirically model the relation between molecular descriptors of a particular compound and its biological activity. These molecular descriptors can be related to the structural or the physicochemical properties of that compound [17]. In the conventional setting, QSAR approaches aim to predict specific end-points related to the binding affinity to the biological target during drug discovery [18].

Drug Development In contrast, mechanistic approaches have been commonly used in drug development [19]. Since *in vivo* data is often more sparse and limited, modelling approaches in drug development often rely on prior knowledge of the biological system. Examples of mechanistic models in drug development include population PK models, PK/PD or physiologically-based pharmacokinetics, which will be discussed in detail in the following section. Nonetheless, more recently, QSAR approaches have also been applied to predict clinical end-points, such as predicting the *in human* PK profile of new chemical entities given their molecular descriptors [20–22]. However, the datasets in which models are trained often contain a limited number of PK end-points, limiting their performance and use in the preclinical setting [12, 23]. QSAR models represent a promising approach in drug development to infer relations between chemical descriptors and clinical end-points without mechanistic assumptions. However, the performance of these models will be highly dependent on the availability of large databases containing multiple chemical descriptors, clinical PK parameters and contextual information of the clinical trial (i.e. route of administration, conditions of the patient population).

1.2.2 Drug development

Before a drug candidate proceeds to clinical trials, safety and efficacy information is assessed through a combination of *in silico*, *in vitro* and animal models to predict a potential human outcome [24].

When clinical data is available, population pharmacokinetic/pharmacodynamic (PK/PD) models are the main asset to understand and quantify exposure-response properties in a specific cohort. However, in the preclinical setting, no human data is yet available, and PK/PD models of comparator drugs are often used to project the anticipated *in human* response and provide a therapeutic range for the first clinical dose of a new compound [14]. It is noteworthy that the confidence of preclinical predictions is highly dependent on the availability and quality of PK information from similar compounds, which is largely gathered from the scientific literature and manually curated databases [14]. Additionally,

allometric scaling is often used to extrapolate PK parameters obtained in animal studies to PK parameters in humans [25]. If there is enough safety and tolerability evidence to justify the first human dose, the candidate drug enters the clinical phase, where its predicted behaviour and potential outcomes will be tested in humans.

The clinical stage of drug development can be divided into three main phases. During early Phase I, clinical trials are often performed in healthy volunteers, and initial PK models are developed to estimate the administration, distribution, metabolism and elimination (ADME) properties as well as potential side effects. These results are used to optimise the dosing regimens of subsequent stages. Along with Phase I, the primary sources (and magnitudes) of variability associated with PK and PD parameters are quantified to evaluate whether they are likely to change in a target population.

In Phase II, the effect of different dosing regimens is studied, and signs of efficacy in the target population are confirmed/rejected [26]. As more clinical data become available, PK/PD models are updated to understand and optimise drug responses in the patient population [14]. Additionally, during Phase II, PK and PD information is used to improve the design of Phase III trials.

During Phase III, the drug effects are compared with current therapies or placebo [26], and conclusions regarding the safety, tolerability and efficacy of the drug candidate are made. Finally, population PK/PD models are validated with Phase III clinical data, and the need for dose adjustments in specific populations (e.g. paediatric) might be studied. Overall, this drug development approach is based on the learn-and-confirm paradigm [14], in which initial models are updated and validated as new data become available during the clinical phases.

Physiologically-based pharmacokinetics Another common approach in drug development is physiologically-based pharmacokinetic (PBPK) modelling, which has become increasingly popular in preclinical and clinical drug development. In PBPK, the different physiological compartments are modelled together with their potential interactions with a range of compounds. The PK parameters are then obtained through a “bottom-up” approach, in which the high-level ADME properties emerge by modelling interactions at the molecular level. This very mechanistic framework allows researchers to integrate detailed knowledge (about the drug and the system) from *in vitro* and *in vivo* data to simulate drug exposure and response in a variety of contexts [27]. These models focus on the molecular interplay between the drug and the underlying system to simulate the PK/PD profile [28]. For this reason, different applications can be found in the scientific literature [29], including preclinical assessment for the first *in human* dose [30], evaluation of potential drug-drug interactions (DDIs) [31–33] and *in silico* profiling of the dose-response curve for specific target populations (e.g. paediatric patients, critically ill patients) [34].

The following sections present the techniques behind PK/PD modelling, followed by the main approaches involved in preclinical PK parameter prediction.

1.2.3 Pharmacokinetic/Pharmacodynamic modelling

Understanding the effect of a specific compound within the human body requires considering its intrinsic molecular activity together with its ability to reach the desired concentration in the action site [35]. As previously mentioned, the science of PK studies the ADME processes while pharmacodynamics (PDs) comprises the biochemical and physiological response to a chemical compound [36]. In broad terms, PK can be described as “what the body does to the drug”, whereas PD relates to “what the drug does to the body”. PK/PD modelling comprises a framework of mathematical and statistical techniques to quantify and explain variability in drug exposure (PK) and response (PD).

In the clinical context, PK models are used to infer ADME properties by studying how drug and metabolite concentrations change over time within the human body. In contrast, PD models estimate different response end-points (e.g. bacterial concentration, disease scores, clinical events) as a function of drug concentration to evaluate the potential drug efficacy or toxicity [37]. Furthermore, the integration of PK and PD models (PK/PD) is often conducted to predict and understand the entire relationship dose-concentration-effect under a unified modelling framework [37]. Overall, these approaches fall under the umbrella of *pharmacometrics*, which has been defined as “the science of developing and applying mathematical and statistical methods to characterise, understand and predict a drug’s pharmacokinetic and pharmacodynamic behaviour” [38].

1.2.3.1 Pharmacokinetic parameters

Once a drug is administered to a patient population, blood samples are periodically taken to measure concentration changes over time. Then, blood cells are often separated from the sample solution to measure the drug concentration in plasma (or serum), which is more reflective of the drug concentrations available for pharmacological effect. Once the concentration data has been collected and preprocessed, the concentration versus time plot is drawn, and several PK parameters regarding ADME mechanisms can be obtained (Figure 1.1). The main parameters reported in a PK study include [39]:

- Maximum concentration: C_{max} (units: mass/volume) and time to reach C_{max} : t_{max} (units: time). Assuming a concentration-time profile with absorption and elimination phases (Figure 1.1), C_{max} and t_{max} can be directly obtained from the observed data by locating the peak concentration and the time at which this concentration was reached, respectively.

- Area under the (concentration-time) curve: **AUC** (units: mass · time / volume). AUC represents the drug exposure through time and can be obtained by computing the integral below the concentration-time curve (Figure 1.1). AUC might be calculated from the time of administration until the last measurement (AUC_{0-t}) with the trapezoidal rule or if the elimination rate constant (k) is known, extrapolated until infinity ($AUC_{0-\infty}$). The rate constant k can be estimated with the slope of the last sample points of the decay phase or by computing the fraction of clearance over the volume of distribution (See equation 1.5). Then, given the concentration at a particular time point (C_t), k and the AUC_{0-t} , $AUC_{0-\infty}$ can be estimated:

$$AUC_{t-\infty} = \frac{C_t}{k} \quad (1.1)$$

$$AUC_{0-\infty} = AUC_{0-t} + AUC_{t-\infty} \quad (1.2)$$

- Bioavailability: **F** (unitless). F represents the fraction of an administered dose that reaches the systemic circulation [40]. If a drug is administered intravenously (i.v), F will be 100%. However, when the administration route is extravascular, F needs to be estimated in the PK analysis to account for absorption processes. For instance, to determine the F of a drug following oral administration (p.o), a PK study has to be performed with both i.v and p.o administration. Then, their time-concentration profiles can be related to derive F :

$$F = 100 \cdot \frac{AUC_{p.o} \cdot Dose_{i.v}}{AUC_{i.v} \cdot Dose_{p.o}} \quad (1.3)$$

- Volume of distribution (or apparent volume of distribution) : **V_D** (units: volume). Given the total amount of an administered substance (Dose), V_D represents the necessary volume to achieve the concentration measured in plasma (C) [41]. Hence, it can be often described as:

$$V_D = \frac{Dose}{C} \quad (1.4)$$

- Clearance: **Cl** (units: volume / time). Cl is defined as the volume of plasma from which a particular drug is completely removed per unit of time [42]. It represents the elimination rate of a particular substance divided by its concentration in plasma. Depending on the excretion route of a particular compound, one can measure the renal (Cl_{renal}), hepatic ($Cl_{hepatic}$) or lung (Cl_{lung}) clearance among other possible elimination routes. The total Cl is then the sum of each individual Cl . Given the k and the V_D of a drug, Cl can be obtained through:

$$Cl = V_D \cdot k \quad (1.5)$$

Additionally, considering the F of a particular compound, Cl can be calculated as the fraction of absorbed dose over AUC :

$$Cl = \frac{Dose \cdot F}{AUC} \quad (1.6)$$

- Half-life: $t_{1/2}$ (units: time). Timepoint of the elimination phase where the drug has lost half of its maximum concentration ($C_{max}/2$) (Figure 1.1). Assuming a first-order elimination rate, $t_{1/2}$ can be determined with the Cl and V_D :

$$t_{1/2} = \frac{\ln 2 \cdot V_D}{Cl} \quad (1.7)$$

- Free fraction: f_u (units: unitless). Several drugs are partially bound to plasma and tissue proteins, but only the free (unbound) fraction of the drug exhibits biological activity [26]. Hence, to develop effective PKPD models, f_u might be estimated to consider the unbound drug concentration in the site of action.

These parameters have become the main metrics to characterise the PK properties of drugs in specific populations and, therefore, the focus of PK analyses. The parameters and units presented in this section represent the main PK scenario, but additional PK parameters and units might be found across the scientific literature. For a more detailed overview of the different PK parameters see Wu et al. 2013 [43]. To obtain PK parameter estimates, one might either use Non-Compartmental Analyses (NCA) or model-based approaches [44].

Non-compartmental analyses NCA uses descriptive approaches and algebraic equations to estimate PK parameters and determine the systemic exposure of a drug [44]. For instance, Figure 1.1 illustrates a graphical interpretation of some PK parameters from the concentration-time data. NCA applies the trapezoidal rule to compute the AUC_{0-t} [45]. Then, it uses the last 3-4 observations to graphically estimate k with the terminal slope of the time-concentration curve [46]. Finally, once AUC and k are estimated, other PK parameters can be derived with the equations previously mentioned. One of the main advantages of this methodology is that it relies on fewer physiological assumptions than model-based approaches. The relative simplicity of NCA also provides fast results and low variability between analysts [41]. However, accurate estimation of AUC with the trapezoidal rule is highly dependent on rich sample schedules. Additionally, NCA approaches have limited accuracy in extrapolating plasma concentration over time and predicting drug PK for other populations or dose regimens. As a result, NCA is often used to determine drug exposure within a single study when rich sample data is available [44].

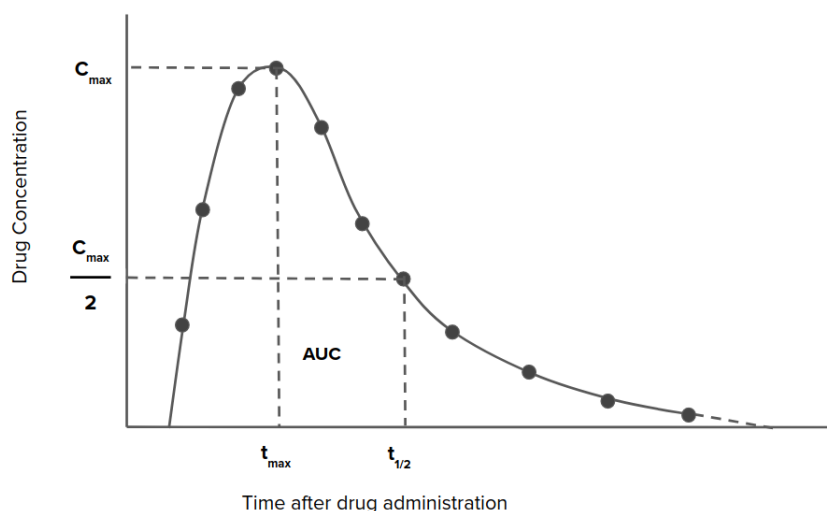


FIGURE 1.1: Example of the drug concentration vs time profile after a single oral administration and graphical representation of the area under the curve (AUC), maximum concentration (C_{max}), time to C_{max} (t_{max}) and half-life ($t_{1/2}$).

Model-based approaches Model-based approaches are beneficial for dealing with heterogeneous and sparse data and characterising variability sources between concentration-time profiles in a population. They are based on differential equations that represent ADME processes within a system. These population models are composed of a structural, statistical and covariate model [47].

The structural model describes the concentration-time relation and is often divided into compartments in which the drug is assumed to be completely mixed. For instance, Figure 1.2 represents a two-compartmental model in which the drug dynamics can be described by means of differential equations [26]. The statistical model is used to represent the inter-individual variability (exposure differences between subjects), inter-occasion variability (differences in the same subject from one dose to another) and residual (unexplained) variability [47]. Then, the covariate model considers the effect of the subject's characteristics such as demographics or physiological state with functions relating a specific parameter with a covariate value. For instance, it is known that the parameter Cl tends to increase as body weight increases [48]. Hence, incorporating an equation that describes the relationship between weight and Cl will often result in better predictions and reduce the unexplained variability in the population model.

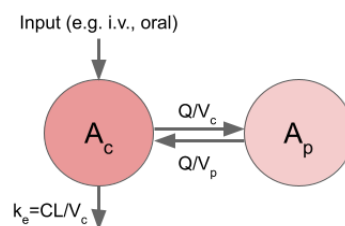


FIGURE 1.2: Structural representation of a 2-compartment PK model. The subscripts c and p stand for central and peripheral compartments, A represents the amount of drug, V the volume of distribution and CL and Q the elimination and intercompartmental clearance, respectively.

Overall, the population PK model is defined with a set of parameters from the structural, statistical and covariate model that need to be estimated from the population data. A variety of estimation methods based on nonlinear regression have been used to find the parameters that best describe the clinical data given an objective function. For a review on population PK parameter estimation methods, see Ette and Williams [49]. The estimated parameters of the population model will then provide information on the population PK parameters, relevant covariates and sources of variability in the population.

1.2.4 Pharmacokinetic preclinical prediction

During preclinical drug development, the prediction of PK parameters is a critical step to detect and discard those molecules with inappropriate *in vivo* properties at an early stage [50]. A wide range of *in silico* models have been developed to predict the potential PK parameters of new chemical entities, but their performance is primarily limited by the quality and amount of data on which they are based [51]. Here, existing resources for human PK parameter prediction are discussed.

Commonly, predictive models such as QSAR are applied to datasets where the ADME properties of multiple compounds are reported together with several molecular descriptors. These descriptors represent the chemical structure of the compounds (e.g. 3-D representation, Simplified Molecular-Input Line-Entry System (SMILES) string) and might include information on their physicochemical properties (e.g. molecular weight, solubility, lipophilicity). Then, different ADME properties (*in vitro* and *in vivo*) are reported and used as end-points for predictive models.

Several databases containing information on many chemical compounds have been developed in the public domain during the last few decades. Amongst the most well-known open-access databases for chemical descriptors and ADME-related data are PubChem [52], DrugBank [53], ChEMBL [54], and ACTorR [55]. Despite the wealth of structural and physicochemical data contained in these databases, the volume and detail of ADME data are frequently insufficient to perform direct pharmacokinetic (PK) parameter prediction studies [56, 57]. For instance, very little information about the study’s design or modelling approach to pharmacokinetic parameters is reported. As a result, the majority of datasets used to predict PK parameters combine data from in-house studies, publicly available databases, and information taken from the scientific literature [58, 59]. Thus, this aggregation process becomes extremely time-consuming, as it needs researchers to manually search for, collect, and standardise PK data from different sources before conducting predictive studies.

Przybylak et al. 2018 [35], provided an excellent review of publicly available datasets curated for ADME prediction. In that study, 31 datasets were selected (based on a

quality assessment) and characterised by the type of ADME end-points, and the number of chemical compounds reported. A majority of these datasets were based on *in vitro* ADME properties, and only a few reported human PK parameters. In addition, rich parameter data was only reported for two ADME end-points (bioavailability and oral absorption), and very scarce PK data were reported for other parameters (e.g. Cl , V_D , $t_{1/2}$). In 2018, Lombardo et al. [58] released a dataset reporting the V_D , Cl , $t_{1/2}$ and f_u for 1352 drugs following intravenous administration in humans together with several structural and physicochemical descriptors. Following this release, several studies have applied QSAR algorithms to Lombardo’s et al. dataset to estimate human PK parameters [60, 61]. Despite the valuable information reported in this dataset, it was still limited to 1352 data points, a single administration route (i.v), and not reporting potential sources of PK parameter variability (e.g. the number of subjects, modelling approach, covariates).

The most comprehensive and detailed dataset of PK parameters currently available in the public domain is PK/DB [10]. PK/DB stores fine-grained information on PK properties from multiple compounds derived from clinical trials (*in vivo* human studies). For each clinical trial reported in PK/DB, the dataset curates standardised data about the demographics of the patient cohort, the study design (e.g. drug, dosing, route of administration), the specific PK estimations (e.g. values and units of Cl , AUC , $t_{1/2}$) as well as the concentration-time data. In PK/DB, curators have to (1) select a corpus of papers relevant to a given drug or problem domain (e.g. meta-analysis of PK parameters for caffeine in humans), (2) manually extract and store the desired metadata from each study, (3) standardise the extracted data (e.g. units) and (4) validate and check the extracted data with multiple curators [10]. This curation approach results in extensive and reliable data for the selected study context, but it is highly limited by the time and expertise required to collate data for each context. So far, PK/DB (v0.9.3) exclusively focuses on substances applied in dynamical liver function tests (i.e. glucose metabolism studies, statins and benzodiazepines) with a total of 512 studies analysed [10].

Overall, despite the high-quality data stored in Lombardo’s et al. and PK/DB datasets, their approaches are still limited by the ability to efficiently scan and keep up-to-date with the ever-growing and unstructured PK literature since they rely on experts constantly curating data from numerous publications.

Text mining and machine learning have recently arisen as valuable tools to automatically process large amounts of textual data. This thesis aims to develop text mining approaches that efficiently structure PK information reported in the scientific literature. The ultimate goal of these automated approaches is to aid researchers in generating more comprehensive and up-to-date ADME datasets that can be used for preclinical predictions, PK meta-analyses and provide more informative priors during PBPK, PK/PD or population PK model building.

1.3 Text mining

With a growing volume of information reported in textual format, text mining has become an essential technique to efficiently extract, integrate and exploit knowledge stored across extensive collections of unstructured text. The main goal of text mining is to derive high-quality information from text using automated pipelines. Natural Language Processing (NLP) is one of the principal methodologies behind text mining applications [62]. NLP is a sub-field of computational linguistics and artificial intelligence that attempts to represent the meaning of text (written in natural language) with computational techniques such as morphological analysis, part-of-speech (POS) tagging, syntactic parsing, and other types of linguistic analyses [63]. Throughout this thesis, the term NLP will be used as a methodology used for text mining purposes. However, one might find both terms used interchangeably across the literature.

This section provides an introduction to text mining and NLP with particular attention to resources and applications for the biomedical domain and previous work on extracting PK parameters.

1.3.1 Underlying techniques

Ananiadou and McNaught [64] divide the main activities involved in text mining into (1) Information Retrieval (IR), which identifies relevant texts (e.g. documents, paragraphs, sentences) from a given collection; (2) Information Extraction (IE), which locates and extracts specific instances and relations from text; and (3) data mining, which analyses and exploits the information extracted across multiple sources. In each of these stages, a wide variety of tasks have been studied and applied across the literature, including text classification, Named Entity Recognition (NER), Entity Linking (EL) and relationship extraction, amongst others. Despite a large number of tasks and approaches in IR and IE, the underlying methods and techniques can be roughly divided into dictionary-based, rule-based and statistical approaches.

Dictionary-based

Dictionary-based approaches use databases and lexical resources (e.g. lexicons¹, terminologies² and ontologies³) to locate specific occurrences in the text. Given a particular input text, the goal is to find and categorise those character sequences (tokens) that

¹ Collection of words in a specific language together with descriptors of its use. This may include phonetic, morphological, syntactic and other linguistic information [65].

² Vocabulary of terms and meanings used in a specific domain [65].

³ Formal, explicit description of concepts in a specific domain, including a set of vocabulary terms together with their description and hierarchical relationships [65].

match entry names in the dictionary. Therefore, terminological resources are often applied for NER and normalisation tasks. In the biomedical domain, this approach has been extensively used to detect and standardise mentions of genes, drugs and proteins, and a number of resources have been generated for this purpose [66–68]. However, some of their main challenges include the existence of synonyms, term variations, and the growing number of newly published terms [64]. Although some methodologies have adapted these approaches to account for morphological variations and synonyms (e.g. [69],[70]), many terms require further tasks to deal with more complex variations.

Rule-based

Rule-based approaches commonly refer to hand-crafted rules that are developed to detect particular patterns of interest. They have been widely applied in IR and IE tasks to account for a broader range of variations than dictionary-based approaches [64]. For instance, in text classification, rule-based approaches might be used to assign the category of an input text depending on its length or the frequency of specific terms. In NER tasks, regular expressions might be generated to characterise morphological and syntactic variations of certain instances, considering not only the character- and word-level variation but, to some extent, the context of the term [71]. However, due to the high variability of natural language, rules usually require extensive domain knowledge, and some tasks might entail highly complex rules to describe the desired pattern. Also, since rules tend to be domain and task-specific, they exhibit poor adaptability to other domains and classes. For this reason, data-driven approaches have been often applied to model heterogeneous patterns.

Statistical

The exponential increase of text data and computer power over the last few decades has resulted in a growing number of studies relying on statistical methods for NLP [71]. The goal of statistical approaches is to understand the distribution of the input text to develop predictive models for specific tasks. This is commonly achieved by utilising Machine Learning (ML) algorithms, which, given a large number of examples (training data), fit the parameters of mathematical models in order to successfully perform future predictions by inferring specific patterns from the data [4, 72]. In ML, each textual input is usually represented with a numerical vector $x \in \mathbb{R}^d$ with d features, where each feature describes a particular attribute of the data. To train (or fit) the ML model, a set of independent n examples is provided, generating a matrix of features $X \in \mathbb{R}^{n \times d}$ [73]. In the context of *supervised learning*, each example x_i is associated to a certain *label*, y_i . The goal is to fit a model ($f(x)$) that relates the input features to the output labels in order to perform future predictions or to better understand the relationship $x \rightarrow f(x) \rightarrow y$. *Unsupervised learning* refers to the situation in which no designated

labels are available. In those cases, the goal might be to infer some structure in X , understand relationships between features, or improve the numerical representations of the raw text [74]. In addition, ML tasks are often categorised as either *classification* or *regression* problems. In regression, the labels are continuous numbers, whereas in classification y belongs to a predefined set of categories [73].

Unlike dictionary- and rule-based approaches, ML becomes particularly efficient to model textual structures with high variability (e.g. multiple text occurrences referring to the same concept) and complex underlying patterns when enough training data is available. However, their main limitation is their dependence on large amounts of high-quality training data, which is often challenging to generate. Additionally, unlike ML approaches, dictionaries and rules are fast and flexible methods that allow users to easily extend and modify those dictionaries/rules when errors are detected, making them reliable and production-friendly approaches. For this reason, text mining applications often combine statistical, rule-based and dictionary-based methods, and deciding when to use one or another will highly depend on the specific task and resources available [64].

1.3.2 Biomedical text mining

With a growing volume of electronic clinical records and scientific publications, biomedical text mining has become a central approach to deal with these large amounts of potentially valuable data. The biomedical text presents several particular challenges, including highly domain-specific terminology, extensive use of abbreviations and acronyms, diversity of naming conventions and the constant appearance of newly published terms, to name but a few [64]. Several resources have been developed to deal with the challenges. These range from lexical, terminological and ontological resources to text mining tools and frameworks for biomedical IR and IE tasks.

1.3.2.1 Resources

Corpora In the biomedical domain, MEDLINE represents one of the primary resources for text mining applications from the scientific literature. MEDLINE is a journal citation database with more than 26 million references indexed by the Medical Subject Headings (MeSH) controlled vocabulary from the U.S. National Library of Medicine (NLM) [75]. These citations, together with their titles, abstracts and other metadata, can be accessed through the PubMed database, which also includes additional records from life science journals, author manuscripts, and online books [76]. For text mining purposes, PubMed citations and their metadata can be easily accessed through the E-utilities API or downloaded from the NCBI FTP sites⁴. As a result, PubMed represents

⁴ <ftp://ftp.ncbi.nlm.nih.gov/pubmed/>

a large volume of indexed biomedical text from more than 30 million citations. However, these resources do not provide valuable information in full-text articles. The Open Access subset from PubMed Central (PMC) represents the central resource for this purpose, which provides free access to the full-text of over 2.5 million PubMed articles [77]. Finally, several new collections of clinical text have recently become available for text mining purposes. Some of these include electric health records (EHR), the MIMIC III database [78], the annotated collections from the i2b2 database [79] and Web resources in which biomedical information has been reported (e.g. medical blogs and forums, Twitter). Other bibliographic databases containing relevant PK data include EMBASE [80], or ClinicalTrials.gov [81] which contain a vast number of summary reports from clinical studies, which often report valuable PK data for constructing ADME datasets.

Knowledge and terminological resources A large number of terminologies and ontologies have been generated to structure the variety of terms in the biomedical domain. One of the most comprehensive and well-known resources is the Unified Medical Language System (UMLS), which integrates several biomedical nomenclatures, controlled vocabularies and ontologies maintained by the NLM [82]. Terms in the UMLS are organised by concept, meaning and term synonyms under the Metathesaurus vocabulary database [83]. Then, the UMLS Semantic Network [84] provides consistent categorisation and defines relationships between concepts in the Metathesaurus. Additionally, the UMLS provides tools such as the Specialist lexicon [85] to perform lexical analysis of raw biomedical text and facilitate subsequent NLP tasks.

TABLE 1.1: Examples of terminological resources for genes, proteins chemical compounds and enzymes. This table was adapted from Piliouras [4].

Domain	Resource	URL
Genes and Proteins	UniProt	http://www.uniprot.org/
	Entrez Gene	http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=gene
Chemicals	PubChem	http://pubchem.ncbi.nlm.nih.gov/
	SureChem	https://www.surechem.com/
	ChemIDplus	http://chem.sis.nlm.nih.gov/chemidplus/
	ChEBI	http://www.ebi.ac.uk/chebi/
	CheMBL	https://www.ebi.ac.uk/chembl/
	DrugBank	http://www.drugbank.ca/
	DiDB	http://www.druginteractioninfo.org/
Enzymes	RxNorm	http://www.nlm.nih.gov/research/umls/rxnorm/
	BRENDA	http://www.brenda-enzymes.org/

Although UMLS has a broad coverage for general biomedical concepts, separate and specialised resources have been developed for specific entities. For instance, some specialised dictionaries for genes, proteins, chemicals and enzymes are shown in Table 1.1. An example of a specialised ontology is the Pharmacogenomics Knowledge Base [86],

which has annotated gene variants, phenotype data and gene-drug-disease relationships from many scientific publications [71].

In PK, Wu et al. 2013 [43] developed an ontology covering those concepts appearing in the PK field. Specifically, their contribution characterised PK parameters from *in vitro* experiments and *in vivo* studies, modelling approaches, types of study design and quantification methods. Also, they included external dictionaries to cover transporters, metabolism enzymes, drug terms and demographics. The authors also provided a manually annotated corpus with 541 PubMed PK-related abstracts based on their ontology. This resource is particularly relevant for this project since it provides a structured representation of those PK parameters appearing in *in vivo* studies together with demographic and modelling concepts.

Annotations With the rise of statistical approaches, scientific and biomedical text annotations represent a crucial resource to train and evaluate NLP algorithms. Biomedical NLP annotations are often performed by experts in a particular biomedical domain, that, given a specific text, annotate desired properties such as token or sentence boundaries, POS tags, sentence categories (e.g. Population/Problem, Intervention, Comparator, and Outcome (PICO) [87]), entities (e.g. chemicals, genes, proteins), relations between entities and others. The annotations can also be performed at different linguistic levels, including grammatical (morphological and syntactical), semantic and or pragmatic annotations [71]. Once textual annotations are provided, ML models are often trained to model the underlying pattern of those manual annotations and perform inference on unseen text.

Some of the main publicly available benchmark datasets for biomedical NLP tasks are presented in Table 1.2. Multiple datasets have been curated to promote the development of text mining systems capable of detecting and normalising specific biomedical entities (e.g. BC5CDR, Linneaus, NCBI-Disease, SCAI[109]), extracting biomedical relationships between entities (e.g. ChemProt, DDI, GAD), answer biomedical questions (e.g. PubMedQA, BioASQ), perform efficient tokenisation, POS tagging, dependency parsing or sentence segmentation of biomedical text (e.g. CRAFT, GENIA) and others. Despite significant efforts have been made to develop ML models that can process biomedical text, the quality and usage of those models will highly depend on (1) the annotation consistency, (2) diversity and (3) size of the datasets used to train those models. Since manual annotation of textual data is a complex and time-consuming task, several tools have also been developed to aid researchers in this process. Some of the most widely used include BRAT [110] (open-source) and Prodigy [111] (commercial).

TABLE 1.2: Summary of main datasets annotated for biomedical NLP tasks

Dataset	Tasks	Domains	Reference
AnatEM	NER	Anatomy	[88]
BC5CDR	NER	Chemicals and Diseases	[89]
BC4CHEMD	NER	Chemicals	[90]
BioNLP13CG	NER	Cancer Genetics	[91]
JNLPBA	NER	Genes and Proteins	[92]
Linnaeus	NER	Species	[93]
NCBI-Disease	NER	Diseases	[94]
S800	NER	Species	[95]
BC2GM	NER	Genetics	[96]
CALBC	NER	General	[97]
MedMentions	NER and EL	General	[98]
ChemProt	Relation Extraction	Protein-Chemical	[99]
DDI	Relation Extraction	Drug-Drug interaction	[100]
GAD	Relation Extraction	Gene-Disease	[101]
PubMedQA	Question Answering	General	[102]
BioASQ	Question Answering	General	[103]
EBM PICO	PICO	General	[104]
BIOSES	Sentence Similarity	General	[105]
HoC	Document classification	General	[106]
CRAFT	Syntactic analysis and NER	General	[107]
GENIA	Syntactic analysis and NER	General	[108]

1.3.2.2 Tasks, supporting tools and evaluation metrics

Biomedical text mining covers a wide spectrum of tasks and research efforts that have resulted in a large number of publications and specialised software. As previously mentioned, some of the main tasks involved in IR, NER, EL and relation extraction, amongst others.

Information Retrieval One of the main steps in text mining applications is to identify those texts that are relevant for IE purposes. Commonly, a set of documents or sentences (named corpus) is provided as an input, and the goal is to either categorise or rank each input based on specific textual features. In many cases, search systems retrieve documents or sentences depending on whether they contain a list of input terms without considering how these might be used in context [112]. Dictionary and rule-based approaches have been extensively used to expand the limitations of a simple match query. For instance, the PubMed search engine allows querying relevant citations to a particular topic by making use of the MeSH indexing together with keywords from the title, abstract and other indexed metadata [113]. While this type of retrieval is often quite efficient for providing a potentially relevant subset of documents, more advanced approaches might be needed for specific domains and queries. For instance, latent semantic analyses and supervised ML approaches might be used to identify potential topics in a given

corpus based on term frequency distribution or to develop classifiers for predefined categories [114]. Some of these approaches have been developed on top of PubMed, giving rise to specialised retrieval systems such as Relemed [115], MScanner [116], or BioReader [117]. In the kinetics domain, Wittig et al. [118] developed SABIO-RK, which is a manually-curated database that stores comprehensive information about biochemical reactions and kinetic properties. More recently, Wang et al. [119] developed a web resource for browsing evidence of supplement-drug interactions automatically extracted from the biomedical literature.

Named Entity Recognition Biomedical IE often starts with the identification of spans of text referring to certain concepts such as drugs, proteins or species. As previously mentioned, NER is particularly challenging in the biomedical domain. For example, the expressions “heart attack” and “myocardial infarction” refer to the same medical complication, despite morphologically unrelated. To detect this with an NER system, one might either (a) construct a dictionary with both mentions under the same concept or (b) build an algorithm comparing the similarity of their contexts. Ontologies such as the UMLS and specialised dictionaries have been widely used for biomedical NER. However, a large number of synonyms, abbreviations, acronyms and constantly incoming new terms makes dictionary-based approaches unlikely to be complete, which often limits their recall in NER tasks [71]. Many rule-based approaches such as EM-PathIE and PASTA [120, 121] have been applied to account for term variations and some contextual patterns. However, statistical approaches (often combined with rules and dictionaries) have exhibited state-of-the-art results in NER tasks when considerable amounts of annotated data are available for those entities [122]. The main advantage of these approaches is their ability to generate and select complex features of the tokens and their surrounding context that would be difficult to identify with hand-crafted rules. ML algorithms generally approach NER problems as a sequence-labelling task, in which each token in a sentence is classified into a particular category based on its features and some contextual information. Studies applying ML for biomedical NER have used conditional random fields (CRFs) [123], semi-Markov linear classifiers [124], and more recently, deep neural networks such as long short-term memory (LSTM)s [125] or Transformer models [122].

Entity Linking After biomedical entities have been recognised, it is often desirable to standardise those entities into their canonical forms or to associate them with an entry in a knowledge base [71]. For instance, after applying a drug NER model to a particular corpus, one might need to link those recognised drug mentions to their ChEBI or MeSH identifiers. EL (also referred to as Entity Normalisation or Standardisation) is the task of grounding entity mentions to a reference knowledge base. To do so, dictionary and rule-based systems are the most widely used approaches, which merely rely on morphological information of entity mentions. Lowercasing and abbreviation

resolution are often performed as preprocessing steps. Then, the recognised mentions are often compared to existing dictionaries through exact, partial or pattern matching approaches before linking them to a dictionary ID. In many cases, biomedical NER systems integrate dictionaries from multiple sources to perform EL of several entity types simultaneously (e.g. proteins, species, drugs) [126]. More recently, ML approaches based on deep neural networks have been used for EL to incorporate semantic information of entity mentions [127–130].

Several publicly available tools have been developed in the biomedical domain for NER and EL based on the approaches mentioned above. One of the most widely used is MetaMap [131], which identifies UMLS concepts from free text and has a variety of features facilitating biomedical text processing. In addition, a large number of tools combining statistical with rule and dictionary-based approaches have been developed. Some of the most common ones are ezTag [132], tmTool [133] or PubTator [134]. One of the best architectures for biomedical NER currently available is BioBERT [122], which exhibited state-of-the-art performance for disease, drug/chemicals, gene/proteins and species NER in several benchmark datasets. BioBERT uses a pre-trained Transformer model that was fine-tuned to perform NER, and it was further extended for EL with BERN [126] using a set of heuristics to link entity mentions to several biomedical knowledge bases. More recently, several ML approaches have focused on jointly modelling both tasks, NER and EL in an end-to-end fashion by using a single neural model as an encoder [130, 135].

Relation extraction Another subsequent task in biomedical IE is relation extraction, which attempts to determine potential relationships among recognised entities. Some common scenarios include protein-protein or DDIs, and gene-disease or gene-phenotype relationships. Although the entity types for NER tasks are generally well established, the types of relationships can be extensive and difficult to categorise into predefined classes. Therefore, depending on the specific task, one might wish to identify the token characterising a relationship between entities (e.g. “increases”, “inhibits”) or classify an entity tuple⁵ (of recognised named entities) into a specific relation type. Relation extraction approaches have become increasingly complex over time, starting with simple techniques such as co-occurrence statistics (measuring the frequency of entities co-appearing) to more advanced methods involving the syntactic analysis and dependency parsing [71]. Several rule-based approaches have been combined with ML algorithms to perform dependency parsing, which uses syntactic descriptors to produce dependency graphs that encode grammatical relations between words. Then, several studies applied ML to extract useful information from the dependency graphs in order to extract or classify relationships between entities [71]. As a relevant example, Zhang et al. 2015 [136] presented a framework to characterise pharmacokinetic DDIs by training a ML classifier

⁵ Pair of entities.

from the features derived in syntactic and dependency parsing. A widely used software for this task is the open-domain Stanford Dependency Parser [137], which has also been applied to biomedical relation extraction tasks [138]. Other approaches such as the fine-tuned versions of BioBERT and SciBERT [139] have exhibited great performance on biomedical relation extraction tasks by encoding the representation of entity mention pairs along with the corresponding context, but without explicit dependency parsing [122, 140]. More recently, in the kinetics domain, Wang et al. [119] fine-tuned a pre-trained RoBERTa model to classify sentences reporting DDIs, and called the resulting model RoBERTa-DDI.

Further tasks and software Apart from the tasks mentioned above, text mining has also been applied to other relevant areas such as biomedical *event* (which is often more abstract than binary relations) extraction [141], biomedical question answering [142] and other approaches for literature-based discovery [138, 143]. Over the last few years, software tools integrating multiple NLP functionalities for biomedical IE tasks have become increasingly popular among researchers in the field. One of the most widely used is cTakes [144], which has been frequently applied to EHRs for characterising patient cohorts, extracting adverse drug events, identifying risk factors, and other similar tasks [145]. Another relevant tool is MedEx [146], which is a rule-based software to extract medication name, strength, route and frequency of administration and has shown robust performance in a variety of contexts [147, 148]. More recently, the scispaCy [149], and Stanza [150] libraries have provided python interfaces to tackle a wide variety of NLP tasks in the biomedical domain, including tokenisation, lemmatisation, POS, dependency parsing, NER and EL. These libraries come with pre-trained pipelines that are particularly fast at inference time, user friendly, and only slightly worse than state of the art models in each task. Therefore, Stanza and scispaCy have been widely adopted in biomedical NLP over the last years and integrated into multiple pipelines requiring different text mining tasks.

Evaluation metrics In information retrieval and extraction, different evaluation metrics might be used to measure the performance of a specific text mining system. The most widely reported metrics include: *Precision*, *Recall*, *Accuracy* and F_1 score. All those metrics are based on the notion of True/False positives and True/False negatives:

1. True Positives (TP): instances from the positive class correctly labelled by the system.
2. True Negatives (TN): instances from the negative class correctly labelled by the system.
3. False Positives (FP): instances misclassified as positive examples.

4. False Negatives (FN): instances from the positive class missed by the systems.

Then, the following metrics can be derived from the confusion matrix:

$$Precision = \frac{TP}{TP + FP} \tag{1.8}$$

$$Recall = \frac{TP}{TP + FN} \tag{1.9}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1.10}$$

$$F_\beta = \frac{(\beta^2 + 1) \cdot Precision \cdot Recall}{(\beta^2 \cdot Precision) + Recall} \tag{1.11}$$

Where β is a parameter that can be adjusted to empathise the influence of either recall or precision in the final metric. In text classification tasks, an intuitive option might be to use accuracy as an evaluation metric, which is the fraction of instances that have been correctly classified. However, in many NLP settings, the proportion of positive and negative instances is often highly imbalanced (e.g. IR, NER), and accuracy can be highly biased towards the most frequent classes [151]. For instance, in binary classification, if one had 90 negative and 10 positive samples, a system tuned to classify all instances as negative would have an accuracy of 90%, but this approach would not detect any positive samples. For this reason, precision and recall are useful metrics when the goal is to retrieve a specific (positive) class. Depending on the task, maximising either precision or recall might be preferable, but it is often desired to balance both metrics. For this reason, F_β (also called *F measure*) is often used to find a trade-off between precision and recall by performing a weighted harmonic mean in which values $\beta < 1$ empathise precision while $\beta > 1$ empathises recall. The balanced *F measure* with $\beta = 1$ is commonly used across the literature and is often referred to as F_1 score. This harmonic mean is preferred over the arithmetic mean to detect cases in which either precision or recall is very low [151]. In contrast, when precision and recall differ greatly, the harmonic mean is always closer to the minimum, penalising low values of both metrics. It is worth noting that ranking and retrieval systems also use precision@k and recall@k, which consider the relevant instances and coverage of the top k items returned by the system [152]. Additionally, measures such as Normalized Discounted Cumulative Gain and Mean Average Precision are commonly used as metrics in information retrieval [153]⁶. When evaluating a classifier performance, one will also find other metrics such

⁶The reader is referred to Chen et al. [153] for a review of the relationship between loss functions and retrieval evaluation metrics.

as the area under the Receiver Operating Characteristics (ROC) curve or the Matthews Correlation Coefficient⁷.

1.3.3 Related work and research gap

Retrieving and extracting PK information from the literature presents some characteristic challenges compared to other biomedical IE tasks. For instance, PK parameter mentions might need to be identified together with their associated numerical expressions (e.g. “The estimated amoxicillin clearance (Cl) was $0.32 \pm 0.1 L/kg/h$ when 800 mg/day were administered”) or drug mentions and their related dose regimen (e.g. “The estimated amoxicillin clearance (Cl) was $0.32 \pm 0.1 L/kg/h$ when 800 mg/day were administered”). Processing these numerical expressions is generally a non-trivial task due to the large number of possible expressions and the presence of alphanumeric characters reporting multiple types of information (e.g. ranges, central and deviation measures, units, frequencies). Additionally, values might be reported in the same sentence referring to different drugs doses, PK parameters and other entities, making the dependency parsing and relationship extraction particularly challenging (e.g.⁸ “*CL and V(d) of the typical individual in this study population (PMA = 34.6 weeks, weight = 1.7kg) were estimated to be $0.066lh^{-1} kg^{-1}$ (95% CI 0.059, $0.073lh^{-1} kg^{-1}$) and $0.572lkg^{-1}$ (95% CI 0.505, $0.639lkg^{-1}$), respectively.*”). Additionally, very little work has focused on developing text mining systems within the PK domain, which makes recognising and linking specific entities (e.g. parameter mentions, units, values) and their relationships particularly challenging.

Kinetic parameters Some text mining research has tackled similar issues for biochemical kinetic systems (e.g. enzyme kinetics) that frequently report parameters from ordinary differential equations. [113]. Since the underlying models are relatively similar to those in PK, parameter mining studies are likely to present similar challenges. Hakenberg et al. 2004 [155], used a ML approach to classify full-text PDF documents depending on whether they contained kinetic parameters obtained experimentally, and they reported a F_1 score of 54% on 791 manually-annotated publications. Other works such as the KiPar algorithm [156] applied rule-based approaches to retrieve and characterise relevant publications to enzyme kinetic parameters based on entity mentions, which exhibited a F_1 of 65% on document retrieval using full-text articles from PMC. In 2009, Tsay et al. [157] developed a system (KIND) to extract kinetic parameter data from articles published at ScienceDirect [158]. Their system relies on a dictionary and rule-based approaches to label the kinetic parameter mentions and disentangles

⁷ For a comprehensive review of evaluation metrics in information retrieval, see Chapter 8 of Manning 2008 [151].

⁸ Example taken from the abstract at Marqués-Miñana et al. [154].

the different types of information reported in alphanumeric expressions. They first detected relevant sentences with a rule-based approach after POS tagging and reported a F_1 score of 81% on sentences from 200 annotated articles. Then, another set of rules was manually constructed to mine specific kinetic parameter mentions and their values, units, boundaries, and other related information in the sentence. The authors reported a relatively high averaged F_1 of 82.2% on the IE task.

Pharmacokinetic text mining The main publication focusing on PK parameter extraction is that of Wang et al. 2009 [11]. Initially, the authors constructed a dictionary of terms based on expert knowledge to retrieve PubMed abstracts reporting PK parameters of a specific drug (midazolam) in healthy human volunteers. The dictionary summarises key factors to determine whether an abstract is relevant, and they reported a F_1 score of 78.1% when using it to filter the corpus resulting from the PubMed query “midazolam AND pharmacokinetics”. The dictionary-based approach was compared against training a ML classifier, which exhibited lower performance on abstract classification (F_1 score = 68.1%). Subsequently, they constructed specific term dictionaries to identify dosing, subjects and PK parameter information from the abstracts. Then, rule-based approaches were applied to extract numerical values referring to systemic and oral Cl of midazolam in healthy humans. After extracting numerical Cl values from the selected abstracts, a linear-mixed model meta-analysis approach was applied to remove potential outliers from the extracted values⁹. With this approach, the authors reported a final F_1 score of 91% on numerical Cl extraction for midazolam in healthy volunteers. According to the authors, applying the same approach on seven different drugs led to a similar performance. Despite these promising results, the IE metrics were based on the number of abstracts from which Cl data had been automatically extracted, and no details regarding the performance at detecting and standardising parameter mentions, values, ranges or units were found. This approach aimed to detect a specific PK parameter (Cl for midazolam) in a specified context (healthy human volunteers), but the feasibility and time required to adopt this approach for different contexts (other parameters, drugs, species, conditions) is yet unclear. Finally, no analyses from full-text articles were performed, and the application to ADME database construction was not studied.

An area where text mining has been applied to pharmacokinetic literature is in the context of drug-drug interactions [43, 159, 160]. When given in combination, the absorption, distribution, metabolism and excretion processes of one drug can be affected by another drug. Hence, several drug-drug interactions reported in the literature are described by mentioning how the PK parameters of one drug might change by the influence of another drug. As a consequence, detecting mentions of pharmacokinetic parameters can

⁹ For this, they assumed a normal distribution of midazolam Cl values at the article and population (across all articles) levels and used an expectation-maximisation algorithm to estimate the mean and variance of each distribution.

be particularly relevant when seeking evidence of causal mechanisms behind drug-drug interactions [43]. However, most of the text mining work that has been developed in this area tackles the identification of the drugs mentioned in a particular interaction and whether the interaction between these drugs happens at the PK or PD level [159]. As a result, except from the PK Ontology developed by Wu et al. [43], most of the tasks tackled by text mining of drug-drug interactions are not directly transferable to the challenge of extracting numerical estimates of PK parameters for constructing ADME datasets.

Finally, another study related to PK text mining work is the MPhil thesis of Piliouras [4]. Piliouras 2014 discussed the issue of extracting numerical estimates of PK parameters in depth. However, the main contributions of the thesis focus on (1) methods for improving the recognition of drug mentions in pharmacokinetics and (2) parsers that can disambiguate the linguistic dependencies between terms in sentences expressing PK interactions between drugs, improving the extraction of such evidence in the scientific literature. However, despite the importance of recognising drug mentions for extracting PK estimates in context, many PK parameter estimates are not reported in the context of drug-drug interactions. Therefore, multiple challenges (e.g. recognising mentions of multiple PK parameters and their relation to numerical estimates and their units) were left for exploration to reach the goal of developing systems that can aid the construction of ADME datasets in drug development.

1.3.3.1 Research gap

Overall, very few studies have applied text mining to extract PK information from the literature. Previous approaches to parameter IE have approached this task with dictionary- and rule-based methods. This is likely to be a consequence of the singular tasks these studies were tackling (e.g. retrieving a specific parameter (e.g. *Cl*) in a specific context (e.g. intravenous administration in healthy volunteers)), where building rules and dictionaries are efficient approaches. However, the kind of PK data required to build comprehensive ADME datasets needs to account for a higher degree of variability (e.g. multiple PK parameters, units, chemical compounds). For instance, it is common to look for other PK parameters (e.g. *bioavailability*, *AUC*, $t_{1/2}$), chemical compounds, routes of administration or other study design factors when building repositories of ADME data [10]. However, with the text mining approaches that have been applied to PK text up until now (dictionary- and rule-based), one would need to construct new rules and dictionaries to account for each PK parameter or study-design context. Additionally, despite a large number of PK publications reporting results and demographic information in tables and full text, the extraction of PK information has focused, so far, on the information reported in abstracts¹⁰.

¹⁰Only one study using full-text and tabular data for PK extraction was found at Wang's PhD thesis [113].

Novel approaches based on NLP and ML have been increasingly used across biomedical text mining applications to deal with the high variability of textual patterns and their influence on the underlying semantics. However, these approaches often require large amounts of high-quality labelled data, which is limited in the PK domain. Therefore, future work is required to develop annotated resources and models that can efficiently process PK literature.

1.4 Research aim, questions and thesis outline

This thesis aimed to enhance the curation of pharmacokinetic datasets by automatically structuring estimates reported across the scientific literature.

Previous work on this domain has focused on extracting a single type of parameter for a specific population and dose regimen, but the generalisation to multiple PK parameters and study types has not been tackled yet. Therefore, this thesis focused on approaches that can generalise to multiple PK parameters and study designs found across the literature. Specifically, the chapters of this thesis discuss the limitations of previous work and suggest improvements for tackling multiple types of parameters and study designs.

In the short term, it is expected that this thesis contributes to (1) the field of PK and drug development, by accelerating the construction of ADME datasets and (2) the field of biomedical text mining, by developing novel approaches and resources to process PK literature. In the long term, automatically extracting information for multiple parameters across study types has the potential of improving preclinical predictions of novel compounds, PK meta-analyses and providing informative priors for pharmacometric model building.

Based on the limitations of existing methods, the following research questions were defined to achieve the aim of this thesis:

- Q₁ How can text mining approaches be applied to identify scientific publications reporting novel PK parameter estimates automatically across study types?
- Q₂ How can the task of recognising parameter mentions of multiple parameter types be accomplished?
- Q₃ How can the specific types of PK parameters be determined given mentions in context?
- Q₄ How can suitable architectures be developed and evaluated to extract PK parameter estimates of multiple parameter types?
- Q₅ How can text mining approaches accelerate the curation of ADME datasets used by pharmacometricians?

Thesis outline Each of the research questions were addressed in-depth in each of the thesis chapters.

Chapter 2 introduced the reader to ML and NLP approaches used across this thesis.

Chapter 3 addressed the task of document classification of PK articles (Q₁).

Chapter 4 studied the development of named entity recognition models for multiple PK parameters while minimising the amount of training data required (Q_2).

Chapter 5 studied limitations and potential approaches to ground PK parameter mentions to a knowledge base with limited training data (Q_3).

Chapter 6 focused on extracting structured information for PK estimates of multiple parameters reported in scientific text by formulating the task in an end-to-end relation extraction setting (Q_4).

Chapter 7 constructed a database of PK estimates with the approaches developed in this thesis, and studied its application in a case study requiring the extraction of multiple parameters and drugs of interest (Q_5).

In each chapter, specific objectives were defined and addressed after reviewing previous work and potential improvements, and they were described in the introductory sections.

Chapter 2

Models in Natural Language Processing

This chapter provides an introduction to the main ML approaches used in NLP, focusing on those models implemented across the different chapters of this thesis.

Before textual data can be processed by ML models, the input text needs to be converted into numerical vectors of fixed size. These vectors are often called *representations* and contain characteristic features of the input text. To perform this conversion, the first step in NLP tasks is tokenisation and consists of splitting the input text into smaller textual units called tokens, which might refer to words, sub-words or characters. A finite vocabulary of tokens is often defined *a priori*, and each token in the vocabulary has an associated identifier. At the beginning of NLP pipelines, each token *id* is often mapped to a fix-sized vector using a look-up table. The main methods used to associate tokens to their vectors can be broadly divided into context-independent and distributional approaches.

2.1 Context-independent

2.1.1 Bag-of-Words

One of the most common and simple ways to represent tokens is through one-hot vectors. In this setting, each token *id* corresponds to a position in an n -dimensional vector, where “ n ” refers to the vocabulary size. To represent the i -th token in the vocabulary, a vector of size n is generated with a one on the i -th dimension and zero on the rest. To represent a sequence of tokens (e.g. sentence, paragraph, document), one of the most common methodologies is the bag-of-words (BoW) approach [161], which generates n -dimensional sequence representations by adding all the one-hot token vectors in a given sequence. The resulting vectors of BoW contain information about the frequency of tokens in the input sequence.

This approach has been widely adopted in document retrieval tasks where the occurrence of certain terms might be enough to determine the retrieval of specific documents. BoW also provides additional benefits such as interpretability (each feature corresponds to one token), speed, and the fact that the length of the resulting vectors (n) is independent to

the length of the input document. However, this representation has several limitations: (1) it completely ignores the relative position of tokens in the document, (2) it generates long feature vectors due to the large size of input vocabularies, and (3) the resulting document representations are highly sparse since most documents typically use a small subset of the tokens in the vocabulary.

N-grams Since the BoW model does not consider any order in the input sequence, all tokens are treated as independent entities. The *n-gram* approach aims to encapsulate some spatial information by generating groups of tokens (of size n) that appear sequentially in the text. For instance, in the bigram model the unigram list: [“Mary”, “likes”, “to”, “watch”, “movies”] would be converted into [“Mary likes”, “likes to”, “to watch”, “watch movies”]. This might be a useful approach to encode information about concepts formed by more than one token (e.g. [“renal”, “clearance”], [“area”, “under”, “curve”]). However, one of the main disadvantages of this approach is its direct impact on the size of the feature vector, which highly increases when bigrams or trigrams are added to the unigram token list.

2.1.2 TF-IDF

Another common approach to represent documents in a given corpus¹ is to perform *term frequency-inverse document frequency* (*TF-IDF*) standardisation of the BoW vectors. This approach aims to attenuate the effect of terms that appear with high frequency across the corpus and give higher relevance to those terms that only appear in a few documents [162]. In this setting, the term frequency (*tf*) scores encoded by BoW are multiplied by the following *idf* scores:

$$idf(t) = \log \frac{N + 1}{df(t) + 1} \quad (2.1)$$

$$tf-idf(t, d) = tf(t, d) \cdot idf(t) \quad (2.2)$$

In which N is the total number of documents in the corpus and $df(t)$ is the number of documents that contain a particular term t . Then, the frequency of each term t in each document d , $tf(d, t)$, is multiplied by its $idf(t)$ to generate the $TF-IDF(t, d)$ score. The constant one is added to the numerator and denominator of equation 2.1 to prevent zero divisions when new terms from new documents are encoded. The $idf(t)$ weight is comprised between 0 and 1 and takes lower values for those terms that appear in a high proportion of documents and higher values for infrequent terms across the corpus.

¹ Collection of documents.

2.2 Distributional

Instead of representing tokens as independent textual units, distributional semantics refers to a family of techniques to represent the meaning of tokens (often words) based on their linguistic contexts of use. The central hypothesis of distributional semantics is that words appearing in similar contexts tend to have similar meanings [161]. Driven by this hypothesis, various models have been proposed to generate representations of tokens based on their contexts.

2.2.1 Count-based

Distributional approaches often generate vectors of much lower dimensionality than the vocabulary size that provides similar word vectors for words that appear in similar contexts. To do so, count-based models perform two steps [1]: (1) construct a word-context matrix, and (2) reduce its dimensionality. The word-context matrix often has one word per row, and columns represent all potential contextual words. The cells are filled with the frequency of contextual words in a given corpus within a specific window size around the central word. Since word-context matrices contain many columns (as many as potential words) and many words only appear in a few possible contexts (producing many zeros in the matrix), dimensionality reduction is applied to reduce the size and sparsity of word vectors. Different count-based approaches have been used, including:

Co-occurrence Counts One of the simplest and most common approaches is to define a context window of size L around each word. Then, a cell $c_{i,j}$ in the co-occurrence matrix is filled by counting how many the contextual word j appears along the central word i within a window of L words [163]. Despite its simplicity, one of the main drawbacks of this approach is the sparsity of the resulting matrix.

Positive Pointwise Mutual Information (PPMI) In PPMI, the co-occurrence matrix has the same rows and columns as co-occurrence counts, and the window size also needs to be defined. However, PMI compares the probability of two words co-occurring (joint probability, $P(a,b)$) to the likelihood that those words were independent (dot product of marginal probabilities $P(a) * P(b)$) through the following equation:

$$PMI(a,b) = \log\left(\frac{P(a,b)}{P(a)P(b)}\right) \quad (2.3)$$

The PPMI of a specific word is then filled by taking the maximum between 0 and $PMI(a,b)$. When the ratio is 1 (log equals 0), the words only co-occur by chance but do not form a unique concept. On the other hand, if words have a low probability

of independent occurrence in the corpus but a high probability of occurring together, they are likely to form a unique concept. This approach attenuates the effect of common words across the corpus and enhances the representation of concepts in the co-occurrence matrix. PPMI was considered one of the state-of-the-art approaches before neural architectures [164].

Latent Semantic Analysis (LSA) LSA is a count-based model used to analyse relationships between documents and the terms they contain by producing a set of topics related to the documents and terms. LSA starts by computing the term frequency matrix mentioned in section 2.1.2, which counts the number of words in each document. Then, Singular Value Decomposition (SVD) [165] is applied to reduce the dimensionality of the matrix. In LSA, document representations are generated such that the vectors of similar documents have a high cosine similarity [166].

2.2.2 Prediction-based

In contrast to count-based approaches, prediction-based methods learn word vectors by improving the predictive ability of specific ML models. In distributional semantics, predictive tasks often focus on enforcing the models to model the relationship between words and their contexts of use. This thesis will mainly use distributed representations learned by neural networks since they have been shown to preserve linear regularities between words significantly better than previous approaches such as LSA [167].

2.2.2.1 Feed-forward neural networks

Over the last years, neural networks have become the leading architecture behind distributional semantics. Neural networks are ML models structured with layers of interconnected neurons (also called perceptrons) that propagate information through the network. A classical feed-forward neural network is shown in Figure 2.1, which is also known as Multilayer Perceptron (MLP). The vertical layer of orange neurons is known as the input layer and contains the input features given to the model. The layers of grey neurons refer to the hidden layers that propagate the information through the network until reaching the output layers (blue neurons), representing the model's output values. Each neuron receives a series of inputs (incoming arrows) and combines them to generate a single output (outcoming arrows). Each input has an associated weight. A neuron performs a weighted sum of its inputs and applies a non-linear transformation (activation function) to the result, generating an output value [168]. If each neuron in a hidden layer is connected to all neurons in the following layer, this is known as a feed-forward or fully-connected layer.

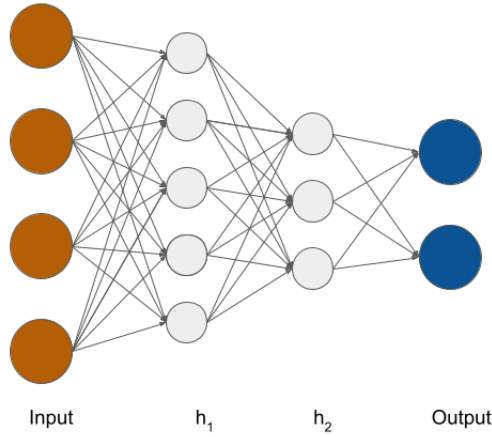


FIGURE 2.1: Graphical representation of a neural network with 4 layers. The input layer is represented in orange and composed by 4 neurons, whereas the output layer is represented by two blue neurons. Two hidden layers are exemplified in grey.

In mathematical notation, the values in every layer of neurons are often represented as a single vector. For instance, in Figure 2.1, the input layer can be represented with a vector X of 4 dimensions and the next layer (h_1) as a vector of 5 values. To generate h_1 , a fully-connected layer performs a multiplication between the input vector (X) and the matrix of weights relating each input-output neuron (W_1). Often, a bias vector (b_1) is added after the vector-matrix multiplication. Finally, the activation function (g) is a non-linear transformation applied to each output value:

$$h_1 = g(W_1X + b_1) \quad (2.4)$$

In the following sections of this thesis, the collection of all weights ($W_{1..n}$) and biases ($b_{1..n}$) of a neural network will be defined as the network parameters θ .

One of the most widely-used activation functions is the sigmoid or logistic function:

$$g(z) = \frac{1}{1 + e^{-z}} \quad (2.5)$$

However, a wide variety of activation functions have been used to train neural networks, such as Hyperbolic Tangent (Tanh) or Rectified Linear Activation (ReLU).

Model training Given a series of input features and expected labels (i.e. training examples), neural networks are trained to minimise the *distance* between their outputs and the desired labels. This *distance* is often measured with a *loss function*, also called *objective function* or *error measure*. An example of a *loss function* is the *mean squared error*, defined as:

$$J(\theta) = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i(\theta) - y_i)^2 \quad (2.6)$$

Where n refers to the number of training examples, \hat{y}_i to the network’s output for the example i , and y_i to the expected labels. There are a variety of loss functions used to train neural networks, which are suited to different applications [73].

To update the weights of a neural network, gradient decent methods are often used to update the model parameters (θ) so that they minimise the loss function ($J(\theta)$). The first step is to compute the gradient of the model parameters’ respect to the loss function $\nabla_{\theta}J(\theta)$. The algorithm used to compute $\nabla_{\theta}J(\theta)$ in neural networks is known as *back-propagation* [169] which relies on the *chain rule of calculus* to compute the partial derivate of each model parameter respect to the loss function².

When the gradients are computed, at each training iteration, the model parameters are updated in the following manner:

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_{\theta_t}J(\theta_t) \quad (2.7)$$

Where η refers to the learning rate and controls the magnitude of the updates. As shown in equation 2.6, the loss function is defined as the average loss over all training examples (n). However, computing the gradient $\nabla_{\theta}J(\theta)$ at every update for all the training examples is an expensive computational operation. Therefore, a common approach is to update the model parameters using a mini-batch of examples, often known as stochastic gradient descent. In practice, the whole set of training examples is split into a series of mini-batches. For every mini-batch, the model parameters are updated by computing the mini-batch loss and its associated gradient. A single pass and update of the mini-batch through the model is defined as a *training step*, and an entire pass of the whole training data through the model (all mini-batch updates) is referred to as *epoch*.

2.2.2.2 Word2Vec

Training neural networks using Language Modelling (LM) objectives has become the main approach to generate distributed representations of tokens. In LM, a model is often asked to predict the probability of a token (or sequence of tokens) given their context tokens or *vice versa*. One of the earliest and most successful applications of this idea is *Word2Vec*, developed by Mikolov et al. 2013 [167]. In essence, *Word2Vec* is a neural network model whose parameters are word vectors. Given a large text corpus, *Word2Vec* uses a sliding window to generate combinations of central and context word pairs (see Figure 2.2).

² For further details on the *back-propagation* algorithm see Ruder 2019 [73].

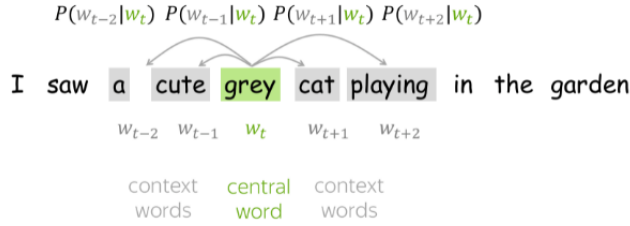


FIGURE 2.2: Illustrative example of the sliding window used in *Word2Vec*. Taken from Voita 2020 [1].

Word2Vec trains a neural network with a single hidden layer to perform an “auxiliary” task. The main LM objective of *Word2Vec* is to either: (1) predict the probabilities of context words given a central word (*Skip-Gram* variation, Figure 2.3a) or (2) predict the probability of a central word given the context words (*CBOW* variation, Figure 2.3b).

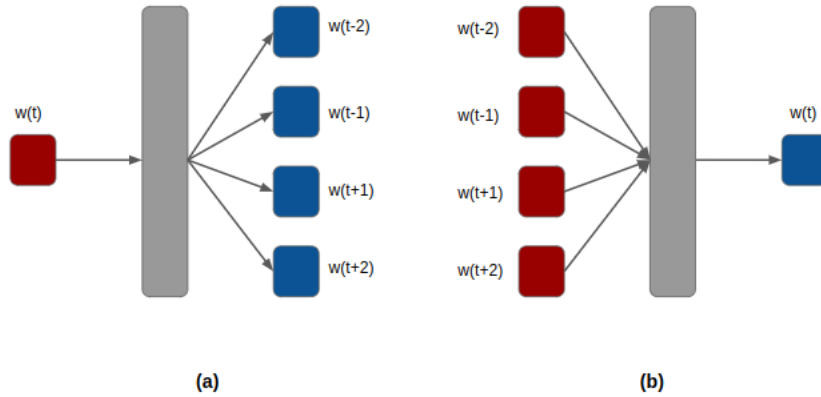


FIGURE 2.3: Schematic representation of the *Skip-Gram* (a) and *CBOW* (b) architectures from *Word2Vec*.

For each position in a text corpus $t = 1, \dots, T$ the *Skip-Gram* model is trained to minimise:

$$J(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{-m \leq j \leq m} \log P(w_{t+j}|w_t, \theta); j \neq 0 \quad (2.8)$$

where m is the window size and $P(w_{t+j}|w_t)$ is the probability of a context word given the central word. Given a vocabulary of words V , a central word vector v_c and a context word u_o , $P(o|c)$ is computed using the softmax function:

$$P(o|c) = \frac{e^{u_o^T v_c}}{\sum_{w \in V} e^{u_w^T v_c}} \quad (2.9)$$

Word2Vec uses stochastic gradient descent to update its parameters so that they minimise the loss function in equation 2.8. In practice, instead of considering all the possible

context words for a given central word, negative sampling is performed for efficient training and to balance the frequency distribution of words in the corpus. As a result, the similarity of the central word is only computed against the correct context word and a specific number of negative (non-context) words. This approach avoids the denominator in equation 2.9 to add all the words in the vocabulary V , but only the positive and selected negative samples for a specific central word.

This training procedure can be performed in a self-supervised fashion since almost unlimited instances of *context - central word* pairs can be automatically generated without manual labels. By training *Word2Vec* on large amounts of data, the weights of the network are adjusted to represent the context of words, and therefore, encode valuable semantic information into a relatively small dimensional space – note that the number of neurons in the hidden layer will determine the size of the embeddings. One of the main advantages of this approach is that once *Word2Vec* has been trained on a large corpus of text, the learnt word vectors can be used as initial representations for downstream applications, which exhibited significant improvements in multiple NLP tasks [167]. The idea of using models pre-trained on large amounts of unlabelled data as starting points for other supervised tasks is known as *transfer learning* which has been widely used in the NLP domain after the appearance of *Word2Vec*.

The following sections present the fundamental neural network architectures used to process textual data and used across this thesis.

2.2.2.3 Recurrent models

Since natural language text is sequential, most of the models used in NLP consider processing sequences of inputs. Therefore, a single training sample is often represented as a sequence of token vectors. The most common architecture for this task is the Recurrent Neural Network (RNN) [170]. RNNs take an ordered list of input vectors x_1, \dots, x_n (e.g. token vectors) together with an initial state vector s_0 , and compute an ordered sequence of output vectors y_1, \dots, y_n and cell states s_1, \dots, s_n [168] (see Figure 2.4). At each time step i , the network receives an input vector x_i together with the previous cell state s_{i-1} , which stores information about the previous inputs. Broadly speaking, simple³ RNNs are feed-forward neural networks accepting an extra input vector (cell state, s_{i-1}) retaining information of previous steps and propagating its current state (s_i) to the following time step. As shown in Figure 2.4, it is noteworthy that the parameters of the network remain constant over each time step and are only updated at the end of each mini-batch (analogous to feed-forward neural networks) [168].

To compute y_i and s_i , the RNN performs two main computations:

³Without gate mechanisms like LSTMs or GRUs.

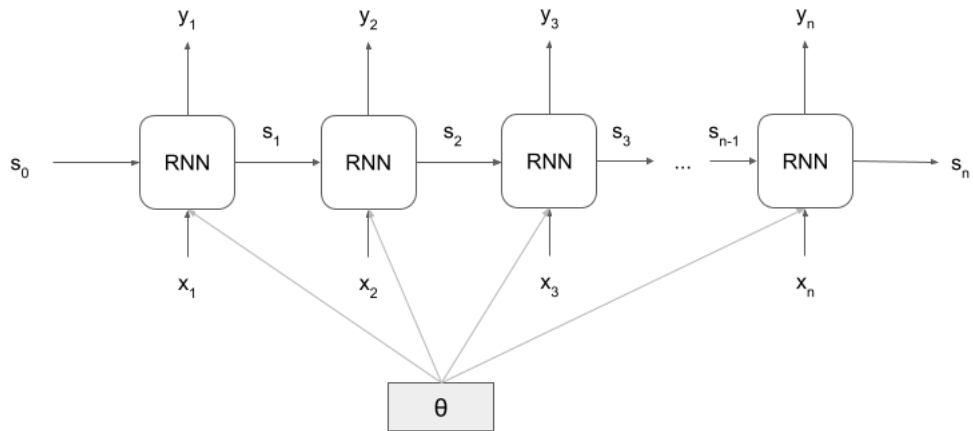


FIGURE 2.4: Schematic illustration of an RNN where θ represents the network parameters, and x_i , s_i and y_i represent the input, cell state, and output vectors at a particular time step.

$$s_i = f_1(s_{i-1}, x_i) \quad (2.10)$$

$$y_i = f_2(s_i) \quad (2.11)$$

Where f_1 is often a feed-forward layer accepting the input vector at time i (x_i) and previous cell state (s_{i-1}) and returning a vector that represents the current cell state (s_i). The f_2 is the operation used to map the state vector s_i to the desired output vector for that time step (y_i).

This architecture allows framing a large number of NLP tasks. For instance, in sentence classification tasks, sentence tokens are provided as sequential inputs, and the last output vector (y_n in Figure 2.4) is often treated as the model prediction. In NER tasks that require one label per token, it is common to use each of the RNN output's (y_1, \dots, y_n) and calculate the loss based on the token-level labels. In machine translation, the final cell state s_n is often used as an input to a decoder that will use s_n as an input vector containing the required information for the whole input sentence to be translated. It is noteworthy that at time i , the network does not have any information about subsequent tokens, which might be required for some tasks. For this reason, it is common to use bidirectional recurrent models (e.g. biRNN) [171] which consist of two RNNs, one processing the sentence left-to-write, and another one right-to-left. For a specific time step i , the output vectors of each RNN are concatenated to include contextual information from both sides of the sequence.

One common issue when training RNNs in practice is the *vanishing or exploding gradient* problem [172]. In RNNs, the hidden state is multiplied by the weight matrix of the neural network at every time step. When calculating the gradient using *back-propagation*, this results in the gradients being multiplied by the same values multiple times, which can

cause the gradient values to either become very large (*explode*) or exceedingly close to zero (*vanish*). This phenomenon makes it hard for simple RNNs to capture long-range dependencies since the gradient in later steps of the sequence quickly diminish in the *back-propagation* process. Architectures using memory cells in RNNs have been widely adopted to deal with *vanishing/exploding* gradients. These approaches contain “gate mechanisms” that decide what needs to be remembered/forgotten by the network and allow preserving gradients across time [168]. As a result, the most successful RNN architectures currently used in practice are LSTMs [173] and Gated Recurrent Units (GRUs) [174] which use gate mechanisms that attenuate the effect of *vanishing/exploding* gradients.

2.2.2.4 Transformers

The best-performing models tackling multiple NLP tasks before 2017 often consisted of initialising token-vectors pre-trained with variants of *Word2Vec* and using recurrent architectures to model the desired tasks (e.g. Xu et al. [175], Lin et al. [176]).

Despite LSTMs and GRUs enabled a better gradient flow when processing sequential inputs, as the length of the input sequence increases, recurrent models struggle to capture long-range dependencies between tokens in a sentence. This limitation is present in any recurrent architecture since their cell state vectors are modified at every step. Consequently, when recurrent networks reach tokens at the end of long sequences, they often rely on the cell state vector to preserve relevant information that might have appeared at the beginning of the sequence. However, it is often difficult to preserve this and other relevant information through long sequences in a single vector representation. Additionally, recurrent models cannot be easily trained in parallel due to their sequential nature [177]. To address these issues, Bahdanau et al. 2014 [178] and Luong et al. 2015 [179] introduced the concept of *attention* to approach machine translation tasks. The *attention* mechanism allowed recurrent models to focus on relevant hidden states of the input sequence at each decoding step during translation tasks.

In 2017, Vaswani et al. 2017 [177] presented the Transformer, which is a neural architecture relying on *attention* mechanisms without any recurrent models. Since their appearance, Transformers have become the state-of-the-art architecture for modelling sequential textual data due to their speed (most computations can be executed in parallel) and high performance when modelling long-range dependencies. The core idea behind *attention* is to generate a mechanism by which, given a sequence of representations (e.g. cell states), can optimally combine those input representations into a new vector that will be used for a specific task (e.g. produce a new token representation). The Transformer uses a specific type of *attention*, so-called *self-attention*, that given a sequence of n input vectors, returns a new sequence of n output vectors that contain information about their surrounding vectors. Unlike *Word2Vec* where each token

relates to a single, context-independent vector, this mechanism allows generating token embeddings that are dependent on their surrounding tokens. The Transformer was first presented to tackle machine-translation tasks using an Encoder-Decoder structure. In machine translation, an Encoder is often used to generate token representations of the input sequence and the Decoder to predict a new sequence of tokens in the target language.

2.2.2.5 BERT

Building upon these developments, Devlin et al. 2018 presented Bidirectional Encoder Representations from Transformers (BERT) [2]. BERT was designed to generate powerful token representations that could be used or easily fine-tuned for a variety of downstream NLP tasks. To do so, the model uses a stack of Transformer Encoder blocks to convert a sequence of n input tokens into a sequence of n contextual token vectors. Then, similar to *Word2Vec*, the model is pre-trained in a self-supervised manner on a large corpus of text using LM objectives that force the model to encode relevant semantic information for each token. Specifically, BERT is pre-trained to perform two self-supervised tasks: (1) “masked language model” (MLM) where the model is asked to predict 15% of randomly masked tokens, and (2) “next sentence prediction”, that given two sentences the model has to predict whether they appear sequentially in the original text [180]. Once pre-trained, the parameters of BERT can be re-used as a starting model to represent tokens and fine-tuned for specific supervised tasks (*transfer learning*) without the need of using a large number of task-specific layers [2]. Overall, BERT exhibited state-of-the-art results in multiple NLP tasks. It has been shown that the token representations learned by BERT often require little adaptation for downstream applications, reducing the need for labelled data or designing complex task-specific architectures in NLP. Since BERT has been used or fine-tuned for multiple tasks within this thesis, this section describes the fundamental blocks behind it.

Tokeniser and vocabulary BERT uses a sub-word tokenisation approach that relies on the principle that common words should not be broken down into smaller tokens while rare words should be segmented into reusable tokens (sub-words). This approach allows representing unseen/rare words at inference time by encoding their sub-word tokens while preserving frequent words as single tokens (this avoids very long input sequences). Specifically, BERT uses the WordPiece tokenisation algorithm [181] with a vocabulary of 30,000 tokens. Additionally, at the beginning of each token sequence, there is a special classification token included [CLS], which attempts to encode sequence-level information into a single vector, and a [SEP] token to specify the separation between sentences (Figure 2.5).

Architecture Once the input sequence has been split into tokens, BERT maps each input token to a vector representation (Embedding layer Figure 2.5) and adds positional encodings to provide information on the relative position of each token in the input sequence. Then, the model goes through several Encoder blocks and finally returns a sequence of token embeddings. Initially, Devlin et al. [2] released BERT-base, and BERT-large, consisting of 12 and 24 Encoder blocks, respectively. As shown on the right panel of Figure 2.5, each Encoder has two main sub-layers: *self-attention* and a fully-connected feed-forward network [177]. Additionally, the “Add & Normalise” layers represent residual connections that add the input block to its output and perform layer normalisation⁴ [182] to ease the flow of the gradient through the network [1].

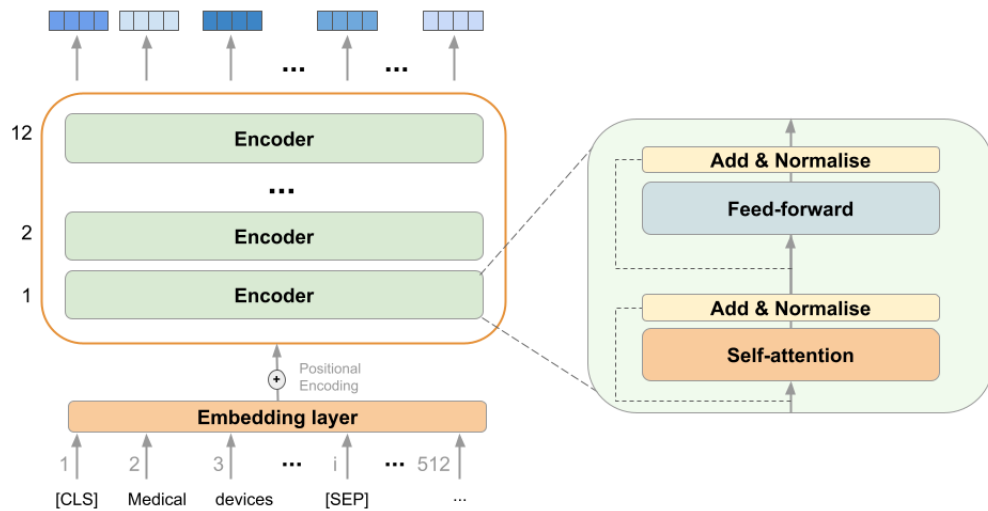


FIGURE 2.5: Graphical representation of BERT-base accepting a sequence of tokens that go through twelve encoder blocks and return a sequence of vectors. Each Encoder represents a Transformer Encoder block displayed on the right panel. [CLS] is a special token added at the beginning of each sequence and [SEP] was originally used to separate sentences [2].

Self-attention As previously mentioned, the *self-attention* operation aims to map a sequence of token embeddings (x_1, \dots, x_n) to another collection of token embeddings (y_1, \dots, y_n) that have been updated according to their context. To implement this idea, each input token (x_i) is mapped to three different representations: query (q_i) , key (k_i) and value (v_i) vectors. Each of these vectors is generated by multiplying the input vector (x_i) by three independent weight matrices (W^Q, W^K, W^V) that are *backpropagated* during the training phase. For a given input word i , the second step consists of computing a score against every other input, determining how much focus to place on other parts of the input sequence (so-called attention scores). All the scores then go through a softmax layer, value vectors are multiplied by their scores, and all the resulting vectors are added together (i.e. weighted sum). In practice, this is computed as a sequence

⁴ See Ba et al. [182] for details on layer normalisation.

of matrix operations for faster processing. If X is the matrix containing all the input vectors, the first operation is:

$$Q = X \cdot W^Q \quad (2.12)$$

$$K = X \cdot W^K \quad (2.13)$$

$$V = X \cdot W^V \quad (2.14)$$

Where Q , K and V are the matrices containing each input's query, key and value representations, respectively. Then, all the outputs of the *self-attention* operation can be obtained through a single equation:

$$Z = \text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2.15)$$

Where d_k is the dimension of the queries and key vectors and Z has the same number of rows as X and represents the resulting token vectors after *self-attention*. In the implementation of the Transformer, the attention operation is performed multiple times (h) independently (attention heads) to allow the model to focus on different aspects of the surrounding vectors. Different weight matrices ($W_{1,\dots,h}^Q, W_{1,\dots,h}^K, W_{1,\dots,h}^V$) are assigned to each attention head and the resulting vectors ($Z_{1,\dots,h}$) are concatenated and multiplied by a final weight matrix W^o . This process is known as *multi-head attention* and enables the model to simultaneously attend to data from different representation subspaces for each input token [177].

2.3 Non-neural models

This section describes ML models used in this thesis for text classification tasks (chapter 3) that did not use neural networks. The models are exclusively described in the context of classification tasks and assume a fixed number of input features.

Logistic Regression Logistic regression models the probability that a response y_i belongs to a particular category y_i given the input features x_i with a *sigmoid* or *logistic function* [183]:

$$\hat{p}(y_i | x_i; \theta) = \frac{e^{\beta_0 + \beta x_i}}{1 + e^{\beta_0 + \beta x_i}} \quad (2.16)$$

In which β_0 and β^5 represent parameters θ that will be estimated from the training data. Note that the possible outputs from equation 2.16 are compressed between 0 and 1, which allows treating the output of logistic regression as a class probability. Logistic regression uses the principle of *maximum likelihood estimation* (MLE) [184] to estimate the optimal θ given the training data. MLE aims to bring the probabilities of the model $\hat{p}(x)$ as close as possible to the empirical probability of the input data $p(x)$ by maximising the log-likelihood function. In supervised binary classification, the empirical probability can be expressed as the label $y_i \in \{0, 1\}$, and the log-likelihood function becomes [183]:

$$\sum_{i=1}^n y_i \log \hat{p}(x_i; \theta) + (1 - y_i) \log(1 - \hat{p}(x_i; \theta)) \quad (2.17)$$

In which n is the total number of training examples. Since maximising equation 2.17 results in the same than minimising the negative log-likelihood, it is common to use the average cross-entropy as a loss function [73]:

$$\mathcal{L}(\theta) = \frac{1}{n} \sum_{i=1}^n -y_i \log \hat{p}(x_i; \theta) - (1 - y_i) \log(1 - \hat{p}(x_i; \theta)) \quad (2.18)$$

Optimisation methods are then used [183] to obtain the parameter combination that minimises $\mathcal{L}(\theta)$ in a specific dataset.

Decision Trees Decision trees are non-parametric algorithms used for supervised classification and regression tasks. The main idea behind this algorithm is to build a model that predicts the target variable y_i based on simple decision rules learned from the input features. As it can be observed in Figure 2.6, decision trees split the feature space X into different regions R_i that will characterise the response variable y . Figure 2.7 shows the basic elements and nomenclature of a decision tree. Different tree algorithms have been developed with this idea [183].

In classification tasks, the goal is to divide the feature space into regions R_i in which the majority of instances belong to a unique class. Decision trees use an iterative *top-down* approach to achieve this goal. Starting at the root node (Figure 2.7) (when all instances belong to the same region), the feature X_j and splitting cut-point s that produce the best subregions are selected at each iteration to generate two new branches. This is known as *recursive binary splitting* [184] and requires a metric to measure the “purity” of a particular region and assess the quality of a split. A commonly used metric is the Gini index:

⁵Note that β will be a vector of coefficients with the same dimensionality than x_i and βx_i represents the dot product.

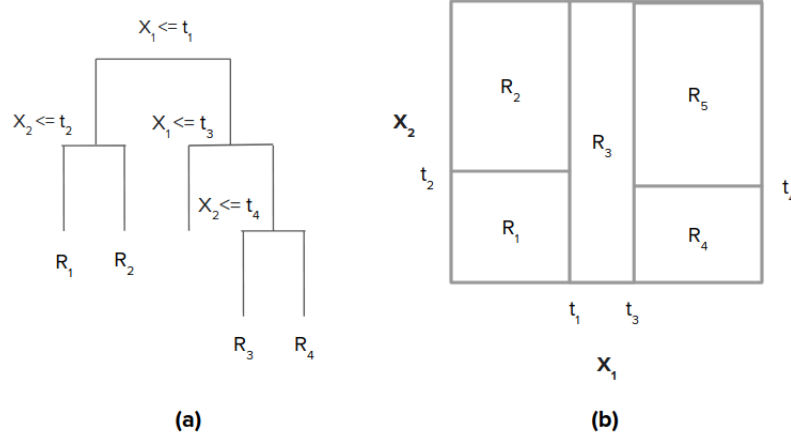


FIGURE 2.6: (a) Decision tree splitting two features X_1 and X_2 at different points t . A 2D visualisation of the regions generated by these splits is observed in the right hand-panel (b)

$$G = \sum_{k=1}^K \hat{p}_{mk}(1 - \hat{p}_{mk}) \quad (2.19)$$

Where \hat{p}_{mk} represents the proportion of training instances that belong to the k_{th} class. From equation 2.19 it can be noticed that G takes small values when most instances of a particular region belong to a single class (\hat{p}_{mk} close to 1). The G from a parent node is compared to the weighted (by the number of samples) average of the G of the resulting two regions to evaluate the quality of a particular split. Then, the feature X_j and cut-point s that produce the largest reduction in G is selected for a split. The determination of the best splitting point for a particular feature tends to be a fast computation, and by scanning through all the features, finding the optimal (j,s) is feasible [183]. This splitting procedure continues for each branch until a stopping criterion is reached, and the last nodes become leaf nodes. If no stopping condition is imposed a priori, the decision tree grows until all the leaf nodes are pure ($G = 0$). However, this approach might split the feature space into overly small regions and overfit the training data.

Different stopping criteria can be imposed to limit the tree's growth and prevent overfitting, such as specifying a maximum depth, a minimum decrease in Gini index to perform a split or a minimum size for the leaf nodes.

Extreme Gradient Boosting Despite their excellent interpretability and ease of train, decision trees do not tend to perform as good as other ML algorithms. One of the main drawbacks is their high variance and instability. In addition, the nature of the hierarchical-building process propagates decisions produced in top nodes down to the child nodes. This fact causes slight variations in the input data can result in a very different sequence of splits [184]. However, ensemble methods have achieved robust

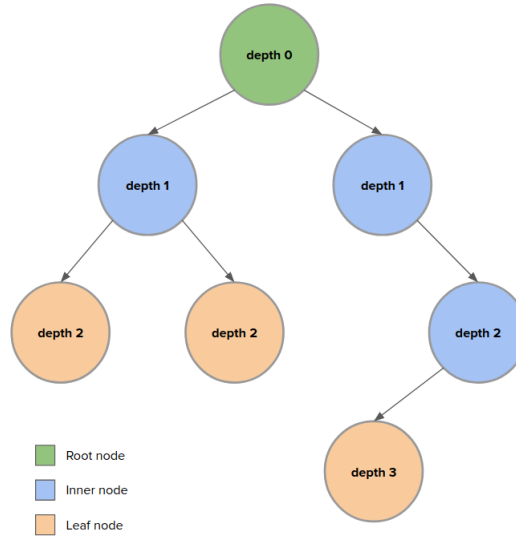


FIGURE 2.7: Example of a decision tree of depth 3.

results in different tasks by aggregating multiple decision trees under a single classifier. Extreme Gradient Boosting (XGBoost) is a boosting method that has been successfully applied in a variety of tasks [185–187].

Boosting belongs to a specific class of ensemble learning methods in which multiple weak learners (e.g. decision trees) are combined to produce a stronger model. How different weak learners are combined defines the type of ensemble method (i.e. bagging, boosting, stacking). Although different models can be used as weak learners, decision trees have been the most widely adopted [183]. Bagging approaches average the decision of multiple weak learners independently fitted to subsets of the dataset, which reduces the variance of the strong learner. In contrast, boosting methods use a sequential approach in which each new weak learner is fitted to focus on those observations that were misclassified by the previous models [188]. This approach aims to reduce the bias of the strong learner, and depending on the aggregation approach, two main boosting methods are found: Adaboost [189] and Gradient Boosting. XGBoost is an advanced implementation of Gradient Boosting developed by Chen et al., 2016 [190]. Boosting methods try to build a strong model $H_L(x)$ as a weighted sum of L weak learners $f_l(x)$:

$$H_L(x) = \sum_{l=1}^L \eta \cdot w_l \cdot f_l(x) \quad (2.20)$$

In which w_l is the weight associated with the new weak learner. The parameter η (learning rate) scales the contribution of each weak learner by a constant value. The η parameter is often treated as a hyperparameter⁶ and smaller values require a larger

⁶ Hyperparameters refer to parameters of the objective function that are not fitted during the training process and need to be specified before fitting the model.

number of weak learners to maintain a specific training error. However, directly finding the optimal combination of parameters (and the number of weak learners) in the ensemble model represents an intractable optimisation problem [188]. For this reason, boosting methods build an iterative process by which weak learners are built one by one based on the previous state of the ensemble:

$$H_l(x) = H_{l-1}(x) + \eta \cdot w_l \cdot f_l(x) \quad (2.21)$$

Where w_l is the weight associated with the new weak learner. In gradient boosting, a new weak learner ($f_l(x)$) is fitted to the negative gradient of the loss function in each iteration [188]. The implementation in XGBoost has several characteristics that have made it particularly efficient in ML applications. These include L1 and L2 regularisation, scalability, sparsity handling and optimisation techniques that make it extremely fast compared to other gradient boosting algorithms [190]. The regularisation term is often applied to the decision tree's parameters and can be expressed as follows:

$$\Omega(f_t) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2 \quad (2.22)$$

Where T is the number of leaves, w_j is a coefficient associated with each leaf, and γ and λ are hyperparameters that can be adjusted for regularisation.

Chapter 3

Pharmacokinetic Document Retrieval

Most content in this chapter has been published at Gonzalez Hernandez et al. [191].

3.1 Introduction

A significant challenge in the curation of ADME datasets is to efficiently collect relevant literature regarding the PK parameters of different compounds. According to Nawaz et al. [192], the PubMed database is growing at a rate of two articles every minute, which complicates and lengthens the process of finding relevant literature. Current search systems are not specific enough to limit query results to those publications reporting PK parameters obtained from *in vivo* studies while ensuring high recall rates [11]. When looking for PK parameters of a particular compound, generic search queries (e.g. “amoxicillin AND pharmacokinetics”) are often performed as a first step, and large amounts of time need to be invested in finding the relevant documents through a number of unrelated publications [193]. This chapter addresses the challenge of efficiently finding those scientific articles publishing PK results derived from *in vivo* data. Centralising studies reporting PK estimates would not only accelerate the curation of ADME datasets, but could also provide a benchmark repository for comparison and evaluation of PK results and provide an initial set of relevant documents for further IE tasks.

The PubMed search engine makes use of the MeSH vocabulary and words appearing in the title and abstract to perform queries relevant to certain biomedical topics. For instance, the MEDLINE indexing includes the MeSH term “Pharmacokinetics”¹ together with a number of sub-related terms (Figure 3.1). However, as previous studies suggest [11], the retrieval of documents with this term results in a high recall but low precision rate for *in vivo* PK data. Since scientific search engines tend to provide low precision rates in document retrieval of specific biomedical fields, a variety

```
graph TD
    A[All MeSH Categories] --> B[Phenomena and Processes Category]
    A --> C[Physiological Phenomena]
    A --> D[Pharmacological and Toxicological Phenomena]
    D --> E[Pharmacokinetics]
    E --> F[Absorption]
    E --> G[Area Under Curve]
    E --> H[Biological Availability]
    E --> I[Biotransformation]
    E --> J[Chronopharmacokinetics]
    E --> K[Cutaneous Elimination]
    E --> L[Drug Liberation]
    E --> M[Hepatobiliary Elimination]
    E --> N[Intestinal Elimination]
    E --> O[Lacrimal Elimination]
    E --> P[Lacteal Elimination]
    E --> Q[Metabolic Clearance Rate]
    E --> R[Pulmonary Elimination]
    E --> S[Renal Elimination]
    E --> T[Salivary Elimination]
    E --> U[Therapeutic Equivalency]
    E --> V[Tissue Distribution]
```

FIGURE 3.1: MeSH indexing of the term “Pharmacokinetics” and its related sub-terms.

¹ <https://www.ncbi.nlm.nih.gov/mesh/68010599>

of studies have applied additional text mining techniques for biomedical IR [155, 156].

In the field of PK, only one text mining publication performing document retrieval was found. As previously mentioned, Zhiping et al., 2009 [11] applied an entity template to PubMed abstracts to perform NER of drugs, routes of administration, specific PK parameters and species. Subsequently, simple rules were applied to the tagged documents to determine whether a particular publication reported PK parameters of midazolam in healthy volunteers. Initially, this entity template was applied to the corpus resulting from the PubMed search “midazolam” (n=7129) and resulted in 43% precision and unreported recall. However, after adding the term *pharmacokinetics* to the original query (“midazolam AND pharmacokinetics”, n=819), the authors reported performance of 91.5% recall, 68.2% precision and F_1 of 78.1%. In the second corpus, the authors manually labelled the resulting 819 PubMed abstracts, from which they identified 20% as being relevant. They also compared the performance of the entity template against a supervised ML approach, in which a support vector machine was trained to perform the binary classification task. The ML approach resulted in a F_1 of 68.1% when trained with a balanced corpus of 100 samples. However, the small number of publications available for training and the minimal exploration of the features encoded might have significantly limited the performance of this approach.

This chapter presents an automated pipeline to identify and characterise scientific publications reporting *in vivo* PK parameters across different study types (e.g. animal/human studies, healthy/non-healthy subjects, different drug regimens).

Due to the heterogeneity of the biomedical literature and the requirement for a high-quality corpus of PK articles, supervised ML approaches were studied to find scientific publications that reported *in vivo* PK parameters. Two main motivations were found to use ML approaches over rule-based systems for this task. (1) Building comprehensive rules that can distinguish documents reporting novel *in vivo* PK parameters becomes particularly challenging to implement due to the multiple linguistic and structural cues involved, which expand over different parts of the text [194]. (2) Additionally, posterior probabilities of ML models could be used for search engines to rank scientific publications depending on their likelihood of reporting PK parameters, prioritising relevant documents to be scanned in the curation of ADME datasets. Hence, based on Q₁ of this thesis, the following objectives were formulated:

- O₁₋₁ Develop an expert-annotated dataset that can be used to train and evaluate models that distinguish publications reporting *in vivo* PK parameters.
- O₁₋₂ Study the effect of different encoders and decoders for classifying scientific publications reporting PK parameter estimates.

O₁₋₃ Apply the optimal classifier to PubMed publications to generate an automatically-updated and centralised repository of PK papers.

O₁₋₄ Deploy the collection of retrieved documents through a search interface where users can search for relevant PK data.

3.2 Methods

The methodology behind this study was divided in the following main sections:

1. **Corpus development:** manual annotation of scientific articles as “Relevant” or “Not Relevant” by field experts depending on whether they reported *in vivo* PK parameters.
2. **Pipeline development:** Study of different NLP pipelines to optimise the retrieval of relevant publications.
3. **Large-scale application:** Deployment of the optimal pipeline to classify a large corpus of PK literature.

The code developed in this chapter can be found at <https://github.com/PKPDAl/PKDocClassifier>.

3.2.1 Corpus development

3.2.1.1 Source

The search “pharmacokinetics” in PubMed returns over half a million entries from which $\approx 20\%$ report *in vivo* PK parameters. Therefore, being able to retrieve relevant publications from this corpus would provide a large collection of potentially valuable PK literature for ADME dataset curation and subsequent IE tasks. Only the keyword “pharmacokinetics” was used as a query since most of other relevant PK terms were included in the MeSH hierarchy under the “pharmacokinetics” node ². A PubMed search for “pharmacokinetics” was conducted without additional filtering, and the resulting list of PubMed IDs was obtained (October 2020). All papers used in this study were sampled from this distribution of papers.

² <https://www.ncbi.nlm.nih.gov/mesh/68010599>

3.2.1.2 Size and criteria

Two collections of documents were created to train, compare, and assess various classification approaches: the *training* and *final test* sets. The *training* and *final test* sets each contained 3992 and 800 randomly selected (without replacement) articles from the list of PubMed Ids downloaded, respectively. The size of each set was established based on the availability of expert annotators, ensuring that each document had at least two independent labels. The following criteria were used to label the articles: If a publication’s title, abstract, tables, or full-text³ contained newly estimated measurements of PK parameters from *in vivo* data (human or animal), the publication was labelled as “Relevant” by the annotators. Publications that did not include estimates of PK parameters, reviews, or papers that only mentioned PK parameters from other studies were deemed “Not Relevant”. Only those original articles in which PK parameters were reported along with their contextual information were considered “Relevant”.

3.2.1.3 Annotation

Two clinical pharmacists and two pharmacometricians from University College London’s Pharmacometrics Group, as well as one clinical pharmacist from the London Health Sciences Center, all of whom had considerable experience in pharmacometric modelling, contributed to the annotation process. Initial annotation guidelines were established⁴, and a training session was conducted before annotations started. Each document in the *training set* was first labelled by two annotators, and each document in the *final test* was initially labelled by at least three annotators. Disagreements were then discussed with the annotation team. Exceptions and divergent views on labelling criteria emerged during the labelling process, and when each instance was resolved, the annotation guidelines were updated accordingly. The Cohen Kappa Coefficient (K , Eq. 3.1) was initially calculated on 100 documents randomly selected from the *final test* set and labelled by all annotators [195] to determine the degree of agreement between annotators. K compares the observed agreement between two annotators assigned to any sample (p_o) against the expected agreement by chance (p_e)⁵:

$$K = \frac{p_o - p_e}{1 - p_e} \quad (3.1)$$

Guidelines were iteratively updated during the labelling process until a high level of pairwise consistency amongst annotators ($K > 0.9$) was achieved, which resulted in 2-5 reviews per document.

³ Despite the full-text article was considered for labelling purposes, note that the only fields used in the classification pipeline are those described in section 3.2.2.1 (without utilising tabular and full-text data).

⁴ <https://github.com/PKPDAl/DRGuidelines.github.io>

⁵ p_e is estimated using a per-annotator empirical prior over the class labels [196].

After the final guidelines were generated, pair-wise F1 between annotators was also computed on the initial 100 documents to assess differences between model performance and inter-annotator agreement. This exercise established an approximate ceiling performance for our models given by the average pair-wise F1 across annotators. To compute pair-wise F1 score, the labels of one annotator were treated as ground truth and the other as the system prediction⁶. Finally, the average pair-wise F1 between each pair of annotators was computed as a ceiling performance metric.

Results on inter-annotator agreements were presented in section 3.3.2.

3.2.2 Pipeline development

3.2.2.1 Parsing

Since full-text and tabular information are not always accessible, the classification pipelines were trained using textual data from the title, abstract, and other PubMed metadata. However, unlike the full-text and tabular data, this information is publicly available for most papers in PubMed. Therefore, developing a retrieval system without utilising the full-text information of papers would expand the scope of the pipeline’s applicability.

To obtain the desired textual information from the labelled documents, the selected PubMed publications were downloaded in XML format through the baseline⁷ and updates⁸ FTP sites. These sites store the information from PubMed entries indexed in XML files. For each paper, the information from multiple fields (e.g. title, abstract authors, MeSH terms and others) was parsed from the XML files into a dataframe. An analysis was later performed to select those fields that improved the classification performance (see details in section 3.2.2.4).

The XML files provide different fields under a series of tags. For instance, in Figure 3.2 it can be observed how the textual information from the title and abstract of a particular PubMed entry is stored under the tags `<ArticleTitle>` and `<AbstractText>`. Hence, rules need to be applied to extract the desired textual information from the XML files and store it in a structured format. For the XML extraction PubMed Parser [197] was used. This python module is frequently used for researchers in biomedical NLP and considers XML tags for all our desired fields. The extracted textual information was stored in a dataframe format for subsequent analyses.

⁶ Note that switching roles would not alter the F1 score [234].

⁷ <ftp://ftp.ncbi.nlm.nih.gov/pubmed/baseline/>

⁸ <ftp://ftp.ncbi.nlm.nih.gov/pubmed/updatefiles/>

```

<ArticleTitle>Determination of 3-(5-tetrazolyl) thioxanthone
10,10-dioxide in human plasma, urine and faeces.</ArticleTitle>
<Pageination>
  <MedlinePgn>93-9</MedlinePgn>
</Pageination>
<Abstract>
  <AbstractText>A gas chromatographic method is described for assay of
3-(5-tetrazolyl) thioxanthone 10,10-dioxide (BW 59C) in human plasma,
urine and faeces. After extraction into 1,2-dichloroethane from
alkaline medium the compound is converted to the heptafluorobutyrate
derivative which is injected into a gas chromatograph and measured
using a 63Ni electron capture detector. The assay produces a linear
calibration curve over the range 0-30 mug/ml when the internal
standard method is used. Reproducibility is good and sensitivity down
to 1 ng injected on column is possible. The method has been used to
investigate the pharmacokinetic properties of BW 59C in man and has
been semi-automated by the use of an autosampler and dedicated
computer.</AbstractText>
</Abstract>

```

FIGURE 3.2: Example of XML tags for title and abstract in PubMed files.

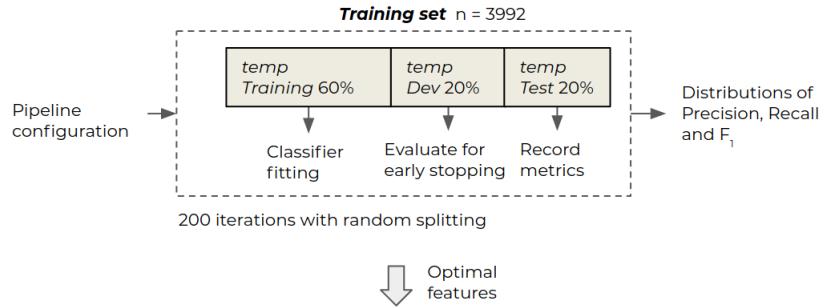
3.2.2.2 Evaluation

During the pipeline development stage, the impact of various pipeline setups was investigated to maximise the classification performance of “Relevant” articles. Analyses were conducted to determine the influence of various classifiers, textual input fields, document representations, and pipeline hyperparameters⁹. Precision (P), Recall (R), and the F_1 score (the harmonic mean of P and R) were used as evaluation criteria to compare the different architectures. The following analyses were sequentially performed to determine the optimal architecture: (1) Classifier comparison, (2) Field selection, (3) N-grams, (4) Distributed representations, and (5) Final pipeline. The first analysis aimed to determine the classifier with the best performance. The second, third, and fourth analyses attempted to pick the best performant document representations for the classification task. Due to the scarcity of training samples and consequent variability on the F_1 , a bootstrapping strategy was used to establish a distribution of metrics for each pipeline configuration (Figure 3.3 A). To prevent bias in the evaluation, the *training set* was randomly divided into *temp training* (60 percent), *temp dev* (20 percent), and *temp test* (20 percent) with stratified sampling using scikit-learn’s python module. The *temp training* was used to fit the classifier, *temp dev* was used to assess the classifier during training and to conduct *early stopping*, and *temp test* was used to analyse pipeline performance after each bootstrap iteration. This procedure was performed 200 times for each pipeline configuration to obtain a distribution of metrics. After identifying the optimal document representations, the whole *training set* was used to choose the optimal hyperparameters, which were then applied to the *final test set* to provide the final metrics (Figure 3.3B). It is worth noting that during the first analysis (Classifier comparison), no early stopping was performed since the default hyperparameters were

⁹ Parameters supplied by the user and not implicitly learned during the training phase.

used for each classifier. As a result, during the Classifier comparison the *training* set was split into 80% *temp training* and 20% *temp test*.

A) Bootstrap, used for: (2) Field selection, (3) N-grams and (4) Distributed embeddings



B) 5-fold CV, used for: (4) Final Pipeline

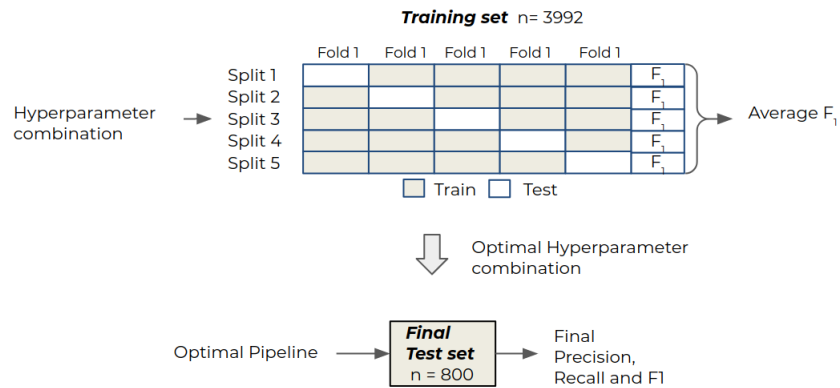


FIGURE 3.3: A) Bootstrap procedure to compare the effect of different features during field selection, n-grams and distributed representations analyses. B) The best-performing features from previous analyses were selected to compare different hyperparameter combinations with 5-fold cross validation. Finally, the best-performing features and hyperparameters were used to apply the pipeline to the final test set.

3.2.2.3 Classifier comparison

The first analysis compared supervised learning models and selected the best-performing ones for the following steps. In this classification task, three classifiers were compared: logistic regression, decision tree and XGBoost. During this analysis, the default hyperparameters from each classifier were used according to their scikit-learn implementation. During this analysis, only the title and abstract were encoded through bag-of-words.

To account for possible prediction biases caused by class imbalance, the weights associated with each sample during the training stage were set inversely proportional to the class frequency on the corpus. In other words, if only 20% of the documents in the training set were “Relevant”, the loss associated with each “Relevant” sample was scaled by a factor of five. This loss weighting technique efficiently eliminated spurious solutions in which predictions were skewed toward the most frequent class.

Logistic regression Logistic regression was analysed as a classifier. In this experiment, L2 regularisation was applied on the model parameters θ , and stochastic gradient descent [198] was used to update the parameters iteratively. Average cross-entropy was used as a loss function. For one training example, i with input features x_i and label y_i , the loss was computed as:

$$\mathcal{L}(\theta)_i = -y_i \log \hat{p}(x_i; \theta) - (1 - y_i) \log(1 - \hat{p}(x_i; \theta)) \quad (3.2)$$

Decision Tree The CART algorithm [199] was used in this study through scikit-learn’s¹⁰ implementation. In this analysis, the Gini index was used to measure the quality of splits and the default values defined in scikit-learn’s implementation were used for the rest of the model hyperparameters. As previously mentioned, different weights were given to samples labelled as Relevant or Not Relevant.

Extreme Gradient Boosting In this experiment, the algorithm XGBoost was analysed as a decoder. Decision Trees were used as weak learners over which the XGBoost was trained to minimise the average cross-entropy loss. Finally, *early stopping* was employed, where the maximum number of iterations was set to 2000 and boosting was stopped earlier if the F_1 performance on the validation (*temp dev*) set did not improve after 100 iterations. This approach was used to prevent overfitting by limiting the number of boosting iterations through an external evaluation (*temp dev* set) and, in turn, limiting the complexity of the model.

¹⁰<https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>

3.2.2.4 Field-selection

After selecting the best-performing classifier, this experiment aimed to detect the most useful fields for the classification task. The following fields were initially retrieved from the PubMed XML files:

- Title of the publication
- Abstract
- List of authors
- Journal of publication
- Publication type: Journal Article, Review etc
- Keywords from the abstract
- MeSH terms
- List of chemicals annotated and indexed by MEDLINE
- Author’s affiliations

Table 3.1 displays the availability of these fields in the *training* and *final test* datasets. The importance of the abstract was first determined by comparing the performance when the title was used alone against the combination of title and abstract. The significance of each field in the metadata section was then studied by comparing its performance when combined with the title and abstract. Finally, those fields that showed an improvement in classification performance against only title + abstract were combined.

During the classifier comparison and field selection analyses, documents were encoded into sparse vectors using a BoW approach (see section 2.1.1 of this thesis for details). Each column in the document vector was filled with the frequency of each term in the input document, and the resulting vector was divided by the document’s total number of tokens to standardise it by the document’s length (L1 vector norm). Each document was tokenised before applying the BoW count. Without further pre-processing, each author, journal, publication type, keyword, MeSH term, chemical and affiliation were handled as single tokens since they are standardised terms in MEDLINE. However, tokenisation is more complex for the *title* and *abstract* since they are extracted in string format. One of the main challenges is that biomedical text significantly differs (in terms of grammar and vocabulary) from general-domain English, and common tokenisers might not consider specific characteristics of biomedical tokens [200]. For this task, the rule-based tokeniser from ScispaCy [149] was used. ScispaCy adds tokenisation

TABLE 3.1: Summary statistics reporting the percentage of documents in which a particular field was available, and the proportion of papers labeled as **Relevant** and **Not Relevant**. The statistics are reported for both training and final test sets.

Field	Training	Final test
Title	100	100
Abstract	87.17	87.67
Authors	99.44	99.63
Journal	100	100
Publication Type	100	100
Keywords	15.41	16.125
MeSH terms	97.67	98.25
Chemicals	93.86	94.13
Affiliations	80.94	79.125
Label		
Relevant	19.81	20.25
Not Relevant	80.19	79.75

rules on top of the general-domain spaCy’s rule-based tokeniser to improve the tokenisation of biomedical text¹¹. Once tokenisation was performed, further pre-processing and cleaning steps were applied to improve the quality of tokens and reduce the vocabulary size (which directly impacts the number of features):

- Initially, all tokens were lowercased.
- Punctuation signs were removed from token units (e.g. “2.2” becomes “22”). Then, tokens that were entirely composed by digits were standardised to “##”. This approach attempted to encode the frequency of numerical information in an abstract but not the specific digits reported. In addition, after some exploration, it was observed that a substantial number of tokens included a numerical value with the units attached without white-spaces (e.g. “43.3l/min”). To separate these cases, tokens starting with a digit value were split into numeric and non-numeric tokens before the digits were standardised. For instance, the token “43.3l/min” becomes [“##”, “lmin”] after pre-processing is applied.
- Common English terms that appear with high frequency (called *stopwords*) and are assumed to add little discriminant value (e.g. “the”, “be”, “in”, “are”) were removed. Then, isolated punctuation signs were also removed since they were assumed to be irrelevant for classification purposes. The list of *stopwords* and punctuation signs included in the spaCy’s 2.2 library [201] were used to identify those tokens.

¹¹Details on spaCy’s tokenisation rules can be found at <https://spacy.io/usage/linguistic-features#how-tokeniser-works> and the additional rules for sentence segmentation and tokenisation implemented for biomedical text can be found at https://github.com/allenai/scispacy/blob/master/scispacy/custom_tokenizer.py.

- Chemical mentions were identified using scispaCy’s named entity recogniser (NER) model trained on the BC5CDR corpus and replaced with the token *drugname* to avoid bias toward particular chemical references.
- Stemming: Due to differences in grammatical expressions, documents include different forms of the same word (e.g. pharmacokinetic, pharmacokinetics, pharmacokinetically). In many information retrieval tasks, it is helpful to standardise these words to a unique form since the size of the vocabulary is reduced, and higher frequencies of important base-forms can be obtained [151]. Two main approaches are often used to reduce inflexions and derivations of related forms into a base token, *stemming* and *lemmatisation*. *Stemming* generally refers to the heuristic process that “chops-off” the end of the words by removing derivational affixes (e.g. pharmacokinetic, pharmacokinetics, pharmacokinetically become pharmacokinet), whereas *lemmatisation* uses existing vocabularies and morphological analyses to return a base or dictionary form [151]. Due to its popularity and efficiency, the rule-based Porter’s algorithm [202] was used to stem the tokens in our corpus¹².

Once the tokens from each field were generated and pre-processed, they were joined into a single list of tokens. However, to differentiate the source of each token, an identifier (initial uppercase letter of the particular field) was attached at the end of each token. For instance, the frequency of the token “pharmacokinet” might have different relevance if the token comes from the title or the abstract. Hence, the identifier allows to differentiate both terms in the input vocabulary for BoW (e.g. “pharmacokinet” vs “pharmacokinetT”). After tokens had been unified into a single list, the BoW encoding was applied.

3.2.2.5 N-grams

Since a number of PK parameters are composed by more than one term (e.g. “area under the curve”, “renal clearance”) the effect of using *n-grams* was explored (see section 2.1.1 for details). The best-performing fields from *field selection* were utilised for this study, and the effect of using *bigrams* and *trigrams* from the abstract and title as additional features was compared against using *unigrams*.

3.2.2.6 Distributed representations

Pre-trained BERT architectures were used to construct distributed representations of scientific documents in this experiment (for details on this architecture, see section 2.2.2.5). It is worth noting that the parameters of pre-trained models were not updated

¹²Details on how Porter’s algorithm works can be found at <https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html>.

(fine-tuned) in this study. Instead, they were used as feature extractors keeping all BERT parameters fixed. Two models based on BERT were employed in this experiment: BioBERT and SPECTER.

BioBERT Lee et al., 2019 [122] presented BioBERT, which extended the BERT_{BASE} model by further pre-training it on PubMed abstracts and PMC full-text articles to learn word representations specific to the biomedical domain. Distributed token representations were constructed in this study by combining the final four hidden layers of BioBERT v1.1, resulting in 768-dimensional vectors for each token. The representation of a document was created by concatenating the BioBERT encodings of the title and abstract. Because the number of tokens varied between documents, a composition function transformed multiple token vectors into fixed-length document representations. Two ways were examined to do this, as seen in Figure 3.4. To begin, the mean of all token representations was calculated (*mean pooling*), resulting in 1536-dimensional document representations (768 for the title and 768 for the abstract). The minimum and maximum values (*min&max pooling*) were then computed and combined with *mean pooling* to generate 4608-dimensional document representations. Note that other composition functions have been applied across the literature (e.g. [203]), such as simple addition or weighted average with *idf* scores.

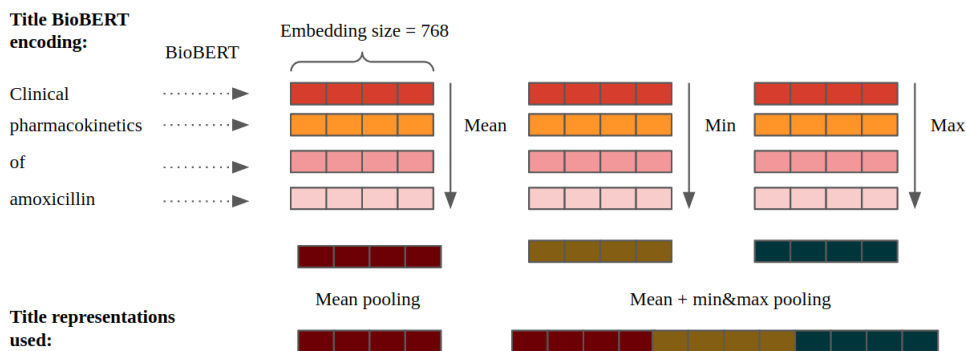


FIGURE 3.4: Example of the approaches used to generate distributed representations for an input title after BioBERT encoding. The same procedure was applied for tokens in the abstract.

SPECTER Rather than employing token-level representations, Cohan et al., 2020 [204] suggested a BERT-based strategy for directly generating document-level representations for scientific papers. SPECTER pre-trains the Transformer using the title and abstract of scientific texts to learn close embeddings for related publications. SPECTER’s pre-training objective is to predict whether or not a given input document cites individual articles. SPECTER encodes the abstract and title of each input publication into a single 768-dimensional vector that serves as the document’s representation during pre-training. The authors reported state-of-the-art performance on seven

document-level tasks [204]. This experiment examined the classification performance by utilising SPECTER document representations as input features for the binary classifier.

3.2.2.7 Final pipeline

The best-performing features from previous analyses were combined into a single model to test whether they provided complementary information. The pipeline’s hyperparameters were then tuned using a five-fold cross-validation (CV) approach combined with an exhaustive grid search through the whole *training set* (Figure 3.3B). For the grid-search analysis, a set of candidate values for each hyperparameter were initially established. The hyperparameters tuned were those from BoW encoder and XGBoost classifier:

1. *min_df*: Minimum number of documents that a specific token should appear to be included in the BoW feature matrix. This hyperparameter has a large impact on the size of the feature vectors generated by BoW.
2. *max_depth*: Maximum depth of each decision tree build in the boosting process.
3. *colsample_bytree*: Proportion of features that each decision tree subsamples at each boosting iteration.
4. *n_estimators*: Number of boosting iterations.

The specified range of values for each hyperparameter is listed in Table 3.2. The learning rate of XGBoost was kept constant at 0.1. The remaining XGBoost hyperparameters were maintained at their default values in the scikit-learn API.

TABLE 3.2: Hyperparameters tuned during cross-validation and their default values. The range represents the different values tested for each hyperparameter in the grid-search procedure. The step size refers to the increase between the starting and stop values.

Parameter	Range (start, stop, step)	Default value
<i>min_df</i>	(2,512,x2)	20
<i>max_depth</i>	(2,64,x2)	4
<i>colsample_bytree</i>	(1/3,1,+1/3)	1
<i>n_estimators</i>	Early stopping	-

3.2.3 Large-scale application

After training the final pipeline, it was used to classify $> 550K$ documents obtained through the PubMed search “pharmacokinetics”. The final pipeline was deployed with Apache Spark for efficient computation and runs weekly updates in Azure Databricks [205]. Using the BERN algorithm [126], the retrieved articles were characterised by the chemicals, diseases and species mentioned in the abstract.

3.3 Results and Discussion

3.3.1 Feature visualisation

A comparison cloud was initially computed to visualise the token frequency distribution in the corpus and study potentially relevant features. This was plotted using the R library *wordcloud* [206] and provides a quick visual inspection of the potential characteristic tokens from each class. The comparison cloud from Figure 3.5 displays the unigram and bigram tokens. The size was plotted proportionally to their relative frequency in the corpus vocabulary, and the vertical position was determined by the proportion of “Relevant” and “Not Relevant” documents in which tokens appeared. As a result, it can be observed how tokens like “*dose*”, “*plasma*” or “*## mg*” have a high frequency in the corpus, but they appear in similar proportions of “Relevant” and “Not Relevant” documents. In contrast, “*pharmacokinetic*”¹³ or “*subject*” appear in a majority of “Relevant” papers. As a result, these tokens are likely to encode important features during the training stages. Finally, high-frequency tokens seemed to appear in a higher proportion of “Relevant” documents (more variation of blue token sizes), which indicates the potential presence of patterns related to word frequencies. In contrast, tokens in the “Not Relevant” area (red), seemed to have a more uniform size, suggesting the absence of a word frequency pattern to characterise “Not Relevant” papers.

3.3.2 Inter-annotator agreement

The initial pairwise K value for the 100 randomly selected articles was 0.68 ± 0.073 (mean \pm standard deviation) [42]. The initial disagreement was primarily due to: (1) the annotator’s accidental omission of “Relevant” instances, (2) discrepancies in the labelling criteria for complicated cases, such as PK studies of endogenous substances, physiologically-based PK studies, and (3) instances where PK parameters were not reported in the abstract, and the full-text was not accessible. Type 1 disagreements were easily identified using the double annotation approach and posterior checking of every disagreed instance. Annotation guidelines were iteratively updated through discussion with the annotation team to minimise conflicts caused by labelling criteria (type 2). Updates were performed at the end of each annotation batch until the inter-annotator agreement exceeded a pairwise K of 0.9. Pair-wise F1 was computed at the end of the annotation process and resulted in 92.01 ± 1.26 across annotators. This was treated as the ceiling performance of our models.

The most complicated examples were those in which the entire text was unavailable, and the abstract was unclear about whether the study estimated PK parameters *in vivo* (type 3). In those instances, extensive checks were conducted across the whole

¹³The capital T indicates that the token came from the title.

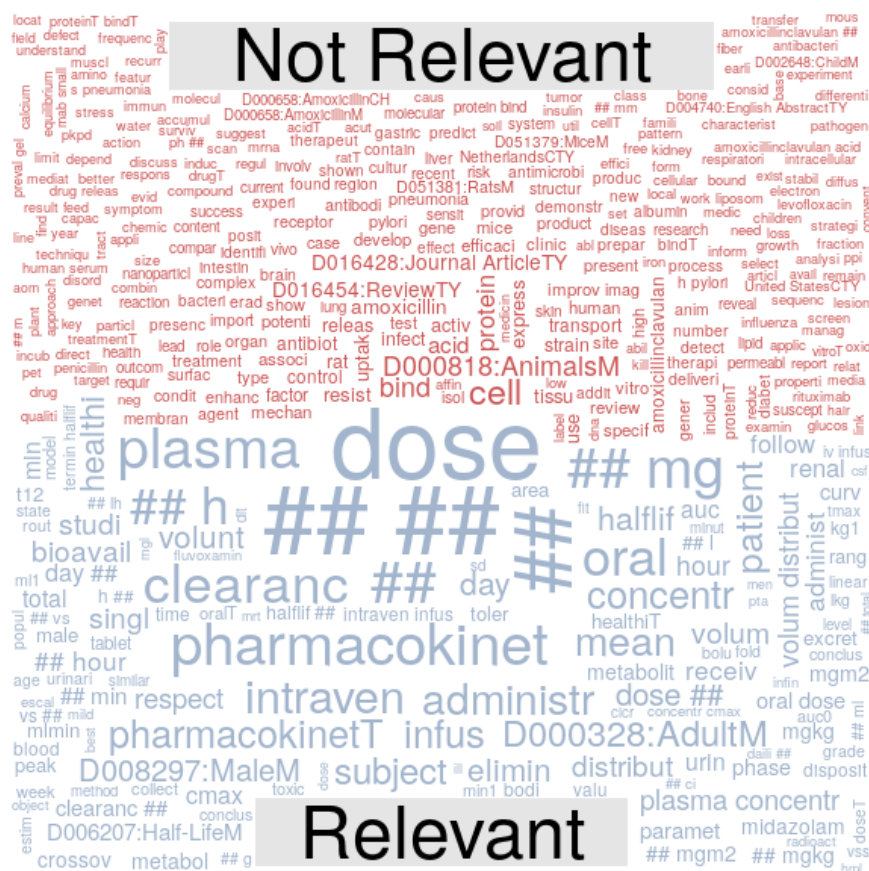


FIGURE 3.5: Comparison word cloud with the unigrams and bigrams obtained after preprocessing each field for bag-of-words encoding. Word size is proportional to the frequency of a token in the vocabulary, whereas the position in the vertical axis refers to the proportion of labels in which that token appears. Note that tokens that are not from the abstract are displayed with a field identifier (e.g. T for tokens from the title or M for mesh terms).

annotation team, and the final label was assigned based on the most frequently used criteria across annotators. Annotation was also particularly time-consuming for those publications that did not report parameter estimates in the abstract, which occurred in over 80% of cases. For those instances, annotators had to find, download and examine the full-text document. As a result, the average annotation time per abstract across annotators was 1.5 minutes without posterior review of disagreements.

3.3.3 Classifier comparison

Table 3.3 and Figure 3.6 display the performance metrics when using unigram features from the abstract and title for Logistic Regression, Decision Tree and XGBoost as classifiers. The best performing classifier was XGBoost, which exhibited over 6% improvement in the median F_1 score in comparison to Decision Tree and Logistic Regression. Although

many hyperparameters and textual field combinations could be studied, XGBoost was the classifier selected for subsequent analyses due to its clear outperformance when using default hyperparameters.

TABLE 3.3: Summary table with performance metrics reported as median (95% CI) and F_1 interquartile variance (IQV) after 200 bootstrap iterations. The performance metrics are compared across pipelines using different classifiers.

Pipeline	Precision (%)	Recall(%)	F_1 (%)	F_1 IQV
XGBoost	71.5 (66.5,76.4)	83.5 (77.8,89.9)	77.0 (73.4,81.6)	8.2
Decision Tree	63.1 (57.9,69.4)	68.7 (59.5,76.6)	66.1 (60.3,70.8)	10.5
Logistic Regression	55.1 (50.6,59.9)	81.6 (76.6,87.4)	65.8 (62.0,70.1)	8.1

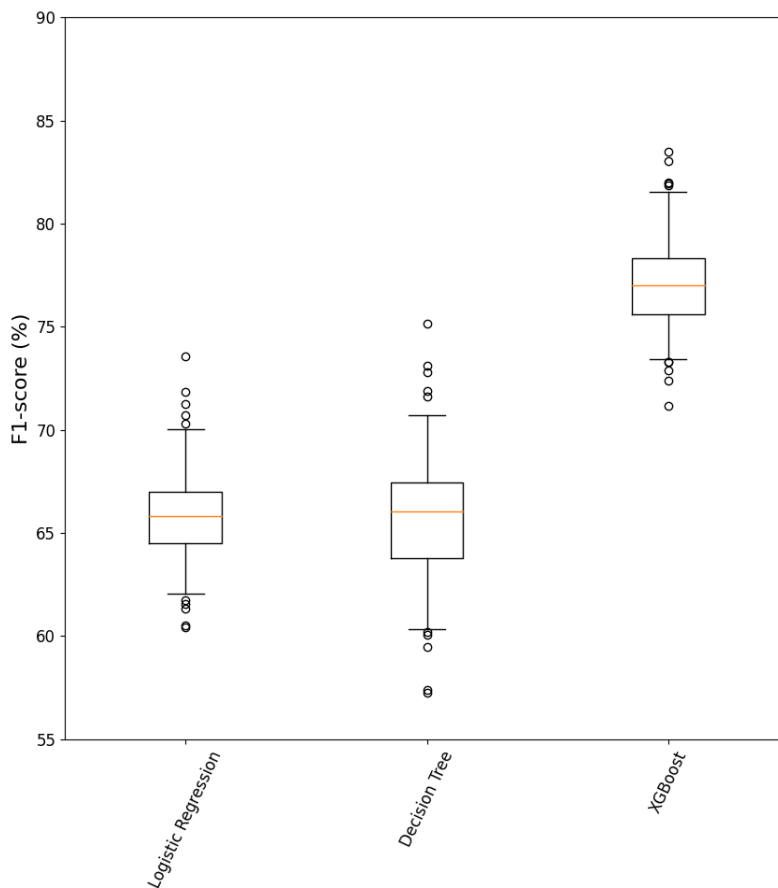


FIGURE 3.6: Distribution of F_1 scores for the different features used in the *classifier comparison* analysis after 200 bootstrap iterations.

3.3.4 Field selection

After 200 bootstrapping rounds on the *training set*, evaluation metrics were calculated to determine the importance of each PubMed field for the classification task. Table 3.4 and Figure 3.7 present the outcomes of this analysis. It was found that combining the abstract and title information (Abstract pipeline, median $F_1=78.2\%$) gave a significant

benefit over employing only the title (Title pipeline, median $F_1=65\%$). Additionally, a number of fields added to the title and abstract did not provide any discriminant information: *chemicals, journal, authors, keywords*. Adding *affiliations* increased the median F_1 score by a very small amount ($\Delta F_1=0.1\%$), but there was no apparent change in the distribution of F_1 scores (Figure 3.7). In comparison to using only the abstract and title, adding the *MeSH terms* and *Publication Type* as extra fields resulted in a significant increase in the distribution of F_1 scores, with a gain of over 1% in the median F_1 . The *Publication Type* was particularly useful for determining if PK articles were Reviews containing PK data from other studies, which facilitated the characterisation of “Not Relevant” publications that had similar word frequencies to the “Relevant” ones.

After the previous observations, the *Title, Abstract, MeSH terms* and *Publication Type* were selected for subsequent analyses and named *optimal fields*. Only using the *optimal fields* exhibited a similar (and slightly higher) performance than the pipeline using all the fields (Table 3.4). Hence, a significant decrease in the feature matrix was obtained while maintaining the classification performance when only using the *optimal fields*. In subsequent analyses, only the *optimal fields* were considered for BoW encoding.

TABLE 3.4: Summary table with performance metrics reported as median (95% CI) and F_1 interquartile variance (IQV) after 200 bootstrap iterations. The performance metrics are compared across pipelines using different fields from PubMed entries.

Pipeline	Precision (%)	Recall(%)	F_1 (%)	F_1 IQV
Title	65.3 (59.0,72.7)	65.8 (55.0,72.8)	65.0 (59.5,71.0)	11.5
Abstract	77.0 (69.8,82.6)	79.8 (73.4,86.1)	78.2 (73.6,82.6)	9.0
Authors*	76.4 (69.4,82.6)	80.4 (72.8,86.1)	78.2 (73.3,82.6)	9.3
Journal*	76.4 (70.2,82.2)	79.8 (72.8,85.4)	78.0 (73.6,82.0)	8.4
Publication Type*	78.0 (71.6,84.3)	81.6 (74.7,87.4)	79.6 (75.5,84.2)	8.7
Keywords*	76.6 (70.2,83.0)	80.4 (72.8,85.5)	78.2 (73.8,82.2)	8.4
MeSH terms*	79.2 (72.1,85.2)	79.8 (72.8,86.1)	79.5 (74.3,83.3)	9.0
Chemicals*	76.0 (69.5,81.9)	80.4 (73.4,86.1)	77.8 (73.0,82.0)	9.0
Affiliations*	76.6 (69.8,82.1)	80.4 (72.8,86.7)	78.3 (73.2,81.9)	8.7
All fields	80.1 (73.0,86.1)	81.6 (74.1,87.4)	80.5 (75.7,84.9)	9.2
Optimal Fields**	80.1 (73.9,86.0)	82.3 (74.1,88.6)	80.6 (75.8,85.2)	9.4

*Tokens from the title and abstract were also included when encoding this field.

**The optimal fields were the title, abstract, MeSH terms and publication type.

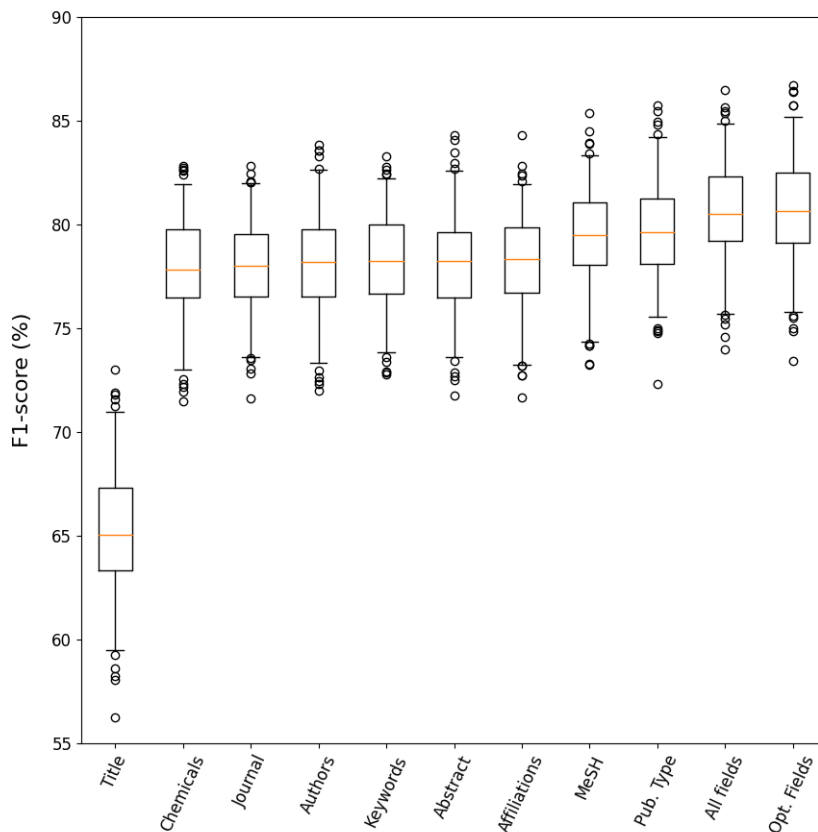


FIGURE 3.7: Distribution of F_1 scores for the different features used in the *field selection* analysis after 200 bootstrap iterations. The fields Chemicals, Journal, Authors, Keywords, Affiliations, MeSH terms and Publication Type were encoded together with the title and abstract tokens. The Optimal Fields include the title, abstract, MeSH terms and Publication Type.

3.3.5 N-grams

The results of BoW encoding utilising bigrams and trigrams are shown in Table 3.5 and Figure 3.8, respectively. *Optimal Fields* from the previous section refers to the same as the Unigrams pipeline in this section. Bigrams and trigrams did not outperform unigrams in this experiment. Bigram and trigram features did not give extra discriminant information despite numerous PK parameters being expressed with more than one term (e.g. distribution volume, maximum concentration, the area under the curve, and systemic clearance, for example). This might be caused by the high diversity of PK parameter mentions, which results in very sparse BoW representations that do not benefit from n-grams. When n-grams are included, the number of features in input document representations goes up significantly. Because of this, the additional information that bigram and trigram features might provide does not seem to be worth the additional expense in dimensionality and sparsity of the feature matrices.

Another analysis that is often performed in document classification tasks is scaling BoW matrices with *TF-IDF* scores. However, since XGBoost used decision trees as weak learners and the *gini index* as the splitting criterion, there was no justification for scaling each feature column with the *IDF* score. The ranking of values in the same column would not be modified after scaling, which would result in the same Gini index per feature column.

TABLE 3.5: Summary table with performance metrics reported as median (95% CI) and F_1 interquartile variance (IQV) after 200 bootstrap iterations. The performance metrics are compared across pipelines using different n-grams from the optimal fields.

Pipeline	Precision (%)	Recall(%)	F_1 (%)	F_1 IQV
Unigrams	80.1 (73.9,86.0)	82.3 (74.1,88.6)	80.6 (75.8,85.2)	9.4
Bigrams	79.9 (72.2,86.9)	81.6 (74.1,88.0)	80.6 (76.2,84.8)	8.6
Trigrams	80.4 (74.4,86.3)	81.0 (73.4,88.0)	80.6 (76.7,84.6)	7.9

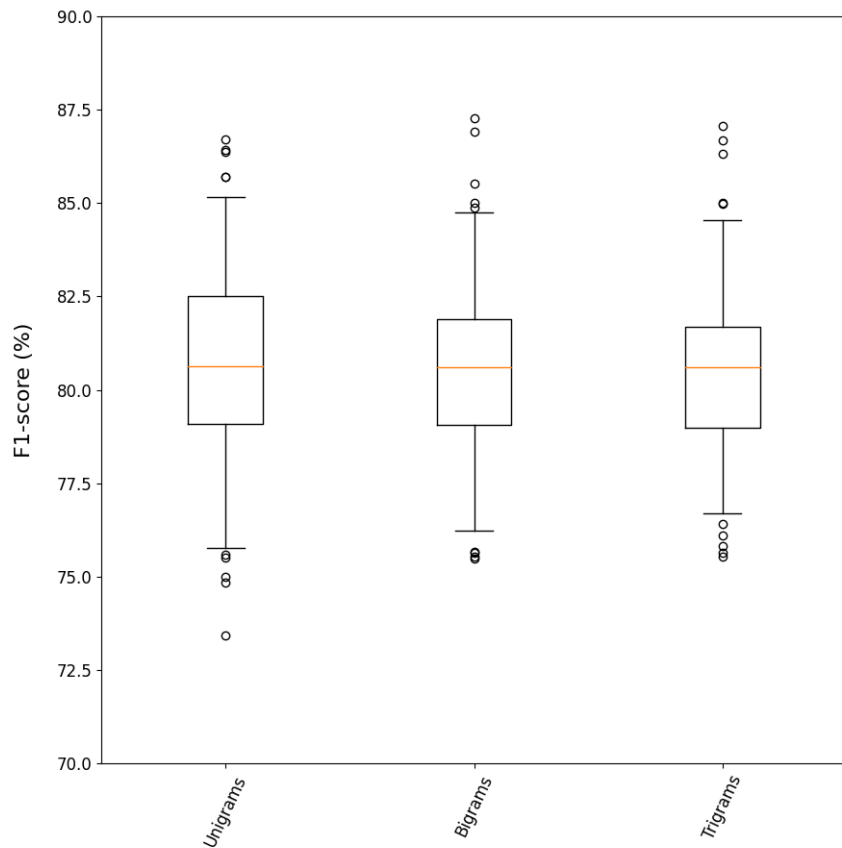


FIGURE 3.8: Distribution of F_1 scores for the *n-grams* analysis after 200 bootstrap iterations.

3.3.6 Distributed representations

This section examined the effect of encoding documents using word and document embeddings. Table 3.6 and Figure 3.9 illustrate the results. The F_1 scores for document representations produced via SPECTER were worse than those obtained by pooling BioBERT embeddings from the title and abstract tokens, with a difference of over 5 % in the median F_1 score. While SPECTER representations have demonstrated state-of-the-art performance for a variety of document-level tasks [204], pooling strategies at the token level may be more suited for this specific task because they are more likely to identify whether specific terms (e.g. PK parameters) appeared in the document.

When using BioBERT representations, adding min&max pooling resulted in a slightly higher median F1 score over using the mean across tokens. If only a small number of keywords (e.g. PK parameters and study-specific terms) contributed to the final predictions, min&max pooling are useful methods to identify the presence of those terms by extracting the most salient features from every token-embedding dimension [207]. However, since these keywords' appearance might also be detected with BoW approaches, in the following analyses, the effect of joining both (1) BoW + BioBERT mean pooling and (2) BoW + BioBERT mean + min&max pooling was studied.

TABLE 3.6: Summary table with performance metrics reported as median (95% CI) and F_1 interquartile variance (IQV) after 200 bootstrap iterations. The performance metrics are compared across pipelines using different distributed document representations.

Pipeline	Precision (%)	Recall(%)	F_1 (%)	F_1 IQV
SPECTER	74.1 (66.5,80.9)	69.0 (62.0,76.6)	71.2 (66.2,75.8)	9.6
Mean pool	78.1 (69.0,85.4)	75.3 (68.3,82.9)	76.6 (71.6,81.3)	9.7
Mean + min&max pool	80.1 (71.8,86.0)	75.9 (69.6,82.9)	77.7 (72.7,81.4)	8.7

Only two token-pooling strategies (mean vs min&max) were studied in this chapter since they represent the most common mechanisms used by previous approaches [207]. However, future studies might benefit from more complex hierarchical pooling approaches such as attentive pooling [208].

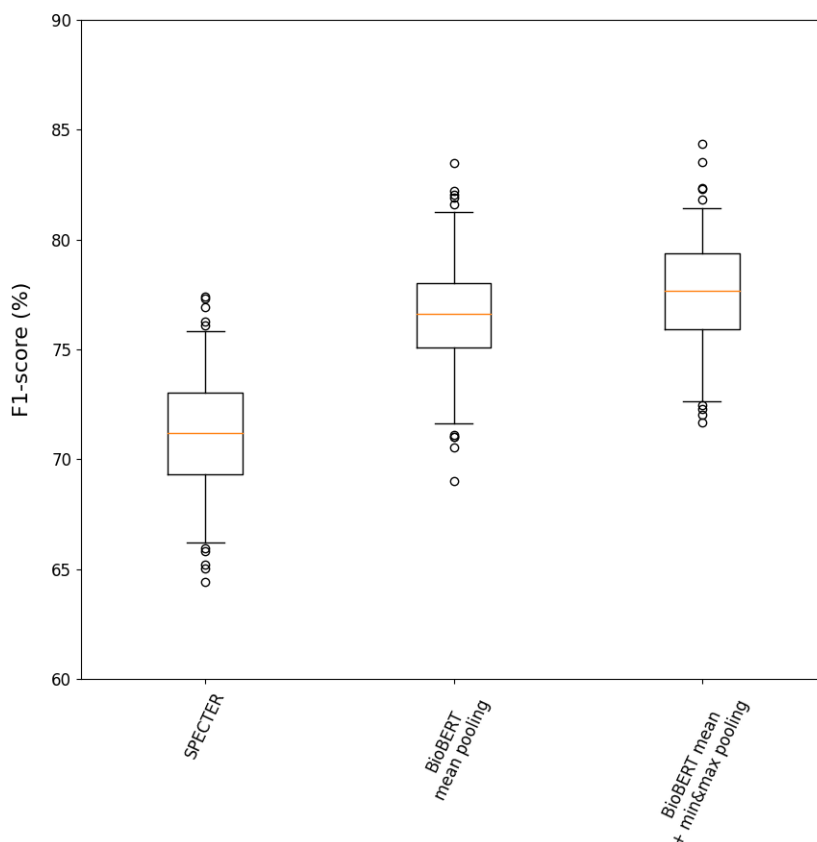


FIGURE 3.9: Distribution of F_1 scores for the *distributed* analysis after 200 bootstrap iterations.

3.3.7 Final Pipeline

Table 3.7 and Figure 3.10 show the outcomes of adding BioBERT mean pooling and BioBERT mean + min&max pooling embeddings to the unigram representations. When BioBERT embeddings were added to unigram representations, the median F_1 scores were slightly higher than when unigrams were used alone. However no significant improvement can be concluded due to the high degree of overlap between the three distributions.

TABLE 3.7: Summary table with performance metrics reported as median (95% CI) and F_1 interquartile variance (IQV) after 200 bootstrap iterations. The performance metrics are compared across pipelines using BoW together with distributed representations.

Pipeline	Precision (%)	Recall(%)	F_1 (%)	F_1 IQV
Unigr.	80.1 (73.9,86.0)	82.3 (74.1,88.6)	80.6 (75.8,85.2)	9.4
Unigr. + mean pool	83.7 (76.7,89.1)	80.4 (74.1,87.3)	81.7 (77.8,86.0)	8.2
Unigr. + mean + min&max	83.8 (75.6,88.8)	79.1 (73.4,85.4)	81.0 (77.2,85.4)	8.2

A five-fold CV approach was used to find the optimal hyperparameters (Figure 3.2) for encoding documents using Unigrams + BioBERT mean pooling since this pipeline reported the highest median F_1 score. The optimal hyperparameters were $min_df=128$,

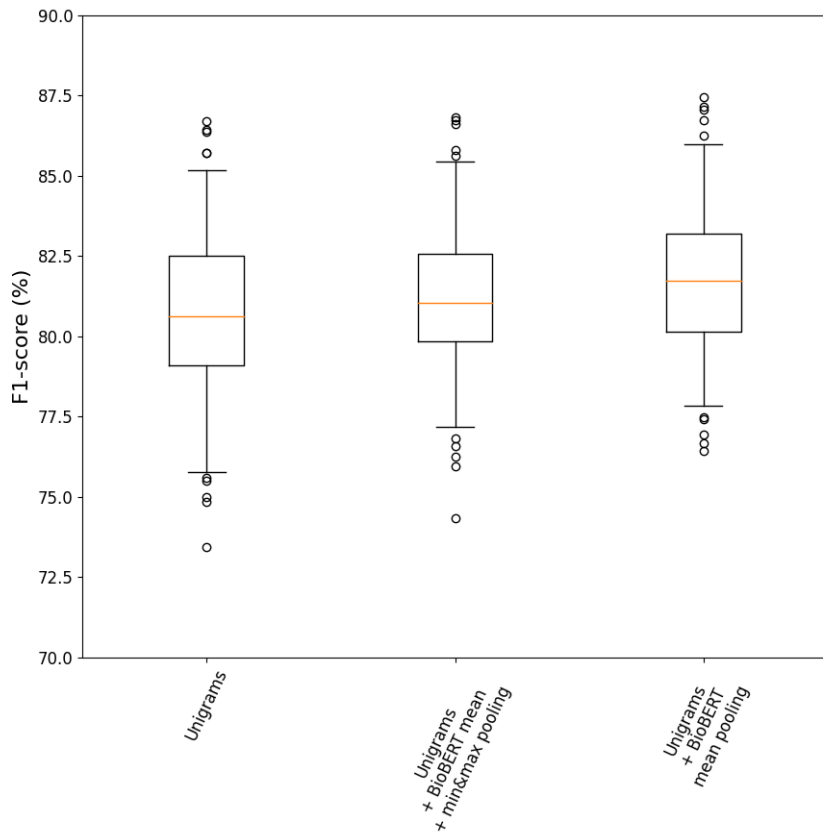


FIGURE 3.10: F_1 score distributions for the pipelines using unigrams together with BioBERT embeddings.

$max_depth=4$ and $colsample_bytree=1.$, which resulted in an average F_1 score of 83.8% over five folds. Finally, the whole *training set* was used to optimise the pipeline’s hyperparameters, which were then applied to the 800 documents in the *test set*. Table 3.8 contains the final metrics. The best estimates of the classifier’s performance on unseen data exhibited an F_1 of 83.8% on the classification of “Relevant” publications, and an overall 93.2% accuracy across all predictions (“Relevant” and “Not Relevant”). Wang et al. [11] used an entity template to detect scientific publications providing midazolam PK parameters in healthy human volunteers and achieved an F_1 of 78.1%. However, the relevance criteria used in this study were significantly broader than those used in Wang et al. [11], including a range of drugs, species, conditions and study designs. Overall, the pipeline presented in this study outperforms previous tools at detecting relevant literature reporting *in vivo* PK parameters.

A qualitative evaluation of the classifier predictions on the *test set* was conducted to identify causes of misclassification. The primary causes of misclassification of “Not Relevant” papers (thus limiting the pipeline’s precision) were: (1) papers reporting PK

results for endogenous substances (e.g. insulin) that were previously considered “Relevant”; and (2) physiologically-based or *in silico* PK studies that included estimated parameter values in the abstract. This suggests that the primary cause for these studies’ misclassification is the very similar word distribution to the “Relevant” articles since, despite not being relevant, they would often report PK parameter values along with demographic data. Misclassification of “Relevant” papers (thereby limiting the pipeline’s recall) was most frequently observed in the following situations: (1) publications without abstracts available in PubMed, (2) PK publications with parameters mentioned in the full-text but not in the abstract, and (3) animal PK studies. Cases 1 and 2 are challenging to detect since establishing their relevance might require information from the whole text, which is not provided to the classifier. However, the ceiling performance given by the average inter-annotator F1 score was 92.01%, which is still far from the current model performance. Additionally, our inspection indicates that there is still significant room for improvement in the detection of animal studies reporting *in vivo* PK parameters. Releasing this labelled corpus openly available aims to encourage the development of NLP pipelines that help to accelerate ADME datasets’ curation.

TABLE 3.8: Performance metrics of the final pipeline on the *test set*.

Precision (%)	Recall(%)	F_1 (%)	Accuracy (%)
84.8%	82.8%	83.8%	93.2%

3.3.8 Large-scale application

In January 2021, the final pipeline identified 120,913 papers as “Relevant” when applied to the corpus obtained from the PubMed search “pharmacokinetics” ($n = 584,961$). According to the annotation timings analysed in this chapter, manually annotating this corpus would have required over 610 days for a single annotator without further review. Furthermore, by utilising BERN [126], all “Relevant” articles were indexed according to the chemicals, species, and diseases stated in the abstract. All publications providing *in vivo* PK parameters have been made publicly available through a search interface at <https://pkpdai-search.com/pkdocsearch>. The classification pipeline was scheduled to run weekly updates to retrieve newly published PK papers. Overall, the interface serves as a centralised repository of articles reporting PK parameters, which researchers can now utilise to compare and efficiently find relevant PK data.

3.4 Conclusion

This chapter described a classification pipeline for identifying scientific publications that reported *in vivo* PK parameters. Objective O_{1-1} was addressed by developing a new

annotated corpus containing document-level labels for over 4000 abstracts. Different encoding and decoding approaches were compared to fulfil objective O_{1-2} , and the optimal pipeline obtained an F_1 score of 83.8 % on the test set. Finally, objectives O_{1-3} and O_{1-4} were addressed by applying this model to a large corpus of pharmacometric literature and developing an online resource¹⁴ including over 120K relevant papers to search for and compare PK outcomes. Furthermore, all labelled data and models were made publicly accessible to the research community at GitHub¹⁵.

In addition to speeding up ADME dataset curation, this open-access automated repository has the potential to enable text mining applications in the PK domain and serve as a central database for PK data search, making PK results easier to find, compare and replicate. The potential application of the search engine developed for curating ADME datasets was studied in detail in chapter 7.

3.5 Future work

Different approaches were independently explored in this chapter to represent and classify PK articles. One promising approach for improving the classification performance is to train a single neural model end-to-end through backpropagation, fine-tuning parameters for both (1) token embeddings and (2) classification layers. For instance, recent approaches have fine-tuned Transformer-based models (e.g. BERT) for specific long-text classification tasks achieving state-of-the-art results [209]. Additionally, to account for the specialised vocabulary found in the pharmacokinetics domain, future studies using BERT-style models could benefit from: (1) further pre-training on this corpus, (2) using pre-trained models with medical vocabularies (e.g. PubmedBERT [210]) or (3) using approaches that start with character-level representations and do not rely on WordPiece tokenisers, such as CharacterBERT [211].

Given the large number of PK publications retrieved in this study, future research in biomedical NLP might use this corpus and the annotated data to develop additional tools that can facilitate the extraction of PK information and evaluate document classification algorithms.

¹⁴<https://pkpdai-search.com/pkdocsearch>

¹⁵<https://github.com/PKPDAl/PKDocClassifier>

Chapter 4

Named Entity Recognition of Pharmacokinetic Parameters

The content of this chapter is based on work conducted by the author during a twelve-month Enrichment Scheme at The Alan Turing Institute.

4.1 Introduction

This chapter addresses the first step towards automated extraction of pharmacokinetic (PK) estimates from the literature, which is critical to accelerating the construction of Administration, Distribution, Metabolism and Excretion (ADME) datasets. Named Entity Recognition (NER) is fundamental for information extraction pipelines. Recognising entities of interest in text allows for subsequent downstream tasks such as relation extraction or entity linking [212]. As a consequence, over the last two decades, a variety of annotated resources have been developed to target different types of biomedical entities such as chemical mentions, genes, species or diseases, amongst others [88]. Some of these entities are crucial to extract pharmacokinetic PK measurements in context. For instance, given the sentence:

The elimination $t_{1/2}$ of leuprolide was 1.28 h in patients with prostate cancer.

An NLP system that extracts PK estimates from the literature might need to recognise a variety of entities such the **PK parameter** (elimination $t_{1/2}$), **chemical** related to the parameter (leuprolide), **estimated value** (1.28), **units** (h), **species** (patients) and **condition/disease** (prostate cancer). As previously mentioned, existing resources have been developed to tackle the main types of biomedical entities and they might be efficient to identify the drug, species and conditions mentioned in the example. Additionally, some entities with relatively consistent patterns (e.g. numerical values) might be easily identified with rule-based algorithms [11].

However, one of the main challenges for building comprehensive ADME datasets that are useful for preclinical drug development is to tackle multiple types of PK parameters [23]. Recognising PK parameters is also critical to characterise DDIs better since many interactions are reported by mentioning their effect on specific PK parameters [43, 213].

For instance, in the following sentences, it can be observed how detecting *clearance*, and *bioavailability* mentions is crucial for understanding the type of DDI:

The bioavailability of oral ondansetron was reduced from 60% to 40% ($P < .01$) by rifampin.

The clearance of mitoxantrone and etoposide was decreased by 64% and 60%, respectively, when combined with valspodar.

Recognising mentions of PK parameters is a challenging task since there are multiple PK parameter types and their mentions are often highly variable across the scientific literature, involving the frequent use of acronyms and long textual spans [43].

Previous work Although several biomedical text mining efforts have focused on DDIs, little work has tackled the NER of PK parameters. Lewinski et al. 2017 [214] developed an annotated corpus for NER in the domain of nanomedicine. The corpus annotated physicochemical properties, PK parameters and biologic response information from 41 drugs in drug product inserts collected from the Drugs@FDA Database¹. The annotation not only considered whether a specific span was a PK parameter but also classified it into whether it was an AUC, Clearance, Cmax, Elimination half-life, Plasma half-life, Tmax or Volume of distribution. However, only 272 mentions of PK parameters were present across the corpus, which highly limited the performance of NER models in recognising PK parameters (below 50% F₁ for all parameter types except plasma half-life). Although this corpus tackled several PK parameters, it did not cover all the ones appearing in the scientific literature (more than 66 were described in the PK Ontology from Wu et al. [43]). Additionally, drug product insert documents have a very different narrative than scientific articles. Therefore, due to the particular domain (nanomedicine), different types of documents (drug inserts), limited annotations of PK parameter spans, and space of parameters covered, models trained on this corpus are unlikely to perform PK NER of multiple parameters in scientific sentences efficiently.

The most similar work to the one presented in this chapter is the corpus developed by Wu et al. 2013 [43] when presenting the PK Ontology. In that study, the authors annotated multiple entities relevant to PKs and DDIs from multiple study types (i.e. clinical PK, pharmacogenetics, in vivo DDI and in vitro drug interaction studies). However, the annotations focused on a specific application domain (DDIs) and abstracts were deliberately selected from PK studies of certain compounds: midazolam and CYP2D6 enzyme. This selection of annotated sentences has limitations for training and evaluating NLP systems used across different PK study types. For instance, NER models trained on this corpus might learn specific features only used in midazolam's abstracts (e.g. some parameters, species or routes of administration might be mentioned more or less frequently than in abstracts from other drugs). Additionally, sentences were only selected

¹ <https://www.accessdata.fda.gov/scripts/cder/daf/>

from abstracts, but it is crucial to train and evaluate models that also use PK sentences from the full text since estimates are often not reported in the abstract but in the results section of the publication. Nonetheless, since PK parameter spans were annotated, the suitability of this corpus to develop PK NER models was studied in this chapter, and a description of the corpus is provided in section 4.2.2.4.

Finally, Kolchinsky et al. [160] studied different classifiers to identify sentences from the scientific literature that reported PK DDIs. To generate valuable features, Kolchinsky et al. mentioned the use of an in-house dictionary, *i-PkParams*, to detect terms relevant to PK parameters and studies. However, the dictionary was not publicly available and could not be used in this work.

Overall, few annotated resources are currently available for recognising PK parameter mentions. Additionally, no model has been found explicitly designed to perform NER of PK parameters in the scientific literature. Therefore, due to its potential benefit for extracting PK estimations from text and characterisation of DDIs, this chapter focuses on developing a NER model for PK parameters that can be effective across different PK study types and abstract and full-text sections. However, very large amounts of training data are often needed to train efficient NER algorithms, which can be prohibitively expensive to annotate. Therefore, this chapter also analyses the effect of active learning approaches for NER of PK parameters. With this context in mind, the Q₂ of this thesis was addressed with the following objectives:

- O₂₋₁ Develop annotated corpora using abstract and full text sections for NER of PK parameters.
- O₂₋₂ Analyse the effect of using active learning to reduce labelling efforts.
- O₂₋₃ Compare the performance of multiple approaches in recognising PK parameter mentions, from rule-based to Transformer models.
- O₂₋₄ Assess the generalisation of the models developed to external annotated corpora.
- O₂₋₅ Compare the effect of training models with the annotations developed in this study against using the corpus annotated by Wu et al. [43].

4.2 Methods

This section describes the methods employed to develop suitable architectures and annotated data to efficiently train NER models of PK parameters. The methods are divided into the following sections:

1. **NER Models** Description of architectures used to recognise PK parameter mentions.
2. **Corpus construction:** Description of the annotation procedure to develop an annotated corpus to train and evaluate PK NER pipelines.

4.2.1 NER models

4.2.1.1 Evaluation

Given some input text (often a sentence/paragraph), the task of NER models consists of identifying mentions of entities within the text. NER is commonly framed as a sequential labelling task, where each token in an input text receives a particular label [215]. Since entities can expand across multiple tokens, different tagging schemes exist to associate token-level labels to named entities composed by one or more tokens [216]. The BIO tagging scheme (also known as IOB2) is the most commonly used, where each token can be classified as the beginning of an entity (B), inside an entity (I), or outside of an entity (O) [210, 216]. To determine the type of entity, the “B” and “I” tags are often followed by the title of the entity type. For instance, a PK entity composed of two tokens followed by a single-token chemical entity might be labelled as [“B-PK”, “I-PK”, “B-CH”] according to the BIO scheme. In this chapter, a single entity type was studied, “PK” using the BIO tagging scheme.

The most common metrics for NER are precision, recall and F_1 scores, but different approaches can be taken when analysing sequences of BIO labels. During the training of ML models for NER, the loss function often computes the loss at the token level. However, when evaluating the performance of NER systems, it is crucial to derive entity-level metrics instead of token-level since these are more reflective of how well the system can recognise the desired named entities. This setting requires defining the concept of *true and false positives* and *false negatives* for a specific sequence prediction to derive precision, recall and F_1 scores at the entity level. Partial overlaps between a system prediction and annotated labels need consideration. Table 4.1 shows an example of a sequence of tokens with BIO predictions against BIO labels. There are two entities mentioned in the input sequence, “terminal half-life” and “renal clearance”, each composed of two tokens. The system correctly identifies the boundaries and type of “terminal half-life”

but predicted only “clearance” instead of “renal clearance”, which is considered a partial match. In NER, it is common to consider these partial matches as false positives (because the predicted entity is incorrect) and false negatives (because the system did not predict the annotated span). However, these criteria might be too restrictive for some applications. It is important to monitor partial matches to evaluate PK NER systems since they might already be helpful for some applications (e.g. capturing the primary PK term might be enough to determine the type of DDI). Therefore, two main criteria were used to evaluate NER models in this chapter based on the metrics presented at the Message Understanding Conference V [217], and SemEval-2013 Task [218]: strict and partial matching.

TABLE 4.1: Example of system prediction and true labels in PK NER. Where “O” stands for out of an entity, “B-PK” beginning of a PK entity and “I-PK” for inside a PK entity.

Token	System Prediction	True Label
the	O	O
terminal	B-PK	B-PK
half-life	I-PK	I-PK
and	O	O
renal	O	B-PK
clearance	B-PK	I-PK
were	O	O

Strict matching This is the most common metric reported in the NER literature. In this scenario, only predictions that have total overlap with the annotations in terms of entity boundaries and types are considered true positives. If the system misses an annotated entity, this is considered a false negative, and if it predicts an entity that has not been annotated, it is considered a false positive. Partial matches are considered as both false positives and false negatives.

Partial matching This metric reflects some information on cases where the system prediction has some entity tokens overlapping with annotated entity tokens. Specifically, if there is a partial overlap between a system prediction and annotated entity, this is considered a half-true positive, i.e. $\text{True Positives} = \text{strict matches} + 0.5 \cdot \text{partial matches}$.

Precision, recall and F_1 scores were independently computed for the two matching criteria in this chapter. The metrics were computed using the *nervaluate*² library, which researchers in the field have extensively used. Specifically, to perform an evaluation

²<https://github.com/MantisAI/nervaluate>

independent of the tokenisers used, the test script accepted annotations at the character level (i.e. defining character start and end offsets of entity mentions). Given an input sentence, it required models to predict character-level entities. Finally, strict and partial matching were computed using the character-level predictions. For instance, the following shows an example of the format required to compare annotations against predictions for two sentences of the test set:

```
Annotations = [
  {"label": "PK", "start": 2, "end": 4},
  {"label": "PK", "start": 1, "end": 2},
  {"label": "PK", "start": 3, "end": 4}
]

Predictions = [
  {"label": "PK", "start": 2, "end": 4},
  {"label": "PK", "start": 1, "end": 2},
  {"label": "PK", "start": 2, "end": 4} # partial match
]
```

4.2.1.2 Rule-based

Given the PK expertise of the annotation team, a set of rules was generated to develop a rule-based model covering well-known PK parameters and their primary surface forms and acronyms. The model was implemented using the entity ruler from spaCy³ which requires a set of token-level patterns and can incorporate rules regarding part-of-speech (POS) and dependency labels. The pre-trained scispaCy (en_core_sci_md) was used as a base tokeniser, POS tagger and dependency parser to incorporate the token-level patterns into the model. For a list of patterns see Appendix B: Named Entity Recognition patterns. Developing the list of terms and rules was an iterative process performed together with experienced pharmacometricians, and the model performance was updated by assessing its performance on the development set. Once the rules covered the most well-known terms, the final rule-based algorithm was only evaluated once on the test set to compare the model performance against ML models.

4.2.1.3 BERT

Most state-of-the-art architectures currently used for NER tasks employ pre-trained Transformer models to obtain contextual token representations [2]. BERT was used in this study as a pre-trained model. As observed in Figure 4.1, a sequence of tokens is given as an input to the BERT model, which processes them through a series of transformer encoder blocks (see details in section 2.2.2.5) and returns a sequence of contextual token embeddings (green boxes in Figure 4.1). Then, a task-specific classification model

³<https://spacy.io/usage/rule-based-matching>

(Token classifier in Figure 4.1) is arranged on top of BERT to generate the final output token labels (e.g. using the BIO tagging scheme). It has been shown that when fine-tuning⁴ BERT models for specific tasks, the output representations already capture many non-linear and long-range dependencies between input tokens [2, 210]. Therefore, the token classifier consisted of a single feed-forward layer accepting an output vector representation from BERT and transforming it to a vector of j dimensions where $j =$ the number of classes (3 in this dataset), which was finally normalised with a softmax operation to convert output values to class probabilities. In other words, the following operation was applied to each output token representation from BERT (T_i) to obtain their token labels:

$$\hat{y}_i = \text{softmax}(W_o T_i + b_o) \quad (4.1)$$

Where \hat{y}_i represents the model output for token i expressed as a vector of j elements, each containing the probability of BIO classes (in this application “O”, “B-PK” or “I-PK”). W_o and b_o are the weight parameters of the token classifier. The softmax operation performs the following transform to each value in the output layer (x_i):

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{w=1}^j e^{x_w}} \quad (4.2)$$

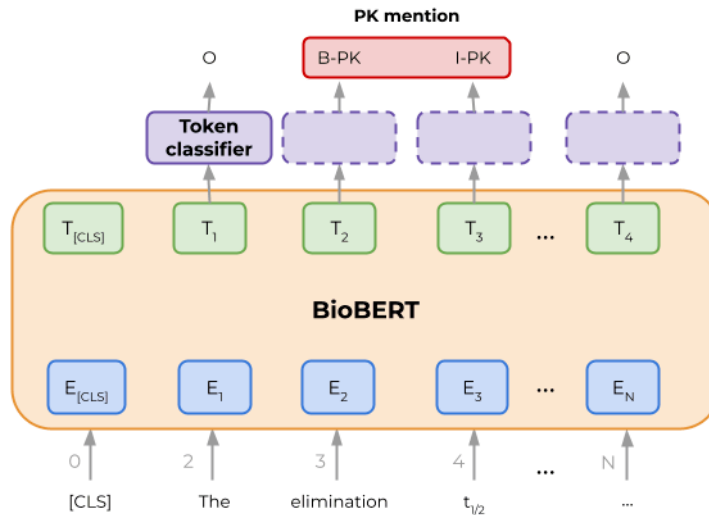


FIGURE 4.1: Illustration of the fine-tuning strategy for NER with BERT-based models. Blue boxes represent the input embeddings (E_i) for each token. Green boxes represent the output embeddings (T_i) from BERT which go through a feed-forward layer (Token Classifier) and are mapped to token-level BIO labels.

Two pre-trained models were compared: BERT_{BASE} [2] which was pre-trained on general-domain English text (English Wikipedia and BooksCorpus), and BioBERT v1.1

⁴ All BERT parameters are updated by gradient descent using backpropagation.

[122] which was further pre-trained on PubMed articles. The implementation was done using PyTorch [219] and the Huggingface Transformers library [220]⁵.

Training Training instances were sentences with their respective entity annotations. BERT tokenisers were used to split each input sentence into a sequence of (sub-word) tokens. Each token was associated with a BIO label (“O”, “B-PK” or “I-PK”) according to the annotations provided. The training was supervised. The model was trained to minimise the categorical cross-entropy loss between the predicted token-level softmax scores and the true labels encoded as one-hot vectors. Both BERT and classification layer parameters were updated during training. The model was trained during 20 epochs and evaluated on the development set at the end of each epoch. In each experiment, the state of the model with the highest entity-level strict F_1 score on the development set at a certain epoch was saved. In most experiments, the development F_1 scores were roughly stable after ten epochs and most models achieved the highest strict F_1 score shortly after that time. The Adam Optimizer with a linear weight decay⁶ of 0.05 was used and dropout a dropout probability of 0.1 was applied on all layers (according to [2]). The experiments were run on a single GPU NVIDIA Titan RTX (24GB).

Hyperparameter tuning In early experiments, it was observed that the development performance was not very sensitive to the choice of hyperparameters as long as they were within specific ranges. Therefore, little hyperparameter tuning was performed. The batch size was established at $B=16$, and larger values were not explored due to memory constraints. Learning rate was the main hyperparameter examined, which was grid-search over $\mu = [1e-5, 2e-5, 3e-5, 4e-5, 5e-5]$. The exploration was performed using BioBERT and a single run for each learning rate, obtaining the best performance when $\mu=3e-5$. Ten runs with different random seeds were performed for each experiment, and the run with the highest F_1 score on the development set was selected to use it on the test set. One critical hyperparameter was the maximum number of tokens for input sequences. After observing the distribution of # tokens/sentence in the training set (Figure 4.2), a maximum of 256 tokens was established to cover most training instances, and those sentences with more than 256 tokens were truncated. At inference time, if sentences contained more than 256 tokens, the sentence was split into several instances, and after performing predictions, BIO labels were re-joined.

4.2.1.4 SpaCy

Although transformer-based pipelines currently outperform alternative architectures for NER, they contain a large number of parameters (e.g. 110M for BERT_{BASE}) which

⁵ <https://huggingface.co/transformers/>

⁶ For details on weight decay regularisation the reader is referred to Loshchilov and Hutter 2017 [221].

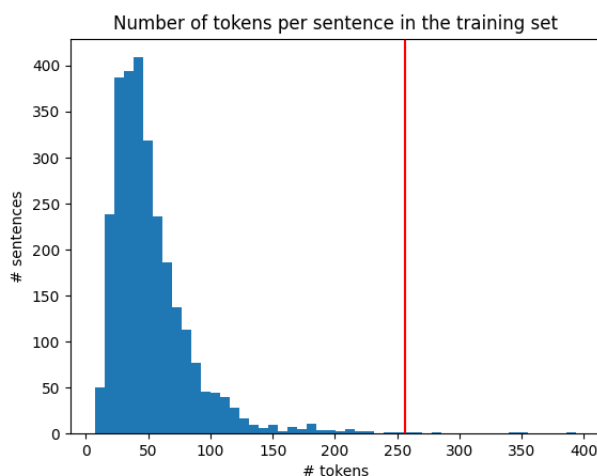


FIGURE 4.2: Distribution of number of BERT tokens in each sentence of the training set. The red line determines 256, which was the maximum length established during the experiments.

often limit their usability due to long inference times and technical expertise required to perform inference. The spaCy library⁷ gathers components tackling multiple NLP tasks (e.g. tokeniser, POS tagger, dependency parser, named entity recognisers) in a single pipeline. Due to their efficiency and competitive performance, spaCy models have been widely adopted by the NLP and Python communities.

To compare the BERT pipelines against simpler ML architectures for NER and provide efficient and easy-to-use PK NER models, the scispaCy [149] model was fine-tuned to perform NER of PK parameters. ScispaCy is built on top of spaCy but focusing on biomedical/scientific text processing. Specifically, scispaCy models contain a custom rule-based tokeniser, trained POS taggers, dependency parsers, and NER and EL components on biomedical text, showing competitive performance on several biomedical NLP benchmarks. The spaCy NER model is a transition-based system based on the chunking model from Lample et al. 2016 [215]. Each token is represented with a combination of hashed, embedded representation of their prefix, suffix, shape and lemmatised features of individual words [149]. A convolutional neural network with residual connections is used to create contextual embeddings for each token. In this work, all components from the scispaCy medium-sized pipeline (en_core_sci_md) were reused, and the NER layer was trained from scratch. Analogous to the BERT pipelines, models were trained for 20 epochs and evaluated on the development set after each epoch. The state of the model with the best performance on the development set was saved. The rest of the hyperparameters and training protocols were the same as those from scispaCy NER pipelines [149].

⁷ spacy.io

It is worth noting that all the architectures implemented in this chapter treated NER as a token classification problem. This decision was because the PK entity spans did not overlap. Hence, each token could only belong to a single entity span. However, future work involving drug names or other biomedical entities might need to address the issue of nested entities with different architectures (e.g. Ju et al. [222]).

4.2.2 Corpus construction

An annotated corpus was constructed to train and evaluate NER models for PK parameters, named PK-NER-corpus.

4.2.2.1 Source

Looking at sentence-level information is often enough to determine whether a specific span of text relates to a PK parameter without further context. Hence, NER annotations and training were performed at the sentence level.

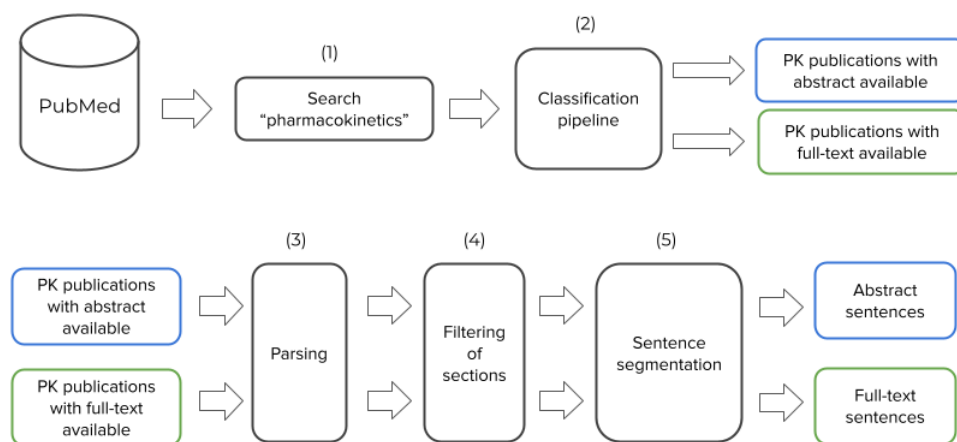


FIGURE 4.3: Flow diagram showing the main processes involved to generate a pool of candidate sentences for NER labelling.

To generate a candidate pool of sentences to be annotated, the steps displayed in Figure 4.3 were performed. Initially, the PubMed search “pharmacokinetics” was applied in June 2020, and the optimal pipeline described in chapter 3 was used to select publications that reported novel estimates of *in vivo* PK parameters, which resulted in a corpus of 114,921 relevant publications. From this corpus, the full-text of 10,132 articles (8.82%) was accessible from the PMC OA subset⁸ (green boxes from Figure 4.3) while only the abstract was accessible for the rest of publications (blue boxes from Figure 4.3). Both, full-text and abstract publications were downloaded in XML format from

⁸ <https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/>

the PubMed⁹ and PMC¹⁰ FTP sites, respectively. Then, the PubMed Parser [197] was used to extract textual data from XML files resulting in abstract and paragraph texts. Since the introduction section rarely contains relevant information for PK text mining purposes, paragraphs from the introduction section were removed before sentence segmentation. Sentence segmentation of scientific text presents characteristic challenges compared to general-domain English text (e.g. punctuation is often present in abbreviations and noun compounds). Therefore, the sentence segmentation algorithm from scispaCy [149], which was designed to deal with scientific and biomedical text, was used to split the abstract and paragraphs into sentences. Overall, this preprocessing resulted in two sets of sentences: the abstract and full-text pools.

The full-text pool contained 721,522 sentences, while over a million sentences were available in the abstract pool. For many IE applications, it is desirable to evaluate and train ML models using both abstract and full-text data. Therefore, 721,522 instances were randomly sampled from the abstract pool and joined with the full-text sentences to generate a balanced set of candidate sentences. This procedure resulted in a balanced pool of 1,443,044 sentences, referred to as the candidate pool. All the sentences labelled during the corpus construction were sampled from the candidate pool.

4.2.2.2 Annotation interface and Guidelines

An interface was developed to annotate PK entities at the sentence level using the commercial tool Prodigy¹¹. This tool was used since it allowed flexible design of UI interfaces and it facilitated the implementation of annotation frameworks with a model-in-the loop. However, note that other annotation tools have been previously used allowing active learning approaches, such as APlenty [223], Inception [224], AlpacaTag [225] or Paladin [226]. Figure 4.4 shows the main components of the interface developed in this chapter. Each annotation instance displayed a single sentence, and field experts were asked to highlight those spans of text relating to PK parameters. If annotators were unsure about how to annotate a specific instance, they could flag the example for review (flag in Figure 4.4). After annotations were performed, each sample stored character-level indices for each annotated span (e.g. {entity: “PK”, start_character: 15, end_character: 20}). The annotation interface was deployed in a virtual machine hosted on Azure, and Prodigy sessions were launched on local ports. Local ports were then exposed to a public address on the internet using *ngrok*¹². Each annotator accessed the interface through a unique web link, and annotations were stored in an SQL database on the virtual machine and backed up on Azure Blob storage.

⁹https://www.nlm.nih.gov/databases/download/pubmed_medline.html

¹⁰<https://ftp.ncbi.nlm.nih.gov/pub/pmc/>

¹¹<https://prodi.gy/>

¹²<https://ngrok.com/>

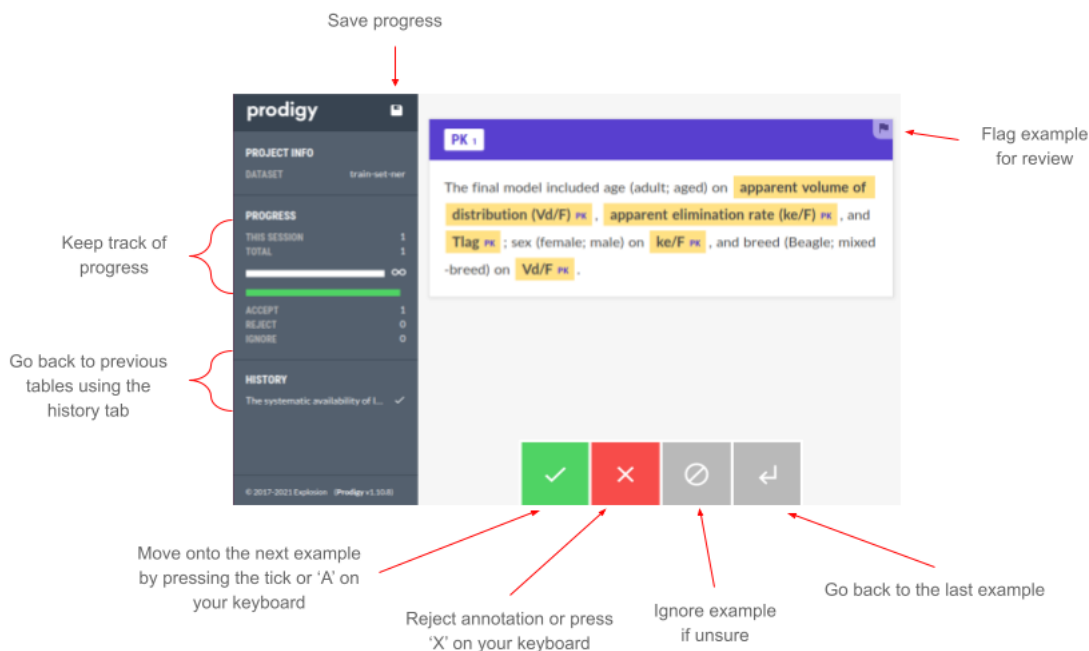


FIGURE 4.4: Screenshot of the interface used to annotate PK entities from scientific text. The example displays a single sentence after being annotated.

The annotators were PK modellers familiar with the different parameters and study types present in the PK literature. However, this work aimed to annotate any PK parameter mention, covering a large collection of parameter types and potential surface forms appearing during annotation. Therefore, explicit and detailed guidelines were required to ensure consistency in various decisions, especially those regarding span boundaries. For instance, amongst many others, some common doubts arising when annotating PK spans were:

- Which noun modifiers should be included as part of the PK span: *renal clearance* or *renal clearance*; *mean clearance* or *mean clearance*.
- How to label abbreviated forms: *oral clearance (CL/F)* or *oral clearance (CL/F)*, *volume of distribution (V_d)* or *voiume of distribution (V_d)*.
- Ratios of PK and PD parameters: *AUC/MIC* or *AUC/MIC*.
- Whether to label parameter mentions appearing inside equations.

Initially, a small set (n=100) of examples were independently labelled by each annotator involved to detect common doubts and causes of disagreement before annotating larger batches of sentences. The resolution of which parameters to include and how to define span boundaries was discussed with the team, and decisions were based on the expected applications of the models developed, i.e. numerical PK parameter extraction

and characterisation of DDIs. To determine the main PK parameters included, the PK ontology published by Wu et al. 2013 [43] was used. Annotation guidelines were provided to annotators before starting the labelling task. As new challenging examples appeared and conflicting instances were resolved, guidelines were updated accordingly. The final annotation guidelines for this task can be found in Appendix A: Guidelines Named Entity Recognition.

4.2.2.3 Annotation process

The team responsible for the annotation involved three researchers with in-depth expertise in PK modelling and one PhD student with NLP and PK expertise. Three datasets were developed to train and evaluate different NER pipelines named: training development and test sets.

Construction of test and development sets The development and test sets were generated from randomly sampled sentences from the candidate pool without replacement. This approach aimed to assess the performance of NER algorithms in both the abstract and full text and to account for the variability of parameter mentions across the literature by including sentences from multiple papers (e.g. different acronyms often used in different publications). In total, 1,500 and 500 sentences were selected for the test and development sets, respectively. Initially, a set of trial sentences were independently labelled by each annotator involved to detect the primary sources of confusion and disagreement. The initial set of guidelines was then developed. Each of the 2000 sentences (test + development) followed a two-stage procedure of (1) initial annotation by one expert, (2) review and standardisation of span boundaries by another bio-NLP expert (similar to Hope et al. [227]). This process was carried out in batches of 200 sentences. After each batch, sources of disagreement were discussed, and annotation guidelines were updated. This approach attempted to ensure consistent and high-quality spans, but some parameter mentions could still have been missed. To account for potential parameter mentions missed during this annotation process, sentences in which model predictions disagreed with the labelled spans were displayed in the terminal to compare the predicted vs labelled spans. This last check allowed detecting parameter mentions missed during annotation and generating hypotheses behind model errors.

Why selective sampling? The construction of development and test sets was particularly labour-intensive and time-consuming since it required iterative reviews of the annotations to check for missing cases and frequent discussions to unify criteria. Additionally, many sentences did not contain PK entities. It was observed that only $\approx 16.4\%$ of the sentences sampled in PK articles contained PK parameter mentions, resulting in 149 (development) and 390 (test) annotated spans (see Table 4.3). The sampling

approach used for the development and test sets preserved the distribution of sentences in PK articles, which is likely to be a frequent application ground for PK NER models. However, training effective NER models for new entity types often require many annotated samples with relevant spans on the training set to account for the variability of surface forms and contexts of use [228]. Hence, applying the same sampling approach used for the development and test sets to the training set might require labelling a vast number of training sentences to model the highly variable distribution of PK parameter mentions across the literature. Therefore, a selective sampling protocol was used to maximise the information provided by labelled sentences and generate a training set covering a large variety of PK parameter mentions.

Construction of training set NER models that rely on hand-crafted rules provide a framework to incorporate prior knowledge regarding entity mentions. It is often possible to construct rules that select positive samples with relatively high precision and low recall, which can be used to construct an initial training dataset covering a collection of well-known entity mentions [194]. Then, statistical models might be incorporated to generalise new patterns that could not have been easily written by rules (e.g. patterns highly dependent on context). Two main approaches were sequentially used to selectively sample the most informative sentences for PK NER: (1) developing a rule-based model to automatically generate an initial dataset and (2) active learning to select samples where the model was most uncertain.

1. Initial dataset The rule-based model described in section 4.2.1 was applied to all the sentences in the candidate pool. Early exploration showed that the patterns developed had relatively high precision on partial matches, but there were frequent mistakes regarding the exact span boundaries. Additionally, some spans required highly contextual information to determine whether they were PK parameters. For instance, the mention of “F” could refer to the PK bioavailability, the statistic from a hypothesis test, a table/figure number or other concepts, but it can only be inferred using contextual information. Similarly, “area under the curve” could refer to the plasma concentration-time curve or any other curves (e.g. ROC curve in machine learning). Hence, a subsequent correction of the labels provided by the rule-based model was required. Three hundred sentences containing matches from the rule-based model were randomly selected to construct an initial training set, and annotators were asked to correct labels provided by the rule-based model. After correction, 86.67% of sentences (260) remained with PK mentions, although their span boundaries often required correction.

2. Active learning Selecting samples to label based on a model prediction is known as active learning (AL). Most machine learning algorithms provide probabilistic outputs of their predictions (e.g. softmax layer in neural networks). This output might be used

as a proxy to estimate the uncertainty¹³ of a model on a specific prediction and prioritise data samples with higher uncertainty for labelling [229]. This approach aims to label data samples close to the model decision boundary to provide training instances that reduce the uncertainty of the model. As shown in Figure 4.5 (A), the scispaCy model (en_core_sci_md [149]) was trained with the initial dataset of 300 sentences (see section 4.2.1.4 for details on the model and training approach).

After training the scispaCy pipeline on the initial dataset, the Prodigy *ner.teach* framework¹⁴ was used to suggest those spans where the model was most uncertain about based on the token-level softmax scores given by the NER model (B and C Figure 4.5). Specifically, Prodigy uses beam search to select the most uncertain spans over the sequence of token-level probabilities. The default settings from the Prodigy *ner.teach* framework were used (e.g. update approach and frequency). Annotators were asked to accept or reject spans predicted by the model, and based on their answers; the model was updated in the loop after every batch of ten binary annotations (D and E Figure 4.5). Note that binary annotations (accept/reject a suggested span) only indicated whether a span was an entity or not, but information from the rest of the sentence tokens was not provided. Hence, the model was updated with this partially annotated data (annotator feedback only provided for those tokens from suggested spans) by updating the target probabilities of each candidate label in each token involved in the suggested spans. The updates were automatically performed during the annotation with the default settings from Prodigy. Binary decisions highly accelerated and simplified the annotation process.

Instead of using large language models like BERT, the scispaCy pipeline allowed fast iterations and updates with the model in the loop, which made it preferable for AL purposes. This procedure was performed for 2500 sentences by a single annotator. Once binary annotations had been performed, another annotator reviewed each of the sentences with binary annotations and highlighted other spans if present in the sentence (step F in Figure 4.5). The model used for AL was re-trained (not updated) from scratch after every batch of 500 sentences was annotated (Figure 4.5, step G). Finally, the dataset collected with AL approaches was joined with the initial rule-based dataset, generating a training set of 2800 sentences (step H Figure 4.5).

The stopping criterion for annotation was based on the annotators' availability and the project's time constraints. However, future studies could use confidence-based stopping criteria for active learning, such as Zhu et al. [231].

Inter-annotator agreement As discussed in Hripcsak and Rothschild [232], Cohen's kappa score is not the most suitable metric for inter-annotator agreement (IAA) of NER

¹³This approach only accounts for aleatoric uncertainty [229]. Note that different sources and quantification methods of uncertainty exist across the literature. For a comprehensive review, the reader is referred to Abdar et al. [230].

¹⁴<https://prodi.gy/docs/recipes#ner-teach>

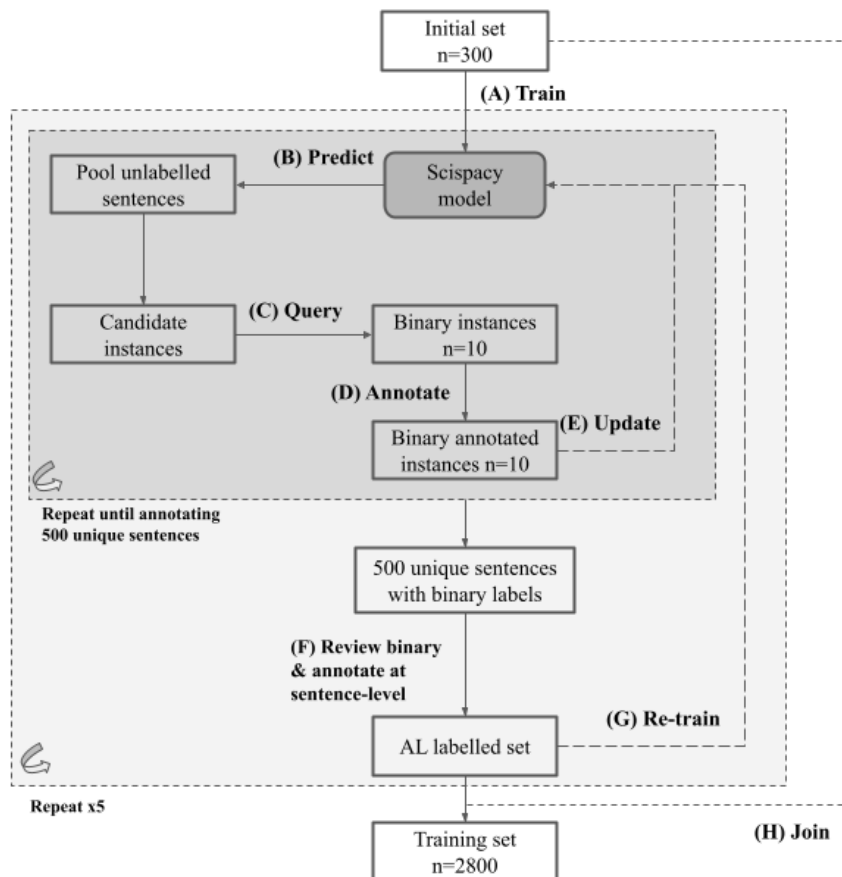


FIGURE 4.5: Schematic representation of the approach used to label instances for the training set by using and updating scispacy NER model in the loop.

annotations since it requires the number of negative samples to be computed *a priori*. However, since named entities are sequences of tokens, there is no pre-defined number of candidates in the annotation task [233]. Instead, pair-wise F1 is often used to measure IAA of named entities [232, 233]. The four annotators independently labelled a total of 200 sentences from the test set, and the average F1 score across each pair of annotators was used as the IAA. This exercise was done with the last batch of the test set, when guidelines had already been updated multiple times, but no corrections were performed before computing the F1 score.

4.2.2.4 External dataset validation

Wu et al. 2013 [43] developed a PK Ontology and presented an annotated corpus using their ontology. The annotated corpus from Wu et al. 2013 will be referred to as PK-Ontology-corpus in this chapter. In that study, 541 abstracts were manually labelled, involving the annotation of key terms, DDI sentences, and DDI pairs. The abstracts came from 4 study types: clinical PK (mostly focusing on midazolam), clinical pharmacogenetics, *in vivo* DDI and *in vitro* DDI studies. One of the key terms annotated

in the PK-Ontology-corpus were PK parameters. Since the source of the abstracts was different from those developed in this chapter (PK-NER-corpus) and focused on specific application domains, the NER models developed in this chapter were assessed in the PK-Ontology-corpus. This external evaluation allowed assessment of the performance of the models developed in an external dataset with different study types, including several DDI sentences and observing differences in the annotation criteria.

In this analysis, the XML annotations from PK-Ontology-corpus were preprocessed to generate character-level NER labels for PK terms. The PK-Ontology-corpus annotated different key terms, but only PK parameters were analysed in this study. The dataset came with three files: *invivo_train.xml*, *invitro_train.xml*, *invivo_test.xml*, *invitro_test.xml*. For this work, *in vitro* and *in vivo* studies were merged, resulting in a training and test dataset. The statistics for those datasets are given in Table 4.2.

TABLE 4.2: Corpus statistics of the PK-Ontology-corpus stratified by the training and test sets.

Dataset	# Sentences	# entity mentions	% sentences with PK mentions
Training	4008	1478	23.68
Test	1021	377	25.27

4.3 Results and Discussion

4.3.1 Corpus statistics

The main statistics for the PK-NER-corpus are shown in Table 4.3. First, the differences between the evaluation (development and test) and the training sets are worth noting. Since the evaluation sets randomly sampled sentences from PK articles, the proportion of sentences containing PK parameter mentions was only 16.40%. Despite preserving the distribution of sentences in which PK NER algorithms might be applied, fewer entity mentions were present in the evaluation sets. On the other hand, 64.25% of sentences in the training set contained mentions of PK parameters, resulting in many entity mentions. This difference in the distribution of parameter mentions was due to the active learning protocol applied to sample training sentences.

A large proportion of sentences from the training set (79.46%) came from sentences in the full-text body of the articles. Although the candidate pool of sentences was balanced (same number of abstract and full-text sentences), the active learning protocol suggested full-text sentences more often than those from the abstract. One potential explanation for this is that authors tend to report a standard set of PK parameters in the abstract (e.g. clearance, $t_{1/2}$), while full-text contains a higher variability of parameter types and mentions (e.g. compartmental parameters, inconsistent acronyms). This fact might cause higher model uncertainty of spans coming from full-text sections which resulted in more sentences from full-text being suggested by the AL loop.

TABLE 4.3: Corpus statistics of the PK-NER-corpus stratified by the training, development and test sets.

Dataset	# Sentences	# entity mentions	% sentences with PK mentions	% of full-text sentences
Training	2800	3680	64.25	79.46
Development	500	149	16.40	50.8
Test	1500	390	16.40	50.8

The average inter-annotator pair-wise F1 scores for exact and partial matching were 89.55% and 93.48%, respectively. These results suggested a high consistency in the annotation of PK parameters. However, differences in span boundaries still accounted for a large proportion of disagreements. Nonetheless, it is worth noting that these metrics reflect the agreement between annotators after stage one, but a subsequent standardisation of span boundaries was performed in all sentences aiming to improve the quality and consistency of the annotations. Therefore, the pair-wise F1 scores reported here does not reflect the ceiling performance for the models, but the expected agreement between pharmacometricians without standardisation of criteria and span boundaries.

After review and standardisation without active learning, the average annotation time per sentence resulted in approximately one minute. The active learning approach accelerated this process, reducing the annotation time to approximately 40 seconds per sentence.

4.3.2 Model Comparison

This section presents the performance comparison of different NER models in the PK-NER-corpus. Table 4.4 summarises the main results on the test set. Models were developed on the training set, and the best hyperparameter combinations were selected using the development set.

Machine learning approaches tend to outperform NER dictionary- and rule-based methods when sufficient training data is available and there is a high diversity in entity mentions' surface forms or strong context dependencies. This is often caused by an improved recall rate [235]. However, rule-based models are often a good starting point to evaluate whether more complex approaches are required since they can often provide high precision [235]. Developing good algorithms based on rules often requires reasoning about the specific domain and the type of entity mentions trying to extract. Therefore, rule-based models can be used as strong baselines for more complex approaches when domain knowledge is available to develop robust rules. The results from this section showed that the rule-based model could not efficiently cover the diversity of PK parameter mentions annotated by field experts on the test set, with a strict F_1 score below 50%. Some of the main challenges of the rule-based approach were (1) great variety of PK parameter types, which limited the pipeline's recall, (2) presence of complementary terms within PK spans that were difficult to encapsulate with rules (e.g. total body clearance), (3) acronyms highly dependent on context (e.g. "F" for bioavailability, "AUC" for the area under the concentration-time curve). There was a large difference (over 15%) in the Precision for the strict vs partial match evaluation. This is a consequence of challenge 2, where rules often detected the primary PK term, but complementary terms determining the parameter subtype were missed. It is important to note that the initial collection of PK terms was constructed from prior expert knowledge. In the future, the recall of the rule-based model could be improved by including a more extensive collection of PK parameter types and acronyms using existing annotations from the training set.

The ML pipelines significantly outperformed the heuristic model with over 30% gain on the strict F_1 score and particularly large improvements on the Recall. Furthermore, as it has been previously reported [236], it was observed that the models based on BERT provided substantial performance benefits in comparison to the scispaCy-based model. On the other hand, from the test set predictions, it was observed that the scispaCy

TABLE 4.4: Results on the test set for different NER models. Metrics are reported at the entity level using strict and partial match.

Model	Strict			Partial		
	P	R	F_1	P	R	F_1
Rule-based	52.8	43.59	47.75	69.25	57.18	62.64
ScispaCy	77.09	82.82	79.85	80.91	86.92	83.81
BERT	81.47	87.72	84.48	84.92	91.43	88.05
BioBERT	90.49	90.26	90.37	92.54	92.31	92.43

pipeline was x10 faster at inference time on CPU in comparison to running BERT models on a single GPU¹⁵.

The BioBERT model outperformed the BERT model pre-trained on general-domain English text, especially on strict entity matching. Specifically, BioBERT provided a large gain (Δ 9%) on the pipeline precision in comparison to all the other models. This result suggests that domain-specific pre-training is crucial for PK NER and exploiting BERT’s transfer learning potential. Therefore, in the subsequent analyses, the optimal BioBERT pipeline was used. It is worth noting that the performance of the fine-tuned BioBERT model was already higher than the initial pair-wise F1 between annotators, suggesting that posterior reviews and standardisation of the corpus improved the annotation consistency and exhibiting the high performance of this model during PK NER.

Other baseline models based on pre-trained language models, such as SciBERT [139], or RoBERTA [237], could have been used. However, the models selected in this study aimed to compare the effect of (1) using machine learning approaches instead of rules, (2) using transformer-based language models instead of CNNs, and (3) using models pre-trained on biomedical text against pre-training on general-domain English text. Therefore, the benchmarking of other architectures was left for future work.

Error inspection Overall, the BioBERT pipeline showed good performance at detecting PK parameter entities with strict F_1 over 90%. However, a qualitative examination of the main reasons behind the model errors was carried out to assess the limitations and potential improvements of the model. Some common examples are presented in Table 4.5.

- When drug names were present within the PK parameter span, the model struggled to predict the entity span successfully. The first examples in Table 4.5 show two scenarios where the model did not include the drug names (tulathromycin and cyclosporine) as part of the spans. Although including drug names within parameter mentions is rare, future approaches might benefit from masking drug

¹⁵NVIDIA Titan RTX 24GB.

names or performing augmentation by replacing drug mentions with other drug terms.

- The prediction of slightly incorrect spans (partial matches) where a nearby term was missed or added by the model was a common source of error. These cases have been previously observed as common challenges in scientific NER [238]. It was also one of the most common causes of disagreement between annotators. Although guidelines were detailed about which terms to include, the number of exceptions was vast, and inconsistencies might still be present in the annotated corpus. Further review of the annotations might improve the performance of the model in these cases.
- In some cases, the model predicted spans of text that, despite not being PK parameters, they related to some parameter that might have numerical estimations associated with it. For example, as it can be observed in Table 4.5, *time to relapse (TTR)* or *AD50* both had contextual similarity to PK parameters since they were present in the form of acronyms and had numerical values associated with them.

TABLE 4.5: Common errors from the BioBERT model on the test set. green [*] = annotated entity, blue [*] = model prediction

Drug mentions	The [[maximal] tulathromycin concentration] in lung and muscle homogenate from a single animal was 4657 ng/g (14 days) and 2264 ng/g (7 days), respectively. Two patients with radiologic signs of gastroparesis had no [[peak cyclosporine levels] at all and were excluded from the correlation analysis.
Partial matches	The mean NTZ [[maximum concentration (Cmax)] in plasma] was 10.2 µg/ml. Cyclosporin A treatment resulted in a significant increase in [[elimination half-life]] , [[mean [residence time]]] and [[area under the concentration versus time curve (AUC)]] of unbound baicalein in the brain.
Non-PK parameters	After immuno-chemotherapy, the median follow-up was 55.5 months [24–108 months] , median [[time to relapse (TTR)]] was 60 months [12 - more than 108 months], and time to second CLL treatment (TST) was 84 months [24 - more than 108 months]. The dose of COU that promoted the bronchodilator effect in guinea pigs ([[AD50]] : 75 mg/kg) [15] was used as a reference to conduct the study.

To showcase the model and allow assessment by researchers in the field, an interactive demo of the model was released at <https://pkpdai-search.com/pknerdemo>.

4.3.3 Active Learning protocol

This section compared selecting samples using the AL protocol to random sampling. Here, AL protocol refers to the process of (1) generating rules and an initial dataset, (2) developing an initial model on this dataset and (3) using and updating the model in the loop to select new samples to be labelled. The development set (n=500) was used as an example of an annotated set randomly sampled. Five hundred sentences from the training set (n=2800) collected with AL were randomly sampled to perform a fair comparison. Ten separate runs with different random seeds were performed. The

AL experiment randomly sampled a different subset of sentences from the training set and randomly initialised the classification layer parameters in each run. The random sampling experiment only involved randomising the classification layer parameters in each run since only 500 sentences were available. The BioBERT model was trained for five epochs with a $\mu=3e-5$, and the final model was applied to the test set at the end of each run.

Table 4.6 and Figure 4.6 show the results of these experiments. Training the BioBERT model with the AL dataset resulted in over 7% increase on the median F_1 score for strict matching compared to training with randomly sampled sentences. Most of this difference was due to the Recall of the AL pipeline, which was significantly higher than the random sampling. These results suggest that the protocol used to generate the training set highly benefited the model performance compared to randomly sampling sentences. Most of this benefit is the consequence of an improved recall, suggesting that the AL dataset contains a wide variety of PK spans not covered by the random sampling dataset. Considering the frequency of named entities in each dataset (i.e. only 16.4% of sentences mentioned PK parameters in the randomly sampled datasets), it is likely that the selective sampling approach implemented for this task was particularly beneficial for covering a wider variety of relevant spans.

TABLE 4.6: Summary table with performance metrics comparing random sampling against active learning protocols. The *Active Learning* dataset was obtained by randomly sampling 500 sentences from the training set. The *Random Sampling* refers to the development set (n=500 sentences) with different model initializations in each run. Metrics were obtained in the test set after training the pipelines for five epochs. Metrics are reported as median values after 10 runs.

Dataset protocol	Strict				Partial			
	P	R	F_1	F_1 IQV	P	R	F_1	F_1 IQV
Active Learning	74.14	84.51	79.78	7.54	77.94	88.92	83.67	6.49
Random Sampling	71.92	75.77	72.61	9.66	78.08	82.78	78.81	8.9

It is important to note that a variety of approaches have been applied for AL in NER [229, 239, 240]. For instance, Bayesian approaches have recently shown promising results [229], although their application comes with computation costs. It is still unclear which AL approaches are most beneficial to make efficient use of a model in the loop. In this chapter, many approaches are left for exploration. For instance, using BERT-based models in the loop instead of scispaCy, using diversity sampling, applying other criteria to estimate uncertainty better, to name a few. However, the framework developed with Prodigy allowed for fast annotations that reduced the labelling load, and the samples suggested in early experiments provided diverse (i.e. in terms of surface forms and parameter types) and challenging spans. Therefore, since the primary focus of this work was to study and develop suitable corpora and models for NER of PK parameters, the AL approach described in this study proved efficient for developing such corpora.

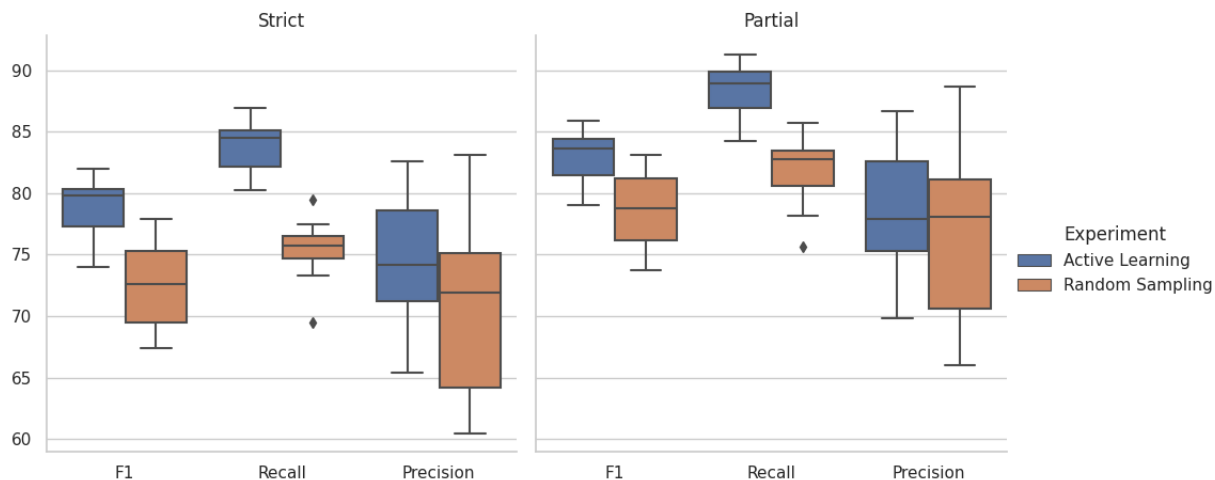


FIGURE 4.6: Distribution of F_1 , Recall and Precision scores for the *Active Learning* and *Random Sampling* datasets ($n=500$ sentences) after 10 runs with different random seeds. The left and right panels display the scores considering strict and partial matching of entities, respectively.

Due to the multiple types of PK parameters and their corresponding surface forms and acronyms appearing in the PK literature, a promising approach for future studies gathering training data could involve diversity sampling, aiming to cover each type of parameter and their diversity of named entities. After analysing the annotated data it was also observed that some entity surface forms had a high frequency in the corpus, which might provide redundant information to the model. Additionally, the model had worse performance on some types of parameters (e.g. “time to maximum concentration”). Hence, AL approaches that aim to cover a wider variety of surface forms and parameter types might be beneficial for this task.

4.3.4 External corpus

To evaluate the model performance on an external corpus and assess annotation differences between the PK-NER-corpus and the one from Wu et al. 2013 [43] (PK-Ontology-corpus), the BioBERT model fine-tuned on the PK-NER-corpus was applied to the test set of the PK-Ontology-corpus, without any further training on this dataset. Surprisingly, the model reported a competitive strict F_1 score of 74.52% without any training on that dataset (See Table 4.7). A substantial increase can be observed when considering partial matching (81.1% F_1), which suggests that the main PK terms are often captured, but disagreements on span boundaries limit the model performance on this dataset. If using the training set from the PK-Ontology-corpus to fit BioBERT¹⁶, the strict F_1 score increases by $\approx 7\%$. Additionally, a substantial increase is observed from

¹⁶In this experiment BioBERT was trained for ten epochs, and the final model was saved.

strict to partial matching, which might be explained by inconsistencies on span boundary annotations in the training set of PK-Ontology-corpus. Although different research groups developed both datasets, it was observed that the models trained on PK-NER-corpus transferred well to the annotation criteria used in Wu et al. 2013. However, when training a model with the PK-Ontology-corpus and applying it to the test set of the PK-NER-corpus, the exact matching metrics were 68.8%, 63.59% and 66.13% for precision, recall and F_1 score, respectively. Since the PK-Ontology-corpus used abstracts deliberately picked (e.g. midazolam PK abstracts), models trained on this corpus probably learnt some features specific to these types of articles but not transferable to PK studies from other types of compounds. These results indicate that PK NER models trained on the PK-NER-corpus generalise better to unseen PK publications than those trained on the PK-Ontology-corpus.

TABLE 4.7: Results obtained on the external validation corpus test set. Metrics from the PK-NER-corpus were obtained by training models on PK-NER-corpus and applying them to the PK-Ontology-corpus. Their own training set was used for the PK-Ontology-corpus to fit the NER model.

Training corpus	Strict			Partial		
	P	R	F_1	P	R	F_1
PK-NER-corpus	77.05	72.15	74.52	83.85	78.51	81.1
PK-Ontology-corpus	79.75	83.55	81.61	86.2	90.32	88.21

Annotation differences An evaluation of true labels against model predictions was performed to detect possible differences in the annotation criteria between the PK-NER-corpus and the PK-Ontology-corpus. The principal sources of divergences are displayed in Table 4.8.

- The first source of divergence were span boundaries. It was observed that terms such as “renal clearance” were not labelled in some examples of the PK-Ontology-corpus while these were labelled in PK-NER-corpus since they provided information on the type of PK parameter. Additionally “mean total area under the concentration-time curve” or “the peak plasma concentration” were labelled in PK-Ontology-corpus. Many PK mentions included the term “the” in the PK-Ontology-corpus, while this term was considered not relevant in the PK-NER-corpus.
- Some PK parameters were not labelled or missed in PK-Ontology-corpus such as t_{max} (time to reach C_{max}) or the volume of distribution.
- The plasma concentrations were not considered a PK parameter in PK-NER-corpus since their values are often noisy and irrelevant for numerical extraction purposes. However, these were often labelled in the PK-Ontology-corpus.

These differences are likely to limit the performance of models trained on the PK-NER-corpus when applied to the PK-Ontology-corpus. However, except for partial matches and mentions of plasma concentration, the model trained on the PK-NER-corpus detected almost all the main PK terms appearing on DDI sentences from PK-Ontology-corpus. This result suggests that the corpus and model developed is aligned with Wu et al. [43] work and it is likely to be a suitable tool to detect PK parameters in DDI contexts.

TABLE 4.8: Predictions vs annotations in the PK-Ontology-corpus. green [*] = annotated entities, blue [*] = predictions of model trained on the PK-NER-corpus.

Span boundaries	The renal [clearance] was reduced by 60% and the elimination half-life increased by 35%. Clarithromycin increased the mean [total area under the concentration-time curve] of repaglinide by 40% (P <.0001) and the [peak plasma concentration] by 67% (P <.005) compared with placebo.
Parameters not in PK-Ontology-corpus	Thus, [[clearance]] was approximately halved, [steady-state volume of distribution] was increased, and [[terminal half-life]] was more than doubled. Itraconazole did not change the [peak time] or the [elimination half-life] of either oxybutynin or N-desethyloxybutynin.
Parameters not in PK-NER-corpus	[Plasma concentrations of bosentan] and its three metabolites were measured on days 1 and 7 of treatment A and on day 6 of treatment B. Both erythromycin and itraconazole greatly increased [plasma buspirone concentrations] , obviously by inhibiting its CYP3A4-mediated first-pass metabolism.

4.4 Conclusions and Future work

This chapter presented a new corpus to train and evaluate NER models to detect mentions of PK parameters in the scientific literature which fulfilled objective O₂₋₁. The active learning protocol accelerated the curation of PK data while improving the information provided by labelled sentences compared to random sampling, which addressed objective O₂₋₂. To address objective O₂₋₃, a variety of models were compared, and BioBERT reported the best performance on PK NER with over 90% F₁ score on strict entity matching. Domain-specific pre-training with transformers was crucial to obtain optimal performance. A lightweight model based on scispaCy was developed to facilitate fast inference and integration with other NLP frameworks. Finally, the best-performing model showed good generalisation on various study types when applied to external annotated corpora and validated its potential application to improve the characterisation of DDIs, which was accomplished through objective O₂₋₄. Additionally, by tackling objective O₂₋₅, it was observed that models trained on the corpus developed in this chapter exhibited significantly better generalisation to unseen PK studies than models trained on previously published corpora.

Future directions of this work could involve:

1. Exhaustive evaluation of the model when characterising DDIs
2. Extension of the model to perform relation extraction of pharmacokinetic parameter measurements
3. Exploration of other AL strategies
4. Study of acronym resolution to improve the recognition of PK parameters
5. Evaluation of the effect of pre-training BERT on PK text

Chapter 5

Few-shot Entity Linking of Pharmacokinetic parameters

The content of this chapter is based on work conducted by the author during a four-month internship at BenevolentAI.

5.1 Introduction

Chapter 4 focused on developing models that could recognise mentions of any pharmacokinetic (PK) parameter in the scientific literature. However, linking those parameter mentions to entities in a knowledge base (KB) is critical to identify the specific parameter type and stratify the PK information extracted from the literature. For instance, when extracting PK parameter estimates from the literature, numerical values might be grouped by PK parameter types and drugs to compare distributions of PK estimations (e.g. Wang et al. [11], Obach et al. [193]). Furthermore, information of PK parameter types has rarely been included in text mining approaches to drug-drug interaction (DDI) [241]. However, such information can be crucial to understand the causal mechanisms behind DDIs, since it can aid the characterisation of PK DDIs by predicting whether the interaction affects the absorption, metabolism, distribution or excretion processes [241]. Pharmacokinetic parameters are generally grouped into main classes (e.g. clearance, half-life, C_{max}, AUC, bioavailability, the volume of distribution)¹ that refer to measurements of different ADME properties. However, more granularity on parameter types is often needed when constructing ADME datasets or comparing parameter distributions. For instance, Wang et al. 2009 [11] showed that when extracting numerical estimates of midazolam’s systemic clearance from the literature, the extracted values had a very different distribution (in terms of mean and standard deviation) than the values obtained when extracting oral clearance.

Additionally, the main parameter classes have many sub-types crucial to stratify before including values in ADME datasets. Specifically, Wu et al. 2013 [43] defined 66 parameter types found across PK studies and relevant for PK text mining. Although Wang et al. 2009 [11] retrieved numerical values for systemic and oral clearance with high precision, the number of estimates missed and the ability to expand their rule-based approach to

¹ For a detailed description of these categories see section 1.2.3.1 in the introduction.

other PK parameters has not been explored. For these reasons, this chapter focuses on the study of EL in the PK domain.

The process of normalising specific spans (often detected by NER models) to entities in knowledge bases is known as entity linking (EL) and plays a fundamental role in structuring knowledge stored across different types of textual documents. Being able to link mentions to KB entities is a critical step for many information extraction (IE) pipelines since it resolves the lexical ambiguity of entity mentions (often containing many different surface forms) and allows the association of attributes and relations reported in the text to specific KB entities [242]. In turn, this allows the construction of nodes in knowledge graphs or items in databases. However, EL is often performed after recognising candidate mentions in the text. For instance, a NER model detecting mentions of person names would not be able to resolve whether the mention “Michael Jordan” refers to the basketball player² or the mathematician³ [242]. Hence, EL would be required to successfully disambiguate the “Michael Jordan” mention and potential associate relations or attributes found in the text to the correct KB entity.

EL is a challenging task in NLP. For example, each entity often has multiple associated mentions (i.e. synonyms), and a single mention can belong to multiple entities (i.e. polysemy) [126]. Additionally, biomedical and general-domain KBs such as Wikipedia or the UMLS contain millions of entities [98]. Due to these challenges, traditional approaches for EL have often relied on extensive expert-curated vocabularies and complex hand-crafted rules that perform approximate dictionary matching [126, 243]. However, the performance of these approaches has been limited by the challenge of keeping terminology lists complete and efficiently dealing with synonyms used by multiple KB entities. More recently, there has been a shift to using neural methods for EL that exploit contextual information to disambiguate named entities and do not exclusively rely on extensive vocabularies and hand-crafted rules [130, 242], which have achieved state-of-the-art results in multiple EL benchmarks [128, 244]. However, applying machine learning for entity linking is challenging since the number of possible classes is extremely large compared to other classification tasks. As a consequence, annotated EL datasets only cover a small fraction of the entities in the KB in their training mentions [244]. This setting requires from models that can link entities not observed at training time or had very few training examples. The setting of having none or few labelled training data for a particular KB entity is referred to as zero- or few-shot, respectively [245].

While the majority of EL work has focused on linking to large and generic knowledge bases, it is frequently desirable to link to specialised entity dictionaries in a specific sub-domain [246]. For instance, the UMLS covers many biomedical concepts (over 3M), but only a few general PK parameters are included (i.e. clearance, half-life). Additionally, training efficient EL systems often requires very large annotated datasets (in the order

² https://en.wikipedia.org/wiki/Michael_Jordan

³ https://en.wikipedia.org/wiki/Michael_I._Jordan

of tens of thousands of example mentions (e.g. Mohan and Li [98])). Unfortunately, labelled data for these specialised KBs are not commonly available and are frequently challenging to develop.

One of the main challenges of mining data from multiple PK parameters is the availability of a system that can recognise a wide variety of PK parameter mentions and successfully determine the specific parameter type in a knowledge base. For this reason, the main aim of this chapter was to develop a novel EL system that could ground PK parameter mentions to a knowledge base. Machine learning approaches for PK entity linking were explored in this study to account for the high variability of PK parameter types and overlapping synonyms and acronyms between them. Due to the limited training data available and sparse PK parameter mentions in the literature, potential challenges and suitable architectures to develop EL models in a specialised sub-domain with limited training data were explored, focusing on few- and zero-shot settings. Hence, the following objectives were defined to tackle Q₃ of this thesis:

- O₃₋₁ Develop annotations to train and evaluate entity linking architectures for PK parameter mentions using the ontology developed by Wu et al. [43].
- O₃₋₂ Study suitable adaptations from existing architectures for few and zero-shot EL to the task of PK EL, where the number of KB entities and training examples is significantly smaller than in most EL tasks.
- O₃₋₃ Compare few- and zero-shot approaches to simpler baselines and analyse the main sources of model errors.

5.2 Methods

In this work, the EL task of PK parameters was framed as a subsequent step of NER. Hence, given a mention span predicted by a NER system, the EL model related that mention to the correct entry in a knowledge base. The following sections present the methodology implemented to develop annotated corpora for PK EL and the model architecture used to address this task.

5.2.1 Corpus construction

The annotated corpus from Wu et al. 2013 (PK-Ontology-corpus) [43] and the PK-NER-corpus presented in chapter 4 provided annotations regarding which spans of text referred to PK parameters. However, no information regarding the specific type of parameter was annotated. Since no annotated corpora were publicly available to evaluate and train EL systems for PK parameters, a corpus was constructed with training, development,

and test sets. This corpus consisted of sentences with PK parameter mentions and their associated entries in a KB.

5.2.1.1 PK Knowledge Base

Amongst other concepts appearing in PK studies, Wu et al. presented a comprehensive collection of PK parameters containing the main parameter names, a description of each parameter, and their units. Specifically, two tables describing *in vivo* and *in vitro* PK parameters were published by Wu et al., covering 65 PK parameter types (Tables 2 and 4 from [43]). After inspection by field experts, this collection was considered to contain the main parameters of interest for ADME dataset construction and to characterise PK DDIs. Therefore, this collection of parameters, together with their descriptions and units, was used as the PK Knowledge Base (KB) for EL in this chapter. Additionally, due to its frequent appearance across PK studies, minimum concentration (C_{min}) was also included. It is worth noting that some less frequent PK parameter mentions might not be covered in this KB despite appearing in PK studies. Therefore, an additional entity named *NIL*⁴ was added as a potential EL label for those parameter mentions not covered by the KB. In this study, the *NIL* entry was used to account for both PK parameter mentions not listed in the KB and incorrect entity mentions detected by the NER system. However, the NER’s partial matches with an associated KB entity were labelled with the correct KB entry. Overall, the final KB used in this chapter included 67 entries (66 PK parameters + *NIL* entry) with parameter titles (main name in the KB), descriptions and units.

5.2.1.2 Data source and annotation

This corpus was intended to train and evaluate EL systems that, given a single PK parameter mention in context (e.g. sentence), could associate that mention to the correct entity in the KB. Therefore, each parameter mention was assigned one of the 67 categories in the KB.

Source Annotations were performed at the sentence level since this often provides enough context to determine the type of PK parameter. To generate a comprehensive corpus of sentences with PK parameters, the best-performing NER model from chapter 4 was applied to abstract and full-text sentences of PK publications. Only sentences from the 114,921 relevant PK publications described in the previous chapters were used. Sentences from the abstract and full-text sections of these publications were available *in-house*. The NER model was applied to over 1.2M sentences. If the model detected

⁴ Common label to indicate that there is no matching entity in the KB for a specific mention [43].

a parameter mention in a specific sentence, an annotation candidate instance was generated containing the sentence and span boundaries of the parameter mention. If one sentence mentioned more than one parameter, this was divided into multiple candidate instances. After generating a collection of candidate instances, 1000, 500 and 1000 instances were randomly sampled to annotate the training, development, and test sets, respectively. The number of samples selected for each dataset was determined based on the availability of expert annotators. More samples were allocated to the training and test set to (1) have enough training samples to model the complexity of the task and (2) obtain final performance metrics that were as reflective as possible to the distribution found in the scientific literature.

Annotation process The annotation interface was implemented in Prodigy, and, for each instance, it displayed a sentence with a single PK parameter mention highlighted (as predicted by the NER model) and 67 candidate options relating to the KB entries. Annotators were asked to select a single option for each mention in context displayed in the interface. The annotation team involved two senior scientists from the Drug Metabolism and Pharmacokinetics (DMPK) team with in-depth expertise on PK modelling and one PhD student with NLP and PK expertise. Guidelines were initially generated and iteratively updated as new challenging examples appeared. Two annotators labelled the development and test sets independently, and disagreements were reviewed and resolved. This process was done in batches of 250 instances to resolve disagreements and unify criteria iteratively. Overall, the agreement significantly improved along the annotation process. Before resolution, two labels were provided for each of the 1500 evaluation instances (development + test), including overlap between all annotators. To estimate the quality of the annotations, the Cohen Kappa Coefficient K (see section 3.2.1.3 for details) was used to measure pairwise inter-annotator agreements before resolving conflicting examples on the development and test sets. Due to the time constraints of this project, a single annotation was generated for each training instance with an expert review. Micro- and macro-accuracy across annotators was also computed on the development set to assess ceiling performance.

5.2.2 EL architectures

Task Given an input text document D and a list of mentions within that document $M_D = \{m_1, \dots, m_N\}$, an EL model returns a list of mention-entity pairs $\{(m_i, \hat{e}_i)\}_{i \in [1, N]}$, where each predicted entity (\hat{e}_i) is an entry in a knowledge base $e \in \mathcal{E}$ and \mathcal{E} represents the space of possible knowledge base entities (e.g. Unified Medical Language System (UMLS)). [128]. The models developed in this chapter assume that a title and a description is available for each KB entry.

It is common to frame EL with a two-stage approach: (1) candidate generation; where a group of KB entities are selected for a particular mention, and (2) candidate ranking; where candidates are inspected in-depth and ranked according to their relative probability of being the correct link [246]. This approach is commonly used for KBs with a vast number of entries (usually over 10,000 [98, 128, 246]) since the candidate ranking is often a computationally expensive operation and candidate generation aims to reduce the number of examples going to the second stage. However, since this work consisted of linking mentions to a KB of 67 candidate entities, it is computationally feasible to consider all KB entries as candidates at inference time. Therefore, the approaches presented in this section do not perform candidate generation but only rank KB entities for a given input mention.

5.2.2.1 Bi-encoder

Architecture To develop a model that used entity descriptions from the KB and had the potential of predicting entities without examples in the training set (zero-shot), a bi-encoder architecture was implemented. The model was similar to the bi-encoders presented by Wu et al. 2019 [128] (candidate generator) and Humeau et al. 2019 [247]. A diagram of the approach is shown in Figure 5.1.

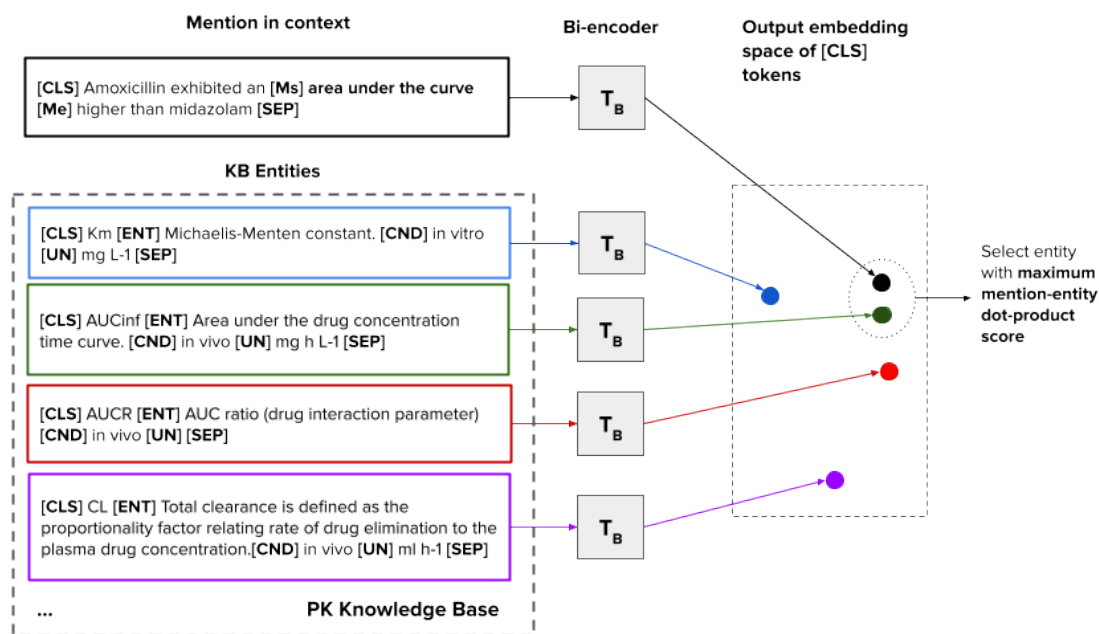


FIGURE 5.1: Schematic description of the bi-encoder approach to represent, encode, and link PK parameter mentions in sentences to Knowledge Base (KB) entries. T_B represents the language model used to encode mentions and KB entries into single-vector representations in an embedding space. Each KB entity representation was scored against the mention representation using the dot-product, and the entity with the maximum dot product was selected as the model prediction.

This architecture starts with a sequence of tokens for every KB entity (τ_e , named candidate entity) and mention in context (τ_m). Then, candidate entities and mentions were encoded into vector representations:

$$y_m = red(T_B(\tau_m)) \quad (5.1)$$

$$y_e = red(T_B(\tau_e)) \quad (5.2)$$

Where τ_m and τ_e are the input token representations for mentions and entities, respectively. T_B represents a language model accepting a list of input tokens and returning a list of contextualised token embeddings, and $red(*)$ is a function reducing the list of token vectors into a single vector. Analogous to Humeau et al. [247] and Wu et al. [128] T_B was a pre-trained Transformer and $red(*)$ was the function selecting the output representation from BERT (last layer) of the $[CLS]$ token. In other words, given N output token embeddings from a Transformer model, $red(*)$ simply selects the first embedding, corresponding to the $[CLS]$ token representation. Experiments were carried out comparing BERT_{BASE} [2] and BioBERT v1.1 [122] as pre-trained models. Note that the pre-trained BERT models used in this work output token vectors with 768 dimensions, which was the length of y_m and y_e vectors. Humeau et al. [247] and Wu et al. [128] trained two independent transformer models (T_1 and T_2) to encode KB entities and mentions. Due to the limited number of training samples and KB entities in this work, the same model parameters (T_B) were used to encode both entities and mentions independently. However, experiments were carried out to compare the effect of this simplification (named *shared vs non-shared* approaches in section 5.3.2).

Input representations The BERT tokeniser split input strings into sub-word tokens. For PK parameter mentions, two special tokens were used to determine where the mention started and ended, $[Ms]$ and $[Me]$:

$$[CLS] \text{ ctxt}_l [Ms] \text{ mention} [Me] \text{ ctxt}_r [SEP]$$

Where ctxt_l and ctxt_r are the sub-word tokens before and after the mention in the sentence. An example of an input representation for a PK mention in context is shown in Figure 5.1. A maximum length of 128 sub-word tokens was established. Entities in the KB had (1) a title describing the main name of the parameter, (2) a brief description of the parameter, (3) a field specifying whether the parameter was measured *in vitro* or *in vivo* and (4) the common units of that parameter. Hence, to generate input representations for KB entities, the special tokens $[ENT]$, $[CND]$, $[UN]$ were used to separate sub-word tokens of those fields:

$$[CLS] \text{ title} [ENT] \text{ description} [CND] \text{ condition} [UN] \text{ units} [SEP]$$

Scoring Analogous to Wu et al. [128], to score the similarity between a specific mention (m_i) and a KB entity (e_j) the dot-product between their vector representations was used:

$$s(m_i, e_j) = y_{m_i} \cdot y_{e_j} \quad (5.3)$$

Optimisation The model was trained to maximise the dot-product between input mentions (m_i) and their correct entity in the KB (e_c) with respect to other negative entities. Specifically, for each training pair of input mention and correct entity (m_i, e_c), the loss was computed as follows:

$$\mathcal{L}(m_i, e_c) = -s(m_i, e_c) + \log \sum_{j=1}^N \exp(s(m_i, e_j)) \quad (5.4)$$

Where N specifies the number of negative samples + 1, and the ordered set of entities $\{e_1, e_2, \dots, e_N\}$ is computed by each training pair and includes the correct entity (e_c) and N-1 incorrect entities randomly sampled from the KB. The number of negative samples (N-1) was treated as a hyperparameter initially set to 7 and explored during preliminary experiments.

Due to their minor impact on the model performance, minimal tuning was performed on other hyperparameters. Models were implemented in PyTorch and optimised with Adam with a linear weight decay of 0.05. The batch size remained constant at 8. The learning rate was grid-searched based on the best micro-accuracy in the development set, and the values explored included $\mu = [2e-7, 2e-6, 2e-5, 2e-4, 2e-3]$. Early experiments with BioBERT indicated 2e-5 as the best-performing learning rate, which was kept constant in subsequent experiments.

Inference When predicting the correct KB entries for a given list of mentions, all entities in the KB were initially encoded and cached. Then, for every mention, its representation was scored against all the KB entity representations, and the entity with the maximum mention-entity dot-product was selected as the model prediction. This approach was extremely fast at inference time compared to EL systems based on Cross-Encoders⁵ that need to pass each (mention, candidate entity) pair through the language model [128].

⁵ Cross-Encoders refer to models accepting mentions and KB entities simultaneously and generating a representation for every entity-mention pair. For a detailed explanation on Cross-Encoders, see [247].

5.2.2.2 Baseline

Since the number of KB entities is quite limited in this study (67) compared to other EL tasks, a competitive baseline might consist of a simple sequence classification model, which has been named softmax classifier in this chapter. Given an input mention in context as a list of tokens (τ_m), a BERT model was used to encode the mention into a single vector (y_m) using the [CLS] token representation. This approach was previously described by equation 5.1. Then, the contextualised mention representation was passed through a linear layer mapping the 768-dimensional vector y_m to an output vector of 67 elements, each relating to one KB entity. Finally, a softmax layer was applied to the output vector. This linear layer + softmax approach is analogous to the token classification approach described in the previous chapter (equation 4.1). The model was trained using categorical cross-entropy loss and the same training protocol as the bi-encoder. Only BioBERT was used for the softmax classifier since early experiments with the bi-encoder already showed significant improvements over using BERT_{BASE} as a pre-trained model.

It is worth noting that, in contrast to the bi-encoder, this baseline did not use textual information in the KB and treats the task as a sequence classification of mentions in sentences. However, this approach used the same sequence encoder (BERT) as the bi-encoder. Given the small number of training samples and KB entities, this allowed for a comparative assessment of the potential benefits and limitations of the bi-encoder approach for training novel EL systems in low-resource and domain-specific scenarios.

5.2.2.3 Evaluation metrics

The labelled corpora is often represented as $\{(m_i, e_i)\}_{i \in [1, N]}$ where e_i is the labelled KB entity for mention m_i . Similarly, model predictions in that corpus are represented as $\{(m_i, \hat{e}_i)\}_{i \in [1, N]}$. To evaluate the model predictions against the labels in the development and test sets, three main metrics were used:

Micro-accuracy This metric computes the overall proportion of mentions for which the model predicted the correct entity [248]. In other words, it is the percentage of cases where $\hat{e}_i = e_i$ over all the evaluation examples N . This metric is often a good indicator of how the model performs overall and what percentage of predictions can be expected to be correct for a given model. However, if the dataset is highly imbalanced, a model might exhibit high micro-accuracy due to high performance on those KB entities with more evaluation examples and poor performance on KB entities with fewer evaluation examples.

Macro-accuracy To evaluate how well the model performs across KB entities, the macro-accuracy applies the same weight to each KB entity [248]. Specifically, the accuracy was first independently computed for each KB entity appearing in the evaluation set, and then, the average across all the KB entity accuracies was calculated. This metric can help identify whether a high micro-accuracy hinders low accuracy on entities with fewer mentions in the evaluation set.

Mean Rank The metrics mentioned above were selected to assess EL performance based on the evaluation used by Wu et al. [128], Logeswaran et al. [246], Humeau et al. [247]. However, they only considered a prediction as correct if the predicted entity equals the annotated one. EL systems often rank KB entities according to their likelihood of being the correct link for a given input mention. Since some KB entities might have very similar biomedical meaning (e.g. *in vitro* intrinsic clearance vs *in vivo* intrinsic clearance), assessing the rank of the correct entity predicted by the model across all KB entities can be informative to understand potential causes of misclassification. For example, low accuracy and high mean rank would indicate that a system is uncertain between a few specific KB entities, but the correct entity is often ranked highly. Correct mention predictions (true positives) had a rank of 1, while incorrect predictions could rank from 2 to 67. For a given prediction, the mean rank starts by computing one minus the predicted rank for the correct entity (r) over the KB size (i.e. 67): $1 - \frac{r}{67}$. Then, the mean over all the predictions is performed [249]. Hence, the mean rank goes from 0 to 1 and provides an informative metric of how highly the correct entities are ranked relative to all KB entities. For instance, in a KB of 100 entities, a mean rank of 0.9 would indicate that the correct entity for a given prediction is, on average, ranked amongst the top 10 candidates.

These metrics provided an assessment of the model performance from different perspectives but merging the performance across entities. A more detailed evaluation was performed through multiple plots to assess the model performance on specific KB entities, especially those with limited or no training samples.

5.3 Results and Discussion

5.3.1 Corpus statistics

Table 5.1 shows the statistics of the annotated dataset. Independent annotators provided at least two annotations for each instance in the development and test sets. The inter-annotator agreement (IAA) given by the pairwise K coefficient ranged from 0.78 to 0.82. This range can be interpreted as a high IAA according to Cohen [250] and considering the number of candidate classes for every mention. The pair-wise micro- and

macro-accuracy between annotators on the development set were 92.11% and 87.23%, respectively. Since many conflicting annotations had a common source of disagreement, a significant improvement was observed after resolving examples in each annotation batch and updating guidelines. It was noted that classifying the parameter mentions into general categories (e.g. clearance, half-life, AUC) was relatively simple for annotators. However, identifying the specific parameter type (e.g. intrinsic vs total clearance) often required in-depth expertise, detailed reading of the context, and guideline checks. Consequently, the labelling process was very time-consuming compared to previous annotations performed in this thesis (i.e. document classification, NER). The average annotation time per instance was approximately 2.5 minutes including resolution of disagreements.

The annotated mentions in the training set only contained mentions for 70.15% of KB entities, leaving 29.85% without explicit training data. Additionally, a large proportion of entities had less than ten mentions in the training set. These results were expected since there are some parameters that authors tend to report in most PK studies, whereas others depend on the study type (e.g. compartmental vs not compartmental modelling). Additionally, due to errors by the NER model and parameters not covered in the KB, $\approx 12\%$ of the examples were labelled as *NIL*. The test set exhibited similar statistics to the training set. It is worth considering that 10.44% of KB entities appearing in the test set did not have any training mentions and were considered the zero-shot cases.

TABLE 5.1: Corpus statistics stratified by the training, development and test sets.

Dataset	# Sentences	annotations/sentence	% of KB entities covered
Training	1000	1 + review	70.15
Development	500	2 + resolution	53.73
Test	1000	2 + resolution	71.64

The classes had a very sparse frequency across the dataset, and it was clear that much more labelled data would be required to accurately represent the underlying entity frequency distribution in the PK literature. However, given the large number of entities in most KBs, this is often the case in annotated EL datasets [98]. Therefore, it was considered a good setting to train EL systems for PK parameters and to study how well they could generalise to entities with none or few training mentions. Selecting candidate mentions with a pre-trained NER model allowed annotators to focus exclusively on EL labels without considering NER labelling, which significantly simplified and accelerated the annotation process. However, this approach might propagate bias of the NER model to the dataset. Overall, the quality of this dataset might be improved in future studies by (1) fixing incorrect NER predictions, (2) including mentions that might have been missed by the NER model, (3) expanding the number of examples in the test set. By developing a dataset with both, NER and EL labels, solutions modelling both tasks simultaneously could also be explored, similar to Wiatrak and Iso-Sipilä 2020 [130].

5.3.2 Preliminary analyses

Model initialisation The first experiment performed compared initialising BERT with BioBERT v1.1 or BERT_{BASE} parameters, which have been pre-trained on general domain and biomedical corpora, respectively. Figure 5.2 shows the micro-accuracy on the development set when training over 20 epochs. As it has also been observed for other NLP tasks in the PK domain (document classifier and NER chapters), the BioBERT model had a consistently better performance than BERT_{BASE}, which suggests that pre-training in biomedical text is also crucial for EL of PK parameters. Furthermore, the performance difference is particularly large in early training stages, indicating that BERT_{BASE} requires many more training steps and potentially more training data than BioBERT to encode relevant information for linking PK parameters. After this result, BioBERT was used in subsequent experiments.

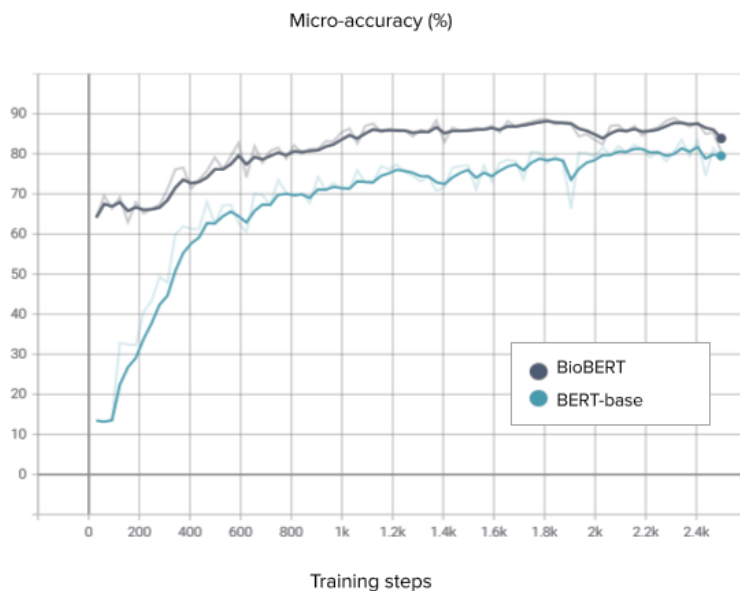


FIGURE 5.2: Performance of the bi-encoder on the development set when starting the model with BERT-base or BioBERT parameters. Experiments were performed over 20 epochs, and one epoch = 125 training steps. Shaded lines represent the raw values, which were smoothed using an exponential average with a weight of 0.1 (non-shaded lines).

Sharing bi-encoder parameters The effect of using one (shared) or two (not-shared) BERT models to encode mentions and KB entities can be observed in Figure 5.3. Although previous studies only used separate models for encoding mentions and entities [128, 246, 247], in this work, it was shown that sharing the same model parameters for encoding mentions and entities provided higher performance ($\approx \Delta 10\%$ micro-accuracy)

and earlier convergence. Specifically, it was observed that when not sharing the bi-encoder weights, during the initial six epochs, the model did not improve the micro-accuracy or mean rank on the development, and then it suddenly started learning relevant patterns. However, the performance was always better when using a single model.

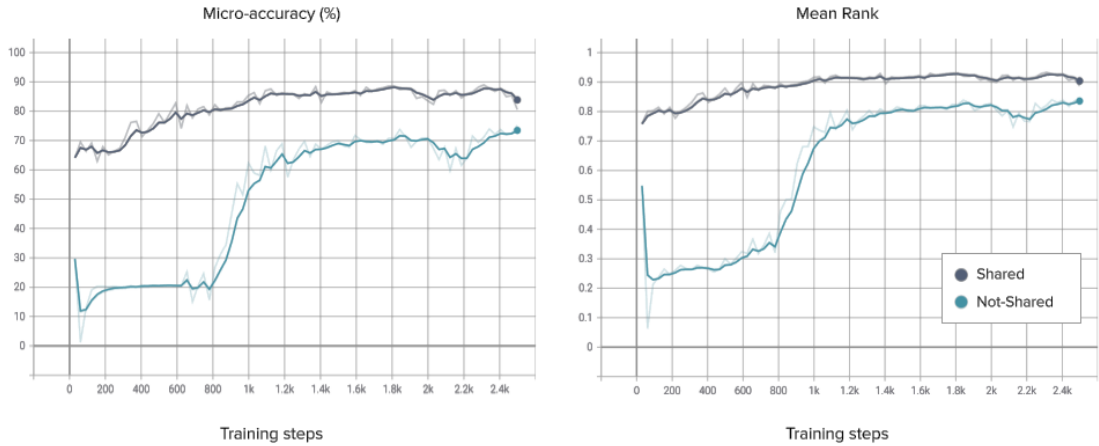


FIGURE 5.3: Performance of the bi-encoder on the development set when sharing or not sharing parameters. The left- and hand-side panels show micro-accuracy and mean rank, respectively. Experiments were performed over 20 epochs, and one epoch = 125 training steps. Shaded lines represent the raw values, which were smoothed using an exponential average with a weight of 0.1 (non-shaded lines).

Although the type and structure of textual information on the KB are very different from the one in scientific sentences, it is worth considering that the shared model tuned approximately 110M parameters while 220M were backpropagated during training of the not-shared model. Given the small number of training samples on this dataset and the limited number of KB entities, it is likely that larger models do not encode additional information but require a longer time to minimise the training loss. Overall, this result suggests that when training bi-encoder architectures for EL in small KBs and limited training data, it might be beneficial to use the same model for encoding KB entities and mentions in context. However, further work is needed to analyse this result in other domains and better understand which scenarios could benefit from sharing parameters. Wu et al. [128] showed that Knowledge Distillation could be used to improve the bi-encoder performance for candidate generation. Future studies on this dataset might also benefit from Knowledge Distillation to find an optimal trade-off between model size and accuracy while exploiting the transfer-learning benefits of pre-trained transformer models.

Negative samples Seven negative KB entities were used for every training sample in the initial model configuration. However, the effect of using a different number of negative samples was examined, and results are displayed in Figure 5.4. When using only three negative KB entities, the model exhibited worse performance. The model had

to select one out of 67 possible entities during evaluation. Since negatives were randomly sampled from the KB, it is likely that when only comparing the correct entity against three negatives during training, the task became too simple compared to the evaluation setting. In other words, since the main challenge of EL often consists of discriminating the correct entity against similar ones (e.g. clearance vs renal clearance), it is likely that hard negatives did not appear enough during training when using three negative samples. Performance was significantly improved when comparing positives with half of the knowledge base (33 negative samples) in comparison to seven negatives. However, when using 60 negatives, the performance was slightly worse than using 33.

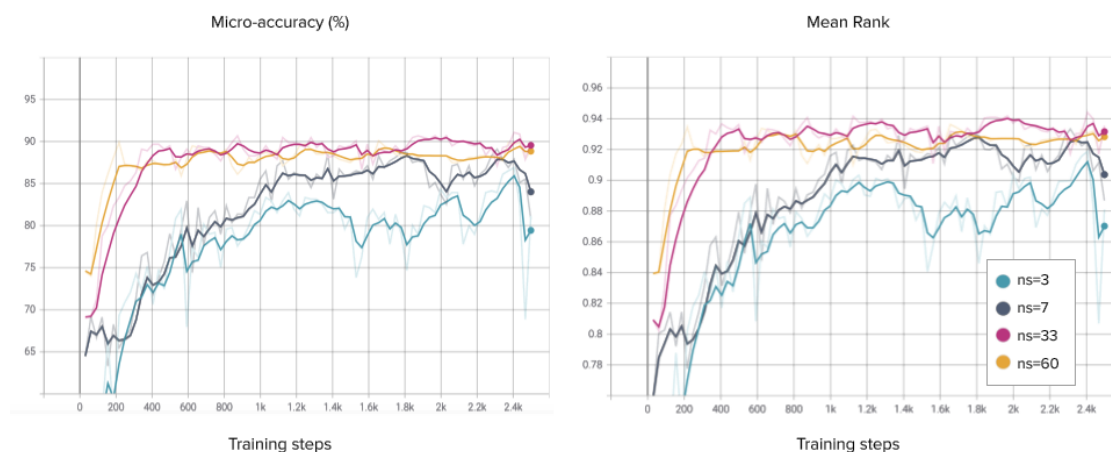


FIGURE 5.4: Performance of the bi-encoder on the development set when changing the number of negative samples used for training. The left- and hand-side panels show micro-accuracy and mean rank, respectively. Experiments were performed over 20 epochs, and one epoch = 125 training steps. Shaded lines represent the raw values, which were smoothed using an exponential average with a weight of 0.1 (non-shaded lines).

These results indicate that PK EL benefits from observing challenging negative entities during training. However, this might be achieved by comparing the correct entity against a specific proportion of randomly sampled negatives, but using all KB entities as negatives might not be optimal. This observation might be explained because the bi-encoder maximises the separation between all negatives (regardless of how similar they might be to the correct entity) and the correct entity. However, not using the whole knowledge base for sampling negatives might provide a regularisation effect, allowing entities with similar biomedical meanings to be encoded closer in an embedding space by only appearing in some cases as hard negatives. Given the results observed, 33 negative samples were used for all the subsequent experiments.

Due to the significant effect of the number of negative samples on model performance, future studies might benefit from further studying this behaviour. For instance, one potential approach would be to select negative samples based on the proximity of entities within a knowledge graph, like Ahrabian et al. [251].

5.3.3 Baseline comparison

The performance of the bi-encoder and softmax classifier on the development set is shown in Figure 5.5. While the bi-encoder reached high performance after ≈ 3 epochs and then slight improvement was observed, the softmax classifier only seemed to start converging by the end of training. This result suggests that the bi-encoder might transfer the knowledge learnt from some entities to others more efficiently due to its distance-based loss. Additionally, the bi-encoder achieved higher micro-accuracy and mean rank at the end of training.

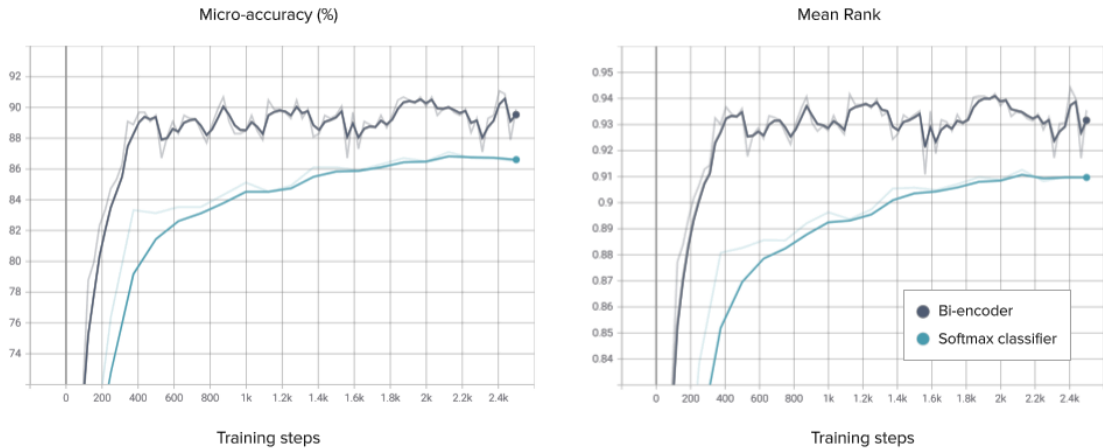


FIGURE 5.5: Performance when using the bi-encoder or the softmax classifier on the development set. The left- and hand-side panels show micro-accuracy and mean rank, respectively. Experiments were performed over 20 epochs, and one epoch = 125 training steps. Shaded lines represent the raw values, which were smoothed using an exponential average with a weight of 0.1 (non-shaded lines).

Table 5.2 also shows the results on the test set when selecting the model state with the highest micro-accuracy on the development set. Interestingly, over $\Delta 25\%$ improvement in macro-accuracy was observed on the test set between the bi-encoder and the softmax classifier. This result indicates that although the accuracy difference over all instances was around 4%, the bi-encoder improved significantly on those KB entities with very few training instances. Additionally, the macro-accuracy on the test set dropped 15.21% and 5.39% compared to the development set for the softmax classifier and bi-encoder, respectively. Hence although overfitting appeared on both models, the softmax classifier was much more sensitive to KB entries with few or no training examples.

TABLE 5.2: Summary table with performance metrics on the development and test sets comparing the softmax classifier against the bi-encoder.

Model	Development			Test		
	Mean Rank	Micro-accuracy	Macro-accuracy	Mean Rank	Micro-accuracy	Macro-accuracy
Softmax classifier	0.91	86.91	65.70	0.89	83.81	50.49
Bi-encoder	0.94	90.81	81.19	0.92	88.08	75.8

Overall, using the bi-encoder approach provided better average performance across all predictions, converged faster and dealt with few and zero-shot examples better when compared to the softmax approach. Hence, these results suggest that even in low resource scenarios (i.e. small KBs and limited training data), the distance-based loss implemented by the bi-encoder deals better with sparse frequency distributions than traditional multi-class sequence classifiers. Finally, it is worth noting that inter-annotation scores in the development set (micro- and macro- accuracy of 92.11% and 87.23%, respectively) remained significantly higher than the best-performing model, indicating that there is still significant room for model improvement in this task.

Other baselines could have been used in this study. For instance, dictionary and rule-based systems or EL models based on n-grams TF-IDF of named aliases are often competitive approaches in EL [149]. However, no list of entity aliases was available a priori for this study, and although some acronyms could potentially be normalised using the surface forms (e.g. $AUC_{0-\infty}$), many parameters required contextual information to be linked since many acronyms were used by multiple KB entities (e.g. CL, AUC, $t_{1/2}$, V_d). Nonetheless, using prior information such as frequency metrics of entity surface forms, known aliases or rule-based models could provide competitive non-ML baselines or additional information for the solution presented.

5.3.4 Few-shot performance

To better understand the performance of the models developed on KB entities with varying numbers of training instances, the accuracy of the test set was stratified. Figure 5.6 shows a box-plot in which the accuracy on the test set was first calculated per entity, and the entities with a similar number of training examples were grouped into the same box-plot. For instance, some entities on the test set had no training samples but had test samples. These are the group 0 in the x-axis of Figure 5.6.

As it can be expected in machine learning models, Figure 5.6 confirms that, on average, the model performance increases with an increasing number of training mentions for a given KB entity. It can be observed that the softmax classifier had a performance of 0% for entities appearing on the test set but not on the training set (zero-shot). This result was expected since no information regarding those entities was learnt during the training phase, and the model had no access to the KB descriptions. However, the bi-encoder showed sparse performance on entities with zero training samples, with accuracies ranging from 0 to 100% depending on the KB entity. The box plot also suggests that there might be a very small number of instances on the test set for entities with few training examples, which generates sparse and noisy metrics. Despite the low number of test samples, the performance of the bi-encoder seems to be consistently better for KB entities with less than 30 training samples. The difference between the bi-encoder and the softmax classifier for KB entities with more than 30 training samples

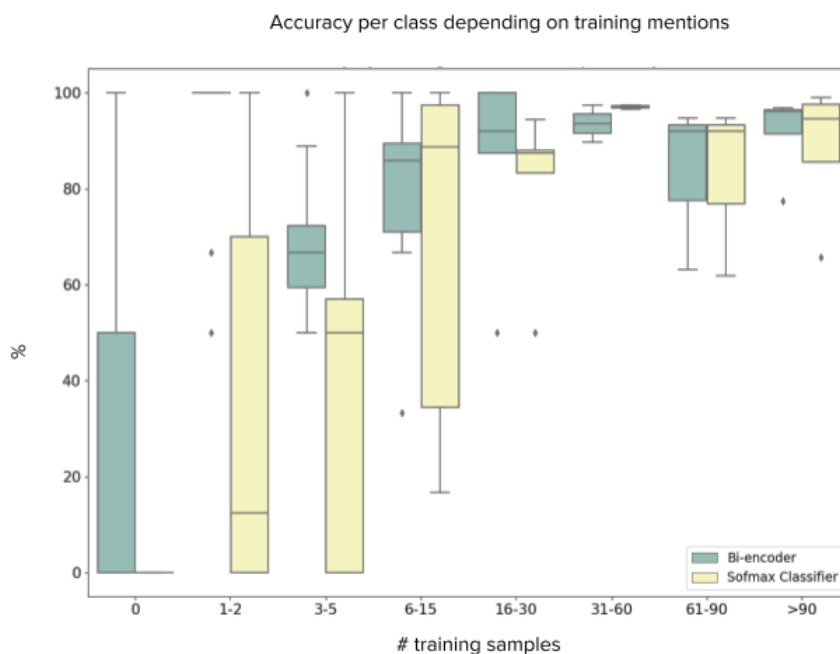


FIGURE 5.6: Box-plot showing the accuracy stratified by class of the bi-encoder and softmax classifier on the test set. The accuracy was computed by class (i.e. KB entity) on the test set, and classes were grouped depending on how many mentions of that class they had on the training set.

was not noticeable. Apart from exhibiting the outperformance of the bi-encoder on a few-shot setting, these results could also be used to establish a minimum number of training samples to achieve a desired minimum performance for a certain entity when gathering more training data. For instance, one might consider collecting more than 16 training examples for an entity to be correctly linked in most cases.

5.3.5 Error analysis

A confusion matrix was generated to identify entities in which the model might be more uncertain. Figure 5.7 shows the predicted against the labelled entities on the test set when using the bi-encoder. The colour determines the proportion of predicted entities from a labelled entity category. From this plot, it was observed that the model was most confused in PK parameters that had the same title and description but were measured *in vitro* or *in vivo* (e.g. intrinsic clearance (CL_{int})). The information required to link these cases was highly contextual since determining whether a parameter was measured *in vitro* or *in vivo* required understanding the conditions of the study (e.g. clinical trial against laboratory conditions), and, on some occasions, it was not available by only reading the sentence.

Entities with lower accuracy often exhibited confusion between 2-4 classes. Hence, future studies might benefit from using the information in this confusion matrix as a proxy for

active learning approaches. For instance, high confusion was observed between volumes of distribution types V , $V1$, $V2$ and NIL entities, which might be caused by the wide variety of overlapping acronyms and surface forms observed for those entities. To improve the model performance, one could focus on sampling mentions where at least two out of the V , $V1$, $V2$ and NIL entities were ranked highly by the model to reduce the uncertainty shown in those types of mentions. However, the feasibility of active learning approaches was left for future exploration.

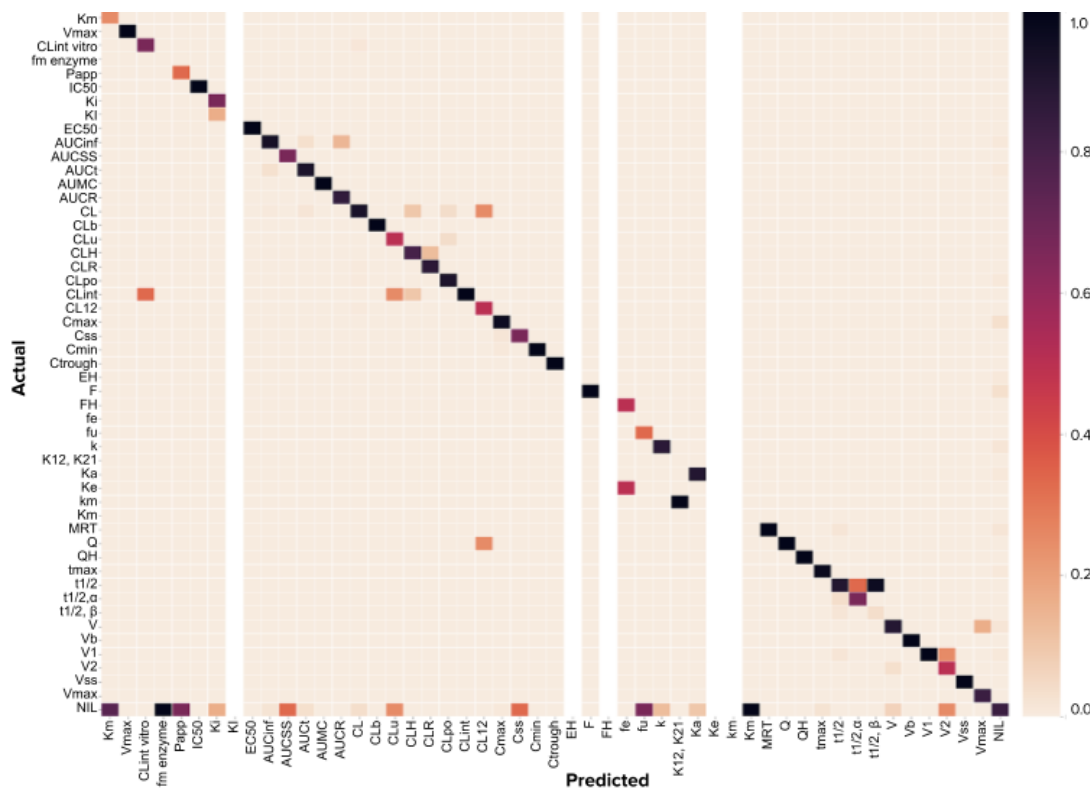


FIGURE 5.7: Confusion matrix of the bi-encoder on the test set showing the predicted vs true labels per class. The scale is calculated as the proportion of samples predicted as one class (Predicted) that belonged to an annotated (Actual) class. Empty columns correspond to entities that were never predicted in the test set.

5.3.6 Zero-shot parameters

A qualitative analysis was conducted to explore the model behaviour when new entities were added to the KB. For this, field experts provided the KB fields (title, condition, description and units) for two parameters that might be of relevance during preclinical drug development but were not included in the original KB during training: fraction of the drug absorbed from the gastrointestinal tract (f_a) and free:brain plasma ratio (k_{puu}). After adding these entities to the KB, the bi-encoder performed inference on the whole KB and the initial collection of ($>1.2M$) sentences with parameter mentions. Then, a collection of mentions that were classified as either f_a or k_{puu} were randomly

sampled and analysed. Tables 5.3 and 5.4 show some of the mentions classified as f_a or $kpuu$ in context.

TABLE 5.3: Examples of mentions classified as fraction of the drug absorbed (f_a).

#	Mention in context linked as f_a
1	The AUC was calculated according to the trapezoidal rule and the absorption fraction (f) was acquired by Eq.2.
2	Similarly, a fraction of dose absorbed (Fa) , furnishes information on the extent of drug absorption from the intestinal mucosa (54).
3	The fractional absorption (F) during a 24-hr period was 47.4 (22.7-98.1)%.
4	The cumulative fraction of bupremorphine absorbed after SC administration is shown in Figure 3.
5	The oral bioavailability (F) was calculated as follows: $F = (AUC_{0-\infty,po} \times Dose_{iv}) / (AUC_{0-\infty,iv} \times Dose_{po})$

TABLE 5.4: Examples of mentions classified as free:brain plasma ratio ($kpuu$).

#	Mention in context linked as $Kpuu$
1	The blood and brain PK parameters (C_{max} , T_{max} , $t_{1/2}$, AUC_{τ} , $K_{pu,u,brain}$) were estimated...
2	The unbound brain/unbound plasma AUC ratio ($K_{p,uu}$) was determined from the total brain and plasma AUCs and ...
3	The relationship between the change in ratio and the change in PKs was analysed using Kendall's tau .
4	Despite advantages of $K_{p,uu}$ from steady state after constant intravenous (i.v.) infusion compared ...
5	These K_{pub} values were used as priors for making inferences in man.

The model provided high precision for the parameter mentions retrieved. The mentions retrieved had some variability of surface forms, but they were highly similar to either the parameter title or the description. Additionally, some bias was observed for acronyms with similar surface forms. For instance, in example 5 of Table 5.3, F does not refer to the fraction absorbed from the gastrointestinal tract but to the drug's bioavailability. However, it is worth noting that the drug's bioavailability (F) is generally considered to be a composite of three processes: $F = f_a \cdot f_q^6 \cdot f_h^7$ [252] and f_a and F are often incorrectly used interchangeably, which created high confusion on those mentions. Additionally, in example 4 in Table 5.4 the model classified Kendall's tau as $kpuu$ while this was an error of the NER model. Overall, the model developed seemed to provide a framework for drug developers to include new KB entities without training data and retrieve parameter mentions with similar surface forms. However, it is also worth considering that *NIL* mentions or incorrect acronyms with highly similar surface forms were also detected in the newly added categories. Finally, the ability of the model to account for more diverse mentions remains to be quantified and further explored.

5.4 Conclusions and Future work

This chapter presented the first work to develop NLP models that link mentions of PK parameters to a KB of PK parameter types. Objective O_{3-1} was tackled by classifying 2500 PK parameter mentions from scientific sentences into one of 67 possible categories.

⁶ Fraction of the drug escaping metabolism or efflux in the gut membrane.

⁷ Fraction of the drug escaping hepatic metabolism prior to entering the systemic circulation.

To the best of our knowledge, this represents the first annotated dataset for PK EL. A high inter-annotator agreement was observed after developing and iteratively updating detailed guidelines. The model developed was based on a previously published bi-encoder that independently encoded KB entities and mentions in context. To address objective O_{3-2} , simplifications were proposed to adapt previous architectures to EL scenarios with small KBs and limited training data, which improved the model’s performance. The solution proposed efficiently dealt with the highly imbalanced dataset, outperforming sequence classification models and showing significant benefits on KB entities with few or no training mentions. To fulfil objective O_{3-3} , an in-depth comparison between sequence classification models and the bi-encoder was performed, and the causes of model errors were discussed in detail. The final performance achieved by the bi-encoder was considered high given the limited number of training samples and the complexity of the task (over 85% micro-accuracy).

One of the main limitations of the dataset developed is that it relies on spans predicted by a NER model. Although highlighting predicted spans allowed faster annotation and the model achieved high performance on PK NER, this approach might not accurately represent the PK parameters’ distribution in the literature. Future studies might benefit from datasets that annotate both named entities and their KB entries, which would allow joint training and evaluation of NER and EL tasks [130, 253, 254]. However, developing such datasets while preserving a sufficient number of named entities for effective model training remains an open challenge. Furthermore, the models presented relied on large pre-trained transformer models. Due to the small KB and limited training data, simpler baselines using heuristic or frequency-based approaches could be explored as an alternative or used as prior information for machine learning models with distance supervision. Finally, future work could use the model presented to enhance ADME dataset construction and improve DDI characterisation by understanding the specific PK parameters mentioned in the scientific literature.

Chapter 6

End-to-End Relation Extraction of Pharmacokinetic estimates

The content of this chapter will be presented at the World Conference on Pharmacometrics (WCoP) 2022.

6.1 Introduction

The need for a large and standardised database of PK measurements has been, for a long time, recognised as a significant limitation in the drug development pipeline [10, 11, 255, 256]. Such a database could accelerate the curation of large ADME datasets in pre-clinical drug development, improve PBPK models by providing informative parameter priors, or allow large-scale PK meta-analyses between drugs or study populations [10]. As a result, efforts such as PK/DB [10] have started, aiming to curate parameter estimates from the scientific literature. However, PK/DB is manually curated by pharmacometricians, which limits the ability of PK/DB to cope with the vast and increasing PK literature. Unfortunately, apart from the manually-curated PK/DB, no other open and freely accessible database of PK parameters exists so far [10]. This chapter studies the main challenges to automatically extracting PK data from the literature while addressing the core problem of recognising and extracting numerical estimations of multiple PK parameter types. For text mining tools to efficiently extract PK measurements, it is essential to recognise multiple entities and their relations from the text.

The top panel of Figure 6.1 illustrates a sentence found in the scientific literature together with key entities that would need consideration in the extraction of PK data. The bottom panel displays the information in a tabular format that could be used to construct a database of PK measurements, relating each measurement to its parameter mention, units, chemical compound, route of administration and conditions of the study population. Text mining systems need to recognise (and potentially link) those key entities and predict their relation to the relevant measurement to go from the raw sentence to the tabular format observed in Figure 6.1. From an extraction perspective, this problem consists of two sequential tasks:

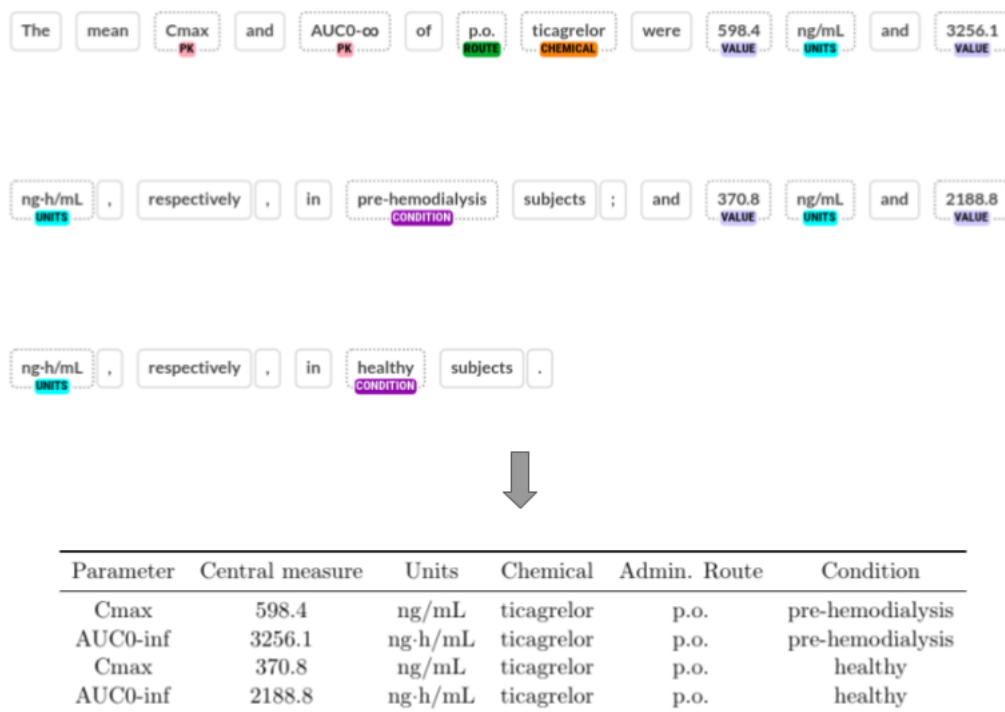


FIGURE 6.1: The top panel shows a sentence with the key entities to extract PK measurements highlighted. The bottom panel shows the sentence in structured format, with the key columns to build PK databases. The sentence was adapted from Teng et al. [3] for illustration purposes.

1. Identifying entities and relations **directly linked** to the PK measurement such as: parameter mentions, their central and deviation values, units or potential comparative terms (e.g. “The Cmax was above 598.4 ± 144 ng/mL”).
2. Identifying **contextual** entities and their relation to a specific central measurement. Amongst others, contextual information could include the drug for which the PK parameter has been estimated, the route of administration, the given dose, the species in which the parameter was measured, demographic information, or the studied population’s conditions.

Regarding the second task, the recognition and linking of most of the relevant contextual entities have been widely studied in biomedical NLP (e.g. chemical compounds, species, diseases) [257]. As a result, multiple tools are available out-of-the-box for recognising and linking these entities to specialised ontologies [236, 258]. However, relating these mentions to the correct PK value remains an open challenge. For example, estimated values of PK parameters are often reported together with their units and parameter mentions within the same sentence in the abstract or results section. However, contextual entities might be mentioned in other sections of the paper¹, making the extraction and

¹Demographic and dosing information is often described in the Methods sections while PK estimations are commonly reported in the Results section.

evaluation of this task particularly challenging. As a result, developing annotations and evaluation frameworks that consider contextual entities of PK measurements would require labelling a wide range of concepts and their relations at multiple levels (entities, knowledge base identifiers, relations) and integrating information in multiple sections of the paper. Additionally, PK measurements would need to be pre-labelled to associate their contextual information.

Automatically extracting central measurements of multiple parameter types and their units and deviation values from the scientific literature would already provide a valuable database for PK applications. For instance, Wang et al. [11] showed that using dictionary-based methods provided high precision to filter PK measurements for one specific drug, administration route, and study population. Therefore, once PK measurements are extracted from scientific publications, different approaches could be used to filter for the relevant context to a specific use case or accelerate the search of PK information for a given compound. For this reason, this chapter focuses on developing text mining resources that exclusively tackle the first task, and complementing the extracted values with relevant context will be approached in future work.

Related work To the best of our knowledge, no existing text mining resources exist to extract PK measurements for multiple parameter types together with their units and deviation values. Wang et al. [11] performed a feasibility study, using dictionary and rule-based methods to extract systemic and oral clearance values of intravenous midazolam in healthy volunteers from scientific abstracts. The text mining approach developed by Wang et al. exhibited high precision on extracting estimates of systemic and oral clearance for a specific drug and context. Yet, no further work has been found extending the approach of Wang et al. to other types of PK parameters, study populations or full-text sections. After careful examination of the methodology developed by Wang et al., the following hypotheses were generated regarding the limitations of adapting Wang et al.’s approach for constructing a public database of PK parameters:

1. **Evaluation framework:** For pharmacometricians to rely on text mining tools that extracted PK parameter estimates, it is essential to provide a robust evaluation of the extraction pipeline. Although precision and recall estimates were reported by Wang et al., the evaluation was not stratified by NER but exclusively relied on the extraction of central values. Additionally, the approach was only evaluated on midazolam abstracts, but the performance on sentences from full-text or abstracts from other drugs was not studied. In the PK-Ontology-corpus of Wu et al. [43], PK parameter mentions and numerical expressions were annotated. However, since their approach was focused on DDIs, relations between parameters, measurements and units were not labelled. Therefore, it does not provide an evaluation framework for extracting PK estimations.

2. **Recognition of PK parameters:** The methodology relies on a list of terms to find mentions of systemic and oral clearance. However, as discussed in chapter 4 and 5, PK parameter surface forms are highly variable, including multiple acronyms and complex spans. Therefore, dictionary and rule-based approaches are not suitable for detecting mentions of multiple PK parameter types effectively, limiting the ability of Wang et al.’s method to extract measurements for other parameters. This challenge was later discussed by Wu et al. [43].
3. **Recognition of units, deviation measurements and relations:** To extract clearance measurements, dictionary and rules implemented by Wang et al. [11] relied on expressions following the pattern: $\langle Value \rangle \pm \langle Value \rangle \langle Units \rangle$ and accounted for little flexibility on the location of multiple $\langle Parameter \rangle$ mentions in the same sentence. When multiple parameters or study type settings were mentioned in the same sentence, the approach focused on discarding all the information that was not exclusively related to their parameter and context of interest. Although this approach proved effective for extracting the systemic and oral clearance of midazolam in healthy volunteers, many PK parameters might be mentioned in the same sentence, and their measurements might be reported for different study designs. Additionally, some parameters might be reported in ranges and might not contain units associated with them (e.g. fraction of the drug unbound, AUC ratio). Finally, deviation measurements are not always expressed after a \pm mention. Therefore, it is unlikely that Wang et al.’s approach has enough flexibility to account for multiple parameters, units, deviations and their relations.

Despite the limitations of Wang et al.’s work to account for a higher degree of variability in parameter mentions, units and numerical expressions, their approach was sufficient to reproduce the distribution of systemic and oral clearance of intravenous midazolam for a specific population. Hence, if numerical values are successfully extracted for multiple parameters, their filtering approach might be effective at obtaining high precision of parameter values for a specific drug, route and study population and discard irrelevant PK values for a specific use-case. To address Q₄ of this thesis, the following objectives were established in this chapter:

- O₄₋₁ To design an annotation framework that allows extracting PK estimations from the scientific literature in an structured format.
- O₄₋₂ To develop training and evaluation corpora that tackles the extraction of PK parameter estimates involving: (1) high diversity of PK parameter types and associated units and (2) sentences from abstract and full-text sections.
- O₄₋₃ To adapt current NLP architectures for end-to-end RE to the extraction of PK measurements and perform ablation studies on their components.

- O₄₋₄ To compare architectures that model NER and RE jointly against models that optimise for a single task.
- O₄₋₅ To apply data augmentation techniques that increase entity mentions' diversity and introduce prior knowledge about the relation between specific PK parameters and their units.
- O₄₋₆ To inspect the leading causes of model errors and discuss potential ways to improve them.

6.2 Methods

The methods in this section described the development of suitable architectures and annotated data to train and evaluate RE models to extract measurements of pharmacokinetic parameters and their units from scientific text. The methods are divided into the following sections:

1. **Corpus construction:** Procedure employed to annotate entities and relations from sentences.
2. **Model development:** Description of the architectures employed to recognise PK entities and their relations.
3. **Data augmentation:** An heuristic approach to incorporate prior knowledge about the relation between specific PK parameters and their most common units into neural RE models.

6.2.1 Corpus construction

An annotated corpus was developed to train and evaluate end-to-end pipelines that extract PK measurements from text, named PK-REX corpus.

6.2.1.1 Entities, relations and guidelines

To automatically extract PK estimations from the literature with their relevant information, the following entities were considered and annotated at the sentence level (shown in Figure 6.2):

1. **PK:** Mentions of parameters. This entity refers to spans mentioning PK parameters, and it is the same concept as the entity described in chapter 4.

2. **Units:** Spans of text corresponding to units of numerical PK estimations.
3. **Value:** Spans encapsulating numerical estimations related to PK parameters (i.e. central and deviation values).
4. **Range:** Two values defining the boundaries of a PK estimation.
5. **Compare:** Textual mentions that provided information about whether a specific value/range mention was the extreme of an estimated parameter (see *higher* in Figure 6.2). This entity appeared with low frequency, but it was important for detecting extracted measurements that were not central estimations of a certain parameter.

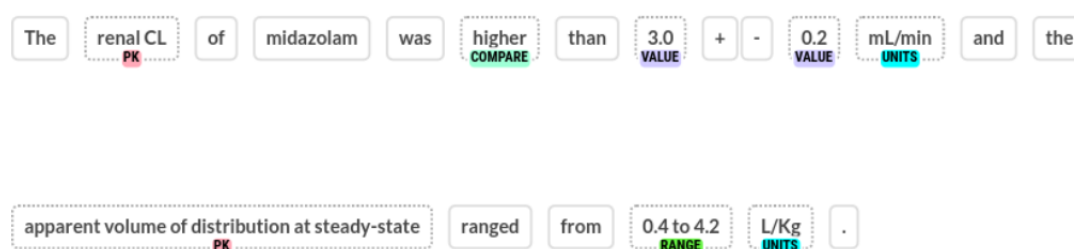


FIGURE 6.2: Example of a sentence after all entities had been annotated.

Three relations types were considered between entities to extract structured information from raw sentences in a usable format (see Figure 6.3):

1. **Central_{val}**²: This relation type happened between PK parameter mentions and their estimated values or ranges. This involved central measurements of the parameter but not measures of deviation or % of increase concerning other experimental conditions. The entities between which this relation could happen were:
 - PK ↔ Value/Range
2. **Deviation_{val}**³: This relation type informed whether a specific measurement was the deviation of a central measurement and only happened between the entities:
 - Value/Range ↔ Value/Range (involved in a *Central_{val}* relation)
3. **Related**: This relation type complemented values/ranges with their units or compare terms and only happened between the following entities:
 - Compare ↔ Value/Range
 - Units ↔ Value/Range

² Abbreviated as C_VAL in the annotation interface.

³ Abbreviated as D_VAL in the annotation interface.

Note that if an automated system can detect these entities and relations, there is no need to understand the directionality of the relations in order to map the top panel in Figure 6.3 to the tabular format in the bottom panel. For instance, if there is a $Deviation_{val}$ relation between two values, by looking at the value involved in a $Central_{val}$ relation, one could disambiguate the central from the deviation value. For this reason, the directionality of the relations was not considered in this work.

It could also be possible to design a system that predicted a single relation class for every entity pair (relation/no-relation) and perform the mapping of the top to bottom panels of Figure 6.3 by looking at the entity types involved in each predicted relation. For instance, if a relation happened between a parameter and a value, it would be straightforward to understand that it is the central measurement. However, the reasoning behind having different relation classes instead of a single relation class is that the syntax behind instances of each of the three relations is very different. Therefore, three relation classes were used to enhance the encoding of distant embeddings between instances of different relation classes, preserving their syntactic differences in the model representations.

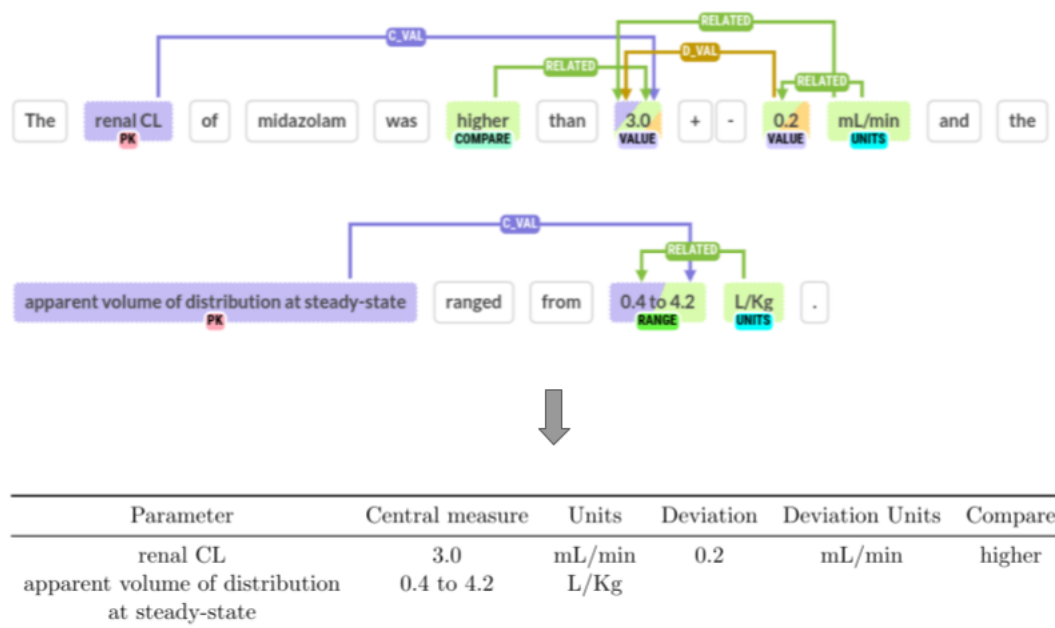


FIGURE 6.3: The top panel shows a sentence where all entities and relations had been annotated. The bottom panel shows how the annotated entities and relations can be mapped into a tabular format that can be integrated into a database of PK measurements.

6.2.1.2 Source

Once relevant entities and relations were defined, the following step consisted of sampling texts to annotate. All the relations tackled in this task appeared between entities within

the same sentence in all cases observed. Therefore, the sampling and annotations were performed at the sentence level. The candidate pool of 1,443,044 sentences (50% abstract and 50% full-text) described in the NER chapter (section 4.2.2.1) was used as a starting pool in this study to develop models that could efficiently deal with PK measurements reported in both, abstract or full-text sentences.

As reported in the NER chapter results, only 16.4% of sentences from the initial candidate pool mentioned PK parameters (for details, see Table 4.3). Additionally, many of these sentences mentioned PK parameters but not their estimated measurements. Therefore, a filtering protocol was applied to promote developing a corpus with a wide variety of PK mentions and relation instances. First, the optimal PK NER model developed in chapter 4 was applied to all the candidate pool sentences. Then, a set of heuristics were developed to filter those sentences that at least had: (1) one PK mention detected by the NER model and (2) a numerical value. Numerical values were detected with a regular expression matching continuous digits with an optional dot between them: “`d+(\.\d+)?`”⁴. Values preceded by a match of any of the following were not considered: “`(table|group|compound|equation|figure|stud(i|y)|phase|formulation|product|fig|fig.|day|trial|subject|eq|eq.) (s)?`”. Similarly, values followed by a lowercased match of any of the following were not considered either: “`(time|patient|phases|sample|subject|dose) (s)?`”. This preliminary NER and filtering approach was applied to the original candidate pool to reduce the number of sampled sentences without PK measurements reported.

From the candidate pool of sentences containing at least one PK parameter mention and one value (detected with the regular expressions), 3600 instances were randomly sampled without replacement and divided into 2100, 500 and 1000 instances for the training, development and test sets.

6.2.1.3 Annotation

Interface The annotation interface was developed in Prodigy and allowed annotating both entities and relations at the sentence level. As observed in Figure 6.4, annotators were presented with a single sentence at a time, and they could swap between the entity and relation annotation modes. Additionally, a comment box was included to store doubts or thoughts for a particular example. Analogous to chapter 4, the annotation interface was deployed in an Azure server, and annotators accessed the task through a unique web link. The annotations of named entities were represented at the character level, and relations were defined with the unique identifiers of each entity and their relation class. Annotations were exported in JSONL format, where each line contained a dictionary with the annotations of one sentence.

⁴Regular expressions mentioned in this chapter were implemented in Python 3.8 using the *re* package: <https://docs.python.org/3/library/re.html>.

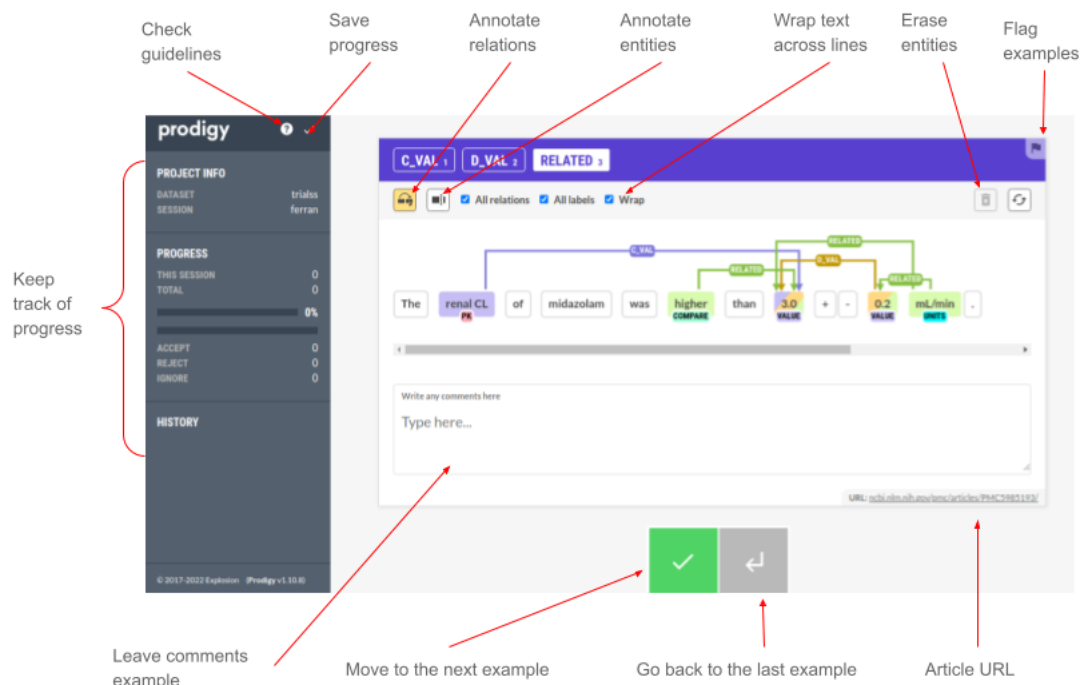


FIGURE 6.4: Screenshot of the interface used to annotate entities and relations from scientific text. The example displays a single sentence after entities and relations were annotated.

Candidate values and ranges were pre-highlighted in the interface using a rule-based system. PK terms were pre-highlighted using the optimal NER model from chapter 4 and a list of dictionary terms was used to pre-annotate Compare entities. The pre-highlighted entities often had to be corrected by the annotators, but it allowed them to quickly focus on the relevant parameter mentions and their surrounding information. After one sentence was annotated and accepted, a set of quality checks were performed and only if a set of conditions were met the annotator could move to the following example. The checks included: (1) ensuring that each relation was between the allowed entities (e.g. if a *Central_{val}* was annotated between two Value mentions a pop-up window would appear and prevent moving forward), (2) ensuring that all the annotated entities were part of relations, and (3) checking that if a *Deviation_{val}* was annotated, one of the values as part of a *Central_{val}* relation while the other was not.

Guidelines Prior to beginning the labelling task, initial annotation guidelines were developed and distributed to the annotators. As new complex cases and disagreements emerged during the annotation process, guidelines were updated with those challenging examples and their resolutions. Appendix C: Guidelines Relation Extraction contains the final guidelines for this work.

Annotation team The multiple entities and relations involved in this task required significant PK expertise and attention from the annotators, who spent, on average, more than a minute per sentence. Given the relevance of this task for academic and industrial applications and the need for a relatively large (in comparison to the NER study) team of annotators to perform the task, a variety of potential end-users were contacted a priori to discuss applications of the extracted data in drug development and ask for annotation on a volunteering basis. Twelve annotators were involved during the annotation process, including two senior pharmacometricians from University College London’s Pharmacometrics group, one postdoctoral researcher in pharmacokinetics from Uppsala University, two postdoctoral researchers in pharmacokinetics from the Mahidol Oxford Tropical Medicine Research Unit, one principal scientist from the DMPK team at AstraZeneca, one pharmacometrician from Sanofi, three senior pharmacometricians and one final-year pharmacy student from Universitat Ramon Lull and one PhD student from UCL with NLP and PK expertise.

Annotation process Each volunteer assisted in a training session where annotation guidelines were reviewed, and about 30 example sentences were annotated and discussed in a group. After this session, annotations were performed in batches of 200 sentences. Each batch followed a three-step procedure similar to the one employed by Hope et al. [227]: (1) initial annotation by one PK expert, (2) review by another PK annotator and (3) final check focusing on span boundary consistency by an annotator with bio-NLP experience. After the second step in each batch, comments from the first and second steps were reviewed, and feedback regarding incorrect annotation patterns was given to the annotators. Finally, the model predictions on the development and test sets were displayed in a terminal window together with their annotated version, which allowed identifying potentially missed entities and relations during the annotation.

6.2.2 Model development

In this section, the task of end-to-end relation extraction is first presented together with the main evaluation metrics used. Then, the architecture developed for end-to-end extraction of PK measurements is described and compared with previous work.

6.2.2.1 Task definition and evaluation

Task End-to-end RE aims to identify named entities and extract relations between them. Given some input text X , the output of any end-to-end RE system is a list of triplets in the form of (s_i, s_j, r) where $s_i, s_j \in S$ and $r \in R$ and S denote all the possible spans in X and R the set of pre-defined relation types [259]. Hence, the annotated data was represented as a list of sentences, each with their corresponding list of relation

triplets and compared to model predictions in the same format. Because end-to-end RE systems need to (1) identify candidate spans and (2) predict relation classes for pairs of spans, this task is often decomposed into two sub-tasks:

1. Named Entity Recognition: which attempts to detect the list of entity mentions (i.e. spans) and their type $\mathcal{E} = \{PK, Units, Value, Range, Compare\}$ from the input text X .
2. Relation Extraction: which compares all pairs of spans in X and outputs a relation class for each pair $R = \{Central_{val}, Deviation_{val}, Related\}$.

Evaluation Evaluation is performed for both NER and RE, and the most common metrics are Precision, Recall and F_1 scores. The evaluation of the NER predictions is analogous to the one in chapter 4, but instead of one, there are five types of entities in this task. As a result, for a given system prediction, F_1 scores were computed per entity, and macro and micro-average was performed across entity types to evaluate the overall system performance. The NER metrics reported in this chapter consider the strict matching of entity boundaries and types, but qualitative evaluations were performed regarding partial matches of PK parameter mentions. Different evaluation settings have been introduced for end-to-end RE systems regarding the span boundaries, and types of a specific relation triplet [259, 260]. Following recent guidelines on end-to-end RE evaluation [260] the strict criteria defined by Zhong and Chen [259] was used in this study. In this setting, for a prediction triplet to be correct, it needs to have the right relation class, and both the boundaries and entity types of each predicted span must be correct. In other words, given the following annotated vs predicted triplet:

```

annotation =
(
{"start":5, "start":12,"type":"PK"},
{"start":17, "start":20,"type":"Value"},
"C_VAL"
)

prediction =
(
{"start":5, "start":7,"type":"PK"},
{"start":17, "start":20,"type":"Value"},
"C_VAL"
)

```

The prediction is considered a false positive because it missed the span boundaries of the PK mention and a false negative because it missed the annotated relation triplet. Precision, Recall and F_1 scores were computed per each relation class. However, since predicting *Deviation_{val}* or *Related* relations without correctly predicting *Central_{val}*

makes the extracted data not useful, the evaluation of the overall system performance in RE was focused on the F_1 score of the $Central_{val}$ relations.

In summary, micro-averaged F_1 scores for NER and $Central_{val}$ F_1 score for RE were considered the main metrics to compare different architectures on the PK-REX corpus. Nonetheless, additional metrics were reported in the results section of this chapter to provide a deeper understanding of the system performance in both tasks.

6.2.2.2 Architecture

End-to-end RE has traditionally been tackled with a pipeline approach, training one model to extract entities and a separate one to classify relations between them [261–263]. However, recent work has shown state-of-the-art performance by sharing representations between tasks and modelling both tasks simultaneously by optimising a single loss function in a multi-task setting [238, 264, 265]. The latter approach has shown that the knowledge required to recognise entities can be helpful to extract relations and vice-versa [238]. This fact seems especially important when extracting PK measurements, where a specific value and PK parameter mention are only considered entities if the value is the central measurement of the mentioned parameter. In other words, to determine whether a specific span is an entity mention, it requires an understanding of whether it holds a relation with another entity in that sentence. For this reason, the architecture proposed in this section models NER and RE jointly to share encoded knowledge from both tasks.

The architecture is illustrated in Figure 6.5. Initially, an input sentence is tokenised into a sequence of sub-words using the BERT tokeniser. Then, tokens are passed through an encoder that aims to incorporate contextual information in each tokens’ representation. The output embeddings from the encoder (T_1, T_2, \dots, T_N) are then used to (1) recognise entities through the token classifier, (2) generate candidate pairs of predicted entities and (3) classify all pairs of recognised entities with a relation classifier. NER and RE use the same encoder to represent tokens and have one task-specific classification layer for each sub-task.

Encoder BioBERT v1.1 [122] was used as an encoder to generate contextual representations of input tokens. However, the effect of using different encoders was also analysed. First, experiments with BERT_{BASE} [2] were performed to analyse the effect of in-domain pre-training. Additionally, experiments using a bi-directional Gated Recurrent Unit (biGRU) were performed to analyse the effect of using a simpler architecture with fewer parameters. For the biGRU experiments, an embedding matrix was learnt from scratch for tokens in the vocabulary of BERT’s tokeniser, mapping each token to a 384-dimensional vector. The size of the hidden layer of the GRU was set to 384 in

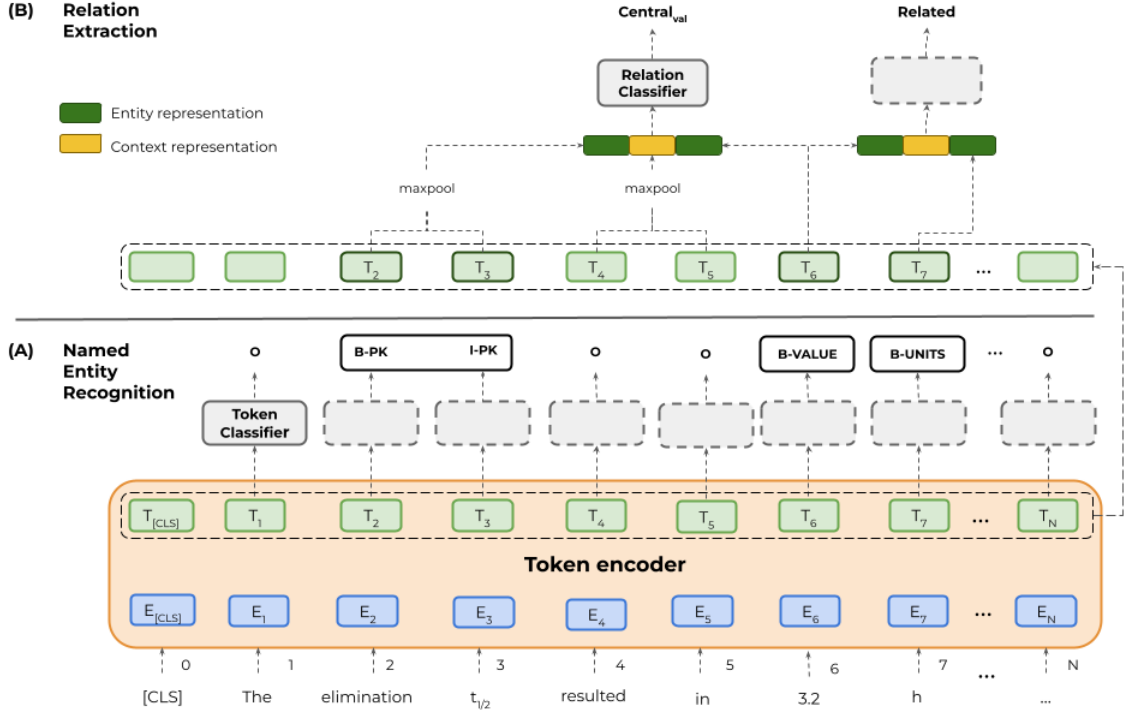


FIGURE 6.5: Illustration of the approach used for joint entity and relation extraction of PK measurements. The model first receives a sequence of token embeddings (blue boxes, E_i) and goes through the encoder layers to generate a sequence of contextual token embeddings (green boxes, T_i) which are shared in both tasks. Then, (A) contextual token embeddings go through the token classifier (feed-forward layer) to output BIO labels that will allow recognising entities. (B) Entities and contexts (span between two entities) are represented by max-pooling their contextual token embeddings. Finally, pairs of entities are concatenated with their context representation and passed through the relation classifier (feed-forward layer).

each direction so that the output representation for each token matched the BERT’s dimension ($d=768$), and a single layer was used.

Named Entity Recognition Analogous to the model presented in chapter 4, NER was treated as a sequential labelling problem. Each output token representation from the encoder (T_i) was classified into one unique BIO class (see section 4.2.1.3 for details) using a feed-forward layer with a sigmoid activation function. The NER loss, \mathcal{L}_{NER} , was computed using the categorical cross-entropy loss between the predicted token-level softmax scores and the actual labels for each BIO class. For a given sentence i with N tokens, the loss was computed as:

$$\mathcal{L}_{NER_i} = - \sum_{n=1}^N \sum_{c=1}^C y_{nc} \log(\hat{y}_{nc}) \quad (6.1)$$

Where C is the list of possible BIO labels and \hat{y}_{nc} is the predicted likelihood that token n belongs to class c .

Candidate entity pairs After NER is performed in a specific sentence, all potential pairs of predicted spans were arranged and filtered before going to the relation classifier. During the filtering phase, those pairs of spans belonging to entity types that had no relation were not considered. In other words, if multiple *PK*, *Value*, and *Units* spans were detected in a specific sentence, all pairs of spans were classified except those between *PK – Units*, *PK – PK* and *Units – Units*, since none of the relations in the PK-REX corpus involved those entity type pairs.

Relation Classification After filtering, each candidate entity pair was classified into one relation class [*Central_{val}*, *Deviation_{val}*, *Relation*, *No Relation*]. Following Taillé et al. [260], the representation of those spans composed by multiple tokens was generated by max-pooling their contextual token embeddings. For instance, in Figure 6.5 B, the contextual embeddings from “*elimination*” and “*t_{1/2}*” (T_2 and T_3) were fused into a single embedding of the same length ($d=768$) by max-pooling across each dimension of the two embeddings. Given the effective results of the max-pooling strategy presented by Eberts and Ulges [238], no other fusion functions were analysed. The input to the relation classifier $x(s_1, s_2)$ was the concatenation of the two span representations $e(s_1)$ and $e(s_2)$ with their context representation $c(s_1, s_2)$:

$$x(s_1, s_2) = [e(s_1); c(s_1, s_2); e(s_2)] \quad (6.2)$$

The context representation for two spans (yellow in Figure 6.5 B) was generated by max-pooling all tokens strictly between them. If there were no tokens present between two spans $c(s_1, s_2) = 0$. Experiments were performed to analyse the effect of using this local context. Relations between entities were symmetric (non-directional) in the PK-REX corpus, and no overlapping spans were annotated. As a consequence, $e(s_1)$ and $e(s_2)$ were arranged according to their relative position in the sentence from left to right. Analogous to the token classifier, a single-feed forward layer was used to classify each candidate span pair. Since only one relation class could be associated between two entities⁵, a softmax operation was used as an activation function. The categorical cross-entropy loss over relation classes was used as the RE loss, \mathcal{L}_{RE} .

Training and optimisation All the parameters from the encoder, the token and the relation classifier were fine-tuned during the training phase. Given sentences with

⁵This was a different setting from most previous work on end-to-end RE where more than one relation can often exist between two entities and sigmoid is used as an activation function [238]

annotated entities and relations, the loss was computed jointly by adding the NER and RE losses:

$$\mathcal{L} = \mathcal{L}_{NER} + \mathcal{L}_{RE} \quad (6.3)$$

Both losses were averaged over each batches' samples. Each batch consisted of B sentences from which samples were drawn for both classifiers:

- For the token classifier, the loss was computed for all tokens in the batch using the BIO labels.
- For the relation classifier, at training time, ground truth (annotated) entities were used to generate candidate pairs. Negative samples (*No Relation* class) were generated with all candidate entity pairs that were not labelled with a relation during the annotation phase. For instance in the sentence:

“The $t_{1/2}$ and CL of amoxicillin were 3 min and 4 mL/min, respectively”

With the following labelled relations: $(t_{1/2}, 3, Central_{val})$, $(CL, 4, Central_{val})$, $(3, \text{min}, Related)$, $(4, \text{mL/min}, Related)$

Negative samples were generated with unconnected entity pairs:

$(CL, 3, No\ Relation)$, $(t_{1/2}, 4, No\ Relation)$, $(4, \text{min}, No\ Relation)$, $(3, \text{mL/min}, No\ Relation)$.

At inference time, only those entities predicted by the NER module were passed to the RE classifier instead of using ground truth entities.

The models were implemented with PyTorch [219] and the Huggingface Transformers library [220]. The models were trained for 50 epochs and evaluated on the development set after each epoch. In each experiment, the state of the model with the highest $Central_{val}$ F_1 score on the development set at a specific epoch was saved. Early experiments suggested the performance on the development set started plateauing after 30 epochs, and most experiments achieved the highest performance short after that time. The Adam Optimizer with a linear weight decay of 0.05 was used, and a dropout probability of 0.1 was applied on all layers. All the experiments were run on a single GPU NVIDIA Titan RTX (24GB). Ten runs were performed for each model or data configuration to compare model architectures, each with a different random seed affecting the initialisation of the NER and RE layers, dropout, and the mini-batch splits. At the end of each run, the state of the model with the best performance on the development set was applied to the test set.

Hyperparameters For all experiments, the maximum sequence length was set to 256 and the batch size to 8. The learning rate for BERT-based models was selected based on the development set performance by performing a grid-search over $\mu = [1e-5, 2e-5, 3e-5, 4e-5, 5e-5]$. The exploration was performed using BioBERT as an encoder, and a single run was performed for each learning rate value, obtaining the best performance on the development set when $\mu=2e-5$. For experiments using a biGRU the learning rate was grid-searched over $\mu = [10e-1, 10e-2, 10e-3, 10e-4, 10e-5]$ resulting in 10e-3 as the optimal value. All the other hyperparameters remained constant to their default values during all the experiments.

Comparison to previous work The architecture presented in this work is inspired by the SpERT model developed by Eberts and Ulges [238]. However, modifications were performed to adapt the architecture for PK measurement extraction. The main differences with SpERT included:

1. Eberts and Ulges [238] used a span-based approach in the NER module, which considers all groups of consecutive tokens as candidate spans (up to a certain length), and classifies each of them into entity types. The approach is often used to deal with overlapping spans. However, Taillé et al. [260] showed that when no overlapping spans were present, there were no benefits of using the span-based classifier over token-level predictions. For this reason, and because it is less computationally intensive, sequential BIO labelling was used in this work.
2. Eberts and Ulges represented entities with an additional width embedding, which encoded the length of a certain entity (i.e. number of tokens). This has been effective in NER models using the span-based approach to recognise such entities better. However, no such benefit has been shown on including width embeddings in the RE layer. For this reason, width embeddings were not used in this work.
3. Most work in end-to-end RE considered predicting the directionality of relations, which was not needed for this task. Therefore, entity pairs were arranged based on their appearance in the original text (from left to right).
4. Instead of a softmax layer in the RE classifier, Eberts and Ulges implemented a sigmoid activation. Since more than one relation can happen between two entities in most RE datasets, the values higher than a certain threshold after the sigmoid activation were considered part of a relation. However, only one label can exist between each entity pair in the PK-REX corpus, which made softmax activation more suitable for this task.

6.2.3 Data augmentation

Understanding the relation between specific parameter types and their most common units of measurement could be particularly important for this task. For instance, Figure 6.6A shows a sentence mentioning three PK parameters followed by their estimated values, which is a frequent pattern to express results across the PK literature. From a model perspective, it can be particularly challenging to understand the syntactic dependencies between PK parameters and their associated values in these sentences. However, it is known by pharmacometricians that *half-life* is always expressed in units of time, *clearance* in units of volume divided by time and often⁶ mass, and that *volume of distribution* is expressed in units of volume and often divided by mass.

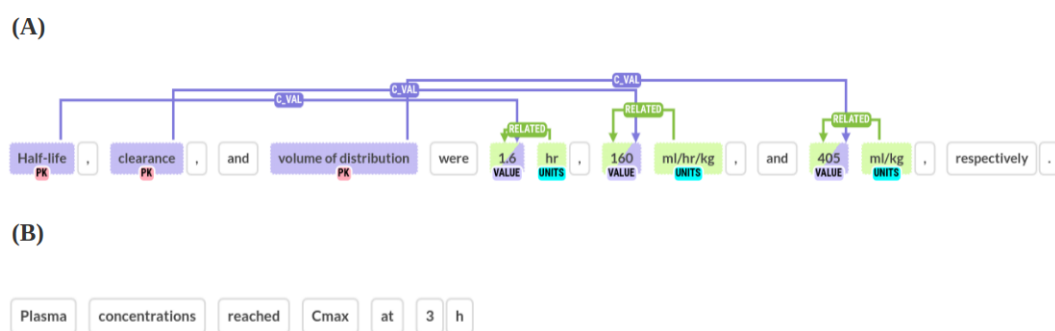


FIGURE 6.6: Illustration of labelled sentences with (A) multiple parameter mentions and their respective values, and (B) a measurement that does not refer to the parameter mentioned.

Similarly, in Figure 6.6 B, without prior knowledge that C_{max} is often expressed in units of concentration, it might not be clear whether the measurement “3h” (which corresponds to the t_{max}) refers to the parameter C_{max} based on syntactic dependencies between entities. Hence, the challenge of predicting potential $Central_{val}$ relations in Figure 6.6 could be simplified with prior knowledge about the relations between specific parameters types and the dimensions⁷ of their units of measurement. Based on this idea, a data augmentation strategy was developed to increase the diversity of training examples.

Data augmentation refers to a set of techniques for enhancing the diversity of training data by producing synthetic instances without explicitly gathering more annotations [266, 267]. Given a sentence with annotated entities and relations, the method aimed to replace mentions of PK , $Units$ and $Compare$ with new ones using a look-up dictionary and replace $Value$ and $Range$ mentions by $\pm 100\%$ of their values. An example of

⁶ Clearance and volume of distribution are often normalised by mean adult body weight (70kg).

⁷ Across this chapter, the term dimensions is used to describe physical quantities that can be measured such as Mass, Volume, Time.

an augmented sentence can be observed in Figure 6.7. For *Value* and *Range* mentions, integers were replaced by integers by rounding the augmented result while decimal values were replaced by synthetic values with either 1, 2, 3 or 4 decimals, which was randomly selected. *Compare* mentions were replaced by randomly selecting⁸ another *Compare* mention annotated in the training set.

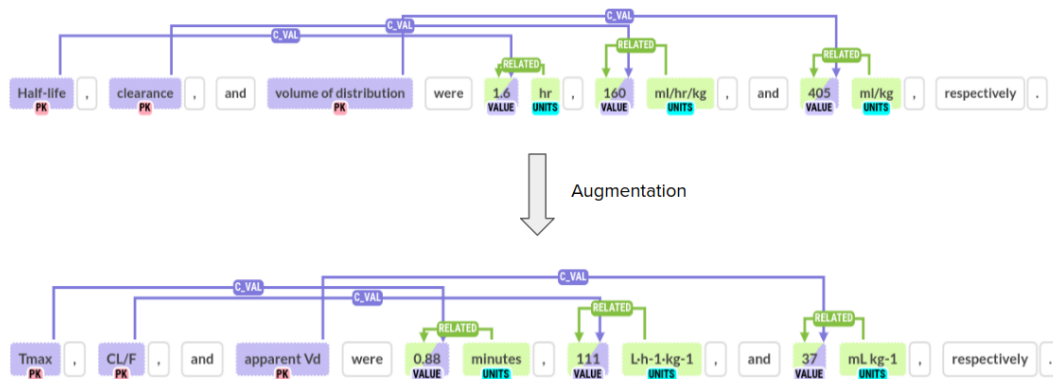


FIGURE 6.7: Example of the augmentation process to generate a synthetic sentence (bottom) given its original one (top).

The look-up dictionaries for *Units* mentions were generated by normalising each mention to a series of tokens and mapping each token to their dimension (Figure 6.8). The following procedure was applied to each annotated *Units* in the training set:

1. Lowercasing all characters except capital M (molar concentrations).
2. Replacing different division and multiplication symbols with “\” and “.”, respectively. In addition, white spaces and consecutive multiplication symbols were replaced by “.”.
3. Mentions were tokenised by the “\” and “.” symbols.
4. Each token was replaced by its standard form using a look-up dictionary of synonyms⁹. For instance, [“microliters”, “microliter”, “micro l”, “microl”, “ μ l”] were all mapped to “ μ l”.
5. Tokens followed by a “-1” or appearing after a back-slash were considered denominator tokens.
6. Parentheses and brackets were removed.
7. Finally, each token was mapped to their dimension using a look-up dictionary (Conversion step in Figure 6.8).

⁸ Without replacement.

⁹ Look-up dictionaries can be found in Appendix D: Unit Dictionaries.

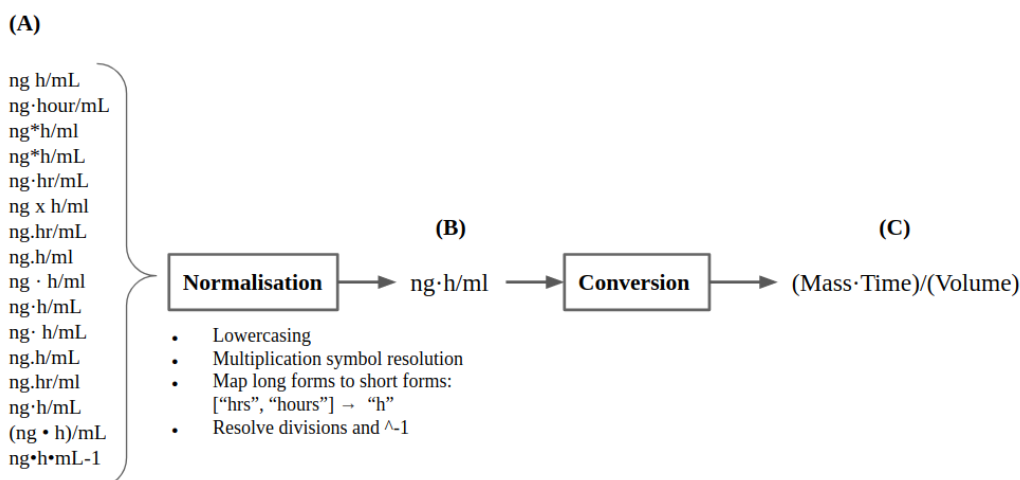


FIGURE 6.8: Illustration of the process implemented to covert *Units* to their dimensions. (A) Input mentions annotated as *Units* go through a series of rule-based steps to be converted into a (B) Normalised form. Finally, normalised forms are tokenised based on multiplication and division symbols, and each token is mapped to its dimension (C).

Then, given a labelled *Units* mention, it was standardised to its dimensions, and a new instance was generated by randomly selecting another *Units* mention from the training set that had the same dimensions. If standardisation was not possible, the original mention was used in the augmented sentence. *PK* mentions were replaced by looking at the dimension of the *Units* related to their central values. Hence, each *PK* mention in the training set was associated with a dimensions string (including *Unitless* if a central value had no units). Then, a labelled *PK* mention was replaced by randomly selecting another mention from the training set with the same dimensions. This process always replaced *PK* and *Units* mentions with training set mentions that had the same dimensions. As a result, parameter types with the same dimensions (e.g. *tmax* and $t_{1/2}$) might be replaced by each other. The augmentation was only performed using the training set data, and augmented sentences were appended to the original ones. Note that those sentences without entities and relations were not augmented. Multiple experiments were performed, with either one, two or three augmented sentences for each original one.

Overall, this augmentation process had two objectives: (1) increase the diversity of *PK*, and *Units* mentions and (2) enhance the model understanding of the relation between certain *PK* parameters and their most common dimensions by preserving these relations in the augmented sentences.

6.3 Results and Discussion

6.3.1 Corpus

Statistics The statistics of the resulting annotated dataset, PK-REX, are presented in Table 6.1. A total of 3,600 sentences were annotated, from which 56.42% contained annotated entities and relations. Sentences were almost evenly sampled from full-text and abstract sections. A total of 13,404 entity mentions were annotated. The entities with the most mentions were Values, Units and PK parameters. This result was expected since most central and deviation measurements involved those entities. Range and Compare mentions were much less frequent, indicating that it was more common for authors to report central measurements of PK parameters in the form of single value estimates instead of ranges. A total of 12,411 relations were annotated, with most of them coming from the *Related* and *Central_{val}* classes. This result indicates that authors tend to report units together with central measurements. More importantly, the fact that the number of annotated *Central_{val}* relations was over x2.5 times the number of *Deviation_{val}* indicates that measures of deviation are not often reported along with central measures of PK parameters (only in 35.8% of cases). Comparing the number of sentences with relations ($\approx 2,031$) with the total number of *Central_{val}* relations in the dataset (4,677) indicates that, on average, those sentences with measurements reported more than a single central measurement, ≈ 2.3 . The average distance between entities in the *Central_{val}*, *Deviation_{val}* and *Related* relations was 15.05, 4.34 and 5.58 sub-word BERT tokens, respectively. Overall, the dataset covered various entity mentions, surface forms and relations. However, it is noteworthy the low number of Range and Compare mentions in all datasets, which might introduce noise in the training and evaluation of NER and RE models.

TABLE 6.1: Corpus statistics summarising the sentences, entities and relations in the dataset stratified by the training, development and test sets.

		Training	Development	Test	Total
Sentences	Amount #	2100	500	1000	3600
	with relations (%)	57.05	53.00	56.80	56.42 [†]
	from full-text (%)	48.71	50.00	50.30	49.33 [†]
Entities	PK	1890	394	856	3140
	Units	2286	474	1056	3816
	Value	3524	702	1557	5783
	Range	314	74	174	562
	Compare	51	18	34	103
Relations	<i>Central_{val}</i>	2794	571	1312	4677
	<i>Deviation_{val}</i>	1049	207	419	1675
	<i>Related</i>	3643	764	1652	6059

[†] Weighted average across datasets.

Annotation quality and sources of disagreement As previously mentioned, the annotation process was performed in batches of 200 sentences and involved (1) annotation of entities and relations by a PK expert, (2) review by another expert and (3) final check focusing on span boundary consistency. Annotators could leave comments for each annotated or reviewed sentence. At the end of each batch, comments were reviewed to resolve disagreements, update guidelines, and provide feedback. This iterative procedure significantly improved the consistency of labels after each batch between the first and second steps. For instance, significant variation was initially observed regarding the inclusion of some PK parameters and their span boundaries, with 28% of sentences being corrected after stage 1 in the first batch. However, as feedback was provided after each annotation batch, the number of sentences that required correction after stage one was reduced to 6% in the last batch. In the initial phases of the annotation process, many sentences reporting % or fold increase of some parameters with respect to baseline values were observed, which generated confusion between annotators and required guideline updates.

In addition to the three passes over all sentences of the PK-REX corpus, 200 instances (the last batch of the test set) were independently re-annotated from scratch by seven annotators. Then, the inter-annotator agreement was examined using the pair-wise F_1 score on those sentences and the mean was computed across each pair of annotators. The average micro and macro- F_1 scores for NER were 88.74% and 92.36%, respectively, exhibiting high agreement on entity surfaces on the first annotation phase. For RE, the average pair-wise scores were 93.02%, 94.47% and 83.2% for *Related*, *Deviation_{val}* and *Central_{val}*, respectively. These results indicate high agreement between annotators when identifying deviation values and units regarding PK measurements. However, a lower agreement was obtained between central values and their PK parameter mentions. After frequent conversations with annotators, it was observed that most of these inconsistencies were caused by disagreement on the span boundaries and the inclusion of some PK parameters. Nonetheless, the annotation consistency of PK parameters was improved in steps 2 and 3 of the annotation process, where checks and standardisation of span boundaries were performed.

Dataset limitations The main limitation of this dataset is its potential bias in selecting candidate sentences. The sampled sentences went through two filtering stages that involved model predictions: (1) selection of PK relevant documents identified by the document classifier and (2) selection of sentences that at least had one PK entity recognised by the model developed in section 4. As a result, if the document classifier missed specific types of documents, these would not appear on this dataset. Using the trained PK NER model for filtering instances with PK parameters mentioned might exclude sentences where the NER model missed a single PK mention. However, in most cases, the optimal NER model detected at least a partial match for most PK parameters

and only in less than 8% of cases (see table 4.4) the model missed the complete mention. Furthermore, if a specific sentence mentioned more than one parameter and only one match (partial or not) was detected by the NER model, the sentence was included in the candidate pool, and these incorrect predictions were later corrected during the annotation process. This filtering resulted in over 50% of sentences in the candidate pool reporting PK estimations (see table 6.1 for details), which provided a rich corpus of PK parameter mentions, measurements, units and relation instances. Overall, it is important to consider that training RE models on this dataset and directly applying them to sentences in the literature without additional filtering might result in the extraction of non-PK measurements due to the filtering approach performed in the sampling stage. For this reason, when deploying systems in production, it will be important to combine models trained on this dataset with filtering approaches to discard irrelevant measurements (e.g. pre-tagging PK parameters or posterior EL of PK mentions recognised with RE models). Finally, previous studies have shown significant benefits on dealing with document-level relation extraction (as opposed to sentence-based) [268, 269]. Hence, future work might benefit from annotation of entire documents.

6.3.2 Multi-Task solution

In this section, the effect of using a multi-task learning approach (MT) that jointly optimised the model to perform NER and RE was compared against a model only optimising for NER (no-MT). For this experiment, BioBERT was used as an encoder in both cases. The MT architecture saved the model with the best $Central_{val}$ $F1$ on the development set, while micro-averaged $F1$ was used as a metric to select the best model for the no-MT experiment. Table 6.2 shows the NER performance on the test set for each entity type and the macro and micro-averaged $F1$ scores after ten runs of each experiment. First, it can be observed that in both architectures, the performance was particularly high for *Value* and *Units* entities. However, entities with fewer mentions on the training set, such as *Compare* or *Range*, had lower $F1$ scores. The performance on *PK* parameters is highly similar to that in chapter 4 (see table 4.4). Although the *PK* entity refers to the same concept as in the NER chapter (i.e. PK parameters), only those parameter mentions that had an associated measurement were labelled in the PK-REX corpus, adding a layer of complexity in PK NER.

Higher performance was obtained when using the MT architecture for all entities in the PK-REX corpus. Although the performance gain was relatively low ($\approx +\Delta F1$ 0.5%), the consistency of this gain across all entity types suggests that having the RE objective combined with NER helped the model perform better on NER. As reported in previous work [270–272], if two tasks are related, sharing the encoder parameters can have a regularisation effect, enhancing the generation of embeddings that are not too specific to a certain task and allowing the model to transfer the knowledge learnt in one task

to another. In PK-REX, for a mention to be labelled, it required that the mention was part of a PK estimation (i.e. involved in a relation) promoting the input embeddings of the NER layer to encode some information about the relations in a sentence. Hence, it is likely that RE provided some additional information to generate better embeddings for NER. Additionally, the performance of relation extraction is directly dependent on the successful prediction of entity types and exact boundaries. The effect of regularisation becomes more clear for entities with fewer mentions on the training set (i.e. *Range* and *Compare*), which had a larger performance gain when using the MT architecture. Although the performance gain of the MT architecture was small, such an approach also helped reduce the number of parameters required to model the task. For instance, an end-to-end solution that does not share encoder representations might require two encoders for each task, increasing the overall number of parameters. Overall, these results indicate that sharing token representations and optimising a single loss for NER and RE is beneficial for extracting PK measurements from the scientific literature compared to treating both tasks independently.

TABLE 6.2: Named Entity Recognition results on the test set for the model using multi-task learning (MT), NER + RE, against a model only optimising for NER (no-MT). The metrics reported consider strict matching over entity mentions. Results are displayed as the median over ten runs with their interquartile variance in subscript.

Entity	Precision		Recall		F1	
	MT	no-MT	MT	no-MT	MT	no-MT
PK	90.82 _{4.02}	89.98 _{3.86}	90.57 _{3.76}	90.09 _{3.05}	90.39 _{2.1}	90.02 _{1.72}
Units	95.49 _{1.87}	95.79 _{1.66}	96.17 _{2.07}	95.69 _{3.85}	95.65 _{0.68}	95.56 _{1.52}
Value	94.83 _{2.78}	94.96 _{2.87}	96.18 _{3.17}	95.21 _{5.94}	95.54 _{2.53}	95.04 _{2.02}
Range	93.49 _{4.9}	93.28 _{6.24}	90.26 _{8.22}	87.39 _{10.33}	91.66 _{4.41}	90.4 _{3.71}
Compare	88.23 _{6.81}	88.23 _{16.99}	66.67 _{9.09}	68.18 _{11.44}	76.53 _{5.82}	75.64 _{8.12}
Micro-average					94.03 _{1.63}	93.69 _{1.60}
Macro-average					90.02 _{2.23}	89.56 _{2.45}

Finally, the performance of the MT solution on the RE task is displayed in Table 6.3. The RE results indicated that deviation measurements and units were successfully linked in most cases. It is worth noting that if values and units are detected correctly, predicting their relation often requires little context (i.e. relatively short distance between values and units). Additionally, the context separating units from their correct value often had no other units between both entities, which simplified the extraction. Similarly, the context strictly between a central value and a deviation value often did not contain any other value entity. Therefore, given the high NER performance of *Value* and *Units*, very few errors were observed for *Deviation_{val}* and *Related* relations. The *Central_{val}* relation exhibited relatively high performance given the difficulty for annotators to correctly identify the span boundaries of PK parameters and their relation to central values, suggesting consistency in the annotated dataset and effective modelling of the problem end-to-end. Examination of the model predictions (see more details in section 6.3.5)

showed that errors on the $Central_{val}$ predictions were mostly caused by incorrect NER predictions and sentences mentioning multiple parameters followed by their respective values. Nonetheless, it was observed that some of the incorrect predictions of PK entities successfully detected a partial match of PK parameters. This result indicates that the $Central_{val}$ performance reported in Table 6.3 could be treated as a lower bound for the extraction of PK measurements since partial matches of PK parameters might be enough in some applications (e.g. if EL is performed correctly). Finally, it was noted that the F_1 scores reported in Table 6.3 were close or superior to the pair-wise inter-annotator performance observed in the initial annotation rounds: 93.02% vs 93.66% for $Related$, 94.47% vs 93.53% for $Deviation_{val}$, 83.2% vs 86.1% for $Central_{val}$, for inter-annotator and MT model cases, respectively. This results suggests that posterior reviews and standardisation of span boundaries significantly improved the consistency of the dataset, and that the model developed has competitive performance compared to the expected agreement between pharmacometricians.

TABLE 6.3: End-to-end relation extraction results on the test set for the MT model configuration. Results are displayed as the median over ten runs with their interquartile variance in subscript.

Relation	P	R	F_1
$Central_{val}$	85.77 _{5.04}	85.46 _{5.07}	86.1 _{3.49}
$Deviation_{val}$	92.33 _{1.9}	94.39 _{6.27}	93.53 _{3.01}
$Related$	93.83 _{1.69}	94.08 _{2.51}	93.66 _{1.52}

6.3.3 Effect of encoder and context

In the following experiment, the effect of using different encoders to generate token representations was studied. To analyse the effect of domain-specific pre-training in the encoder, the BioBERT model was replaced with BERT_{BASE}, which was pre-trained on general-domain English text. As it can be observed in Table 6.4, there was a significant benefit of pre-training in biomedical text, with BioBERT exhibiting over 3% gains in all metrics in comparison to BERT_{BASE}. The largest gain ($\approx \Delta 6\%$) was observed in the $Central_{val}$ relation, indicating that pre-training on biomedical text highly improved not only PK NER but also the understanding between parameter mentions and their measurements. These results are in-line with previous findings from Wadden et al. [273] and Eberts and Ulges [238]. It is worth mentioning that previous work on end-to-end relation extraction showed improvements between 1.1-4.4% on the SciERC and GENIA datasets with in-domain pre-training [238, 273]. However, 5.9% improvement was obtained in this task for $Central_{val}$, suggesting that in-domain pre-training is particularly useful for this task. Hence, it is likely that further pre-training on PK literature helps the model performance, and it might be a promising area for future work on this dataset.

For tasks requiring relatively local context and where limited data is available, Transformer models might contain too many parameters and overfit to the training data, and architectures with fewer parameters could be better suited, obtaining similar representations with fewer parameters [274]. For instance, Lai et al. [274] showed how carefully designed CNNs could highly reduce the number of trainable parameters and achieve better performance on biomedical EL than BERT-based models. BioBERT was replaced by a biGRU with no pre-training to study a simpler encoder, aiming to compare the effect of using a model with fewer parameters (x50 fewer parameters were trained in the biGRU) and a simpler architecture. Nonetheless, the biGRU exhibited significantly lower performance than BERT models, highlighting the benefits of pre-training and fine-tuning with Transformer models for both NER and RE tasks compared to simpler recurrent models. These results are in-line with Taillé et al. [260] who compared the Transformer architectures to a biLSTM in end-to-end relation extraction. However, the biGRU implemented in this study was relatively simple, and other designs of contextual representations and pre-trained token vectors might yield better results on this task.

TABLE 6.4: Results on the test set when using different encoder models. Results are displayed as the median over ten runs with their interquartile variance in subscript. NER metrics are the micro- and macro- averaged F_1 scores over all entities, and RE metrics are the F_1 scores for each relation class.

Encoder	NER		RE		
	macro- F_1	micro- F_1	<i>Related</i>	<i>Deviation_{val}</i>	<i>Central_{val}</i>
BiGRU	75.4 _{3.17}	84.12 _{2.13}	83.58 _{2.61}	85.7 _{4.5}	72.92 _{5.78}
BERT _{BASE}	85.82 _{4.07}	90.81 _{1.77}	89.44 _{1.69}	90.27 _{2.2}	80.16 _{4.14}
BioBERT	90.02 _{2.23}	94.03 _{1.63}	93.66 _{1.52}	93.53 _{3.01}	86.1 _{3.49}

The effect of removing the local context between entities was studied. For this, the input to the RE layer was simplified to the entity embeddings. In other words, the yellow vector from Figure 6.5 B was removed. Table 6.5 shows the results of this experiment. Surprisingly, it was observed that the local context improved not only RE but also NER. Both micro and macro- F_1 scores were slightly improved, suggesting that explicitly encoding local context between entities in RE layers can also help recognise entities better. One potential explanation behind this result is that, when predicting *Central_{val}* between *PKs* and *Value*, it might be important to detect whether other parameters and values have been mentioned between the parameter and value being predicted. For instance, when predicting a relation between “t1/2” and “0.5” in the following sentence: “The t1/2, and tmax of midazolam were 0.2 and 0.5 h, respectively”, it would be crucial to understand that another PK parameter and value were mentioned in the local context. Hence, max-pooling those tokens in the local context could promote the network to generate embeddings that better differentiate PK parameters and Values from other tokens, which, in turn, could benefit the recognition of those entities. However, it is important to note that the median improvements on NER were below 0.5, and this hypothesis would need to be further analysed to draw conclusions.

For relation extraction, local context seemed to provide a significant improvement for all relation types, and especially for the *Central_{val}*. This result suggests that entity embeddings might capture local information around the entity mentioned while failing to incorporate longer-range dependencies. The results obtained in this experiment are in-line with Eberts and Ulges [238]. Although recurrent and Transformer models have improved the detection of long-range dependencies in sequential inputs, the noise introduced with long context still represents a challenge in relation extraction [238, 259]. Using this local context, the model can focus on those tokens that might be more informative about the dependencies between both entities. Nonetheless, future studies might benefit from further exploring different contextual representations for RE of PK measurements.

TABLE 6.5: Results on the test set when using different representations as input to the relation classifier. Local context is the max pooling of all tokens strictly between two entities. No context only used the concatenation of each entity representation in a specific relation. Results are displayed as the median over ten runs with their interquartile variance in subscript. NER metrics are the micro- and macro- averaged F_1 scores over all entities and RE metrics are the F_1 scores for each relation class.

RE layer representaiton	NER		RE		
	macro- F_1	micro- F_1	<i>Related</i>	<i>Deviation_{val}</i>	<i>Central_{val}</i>
Local context	90.02 _{2.23}	94.03 _{1.63}	93.66 _{1.52}	93.53 _{3.01}	86.1 _{3.49}
No context (E1E2)	89.47 _{2.16}	93.69 _{1.0}	91.61 _{1.84}	90.52 _{4.44}	81.04 _{2.96}

6.3.4 Augmentation

In the last experiment, heuristic data augmentation was performed to enhance the model understanding of the relation between specific parameter mentions and their most common dimensions and increase the diversity of units and PK mention synonyms. Given an input sentence from the training set, several replicates were created using the augmentation dictionaries automatically generated from the training data and a set of rules to generate new value entities. Three independent analyses were performed, using the original sentence in the training set to generate one (x2)¹⁰, two (x3), or three (x4) augmented sentences. Table 6.6 and Figures 6.9, 6.10, 6.11 show the results of this experiment.

Almost no improvement was observed in macro or micro-averaged F1 scores for NER. In fact, Figure 6.10 shows a slight decrease in the macro-averaged F1 scores when increasing the number of augmented sentences. From Figure 6.11, it can be observed that this slight decrease is a consequence of lower performance on the *Compare* entity. Since very few mentions of *Compare* terms was present on the training set, the augmentation approach always replaced *Compare* mentions with similar terms, biasing the model towards those specific surface forms but not aiding the generalisation of the *Compare* entity.

¹⁰Note that x2 refers to the original + the augmented sentence.

However, the RE performance was more influenced by the augmentation approaches. First, in Figure 6.11 the performance on the $Central_{val}$ relation seems to increase linearly as more augmented sentences were generated, up to x3. This result indicates that the augmentation approach used in this study improved the detection of central measurements of PK parameters. However, the performance drops again when generating more than two augmented replicates of the original sentence. This drop is likely to result from the model overfitting to irrelevant context. One of the main drawbacks of the augmentation approach developed in this study is that only NER mentions were augmented while keeping the remaining context static. Hence, this approach might bias the model towards detecting incorrect dependencies between the presence of irrelevant contextual tokens and the appearance of $Central_{val}$ relations. Finally, when looking at Figure 6.11, it can be observed that the NER performance of PK terms was improved on the x3 experiment. This result suggests that the improved performance on $Central_{val}$ is partially influenced by a better recognition of PK parameters.

Overall, due to the inconsistent results across NER and relation types and the similar performance observed when performing augmentation, it cannot be concluded that the augmentation approach implemented in this study provided a significant benefit for this task. The diversity of PK surface forms in the training set was relatively high, and the improvement observed in $Central_{val}$ relations might not be observed in situations where the diversity of entity surface forms is low. The overfitting observed in the x4 experiment suggests a clear limitation on this augmentation approach. Future studies might benefit from approaches that deal with the context overfitting problem [266]. One approach could also be to augment non-entity spans such as drug names or administration routes mentioned in the sentence.

Finally, a growing number of studies make use of distant supervision to improve the performance of RE systems [275], which was not explored in this chapter. Future work on this dataset might benefit from exploiting priors of known PK parameters stored in PK databases such as PKDB or DrugBank.

TABLE 6.6: Results on the test set when performing augmentation on the training set. Results are displayed as the median over ten runs with their interquartile variance in subscript. NER metrics are the micro- and macro- averaged F_1 scores over all entities, and RE metrics are the F_1 scores for each relation class.

Strategy	NER		RE		
	macro- F_1	micro- F_1	<i>Related</i>	<i>Deviation_{val}</i>	<i>Central_{val}</i>
No augmentation	90.02 _{2.23}	94.03 _{1.63}	93.66 _{1.52}	93.53 _{3.01}	86.1 _{3.49}
Augmentation x2	89.76 _{2.1}	93.74 _{1.38}	93.11 _{3.84}	92.71 _{3.71}	86.7 _{3.13}
Augmentation x3	89.77 _{1.87}	94.12 _{1.36}	93.23 _{2.58}	93.66 _{2.34}	87.4 _{2.92}
Augmentation x4	89.46 _{2.69}	93.74 _{1.09}	93.7 _{2.55}	93.68 _{3.0}	86.72 _{1.77}

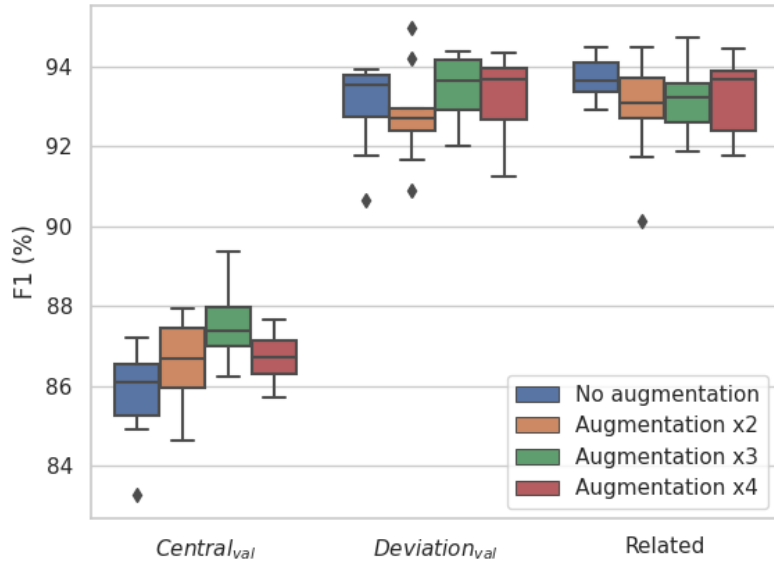


FIGURE 6.9: Box-plot showing the F_1 scores on the test set for each relation class when performing augmentation of sentences in the training set. Experiments were performed augmenting each annotated sentence once (x2), twice (x3) or three times (x4) or not performing augmentation at all. Ten runs with different seeds were performed for each experiment.

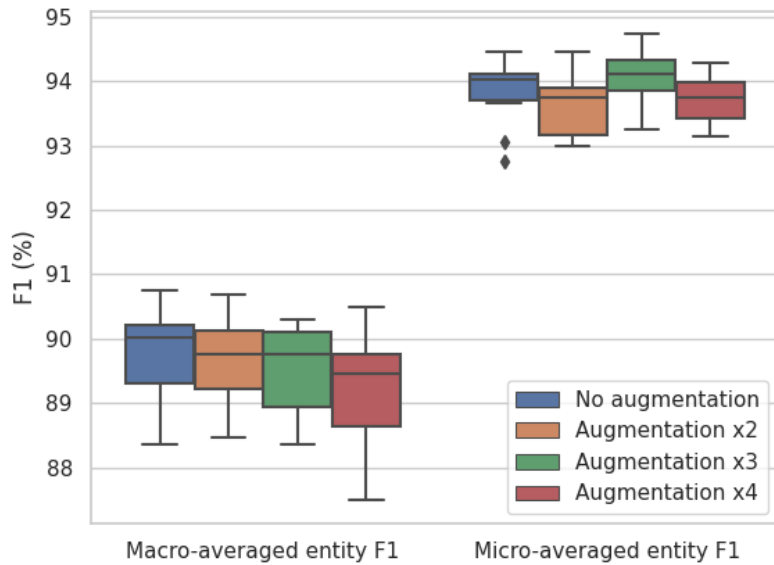


FIGURE 6.10: Box-plot showing the NER micro- and macro-averaged F_1 scores when performing augmentation of sentences in the training set. Experiments were performed augmenting each annotated sentence once (x2), twice (x3) or three times (x4). Ten runs with different seeds were performed for each experiment.

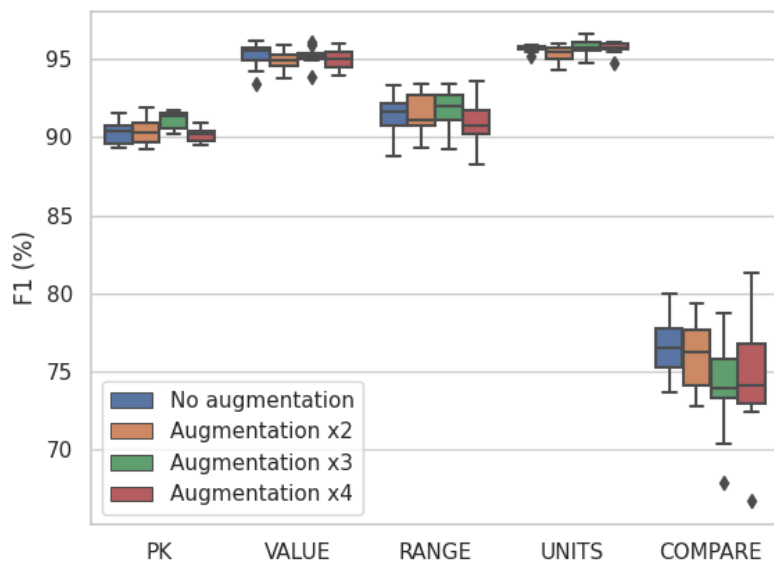


FIGURE 6.11: Box-plot showing the F_1 scores for each entity type when performing augmentation of sentences in the training set. Experiments were performed augmenting each annotated sentence once (x2), twice (x3) or three times (x4). Ten runs with different seeds were performed for each experiment.

6.3.5 Error analysis

Causes of incorrect predictions were first inspected for the best¹¹ base model (No Augmentation). The examination was done qualitatively by comparing model predictions against annotations in the test set. A total of 12% of annotated sentences in the test set presented some sort of misalignment (either at the NER or RE level) with the base model predictions. From those sentences with incorrect predictions, 79% had some NER span incorrectly predicted, and 21% were exclusively related to incorrect prediction of relations between entities. Hence, the main limitation behind optimal PK measurement extraction seemed to be span boundary detection.

NER analysis For *Units*, *Value*, *Range* and *Compare* entities, it was observed that partial matches did not cause most NER errors. Instead, they were often caused by either missing all the entities relating to a measurement (including *PK*) or predicting all the entities relating to a measurement that was not a PK parameter (e.g. doses or concentrations). Over-predicting values seemed to occur much more often than missing annotated measurements. Two of those examples can be observed in Figure 6.12. For *PK* mentions, partial matches were a common cause of the model error. However, as it can be observed in the example from Figure 6.12, the values and units related to

¹¹Best performing model in the development set.

that parameter were extracted correctly. In those cases, extracting a partial match is likely to be useful for different use-cases, especially when used in combination with EL systems. Finally, this comparison check also allowed for detecting wrongly annotated sentences. In the last example from Figure 6.12 it can be observed how the model did not predict any entity since the estimated value does not refer to the parameter “V1” but a reduction from a baseline value. However, this was incorrectly annotated as a measurement.

Over-predicted

Annotation: Over the 15-week trial, pUA in PR averaged 1.0 ± 0.4 mg/dL; $T_{1/2}$ for pUox was approximately 13 days

Prediction: Over the 15-week trial, pUA in PR averaged 1.0 ± 0.4 mg/dL; $T_{1/2}$ for pUox was approximately 13 days

Missed

Annotation: After IA bolus, IFN is distributed from the blood with a $t_{1/2}$ below, 2 hours

Prediction: After IA bolus, IFN is distributed from the blood with a $t_{1/2}$ below, 2 hours

Partial match

Annotation: The HDV half-life in blood was 1.87 days

Prediction: The HDV half-life in blood was 1.87 days

Wrong annotation

Annotation: During haemodialysis, the central volume V1 was estimated to be reduced by 0.198 L/h.

Prediction: During haemodialysis, the central volume V1 was estimated to be reduced by 0.198 L/h.

FIGURE 6.12: Examples from the test set of main causes of model error at the NER level. One example is presented for over-predicted, missed, partial match, and wrongly annotated sentences.

RE analysis From those sentences with entities correctly predicted but with RE errors, 44% mentioned consecutive *PK* parameters followed by their respective values (e.g. Figure 6.13). Since the architecture implemented max-pools tokens between two entities, this might not be representative enough to account for the number of parameters and values mentioned between a specific *PK* and *Value*. Instead, an experiment was performed to examine whether a simple rule could fix those cases. For this, when multiple parameter values were mentioned without any other entity between them, if a multiple number of central measurements (respect to the number of *PK* mentions) was reported, the first *PK* mention was related to the first *Value/Range* mention (excluding deviations), the second *PK* mention to the second *Value/Range*, and so on. With this approach, the performance of the *Central_{val}* relation on the test set was improved from 86.7% to 86.82%. It is worth noting that this rule-based approach failed in some cases where consecutive results were reported for each parameter referring to different experimental designs, as shown in Figure 6.13. When using the augmentation x3, the median *Central_{val}* increased 0.7%, which might be the result of better predicting those

cases and the *PK* mentions. After inspection, only 35% of incorrect RE sentences from the augmentation x3 model predictions followed the consecutive pattern, suggesting that the augmentation approach helps reduce this kind of error.

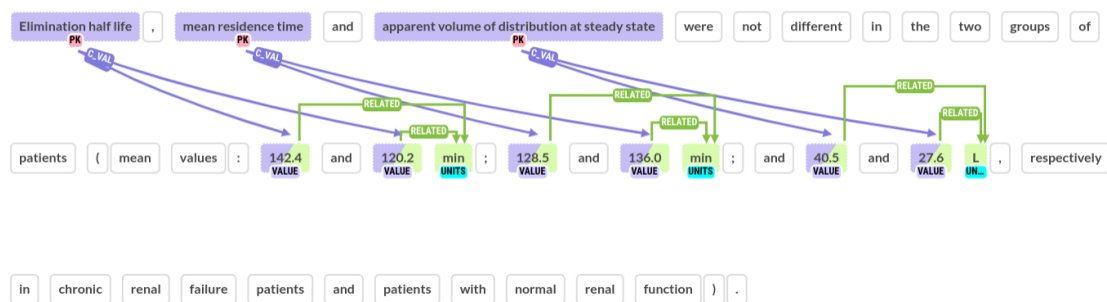


FIGURE 6.13: Example of a complex annotated case where the rule-based algorithm to associate consecutive *PK* parameters and their respective *Values* would not work.

Another common cause of *Central_{val}* mispredictions were in estimations expressed with the pattern “< *Value* > ± < *Value* >< *Units* >< *Parameter* >”. Since the parameter is, in most cases, mentioned on the left-hand side of the measurement, the model often mispredicted these cases, associating a *Central_{val}* with the deviation measurement. Similarly, units mentioned on the left-hand side of the measurement were often not related to the correct values. Finally, the most common source of *Deviation_{val}* errors was observed in cases following the pattern “< *PK* >< *Value* > (< *Range* >) < *Units* >”. In these cases, during the annotation process, the *Range* was considered the deviation of the *Value* to not extract repeated measurements in the PK database. However, in most cases, the model did not predict a *Deviation_{val}* between the *Value* and *Range* but instead a *Central_{val}* between the *PK* and *Range*.

Overall, from the error inspection, it was clear that most measurements were extracted correctly. However, some *PK* parameters were still missed with no apparent explanation (e.g. *t_{max}*), and concentration and doses were often extracted. From a NER perspective, future work might benefit from strategies focusing on improving the detection of *PK* parameter mentions. From the RE point of view, most errors seemed to appear in sentences mentioning multiple parameters and, subsequently, their respective values. The augmentation approach reduced the number of mispredicted sentences that followed this pattern. However, a large proportion of mispredictions still appeared in those cases. Hence, better encoding of the context between *PK* and *Value/Ranges* could benefit the overall model performance. For instance, explicitly including the embeddings of predicted *Units* for a particular *Value/Ranges* when predicting *Central_{val}* relations could help the model understand the dependencies between parameters and units better.

6.4 Conclusions and Future work

This chapter presented the first resource to train and evaluate models for extracting PK measurements from the scientific literature, named PK-REX. To fulfil objective O₄₋₁, a framework was designed to annotate sentences and extract PK estimations in a structured format based on PK entities and their relations. Annotations were performed by expert pharmacometricians and exhibited high consistency on entities and their relations, which addressed objective O₄₋₂.

To address objective O₄₋₃, a multi-task learning approach was implemented to jointly predict entities and their relations, achieving F1 scores over 85% on end-to-end relation extraction (RE). Sharing token representations for NER and RE helped the model performance on NER, exhibiting the benefits of jointly modelling both tasks, which addressed objective O₄₋₄. Across the methods studied, pre-training transformer models in biomedical text largely outperformed alternative approaches to encode sentence tokens. Additionally, using explicit representations of the context strictly between two entities was crucial to obtain optimal performance on RE.

A heuristic data augmentation approach was developed to tackle objective O₄₋₅, increasing the diversity of training samples based on the relationship between specific parameter types and their units. The augmentation approach seemed to improve the recognition of PK parameter mentions and their relation to central measurements but this improvement was not consistent in other relation types. Additionally, the model seemed to overfit the training data when more than two augmented sentences were generated for each original one. Overall, it could not be concluded that the augmentation approach provided a benefit for this task.

Finally, an inspection of model errors was performed to address objective O₄₋₆. The analyses showed that most limitations of the model performance were caused by NER of PK parameters and incorrect *Central_{val}* predictions when multiple PK parameters were mentioned following their respective values.

Future work The work presented in this chapter can be advanced by exploring its feasibility in constructing ADME datasets for preclinical drug development. For example, a database of PK measurements could automatically be generated using the models presented in this chapter, and different strategies could be studied to associate each measurement to the correct chemical compound and study population. Further annotations would also help improve the model performance and its robust evaluation. On this end, annotating the relations between PK measurements and their chemical compounds and study populations would make the extracted PK data more usable for constructing ADME datasets. Although the models developed have exhibited the ability to automatically extract useful features for RE, other contextual representations could be studied,

or manually-engineered syntactic and semantic features could be added to enhance the model's understanding of the relations between specific parameters and their estimated values.

Chapter 7

Pharmacokinetics database

The previous chapters in this thesis have tackled several tasks for improving text mining in the pharmacokinetics (PK) domain, from detecting relevant documents to extracting parameter estimates. However, for researchers in PK to exploit these resources effectively, it is essential to evaluate the potential applications and limitations of using text mining in their use cases. In this chapter the Q_5 of this thesis was addressed: *How can text mining approaches accelerate the curation of ADME datasets used by pharmacometricians?*. Therefore, this chapter discusses how the resources developed across this thesis could be used for pharmacometricians and highlights potential limitations and areas for future work. Additionally, the chapter presents an overall discussion of the methods used across this thesis when applied to extract PK estimates. The following objectives were established to answer Q_5 :

- O₅₋₁ Develop a database of PK estimates automatically extracted from the literature by using the models developed in previous chapters.
- O₅₋₂ Evaluate the performance of the extraction pipeline and the quality of the database.
- O₅₋₃ Use the database to perform a case study showcasing the extraction of multiple parameters and drugs of interest.
- O₅₋₄ Assess whether text mining approaches can accelerate the retrieval of publications reporting novel PK estimates for the different drugs tackled in the case study.
- O₅₋₅ Identify and discuss the key applications and limitations of using text mining approaches to accelerate the curation of ADME datasets.

To do so, two analyses were performed:

1. **Database** PK estimates were automatically extracted from a large corpus of publications generating a database of measurements. Then, analyses were performed to determine the most common parameters and units reported, and the reliability of the information reported in the database was assessed. Finally, a potential interface was presented to stratify the extracted estimates by drugs, parameter types, units, and contextual information.

2. **Case study** The models developed were used to accelerate a PK meta-analysis previously published and determine where could text mining have been most helpful in the extraction and comparison of PK estimates across drugs and study populations.

7.1 Database

Comprehensive databases of PK parameter estimates are critical for preclinical predictions of candidate drugs, perform PK meta-analyses and provide informative priors for pharmacometric model building [10, 11, 255]. For this reason, one of the most promising applications of text mining is in the curation process of these databases. In this section, the models developed in this thesis were sequentially applied to extract PK estimates and automatically generate a database of measurements to explore and evaluate the extraction of numerical data with text mining. Then, the quality of the extracted data was assessed, and summary statistics were reported and discussed. Finally, a potential interface to filter values by drugs, parameter types, units and other fields was presented.

7.1.1 Construction

For this study, the extraction of PK parameters was narrowed down to abstracts from scientific publications. Despite the wealth of PK information reported in full-text articles or clinical trial reports, these documents are often not available open access and might be only published in PDF format. Hence, to allow the reproduction of the methods presented, the extraction was limited to scientific abstracts published in PubMed, and the extraction and use of PK estimates in full-text documents were left for future studies.

The following steps were computed sequentially to construct a database of PK estimates from scientific abstracts:

1. **Initial Search and Parsing** The PubMed search “pharmacokinetics” was performed in January 2021, which returned 583,961 publications. The XML of these publications was downloaded from PubMed and parsed with PubMed Parser [197]. The metadata of each abstract was stored in JSON format, and the biomedical Stanza Python library [150] was used to split the abstract text into sentences.
2. **Document Retrieval** The optimal pipeline from chapter 3 was used to identify those publications that were likely to report PK parameters, returning 120,913 articles classified as “Relevant”.

3. **Relation Extraction** From those relevant publications, the optimal pipeline from chapter 6¹ was applied to extract PK estimates through relation extraction. However, like in the sampling performed for relation extraction (see section 6.2.1.2 for details), the model was only applied to sentences reporting at least one numerical value, which highly reduced the inference time.
4. **Entity Linking** Those PK parameter mentions detected by the RE model were linked to the PK knowledge base using the optimal model presented in chapter 5. Each numerical estimate in the final database was related to a specific PK parameter type with this approach.
5. **Formatting and unit standardisation** Each *Central_{val}* relation predicted by the relation extraction model established one row in the database, named PK estimate, and was initially composed by one parameter mention and its central value or range. Then, using the rest of the relations predicted by the model, the entity linker, and the abstract metadata, each PK estimate was associated with their parameter identifier (from the entity linker), potential deviation measurements, units, compare terms, original sentence, and PubMed identifier. Finally, unit mentions were normalised using the approach described in the augmentation section 6.2.3 (Figure 6.8).

The resulting database was named PKUnlocked and fulfilled objective O₅₋₁.

7.1.2 Statistics

The initial results from the construction process are displayed in Table 7.1. First, it can be observed that only 93.8% of papers retrieved had the abstract available in PubMed, while for the rest, only the title was available in the XML file. Publications without abstracts were then excluded and often involved old articles or publications not written in English. From those papers with abstract, in 57.64% of cases, the pipeline extracted some PK estimate from the abstract. This result is aligned with comments from PK annotators involved in chapter 3, who noticed that several abstracts mentioned some parameter or PK modelling approach, but values were only reported in the full text. Therefore, this result suggests that extracting information from the full text is crucial for obtaining high recall on the extraction of PK parameter estimates from the literature.

TABLE 7.1: Statistics of the construction process of PKUnlocked.

Initial papers #	Papers with abstract # (%)	Abstracts with PK estimates # (%)	PK estimates #
120,913	113,415 (93.80%)	65,377 (57.64%)	274,277

¹ The relation extraction pipeline with x3 augmentation was used.

A total of 274,277 PK estimates were extracted and present in PKUnlocked. This result indicates that for those publications where PK estimations were reported in the abstract, one could expect an average of 4.2 estimates reported. Figure 7.1 stratified those estimations by the type of PK parameter, showing the 15 parameter types with most estimations reported and the rest grouped into the *Others* class. As it can be observed, a large proportion of PK estimates were classified as *NIL*, which accounted for outside of the knowledge base entities. After exploration of these cases, it was observed that the model often extracted measures that were not PK parameters but other parameters often related to pharmacodynamics, such as *flow rates*, *maximum tolerated doses*, *viral load*, *recovery rates*, amongst others. This high number of non-PK parameters extracted could be the result of not applying the PK NER algorithm presented in chapter 4. It is worth noting that the relation extraction model was trained with sentences that at least mentioned one PK parameter detected by the NER model. However, this filtering was not applied to build PKUnlocked to avoid missing potential PK parameters not detected by the NER model and accelerate the overall prediction. As a result, many non-PK measurements were detected by the relation extraction model that seemed to focus on extracting measurements that were expressed with a similar syntactic structure than those for PK parameters, regardless of whether they were PK parameters. However, this behaviour could be desirable if the entity linker correctly classified non-PK mentions as *NIL* entities.

Finally, it was observed that the most common measurements were related to the drug's maximum concentration (C_{max}) and elimination half-lives ($t_{1/2}$ and $t_{1/2,\beta}$), followed by clearances (CL and CL_{po}), area under the curves (AUC_{inf} , AUC_t), bioavailability (F), time to reach C_{max} (t_{max}), volumes of distribution (V , V_{ss}), IC_{50} , $C_{through}$ and the mean residence time (MRT), respectively.

7.1.3 Quality assessment

Precision, Recall and F1 were estimated for the overall extraction pipeline to quantify the expected quality of PKUnlocked. For this, 50 articles were randomly sampled from the pool of 113,415 abstracts for inspection. Since the most likely use case of PKUnlocked is the collation of values for specific parameters, for each abstract, the number of true positives, false positives and false negatives was annotated with the following considerations:

- **True Positive:** Rows in PKUnlocked where the parameter type was correctly predicted (according to its knowledge base identifier) and associated to the correct central and deviation values, units and compare mentions considering strict matching.

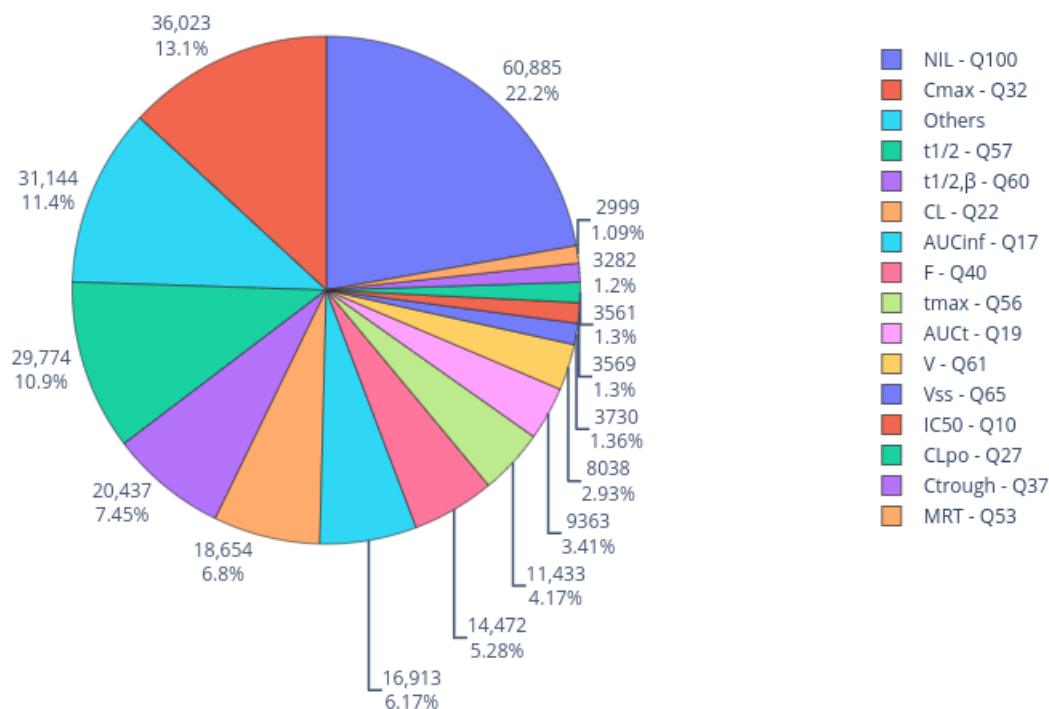


FIGURE 7.1: Pie chart displaying the frequency of PK estimates in PKUnlocked stratified by the type of parameter. The legend shows the main entity name together with the knowledge base identifier, sorted by their frequency from top to bottom. The top-15 most frequent parameter types were included and the rest were grouped into the *Others* category. The chart shows the total number of estimates for a given entity followed by their relative % over all estimates in the PKUnlocked.

- **False Positive:** Rows in PKUnlocked where either the parameter identifier or the span boundaries of central and deviation values, units, or compare terms were incorrect.
- **False Negatives:** Rows that should have been extracted in PKUnlocked but were not extracted. Note that, in cases where the parameter mentioned, value and units were correctly extracted, but the parameter id was incorrect, that would have been considered both, False positive and False negative.

Since the knowledge base identifiers would be used to filter parameter values and aggregate estimates across study types, *NIL* predictions were not considered as positive examples. For instance, if a value and units were correctly extracted for “MIC” (which was not in the knowledge base) and the predicted entity type was *NIL*, that was ignored and not considered a true or false positive. Instead, if “MIC” would have been predicted as another entity in the knowledge base, that would be considered a false positive. Similarly, estimates with parameter mentions that could not be linked to a knowledge base identifier (apart from *NIL*) were not considered in the count of true/false positives. In other words, the evaluation considered estimates of all knowledge base entities as positives except *NIL*.

Ketamine distribution described by a recirculatory pharmacokinetic model is not stereoselective.

Differences in the pharmacokinetics of the enantiomers of ketamine have been reported.

The authors sought to determine whether these differences extend to pulmonary uptake and peripheral tissue distribution and to test the hypothesis that tissue distribution of the stereoisomers differs because of carrier-mediated drug transport.

The dispositions of markers of intravascular space and blood flow (indocyanine green, ICG) and total body water and tissue perfusion (antipyrine) were determined along with S-(+)- and R-(-)-ketamine in five mongrel dogs.

The dogs were studied while anesthetized with 2.0% halothane.

Marker and drug dispositions were described by recirculatory pharmacokinetic models based on frequent early and less-frequent later arterial blood samples.

These models characterize pulmonary uptake and the distribution of cardiac output into parallel peripheral circuits.

Plasma elimination clearance PK of the S-(+)-ketamine enantiomer, 29.9 VALUE ml x min(-1) x kg(-1) UNITS, was higher than that of the R-(-)-enantiomer, 22.2 VALUE ml x min(-1) x kg(-1) UNITS.

The apparent pulmonary tissue volumes PK of the ketamine S-(+) and R-(-)-enantiomers (0.31 VALUE l UNITS) did not differ and was approximately twice that of antipyrine (0.16 VALUE l UNITS).

The peripheral tissue distribution volumes and clearances and the total volume of distribution PK (2.1 VALUE l/kg UNITS) were the same for both stereoisomers when elimination clearances were modeled from the rapidly equilibrating peripheral compartment.

Although the elimination clearance of S-(+)-ketamine is 35% greater than that of the R-(-)-enantiomer, there is no difference in the apparent pulmonary tissue volume or peripheral tissue distribution between the stereoisomers, suggesting that physicochemical properties of ketamine other than stereoisomerism determine its perfusion-limited tissue distribution.

Parameter	ParamID	Value	Units	Units_std	Deviation	DevUnits	DevUnits_std	Compare
Plasma elimination clearance	CL-Q22	29.9	ml x min(-1) x kg(-1)	[ml] / [kg·min]				
Plasma elimination clearance	CL-Q22	22.2	ml x min(-1) x kg(-1)	[ml] / [kg·min]				
apparent pulmonary tissue volumes	NIL-Q100	0.31	l	l				
apparent pulmonary tissue volumes	NIL-Q100	0.16	l	l				
total volume of distribution	V-Q61	2.1	l/kg	[l] / [kg]				

FIGURE 7.2: Example of the evaluation interface built for PKUnlocked. The abstract is displayed with the title, and the pipeline predictions were highlighted at the entity level. The extracted rows for a given abstract were displayed at the bottom in tabular format. From this visualisation, true/false positives and false negatives were derived.

A web interface was built with Dash² to facilitate the evaluation, displaying an abstract, and if present, the extracted information in PKUnlocked. The interface allowed to randomly select an abstract from the pool of 113,415 abstracts, and if the pipeline extracted estimates from that abstract, the corresponding PKUnlocked rows were displayed. An example of one abstract’s visualisation can be observed in Figure 7.2. By looking at the tables at the bottom of the interface and the whole abstract, the number of true/false positives and false negatives were annotated for a given abstract.

The pipeline’s Precision, Recall and F1 scores derived from this evaluation were 84.04%, 90.28% and 87.05%, respectively. The first pattern observed after the inspection was that the majority of non-PK parameter estimates were correctly linked to the *NIL* class, which attenuated the effect of extracting estimates not related to PK parameters. Overall, the pipeline missed very few measurements, and the recall was mostly limited by errors at the NER level, where values or unit boundaries were incorrectly predicted. Occasional *Central_{val}* mispredictions were also observed when the estimates of multiple parameter types were reported in the same sentence. Finally, EL errors also accounted for a number of both false negatives and false positives. Figure 7.3 shows a common source of error that generated false positives and false negatives. In these cases, the model correctly detected entities and relations, but it did not successfully associate the first parameter mentions *distribution* and *central* to the correct identifier. This pattern suggested that the entity linker might rely more on parameter surface forms instead of relevant context.

Another common source of error limiting the pipeline’s precision was the incorrect link of concentration mentions (without a knowledge base identifier) such as “total plasma concentration” or “salivary concentration” to the Cmax entity instead of *NIL* or “MIC” mentions linked to “IC50” instead of *NIL*. This pattern suggested that when surface forms of *NIL* mentions were similar to the knowledge base title of another entity, it generated confusion in the model predictions.

Overall, the evaluation performed achieved objective O₅₋₂ and indicated that the information stored in PKUnlocked covered a large proportion of PK estimates reported in scientific abstracts while also having several non-PK estimations. It is worth noting that, in most occasions, when extracting estimates for a specific parameter, several false positives could be easily removed by discarding estimates with units not related to the parameter of interest (e.g. “h” associated with a clearance estimate).

² <https://dash.plotly.com/>

(A) After intravenous administration, the mean **distribution PK** and **elimination half-lives PK** of ampicillin were **0.121 VALUE** and **0.624 VALUE** **h UNITS**, respectively.

Parameter	ParamID	Value	Units	Units_std	Deviation	DevUnits	DevUnits_std	Compare
distribution	CL12-Q30	0.121	h	h				
elimination half-lives	t1/2,β-Q60	0.624	h	h				

(B) **Central PK** and **intercompartmental clearances PK** were **3.6 VALUE** **liters/h UNITS** (relative standard error [RSE], **15.7 VALUE** **% UNITS**) and **6.58 VALUE** **liters/h UNITS** (RSE, **16.4 VALUE** **% UNITS**), respectively, and **central PK** and **peripheral volumes PK** were **7.3 VALUE** **liters UNITS** (RSE, **11.8 VALUE** **% UNITS**) and **3.9 VALUE** **liters UNITS** (RSE, **9.7 VALUE** **% UNITS**), respectively.

Parameter	ParamID	Value	Units	Units_std	Deviation	DevUnits	DevUnits_std	Compare
Central	CL12-Q30	3.6	liters/h	[l]/[h]	15.7	%	%	
intercompartmental clearances	CL12-Q30	6.58	liters/h	[l]/[h]	16.4	%	%	
central	CL12-Q30	7.3	liters	l	11.8	%	%	
peripheral volumes	V2-Q64	3.9	liters	l	9.7	%	%	

FIGURE 7.3: Example of two sentences where the extraction pipeline successfully predicted all entities and relations for each estimate but did not link some of the parameter mentions to the correct knowledge base identifier (i.e. *distribution* in example A and *central* in example B).

7.1.4 Potential use

When constructing ADME datasets, pharmacometricians might look for estimates of specific parameters (e.g. AUC, Clearance), drugs (e.g. amoxicillin, midazolam), administration routes (e.g. oral, intravenous), dose regimens, sub-populations (e.g. healthy/ill, child/adult) or clinical study designs. Therefore, when using databases similar to PKUnlocked, end-users must be able to filter the desired estimates according to their criteria. For this reason, a search interface built with Dash was constructed and proposed for filtering the desired estimates from PKUnlocked according to different criteria and is displayed in Figure 7.4.

Estimates from PKUnlocked are displayed in a dynamic table, where values can be filtered by the categories in each field/column (e.g. values were filtered by CL in Figure 7.4). Additionally, estimates could be inspected in context by selecting a specific cell and

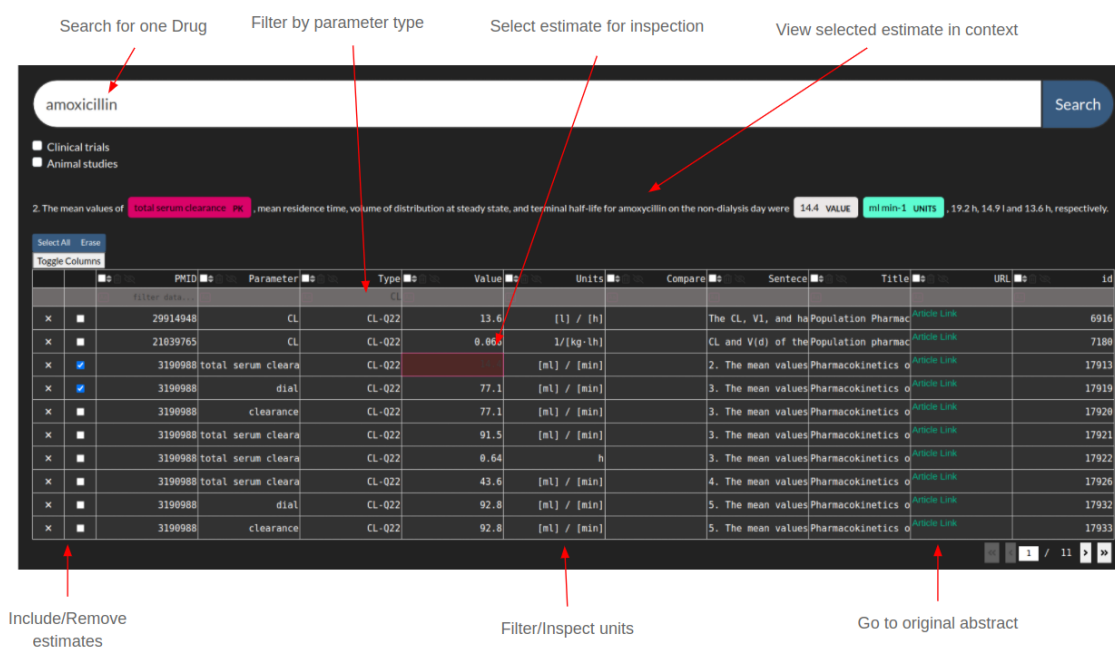


FIGURE 7.4: Example of a search interface to filter estimates from PKUnlocked.

visualising the sentence and their entities on the interface. Drug names could be introduced in the search bar. For this, the E-Utilities³ was used to retrieve the PubMed Ids resulting from searching that drug name in PubMed, and those rows in PKUnlocked that had PubMed Ids matching the search were selected. This approach allowed integrating the PubMed search engine to filter for estimates in abstracts mentioning a specific drug. In other words, when searching for “amoxicillin”, the PKUnlocked interface returned all the estimates appearing in abstracts from PKUnlocked and subset those estimates by the ones that also appeared in abstracts when searching “amoxicillin” in PubMed. Hence, this resulted in a list of candidate estimates for the desired drug, but users could click on the desired cell, see the estimate in context and decide whether to use that estimate by selecting the checkbox on the left-hand side of the dynamic table. It is noteworthy that this approach integrates any search that one could perform in PubMed (e.g. “amoxicillin AND clavulanic AND healthy”) to the PKUnlocked interface. The sentence in which the value appeared was displayed with the related entities highlighted to see the estimate in context. Additionally, users could see the article title (often helpful to filter for human/animal studies) and, if needed, go to the full-text article. After inspection, the selected estimates could be downloaded for plotting or use in ADME datasets. Early inspection indicated that sentence and title were often enough to determine the drug related to the estimate, the study population conditions, and the administration

³ Public API that allows retrieval of entries in PubMed, PMC and other NCBI databases <https://www.ncbi.nlm.nih.gov/books/NBK25501/>.

route. However, additional context was required occasionally, and future approaches might benefit from displaying more context related to the estimated parameter.

This interface allowed users to inspect the extracted values in detail before using them for specific applications while accelerating the search of PK estimates with text mining. However, it is noteworthy that multiple approaches could be used to build search interfaces, and this was only an approach for a simple yet flexible and fast interface to curate ADME datasets using the models developed in this thesis.

7.2 Case study

Lonsdale et al. 2019 [276] performed a meta-analysis of PK estimates from commonly used beta-lactams: amoxicillin \pm clavulanate, piperacillin-tazobactam and meropenem. The study compared the clearance and volume of distribution of each drug reported across the literature for critically and non-critically ill neonates, children and adults. The authors screened 2082 articles and extracted PK estimates from 130 publications. Given the variety of drugs considered, study populations, and the reported detail on the search and extraction of PK estimates, this publication was selected as a case study to tackle objective O₅₋₃, analysing areas where the models developed in this thesis could have accelerated the curation process. The analyses were performed at two levels: (1) selection of documents and (2) extraction of estimates.

7.2.1 Document search

One area where the models developed in this thesis might accelerate the extraction performed by Lonsdale et al. [276] is in the identification of relevant literature. The authors used PubMed and EMBASE to screen through publications with values of interest. However, in this thesis, the document classifier presented in chapter 3 was deployed through a search interface at PKPDAI⁴ specifically to accelerate the search of publications reporting PK estimates. The PKPDAI search engine uses E-Utilities to allow users to query one or more⁵ drug names and retrieve the list of identifiers resulting from performing that search in PubMed and also classified as “Relevant” by the document classifier. Then, those publications retrieved by the document classifier and also appearing in the list of identifiers returned by E-utilities are displayed in the interface, sorted by their likelihood⁶ of being classified as “Relevant” (i.e. reporting estimations of PK parameters). Hence, the different searches performed by Lonsdale et al. in PubMed were reproduced using the search engine in PKPDAI:

⁴ <https://pkpdai-search.com/pkdocsearch>

⁵ Allowing for operators such as “AND” or “OR” to query multiple drugs.

⁶ Likelihood of a particular class was obtained using the posterior probability of XGBoost predictions.

Amoxicillin ± Clavulanic For this search, Lonsdale et al. reviewed 854 articles from PubMed⁷ and selected 23 articles from which parameter estimates were extracted for the PK meta-analysis. The search “amoxicillin OR clavulanic OR clavulanate” was performed at PKPDAI filtering for publications before 2018, which resulted in 488 articles. In those 488 articles, 22 of the 23 selected publications were present (95.95% recall) and ranked amongst the top 280 entries in PKPDAI. The only publication missed by the document classifier (pmid=445959) did not have the abstract available in the XML file, which is likely to have caused this omission. Hence, the PKPDAI search provided a reduction of 42.86% in the number of publications that required screening while missing one relevant publication.

Piperacillin-Tazobactam For these drugs, Lonsdale et al. screened 661 publications and selected 54 to extract parameter estimates. The search “piperacillin OR tazobactam” in PKPDAI returned 300 documents from which all the 54 selected publications were present (100% recall). Hence, the search engine reduced the number of publications requiring screening by 54.61% while preserving all the relevant publications.

Meropenem A total of 567 publications were screened by Lonsdale et al. for meropenem, and 52 were selected for including PK estimates in the analysis. In PKPDAI, the search “meropenem” returned 265 entries from which 51/52 of the publications selected were present. The only publication missed (pmid=15793108) only mentioned pharmacodynamic estimates in the abstract, but PK estimates were not mentioned or discussed. PKPDAI reduced the number of publications that needed screening by 53.26% while only missing one relevant publication for this drug.

Overall, the recall rate of the document classifier search presented in this thesis was 98.46% for the publications used in Lonsdale et al.’s meta-analysis while reducing the number of publications requiring screening almost 50% (from 2082 to 1053). Additionally, it is worth noting that the PKPDAI’s search engine ranks publications by their likelihood of reporting PK parameters. Hence, this is likely to accelerate the identification of the studies included in Lonsdale et al. and enhance the finding of new relevant publications. Overall, this analysis achieved objective O₅₋₄. Finally, further search criteria could have been used to reduce the number of papers requiring screening (e.g. filtering by articles not mentioning animals in the MeSH terms), but those were not applied to perform a one-to-one comparison with Lonsdale et al. approach.

7.2.2 Extraction of estimates

Pharmacokinetic estimates were independently extracted for each drug in Lonsdale et al. [276]. However, it is worth noting that parameter estimates were not often reported in

⁷ See Figure 1 in Lonsdale et al. [276] for details on search results.

the abstract but in the full text. Therefore, analyses were performed to determine the proportion of papers that reported estimates in abstracts and from those which were stored in PKUnlocked. The parameters extracted in Lonsdale et al. [276] included total/systemic clearance and volume of distribution.

Amoxicillin ± Clavulanic From the 23 papers analysed in Lonsdale et al., ten reported clearance and volume estimates in the abstract (43.48%), indicating that the majority of estimates were extracted from full text. For clearance, out of 14 values found in the abstracts, 14 were correctly extracted in PKUnlocked. For the volume of distribution, from 12 values found, 11 were correctly extracted (one completely missed) in PKUnlocked. Overall, for amoxicillin and clavulanic abstracts, recall rates of clearance and volume of distribution estimates were 100% and 91.67%, respectively.

Piperacillin-Tazobactam From the 54 papers analysed by Lonsdale et al., 23 (42.49%) reported clearance estimates in the abstract and 18 (33.33%) reported volume of distribution estimates in the abstract. For clearance, 34 estimates were found in the abstracts (including both drugs), from which 33 were correctly stored in PKUnlocked. For the volume of distribution, from 24 values found in abstracts, 21 were correctly stored in PKUnlocked. In summary, for piperacillin and tazobactam abstracts, recall rates in PKUnlocked of clearance and volume of distribution estimates were 97.05% and 87.5%, respectively.

Meropenem From 53 papers analysed by Lonsdale et al. for meropenem, 24 (45.28%) reported clearance estimates in the abstract and 22 (41.51%) reported volume of distribution. Out of 24 clearance values found in the abstract, 22 were correctly extracted in PKUnlocked. For the volume of distribution, 22 values were found in abstracts, and 21 were correctly extracted. This resulted in a recall rate for clearance and volume of distribution estimates of 91.67% and 94.45%, respectively.

Overall, for those estimates reported in abstracts and used by Lonsdale et al., PKUnlocked achieved 95.83% and 91.38% recall for clearance and volume of distribution estimates, respectively. However, it is worth noting that those values used in Lonsdale et al.'s meta-analysis and extracted from the full-text ($\approx 60\%$) were not accessible in PKUnlocked. Therefore, this analysis suggests that applying the extraction pipeline to full text articles is crucial to obtain optimal recall of parameter estimates used in PK meta-analyses.

Finally, to assess the ability of PKUnlocked to reproduce PK profiles for the drugs and parameter types used by Lonsdale et al., the interface described in section 7.1.4 was used to search for clearance and volume of distribution estimates for amoxicillin,

clavulanic, piperacillin, tazobactam and meropenem. For each drug and parameter type the following procedure was applied:

1. **Drug and parameter filtering** The name of a single drug was searched on PKUnlocked interface. Then, estimates were independently filtered for total/systemic clearance (identifier CL-Q22 in the knowledge base) and volume of distribution (identifiers V-Q61 and Vss-Q65 in the knowledge base).
2. **Unit filtering** Only estimates with standardised units as “l/h” or “ml/min” were used for clearance, and “l” or “ml” for the volume of distribution. Notably, many estimates ($\approx 30\%$) found in PKUnlocked also reported clearance and volume of distribution estimates divided by body weight in “kg”. Converting those values to “l/h” or “l” would require extracting the mean weight of the patient population, which would have increased the curation time significantly. For this reason, values standardised by body weight were not used in this study, but future studies might benefit from approaches normalising these estimates.
3. **Unit conversion** Then, estimates reported in “ml/min” and “ml” were automatically converted to “l/h” and “l”, respectively, by applying the corresponding conversion factor to the central measurements.
4. **Manual inspection/cleaning** The candidate values for each drug were individually examined by a pharmacometrician to discard (1) estimates referring to other drugs, (2) estimates measured in non-human patients, (3) pipeline extraction errors, (4) estimates mentioned in one publication but experimentally obtained in another study (occasionally generating duplicates in PKUnlocked).
5. **Box plot for comparison** Finally, the central values for each estimate as reported in PKUnlocked before and after inspection were displayed using box-plots for each drug and parameter type independently.

The resulting estimates for each drug and parameter type were displayed in Figure 7.5 and 7.6, before and after manual inspection. Different observations were done during the curation process and analysis of extracted values:

Manual inspection All the steps were automatised except from the manual inspection (4). For this, it was observed that, on average, cleaning clearance and volume of distribution estimates for a specific drug and parameter using PKUnlocked interface took between 5 to 30 minutes (depending on the initial number of estimates). During inspection, removal of estimates for a specific drug was mainly due to different factors. First, a few estimates (especially the ones with low clearance values) came from studies performed in animals. Although animal studies were not explicitly removed in the

original search to preserve high recall, several approaches could be used in the future to filter those values a-priori, such as locating animal mentions in titles (e.g. Wang et al. [11]) or using PubMed search engine indexing (e.g. excluding studies with MeSH term “Animals”). Additionally, those drugs usually administered in combination with others often had multiple estimates not relating to the drug for which the values were reported. For instance, before cleaning, several estimates appeared in both, piperacillin and tazobactam or amoxicillin and clavulanic, since the same abstract mentioned both compounds. Hence, careful examination was performed in those cases. To reduce this curation time, future studies might benefit from text mining approaches locating drug mentions in the same sentence, paragraph or title, and relating each central measurement to the drug mentioned. Occasionally, a few mistakes ($n < 10$ for all clearance and volume of distribution estimates) were detected due to wrong units or entity linking. However, these were rare since the filtering by units and parameter type ensured relatively high precision of correct estimate extraction.

Extreme values Since the extracted values were not normalised by body weight or age, the estimates had relatively high dispersion with clearance values ranging from 0.2 to 35 l/h. After inspection it was observed that low clearance values were often related to studies performed in young children or patients with advanced renal diseases. On the other hand, high clearance values were detected in studies with obese or pregnant patients.

Further stratification and context Lonsdale et al. [276] compared the distribution of estimates for different post-menstrual ages. Additionally, other factors could influence the PK estimates, such as other patient conditions, gender, the route of drug administration, the dose or even the PK modelling approach. Hence, the distributions observed in Figure 7.5 and 7.6 could be further stratified by routes, ages or conditions. Although this was not performed in this case study, it is worth noting that the PKUnlocked interface allowed inspecting estimates in context and one could efficiently filter/stratify those values using multiple criteria. Additionally, since the search engine was wrapped around the PubMed search engine, Medline indexing could also be exploited to pre-filter estimates appearing in certain study types. Rule- and dictionary-based approaches could also be used to filter values for specific contexts. For instance, Wang et al. [11] used a list of dictionary terms to filter clearance estimates by specific routes of administration and conditions (i.e. intravenous administration of midazolam in healthy volunteers). However, manual inspection by experts provided two benefits: (1) confidence by end-users on the information extracted and (2) ensuring that potentially relevant values were not taken away by other filtering approaches. Hence, since there is an infinite number of potential study designs, it is believed that any PK search interface using text mining would benefit from allowing experts to verify and select relevant parameter estimates during the curation process rather than extracting all estimates automatically end-to-end.

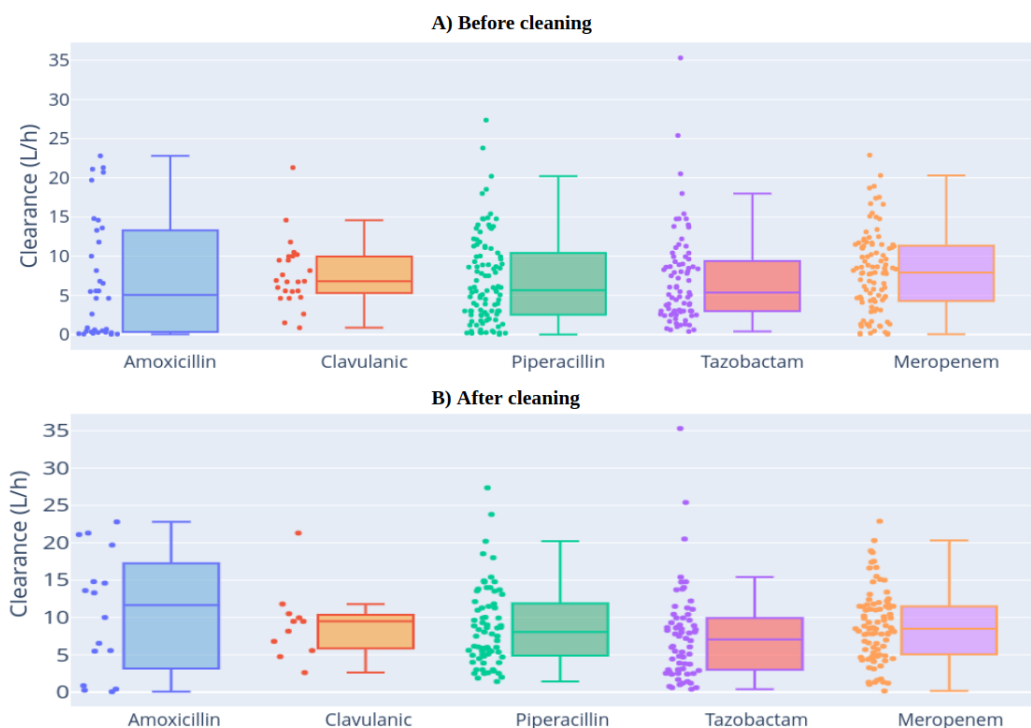


FIGURE 7.5: Box-plots displaying the distribution of clearance estimates extracted from the scientific literature and stored in PKUnlocked for Amoxicillin, Clavulanic, Piperacillin, Tazobactam and Meropenem. Individual estimates were represented next to each box-plot. The top panel (A) shows the values before being inspected by a pharmacometrician and the bottom panel (B) shows the values after inspection/removal of values not associated to that drug or measured in non-human studies.

Missed values Despite a large number of parameter estimates were extracted for the drugs and parameters studied, a large number of steps were sequentially applied to filter for relevant values, potentially cascading errors and missing relevant estimations. Therefore, it is important to acknowledge the following factors as the main potential causes for missing relevant PK estimates in this case study:

1. **Cascading errors through the extraction pipeline:** First, relevant documents could have been missed due to inaccurate model predictions by the document classifier. As discussed in chapter 3, the main causes for the document classifier missing relevant documents were (1) publications without abstract available, (2) publications not mentioning PK parameters in the abstract and (3) animal PK studies. Hence, one could expect lower recall for those cases. Additionally, only the relevant documents went to the relation extraction and entity linker algorithms, and, despite their high performance, one could expect relevant values being missed due to misspredictions in those stages.
2. **Unit standardisation** As previously mentioned, a large number of clearance and volume of distribution estimates were omitted in this case-study since their values

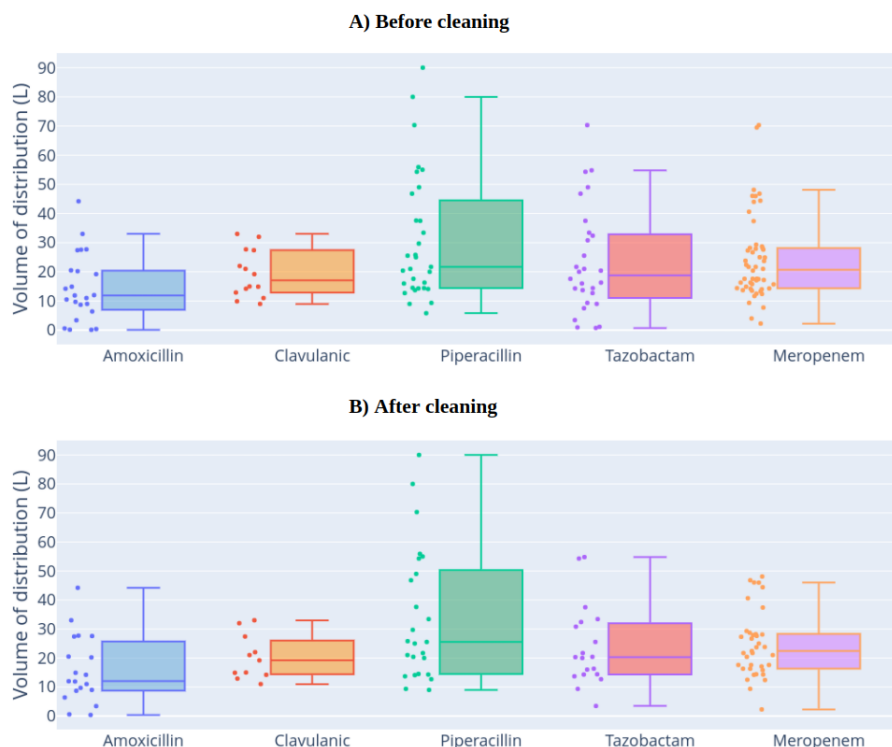


FIGURE 7.6: Box-plots displaying the distribution of volume of distribution estimates extracted from the scientific literature and stored in PKUnlocked for Amoxicillin, Clavulanic, Piperacillin, Tazobactam and Meropenem. Individual estimates were represented next to each box-plot. The top panel (A) shows the values before being inspected by a pharmacometrician and the bottom panel (B) shows the values after inspection/removal of values not associated to that drug or measured in non-human studies.

were standardised by the patient’s mean body weight. Hence, to ensure high recall rates for those parameter estimates, the body weight of the population studied will be a relevant factor to extract in future studies willing to maximise the number of estimates for a specific drug and context.

Comparison to clinical knowledge As reported by Lonsdale et al. and also observed in the extracted data, the drugs targetted in this case study had very similar PK profiles. Perhaps the most noticeable difference of extracted estimates between drugs was observed in the upper bound of the volume of distribution between piperacillin and the rest of the drugs (Figure 7.6). This remarkably high upper bound was also observed in Lonsdale et al. meta-analysis (see Table 1 in Lonsdale et al. [276]), which suggests that despite the potential values missed in PKUnlocked, the information extracted from abstracts was able to exhibit the same clinical difference. The PKs of vancomycin were compared to analyse further the ability of the information extracted to showcase known PK differences between drugs. As reported in Drugbank [53], the elimination half-life ($t_{1/2}$) of vancomycin in healthy patients ranges from 4 to 11 hours and can highly exceed those values in anephric patients, which is slower than the drugs studied in Lonsdale et al.

meta-analysis, which have a half-life around one hour in healthy volunteers according to DrugBank [53]. Hence, the half-lives of all drugs and vancomycin were automatically extracted from PKUnlocked and displayed in Figure 7.7 without manual curation. Despite the potential values missed in PKUnlocked and the number of estimates relating to other drugs or species, Figure 7.7 shows that the PK differences known by pharmacometricians between these drugs could be reproduced without expert intervention, with the distribution of vancomycin’s half-life estimates being significantly higher than in the other drugs. Although the clinical distribution of elimination $t_{1/2}$ could be reproduced for different drugs, a proportion of the estimates displayed in Figure 7.7 will refer to other drugs mentioned in the same abstract or species, and manual inspection or further pre-processing might be needed for this information to be used in applications that require high precision of a drug’s PK estimates.

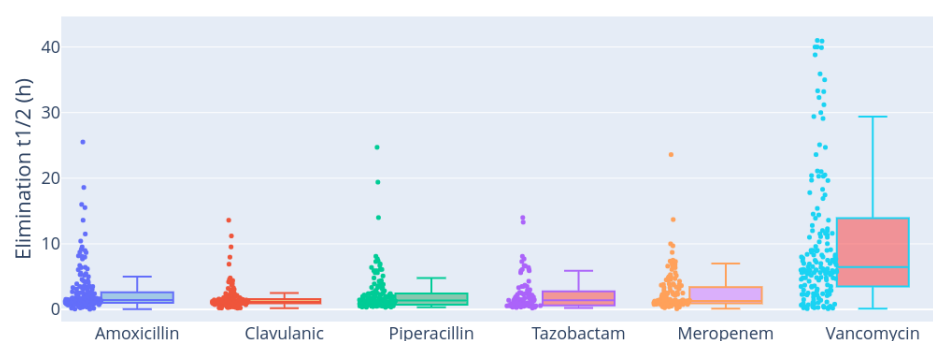


FIGURE 7.7: Box-plots displaying the half-life estimates extracted from the scientific literature and stored in PKUnlocked for Amoxicillin, Clavulanic, Piperacillin, Tazobactam, Meropenem and Vancomycin. Individual estimates were represented next to each box-plot. Values are shown after the drug was searched in PKUnlocked without being filtered/inspected by a pharmacometrician. The graph only displays values up to 45h but a few estimates for vancomycin reached up to 180h.

In conclusion, this chapter discussed and evaluated the application of the text mining approaches developed in this thesis from multiple angles and detected key applications and limitations for future studies, which fulfilled objective O_{5-5} .

Chapter 8

Conclusions

In this chapter, the main thesis aim and research questions established in section 1.4 were revisited and briefly discussed how they were answered by fulfilling the objectives established in each chapter. It is noteworthy that the different objectives were already discussed across chapters 3 to 7, and this section provides an overview of how the different research questions established were answered across chapters.

8.1 Summary of research findings

Q_1 : How can text mining approaches be applied to identify scientific publications reporting novel PK parameter estimates automatically across study types?

Chapter 3 addressed the task of document classification of PK articles. By addressing the objectives established in that chapter, it was estimated that an F_1 of 83.8% could be achieved to identify scientific publications reporting novel PK parameter estimates in PubMed. This result was obtained in annotated PubMed documents using different study designs, utilising bag of words and BioBERT embeddings to represent documents and XGBoost as a classifier. Additionally, it was shown that an extensive repository of PK publications reporting PK estimates could be automatically retrieved by applying the pipeline to PubMed articles. Finally, by addressing the fourth objective in chapter 7, it was observed that this collection of documents combined with a search engine could be used to accelerate a PK meta-analysis previously published by identifying publications in which PK parameters were estimated for specific drugs of interest.

Q_2 : How can the task of recognising parameter mentions of multiple parameter types be accomplished?

This research question was mainly answered in chapter 4. In this study, it was noticed that a low proportion of sentences across the literature mentioned PK parameters, which made the task of developing annotated data covering the diversity of PK parameters particularly challenging. However, by utilising active learning, the curation of PK data could be accelerated while improving the information provided by labelled sentences. Furthermore, by fine-tuning BioBERT on annotated corpora, the best performance NER was obtained, with over 90% F_1 score on strict entity matching of PK

parameter mentions. Finally, it was also noticed that training models on previously published corpora did not generalise well to multiple types of PK parameters found across the literature.

Q₃: How can the specific types of PK parameters be determined given mentions in context?

This research question was answered by addressing the objectives in Chapter 5. First, it was noticed that annotated data was not available for this task and that some parameter types had very sparse frequencies across the literature. By addressing the objectives in this chapter, a dataset annotating PK parameter mentions to one of 67 knowledge base entries was built together with a system that could efficiently link parameter types with very few or no training mentions. Specifically, it was observed that by encoding descriptions in a knowledge base and using a distance-based loss function, the model could deal with entities with very few training mentions more efficiently than sequence classification models. Additionally, simplifications of previously published architectures were proposed, and they were found to be beneficial when applying entity linking to settings with limited training mentions and a relatively small number of knowledge base entries. Finally, the chapter also analysed which types of PK parameters were more challenging to link given the training data used in the study and how researchers could add new PK entities to the knowledge base without requiring training samples for the model to link them. Overall, the best-performing model achieved a micro-accuracy of 88.08% across PK parameter types.

Q₄: How can suitable architectures be developed and evaluated to extract PK parameter estimates for multiple parameter types?

Chapter 6 answered this question by addressing six objectives established. In this chapter, the task of extracting PK estimates for multiple parameters in structured form was formulated as an end-to-end relation extraction problem with multiple entities and relations between them. By addressing the different objectives, a new dataset was annotated by field experts to train and evaluate models for this task, and different approaches were developed and compared to model the task. Specifically, it was found that jointly modelling the detection of relevant entities and their relations in a multi-task setting improved the model performance in comparison to optimising for each task independently. Furthermore, pre-training transformer models in biomedical text largely outperformed alternative approaches to encode sentence tokens, and using explicit representations of the context strictly between two entities was found crucial to obtain optimal performance. Additionally, it was shown that heuristic data augmentation could improve the recognition of PK parameter mentions and their relation to numerical estimates. Finally, an in-depth discussion of the leading causes of model errors was provided, and future directions were proposed. Overall, the best-performing system obtained an end-to-end F_1 score over 87% for extracting the different relations involved in parameter estimations, covering multiple types of PK parameters and numerical expressions.

Q₅: How can text mining approaches accelerate the curation of ADME datasets used by pharmacometricians?

This final research question was answered by addressing the objectives of Chapter 7. First, it was shown how the models developed in this thesis could be integrated into a single extraction pipeline to generate a database with over 250,000 parameter estimates reported in scientific abstracts, named PKUnlocked and covering 66 types of PK parameters. Next, the performance of the extraction pipeline and the quality of the database were evaluated. Then, the application of the database in accelerating a PK meta-analysis covering multiple drugs and parameters was studied. First, it was found that the work from chapter 3 significantly reduced the number of publications needed to be screened for curating the dataset, which showcased its application for accelerating different types of ADME datasets. Then, the recall rate for clearance and volume of distribution estimates for the multiple drugs studied was established at 95.83% and 91.38% in PKUnlocked. This result was obtained by comparing the estimates from scientific abstracts used in a published meta-analysis for multiple drugs and those stored in PKUnlocked. This result highlighted the ability of the work in this thesis to accelerate the curation of PK parameters used in ADME datasets. Finally, potential approaches to search and filter estimates of interest in PKUnlocked were compared for multiple drugs, and it was found that with minimal expert checks, the PK profiles of multiple drugs could be reproduced using estimates from several parameters. Finally, limitations of the approach presented were analysed, and it was found that a crucial area for improving the application of this pipeline during curation of ADME datasets in future studies was to extract estimates reported in the full text.

After addressing the objectives in each chapter, it was considered that the aim established in this thesis was attained:

Enhancing the curation of pharmacokinetic datasets by automatically structuring estimates reported across the scientific literature

8.2 Open Problems and Future work

The work presented in this thesis can be advanced by exploring its feasibility in drug development and pharmacokinetic modelling applications. For instance, distributions of parameter estimates automatically extracted could be used as priors for improving physiologically-based pharmacokinetic models in drug development. Additionally, the feasibility of improving predictions of *in vivo* PK parameters by semi-automatic curation of extensive ADME datasets could be evaluated in future studies. Another potential application is using the named entity recogniser and linker to improve the characterisation of drug-drug interactions by determining the specific type of parameter involved in a particular interaction.

From an extraction perspective, the methods applied could be used to extract estimates reported in sections from the article full text, which could significantly increase the number of estimates in the database. In the full text, many pharmacokinetic estimations are reported in tables. The extraction of numerical estimates from tables remains an open challenge since it might require dealing with the semi-structured format of tables and accurately parsing this information from sources such as PDFs. However, if table extraction is addressed, the amount and quality of the information in PKUnlocked could be highly extended. Finally, the application of the different tools developed in this thesis could be studied in clinical trial reports or drug inserts, which often contain a wealth of pharmacokinetic estimates.

Bibliography

- [1] Elena Voita. NLP course for you, Sep 2020. URL https://lena-voita.github.io/nlp_course.html.
- [2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [3] Renli Teng, Sharen Muldowney, Yonggang Zhao, Jolene Kay Berg, Jonathan Lu, and Naeem D Khan. Pharmacokinetics and pharmacodynamics of ticagrelor in subjects on hemodialysis and subjects with normal renal function. *European journal of clinical pharmacology*, 74(9):1141–1148, 2018.
- [4] Dimitrios Piliouras. *Text Mining for Drug Discovery*. PhD thesis, 2014.
- [5] Michael Schlander, Karla Hernandez-Villafuerte, Chih-Yuan Cheng, Jorge Mestre-Ferrandiz, and Michael Baumann. How much does it cost to research and develop a new drug? a systematic review and assessment. *PharmacoEconomics*, pages 1–27, 2021.
- [6] Chi Heem Wong, Kien Wei Siah, and Andrew W Lo. Estimation of clinical trial success rates and related parameters. *Biostatistics*, 20(2):273–286, 2019.
- [7] Joseph A. DiMasi, Henry G. Grabowski, and Ronald W. Hansen. Innovation in the pharmaceutical industry: New estimates of R&D costs. *Journal of Health Economics*, 47:20–33, 2016. ISSN 18791646. doi: 10.1016/j.jhealeco.2016.01.012.
- [8] Paul Morgan, Piet H Van Der Graaf, John Arrowsmith, Doug E Feltner, Kira S Drummond, Craig D Wegner, and Steve DA Street. Can the flow of medicines be improved? fundamental pharmacokinetic and pharmacological principles toward improving phase ii survival. *Drug discovery today*, 17(9-10):419–424, 2012.
- [9] Alan M. Palmer. New horizons in drug metabolism, pharmacokinetics and drug discovery. *Drug News Perspect*, 16(1):57–62, 2003.
- [10] Jan Grzegorzewski, Janosch Brandhorst, Kathleen Green, Dimitra Eleftheriadou, Yannick Dupont, Florian Barthorscht, Adrian Köller, Danny Yu Jia Ke, Sara De Angelis, and Matthias König. Pk-db: pharmacokinetics database for individualized and stratified computational modeling. *Nucleic acids research*, 49(D1): D1358–D1364, 2021.
- [11] Zhiping Wang, Seongho Kim, Sara K Quinney, Yingying Guo, Stephen D Hall, Luis M Rocha, and Lang Li. Literature mining on pharmacokinetics numerical data: a feasibility study. *Journal of biomedical informatics*, 42(4):726–735, 2009.

- [12] Franco Lombardo, Giuliano Berellini, and R. Scott Obach. Trend analysis of a database of intravenous pharmacokinetic parameters in humans for 1352 drug compounds. *Drug Metabolism and Disposition*, 46(11):1466–1477, 2018. ISSN 1521009X. doi: 10.1124/dmd.118.082966.
- [13] Efpia Mid, Workgroup Sf, R Burghaus, V Cosson, S Y A Cheung, M Chenel, O Dellapasqua, and N Frey. Good Practices in Model-Informed Drug Discovery and Development : Practice , Application , and Documentation. (March):93–122, 2016. doi: 10.1002/psp4.12049.
- [14] Jenny Y. Chien, Stuart Friedrich, Michael A. Heathman, Dinesh P. de Alwis, and Vikram Sinha. Pharmacokinetics/pharmacodynamics and the stages of drug development: Role of modeling and simulation. *AAPS Journal*, 7(3), 2005. ISSN 15221059. doi: 10.1208/aapsj070355.
- [15] S. A.G. Visser, D. P. De Alwis, T. Kerbusch, J. A. Stone, and S. R.B. Allerheiligen. Implementation of quantitative and systems pharmacology in large pharma, 2014. ISSN 21638306.
- [16] Sergey Ermakov, Brian J. Schmidt, Cynthia J. Musante, and Craig J. Thalhauser. A Survey of Software Tool Utilization and Capabilities for Quantitative Systems Pharmacology: What We Have and What We Need. *CPT: Pharmacometrics and Systems Pharmacology*, 2019. ISSN 21638306. doi: 10.1002/psp4.12373.
- [17] Sumudu P. Leelananda and Steffen Lindert. Computational methods in drug discovery. *Beilstein Journal of Organic Chemistry*, 12(January):2694–2718, 2016. ISSN 18605397. doi: 10.3762/bjoc.12.267.
- [18] Rodolfo S. Simões, Vinicius G. Maltarollo, Patricia R. Oliveira, and Kathia M. Honorio. Transfer and multi-task learning in QSAR modeling: Advances and challenges. *Frontiers in Pharmacology*, 9(FEB):1–7, 2018. ISSN 16639812. doi: 10.3389/fphar.2018.00074.
- [19] Theodora Katsila, Georgios A. Spyroulias, George P. Patrinos, and Minos Timotheos Matsoukas. Computational approaches in target identification and drug discovery. *Computational and Structural Biotechnology Journal*, 14:177–184, 2016. ISSN 20010370. doi: 10.1016/j.csbj.2016.04.004.
- [20] M. Paul Gleeson, Nigel J. Waters, Stuart W. Paine, and Andrew M. Davis. In silico human and rat Vss quantitative structure-activity relationship models. *Journal of Medicinal Chemistry*, 49(6):1953–1963, 2006. ISSN 00222623. doi: 10.1021/jm0510070.
- [21] Yuchen Wang, Haichun Liu, Yuanrong Fan, Xingye Chen, Yan Yang, Lu Zhu, Junnan Zhao, Yadong Chen, and Yanmin Zhang. In Silico Prediction of Human Intravenous Pharmacokinetic Parameters with Improved Accuracy. *Journal of*

- Chemical Information and Modeling*, 59(9):3968–3980, 2019. ISSN 1549-9596. doi: 10.1021/acs.jcim.9b00300.
- [22] Zhuyifan Ye, Yilong Yang, Xiaoshan Li, Dongsheng Cao, and Defang Ouyang. An Integrated Transfer Learning and Multitask Learning Approach for Pharmacokinetic Parameter Prediction. *Molecular Pharmaceutics*, 2019. ISSN 15438392. doi: 10.1021/acs.molpharmaceut.8b00816.
- [23] Filip Miljkovic, Anton Martinsson, Olga Obrezanova, Beth Williamson, Martin Johnson, Andy Sykes, Andreas Bender, and Nigel Greene. Machine learning models for human in vivo pharmacokinetic parameters with in-house validation. *Molecular pharmaceutics*, 2021.
- [24] Mark C Rogge. The scope of preclinical drug development: an introduction and framework. In *Preclinical Drug Development*, pages 13–18. CRC Press, 2016.
- [25] Iftexhar Mahmood. Allometric issues in drug development. *Journal of Pharmaceutical Sciences*, 88(11):1101–1106, 1999. ISSN 00223549. doi: 10.1021/js9902163.
- [26] Elisabet I. Nielsen and Lena E. Friberg. Pharmacokinetic-pharmacodynamic modeling of antibacterial drugs. *Pharmacological Reviews*, 65(3):1053–1090, 2013. ISSN 00316997. doi: 10.1124/pr.111.005769.
- [27] Masoud Jamei. Recent Advances in Development and Application of Physiologically-Based Pharmacokinetic (PBPK) Models: a Transition from Academic Curiosity to Regulatory Acceptance, 2016. ISSN 2198641X.
- [28] Abdul Naveed Shaik and Ansar Ali Khan. Physiologically based pharmacokinetic (PBPK) modeling and simulation in drug discovery and development. *ADMET and DMPK*, 7(1):1–3, 2019. ISSN 18487718. doi: 10.5599/admet.667.
- [29] P. Zhao, L. Zhang, J. A. Grillo, Q. Liu, J. M. Bullock, Y. J. Moon, P. Song, S. S. Brar, R. Madabushi, T. C. Wu, B. P. Booth, N. A. Rahman, K. S. Reynolds, E. Gil Berglund, L. J. Lesko, and S. M. Huang. Applications of physiologically based pharmacokinetic (PBPK) modeling and simulation during regulatory review. *Clinical Pharmacology and Therapeutics*, 2011. ISSN 00099236. doi: 10.1038/clpt.2010.298.
- [30] Sven Björkman. Prediction of the volume of distribution of a drug: which tissue-plasma partition coefficients are needed? *Journal of Pharmacy and Pharmacology*, 54(9):1237–1245, 2002. ISSN 0022-3573. doi: 10.1211/002235702320402080.
- [31] Trevor N. Johnson, Thomas Kerbusch, Barry Jones, Geoffrey T. Tucker, Amin Rostami-Hodjegan, and Peter A. Milligan. Assessing the efficiency of mixed effects modelling in quantifying metabolism based drug-drug interactions: Using in vitro data as an aid to assess study power. *Pharmaceutical Statistics*, 2009. ISSN 15391604. doi: 10.1002/pst.373.

- [32] Karen Rowland Yeo, Masoud Jamei, Jiansong Yang, Geoffrey T. Tucker, and Amin Rostami-Hodjegan. Physiologically based mechanistic modelling to predict complex drug-drug interactions involving simultaneous competitive and time-dependent enzyme inhibition by parent compound and its metabolite in both liver and gut-The effect of diltiazem on the time-c. *European Journal of Pharmaceutical Sciences*, 2010. ISSN 09280987. doi: 10.1016/j.ejps.2009.12.002.
- [33] Sabarinath S. and Madabushi R. Physiologically based pharmacokinetic (PBPK) simulations for quantitatively predicting drug interactions of simvastatin and diltiazem. *Journal of Clinical Pharmacology*, 2011. ISSN 0091-2700. doi: 10.1177/0091270010418046.
- [34] Fang Li, Partha Nandy, Shuchean Chien, Gary J. Noel, and Christoffer W. Tornøe. Pharmacometrics-based dose selection of levofloxacin as a treatment for postexposure inhalational anthrax in children. *Antimicrobial Agents and Chemotherapy*, 2010. ISSN 00664804. doi: 10.1128/AAC.00667-09.
- [35] K. R. Przybylak, J. C. Madden, E. Covey-Crump, L. Gibson, C. Barber, M. Patel, and M. T.D. Cronin. Characterisation of data resources for in silico modelling: benchmark datasets for ADME properties. *Expert Opinion on Drug Metabolism and Toxicology*, 14(2):169–181, 2018. ISSN 17447607. doi: 10.1080/17425255.2017.1316449. URL <https://doi.org/10.1080/17425255.2017.1316449>.
- [36] Roosmarijn F.W. De Cock, Chiara Piana, Elke H.J. Krekels, Meindert Danhof, Karel Allegaert, and Catherijne A.J. Knibbe. The role of population PK-PD modelling in paediatric clinical research. *European Journal of Clinical Pharmacology*, 67(SUPPL. 1), 2011. ISSN 00316970. doi: 10.1007/s00228-009-0782-9.
- [37] Joseph F. Standing. Understanding and applying pharmacometric modelling and simulation in clinical practice and research. *British Journal of Clinical Pharmacology*, 83(2):247–254, 2017. ISSN 13652125. doi: 10.1111/bcp.13119.
- [38] Ene I. Ette and Paul J. Williams. *Pharmacometrics: The Science of Quantitative Pharmacology*. 2006. ISBN 9780471677833. doi: 10.1002/9780470087978.
- [39] Peter L. Bonate and Danny R. Howard. *Pharmacokinetics in drug development*. 2011. ISBN 9781441979360. doi: 10.1007/978-1-4419-7937-7.
- [40] D. R. Mould and R. N. Upton. Basic concepts in population modeling, simulation, and model-based drug development - Part 2: Introduction to pharmacokinetic modeling methods. *CPT: Pharmacometrics and Systems Pharmacology*, 2(4), 2013. ISSN 21638306. doi: 10.1038/psp.2013.14.
- [41] Johan Gabrielsson and Daniel Weiner. Non-compartmental analysis. *Methods in Molecular Biology*, 2012. ISSN 10643745. doi: 10.1007/978-1-62703-50-2_16.

- [42] Arthur J Huang, Shiew-Mei and Lertora, Juan JL and Markey, Sanford P and Atkinson Jr. *Principles of clinical pharmacology*. Third edition, 2012. ISBN 9788578110796. doi: 10.1017/CBO9781107415324.004.
- [43] Heng-Yi Wu, Shreyas Karnik, Abhinata Subhadarshini, Zhiping Wang, Santosh Philips, Xu Han, Chienwei Chiang, Lei Liu, Malaz Boustani, Luis M Rocha, et al. An integrated pharmacokinetics ontology and corpus for text mining. *BMC bioinformatics*, 14(1):1–15, 2013.
- [44] Chayan Acharya, Andrew C. Hooker, Gülbeyaz Yıldız Türkyılmaz, Siv Jönsson, and Mats O. Karlsson. A diagnostic tool for population models using non-compartmental analysis: The ncappc package for R. *Computer Methods and Programs in Biomedicine*, 2016. ISSN 18727565. doi: 10.1016/j.cmpb.2016.01.013.
- [45] Win L. Chiou. Critical evaluation of the potential error in pharmacokinetic studies of using the linear trapezoidal rule method for the calculation of the area under the plasma level-time curve. *Journal of Pharmacokinetics and Biopharmaceutics*, 1978. ISSN 0090466X. doi: 10.1007/BF01062108.
- [46] Johan Gabrielsson Weiner and Daniel. Non-compartmental Analysis Johan. In , *Computational Toxicology*, volume 929, pages 10–13. 2012. ISBN 9781627030502. doi: 10.1007/978-1-62703-050-2.
- [47] D. R. Mould and R. N. Upton. Basic concepts in population modeling, simulation, and model-based drug development. *CPT: Pharmacometrics and Systems Pharmacology*, 1(1):1–14, 2012. ISSN 21638306. doi: 10.1038/psp.2012.4.
- [48] Eva Germovsek, Charlotte I.S. Barker, Mike Sharland, and Joseph F. Standing. Scaling clearance in paediatric pharmacokinetics: All models are wrong, which are useful? *British Journal of Clinical Pharmacology*, 83(4):777–790, 2017. ISSN 13652125. doi: 10.1111/bcp.13160.
- [49] Ene I. Ette and Paul J. Williams. Population pharmacokinetics II: Estimation methods. *Annals of Pharmacotherapy*, 38(11):1907–1915, 2004. ISSN 10600280. doi: 10.1345/aph.1E259.
- [50] Melanie A. Felmlee, Marilyn E. Morris, and Donald E. Mager. *Mechanism-based pharmacodynamic modeling*, volume 929. 2012. ISBN 9781627030496. doi: 10.1007/978-1-62703-50-2_21.
- [51] John C. Dearden. In silico prediction of ADMET properties: How far have we come? *Expert Opinion on Drug Metabolism and Toxicology*, 2007. ISSN 17425255. doi: 10.1517/17425255.3.5.635.
- [52] George P Rédei. PubChem. In *Encyclopedia of Genetics, Genomics, Proteomics and Informatics*. 2008. doi: 10.1007/978-1-4020-6754-9_13806.

- [53] David S. Wishart. DrugBank. In *Principles of Pharmacogenetics and Pharmacogenomics*. 2012. ISBN 9781139051194. doi: 10.1017/CBO9781139051194.008.
- [54] EMBL-EBI. ChEMBL, 2011.
- [55] Richard Judson, Ann Richard, David Dix, Keith Houck, Fathi Elloumi, Matthew Martin, Tommy Cathey, Thomas R. Transue, Richard Spencer, and Maritja Wolf. ACToR - Aggregated Computational Toxicology Resource. *Toxicology and Applied Pharmacology*, 2008. ISSN 0041008X. doi: 10.1016/j.taap.2007.12.037.
- [56] George Papadatos, Anna Gaulton, Anne Hersey, and John P. Overington. Activity, assay and target data curation and quality in the ChEMBL database. *Journal of Computer-Aided Molecular Design*, 29(9):885–896, 2015. ISSN 15734951. doi: 10.1007/s10822-015-9860-5.
- [57] Maria-Anna Trapotsi. *Development and evaluation of ADME models using proprietary and opensource data By Maria-Anna Trapotsi July 2017*. PhD thesis, 2017.
- [58] Franco Lombardo, Giuliano Berellini, and R. Scott Obach. Trend analysis of a database of intravenous pharmacokinetic parameters in humans for 1352 drug compounds. *Drug Metabolism and Disposition*, 2018. ISSN 1521009X. doi: 10.1124/dmd.118.082966.
- [59] C. W. Yap, Z. R. Li, and Y. Z. Chen. Quantitative structure-pharmacokinetic relationships for drug clearance by using statistical learning methods. *Journal of Molecular Graphics and Modelling*, 24(5):383–395, 2006. ISSN 10933263. doi: 10.1016/j.jmgm.2005.10.004.
- [60] Giuliano Berellini and Franco Lombardo. An Accurate In Vitro Prediction of Human VDss Based on the Øie-Tozer Equation and Primary Physicochemical Descriptors. 3. Analysis and Assessment of Predictivity on a Large Dataset. *Drug metabolism and disposition: the biological fate of chemicals*, 2019. ISSN 1521009X. doi: 10.1124/dmd.119.088914.
- [61] Sangwoo Ryu, Keith Riccardi, Roshan Patel, Larisa Zueva, Woodrow Burchett, and Li Di. Applying Two Orthogonal Methods to Assess Accuracy of Plasma Protein Binding Measurements for Highly Bound Compounds. *Journal of Pharmaceutical Sciences*, 2019. ISSN 15206017. doi: 10.1016/j.xphs.2019.08.004.
- [62] Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saied Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut. A Brief Survey of Text Mining: Classification, Clustering and Extraction Techniques. 2017. URL <http://arxiv.org/abs/1707.02919>.
- [63] Anne Kao and Stephen R. Poteet. *Natural language processing and text mining*. 2007. ISBN 184628175X. doi: 10.1007/978-1-84628-754-1.

- [64] Sophia Ananiadou and John McNaught. Text Mining for Biology and Biomedicine. *Computational Linguistics*, 2006. ISSN 08912017. doi: 10.1162/coli.2007.33.1.135.
- [65] Michael Bada. Mapping of biomedical text to concepts of lexicons, terminologies, and ontologies. *Methods in Molecular Biology*, 2014. ISSN 10643745. doi: 10.1007/978-1-4939-0709-0_3.
- [66] Hugh Griffin. European bioinformatics institute (EBI). *Molecular Biotechnology*, 1995. ISSN 10736085. doi: 10.1007/BF02789113.
- [67] John M. Hancock, Marketa J. Zvelebil, and Marketa J. Zvelebil. UniProt. In *Dictionary of Bioinformatics and Computational Biology*. 2004. doi: 10.1002/9780471650126.dob0721.pub2.
- [68] Scott Federhen. The NCBI Taxonomy database. *Nucleic Acids Research*, 2012. ISSN 03051048. doi: 10.1093/nar/gkr1178.
- [69] Lorraine Tanabe and W. John Wilbur. Tagging gene and protein names in biomedical text. *Bioinformatics*, 2002. ISSN 13674803. doi: 10.1093/bioinformatics/18.8.1124.
- [70] Yoshimasa Tsuruoka and Jun'ichi Tsujii. Improving the performance of dictionary-based approaches in protein name recognition. *Journal of Biomedical Informatics*, 2004. ISSN 15320464. doi: 10.1016/j.jbi.2004.08.003.
- [71] Matthew S. Simpson and Dina Demner-Fushman. Biomedical text mining: A survey of recent progress. In *Mining Text Data*. 2013. ISBN 9781461432234. doi: 10.1007/978-1-4614-3223-4_14.
- [72] Thomas Burr. Pattern Recognition and Machine Learning Pattern Recognition and Machine Learning . Christopher M. Bishop New York : Springer , 2006 . ISBN 0-38731073-8 . xx + 738 pp. 74.95 . *Journal of the American Statistical Association*, 2008. ISSN 01621459. doi: 10.1198/tech.2007.s518.
- [73] Sebastian Ruder. *Neural transfer learning for natural language processing*. PhD thesis, NUI Galway, 2019.
- [74] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, 2013.
- [75] Nlm.nih.gov. MEDLINE®: Description of the Database, 2019.
- [76] Ashraf Haroon. PubMed (<http://www.ncbi.nlm.nih.gov/PubMed>), 1998. ISSN 01406736.

- [77] Donald C. Comeau, Chih Hsuan Wei, Rezarta Islamaj Doğan, Zhiyong Lu, and Jonathan Wren. PMC text mining subset in BioC: About three million full-text articles and growing. *Bioinformatics*, 2019. ISSN 14602059. doi: 10.1093/bioinformatics/btz070.
- [78] Alistair E.W. Johnson, Tom J. Pollard, Lu Shen, Li Wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. MIMIC-III, a freely accessible critical care database. *Scientific Data*, 2016. ISSN 20524463. doi: 10.1038/sdata.2016.35.
- [79] Shawn N. Murphy and Adam Wilcox. Mission and Sustainability of Informatics for Integrating Biology and the Bedside (i2b2). *eGEMs (Generating Evidence & Methods to improve patient outcomes)*, 2014. ISSN 2327-9214. doi: 10.13063/2327-9214.1074.
- [80] Elsevier. Embase®. URL <http://www.elsevier.com/online-tools/embase/>.
- [81] Deborah A Zarin, Tony Tse, Rebecca J Williams, Robert M Califf, and Nicholas C Ide. The clinicaltrials.gov results database—update and key issues. *New England Journal of Medicine*, 364(9):852–860, 2011.
- [82] Olivier Bodenreider. The Unified Medical Language System (UMLS): Integrating biomedical terminology. *Nucleic Acids Research*, 2004. ISSN 03051048. doi: 10.1093/nar/gkh061.
- [83] D. M. Pisanelli, A. Gangemi, and G. Steve. An ontological analysis of the UMLS Metathesaurus. *Proceedings Annual Symposium. AMIA Symposium*, 1998. ISSN 1531605X.
- [84] Alexa T. McCray. UMLS semantic network. In *Proceedings - Annual Symposium on Computer Applications in Medical Care*, 1989.
- [85] Allen C Browne, Alexa T McCray, and Suresh Srinivasan. The specialist lexicon. *National Library of Medicine Technical Reports*, 2000.
- [86] Caroline F. Thorn, Teri E. Klein, and Russ B. Altman. PharmGKB: The pharmacogenomics knowledge base. *Methods in Molecular Biology*, 2013. ISSN 10643745. doi: 10.1007/978-1-62703-435-7_20.
- [87] Byron C Wallace, Joël Kuiper, Aakash Sharma, Mingxi Zhu, and Iain J Marshall. Extracting pico sentences from clinical trial reports using supervised distant supervision. *The Journal of Machine Learning Research*, 17(1):4572–4596, 2016.
- [88] Sampo Pyysalo and Sophia Ananiadou. Anatomical entity mention recognition at literature scale. *Bioinformatics*, 30(6):868–875, 2014.

- [89] Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wieggers, and Zhiyong Lu. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016, 2016.
- [90] Martin Krallinger, Obdulia Rabal, Florian Leitner, Miguel Vazquez, David Salgado, Zhiyong Lu, Robert Leaman, Yanan Lu, Donghong Ji, Daniel M Lowe, et al. The chemdner corpus of chemicals and drugs and its annotation principles. *Journal of cheminformatics*, 7(1):1–17, 2015.
- [91] Sampo Pyysalo, Tomoko Ohta, Rafal Rak, Andrew Rowley, Hong-Woo Chun, Sung-Jae Jung, Sung-Pil Choi, Jun’ichi Tsujii, and Sophia Ananiadou. Overview of the cancer genetics and pathway curation tasks of bionlp shared task 2013. *BMC bioinformatics*, 16(10):1–19, 2015.
- [92] Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. Introduction to the bio-entity recognition task at jnlpba. In *Proceedings of the international joint workshop on natural language processing in biomedicine and its applications*, pages 70–75. Citeseer, 2004.
- [93] Martin Gerner, Goran Nenadic, and Casey M Bergman. Linnaeus: a species name identification system for biomedical literature. *BMC bioinformatics*, 11(1):1–17, 2010.
- [94] Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. Ncbi disease corpus: a resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47:1–10, 2014.
- [95] Evangelos Pafilis, Sune P Frankild, Lucia Fanini, Sarah Faulwetter, Christina Pavloudi, Aikaterini Vasileiadou, Christos Arvanitidis, and Lars Juhl Jensen. The species and organisms resources for fast and accurate identification of taxonomic names in text. *PloS one*, 8(6):e65390, 2013.
- [96] Larry Smith, Lorraine K Tanabe, Rie Johnson nee Ando, Cheng-Ju Kuo, I-Fang Chung, Chun-Nan Hsu, Yu-Shi Lin, Roman Klinger, Christoph M Friedrich, Kuzman Ganchev, et al. Overview of biocreative ii gene mention recognition. *Genome biology*, 9(2):1–19, 2008.
- [97] Dietrich Rebholz Schuhmann, Antonio Jimeno Yepes, Erik van Mulligen, Ning Kang, Jan Kors, David Milward, Peter Corbett, Ekaterina Buyko, Katrin Tomanek, Elena Beisswanger, et al. The calbc silver standard corpus for biomedical named entities—a study in harmonizing the contributions from four independent named entity taggers. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC’10)*, 2010.

- [98] Sunil Mohan and Donghui Li. Medmentions: a large biomedical corpus annotated with umls concepts. *arXiv preprint arXiv:1902.09476*, 2019.
- [99] Martin Krallinger, Obdulia Rabal, Saber A Akhondi, Martín Pérez Pérez, Jesús Santamaría, Gael Pérez Rodríguez, Georgios Tsatsaronis, and Ander Intxaurre. Overview of the biocreative vi chemical-protein interaction track. In *Proceedings of the sixth BioCreative challenge evaluation workshop*, volume 1, pages 141–146, 2017.
- [100] María Herrero-Zazo, Isabel Segura-Bedmar, Paloma Martínez, and Thierry Declerck. The ddi corpus: An annotated corpus with pharmacological substances and drug–drug interactions. *Journal of biomedical informatics*, 46(5):914–920, 2013.
- [101] Àlex Bravo, Janet Piñero, Núria Queralt-Rosinach, Michael Rautschka, and Laura I Furlong. Extraction of relations between genes and diseases from text and large-scale data analysis: implications for translational research. *BMC bioinformatics*, 16(1):1–17, 2015.
- [102] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W Cohen, and Xinghua Lu. Pubmedqa: A dataset for biomedical research question answering. *arXiv preprint arXiv:1909.06146*, 2019.
- [103] Anastasios Nentidis, Konstantinos Bougiatiotis, Anastasia Krithara, and Georgios Paliouras. Results of the seventh edition of the bioasq challenge. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 553–568. Springer, 2019.
- [104] Benjamin Nye, Junyi Jessy Li, Roma Patel, Yinfei Yang, Iain J Marshall, Ani Nenkova, and Byron C Wallace. A corpus with multi-level annotations of patients, interventions and outcomes to support language processing for medical literature. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2018, page 197. NIH Public Access, 2018.
- [105] Gizem Soğancıoğlu, Hakime Öztürk, and Arzucan Özgür. Biosses: a semantic sentence similarity estimation system for the biomedical domain. *Bioinformatics*, 33(14):i49–i58, 2017.
- [106] Simon Baker, Ilona Silins, Yufan Guo, Imran Ali, Johan Högberg, Ulla Stenius, and Anna Korhonen. Automatic semantic classification of scientific literature according to the hallmarks of cancer. *Bioinformatics*, 32(3):432–440, 2016.
- [107] Karin Verspoor, Kevin Bretonnel Cohen, Arrick Lanfranchi, Colin Warner, Helen L Johnson, Christophe Roeder, Jinho D Choi, Christopher Funk, Yuriy Malenkiy, Miriam Eckert, et al. A corpus of full-text journal articles is a robust evaluation tool for revealing differences in performance of biomedical natural language processing tools. *BMC bioinformatics*, 13(1):1–26, 2012.

- [108] J-D Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1):i180–i182, 2003.
- [109] Corpora for Chemical Entity Recognition - Fraunhofer SCAI — [scai.fraunhofer.de. https://www.scai.fraunhofer.de/en/business-research-areas/bioinformatics/downloads/corpora-for-chemical-entity-recognition.html](https://www.scai.fraunhofer.de/en/business-research-areas/bioinformatics/downloads/corpora-for-chemical-entity-recognition.html), 2008. [Accessed 03-Jul-2022].
- [110] Pontus Stenetorp, Sampo Pyysalo, Goran Topic, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. BRAT: A Web-based tool for NLP-Assisted text annotation. In *EACL 2012 - Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 2012. ISBN 9781937284190.
- [111] Matthew Montani, Ines and Honnibal. Prodigy: A new annotation tool for radically efficient machine teaching. *Artificial Intelligence*, to appear, 2018.
- [112] John M Walker. *Biomedical Literature Mining IN Series Editor*. 2015. ISBN 9781493907083.
- [113] Zhiping Wang. Biomedical literature mining for pharmacokinetics numerical parameter collection. 2012.
- [114] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1999*, 1999. ISBN 1581130961. doi: 10.1145/312624.312649.
- [115] Mir S. Siadaty, Jianfen Shu, and William A. Knaus. Relemed: Sentence-level search engine with relevance score for the MEDLINE database of biomedical articles. *BMC Medical Informatics and Decision Making*, 2007. ISSN 14726947. doi: 10.1186/1472-6947-7-1.
- [116] Graham L. Poulter, Daniel L. Rubin, Russ B. Altman, and Cathal Seoighe. MScanner: A classifier for retrieving Medline citations. *BMC Bioinformatics*, 2008. ISSN 14712105. doi: 10.1186/1471-2105-9-108.
- [117] Christian Simon, Kristian Davidsen, Christina Hansen, Emily Seymour, Mike Bogetofte Barnkob, and Lars Rønn Olsen. BioReader: A text mining tool for performing classification of biomedical literature. *BMC Bioinformatics*, 2019. ISSN 14712105. doi: 10.1186/s12859-019-2607-x.
- [118] Ulrike Wittig, Renate Kania, Martin Golebiewski, Maja Rey, Lei Shi, Lenneke Jong, Enkhjargal Alгаа, Andreas Weidemann, Heidrun Sauer-Danzwith, Saqib Mir, et al. Sabio-rk—database for biochemical reaction kinetics. *Nucleic acids research*, 40(D1):D790–D796, 2012.

- [119] Lucy Lu Wang, Oyvind Tafjord, Arman Cohan, Sarthak Jain, Sam Skjonsberg, Carissa Schoenick, Nick Botner, and Waleed Ammar. Supp. ai: finding evidence for supplement-drug interactions. *arXiv preprint arXiv:1909.08135*, 2019.
- [120] K. Humphreys, G. Demetriou, and R. Gaizauskas. Two applications of information extraction to biological science journal articles: enzyme interactions and protein structures. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, 2000. ISSN 2335-6928. doi: 10.1142/9789814447331_0048.
- [121] R. Gaizauskas, G. Demetriou, P. J. Artymiuk, and P. Willett. Protein structures and information extraction from biological texts: The PASTA system. *Bioinformatics*, 2003. ISSN 13674803. doi: 10.1093/bioinformatics/19.1.135.
- [122] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.
- [123] John Lafferty, Andrew McCallum, and Fernando C N Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning*, 2001. ISSN 1750-2799. doi: 10.1038/nprot.2006.61.
- [124] Robert Leaman and Zhiyong Lu. TaggerOne: Joint named entity recognition and normalization with semi-Markov Models. *Bioinformatics*, 2016. ISSN 14602059. doi: 10.1093/bioinformatics/btw343.
- [125] Thanh Hai Dang, Hoang Quynh Le, Trang M. Nguyen, and Sinh T. Vu. D3NER: Biomedical named entity recognition using CRF-biLSTM improved with fine-tuned embeddings of various linguistic information. *Bioinformatics*, 2018. ISSN 14602059. doi: 10.1093/bioinformatics/bty356.
- [126] Donghyeon Kim, Jinhyuk Lee, Chan Ho So, Hwisang Jeon, Minbyul Jeong, Yonghwa Choi, Wonjin Yoon, Mujeeb Sung, and Jaewoo Kang. A neural named entity recognition and multi-type normalization tool for biomedical text mining. *IEEE Access*, 7:73729–73740, 2019.
- [127] Haodi Li, Qingcai Chen, Buzhou Tang, Xiaolong Wang, Hua Xu, Baohua Wang, and Dong Huang. Cnn-based ranking for biomedical entity normalization. *BMC bioinformatics*, 18(11):79–86, 2017.
- [128] Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. Scalable zero-shot entity linking with dense entity retrieval. *arXiv preprint arXiv:1911.03814*, 2019.

- [129] Daniel Loureiro and Alípio Mário Jorge. Medlinker: Medical entity linking with neural representations and dictionary matching. *Advances in Information Retrieval*, 12036:230, 2020.
- [130] Maciej Wiatrak and Juha Iso-Sipilä. Simple hierarchical multi-task neural end-to-end entity linking for biomedical text. In *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis*, pages 12–17, 2020.
- [131] Alan R. Aronson and François Michel Lang. An overview of MetaMap: Historical perspective and recent advances. *Journal of the American Medical Informatics Association*, 2010. ISSN 10675027. doi: 10.1136/jamia.2009.002733.
- [132] Dongseop Kwon, Sun Kim, Chih Hsuan Wei, Robert Leaman, and Zhiyong Lu. EzTag: Tagging biomedical concepts via interactive learning. *Nucleic Acids Research*, 2018. ISSN 13624962. doi: 10.1093/nar/gky428.
- [133] Chih Hsuan Wei, Robert Leaman, and Zhiyong Lu. Beyond accuracy: Creating interoperable and scalable text-mining web services. *Bioinformatics*, 2016. ISSN 14602059. doi: 10.1093/bioinformatics/btv760.
- [134] Chih-Hsuan Wei, Hung-Yu Kao, and Zhiyong Lu. Pubtator: a web-based text mining tool for assisting biocuration. *Nucleic acids research*, 41(W1):W518–W522, 2013.
- [135] Sendong Zhao, Ting Liu, Sicheng Zhao, and Fei Wang. A neural multi-task learning framework to jointly model medical named entity recognition and normalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 817–824, 2019.
- [136] Yaoyun Zhang, Heng Yi Wu, Jun Xu, Jingqi Wang, Ergin Soysal, Lang Li, and Hua Xu. Leveraging syntactic and semantic graph kernels to extract pharmacokinetic drug drug interactions from biomedical literature. *BMC Systems Biology*, 10(Suppl 3), 2016. ISSN 17520509. doi: 10.1186/s12918-016-0311-2. URL <http://dx.doi.org/10.1186/s12918-016-0311-2>.
- [137] Marie-Catherine de Marneffe and Christopher D. Manning. The Stanford typed dependencies representation. 2008. doi: 10.3115/1608858.1608859.
- [138] Bethany Percha and Russ B. Altman. A global network of biomedical relationships derived from text. *Bioinformatics*, 34(15):2614–2624, 2018. ISSN 14602059. doi: 10.1093/bioinformatics/bty114.
- [139] Iz Beltagy, Arman Cohan, and Kyle Lo. SciBERT: Pretrained Contextualized Embeddings for Scientific Text. *ArXiv*, 2019.
- [140] Ashok Thillaisundaram and Theodosia Togia. Biomedical relation extraction with pre-trained language representations and minimal task-specific architecture. *arXiv preprint arXiv:1909.12411*, 2019.

- [141] Sebastian Riedel and Andrew McCallum. Fast and robust joint models for biomedical event extraction. In *EMNLP 2011 - Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2011. ISBN 1937284115.
- [142] Zhongmin Shi, Gabor Melli, Yang Wang, Yudong Liu, Baohua Gu, Mehdi M. Kashani, Anoop Sarkar, and Fred Popowich. Question answering summarization of multiple biomedical documents. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2007. ISBN 9783540726647. doi: 10.1007/978-3-540-72665-4_25.
- [143] Howard Turtle and W. Bruce Croft. Evaluation of an Inference Network-Based Retrieval Model. *ACM Transactions on Information Systems (TOIS)*, 1991. ISSN 15582868. doi: 10.1145/125187.125188.
- [144] Guergana K. Savova, James J. Masanz, Philip V. Ogren, Jiaping Zheng, Sunghwan Sohn, Karin C. Kipper-Schuler, and Christopher G. Chute. Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): Architecture, component evaluation and applications. *Journal of the American Medical Informatics Association*, 2010. ISSN 10675027. doi: 10.1136/jamia.2009.001560.
- [145] Yanshan Wang, Liwei Wang, Majid Rastegar-Mojarad, Sungrim Moon, Feichen Shen, Naveed Afzal, Sijia Liu, Yuqun Zeng, Saeed Mehrabi, Sunghwan Sohn, and Hongfang Liu. Clinical information extraction applications: A literature review. *Journal of Biomedical Informatics*, 77(November 2017):34–49, 2018. ISSN 15320464. doi: 10.1016/j.jbi.2017.11.011.
- [146] Hua Xu, Shane P. Stenner, Son Doan, Kevin B. Johnson, Lemuel R. Waitman, and Joshua C. Denny. MedEx: A medication information extraction system for clinical narratives. *Journal of the American Medical Informatics Association*, 2010. ISSN 10675027. doi: 10.1197/jamia.M3378.
- [147] Hua Xu, Melinda C. Aldrich, Qingxia Chen, Hongfang Liu, Neeraja B. Peterson, Qi Dai, Mia Levy, Anushi Shah, Xue Han, Xiaoyang Ruan, Min Jiang, Ying Li, Jamii St Julien, Jeremy Warner, Carol Friedman, Dan M. Roden, and Joshua C. Denny. Validating drug repurposing signals using electronic health records: A case study of metformin associated with reduced cancer mortality. *Journal of the American Medical Informatics Association*, 2015. ISSN 1527974X. doi: 10.1136/amiajnl-2014-002649.
- [148] Dina Demner-Fushman, James G Mork, Willie J Rogers, Sonya E Shooshan, Laritza Rodriguez, and Alan R Aronson. Finding medication doses in the literature. *Annual Symposium proceedings. AMIA Symposium*, 2018:368–376, 2018. ISSN 1942-597X.

- [149] Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. 2019. doi: 10.18653/v1/w19-5034.
- [150] Yuhao Zhang, Yuhui Zhang, Peng Qi, Christopher D Manning, and Curtis P Langlotz. Biomedical and clinical english model packages for the stanza python nlp library. *Journal of the American Medical Informatics Association*, 28(9):1892–1899, 2021.
- [151] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. 2008. doi: 10.1017/cbo9780511809071.
- [152] Yinan Zhang, Xueqing Liu, and ChengXiang Zhai. Information retrieval evaluation as search simulation: A general formal framework for ir evaluation. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval*, pages 193–200, 2017.
- [153] Wei Chen, Tie-Yan Liu, Yanyan Lan, Zhi-Ming Ma, and Hang Li. Ranking measures and loss functions in learning to rank. *Advances in Neural Information Processing Systems*, 22, 2009.
- [154] María Remedios Marqués-Miñana, Anas Saadeddin, and José Esteban Peris. Population pharmacokinetic analysis of vancomycin in neonates. A new proposal of initial dosage guideline. *British Journal of Clinical Pharmacology*, 2010. ISSN 03065251. doi: 10.1111/j.1365-2125.2010.03736.x.
- [155] Jörg Hakenberg, Sebastian Schmeier, Axel Kowald, Edda Klipp, and Ulf Leser. Finding kinetic parameters using text mining. *OMICS A Journal of Integrative Biology*, 8(2):131–152, 2004. ISSN 15362310. doi: 10.1089/1536231041388366.
- [156] Irena Spasić, Evangelos Simeonidis, Hanan L. Messiha, Norman W. Paton, and Douglas B. Kell. KiPar, a tool for systematic information retrieval regarding parameters for kinetic modelling of yeast metabolic pathways. *Bioinformatics*, 25(11):1404–1411, 2009. ISSN 13674803. doi: 10.1093/bioinformatics/btp175.
- [157] Jyh Jong Tsay, Bo Liang Wu, and Chang Ching Hsieh. Automatic extraction of kinetic information from biochemical literatures. *6th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2009*, 5:28–32, 2009. doi: 10.1109/FSKD.2009.779.
- [158] Karen Hunter. Scienedirect™. *The Serials Librarian*, 33(3-4):287–297, 1998.
- [159] A. Kolchinsky, A. Lourenço, L. Li, and L. M. Rocha. Evaluation of linear classifiers on articles containing pharmacokinetic evidence of drug-drug interactions. *Pacific Symposium on Biocomputing*, pages 409–420, 2013. ISSN 23356936. doi: 10.1142/9789814447973_0040.

- [160] Artemy Kolchinsky, Anália Lourenço, Heng-Yi Wu, Lang Li, and Luis M Rocha. Extraction of pharmacokinetic evidence of drug–drug interactions from the literature. *PloS one*, 10(5):e0122199, 2015.
- [161] Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.
- [162] Fernando Enríquez, José A. Troyano, and Tomás López-Solaz. An approach to the use of word embeddings in an opinion classification task. *Expert Systems with Applications*, 66:1–6, 2016. ISSN 09574174. doi: 10.1016/j.eswa.2016.09.005.
- [163] Jimmy Lin. Scalable language processing algorithms for the masses: A case study in computing word co-occurrence matrices with mapreduce. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 419–428, 2008.
- [164] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems*, 27, 2014.
- [165] Kirk Baker. Singular value decomposition tutorial. *The Ohio State University*, 24, 2005.
- [166] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. Matrix decompositions and latent semantic indexing. *Introduction to information retrieval*, 1, 2008.
- [167] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, pages 1–12, 2013.
- [168] Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.
- [169] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.
- [170] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.
- [171] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.
- [172] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR, 2013.
- [173] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

- [174] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- [175] Kai Xu, Zhanfan Zhou, Tianyong Hao, and Wenyin Liu. A bidirectional lstm and conditional random fields approach to medical named entity recognition. In *International Conference on Advanced Intelligent Systems and Informatics*, pages 355–365. Springer, 2017.
- [176] Bill Yuchen Lin, Frank F Xu, Zhiyi Luo, and Kenny Zhu. Multi-channel bilstm-crf model for emerging named entity recognition in social media. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 160–165, 2017.
- [177] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
- [178] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- [179] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.
- [180] Faiza Khan Khattak, Serena Jebblee, Chloé Pou-Prom, Mohamed Abdalla, Christopher Meaney, and Frank Rudzicz. A survey of word embeddings for clinical text. *Journal of Biomedical Informatics: X*, 4(October):100057, 2019. ISSN 2590177X. doi: 10.1016/j.yjbinx.2019.100057. URL <https://doi.org/10.1016/j.yjbinx.2019.100057>.
- [181] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.
- [182] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
- [183] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *Elements of Statistical Learning 2nd ed.* 2009. ISBN 9780387848570. doi: 10.1007/978-0-387-84858-7.
- [184] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning.* 2013. ISBN 9781461471370.

- [185] Jiancheng Zhong, Yusui Sun, Wei Peng, Minzhu Xie, Jiahong Yang, and Xiwei Tang. XGBFEMF: An XGBoost-Based framework for essential protein prediction. *IEEE Transactions on Nanobioscience*, 2018. ISSN 15361241. doi: 10.1109/TNB.2018.2842219.
- [186] Dahai Zhang, Liyang Qian, Baijin Mao, Can Huang, Bin Huang, and Yulin Si. A Data-Driven Design for Fault Detection of Wind Turbines Using Random Forests and XGboost. *IEEE Access*, 2018. ISSN 21693536. doi: 10.1109/ACCESS.2018.2818678.
- [187] Sukhpreet Singh Dhaliwal, Abdullah Al Nahid, and Robert Abbas. Effective intrusion detection system using XGBoost. *Information (Switzerland)*, 2018. ISSN 20782489. doi: 10.3390/info9070149.
- [188] Alexey Natekin and Alois Knoll. Gradient boosting machines, a tutorial. *Frontiers in Neurorobotics*, 7(DEC), 2013. ISSN 16625218. doi: 10.3389/fnbot.2013.00021.
- [189] Yoav Freund and Robert E. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 1997. ISSN 00220000. doi: 10.1006/jcss.1997.1504.
- [190] Tianqi ; Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM*, pages 785–794, 2016. ISSN 00045772.
- [191] Ferran Gonzalez Hernandez, Simon J Carter, Juha Iso-Sipilä, Paul Goldsmith, Ahmed A Almousa, Silke Gastine, Watjana Lilaonitkul, Frank Kloprogge, and Joseph F Standing. An automated approach to identify scientific publications reporting pharmacokinetic parameters. *Wellcome Open Research*, 6:88, 2021.
- [192] Raheel Nawaz, Paul Thompson, and Sophia Ananiadou. Negated bio-events: Analysis and identification. *BMC Bioinformatics*, 2013. ISSN 14712105. doi: 10.1186/1471-2105-14-14.
- [193] R. Scott Obach, Franco Lombardo, and Nigel J. Waters. Trend analysis of a database of intravenous pharmacokinetic parameters in humans for 670 drug compounds. *Drug Metabolism and Disposition*, 36(7):1385–1405, 2008. ISSN 00909556. doi: 10.1124/dmd.108.020479.
- [194] Tom Crossland, Pontus Stenetorp, Sebastian Riedel, Daisuke Kawata, Thomas D. Kitching, and Rupert A.C. Croft. Towards machine-assisted meta-studies: The Hubble constant. *Monthly Notices of the Royal Astronomical Society*, 13(3):3217–3228, 2020. ISSN 13652966. doi: 10.1093/mnras/stz3400.
- [195] J CARLETTA. Assessing Agreement on Classification Tasks: The Kappa Statistic. *Computational linguistics*, 1996. ISSN 0891-2017.

- [196] Ron Artstein and Massimo Poesio. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596, 2008.
- [197] Achakulvisut Titipat and Daniel Acuna. Pubmed Parser: A Python Parser for PubMed Open-Access XML Subset and MEDLINE XML Dataset, 2015. URL https://github.com/titipata/pubmed_parser.
- [198] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 2017. ISSN 14364646. doi: 10.1007/s10107-016-1030-6.
- [199] Jason M Klusowski. Analyzing cart. *arXiv preprint arXiv:1906.10086*, 2019.
- [200] Noa P. Cruz Diaz and Manuel Maña López. An Analysis of Biomedical Tokenization: Problems and Strategies. 2015. doi: 10.18653/v1/w15-2605.
- [201] Mattew Honnibal and Ines Montani. spaCy2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing., 2017.
- [202] M. F. Porter. An algorithm for suffix stripping, 1980. ISSN 00330337.
- [203] Hubert Soyer, Pontus Stenetorp, and Akiko Aizawa. Leveraging monolingual data for crosslingual compositional word representations. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, 2015.
- [204] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. SPECTER: Document-level Representation Learning using Citation-informed Transformers. 2020. URL <http://arxiv.org/abs/2004.07180>.
- [205] Leila Etaati and Leila Etaati. Azure Databricks. In *Machine Learning with Microsoft Technologies*. 2019. doi: 10.1007/978-1-4842-3658-1_10.
- [206] Ian Fellows. Package wordcloud, 2018.
- [207] Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. *arXiv preprint arXiv:1805.09843*, 2018.
- [208] Chuhan Wu, Fangzhao Wu, Tao Qi, Xiaohui Cui, and Yongfeng Huang. Attentive pooling with learnable norms for text representation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2961–2970, 2020.
- [209] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

- [210] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23, 2021.
- [211] Hicham El Boukkouri, Olivier Ferret, Thomas Lavergne, Hiroshi Noji, Pierre Zweigenbaum, and Junichi Tsujii. Characterbert: Reconciling elmo and bert for word-level open-vocabulary representations from characters. *arXiv preprint arXiv:2010.10392*, 2020.
- [212] Xuan Wang, Yu Zhang, Qi Li, Cathy H Wu, and Jiawei Han. Penner: Pattern-enhanced nested named entity recognition in biomedical literature. In *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 540–547. IEEE, 2018.
- [213] Richard D Boyce, Gregory Gardner, and Henk Harkema. Using natural language processing to extract drug-drug interaction information from package inserts. In *BioNLP: Proceedings of the 2012 Workshop on Biomedical Natural Language Processing*, pages 206–213, 2012.
- [214] Nastassja A Lewinski, Ivan Jimenez, and Bridget T McInnes. An annotated corpus with nanomedicine and pharmacokinetic parameters. *International journal of nanomedicine*, 12:7519, 2017.
- [215] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.
- [216] Chester Palen-Michel, Nolan Holley, and Constantine Lignos. Seqscore: Addressing barriers to reproducible named entity recognition evaluation. *arXiv preprint arXiv:2107.14154*, 2021.
- [217] Nancy Chinchor and Beth M Sundheim. Muc-5 evaluation metrics. In *Fifth Message Understanding Conference (MUC-5): Proceedings of a Conference Held in Baltimore, Maryland, August 25-27, 1993*, 1993.
- [218] Isabel Segura Bedmar, Paloma Martínez, and María Herrero Zazo. Semeval-2013 task 9: Extraction of drug-drug interactions from biomedical texts (ddiextraction 2013). Association for Computational Linguistics, 2013.
- [219] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.
- [220] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz,

- et al. Huggingface’s transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.
- [221] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [222] Meizhi Ju, Makoto Miwa, and Sophia Ananiadou. A neural layered model for nested named entity recognition. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1446–1459, 2018.
- [223] Minh-Quoc Nghiem and Sophia Ananiadou. Aplenty: annotation tool for creating high-quality datasets using active and proactive learning. In *EMNLP (Demonstration)*, pages 108–113, 2018.
- [224] Richard Eckart De Castilho, Jan-Christoph Klie, Naveen Kumar, Beto Boudlosa, and Iryna Gurevych. Linking text and knowledge using the inception annotation platform. In *2018 IEEE 14th International Conference on e-Science (e-Science)*, pages 327–328. IEEE, 2018.
- [225] Bill Yuchen Lin, Dong-Ho Lee, Frank F Xu, Ouyu Lan, and Xiang Ren. Alpacatag: an active learning-based crowd annotation framework for sequence tagging. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, 2019.
- [226] Minh-Quoc Nghiem, Paul Baylis, and Sophia Ananiadou. Paladin: an annotation tool based on active and proactive learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 238–243, 2021.
- [227] Tom Hope, Aida Amini, David Wadden, Madeleine van Zuylen, Sravanthi Parasa, Eric Horvitz, Daniel Weld, Roy Schwartz, and Hannaneh Hajishirzi. Extracting a knowledge base of mechanisms from covid-19 papers. *arXiv preprint arXiv:2010.03824*, 2020.
- [228] Xu Wang, Chen Yang, and Renchu Guan. A comparative study for biomedical named entity recognition. *International Journal of Machine Learning and Cybernetics*, 9(3):373–382, 2018.
- [229] Aditya Siddhant and Zachary C Lipton. Deep bayesian active learning for natural language processing: Results of a large-scale empirical study. *arXiv preprint arXiv:1808.05697*, 2018.
- [230] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 2021.

- [231] Jingbo Zhu, Huizhen Wang, Eduard Hovy, and Matthew Ma. Confidence-based stopping criteria for active learning for data annotation. *ACM Transactions on Speech and Language Processing (TSLP)*, 6(3):1–24, 2010.
- [232] George Hripcsak and Adam S Rothschild. Agreement, the f-measure, and reliability in information retrieval. *Journal of the American medical informatics association*, 12(3):296–298, 2005.
- [233] Louise Deleger, Qi Li, Todd Lingren, Megan Kaiser, Katalin Molnar, et al. Building gold standard corpora for medical natural language processing tasks. In *AMIA Annual Symposium Proceedings*, volume 2012, page 144. American Medical Informatics Association, 2012.
- [234] Thorsten Brants. Inter-annotator agreement for a german newspaper corpus. In *LREC*. Citeseer, 2000.
- [235] Bosheng Song, Fen Li, Yuansheng Liu, and Xiangxiang Zeng. Deep learning methods for biomedical named entity recognition: a survey and qualitative comparison. *Briefings in Bioinformatics*, 22(6):bbab282, 2021.
- [236] Leon Weber, Mario Sanger, Jannes Munchmeyer, Maryam Habibi, Ulf Leser, and Alan Akbik. Hunflair: an easy-to-use tool for state-of-the-art biomedical named entity recognition. *Bioinformatics*, 37(17):2792–2794, 2021.
- [237] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- [238] Markus Eberts and Adrian Ulges. Span-based joint entity and relation extraction with transformer pre-training. *arXiv preprint arXiv:1909.07755*, 2019.
- [239] Yanyao Shen, Hyokun Yun, Zachary C Lipton, Yakov Kronrod, and Animashree Anandkumar. Deep active learning for named entity recognition. *arXiv preprint arXiv:1707.05928*, 2017.
- [240] Dan Shen, Jie Zhang, Jian Su, Guodong Zhou, and Chew Lim Tan. Multi-criteria-based active learning for named entity recognition. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 589–596, 2004.
- [241] Heng-Yi Wu, Chien-Wei Chiang, and Lang Li. Text mining for drug–drug interaction. *Biomedical Literature Mining*, pages 47–75, 2014.
- [242] Ozge Sevgili, Artem Shelmanov, Mikhail Arkhipov, Alexander Panchenko, and Chris Biemann. Neural entity linking: A survey of models based on deep learning. *arXiv preprint arXiv:2006.00575*, 2020.

- [243] Shikhar Vashishth, Rishabh Joshi, Ritam Dutt, Denis Newman-Griffis, and Carolyn Rose. Medtype: Improving medical entity linking with semantic type prediction. *arXiv preprint arXiv:2005.00460*, 2020.
- [244] Rico Angell, Nicholas Monath, Sunil Mohan, Nishant Yadav, and Andrew McCallum. Clustering-based inference for biomedical entity linking. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2598–2608, 2021.
- [245] Dhruv Agarwal, Rico Angell, Nicholas Monath, and Andrew McCallum. Entity linking and discovery via arborescence-based supervised clustering. *arXiv preprint arXiv:2109.01242*, 2021.
- [246] Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. Zero-shot entity linking by reading entity descriptions. *arXiv preprint arXiv:1906.07348*, 2019.
- [247] Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring. *arXiv preprint arXiv:1905.01969*, 2019.
- [248] Ben Hachey, Will Radford, Joel Nothman, Matthew Honnibal, and James R Curran. Evaluating entity linking with wikipedia. *Artificial intelligence*, 194:130–150, 2013.
- [249] Max Berrendorf, Evgeniy Faerman, Laurent Vermue, and Volker Tresp. On the ambiguity of rank-based evaluation of entity alignment or link prediction methods. *arXiv preprint arXiv:2002.06914*, 2020.
- [250] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.
- [251] Kian Ahrabian, Aarash Feizi, Yasmin Salehi, William L Hamilton, and Avishek Joey Bose. Structure aware negative sampling in knowledge graphs. *arXiv preprint arXiv:2009.11355*, 2020.
- [252] Akihiro Hisaka, Mikiko Nakamura, Ayako Tsukihashi, Saori Koh, and Hiroshi Suzuki. Assessment of intestinal availability (fg) of substrate drugs of cytochrome p450s by analyzing changes in pharmacokinetic properties caused by drug–drug interactions. *Drug Metabolism and Disposition*, 42(10):1640–1645, 2014.
- [253] Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. End-to-end neural entity linking. *arXiv preprint arXiv:1808.07699*, 2018.
- [254] Pedro Henrique Martins, Zita Marinho, and André FT Martins. Joint learning of named entity recognition and entity linking. *arXiv preprint arXiv:1907.08243*, 2019.

- [255] Vikas Kumar, Mohammad Faheem, Keun Woo Lee, et al. A decade of machine learning-based predictive models for human pharmacokinetics: Advances and challenges. *Drug discovery today*, 2021.
- [256] Diane R Mould and Richard Neil Upton. Basic concepts in population modeling, simulation, and model-based drug development—part 2: introduction to pharmacokinetic modeling methods. *CPT: pharmacometrics & systems pharmacology*, 2(4):1–14, 2013.
- [257] Veysel Kocaman and David Talby. Biomedical named entity recognition at scale. In *International Conference on Pattern Recognition*, pages 635–646. Springer, 2021.
- [258] Mujeen Sung, Minbyul Jeong, Yonghwa Choi, Donghyeon Kim, Jinhyuk Lee, and Jaewoo Kang. Bern2: an advanced neural biomedical named entity recognition and normalization tool. *arXiv preprint arXiv:2201.02080*, 2022.
- [259] Zexuan Zhong and Danqi Chen. A frustratingly easy approach for entity and relation extraction. *arXiv preprint arXiv:2010.12812*, 2020.
- [260] Bruno Taillé, Vincent Guigue, Geoffrey Scoutheeten, and Patrick Gallinari. Let’s stop incorrect comparisons in end-to-end relation extraction! *arXiv preprint arXiv:2009.10684*, 2020.
- [261] Vikas Yadav and Steven Bethard. A survey on recent advances in named entity recognition from deep learning models. *arXiv preprint arXiv:1910.11470*, 2019.
- [262] Dongxu Zhang and Dong Wang. Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006*, 2015.
- [263] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, 2014.
- [264] Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Systems with Applications*, 114:34–45, 2018.
- [265] Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. A general framework for information extraction using dynamic span graphs. *arXiv preprint arXiv:1904.03296*, 2019.
- [266] Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*, 2021.
- [267] Connor Shorten, Taghi M Khoshgoftaar, and Boriko Furht. Text data augmentation for deep learning. *Journal of big Data*, 8(1):1–34, 2021.

- [268] Sunil Kumar Sahu, Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. Inter-sentence relation extraction with document-level graph convolutional neural network. *arXiv preprint arXiv:1906.04684*, 2019.
- [269] Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. Connecting the dots: Document-level neural relation extraction with edge-oriented graphs. *arXiv preprint arXiv:1909.00228*, 2019.
- [270] Isabelle Augenstein, Sebastian Ruder, and Anders Søgaard. Multi-task learning of pairwise sequence classification tasks over disparate label spaces. *arXiv preprint arXiv:1802.09913*, 2018.
- [271] Victor Sanh, Thomas Wolf, and Sebastian Ruder. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6949–6956, 2019.
- [272] Junchi Zhang, Yue Zhang, Donghong Ji, and Mengchi Liu. Multi-task and multi-view training for end-to-end relation extraction. *Neurocomputing*, 364:245–253, 2019.
- [273] David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. Entity, relation, and event extraction with contextualized span representations. *arXiv preprint arXiv:1909.03546*, 2019.
- [274] Tuan Lai, Heng Ji, and ChengXiang Zhai. Bert might be overkill: A tiny but effective biomedical entity linker based on residual convolutional neural networks. *arXiv preprint arXiv:2109.02237*, 2021.
- [275] Naoki Iinuma, Makoto Miwa, and Yutaka Sasaki. Improving supervised drug-protein relation extraction with distantly supervised models. In *Proceedings of the 21st Workshop on Biomedical Language Processing*, pages 161–170, 2022.
- [276] Dagan O Lonsdale, Emma H Baker, Karin Kipper, Charlotte Barker, Barbara Philips, Andrew Rhodes, Mike Sharland, and Joseph F Standing. Scaling beta-lactam antimicrobial pharmacokinetics from early life to old age. *British Journal of Clinical Pharmacology*, 85(2):316–346, 2019.

Appendices

Appendix A: Guidelines Named Entity Recognition

This appendix includes guidelines developed to annotate PK parameters from sentences. Annotators were asked to base their labelling decisions on these guidelines. As new cases appeared, guidelines were updated accordingly.

Annotation Guidelines PK Named Entity Recognition (NER)

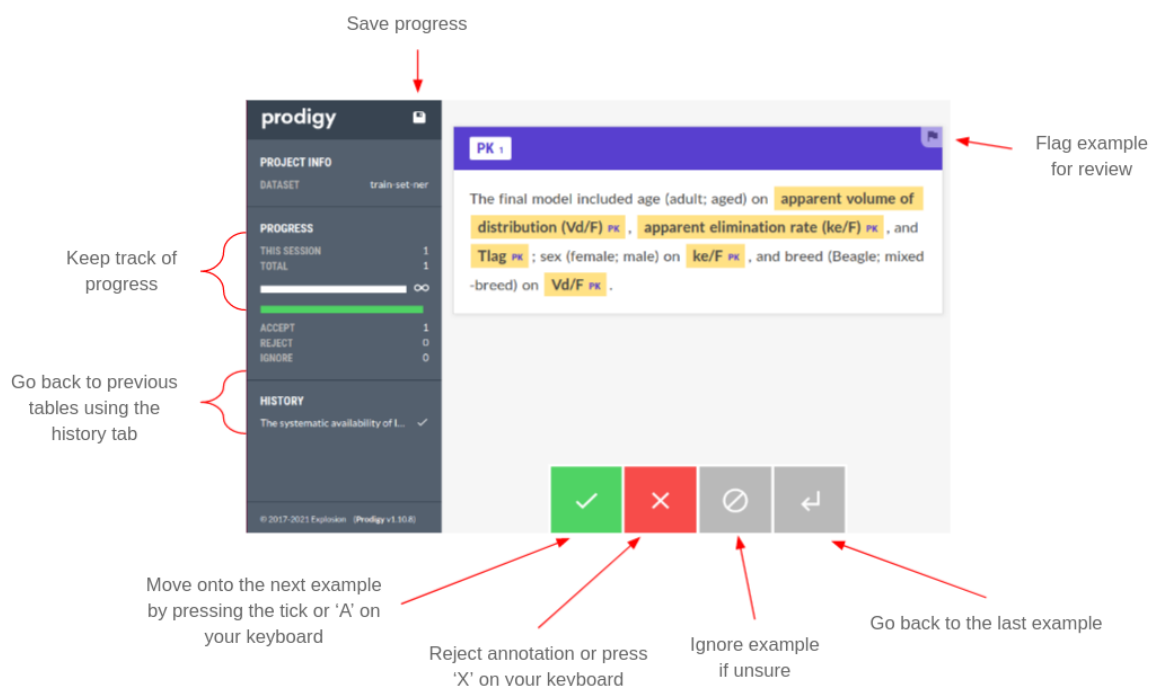
Background

Aim: We aim to develop a named entity recogniser to identify spans of text relating to PK parameter mentions from scientific text.

Method: To do so, we sample sentences within our PK corpus (~120K articles reporting in vivo PK parameters) from the abstract, methods, results and discussion. Then, we ask annotators to label spans of text corresponding to pharmacokinetic parameters.

Task description and interface

The interface displays a single sentence, and the annotator is required to highlight the spans of text relating to PK parameters.



✔ Accept the annotation and move to the following example

⊘ Ignore annotation

🚩 Flag example for review

↔ Go back to the previous example

✗ Reject annotation (only used during active learning)

Questions & Answers

What do we consider “pharmacokinetic parameter” entities? What is included and what is not?

We consider PK entities, those mentions referring to kinetic parameters of pharmacological substances measured either in vivo or in vitro. As a reference, we took the following PK Ontology for the main PK parameters:

<https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-14-35/tables/4>
<https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-14-35/tables/2>

However, bear in mind that other PK parameters might appear that are not included in those tables, and some labelling decisions depend on the end-use of the algorithm. We focus on detecting PK parameters for numerical extraction and aiding the characterisation of DDIs, and most labelling decisions are taken with these applications in mind.

This section will try to cover most doubts that appear during the labelling of PK sentences to help annotators resolve their doubts.

Main Cases

Modifiers

Very often, we find noun **modifiers** describing the type of pharmacokinetic parameter, for instance:

1. “**renal** clearance”
2. “**amoxicillin** half-life”
3. “**mean** Vd”
4. “AUC **ratio**”, etc.

Answer: When the modifier provides information about the specific **type or subtype of the PK parameter** and can be labelled within a single span, we include it as part of the span to

facilitate subsequent entity-linking efforts. For instance, “renal” and “ratio” would be included since they provide information on the type of parameter (rCL and AUCR, respectively). However, “mean” is not considered to be an entity in this application. If we were to develop an end-to-end information extraction system, “mean” would need to be a different entity type (e.g. type of measurement). “Amoxicillin” refers to a drug, so it is best to keep it outside the PK entity if possible.

1. “renal clearance”
2. “amoxicillin half-life”
3. “mean V_d ”
4. “AUC ratio”, etc.

Abbreviations and long-forms

We often see the “full mention” of a pharmacokinetic parameter followed by its abbreviation, for instance:

“The area under the curve from 0 to 24h (AUC0-24h) was calculated for the compound...”
“Oral clearance (CL/F) was 2.3..”

One common doubt is whether to label the complete mention (or long-form) and the abbreviation separately or together.

Answer: Since it is easier to relate a single span to a numerical value or a DDI, we always label the longest span that related to a PK mention

“The area under the curve from 0 to 24h (AUC0-24h) was calculated for the compound...”
“Oral clearance (CL/F) was 2.3..”

Also, on some occasions, we found examples in the following forms:

- “TAN possessed a moderate apparent volume of distribution of the central compartment ($V_c = 4.2 \pm 0.82$ l/kg), rapid clearance (CL = 94 \pm 2 ml/min/kg).”
- “The bioavailability F of compound A was....”

In both cases, we will always try to label the whole span as a single parameter unless it overlaps with other concepts/entities. On other occasions, we find the long-form + abbreviation without parenthesis. In these cases, if possible, we will also label it as a single span:

The efflux transfer constant k_{out} PK was calculated as 0.05 min⁻¹,
equal to t_{1/2,brain} PK of 13.8 min.

The **half-life ($t_{1/2}$) PK** was calculated as $\ln 2/k$ and the **elimination rate constant k PK** was estimated as the negative of the slope from a linear regression of log concentration of time.

Creatinine Clearance

Creatinine clearance is often measured as a covariate in PK studies and appears in multiple sentences.

Answer: Creatinine is an endogenous substance, and therefore the clearance never relates to the clearance of an administered compound. Accordingly, creatinine clearance is **not labelled as a PK mention**.

Same with albumin:

Human serum albumin clearance studies were performed to determine the effect of endogenous MSA on the pharmacokinetic behavior of administered human albumin.

Parameter ratios

Often we find ratios of the same parameters:

PK 1

The mean **ratios of AUC0-last PK** for M1 and M2 compared to alisertib following a single dose of alisertib were 0.45 and 0.41, respectively and the mean **ratios of AUC0-10h PK** for M1 and M2 compared to alisertib following multiple doses of alisertib were 0.44 and 0.42, respectively.

PK 1

The **accumulation ratio (Racc) PK** was calculated as **AUC0-24PM, ss / AUC0-24,sd PK**.

Answer: We will include both parameters of the ratio within the mention. In addition, the term ratio or ratios is also included if present.

On other occasions, we find normalised parameters or different parameters that are divided by each other:

The final model included age (adult; aged) on **apparent volume of distribution (Vd/F_{PK})**, **apparent elimination rate (ke/F_{PK})**, and **$Tlag_{PK}$** ; sex (female; male) on **ke/F_{PK}** , and breed (Beagle; mixed-breed) on **Vd/F_{PK}** . Addition of the covariates to the model explained 78% of the interindividual variability (IIV) in **Vd/F_{PK}** , 36% in **ke/F_{PK}** , and 24% in **$Tlag_{PK}$** , respectively.

In these cases, we include both parameters in the ratio as part of one single span since the whole span is generally the one that is discussed in numerical estimations or DDI context.

Ratios defining other parameters

Quite often, we find mentions like:

“0.6/Ke”

“Ln2/k”

“Ln2/lambda”

These mentions often refer to how a parameter was calculated and are not helpful entities for PK numerical extraction or DDI.

The **half-life ($t_{1/2}$)_{PK}}** of the drug was calculated by use of the relationship $t_{1/2} = 0.693/Kel$.

The **elimination half-life ($T_{1/2}$)_{PK}}** was calculated as $0.693/\lambda_z$.

Terminal elimination $t_{1/2}$ _{PK}} was calculated as $\ln(2)/\lambda_z$.

The **half-life (t_{1/2})_{PK}** was calculated as $\ln 2/k$ and the **elimination rate constant k_{PK}** was estimated as the negative of the slope from a linear regression of log concentration of time.

Clearance (Cl)_{PK} was calculated as $0.693 \times V_d/T_{1/2}$ where **V_d_{PK}** is the **apparent volume of distribution_{PK}** and **T_{1/2}_{PK}** is the **elimination half-life_{PK}**.

Answer: Only if a numerical estimation is related to this coefficient and the whole parameter mention is not present we will consider it as a PK span. Otherwise, we will not label those as PK mentions.

In the examples above, $V_d/T_{1/2}$ were not labelled as parameters either since it is part of the definition of CL calculation.

Equations

We often encounter equations that describe how PK parameters were calculated. For instance:

The following equations were used to predict the xanthotoxol

clearance_{PK} in human [13]: $(2) CL_{int\ in\ vitro} = V_{max}/K_m, CL_{int\ in\ vivo} =$

$CL_{int\ in\ vitro} \cdot SF, CL_H = Q_H \cdot f_u \cdot CL_{int\ in\ vivo} / (Q_H + f_u \cdot CL_{int\ in\ vivo}),$ where

The DDI modeling performance was assessed by comparison of predicted vs. observed victim drug plasma concentration-time profiles during co-administration, DDI **AUC ratios_{PK}** (Eq. (2)), and DDI

C_{max} ratios_{PK} (Eq. (3)): $(2) DDI\ AUC\ ratio = \frac{AUC_{victim\ drug\ during\ co-administration}}{AUC_{victim\ drug}}$ $(3) DDI\ C_{max}\ ratio = \frac{C_{max\ victim\ drug\ during\ co-administration}}{C_{max\ victim\ drug}}$

If rapid achievement of steady-state morphine concentration is desired
, an intravenous loading dose may be calculated using Equation 7 or
simply $DL = C_{target} \cdot V_d$.

Answer: It is often hard to separate the PK terms within the equation, and, in almost all cases, they do not provide valuable information to extract numerical values or characterise DDIs. **Therefore, we will not label parameters within equations.**

Less-common cases

These are rare mentions. However, to ensure consistency across the dataset, make sure the following cases are always labelled as described:

Drug mentions within parameter mentions

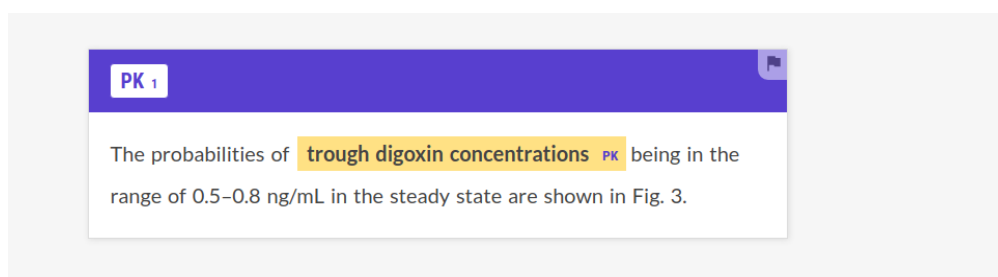
Occasionally, we will find cases in which a drug is mentioned within a PK parameter, e.g.

“The area under the *midazolam* concentration-time curve was....”

In general, we will try to avoid including drugs into PK spans to avoid nested entities, but in this case, we cannot avoid it, so we will label the whole span as PK mention and rely on separate NER models for drugs and PK or models that can deal with overlapping entities:

“The area under the *midazolam* concentration-time curve was....”

Other examples:



On other occasions such as:

“Median plasma AG10-AG t_{max} was 1 hour across all dose levels.”

We might wonder whether to label only t_{max} or with the modifier plasma t_{max}:

1. “Median plasma AG10-AG t_{max} was 1 hour across all dose levels”
2. “Median plasma AG10-AG t_{max} was 1 hour across all dose levels”

Because in this case, including “plasma” would also require including the compound mentioned “AG10-AG”, we will only label the span “tmax” as an entity. The idea is that “plasma” is not an essential term to understand the parameter being discussed, and posterior entity linking systems could take “plasma” into account. So we would select option 1

Modifiers + parameter cue

In some cases, we also encounter the following:

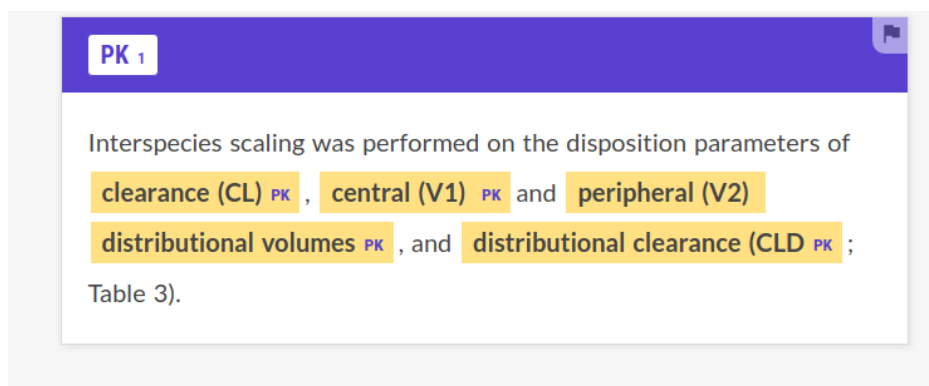
“The **systemic and oral clearance** of midazolam was x and y h/kg, respectively.”

In these cases, we have two parameter mentions that might refer to different numerical estimations. In these cases, we will label in the following manner:

“The **systemic** and **oral clearance** of midazolam was x and y h/kg, respectively.”

Even though “systemic” would not refer to a PK parameter, it does refer to systemic clearance if found alone. Therefore, context-aware entity linking systems could still account for this fact, and each mention could still be related to different values.

Other examples:



The screenshot shows a document snippet with a purple header containing the label "PK 1". The main text reads: "Interspecies scaling was performed on the disposition parameters of clearance (CL) PK, central (V1) PK and peripheral (V2) distributional volumes PK, and distributional clearance (CLD) PK ; Table 3)." The terms "clearance (CL) PK", "central (V1) PK", "peripheral (V2) distributional volumes PK", and "distributional clearance (CLD) PK" are highlighted in yellow.

Central and peripheral

Normalised + parameter mention

Answer: Yes if possible:

“**Normalized CL** was..”

It might help posterior entity linking.

The PopPK model in adults showed a median **AUC_{PK}** estimated at 962 $\mu\text{mol h/L}$ (5th–95th percentiles, 765–1403 $\mu\text{mol h/L}$) and BW **normalized CL_{PK}** and **Vdtotal_{PK}** of 0.10 L/h per kg (0.08–0.13 L/h) and 0.20 L/kg (0.16–0.25 L/kg), respectively.

Serum + parameter mention

Answer: Yes if possible:

“Serum **CL** was..”

It might help posterior entity linking.

In vitro / in vivo + parameter mention

Answer: No need. It is a contextual entity that might need to be labelled as a separate type for specific information extraction applications.

“In vitro **CL** was x..”

“In vivo **t1/2** was y..”

Free/total + parameter mention

Answer: Yes

PK₁

Administration of PF1 at 1 mg/kg in the nondepot formulation resulted in a **total maximal plasma concentration (C_{max})_{PK}** of 331 nM (**free C_{max}_{PK}** = 0.6 nM) at 2 hours postdose.

The predicted **free AUC/MIC** **PK** level was more sensitive to changes in free fraction levels when compared to the **total AUC/MIC** **PK** (Figure6c, right).

Parameters to include

We will often have doubts on whether some parameter mentions are included or not. However, as we mentioned above, any parameter included in the following tables is considered PK according to the PK Ontology.

Table 2:

Experiment types	Parameters	Description	Unit	References
Single Drug Metabolism Experiment	K_m	Michaelis-Menten constant.	mg L ⁻¹	Segel p28.
	V_{max}	Maximum velocity of the enzyme activity.	mg h ⁻¹ mg ⁻¹ protein	Segel p19
	CL_{int}	Intrinsic metabolic clearance is defined as ratio of maximum metabolism rate, V_{max} , and the Michaelis-Menten constant, K_m .	ml h ⁻¹ mg ⁻¹ protein	RT p165
	Metabolic ratio	Parent drug/metabolite concentration ratio	NA	
	$f_{m_{enzyme}}$	Fraction of drug systemically available that is converted to a metabolite through a specific enzyme.	NA	RT xiii
Single Drug Transporter Experiment	P_{app}	The apparent permeability of compounds across the monolayer cells.	cm/sec	Transport Consortium
	R_e	R_e is the ratio of basolateral to apical over apical to basolateral.	NA	Transport Consortium
	Radioactivity	Total radioactivity in plasma and bile samples is measured in a liquid scintillation counter	dpm/mg protein	Transport Consortium
	Uptake Volume	The amount of radioactivity associated with the cells divided by its concentration in the incubation medium.	ul/mg protein	Transport Consortium
Drug Interaction Experiment	IC_{50}	Inhibitor concentration that inhibits to 50% of enzyme activity.	mg L ⁻¹	
	K_i	Inhibition rate constant for competitive inhibition, noncompetitive inhibition, and uncompetitive inhibition.	mg L ⁻¹	Segel p103
	K_{deg}	The natural degradation rate constant for the Enzyme.	h ⁻¹	Rostami-Hodjegan and Tucker
	K_i	The concentration of inhibitor associated with half maximal Inactivation in the mechanism based inhibition.	mg L ⁻¹	Rostami-Hodjegan and Tucker
	K_{inact}	The maximum degradation rate constant in the presence of a high concentration of inhibitor in the mechanism based inhibition.	h ⁻¹	Rostami-Hodjegan and Tucker
	E_{max}	Maximum induction rate	Unit free	Rostami-Hodjegan and Tucker
	EC_{50}	The concentration of inducer that is associated with the half maximal induction.	mg L ⁻¹	Rostami-Hodjegan and Tucker

Table 4:

Name	Description	Unit	reference
AUC _{inf}	Area under the drug concentration time curve.	mg h L ⁻¹	RT p37
AUC _{SS}	Area under the drug concentration time curve within a dosing curve at steady state.	mg h L ⁻¹	RT pxi
AUC _t	Area under the drug concentration time curve from time 0 to t.	mg h L ⁻¹	RT p37
AUMC	Area under the first moment of concentration versus time curve.	mg ² h L ⁻²	RT p486
AUCR	AUC ratio (drug interaction parameter).	Unit free	
CL	Total clearance is defined as the proportionality factor relating rate of drug elimination to the plasma drug concentration.	ml h ⁻¹	RT p23
CL _b	Blood clearance is defined as the proportionality factor relating rate of drug elimination to the blood drug concentration.	ml h ⁻¹	RT p160
CL _u	Unbound clearance is defined as the proportionality factor relating rate of drug elimination to the unbound plasma drug concentration.	ml h ⁻¹	RT p163
CL _H	Hepatic portion of the total clearance.	ml h ⁻¹	RT p161
CL _R	Renal portion of the total clearance.	ml h ⁻¹	RT p161
CL _{po}	Total clearance of drug following an oral dose.	ml h ⁻¹	
CL _{IV}	Total clearance of drug following an IV dose.	ml h ⁻¹	
CL _{int}	Intrinsic metabolic clearance is defined as ratio of maximum metabolism rate, V _{max} , and the Michaelis-Menten constant, K _m .	ml h ⁻¹	RT p165
CL ₁₂	Inter-compartment distribution between the central compartment and the peripheral compartment.	ml h ⁻¹	
CL ratio	Ratio of the clearance (drug interaction parameter).	Unit free	
C _{max}	Highest drug concentration observed in plasma following administration of an extravascular dose.	mg L ⁻¹	RT pxii
C _{max} ratio	The ratio of C _{max} (drug interaction parameter).	Unit free	
C _{SS}	Concentration of drug in plasma at steady state during a constant rate intravenous infusion.	mg L ⁻¹	RT pxii
C _{SS} ratio	The ratio of C _{SS} (drug interaction parameter).	Unit free	
E	Extraction ratio is defined as the ratio between blood clearance, CL _b , and the blood flow.	Unit free	RT p159
E _H	Hepatic extraction ratio.	Unit free	RT p161
F	Bioavailability is defined as the proportion of the drug reaches the systemic blood.	Unit free	RT p42
F _G	Gut-wall bioavailability.	Unit free	
F _H	Hepatic bioavailability.	Unit free	RT p167
F _R	Renal bioavailability.	Unit free	RT p170
f _e	Fraction of drug systemically available that is excreted unchanged in urine.	Unit free	RT pxiii
f _m	Fraction of drug systemically available that is converted to a metabolite.	Unit free	RT pxiii
f _u	Ratio of unbound and total drug concentrations in plasma.	Unit free	RT pxiii
k	Elimination rate constant.	h ⁻¹	RT pxiii
K ₁₂ , k ₂₁	Distribution rate constants between central compartment and peripheral compartment.	h ⁻¹	
k _a	Absorption rate constant.	h ⁻¹	RT pxiii
k _e	Urinary excretion rate constant.	h ⁻¹	RT pxiii
k _m	Rate constant for the elimination of a metabolite.	h ⁻¹	RT pxiii
K _m	Michaelis-Menten constant.	mg L ⁻¹	RT pxiii
MRT	Mean time a molecular resides in body.	h	RT pxiv
Q	Blood flow.	L h ⁻¹	RT pxiv
Q _H	Hepatic blood flow.	L h ⁻¹	RT pxiv
t _{max}	Time at which the highest drug concentration occurs following administration of an extravascular dose.	h	RT pxiv
t _{1/2}	Half-life of the drug disposition.	h	RT pxiv
t _{1/2} ratio	Half-life ratio (drug interaction parameter).	Unit free	
t _{1/2,α}	Half-life of the fast phase drug disposition.	h	
t _{1/2,β}	Half-life of the slow phase drug disposition.	h	
V	Volume of distribution based on drug concentration in plasma.	L	RT pxiv
V _b	Volume of distribution based on drug concentration in blood.	L	RT pxiv
V ₁	Volume of distribution of the central compartment.	L	RT pxiv
V ₂	Volume of distribution of the peripheral compartment.	L	
V _{SS}	Volume of distribution under the steady state concentration.	L	RT pxiv
V _{max}	Maximum rate of metabolism by an enzymatically mediated reaction.	mg h ⁻¹	RT pxiv
λ ₁ , λ ₂	Disposition rate constants in a two-compartment model.	h ⁻¹	GP p84

Apart from those, other parameters also appeared. Here, we will cover most cases that we have come across:

MIC

Answer: MIC alone is not considered as a PK parameter

- “The MIC of MDZ was...”

PK/MIC

We often encounter ratios of PK/PD parameters. For instance:

- “The AUC/MIC was...”
- “The CL/MIC was...”

Answer: We do consider this whole span, as PK mentions

- “The AUC/MIC was...”
- “The CL/MIC was...”

PK 1

The in vitro MIC and MBC data were integrated with in vivo PK data to determine the PK/PD indices such as AUC_{0-24}/MIC_{PK} , AUC_{0-24}/MBC_{PK} , C_{max}/MIC_{PK} , C_{max}/MBC_{PK} , $T > MIC$, and $AUC_{0-24_{PK}} > MIC$, which are presented in Table 5.

PK 1

Moxifloxacin regimens produced a range of area under the concentration-time curve (AUC)/MIC ratios $_{PK}$ ranging from 9.2 to 444 and peak/MIC ratios $_{PK}$ ranging from 1.3 to 102.

Bioavailability

Answer: Always include with modifier if present. Careful with the abbreviation F, since it can sometimes refer to other concepts:

Both values of F were less than $F_{0.05(2,2)}$, with the value 19.00 indicating that the regression equations were not statistically significant, and the arranged ethosome compositions had no significant effect on EE or DSD.

Figure 2E,F illustrates the intestinal stability of PMX53 and PMX205 when incubated at 37 °C for up to 60 min.

Test conditions were the same as described in the “Fluorescence properties of compound F” section.

Absorption

Absorption is mentioned very frequently as a general property of chemical compounds. For instance:

“The **absorption** of drug X (F=45%) increased when co-administered with drug Y (F=68%)

However, it often does not refer to numerical values of parameters but is derived from other parameters (most often **bioavailability (F)**).

Answer: We will label it as PK if the mention refers to a kinetic parameter (e.g. absorption rate) but not if it refers to a general property. Therefore in the example above, absorption would not be labelled as PK, but F would:

“The absorption of drug X (F=45%) increased when co-administered with drug Y (F=68%)

Overall, systemic absorption of single oral doses of HC-ER 20 mg was comparable in the fed and fasted states.

Exposure

Like absorption, sometimes we find sentences describing the “exposure” of a drug, often measured through the AUC or Cmax.

We will not label exposure as a parameter but the actual parameter used to describe the exposure (often AUC).

Exposure (Cmax PK and AUC0-24 PK) did not appear to increase dose proportionally for the 100-600 mg dose levels, but conclusions were limited by the small patient numbers.

Concentrations

Multiple concentrations are mentioned across PK literature. However, here we will **include**:

- Cmin
- Cmax
- Cthrough
- Cavg
- C_{ss}
- Tissue-to-blood concentration ratio
- X-to-Y concentration ratio

We will **not include**:

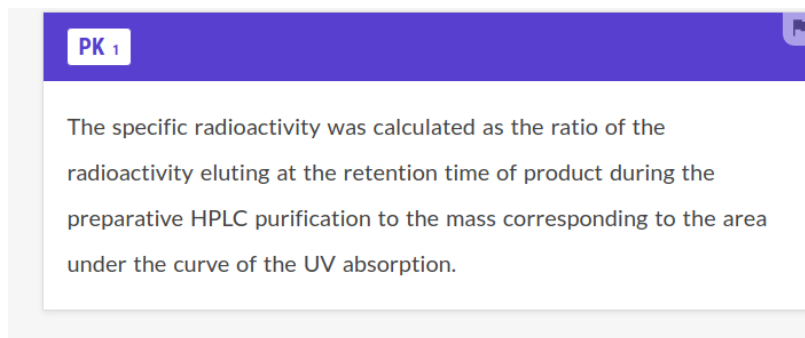
- Clast
- C0
- Minimum quantifiable C
- Cin
- Cout
- Cp (concentration in compartment x)
- MIC
- Other C

Plasma concentration-time curve

Not as such.

AUC

Most often refers to the area under the concentration-time curve. Then we would label it as a PK parameter. However, on some occasions, it refers to other, non-PK relevant curves (e.g. chromatography). In those cases, they will not be labelled:



The area under the receiver operator characteristic curve (AUC) was used to determine the predictive value of on-entry CCI for Δ CCIAKI and on-entry pCr for Δ pCrAKI.

AUMC

The area under the moment curve: Yes, always include

AUEC

The area under the effect curve -> No, considered PD parameter.

IC50

Yes, always → present in Table 2

Bmax

No, this refers to the number of binding sites

Dissociation and association constants

No, physicochemical properties

Permeability and radioactivity

We found many mentions across the literature. Mainly:

- Apparent permeability (P_{app}), Ratio of the basolateral to apical permeability and Apical to basolateral permeability (R_e) Radioactivity, permeability

These mentions are present in table 2. These are less relevant for this application since they are in vitro parameters, but we will label them if present.

Kdeg

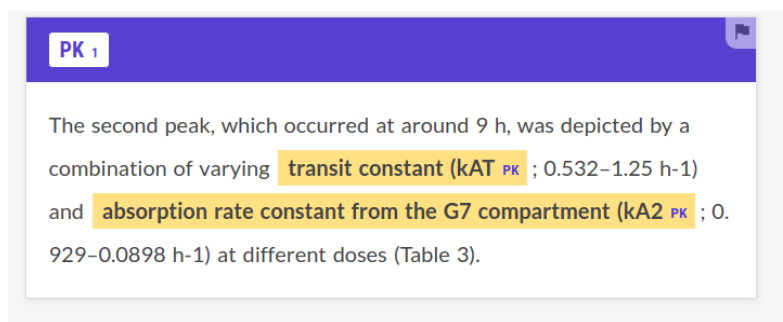
Yes → Table 2

KI

Yes → Table 2

Transit constant

Yes, if mentioned in PK context



Similarly, the parameters for disposition and gastrointestinal transit absorption process ($k_{A\ PK}$, $k_{A2\ PK}$, $k_{AT\ PK}$, and F_{PK}) were obtained by simultaneously fitting the IV and oral data from dogs with Eqs 4–9 (Fig 2B).

Mean residence time

Yes

Retention times

Not if mentioned in the context of chromatography. On some occasions, retention is used interchangeably with residence:

Some pharmacokinetic parameters, such as the **area under the curve (AUC) _{PK}**, the **mean retention time (MRT) _{PK}** and the **elimination half-life (t_{1/2}) _{PK}** of 5-FA-PAE were much higher than

Formation constant

No, this is often a physicochemical property

Solubility

No, this is often a physicochemical property.

R_f

Very rare, but it has eventually been referred to as Retention Fraction. Not considered PK entity.

Hill coefficient

Yes

K_{app}

Depending on the context, it can refer to different concepts. If PK parameter is in vivo or in vitro, yes. It is not a frequent mention but if it appears, check the context.

Dose

No, not considered PK parameter

MTD

No, considered PD property

Perfusion rate

No, most often not reported in PK context

Emax

Yes, included in table 2

I_{max} (maximum Inhibition)

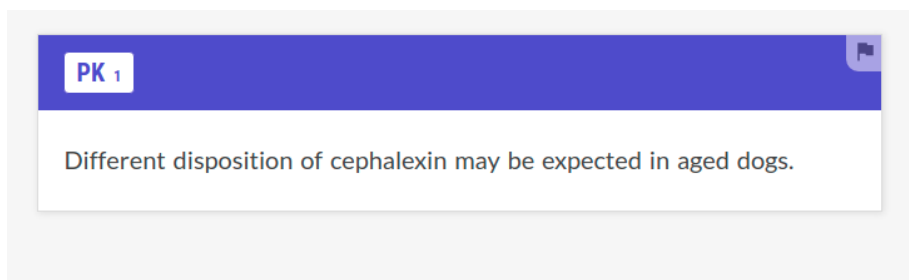
Yes, to be consistent with labelling E_{max}

IC₅₀

Yes

Disposition constants

Disposition rate constants are considered PK parameters. However, when discussing the general concept of disposition, we do not label it as PK, for instance:



A₀

Yes

AE

In general, it refers to adverse effects, so we do not label it as PK. However, it can occasionally refer to PK parameters:

Cumulative amount of drug excreted unchanged in urine (A_e)^{PK},
fraction of dose excreted unchanged into urine (f_e)^{PK}, and renal
clearance of drug (CLR)^{PK} were estimated for each subject with
urinary excretion data.

Bleeding rate

No, endogenous and not related to compound.

Peak areas

Often found in chromatography studies but not relevant as a PK parameter:

Peak areas in the chromatograms for the spiked plasma samples containing the above lowest concentrations were compared with the signal-to noise ratio ≥ 10 .

Flow rate

In general, no, unless is the blood flow rate (which is included in table 4)

Acetonitrile and water containing 0.1% (v/v) formic acid was adopted as the mobile phase, the flow rate was 0.3 mL/min.

However, the mobile phase was a mixture of acetonitrile/water/tert-butyl methyl ether/phosphoric acid (525/425/50/1, v/v/v/v) at a flow rate of 1.5 mL/minute for in vivo determination.¹⁶

C_{in} C_{out}

Influent and effluent drug concentrations. Not relevant, often reported for in silico models.

Glomerular filtration rate

No, often mentioned as a covariate

Diffusion coefficient

No

Bound/unbound fraction

Yes

Times

T_{max} and t_{lag} yes, others no (e.g. t_0 , t_{last}). We can apply the same rules as for the concentrations.

Transfer constants between compartments

Yes

K_{cp} , K_{pc} , K_{cb} and K_{bc} represent first-order transfer constants connecting the various compartments.

Biotransformation

No

Appendix B: Named Entity Recognition patterns

This appendix shows the list of patterns included to recognise PK parameters through rules. These patterns were implemented within a spaCy model¹. If more than one pattern generated overlapping spans, the longest span was selected.

```
1 {"label": "PK", "pattern": [{"DEP": "amod"}, {"LOWER": {"IN": ["cl",  
    "cl/f", "clz/f", "clearance", "clearances", "seroclearance"]  
    ]}}]}  
2 {"label": "PK", "pattern": [{"LOWER": {"IN": ["cl", "crcl", "cl/f",  
    "clz/f", "clearance", "clearances", "seroclearance"]}}]}  
3 {"label": "PK", "pattern": [{"LOWER": "compartmental"}, {"LOWER":  
    "clearance"}]}  
4 {"label": "PK", "pattern": [{"LOWER": "renal"}, {"LOWER": "  
    clearance"}]}  
5 {"label": "PK", "pattern": [{"LOWER": "non-renal"}, {"LOWER": "  
    clearance"}]}  
6 {"label": "PK", "pattern": [{"LOWER": "elimination"}, {"LOWER": "  
    rate"}, {"LOWER": "constant", "OP": "?"}]}  
7 {"label": "PK", "pattern": [{"LOWER": {"IN": ["auc", "auc/mic", "  
    aucbrain", "auctissue", "aucbrain/aucplasma", "aucplasma", "  
    aucinf"]}}]}  
8 {"label": "PK", "pattern": [{"DEP": "amod"}, {"LOWER": {"IN": ["auc  
    ", "auc/mic", "aucbrain", "auctissue", "aucbrain/aucplasma", "  
    aucplasma", "aucinf"]}}]}  
9 {"label": "PK", "pattern": [{"LOWER": {"REGEX": "(?:auc)(?:(  
    (?!\s)\w|\d)\S+|\b)"}}, {"text": ""}, {"OP": "?"}, {"LOWER": "  
    :ratio", "OP": "?"}]}  
10 {"label": "PK", "pattern": [{"LOWER": "area"}, {"LOWER": "under"},  
    {"LOWER": "the", "OP": "?"}, {"LOWER": "plasma", "OP": "?"}, {"  
    LOWER": "concentration", "OP": "?"}, {"LOWER": "concentration-  
    time", "OP": "?"}, {"LOWER": "profile", "OP": "?"}, {"LOWER": "  
    curve", "OP": "?"}]}  
11 {"label": "PK", "pattern": [{"LOWER": {"IN": ["vd", "vss", "vz", "  
    clz/f", "vz/f", "v/f", "v(d)/f"]}}]}  
12 {"label": "PK", "pattern": [{"DEP": "amod"}, {"LOWER": {"IN": ["vd"  
    ", "vss", "vz", "clz/f", "vz/f", "v/f", "v(d)/f"]}}]}  

```

¹ For details on spaCy patterns see <https://spacy.io/usage/rule-based-matching>.

```

13 {"label": "PK", "pattern": [{"LOWER": "volume"}, {"LOWER": "of"}, {"
    "LOWER": "distribution"}, {"LOWER": "at", "OP": "?"}, {"LOWER":
    "steady", "OP": "?"}, {"LOWER": "state", "OP": "?"}]}
14 {"label": "PK", "pattern": [{"DEP": "amod"}, {"LOWER": "volume"}, {"
    "LOWER": "of"}, {"LOWER": "distribution"}, {"LOWER": "at", "OP"
    : "?"}, {"LOWER": "steady", "OP": "?"}, {"LOWER": "state", "OP":
    "?"}]}
15 {"label": "PK", "pattern": [{"LOWER": {"IN": ["bioavailabilities"
    , "bioavailability", "f", "fd", "fc"]}}]}
16 {"label": "PK", "pattern": [{"DEP": "amod"}, {"LOWER": {"IN": ["
    bioavailabilities", "bioavailability", "f", "fd", "fc"]}}]}
17 {"label": "PK", "pattern": [{"LOWER": {"IN": ["halflife", "half-
    life", "halflives", "halftime", "t1/2"]}}]}
18 {"label": "PK", "pattern": [{"DEP": "amod"}, {"LOWER": {"IN": ["
    halflife", "half-life", "halflives", "halftime", "t1/2"]}}]}
19 {"label": "PK", "pattern": [{"LOWER": {"IN": ["cmax", "cmin"]}}]}
20 {"label": "PK", "pattern": [{"LOWER": {"IN": ["c(max", "c(min)"]}},
    {"text": ")"}, {"OP": "?"}]}
21 {"label": "PK", "pattern": [{"DEP": "amod"}, {"LOWER": {"IN": ["
    cmax", "cmin"]}}]}
22 {"label": "PK", "pattern": [{"DEP": "amod"}, {"LOWER": {"IN": ["c(
    max", "c(min)"]}}, {"text": ")"}, {"OP": "?"}]}
23 {"label": "PK", "pattern": [{"LOWER": "peak"}, {"LOWER": "plasma",
    "OP": "?"}, {"LOWER": "concentration"}]}
24 {"label": "PK", "pattern": [{"LOWER": "time", "OP": "?"}, {"LOWER":
    "to", "OP": "?"}, {"LOWER": "maximum"}, {"LOWER": "
    concentration"}]}
25 {"label": "PK", "pattern": [{"LOWER": "time", "OP": "?"}, {"LOWER":
    "to", "OP": "?"}, {"LOWER": "maximum"}, {"LOWER": "plasma"}, {"
    LOWER": "concentration"}]}
26 {"label": "PK", "pattern": [{"LOWER": "mean"}, {"LOWER": "
    residence"}, {"LOWER": "time"}]}
27 {"label": "PK", "pattern": [{"LOWER": "mrt"}]}
28 {"label": "PK", "pattern": [{"LOWER": "fraction"}, {"LOWER": "
    unbound"}]}
29 {"label": "PK", "pattern": [{"LOWER": "free"}, {"LOWER": "fraction
    "}]}}
30 {"label": "PK", "pattern": [{"LOWER": "absorbtion"}, {"LOWER": "
    rate"}]}
31 {"label": "PK", "pattern": [{"DEP": "amod"}, {"LOWER": "absorbtion
    "}, {"LOWER": "rate"}]}

```

```

32 {"label":"PK","pattern":[{"LOWER":{"IN":["kel","ka","tmin","tss","fe","krel","tlag","kpc","kc","kf","kelp","k\u03b2","pmax","\u03b8z","fue","ko","kobs","tlc","vm","absorbencies","ku","kl","vmax","vs","kout","mru","fu","k12","k21","tmax","cl","cl12","q","ke","keff","kmax","biodisponibility","qh","q","e","eh","v1","v2","\u03bb","\u03bbz"]}}]}
33 {"label":"PK","pattern":[{"DEP":"amod"},{"LOWER":{"IN":["kel","ka","tmin","tss","fe","krel","tlag","kpc","kc","kf","kelp","k\u03b2","pmax","\u03b8z","fue","ko","kobs","tlc","vm","absorbencies","ku","kl","vmax","vs","kout","mru","fu","k12","k21","tmax","cl","cl12","q","ke","keff","kmax","biodisponibility","qh","q","e","eh","v1","v2","\u03bb","\u03bbz"]}}]}

```

Appendix C: Guidelines Relation Extraction

This appendix includes guidelines developed to annotate entities and relations of PK estimations from scientific sentences. Annotators were asked to base their labelling decisions on these guidelines. As new cases appeared, guidelines were updated accordingly.

Annotation Guidelines PK Relation Extraction

Background

Aim: We aim to develop a relation extraction model to retrieve numerical estimations of PK parameter values from scientific articles. To do so, we focus on relations existing at the sentence level between multiple entities. In this annotation task, we will focus on information around the PK numerical estimations and future studies will complement numerical estimations with contextual data e.g. drugs, diseases, conditions, covariates etc. In this annotation task, we aim to collect annotation data to train a relation extraction algorithm capable of (1) detecting all the relevant **entities** involved (e.g. PK mentions, numerical values, units, etc.) and (2) their **relations** in scientific text.

Method: To do so, we sample sentences from the abstract, methods, results and discussion sections within our corpus of PK articles. Then, annotators will be asked to label spans of text corresponding to 5 entities and their relations.

Entities and relations

In this annotation task, PK experts will have to annotate **(1) entities** and **(2) relations** for every sentence displayed in the interface. In this section, we describe the types of entities and relations involved.

Entities

In natural language processing (NLP) entities are **spans of text** that correspond to specific **concepts**. For instance, mentions of organisations, persons, countries etc. In this task, we are interested in **five** entities:

1. PK

Mentions of pharmacokinetic (PK) parameters. This entity refers to spans of text that mention kinetic parameters. Any type of kinetic parameter in the context of PKs will be highlighted. For a list of PK, parameter types see the following tables: [in vivo](#), [in vitro](#). If not sure whether a specific mention should be labelled see but see the Questions & Answers section. Example:

The median renal CL of midazolam was higher than 3.0 mL / min

TO ANNOTATE: PK parameters are **pre-highlighted** in the interface by our model. But, there will be cases in which the model **missed** those mentions or **incorrectly predicted** them. Therefore, we need to **check** that PK mentions are well annotated and **correct any mistakes**.

2. UNITS

Spans of text corresponding to units of numerical PK estimations. Example:

The median renal CL of midazolam was higher than 3.0 mL/min
UNITS

TO ANNOTATE: In the initial annotation rounds, **UNITS will not be pre-highlighted**. This means that in every example the user needs to **look for any units** and highlight their spans. This is the entity that requires the **most attention** from the annotator. After the initial annotation rounds, UNITS will be pre-highlighted.

3. VALUE

Spans of text that refer to numerical values. This includes single numbers, decimals, exponential expressions etc. Example:

The median renal CL of midazolam was higher than 3.0 mL / min
VALUE

TO ANNOTATE: For this, we use a set of **rules** to **pre-highlight** values in the interface. In general, the rules work well and tend to encapsulate numerical values. However, there might be eventual mistakes and the user **might need to correct** them if that happens.

4. RANGE

Two numerical values defining a range. On some occasions, numerical estimations might be expressed in the form of ranges:

The renal CL of midazolam ranged from 3.0 to 5.3 mL / min
RANGE

TO ANNOTATE: We **pre-highlight** those terms using regular expressions. However, the rules are still quite limited and new cases are likely to appear. This means that the user will need to **pay attention** to potential **new RANGE** spans.

5. COMPARE

Comparative terms. The mentions of this entity aim to provide information on whether specific PK estimations refer to the maximum or minimum estimated value. In the literature one might find:

The median renal CL of midazolam was higher than 3.0 mL / min

The COMPARE entity will help us to acknowledge that 3.0 is the minimum value. Some COMPARE terms include: >, <, higher, lower, maximum, minimum, exceeded etc. In essence, COMPARE will give information on whether the extracted number refers to an **estimated boundary** (e.g. max, min, >, <, exceeded). Mentions like **“approximately”, “~”, “close to” are not considered COMPARE** mentions since they do not indicate whether the value is a minimum or maximum but the confidence of the prediction.

TO ANNOTATE: We **pre-highlight** those terms using an in-house dictionary. The dictionary is still quite limited and new cases are likely to appear. This means that the user will need to **pay attention** to potential **new COMPARE** mentions.

Relations

Once entities are annotated, the next step is to annotate **relations between entities**. Relations always need to happen between entities and **some relations only happen between specific types of entities**.

In this task we consider **three** relations:

1. C_VAL

Central Value. This is a relation between a PK parameter mention and its estimated value. This type of relation only happens between the following entity types:

PK → VALUE

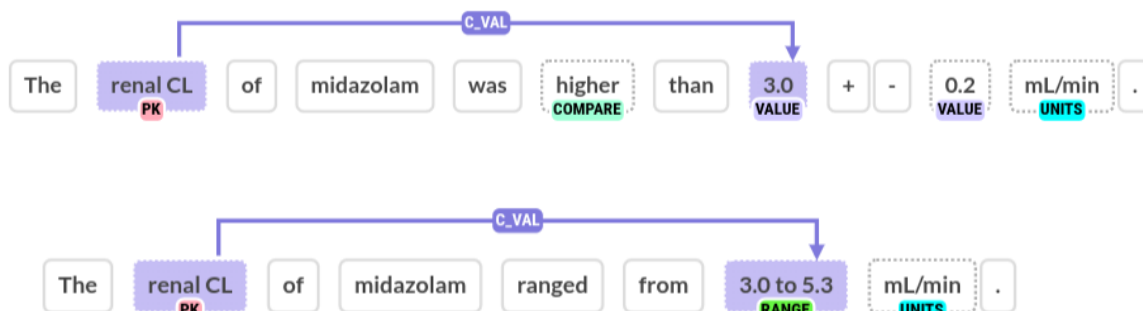
PK → RANGE

It's only annotated when the VALUE/RANGE refers to a measurement but NOT A COMPARISON. For instance, there are many comparative sentences that mention:

*“The **CL** of midazolam increased by 3% when co-administered with amoxicillin”*

In this case, there is **no C_VAL** relation between CL and 3 since 3 is not a numerical estimate of CL.

Examples:



2. D_VAL

Deviation Value. This relation happens between central measurements and their deviation values/ranges. This relation is **only annotated if a C_VAL relation exists** and between deviation values/ranges and central values/ranges. So, this type of relationship only happens between the following entity types:

- VALUE → VALUE (previously labelled with C_VAL relation)
- VALUE → RANGE (previously labelled with C_VAL relation)
- RANGE → VALUE (previously labelled with C_VAL relation)
- RANGE → RANGE (previously labelled with C_VAL relation)

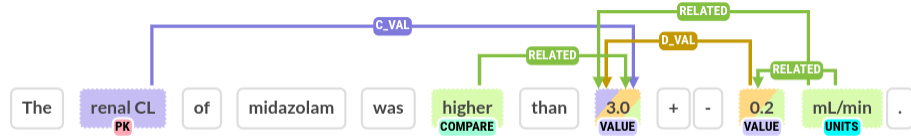
Example:



3. RELATED

This type of relation happens between multiple entity types and serves for adding any complementary information for the central or deviation values:

- COMPARE → VALUE/RANGE (previously labelled with C_VAL or D_VAL)
- UNITS → VALUE/RANGE (most common)



Example:

NOTE: We will **only** annotate relations between units and values/ranges if the **VALUE/RANGE** is part of a **C_VAL** or **D_VAL** relation.

Task description and interface

The interface displays a single sentence with some entities pre-highlighted by the NER model. There are several options available on the interface:

Instructions can be found by clicking the ? button

Save progress

Annotate relations

Annotate entities

Wrap text across lines

Erase entities

Flag examples

Keep track of progress

Go back

Leave comments example

Move onto the next example by pressing the tick or 'A' on your keyboard

Go back to the last example

Article URL

[Click here for a video tutorial on the interface](#)

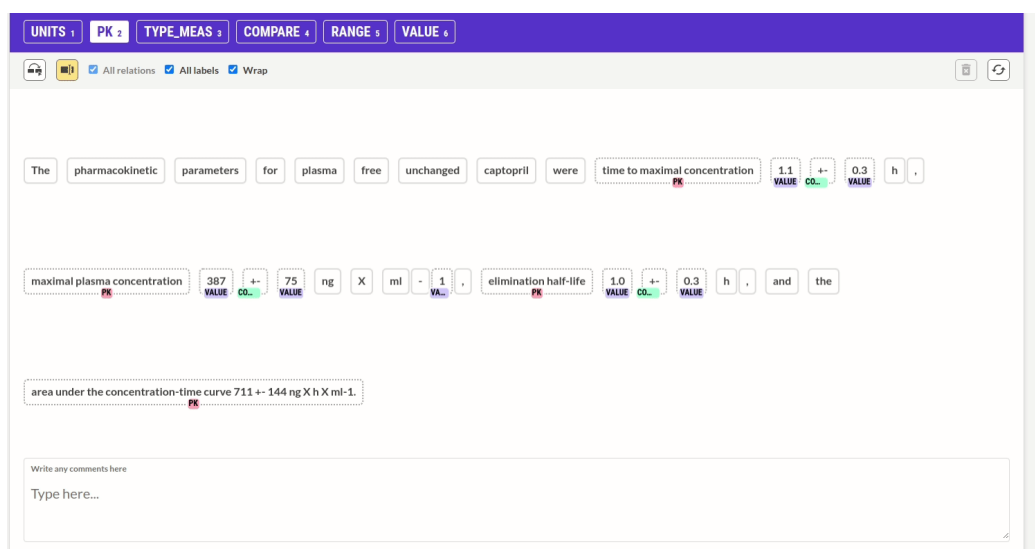
Questions, Answers and Common doubts

Q: “What happens if I want to label a span across two lines?”

A: You will need to unselect the “Wrap” tick to be able to see the sentence in a single line. Then select the span and you can click “Wrap” again. For an example see the clip in the next question

Q: “How are we meant to erase spans?”

A: Select one of the available entities, then click on top of the entity that you’d like to erase and select the bin button on the top right. See following clip:



Q: “Do we need to annotate relations between COMPARE and VALUE if the relation doesn’t refer to a central value (C_VAL) or deviation (D_VAL) measurement?”

A: Let’s **not label** it. We only associate COMPARE-VALUE if that VALUE is also part of a C_VAL or D_VAL relation.

In-depth doubts and resolutions from the initial sentences in the test set can be found here: [Comments test set 0-200](#)

- Percentages and fold increases

Often authors report a comparison of PK parameters between different conditions/drug treatments/patients in the form of % of increase or fold-increase. Some of you had doubts about whether we needed to annotate the relation between PK and those %/fold increases.

The answer is no. In this task, we are trying to extract exact measurements for PK parameters mentioned in the sentence. Unless the PK mention itself is a ratio (e.g. AUCR, AUC/MIC, relative bioavailability (see next point below)) we won't link their relation to the fold/% of increase. So in these examples, there won't be any C_VAL relation between PK and VALUES:



However, if the PK mention is a ratio itself (AUC/MIC, AUC1/AUC2) we will annotate the numerical value associated with it.

- Ratios

Single mentions in the form of ratios such as AUC/MIC, AUC1/AUC2, AUCR, relative bioavailability can be considered PK parameters and linked to their estimated values. However, if we cannot understand that the measurement is a ratio given the PK span mention we will not label it. For instance in the example:

“geometric mean ratios for AUC were 1,2,3...

We would need an additional entity type potentially “type of measurement” (instead, relative bioavailability, AUC/MIC etc, are known parameters that people might search for) to understand that 1,2,3 are ratios of AUCs and not AUC so we won't annotate any C_VAL in that case.

On the other hand, if the sentence says:

“The AUC1/AUC2 were 1,2,3”

From the PK mention “AUC1/AUC2” we can understand that this is a ratio without additional information, so we will label the C_VAL and related entities.

- P-values

Since p-values are the result of a comparison to a null hypothesis and not an estimated PK value **we won't annotate them**. When finding p-values you can leave them as they are without the need to correct entities or annotate any relations

- PD parameters

Yes, we are including PD parameters if they are mentioned, so please label/correct them if not detected by the algorithm

- Remove irrelevant VALUE entities?

We often find numerical values that are part of chemicals, enzymes, genes, tables, equations and don't have anything to do with real estimations of values that have units associated with them.

Next . to determine how the Caco - 2 data affected the prediction of intestinal Peff . the Peff of multiple BKIs was

predicted both with and without the incorporation of Caco - 2 - derived Papp values (Figure 2 B) .

In these cases, we could either leave those VALUE entities or remove them. We ask annotators to **not modify irrelevant VALUE entities unless they are part of a C_VAL or D_VAL relation**. So, in general, we can leave those values as they appear and only modify them if they need to be modified in order to make a C_VAL or D_VAL relation properly (rare case). For instance, if in the following sentence we only find this highlighted:

“The clearance of MDZ was $3 \cdot 10^{-2}$ ”

The annotator would need to modify this span in order to annotate the C_VAL relation between clearance and the estimated value:

“The clearance of MDZ was $3 \cdot 10^{-2}$ ”

Otherwise, no need to touch VALUE entities since those VALUES with no associated relations will be removed automatically.

- Confusing COMPARE entity

COMPARE is only used to identify those values that are maximum, minimums or lower than or higher than the reported value. For instance:

“The CL was higher than 3”

“The CL was > 3”

“The CL was < 3”

In these cases, we want to relate those COMPARE to the value since it will tell us that 3 is not the exact estimate for CL. However, on some occasions, it was not clear whether they should be linked to the value. Consider the following reduced



In this case, reduced does not affect 8.68% since the estimated value for “absolute bioavailability” is not higher or lower than 8.68% but 8.68%. For this reason, we will leave it as it is and not annotate any relation to these cases.

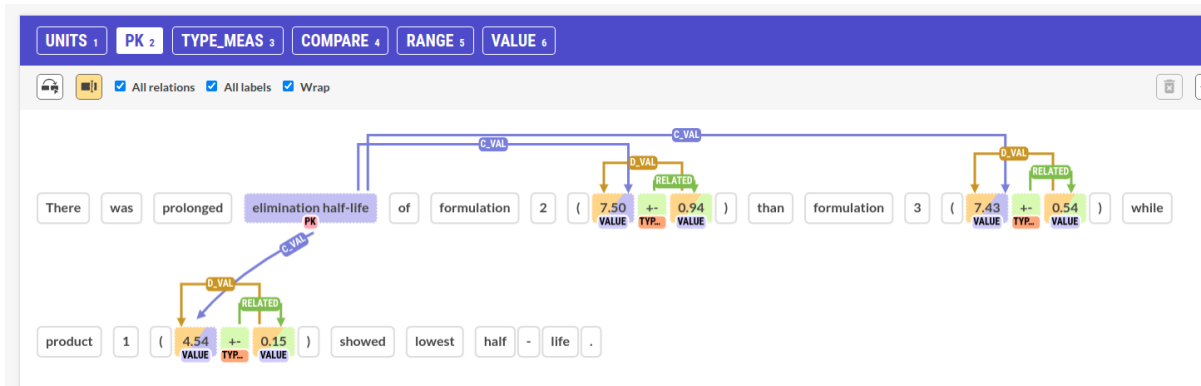
- What about all the missing context? Drugs, doses, species, study design etc are not annotated?

We saw many comments about how to annotate the drugs, doses, and species related to the PK measurements. We certainly care about this context to disambiguate values and also to filter for the desired parameters in a specific drug/population/etc.

However, to simplify the labelling process, we split the task into 2 parts and **won't annotate this complementary information in this task**. When we finish the annotations of this task, then we will complement the central values with all the relevant context, but this will come at a **later stage**.

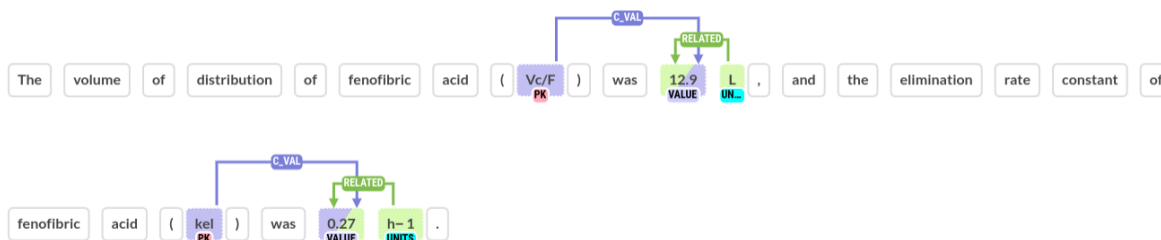
- Two mentions referring to the same parameter

On some occasions, we observed more than one PK mention referring to the same PK value and parameter. For instance, in the following sentence we can see that half-life is mentioned twice, at the beginning, and end of the sentence:



On these occasions, we will prefer to label only the **closest left-side PK mention** as part of the relation

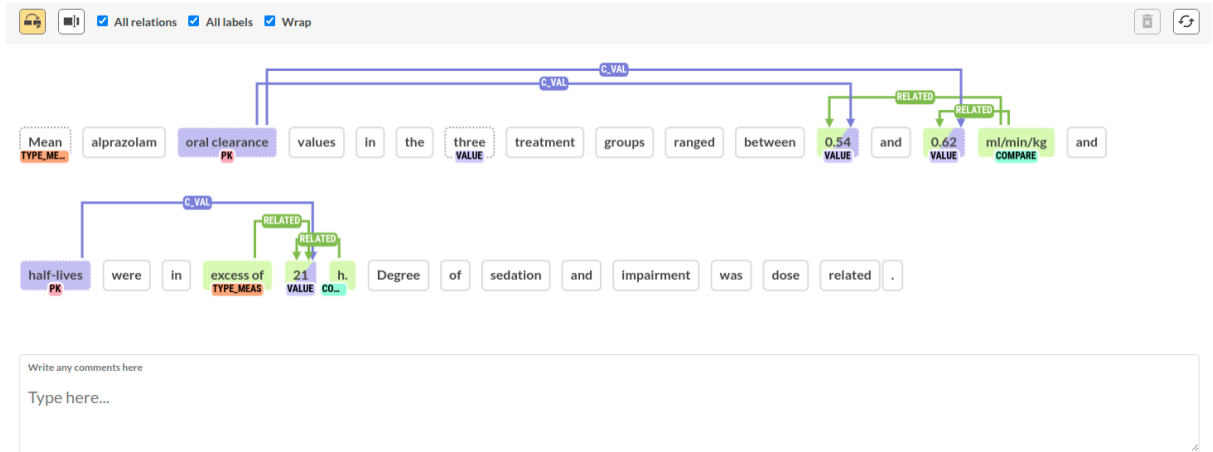
Similarly:



In this case, labelling volume of distribution of fenofibric acid (Vc/F) as a single PK mention would have the issue of including the compound inside the PK entity, so the actual PK entities would be volume of distribution of fenofibric acid (Vc/F). In these cases, we will only consider the **second mention (abbreviated form)** as part of the relation and associate it with the central value

- Range vs Values

Sometimes we might be confused on whether labelling 2 values or 1 range:



In general, if the values refer to different experiment designs (different doses, drugs, patient cohorts etc) we will label 2 values. However, if the range is for a single condition and is the result of variability in the measurement we will label it as a range:

“The AUC for midazolam ranged from 3 to 5”

“The AUCs for midazolam and amoxicillin were 3 and 5”

If units are within the range, then we label 2 separate values. For instance:

“Clearance ranged from 0.54 ml/min/kg to 0.62 ml/min/kg”

If we decide to make **0.54 ml/min/kg to 0.62 ml/min/kg** a range span, that would overlap with units, which can be an issue. So we will label two separate values and C_VAL relations.

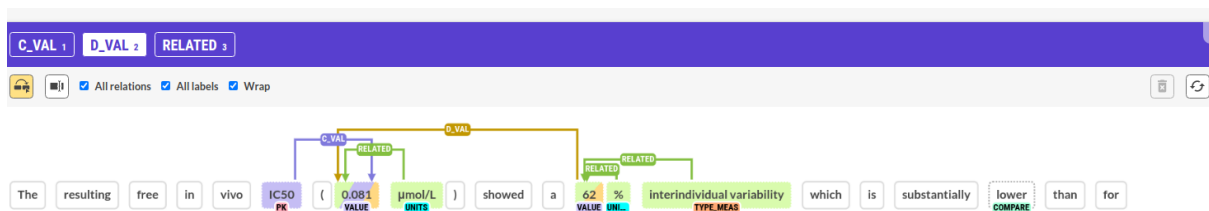
- Value +- Range

This refers to the cases wherein “VALUE (RANGE)” the value is the central value of a PK parameter and RANGE is the measured range of the parameter. You could either consider the RANGE a central value of the parameter or a D_VAL of the VALUE. We decided to label these cases as the latter since then we know both, VALUE and RANGE come from the same estimation and there is no variation between them given by the experimental context:



- IIV

We considered inter-individual variability as a form of deviation:



Appendix D: Unit Dictionaries

```
UNIT_SYNONYMS = {
    ".": ["x", "*", "•", "."],
    "μg": ["micrograms", "micro g", "microg", "microgram", "μg", "mug"],
    "h": ["hr", "hrs", "hour", "hours"],
    "%": ["percent", "percentage"],
    "μl": ["microliters", "microliter", "micro l", "microl", "μl"],
    "l": ["liters", "litre", "liter", "litres"],
    "dl": ["deciliter", "dliter"],
    "min": ["minutes", "minute", "mins"],
    "d": ["days", "day"],
    "month": ["months"],
    "kg": ["kilogram", "kilograms"],
    "s": ["sec"],
    "ms": ["milisec", "miliseconds", "msec"],
    "nM": ["nmol", "nanomol"],
    "mM": ["mmol", "milimol"],
    "μM": ["mumol", "micromol", "micromols", "mumol", "μmol", "μmol", "μM"],
    "pM": ["pmol", "pmols", "picomol"]
}

MAGNITUDES = {
    "TIME": ["ms", "s", "min", "h", "d", "month"],
    "MASS": ["ng", "μg", "mg", "g", "kg", "pg"],
    "VOLUME": ["nl", "μl", "ml", "l", "dl"],
    "CONCENTRATION": ["pM", "nM", "μM", "mM", "M"],
    "PERCENTAGE": ["%"],
}
```
