KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Technology

Data-Efficient Methods for Natural Language Processing: Applications in Healthcare

Heereen Shim

Supervisors:
Prof. dr. ir. Bart Vanrumste
Prof. dr. Stijn Luca
PDEng. MSc. Dietwig Lowet

Dissertation presented in partial fulfilment of the requirements for the degree of Doctor of Engineering Technology (PhD)

January 2023

Data-Efficient Methods for Natural Language Processing: Applications in Healthcare

Heereen SHIM

Examination committee:
Prof. dr. ir. Eric Demeester, chair
Prof. dr. ir. Bart Vanrumste, supervisor
Prof. dr. Stijn Luca, supervisor
(Ghent University)
PDEng. MSc. Dietwig Lowet, supervisor
(Philips Research)
Prof. dr. Marie-Francine Moens
Prof. dr. ir. Mathias Verbeke
Prof. dr. ir. Thomas Demeester
(Ghent University)
Dr. ir. Aki Härmä
(Philips Research)
Prof. dr. Francesca Spigarelli
(University of Macerata)

Dissertation presented in partial fulfilment of the requirements for the degree of Doctor of Engineering Technology (PhD)

January 2023

© 2023 KU Leuven – Faculty of Engineering Technology
Uitgegeven in eigen beheer, Heereen Shim, Andreas Vesaliusstraat 13 box 2600, 3000 Leuven (Belgium)

Alle rechten voorbehouden. Niets uit deze uitgave mag worden vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotokopie, microfilm, elektronisch of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of the publication may be reproduced in any form by print, photoprint, microfilm, electronic or any other means without written permission from the publisher.

Acknowledgement

The four years of my PhD have been an unforgettable journey filled with both joy and struggle. This would have been very difficult if I needed to go through this period by myself. I am grateful for the support and guidance of all the people who helped me along the way.

First and foremost, I would like to express my gratitude to my supervisors, Dietwig Lowet, Bart Vanrumste, and Stijn Luca. Their constant support and wise advice were invaluable to me throughout my PhD journey. I am honored to have had such exceptional supervisors and I am grateful for all that they have done for me.

Dietwig gave me close and frequent guidance over the last four years. I am grateful for the many interesting discussions we had, as well as his advice on how to find interesting problems not only in the scientific domain but also in the industrial context, helped me to navigate through my PhD and successfully land in a new place afterwards.

I am particularly grateful to Bart for his persistent support and the freedom he gave me to choose my PhD topic. His constructive feedback on the challenges of machine learning played a significant role in the completion of my thesis. I am also grateful for his support in every step of my PhD, from organizing the examination committee to conducting research activities, and his kindness in helping me to get involved in the e-Medial lab while working remotely. I will always be thankful for his kind support and encouragement.

I would also like to express my gratitude to Stijn for the critical feedback he provided during my studies. His comments during our team meetings helped me to substantially improve the quality of my research. I am grateful for his patience and support throughout my four-year journey.

Secondly, I would like to extend my gratitude to my assessors, Prof. Sien Moens, Prof. Francesca Spigarelli, and Prof. Jesse Davis, who took the time to assess

and provide feedback on my research throughout my PhD. Also, I would like to thank my jury members, Prof. Thomas Demeester, Dr. Aki Härmä, and Prof. Mathias Verbeke, for their time and effort in assessing my work and for discussions helped me improve my thesis. Extra thanks to Prof. Thomas Demeester and Dr. Aki Härmä for agreeing to review my thesis and travel to Leuven for this. Special thanks to Prof. Eric Demeester for serving as the chair of the examination committee and for my defenses.

I am incredibly fortunate to have had the opportunity to work with such an exceptional group of individuals at Philips Research and the AI for Engagement team. I would like to thank Marieke van der Hoeven for including me in the Digital Engagement, Cognition & Behavior department. The department meetings were enlightening and I learned a great deal from my colleagues. A special thanks to my team lead Martijn Krans for making me feel like a valued member of the team. His kind consideration and support were instrumental in helping me to balance my PhD work with Philips and develop additional skills as a scrum master. I am also grateful to Arlette van Wissen, Aki Härmä, and Rim Helaoui for inviting me to participate in the Philhuman project's training circles. The opportunities to learn from other Marie Curie students and take training, particularly on biases in AI, greatly contributed to my work in the final chapter of my thesis. I extend my sincere thanks for their warmth and hospitality.

I am grateful for the opportunity to have conducted my PhD studies within the framework of the HEART project funded by the European Union's Horizon 2020 research and innovation programme through the Marie Sklodowska-Curie grant. I would like to thank the project's partners, the University of Macerata, Fudan University, Isinnova, Eurocentro, and Philips China, for their assistance in organising international collaborations and providing invaluable opportunities to learn interdisciplinary skills. Extra thanks to Prof. Wei Chen for hosting me in her research lab during my stay in Shanghai. Barbara Chiucconi deserves special thanks for smooth operations and procedures, even during the extremely difficult time of the pandemic.

I also want to thank my colleagues and friends who made my PhD journey unforgettable. Firstly, I am grateful to my dear friends from the HEART project, Chetanya, Koustabh, Oleksandr, Nuoya, and Yuan. The experience of working with them from Eindhoven to Shanghai was truly amazing. I will always cherish our time spent together for lunch, dinner, drinks (even though I was usually the only one drinking), and impromptu trips during the pandemic period. I also want to thank my colleagues from the e-Media lab, Ahmed, Benjamin, Duowei, Hannelore, Ine, Kymeng, and Yiyuan. I am especially grateful to Yiyuan for listening to my concerns and meeting with me whenever I visited Leuven. You were right, the PhD project did get better in the third year. I also want to

thank my other PhD friends, Allmin and Alex from the Philhuman project, as well as the wonderful individuals, Eunjee, Tim, and Núria whom I met outside of my PhD work and shared memorable moments with in Eindhoven.

Furthermore, I am eternally grateful to my two favourite girls, Eunyoung and Heeyeon. My life in Seoul and Eindhoven would not have been complete without them. I am grateful for their unconditional love, understanding, and encouragement that has helped me to become a better person.

Lastly, I would like to express my gratitude to my parents and friends in South Korea, who have always provided me with an emotional support system. I am truly grateful to my parents for their unwavering belief in me and for giving me the freedom to explore the world. I would also like to express my deepest appreciation to my grandmother, who told me to *Live Cool* when I started my PhD journey. She will always be remembered and missed.

Heereen Shim

from Amsterdam, 17. January. 2023.

Abstract

Natural language processing is the study of processing language data to perform human language-related tasks. With the advance of machine learning models, such as deep neural networks, natural language processing technologies have been applied to many use cases, including document classification, sentiment analysis, and information extraction. Deep neural networks perform a target task by learning from data without human intervention for inference. However, their power comes at the cost of large, labelled training data which require a lot of human labour.

In this dissertation, we investigate and propose data-efficient methods for training neural networks-based natural language processing models for healthcare applications, where data and labels are scarce. Our four main contributions demonstrate how data-efficient methods that maximise the utility of labelled and unlabelled data and exploit knowledge can be used to train neural networks-based natural language processing models in data- and label-scarce settings.

Firstly, we present a data-efficient method that combines a data augmentation technique and a semi-supervised learning approach to a setting where there is a small labelled dataset and a relatively large unlabelled dataset. The data augmentation method applies text editing operations to input texts, and the semi-supervised learning method utilises a trained model's predictions as pseudo-labels. We evaluate our method on a custom dataset containing user complaints about their sleep and analysed the effect of the proposed method.

Secondly, we focus on active learning methods, particularly pool-based active learning, which is when there is a relatively large amount of unlabelled data and a small amount of labelled data at the beginning, and the fixed number of data points are iteratively labelled and added to the labelled set. We first analyse the limitations of existing active learning methods and proposed a label-efficient training method that mitigates them. The proposed method combines the

strength of self-supervised learning, data augmentation, and active learning to fully utilise both unlabelled and labelled data. We evaluate our method on our custom dataset and a benchmark dataset and find that the proposed method outperforms the existing state-of-the-art methods.

Thirdly, we study how to add numeracy skills into a language model by using synthetic data for a temporal information extraction task. We propose a rule-based synthetic data generation method that can increase the size of the training data and a novel multi-task model architecture that can extract temporal expressions and normalise them into standard formats. We evaluate our methods on a custom dataset containing free text sleep diaries. We find that multi-task learning that includes an auxiliary task, which is related to the target task, can contribute to the target performance improvement when using synthetic data for training.

Lastly, we investigate the opportunities of applying the data-efficient methods to a clinical NLP application and discussed the important problem of bias. We first study the underlying bias in the public benchmark dataset and analyse the effect of bias on the model's behaviour. We find that the benchmark-trained model performs differently across demographic groups because the benchmark dataset is imbalanced. We then propose novel approaches to mitigate this problem. We evaluate our methods on the clinical benchmark dataset and show that the proposed method can achieve better fairness scores resulting in equal performances across different demographic groups.

The main conclusion of this dissertation is that the proposed data-efficient methods are the most effective in low-resource settings when there is a small-sized training data set or a small subset of the training dataset is labelled. The contributions of this dissertation are a starting point for future research into developing deep neural networks-based natural language processing systems for low-resource application domains.

Beknopte samenvatting

Natuurlijke taalverwerking is de studie van het verwerken van taalgegevens om menselijke taalgerelateerde taken uit te voeren. Met de opmars van machine learning-modellen, zoals diepe neurale netwerken, zijn natuurlijke taalverwerkingstechnologieën toegepast op veel gebruiksscenario's, waaronder documentclassificatie, sentimentanalyse en informatie-extractie. Diepe neurale netwerken voeren een doeltaak uit door te leren van gegevens zonder menselijke tussenkomst voor gevolgtrekking. Hun kracht gaat echter ten koste van grote, gelabelde trainingsgegevens die veel menselijke arbeid vergen.

In dit proefschrift onderzoeken en stellen we data-efficiënte methoden voor voor het trainen van op neurale netwerken gebaseerde natuurlijke taalverwerkingsmodellen voor toepassingen in de gezondheidszorg, waar data en labels schaars zijn. Onze vier belangrijkste bijdragen laten zien hoe data-efficiënte methoden die het nut van gelabelde en niet-gelabelde data maximaliseren en kennis exploiteren, kunnen worden gebruikt om op neurale netwerken gebaseerde natuurlijke taalverwerkingsmodellen te trainen in data- en label- schaarse omgevingen.

Ten eerste presenteren we een data-efficiënte methode die een techniek voor data-augmentatie en een semi-gesuperviseerde leerbenadering combineert in een omgeving met een kleine gelabelde dataset en een relatief grote ongelabelde dataset. De methode voor gegevensvergroting past tekstbewerkingsbewerkingen toe op invoerteksten en de methode voor semi-gesuperviseerd leren gebruikt de voorspellingen van een getraind model als pseudo-labels. We evalueren onze methode op een aangepaste dataset met klachten van gebruikers over hun slaap en analyseerden het effect van de voorgestelde methode.

Ten tweede richten we ons op actieve leermethoden, met name op pool-gebaseerd actief leren, dat is wanneer er een relatief grote hoeveelheid niet-gelabelde gegevens en een kleine hoeveelheid gelabelde gegevens aan het begin zijn, en het vaste aantal gegevenspunten iteratief worden gelabeld en toegevoegd

aan de gelabelde set. We analyseren eerst de beperkingen van bestaande actieve leermethoden en stelden een label-efficiënte trainingsmethode voor die ze verzacht. De voorgestelde methode combineert de kracht van zelfgestuurd leren, gegevensvergroting en actief leren om zowel ongelabelde als gelabelde gegevens volledig te benutten. We evalueren onze methode op onze aangepaste dataset en een benchmark-dataset en stellen vast dat de voorgestelde methode beter presteert dan de bestaande state-of-the-art methoden.

Ten derde bestuderen we hoe rekenvaardigheden kunnen worden toegevoegd aan een taalmodel door synthetische gegevens te gebruiken voor een tijdelijke informatie-extractietaak. We stellen een op regels gebaseerde methode voor het genereren van synthetische gegevens voor die de omvang van de trainingsgegevens kan vergroten en een nieuwe multi-task modelarchitectuur die tijdelijke uitdrukkingen kan extraheren en normaliseren in standaardformaten. We evalueren onze methoden op een aangepaste dataset met slaapdagboeken met vrije tekst. We vinden dat multi-task leren met een hulptaak, die gerelateerd is aan de doeltaak, kan bijdragen aan de prestatieverbetering van het doel bij het gebruik van synthetische data voor training.

Ten slotte onderzoeken we de mogelijkheden om de data-efficiënte methoden toe te passen op een klinische NLP-toepassing en bespraken we het belangrijke probleem van vooringenomenheid. We bestuderen eerst de onderliggende bias in de openbare benchmarkdataset en analyseren het effect van bias op het gedrag van het model. We constateren dat het benchmark-getrainde model verschillend presteert tussen demografische groepen, omdat de benchmark-dataset onevenwichtig is. Vervolgens stellen we nieuwe benaderingen voor om dit probleem te verminderen. We evalueren onze methoden op basis van de klinische benchmarkgegevensset en laten zien dat de voorgestelde aanpak betere eerlijkheidsscores kan behalen in termen van gelijke prestaties in verschillende demografische groepen.

De belangrijkste conclusie van dit proefschrift is dat de voorgestelde data-efficiënte methoden het meest effectief zijn in omgevingen met weinig middelen, wanneer er een kleine trainingsdataset is of een kleine subset van de trainingsdataset is gelabeld. De bijdragen in dit proefschrift zijn een startpunt voor toekomstig onderzoek naar de ontwikkeling van op diepe neurale netwerken gebaseerde natuurlijke taalverwerkingssystemen voor toepassingsdomeinen met weinig middelen.

List of Abbreviations

ABC as binary classification

ABSA aspect-based sentiment analysis

AI arrtificial intelligence

ANN arrtificial neural networks

BERT bidirectional encoder representations from Transformers

CNN convolutional neural networks

DAWL distribution-aware weighted loss

DNN deep neural networks

EDA easy data augmentation

FFNN feed-forward neural networks

GRU gated recurrent unit

ICD international classification of diseases

LETS label-efficient training scheme

LSTM long short-term memory networks

MLM masked language modelling

MLP multi-layer perceptrons

NLP natural language processing

NSP next sentence prediction

RNN recurrent neural networks

seq2seq sequence-to-sequence

UMLS universal medical language system

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Language is a medium of communication that allows humans to share information and interact with each other. Language is difficult because it is ambiguous, contextual, and highly nuanced, and it changes constantly across space and time. Therefore, understanding and using language are some of the most fundamental abilities of human intelligence. Because of language's complexity, building a computer system with language understanding skills is one of the most challenging tasks in the field of artificial intelligence (AI). The progress of natural language processing (NLP), the study of processing language data, often in written formats (text), to perform human language-related tasks, has been falling behind other fields of AI that outperform humans in game playing (Silver et al., 2016) or image recognition tasks (He et al., 2016).

Over the past few years, the performance of NLP systems has exponentially improved and achieved groundbreaking results in multiple NLP benchmark tasks (Vaswani et al., 2017; Devlin et al., 2019). This recent progress in NLP has been powered by deep neural networks (DNN) (Krizhevsky et al., 2012; Goodfellow et al., 2016) that can learn how to extract meaningful features for the target tasks from the labelled training data, which is called supervised learning. Moreover, self-supervised learning, which learns from unlabelled by using inherent learning signals[1], enables the use of a large, unlabelled corpus for pre-training neural networks. Further, many researchers have demonstrated

[1]For example, raw text can be used to train a system that can predict the next word given in the previous words because it does not require additional annotation.

that pre-trained neural network models, such as word embeddings (Mikolov et al., 2013; Pennington et al., 2014a; Bojanowski et al., 2017) and language models (Peters et al., 2018; Howard and Ruder, 2018), can be easily fine-tuned to various down-stream applications. This two-step training (i.e., first pre-training and then fine-tuning) is a type of transfer learning (Pan and Yang, 2009) and is becoming the standard workflow in NLP. The success of transfer learning has continued and has resulted in the era of large pre-trained language models with hundreds of millions of parameters, such as bidirectional encoder representations from Transformers (BERT) (Devlin et al., 2019), GPT (Radford et al., 2018), and T5 (Raffel et al., 2020). These neural NLP models benefit various downstream NLP tasks (Rogers et al., 2020), such as language translation (Bahdanau et al., 2015) and text generation (Brown et al., 2020).

In this dissertation, we investigate how NLP technologies can be applied to a healthcare domain. Specifically, we focus on a personal healthcare domain that aims to empower people to live healthy lifestyles and encourages people to engage actively in the healthcare process. Our goal is to develop digital healthcare tools that can make healthcare services more accessible and personalised. By using digital healthcare tools, users can receive healthcare services in out-of-hospital settings. Moreover, users can actively engage in healthcare services by logging their statuses and monitoring their progress. Allowing people to describe their experiences in their own words is especially critical because people can directly report their issues and are not limited to multiple choice responses. The use of natural language also gives healthcare providers insight into user-specific circumstances that are difficult to capture with sensor devices. NLP technologies enable the analysis of user-generated texts and will play a key role in understanding healthcare recipients' experiences.

There are some challenges in applying NLP technologies, especially neural NLP models, to the personal healthcare application domain. First, existing pre-trained language models have been trained with a general corpus, obtained from books (Zhu et al., 2015), online encyclopaedias (i.e., English Wikipedia), or websites, all of which differ from data obtained within personal healthcare use cases. A language model needs to be pre-trained with a corpus from a target domain (Gururangan et al., 2020), especially for use cases with domain-specific texts, such as texts from biomedical domain (Lee et al., 2020b). Unfortunately, it is difficult to obtain a large corpus from each target application domain. Especially since the introduction of the General Data Protection Regulation (GDPR), collecting a large amount of data from personal healthcare users could be challenging or might not be possible because of privacy issues.

Another option is to fine-tune large pre-trained language models rather than pre-training from scratch. The remaining issue, however, is that large pre-trained language models still require labelled, task-specific datasets for fine-tuning.

Moreover, it is well known that the performance of neural models depends on the size of the training datasets (Wang et al., 2020; Qi and Luo, 2020). Further, fine-tuning a large pre-trained model on a downstream task with small datasets is unstable and prone to degenerating performance (Phang et al., 2018; Lee et al., 2020a). In other words, a large training dataset is still required for a target downstream task to fully exploit the power of pre-trained language models, even after by fine-tuning (Adadi, 2021).

This dependency on large training datasets creates two challenges when applying neural NLP models in a personal healthcare domain. The first challenge is data scarcity (when there is a lack of training data). Data scarcity can be caused by the inherent problem of real-world data that can be not equally distributed. For example, imbalanced data (when there are many more data points for one label category than others) is a common feature of real-world datasets (He and Garcia, 2009). In an imbalanced dataset, data with rare labels are scarce. Moreover, the issue of data scarcity is inevitable when collecting rare event data. For example, in the clinical domain, there are rare diseases that affect a small percentage of the population. Compiling large datasets for rare diseases is practically impossible.

The second challenge is label scarcity (when there is a limited amount of labelled data). Data labelling is a time-consuming and unscalable process that requires significant human resources (Chui et al., 2018). Crowdsourcing could be an alternative to obtaining a large amount of labelled data in a short time by hiring multiple annotators. However, the limitation of crowdsourcing is the noise associated with labels from multiple ordinary annotators (Rodrigues and Pereira, 2018). Further, because of privacy concerns, such a practice could be limited to the use of crowdsourcing platforms for data labelling. Especially in fields that require domain-specific knowledge, such as the clinical domain, crowdsourcing might not be possible and data labelling could be very expensive. Moreover, in real-world scenarios, the labelling scheme may be changed after deployment. For example, when developing a classifier (e.g., a disease classifier), new labels might need to be added after development because of the emergence of new categories (e.g., COVID-19). Therefore, scalable data labelling is critical when building applications in a real-world setting.

Data-efficient methods promise to mitigate these gaps by maximising the utility of training data, using efficient learning strategies, or exploiting external resources. Data-efficient methods have been a fundamental element of many machine learning development pipelines. For example, data augmentation has been widely used in the image recognition domain (Shorten and Khoshgoftaar, 2019). Another label-efficient learning strategy, such as semi-supervised learning, which utilises unlabelled data in a supervised learning setting, has been widely studied in the machine learning field (Zhu, 2005). Active learning (Settles,

2009), which is an iterative learning framework that selects the most informative data points to be manually labelled, is also a particular type of label-efficient learning strategy. Another line of work to mitigate data scarcity or label scarcity is to utilise external resources or knowledge. For example, several knowledge bases, which are structured data systems, exist that can be used to improve supervised learning with limited target data, especially when domain knowledge is important. Knowledge can also be injected into machine learning models via additional loss terms or by generating synthetic data. Specifically, synthetic data generation is a promising data-efficient method that can mitigate data scarcity issues (von Rueden et al., 2021).

In this dissertation, we investigate how to fine-tune large pre-trained language models for various downstream tasks to achieve better performance when data or labels are scarce in target application domains. To this end, we develop novel data-efficient methods that allow models to generalise better with less training data or labelled data for a variety of use cases. We demonstrate that models employing the proposed methods outperform both models without data-efficient methods as well as existing data-efficient methods. We expect many application areas, especially in domains where data and labels are scarce, can benefit from these data-efficient methods.

In the remainder of this chapter, we describe the research context, the research objectives, and research questions. Further, we provide the thesis outline and summarise the main contributions of this work.

1.2 Research Context

In this dissertation, we study three use cases that originated from the business context of Philips Research. Specifically, we consider a personal healthcare domain that aim to empower people to live healthy lifestyles and encourage people to engage actively in the healthcare process. Within these use cases, NLP technologies play a key role by enabling better interfaces where people can provide inputs (e.g., health-related issues) in their own words.

Also, the research in this dissertation was pursued in the context of the HEART project: "HEalth related Activity Recognition system based on IoT" (EU Horizon 2020 MSCA No. 766139). The HEART project aims to build a system for remotely monitoring human activities and extracting health-related information from data, including user-reported health measures (e.g., weights) and free-form inputs (e.g., free texts) in an out-of-hospital setting. This dissertation includes the research works on processing the user-generated free-text data to extract not

only objective but subjective and contextual information that can be difficult to obtain from sensor data.

1.3 Research Questions

This thesis studies data-efficient methods for fine-tuning neural NLP models, such as large pre-trained language models when one has small-sized training datasets for target tasks or when the datasets are not fully labelled, while minimally impacting performances.

We argue that data-efficient methods support supervised learning for fine-tuning large pre-trained language models in resource-scarce settings. Since the main focus of this thesis is the data-hungry issue of current deep neural models, resources in this context refer to data used for fine-tuning. More specifically, we consider the following resource-scarce settings:

1. Data-scarcity: When there is not enough training data from a target application domain.
2. Label-scarcity: When there is a large amount of training data from a target application domain but these data are not yet labelled.

Overall, we pose three research questions that are addressed by the approaches proposed in this thesis:

RQ1. How can we fine-tune a pre-trained language model when only a small-sized training set for the target task is available?

RQ2. How can we fine-tune a pre-trained language model when only a small subset of the target dataset is labelled?

RQ3. Can we exploit other resources (e.g., knowledge, databases, et cetera) during fine-tuning to improve the performance of a pre-trained language model?

1.4 Outline of Thesis

Figure 1.1 illustrates an overview of the present thesis. Throughout the thesis, we consider three novel NLP use cases in a personal healthcare domain. Specifically, we focus on potential applications for a sleep coaching program, from assessment to coaching and monitoring. Additionally, we consider one clinical application

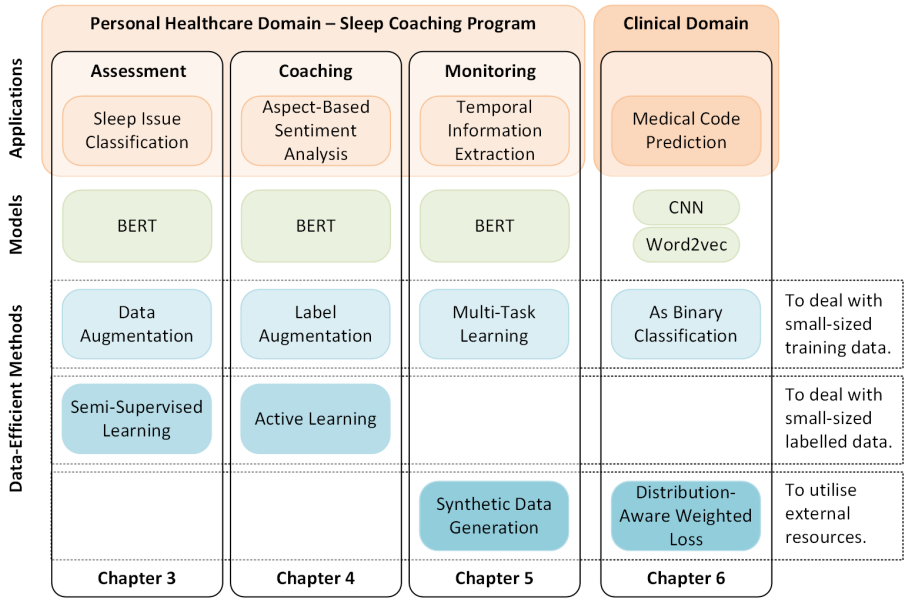


Figure 1.1: The overview of the thesis.

(i.e., medical code prediction) and use a benchmark dataset from a clinical domain to validate the usability of the proposed data-efficient methods. We use BERT (Devlin et al., 2019), which is a well-known pre-trained language model, as a baseline model architecture in this dissertation, except in the last chapter. In the last chapter we focus on a clinical use case, and convolutional neural networks (CNNs) with pre-trained word embeddings (word2vec) are used.

In the following sections, we outline the chapters by describing the context, the research questions, and our contribution to each use case.

Ch 3. Assessment: Sleep Issue Classification

In Chapter 3, we introduce a use case that involves understanding participants' complaints based on free text. For this use case, we aim to build a machine learning model that can classify user-generated free texts into pre-defined sleep issue categories. Since a machine learning model learns from data, it requires a dataset before the model can be developed. We will likely not have enough training data for model building at the beginning of a development process. Considering the development cycle of a machine learning model, we can obtain a new batch of data after the initial model deployment. Then we can use the

new data to train an improved model. Since the new data do not contain labels, however, manual labelling is needed.

To this end, we study two questions: The first is how to train a large neural network model with a small-sized training dataset. The second is how to utilise unlabelled data without manually labelling them. The first question is linked to the **RQ1**, and the second question is linked to the **RQ2** mentioned in Ch 1.3.

To address these questions, we propose a method which combines data augmentation and semi-supervised learning. The proposed data augmentation technique increases the size of the labelled data set. The proposed semi-supervised learning is an iterative learning framework that annotates unlabelled data based on the trained model's predictions. We study the effect of data augmentation and semi-supervised learning while varying the size of the initial training set. We find that data augmentation improves performance significantly, especially when the initial training set is small. Experimental results also indicate that semi-supervised learning can provide further performance improvements. Another interesting finding is that data augmentation is beneficial to the minority label classes when the dataset is imbalanced.

Ch 4. Coaching: Aspect-Based Sentiment Analysis

In Chapter 4, we introduce a use case of understanding each participant's experience with the coaching program by analysing their reviews. We aim to build an aspect-based sentiment analysis system that can analyse these reviews to understand what people liked and disliked. For example, a system can detect the expressed sentiment values (i.e., positive, neutral, negative) when a review text contains multiple opinions associated with different aspects (e.g., "*My sleep quality has been improved (positive) but I did not like decaffeinated coffee (negative)*"). In this use case, we consider a real-world scenario when we obtain a series of new data batches, but the labelling budget is limited and the data sets are imbalanced in terms of label classes. Therefore, we aim to develop an active learning framework with the goal of iteratively annotating unlabelled data by selecting the most informative data points. We also aim to train a model to perform equally well for both frequent and rare label classes.

In other words, there are two goals: the first is to use the labelling budget efficiently to improve performance, which is linked to the **RQ2**. The other goal is to deal with an imbalanced dataset that causes performance differences between frequent and rare label classes, which is linked to the **RQ1**.

To this end, we propose a label-efficient training scheme. The proposed method utilises unlabelled data by employing self-supervised pre-training. This method

also increases the size of samples with rare labels by using a novel label augmentation technique. Lastly, the proposed active learning algorithm selects data points by using two different uncertainty scores to handle performance differences between frequent and rare classes. We experimentally prove the effectiveness of the proposed method using a custom dataset and a benchmark dataset. The results suggest that the proposed method can achieve competitive performance with only half or even a third of labelled samples in label-scarce settings. Moreover, the proposed method can improve model generalisation by increasing performance with the same amount of labelled data compared to other models trained without the proposed methods.

Ch 5. Monitoring: Temporal Information Extraction

In Chapter 5, we introduce a use case for monitoring sleep activity using a free-text sleep diary tool. This use case focuses on extracting temporal information related to sleep events from free text. A traditional method solves temporal information extraction tasks with a two-stage approach: temporal expression detection, which is a task to extract mentioned temporal expressions (e.g., "*10 in the evening*") from text, and temporal expression normalisation, which is a task to translate the extracted temporal expressions into standard formats (e.g., 22:00). To exploit the pre-trained language model, we reformulate the temporal information extraction task as a question and answer task (Given document: sleep diary text, Q: when did the user go to bed last night? A: *"at midnight"* (00:00)). One challenge is that pre-trained language models lack numeracy skills that apply numerical concepts for extracting temporal information from text. Collecting more data and training a model with them is one of the most intuitive approaches. Another approach could be utilising human knowledge, such as common structures (i.e., *"ten to twelve"* (11:50), *"ten thirty"* (10:30)), common-sense (i.e., *"ten in the evening"* (22:00)), or constraints (i.e., the value of hour $h \in [0, 24]$, the value of minute $m \in [0, 60)$) of temporal expressions.

In other words, we study the following questions: the first is how to train a large language model to learn numeracy skills when only a small amount of training data is available. This question is linked to the **RQ1**. The other question is how to inject human knowledge of temporal expressions into a machine learning model. This question is linked to the **RQ3**.

To answer these questions, we formulate the following hypothesis: we can first use human knowledge of typical temporal expressions to program regular expressions for generating synthetic data. The generated synthetic data can then be used to inject numeracy skills into a language model for extracting temporal information from text. To this end, we propose a synthetic data generation

algorithm to augment the training data. We also propose a multi-task learning approach by introducing an auxiliary task, which is related to the target task, to utilise additional training signals. Experimental results indicate that synthetic data can improve the performance of temporal information extraction tasks that require numeracy skills. We also find that training a model with an auxiliary task can improve its performance on the target task when synthetic data are used for training.

Ch 6. Medical Code Prediction: Multi-Label Classification

In the final study of this dissertation, we validate the usability of the previously proposed methods for a clinical use case of medical code prediction. In Chapter 6, we investigate the underlying bias in a clinical benchmark dataset. Since the benchmark dataset contains patient data, we focus on the patient demographics to find demographic imbalances and study their effect on a benchmark-trained model's performance. Specifically, we aim to address the issue of an unfair model which performs differently across different demographic groups, which is caused by a demographically imbalanced training dataset.

In this investigation, we study how to address the performance differences caused by imbalanced data, which is linked to the **RQ1**. We also explore whether we can utilise additional resources, such as knowledge of data, label information, or an external clinical database, to further improve the model's performance, which is linked to the **RQ3**.

In our first data analysis study (Ch 6.1), we analyse the benchmark dataset to identify an existing gap and find that the benchmark dataset is imbalanced in terms of patient demographics. This data imbalance creates performance differences across different demographic groups. In the following model development study (Ch 6.2), we propose two approaches to address these performance differences. The first method is an ensemble approach that aims to build multiple group-specific models. We propose a novel distribution-aware weighted loss function that utilises knowledge of label distribution for group-specific weighting. The second approach formulates the medical coding task, which is multi-label classification, as binary classification. We propose a novel model architecture that utilises label information as an input. We also utilise an external clinical database for data augmentation. Experimental results demonstrate that the ensemble approach with the proposed loss function can improve global performance and the proposed binary classification approach can help a model perform equally well across different demographic groups. However, experimental results indicate that the proposed data augmentation

using the clinical database harms the performance of a binary classification model.

1.5 List of Publications

The work in this dissertation primarily relates to the following peer-reviewed journal articles and conference proceedings (in order of publication) :

Journal Articles

1. **Shim, H.**, Lowet, D., Luca, S.,& Vanrumste, B. (2021). LETS: A Label-Efficient Training Scheme for Aspect-Based Sentiment Analysis by Using a Pre-Trained Language Model. *IEEE Access*.

Conference Proceedings

1. **Shim, H.**, Lowet, D., Luca, S., & Vanrumste, B. (2022). An exploratory data analysis: the performance differences of a medical code prediction system on different demographic groups. In *Proceedings of the NAACL 2022 Clinical NLP Workshop*.
2. **Shim, H.**, Lowet, D., Luca, S., & Vanrumste, B. (2021). Synthetic Data Generation and Multi-Task Learning for Extracting Temporal Information from Health-Related Narrative Text. In *Proceedings of the EMNLP 2021 W-NUT Workshop: The Seventh Workshop on Noisy User-generated Text*.
3. **Shim, H.**, Luca, S., Lowet, D., & Vanrumste, B. (2020). Data augmentation and semi-supervised learning for deep neural networks-based text classifier. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*.

The following peer-reviewed conference proceedings and journal article are related, but will not be extensively discussed in this thesis:

1. **Shim, H.** (2022). Data-Efficient Algorithms and Neural Natural Language Processing: Applications in the Healthcare Domain. In *Proceedings of the IJCAI-ECAI. Doctoral Consortium*.

2. Lepore, D., Dolui, K., Tomashchuk, O., **Shim, H.**, Puri, C., Li, Y., Chen, N., & Spigarelli, F. (2022). Interdisciplinary research unlocking innovative solutions in healthcare. Technovation, 102511.

3. **Shim, H.**, Lowet, D., Luca, S., & Vanrumste, B. (2021). Building blocks of a task-oriented dialogue system in the healthcare domain. In *Proceedings of 2021 NAACL Workshop NLPMC: the Second Workshop on Natural Language Processing for Medical Conversations.*

4. **Shim, H.** (2021). Development of Conversational AI for Sleep Coaching Programme. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop.*

Finally, while not directly related, the following article has also been completed over the course of the PhD as a result of supervising master students:

1. Kicken, K., De Maesschalck, T., Vanrumste, B., De Keyser, T., & **Shim, H**. (2020). Intelligent Analyses on Storytelling for Impact Measurement. In *Proceedings of the 2021 EMNLP Workshop W-NUT: The Sixth Workshop on Noisy User-generated Text.*

Chapter 2

Fundamentals

In this chapter, we provide background knowledge to set the stage for the following chapters. We start by providing a brief introduction to artificial neural networks, as all models in the thesis can be considered neural networks. Afterwards, we introduce word embeddings that are used to represent words as high-dimensional vectors. Lastly, we describe the progress in the modern neural architectures with the emphasis on Transformer-based models, including BERT which is the backbone architecture of the proposed models in the thesis (Chapters 3, 4, 5).

2.1 Artificial Neural Networks

Artificial neural networks (ANN) are a subset of machine learning models that are inspired by the biological neural networks in animal brains, mimicking the way that biological neurons signal to one another. ANN consist of an input layer, one or more hidden layers, and an output layer. As illustrated in Figure 2.1, each neuron, takes a number of inputs $\boldsymbol{x} = \{x_1, ..., x_N\}$ with corresponding weights $\boldsymbol{w} = \{w_1, ..., w_N\}$ and a bias b. The output of a single neuron y is a weighted sum of the inputs, followed by an activation function:

$$y = f(\sum_{i=1}^{N} x_i w_i + b) \tag{2.1}$$

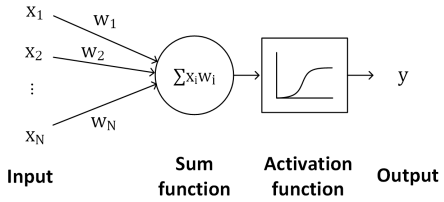


Figure 2.1: Illustration of a neuron that takes a number of inputs $\boldsymbol{x} = \{x_1, ..., x_N\}$ with corresponding weights $\boldsymbol{w} = \{w_1, ..., w_N\}$, and produce y as an output. A bias term b is omitted in the illustration.

where $f(\cdot)$ is an activation function. A classical option for an activation function is the logistic sigmoid function defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$

which squashes the input values from $[-\infty, \infty]$ into the interval $[0, 1]$ that is often interpreted as output probability $p(y = 1|\boldsymbol{x})$ for a binary classification problem. For a multi-label classification problem, the softmax function is used as an activation function in the final classification layer. Softmax function takes a vector $\boldsymbol{x} = \{x_1, ..., x_K\}$, as defined as:

$$\text{softmax}(\boldsymbol{x})_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{2.3}$$

where all the output values of the function sum to 1, thus is often considered as a probability distribution over multiple classes.

Figure 2.2 illustrates the example of ANN with an input layer, two hidden layers, and an output layer, also called feed-forward neural networks (FFNN) or multi-layer perceptrons (MLP). If there is more than one hidden layer, the depth of networks is regarded as "*deep*".

2.2 Word Representations

Word representation, also called word embedding, is proposed to allow computers to process language data effectively. Unlike image or audio signals which naturally capture the information of data and can be represented as vectors,

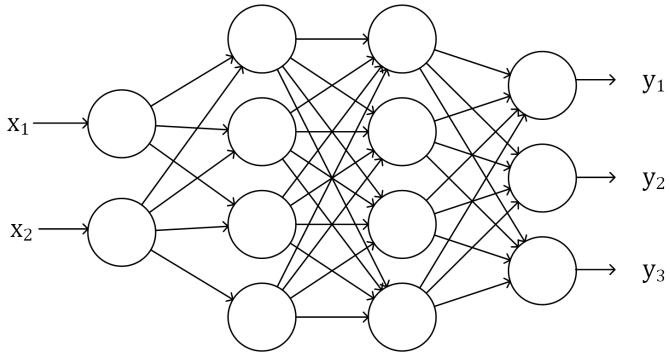


Figure 2.2: An example of feed-forward neural networks, consisting of two hidden layers, each with four nodes. Bias terms and activation functions are not shown.

words have been regarded as discrete atomic symbols. Therefore, how to represent a word plays a critical role in NLP.

2.2.1 Sparse Representations

A classical approach is to create a one-hot vector whose size is $|V|$, where $V = \{w_1, w_2, ..., w_n\}$ is a pre-defined dictionary containing n words. The drawback of this approach is that these one-hot vectors capture no semantic meaning. Also, since the size of the vocabulary increases linearly as the number of words is growing, increasing vocabulary can cause a curse of dimensionality. Therefore the utility of these one-hot vectors is limited.

2.2.2 Dense Representations

To overcome the limitation of one-hot representation, Bengio et al. (2003) proposed a neural language model to produce distributed representations for words, which are called word embeddings. Following works on creating word embeddings by training neural networks with a large corpus show that the trained word embeddings can be used for various downstream NLP tasks (Collobert and Weston, 2008; Mikolov et al., 2013; Pennington et al., 2014b). However, the limitation of these pre-trained word embeddings is that they produce the same representations for the same words that are used to deliver different meanings. In other words, these pre-trained embeddings fail at capturing the context of the text.

2.2.3 Contextualised Representations

Contextualised word embeddings are proposed to mitigate the limitation of dense word embeddings by producing different representations for the same words that are used in different contexts. For example, a word *bank* can be used to deliver different meanings, such as either "the land alongside or sloping down to a river or lake" or "a financial establishment that uses money deposited by customers for investment." Early approaches use bidirectional Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) to learn contextualised word vectors by combining sequential representations that are conditioned on forward and backward directions (Melamud et al., 2016; Peters et al., 2017; McCann et al., 2017; Peters et al., 2018). Following works (Radford et al., 2018; Devlin et al., 2019) use Transformer architecture (Vaswani et al., 2017) that is built from attention mechanism (Bahdanau et al., 2015; Luong et al., 2015) to improve the long-term dependency and parallelise training process.

2.3 Modern Neural Architectures

In this section, we will briefly the recent progress modern neural network architectures that play critical roles in NLP.

2.3.1 Sequential models

Text data consists of a sequence of words or characters. Several neural network models are particularly designed for sequential modelling, such as recurrent neural networks (RNN)(Rumelhart et al., 1985). The main idea of RNN is to use a loop (recurrent connection) that allows information to be passed to the next time step for sequential modelling. However, one problem of RNN is that it has limited long-term dependency because of gradient vanishing. The gradient vanishing issue means that the training signals become too small as the length of the sequence becomes longer and a model fails at learning long-term relationships in the data.

To address this, LSTM (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Cho et al., 2014b) are proposed. The key idea of LSTM and GRU is to use multiple gates (e.g., forget gate and input gate) to decide how much information should be kept or removed while passed to the next step. Thanks to this gating mechanism, LSTM variants are widely used for sequential modelling problems (Greff et al., 2016).

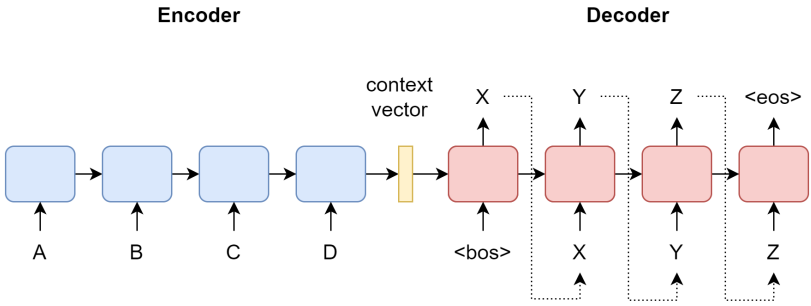


Figure 2.3: Illustration of a sequence-to-sequence architecture consisting of an encoder and a decoder. <bos> and <eos> refer special tokens for *beginning of sentence* and *end of sentence*, respectively.

2.3.2 Sequence-to-Sequence Models

Another important model architecture in the NLP domain is sequence-to-sequence (seq2seq) architecture. Seq2seq architecture is proposed for a task where a sequence of input is mapped to a sequence of output, such as machine translation task (Sutskever et al., 2014). Figure 2.3 illustrates an example of seq2seq architecture. Basic seq2seq architecture consists of two separate sequential models (e.g., RNN or LSTM), called an encoder and a decoder, respectively. An encoder takes the entire input sequence and compresses it into a fixed length vector, which is called a context vector. Then a decoder takes the context vector and produces a sequence of outputs in an auto-regressive way. The drawback of this approach is that the input information is encoded into one vector requiring a large compression when the input sequence is long. Therefore, a decoder is required to re-cover a lot of information from a single vector.

2.3.3 Attention Mechanism

Attention mechanism (Bahdanau et al., 2015) is proposed to address the problem of a single context vector that captures the entire input sequence in a sequence-to-sequence setting. A key concept is to produce a different context vector for each output element by using weights, which are often called attention scores. For example, an attention layer, which is parameterised by a simple feed-forward network, produces weights and the produced weights are used to create a context vector, which is a weighted average of a sequence of input, per each output element.

Formally, a seq2seq model consisting of an encoder and a decoder network takes a input sequence $\boldsymbol{x} = [x_1, x_2, ..., x_n]$, and produces a output sequence $\boldsymbol{y} = [y_1, y_2, ..., y_m]$. The encoder network (e.g., RNN) produce a sequence of hidden state $\boldsymbol{h} = [h_1, h_2, ..., h_n]$. The decoder network (e.g., RNN) produce a hidden state $\boldsymbol{s}_t = f(\boldsymbol{s}_{t-1}, y_{t-1}, \boldsymbol{c}_t)$ at time step $t \in \{1, ..., m\}$, where the context vector $\boldsymbol{c}_t$ is a sum of hidden states of the input sequence $\boldsymbol{h}_i$, weighted by alignment scores $a_{t,i}$:

$$\boldsymbol{c}_t = \sum_{i=1}^{n} a_{t,i} \boldsymbol{h}_i \tag{2.4}$$

where alignment scores $a_{t,i}$ are computed by an attention layer. The attention layer calculates how well current output y_t and inputs x_i are aligned by computing attention scores $a_{t,i}$ for pairs (y_t, x_i):

$$a_{t,i} = \text{align}(y_t, x_i) \tag{2.5}$$

$$= \frac{e^{\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i)}}{\sum_{j=1}^{n} (e^{\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_j)})} \tag{2.6}$$

There are many variants of scoring functions (Luong et al., 2015; Vaswani et al., 2017). In the original attention paper by Bahdanau et al. (2015), the scoring function is parameterised as a feed-forward neural network and defined as:

$$\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i) = \boldsymbol{v}_{\boldsymbol{a}}^T \tanh(\boldsymbol{W}_a[\boldsymbol{s}_{t-1}; \boldsymbol{h}_i]) \tag{2.7}$$

where $\boldsymbol{v_a}$ and $\boldsymbol{W}_a$ are trainable weight matrices which are jointly trained with the sequence-to-sequence model during training.

Attention mechanism becomes very successful not only in machine translation but also in computer vision field (Xu et al., 2015).

2.3.4 Transformer

Transformer (Vaswani et al., 2017) is also a seq2se2 model that follows an encoder-decoder structure. The main difference between Transformer and other seq2seq models is that Transformer removes recurrent connections to rely entirely on an attention mechanism to capture global dependencies between input and

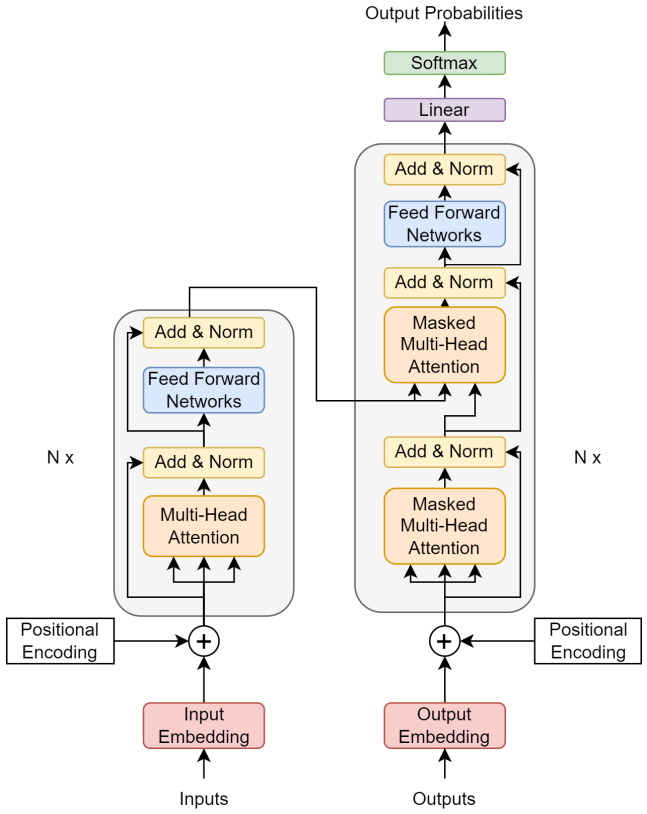


Figure 2.4: The transformer architecture. The left half shows the encoder part and the right half shows the decoder part. More details can be found in the original paper (Devlin et al., 2019)

output. This eliminates the sequential nature and allows parallelization of the training procedure, which results in an improvement in batching.

Figure 2.4 illustrates an overview of the transformer architecture. The encoder of the Transformer consists of $N = 6$ identical layers of which each layer has two sub-layers including multi-head self-attention mechanism and a position-wise fully connected feed-forward network. There is a residual connection (He et al., 2016) around each of the two sub-layers, followed by layer normalisation (Ba et al., 2016). The decoder of the Transformer also consists of $N = 6$ identical layers of which each layer contains three sub-layers.

The encoder's input vector $\boldsymbol{x}_i \in \mathbb{R}^d$ is used to create three vectors, a query

vector $\boldsymbol{q}_i \in \mathbb{R}^{d_k}$, a key vector $\boldsymbol{k}_i \in \mathbb{R}^{d_k}$, and a value vector $\boldsymbol{v}_i \in \mathbb{R}^{d_v}$. These vectors are created by multiplying the input vectors by three trainable matrices (i.e., $\boldsymbol{W}_q \in \mathbb{R}^{d \times d_q}$, $\boldsymbol{W}_k \in \mathbb{R}^{d \times d_k}$, and $\boldsymbol{W}_v \in \mathbb{R}^{d \times d_v}$) that we trained during the training process. For computation efficiency, matrices of queries Q, keys K, and values V are used to compute the attention function. Transformer uses scaled dot-product attention, which is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{2.8}$$

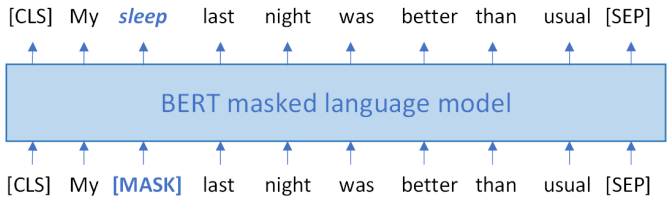
This attention function is deployed multiple times by using h different heads, which is called multi-head attention. Each head has a learned linear projection layer that takes queries, keys, and values and maps another feature space. Then Attention function in each head is applied to these projected queries, keys and values. The final output of multi-head attention is the concatenation of the attention output of each head. The benefit of multi-head attention is that it encourages the model to attend to information at different positions.

Similar to other seq2seq models, Transformers also utilises the learned embeddings to translate a token into a vector representation. However, attention-based architecture lacks the sequential information of the input since it eliminates the recurrent connections. To mitigate this, Transformer uses positional encoding to indicate the position of an entity in an input sequence by using sine and cosine functions of different frequencies. Then these positional embeddings are added into word embeddings to inject information about the position of each token in the sequence.

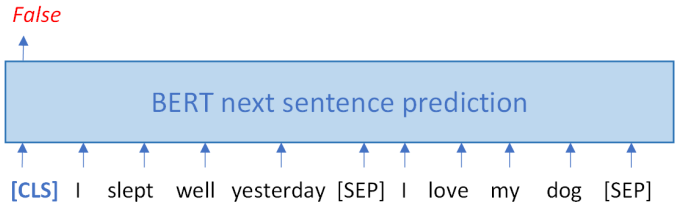
2.3.5 BERT

BERT is built from an encoder part of Transformer to create bi-directional representations. BERT can be easily adapted to downstream tasks achieving state-of-the-art results in various benchmarks (Devlin et al., 2019). Because of its ability to be fine-tuned to a wide range of downstream tasks, BERT became a model of choice in the past few years (Rogers et al., 2020).

The strength of BERT comes from two novel pre-training tasks that encourage a model to learn how to produce deep contextualised representations. Figure 2.5 illustrates the two pre-training tasks of BERT. One pre-training task is masked language modelling (MLM) which asks a system to predict randomly masked input tokens. MLM forces a system to focus on information within the sentence by utilising information from surrounding words. The other pre-training task is next sentence prediction (NSP) which asks a system to predict whether two



(a) Masked language modelling task.



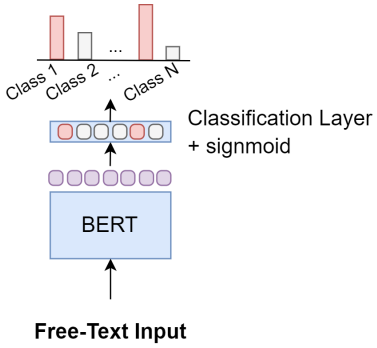
(b) Next sentence prediction task.

Figure 2.5: Illustrations of pre-training tasks of BERT. `[CLS]` and `[SEP]` are special tokens used for classification and separation, respectively. Final classification layers, which are parameterised as single-layer feedforward neural networks, are omitted in the images.

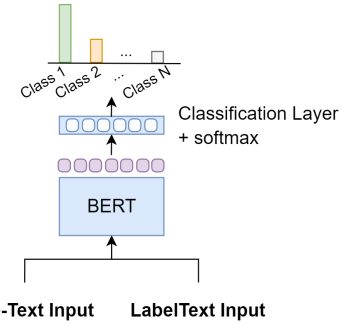
input sentences are adjacent to each other or not. NSP encourages a system to learn coherency by capturing the relation between sentences.

The input and output formulation of BERT is as follows: each input word is tokenized into sub-word tokens by using Wordpiece tokeniser (Wu et al., 2016). A special token (`[CLS]`) is inserted at the beginning of each input sequence. When creating sequence pairs for NSP, two sequences are combined with a special token (`[SEP]`) in between. Then three embedding layers are employed to create token embeddings, segmentation embeddings, and position embedding, respectively. The final embeddings are the sum of these three embeddings.

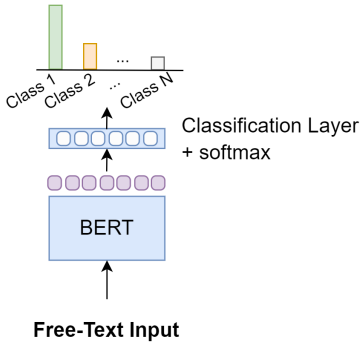
For fine-tuning, a fully connected layer is added on top of the final encoder layer. Typically, the output of the final encoder layer corresponding to a `[CLS]` token is used for classification. Therefore, the final representation of `[CLS]` token is fed into a final classification layer. Figure 2.6 summarises different fine-tuning tasks in this dissertation. For multi-label classification, sigmoid functions are used as activation functions in the final classification layer with multiple neurons (Ch. 3). For sentence-pair classification and multi-class classification, softmax function is used as an activation function in the final classification layer (Ch. 4, Ch. 5). For a binary classification approach, a softmax function is used for the



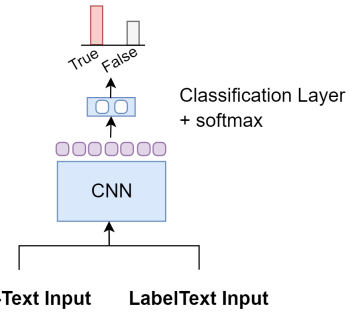
(a) Multi-label classification (Ch. 3).



(b) Sentence-pair classification (Ch. 4).



(c) Multi-class classification (Ch. 5).



(d) Binary classification (Ch. 6).

Figure 2.6: Illustrations of different fine-tuning tasks in this dissertation.

final classification layer with two neurons (i.e., True or False) (Ch. 6)[1].

[1]Please note that the CNN model is used in the final chapter (Ch. 6) because clinical documents are typically very long and the computational complexity increases quadratically as the length of the input sentence increases when using BERT.

Chapter 3

Assessment: Sleep Issue Classification

This chapter was previously published as:

Shim, H., Luca, S., Lowet, D., & Vanrumste, B. (2020, March). Data augmentation and semi-supervised learning for deep neural networks-based text classifier. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pp. 1119-1126.

In this chapter, we introduce a use case of sleep issue assessment and study how to train a neural networks model to extract mentioned sleep issues and parse them into a set of pre-defined issue categories. Specifically, we aim to train the model in a low-resource setting when a small-sized labelled set is given as an initial training set and a large-sized unlabelled set is additionally given as an additional training set. For this, we study how to exploit both labelled data and unlabelled data without an additional labelling budget by using data-efficient methods. Moreover, we investigate when and how data-efficient methods are beneficial in terms of performance improvement and how they affect the performance of the model.

For this study, we collected a custom dataset that contains free-text data about sleep issues for experiments and empirically evaluated different data-efficient methods (i.e., data augmentation and semi-supervised learning) by varying the size of the initial training set. We show that each data-efficient method can improve the performance of the model. Finally, we propose the best model,

which is a combination of data augmentation and semi-supervised learning. Performance analyses show that increasing the size of samples with minority classes is critical for performance improvement.

This chapter studies the following research questions:

RQ1. How can we fine-tune a pre-trained language model when only a small-sized training set for the target task is available?

RQ2. How can we fine-tune a pre-trained language model when only a small subset of the target dataset is labelled?

3.1 Introduction

User feedback contains rich information about the users and is essential for user-driven development. Many products are providing in-app survey and collecting feedbacks from the users to identify their needs for the better quality of service and support. Especially, open-ended questions are good for understanding user-specific problems. Unlike a closed-ended question that provides pre-defined options limiting the users' answer, the open-ended questions allow the users to answer it in free-text format such that they can answer based on their situation and feeling (Dohrenwend, 1965). The answers to these open-ended questions can be used to obtain detailed information on the users.

One of the biggest challenges with analysing free-text is how to automate the process. Manually analysing free-text is labour-intensive and not suitable as the amount of user feedback is increasing. In this case, developing a free-text analysis tool with the help of recent advances of deep neural networks could be a solution. However, there are technical challenges in applying the deep neural networks to the real-world application: one is labelling data for training the model. Data labelling is a time-consuming and tedious task and it requires a lot of human and financial resources (Chui et al., 2018). Moreover, since the labels are prone to be added or deleted as a new batch of user feedback is obtained, the data labelling process is expected to be repeated frequently throughout product development. Minimised manual labelling could mitigate these issues.

In this paper, we focus on the topic of sleep. The goal is to understand user-specific situations and problems via free-text to provide personalised coaching service to users who want to optimise their nights of sleep. As the first step, we collected experimental data containing pairs of a free-text sentence and a set of sleep issues via a web-based survey (Section 3.3). To automate analysing these

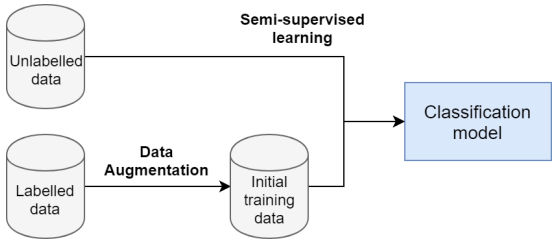


Figure 3.1: Overview of the proposed approach.

free-text data, we aim to build a neural networks-based text classifier with the limited number of labelled data. In this paper, we propose a method which is a combination of data augmentation and semi-supervised learning as shown in Figure 3.1 (Section 3.4). We evaluate our method and show the proposed method achieves similar performance while reducing the amount of labelled data (Section 3.5). Also, we analyse the error of the model and investigate the effects of the proposed method (Section 3.6).

3.2 Related Work

Neural Language Model for NLP Tasks

One of the breakthroughs in neural networks based natural language processing (NLP) is attention mechanism (Bahdanau et al., 2015; Luong et al., 2015). The attention mechanism is firstly proposed to solve long term dependency problems of sequential models (Hochreiter and Schmidhuber, 1997; Cho et al., 2014a) that use a single context vector compressing every input from previous time steps. Attention mechanism allows the models to take hidden states from several time steps as inputs and calculate the degree of importance regarding the current time step's input. After Vaswani et al. (2017) proposed Transformer architecture with sorely attention mechanisms, Transformer architecture has been widely used for language models (Radford et al., 2018; Devlin et al., 2019) to capture complex linguistic patterns. These pre-trained language models can be easily fine-tuned on various downstream NLP tasks (Yang et al., 2019; Wu and Hu, 2018; Liu et al., 2019). However, it is widely accepted that the performance of the neural networks-based model is highly dependent on the size and the quality of training data.

Data Augmentation for Language Data

Data augmentation is a technique that can increase the size of a data. Many researchers have been working on data augmentation in various fields, including vision (Krizhevsky et al., 2012) and speech (Ragni et al., 2014). Compared to these fields, data augmentation for language is less studied and there is no standard method yet. Some researchers proposed data augmentation methods for language data, including synonym replacement by using a thesaurus (Zhang and Wallace, 2017), similar word replacement by using a pre-trained word embedding (Wang and Yang, 2015), contextual word replacement by using a pre-trained language model (Kobayashi, 2018), and sentence rephrase by back-translation (Sennrich et al., 2016). However, these techniques are computationally expensive compared to their performance gain. Because of this reason, simple text editing operations are commonly used in practice. Recent research empirically shows that simple text editing operations could contribute substantially to improvements in various text classification tasks (Wei and Zou, 2019). We will explain this simple text editing method in Section 3.4.2

Semi-Supervised Learning

Semi-supervised learning focuses on leveraging both labelled and unlabelled data to build a better classifier. Pseudo-labelling, also known as self-training, is a type of semi-supervised learning methods which is used to add more labels with iterative training. In spite of its simplicity, using these pseudo-labels can improve classifier's performance, especially when there are little labelled training data (Lee, 2013). However, since the classifier uses its predictions to teach itself, pseudo-labelling might reinforce the initial model's error (Zhu, 2005).

3.3 Data

For experiments, free-text data was collected via a web-based survey. Participants was asked to fill in questionnaires in free-text sentences and select sentences representing to their answers. The free-text responses to the open-ended questions will be used as input for the classification model, while the selected sentences will be used as ground truth labels to train and validate the model. In the following subsections, we explain the data collection protocol and provide initial data analysis result.

Table 3.1: Example of question and answer. Red coloured text shows a spelling error.

Question	What is going on with your sleep?
Answer	I mainly have tow problems. The first is that it's hard for me to stay asleep for more than about an hour without waking up. The other problem is I sometimes have trouble either going to sleep or getting back to sleep once I wake up.

3.3.1 Participants

We recruited American adults for the survey by using Amazon's Mechanical Turk (MTurk) platform. Before participating in the survey, participants were informed of the background, purpose, and legal basis of the survey and their rights. When participants signed up for participating in the study, they received the link to the web page hosting the survey. Additional inclusion criteria were applicable:

Inclusion criteria for subject selection

- They are 18 years or older
- They have an MTurk-approval rate of 97% or higher (this means that at least 97% of the prior tasks they completed on MTurk were of acceptable quality)
- They are proficient in English
- They are willing and able to provide informed consent

3.3.2 Survey Questions

Participants described issues related to their sleep with at least one complete sentence. Beforehand, the participants were provided with a guide to imagine that they are sitting at the doctor's office because they are having some sleep issues. Table 3.1 illustrates an example of the question and a user's answer. As we can see, the answer contains spelling errors. After describing sleep-related issues, participants were asked to select at most 3 sentences that capture the meaning of their answers from Table 3.2. Labels of the selected sentences are used as ground truth labels in experiments.

Table 3.2: Options for selecting matched sentences.

Label	Sentence
troubleFallingAsleep	I lie in bed awake have trouble falling asleep
troubleStyaingAsleep	I have been waking up frequently
wakeUpTooEarly	I am waking up too early (before I want/have to)
staysUpLate	I am staying up (too) late
sleepsInLater	I am sleeping in (too) late
problemWakingUp	I have trouble waking up
SnoringBothersMe	I am bothered by my snoring
SnoringBothersOthers	Others are bothered by my snoring
SnoringStoppedBreathing	I stop breathing during the night
otherIssue	I have another concern
goodSleep	I have no sleep concern

3.3.3 Data Analysis

In total 16,096 sentences were collected. We split data into train and test set: the train set consists of 14,363 samples and the test set consists of 1,733 samples. Figure 3.2 illustrates the label distribution of train set. It is observed that the data distribution is highly skewed: it has imbalances between labels and between single- and multiple-labelled data.

3.4 Method

3.4.1 Classification Model

Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) is used as a baseline text classifier. We use a pre-trained BERT model and fine-tune on our data to detect multiple sleep issues from the given free-text input. As it is illustrated in Figure 3.3, we add a dense layer on the top of the pre-trained BERT model and the final hidden vector of the classification token [CLS] is fed into this dense layer. To perform multi-label classification, the sigmoid function is used for an activation function and binary cross-entropy is used as a loss function. For more details on tokenization and BERT model, please refer the original paper (Devlin et al., 2019).

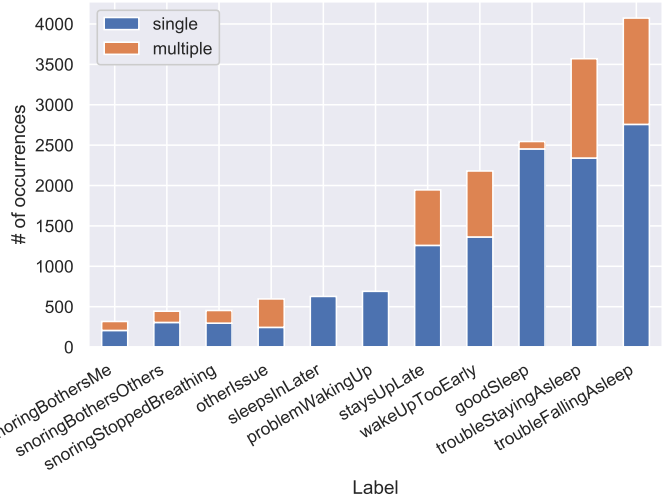


Figure 3.2: Label distribution of the train set. Blue graphs represent when the sample is single-labelled and orange graphs represents when the sample is multi-labelled.

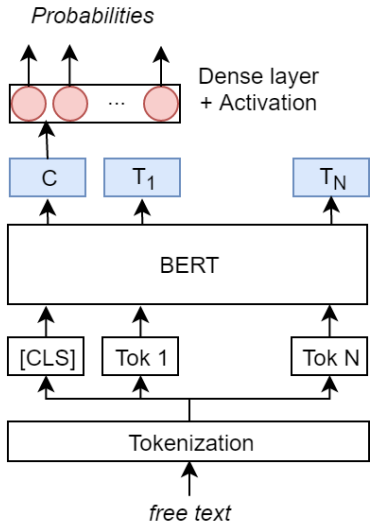


Figure 3.3: Overview of the BERT model for multi-label classification.

Table 3.3: Examples of text editing operations.

Operation	Text
Original	I snore a lot.
Synonym replacement	I snore a ~~lot~~ entirely.
Random noise injection	I snore~~t~~ a lot.
Random swap	snore I a lot.
Random deletion	I snore a ~~lot~~.

3.4.2 Data Augmentation

We use Easy Data Augmentation (EDA) technique (Wei and Zou, 2019) which consists of four different text editing operations:

- **Synonym replacement**: N words are randomly selected from the sentence and replaced with one of its synonyms chosen at random.
- **Random noise injection**: N words are randomly selected from the sentence and a single character of each word is replaced with a random alphabetical character.
- **Random swap**: we randomly choose two words in the sentence and swap their positions and repeat N times.
- **Random deletion**: N words from the sentence are randomly chosen and removed from the sentence.

The value of N is decided based on the length of each sentence. We set a percentage $p = 0.1$ and calculated $p \times len(sentence)$, where the number words in the sentence is used as a length of the sentence. Rounded up value of $p \times len(sentence)$ is used as the value of N.

During data augmentation, we select sample sentences consisting of more than 5 words to avoid too short sentences. Four operations are applied separately to each sentence. For synonym replacement, we only select a word that contains more than two characters to avoid selecting too short words. Also, unlike the original paper (Wei and Zou, 2019), we insert random noise rather than a synonym. This can be seen as introducing misspelling to make the model robust to a spelling error, which is common in user-generated free-text. Table 3.3 illustrates examples of text editing operations.

3.4.3 Pseudo-Labelling

We use pseudo-labelling (Zhu, 2005) as a semi-supervised learning method. During pseudo-labelling procedure, a classifier is trained on the initial labelled data and then the trained model is used to do classify unlabelled data. The predicted labels with a high confidence score, which are called pseudo-labels, are then added to the new training set. For selecting pseudo-labels, we set a threshold value of 0.6. Then the classifier is re-trained with the new training set consisting of the initial labelled and the pseudo-labelled data. This process is repeated until it reaches a certain termination condition. In this paper, we set the termination condition based on the number of iterations and the size of the pseudo-labelled data. Until the number of iterations reaches the limit, which is set as 5, we check the size of the pseudo-labelled data. If the size of the pseudo-labelled data is not bigger than the pseudo-labelled data from the previous step, the pseudo-labelling process is terminated. Algorithm 1 illustrates the pseudo-labelling procedure.

Algorithm 1: Pseudo-labelling procedure

Data: Training set D_t, labelled set D_l, unlabelled set D_u
Result: New training set $\hat{D}_t$, trained model M_i

$D_t \leftarrow D_l$
$i = 0$
termination condition $\leftarrow$ False
while *termination condition == False* **do**
    $M_i \leftarrow \text{train}(D_t)$
    predictions $\leftarrow$ inference(M_i, D_u)
    $D_p \leftarrow$ thresholding(predictions)
    $\hat{D}_t \leftarrow D_t + D_p$
    termination condition $\leftarrow$ check condition(D_p, i)
    if *termination condition == False* **then**
        $D_t \leftarrow \hat{D}_t$
        $i += 1$
    end
end

3.5 Experiments and Results

To validate our method, 4 experiments were conducted: Firstly, we check the baseline model's performance without data augmentation and pseudo-labelling.

Secondly, we apply data augmentation and train the model with augmented data. Thirdly, we apply pseudo-labelling and iteratively train the model. Lastly, we train the initial model with augmented data and iteratively train the model with pseudo-labels. The purpose of these experiments is to evaluate how the proposed method can contribute to performance improvement.

Evaluation Metric

In our experiment, we use f1 score for each label as an evaluation metric, which is defined as follows:

$$\begin{aligned} Precision &= \frac{tp}{tp + fp} \\ Recall &= \frac{tp}{tp + fn} \\ F1 &= 2 \times \frac{precision \times recall}{precision + recall} \end{aligned} \tag{3.1}$$

where tp, fp, and fn represent true positive, false positive, and false negative of each label, respectively.

Additionally, we use marco-, micro-averaged f1 scores (Sorower, 2010). Macro-averaged f1 is per label averaged and does not take label imbalance into account. Micro-averaged f1 is calculated by counting the total true positives, false negatives, and false positives so that it would be more affected by the performance of the classes which has more examples.

Settings

All experiments were performed on the Windows 10 operating system. The detailed specification of hardware and software is summarized in Table 3.4. We used PyTorch version of BERT (Huggingface). The smallest model, whose size of the final hidden vector of classification token is 512, was used for the experiments with pre-trained model weights. Softmax function in the final output layer was changed to sigmoid function to perform multi-label classification. We did not change other hyperparameter settings except the number of training epochs: we trained the model on training datasets for 10 epochs.

Table 3.4: Detailed implementation specification.

Item	Specification
CPU	Intel®Xeon®W-2123 CPU @ 3.60 GHz
GPU	NVIDIA GeForce GTX 1080 ti, 11 GB memory
Graphic driver	NVIDIA graphic driver version 416.34
CUDA	Version 10.0
OS	Windows 10, 64-bit
Python	Version 3.6.6
Pytorch	Version 1.0.1

3.5.1 Baseline Model

We trained the BERT with the entire training data. Table 3.5 shows the result. It is observed that the trained model achieved varying performances over labels: it achieves relatively high performance on some labels (e.g., *troubleFallingAsleep*, *troubleFallingAsleep*, and *goodSleep*) that occurred more in train set than other labels (e.g.,*otherIssue* and *snoringBothersMe*) where the model achieves low performances. This result implies that the trained model tends to performs well on some labels with more training data compared to other labels with less training data. We call these labels with relatively few training samples as minority labels. This results in a large difference between macro- and micro-averaged performances. In our case, the macro-averaged f1 score is suitable for evaluating the model's performance. Because micro-averaged performance might mislead interpretation that the trained model works well even though it misclassifies minority labels.

We further investigate how the size of the training set affects the model's performance. We trained model with the following training set fractions (%): {10, 30, 50, 70, 90, 100} whose sizes (k) are: {1.4, 4.3, 7.2, 10.0, 12.9, 14.3}. In Figure 3.4, the red dashed line shows the performance of the model based on the size of the training set. Interestingly, the model achieves nearly 90% of performance upper limit - that can be achieved when around 14,300 samples are used - with only using around 4,300 samples. Also, we can see that the size of the training set does not have a significant impact on the model's performance after around 7,200. This shows that after some number of data, the performance tends to be saturated. Details of the training set size and its performance are described in the following Section 3.6.2.

Table 3.5: Classification results when using entire training data.

Label	Precision	Recall	F1
troubleFallingAsleep	0.79	0.84	0.82
troubleStayingAsleep	0.78	0.79	0.79
wakeUpTooEarly	0.74	0.68	0.71
staysUpLate	0.74	0.65	0.69
problemWakingUp	0.75	0.74	0.75
sleepsInLater	0.64	0.57	0.60
snoringBothersOthers	0.82	0.65	0.73
snoringBothersMe	0.67	0.38	0.49
snoringStoppedBreathing	0.78	0.56	0.65
goodSleep	0.97	0.94	0.96
otherIssue	0.42	0.25	0.31
Macro-averaged	0.74	0.64	0.68
Micro-averaged	0.79	0.75	0.77

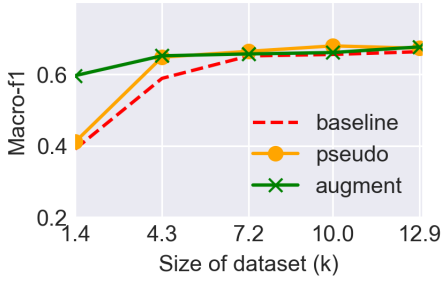


Figure 3.4: Performances based on various dataset sizes. X-axis represents the size data used for training. Y-axis represents macro-averaged f1 score.

3.5.2 Data Augmentation

We investigated the effect of data augmentation while varying the size of the training data. In Figure 3.4, the green line shows the performance of the model with data augmentation. From Figure 3.4, it can be seen that the data augmentation provides the largest performance increase when the training data is the smallest: it is observed that macro-f1 score is increased from 39.13% to 59.7% when only using 1,400 samples. However, the amount of performance improvement decreases as the training size increases. This suggests that the best scenario to apply data augmentation is when the only small size of data is available for training. Details of additional training data made by data augmentation are given in the following Section 3.6.2.

3.5.3 Pseudo-Labelling

We iteratively trained a model with a subset of training data as a labelled data and the remaining as an unlabelled data by applying pseudo-labelling. We investigate how the size of the labelled training data could affect the final model's performance. In Figure 3.4, the orange line shows the performance of the model trained with pseudo-labelling. Similar to the data augmentation result, the performance improvement tends to decrease as the size of labelled data increases. The largest improvement is observed when around 4,300 samples are given as labelled data: the model use 10,000 of unlablled data with pseudo-labels achieves a macro-f1 score of 64.8% while the model trained without pseudo-labels achieves 58.9%. One noticeable thing is that when only around 1,400 samples are given as a labelled data, there is almost no performance increase even after iterative training with 12,900 unlabelled samples with pseuo-labels: it only increases from 39.1% to 41.1%. We will discuss this in the following Section 3.6.3.

3.5.4 Data Augmentation + Pseudo-Labelling

We apply data augmentation to the initially given labelled data and iteratively train the model by using unlabelled data with pseudo-labels. As it can be seen from the Table 3.6, the baseline model, without data augmentation and pseudo-labelling, achieves macro-averaged f1 of 39.1% when around 1,400 of labelled data are given for training. If we train the model with 1,400 of labelled data and 11,900 of unlabelled data with pseudo-labels, it slightly improves the performance to 41.2%. Compared to this, if data augmentation is applied to the 1,400 of labelled data and the model is iteratively add more training data with

Table 3.6: Details of models' training data sizes and compared performances. Comparison between a baseline model (BASE), PL (a model with pseudo-labelling), DA (model with data augmentation), and DA +PL (a model with data augmentation and pseudo-labelling)

Model	Labelled	Augmented	Unlabelled	Macro-f1
BASE	1,436	0	0	0.39
PL	1,436	0	12,927	0.41
DA	1,436	5,607	0	0.60
DA + PL	1,436	5,607	12,927	**0.63**

pseudo-labels, it achieves macro-averaged f1 score of 62.7%. However, it can be interpreted that this improvement is mainly derived from data augmentation, because the model trained with 1,400 of labelled data and additional augmented data without pseudo-labels already achieves macro-averaged f1 score of 59.7%. In other words, data augmentation can contribute to performance improvement significantly when there is a little amount of labelled data. On top of that, pseudo-labelling can provide additional performance increase by using unlabelled data.

3.6 Discussion

3.6.1 Misclassification Analysis

To analyse the misclassification, firstly we plot a confusion matrix based on predictions made by the baseline model from Section 3.5.1. Since our case is multi-label classification, we select samples whose ground truth label set contains only single labels. We added *otherMisclassification* label, which means that the trained model predicted more than one labels. As it is shown in Figure 3.5, misclassification happens more often in between similar labels: the trained model often misclassifies *snoringBothersMe* as *snoringBothersOthers* and sometimes fails to distinguish *problemWakingUp* and *sleepsInLater*. We speculate that this is because samples with these labels are too close to be distinguished from each other. This suggests avoiding pre-defining too similar labels for classification.

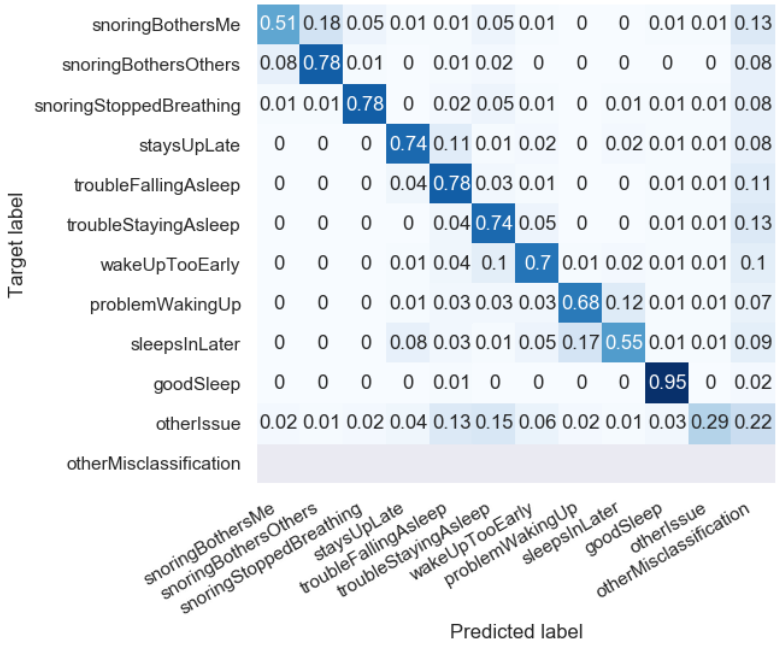


Figure 3.5: Normalised confusion matrix of the trained model's predictions. A row represents target label, whereas a column represents predicted label. The values of the diagonal elements represent the degree of correctly predicted classes.

3.6.2 Data Augmentation Result Analysis

We investigate the effects of the size of the training set and data augmentation on the model's performance. Table 3.7 shows the size of each fraction with and without data augmentation and performance with each dataset. To analyse the effect of data augmentation, two comparisons are given: model trained with augmented data when (1) data augmentation is only applied to training samples of minority labels or (2) data augmentation is applied to entire data. Minority labels mean the labels with relatively few training samples, including *snoringBothersMe, snoringBothersOthers, snoringStoppedBreathing, otherIssue, sleepsInLater, and problemWakingUp*. In Table 3.7, min and max represent a minimum and a maximum number of training samples per label, respectively. For example, in 10% of data, the smallest label set (*snoringBothersMe*) consists of 29 samples and the largest label set (*troubleFallingAsleep*) consists of 399 samples. From the Table 3.7, we can see that even when data augmentation is

Table 3.7: Size of training set and data augmentation result and trained model's performance.

Percent of dataset	Min	Max	Total	Macro-f1
10%	29	399	1436	0.39
30%	98	1,221	4309	0.59
50%	167	2,044	7,182	0.65
70%	219	2,829	10,054	0.66
90%	297	3,623	12,927	0.66
100%	317	4,073	14,363	0.68
Data augmentation on data of minority labels				
10% + data augmentation	133	486	2,555	0.56
30% + data augmentation	460	1,376	7,170	0.60
50% + data augmentation	774	2,126	11,385	0.64
70% + data augmentation	1,005	2,750	15,326	0.66
90% + data augmentation	1,331	3,371	18,970	0.65
100% + data augmentation	1,551	5,260	26,206	0.68
Data augmentation on entire data				
10% + data augmentation	133	1,946	7,043	0.60
30% + data augmentation	458	5,913	20,809	0.65
50% + data augmentation	776	9,708	34,059	0.66
70% + data augmentation	1,006	13248	47,024	0.66
90% + data augmentation	1,335	16749	59,595	0.68
100% + data augmentation	1,557	20,283	71,377	0.69

applied to only data of minority labels, the model's performance is similar to when data augmentation is applied to entire data which means that there are more than two times many training samples. This implies that what plays a key role in data augmentation is the number of training samples of minority labels, not the total size of the training set.

3.6.3 Pseudo-Labelling Result Analysis

In previous Section 3.5.3, we showed that the model trained with 1,400 labelled data and 12,900 unlabelled data with pseudo-labels achieves the almost same performance of the model trained without pseudo-labels. We hypothesise that this is because the initial model trained with 1,400 samples with ground truth is not robust enough to get sufficient pseudo-labels. To validate this, we investigate the number of pseudo-labels obtained by using the initial model. As it is shown in Figure 3.6, there is almost no pseudo-labelled data of minority labels. This means that no additional samples of minority labels will be added to the new

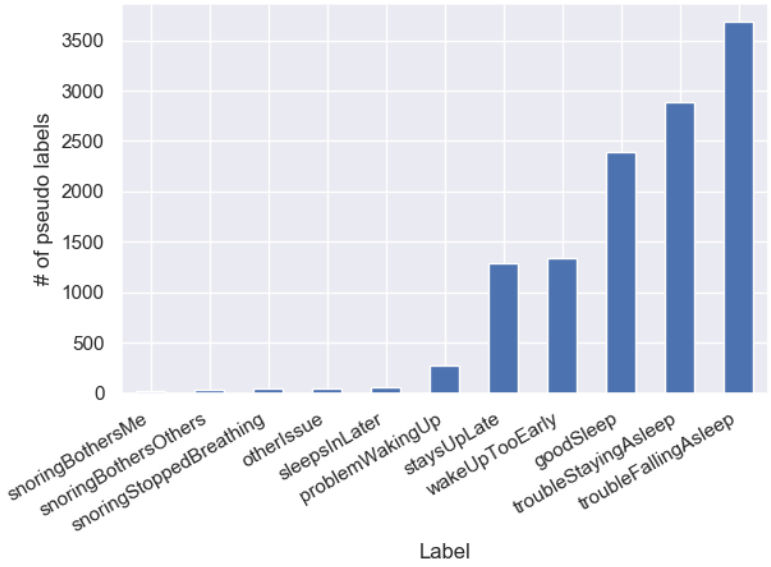


Figure 3.6: The number of pseudo-labelled data obtained by using the initial model trained with 1,400 labelled data.

training set for the next iteration. This could enhance the data imbalance and result in poor performance at the end of iterative training. This suggests that the initial model's performance, especially for minority labels, is critical in the pseudo-labelling method.

Another observation from analysing pseudo-labelling result is that the termination condition of the size of pseudo-labelled data is not strict enough: as shown in Table 3.8, during the iterative training the size of pseudo-labelled data is increasing, but with negligible margin after the first iteration. Therefore, the iteration was repeated until it met the termination condition of the iteration number, which is set as 5 times in this paper. In future work, adding a margin for the termination condition of the size of pseudo-labelled data is foreseen to avoid unnecessary iterations.

Table 3.8: The number of pseudo-labels and increase over iterative training.

Iteration	Pseudo-labels	Increase
1	10,059	10,059
2	11,705	1,646
3	12,139	434
4	12,344	205
5	12,451	107

Table 3.9: Details of each model's training set, approximated training time, and performance. Comparison between a baseline model (BASE), PL (a model with pseudo-labelling), DA (model with data augmentation), and DA with PL (a model with data augmentation and pseudo-labelling).

Model	#. train data	Time	Macro-f1
BASE with 100% of data	14,363	45m	0.68
DA on 100% of data	71,377	3h30m	0.69
BASE with 10% of data	1,436	5m	0.39
DA on 10% of data	7,043	20m	0.60
PL with 10% of data	$\leq$ 14,363	$\leq$ 45m	0.41
DA on 10% of data + PL	$\leq$ 19,970	$\leq$ 20m + 45m	0.63

3.6.4 Data Augmentation and Pseudo-Labelling Efficiency Analysis

To evaluate the efficiency of the proposed method, we investigate the computation power required to train each model. Table 3.9 summarises the required training time for each model and its training data. For pseudo-labelling, the values of training set and training time are for a single iteration and the value of the performance is the final model's performance. It is observed that data augmentation on 100% of data does not contribute to performance increase significantly when considering its increase of training time. Unlikely, data augmentation on 10% of data provides a relatively high performance boost with only around 15 minutes of training time increase. For pseudo-labelling, it seems not efficient compared to data augmentation, because it requires multiple training sessions. As it is described in Section 3.6.3, considering the number of newly added pseudo-labels sharply decreases after the first iteration, the best scenario is to train the model with augmented data and conduct pseudo-labelling only 1-2 times.

3.7 Conclusion

In this paper, we propose a method which is a combination of data augmentation and semi-supervised learning to reduce manual data labelling process for developing a deep neural networks-based text classification model. To validate our method, experiments on how each method could contribute to the performance improvement with various settings were conducted. We experimentally showed that applying data augmentation can improve the model's performance, especially when there is little training data. Also, the result shows that the size of minority labels is critical to the model's performance when the training data is imbalanced. Furthermore, using unlabelled data with pseudo-labels can provide additional performance improvement. However, for the pseudo-labelling, the training time increases as the training sessions are iterated. These results suggest two possible scenarios: Firstly, develop an initial model with augmented data when there is little training data. Secondly, apply pseudo-labelling when there is additional data which is not labelled yet and iterate the process only 1-2 times. We expect this method can boost the development process by reducing manual data labelling.

Chapter 4

Coaching: Aspect-Based Sentiment Analysis

This chapter was previously published as:

Shim, H., Lowet, D., Luca, S.,& Vanrumste, B. (2021). LETS: A Label-Efficient Training Scheme for Aspect-Based Sentiment Analysis by Using a Pre-Trained Language Model. *IEEE Access, 9*, pp. 115563-115578.

In this chapter, we introduce a use case for analysing user reviews of a behaviour change coaching programme. There are two goals: one is to understand fine-grained user experience (i.e., sentiment polarities towards multiple pre-defined aspects). The other is to build a scalable data labelling algorithm so that we can train a machine learning model with a limited labelling budget. Since a scalable data labelling algorithm is also beneficial when the labelling scheme is updated (i.e., adding or deleting labels) after the training, we expect that an active learning framework can address the low-resource issue by effectively improving the performance. To this end, we design experiments to test this hypothesis.

Firstly, we analyse the general active learning frameworks and identify that active learning algorithms do not utilise unlabelled data or labelled data for fine-tuning. Another interesting observation is that active learning algorithms perform poorly when the amount of labelled data is small, which is referred to as a cold-start issue. In addition to this, when the training dataset is imbalanced, there might be performance differences between majority classes, frequent label

classes in the training set, and minority label classes, rare label classes in the training set. We observed that these performance differences result in biased sample selection when an active learning algorithm uses a trained model's prediction. In other words, the active learning algorithm fails to select the informative samples with rare label classes since a model is not fully trained to classify rare classes.

To this end, we propose a novel active learning framework that consists of multiple components for not only effectively reducing manual labelling efforts but also maximising the utility of data. Experimental results show that the proposed method outperforms other the-state-of-the-art active learning algorithms by achieving 2 times faster performance improvement in a low-resource setting and better generalisability. Lastly, we apply the proposed method to another benchmark dataset from another domain and show the effectiveness of the proposed method.

This chapter studies the following research questions:

RQ1. How can we fine-tune a pre-trained language model when only a small-sized training set for the target task is available?

RQ2. How can we fine-tune a pre-trained language model when only a small subset of the target dataset is labelled?

4.1 Introduction

Recently proposed pre-trained language models (Devlin et al., 2019; Radford et al., 2018; Yang et al., 2019) have shown their ability to learn contextualised language representations and can be easily fine-tuned to a wide range of downstream tasks. Even though these language models can be trained without manually labelled data thanks to the self-supervised pre-training paradigm, large-scale labelled datasets are required for fine-tuning to downstream tasks. Data labelling can be labour-intensive and time-consuming creating a bottleneck in the development process of machine learning applications. Moreover, in real-world scenarios, the labelling scheme can be changed by adding or changing labels after deployment. Therefore, it is critical to be able to fine-tune the model with a limited number of labelled data to reduce manual labelling efforts and foster fast machine learning applications development.

One of the possible solutions is to apply active learning to reduce manual labelling efforts. Active learning is an algorithm designed to effectively minimise

manual data labelling by querying the most informative samples for training (Settles, 2009). Active learning has been extensively studied (Dasgupta and Hsu, 2008; Settles, 2009) and applied to various applications, from image recognition (Wang et al., 2016b; Gal et al., 2017) to natural language processing (NLP) tasks (Shen et al., 2017; Siddhant and Lipton, 2018). Even though active learning guides how to strategically annotate unlabelled data, it does not utilise the unlabelled data or labelled data for fine-tuning. For example, unlabelled data points can be used for self-supervised learning or already labelled data points can be further utilised during supervised learning, such as by using data augmentation techniques.

To not only effectively reduce manual labelling efforts but also maximise the utility of data, we propose a novel **L**abel-**E**fficient **T**raining **S**cheme, LETs in short. The proposed LETS integrates three elements as illustrated in Fig. 4.1: (i) a task-specific pre-training to exploit unlabelled task-specific corpus data; (ii) label augmentation to maximise the utility of labelled data; and (iii) active learning to strategically prioritise unlabelled data points to be labelled. In this paper, we apply LETS to a novel aspect-based sentiment analysis (ABSA) use-case for analysing the reviews of a mobile-based health-related program. The introduced health-related program is designed to support people to improve their sleep quality by restricting sleep-related behaviour. We aim to provide a tailored program by analysing reviews of individual experience. To the best of our knowledge, this is the first attempt to implement an automated ABSA system for health-related program reviews. To illustrate the success of the novel use-case, we have collected a new dataset and experimentally show the effectiveness of the proposed LETS with the collected dataset and a benchmarks dataset.

The main contributions of this paper include the followings:

- A novel use-case of natural language processing and machine learning techniques for the healthcare domain is introduced (Sec. 4.3);
- A novel label-efficient training scheme that integrates multiple components is proposed (Sec. 4.4);
- A label augmentation technique is proposed to maximise the utility of labelled data (Sec. 4.4.2);
- A new query function is proposed to search different boundaries with two uncertainty scores for active learning with the imbalanced dataset (Sec. 15);

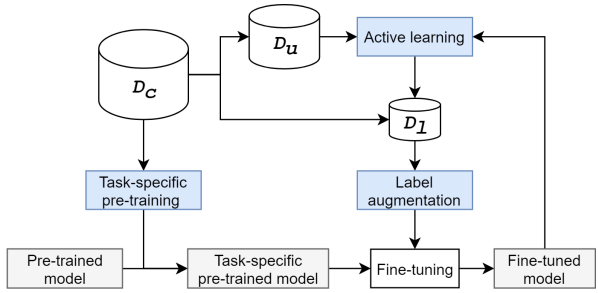


Figure 4.1: Overview of the proposed **L**abel-**E**fficient **T**raining **S**cheme (LETS). Task-specific pre-training utilises unlabelled task-specific corpus data set D_c. Label augmentation exploits labelled data set D_l. Active learning algorithm selects data from the unlabelled data set D_u for manual labelling.

- A new evaluation metric for an ABSA system is proposed to correctly evaluate the performance of a system in the end-to-end framework (Sec. 4.5.3).

4.2 Related Work

4.2.1 Aspect-Based Sentiment Analysis

ABSA is a special type of sentiment analysis that aims to detect opinion toward fine-grained aspects. Since ABSA can capture insights about user experiences, ABSA has been widely studied in various industries, from consumer product sector (Xu et al., 2019; Do et al., 2019) to service sector (Ruder et al., 2016; Wang et al., 2016c; Brun and Nikoulina, 2018; Sun et al., 2019a). ABSA entails two steps: aspect category detection and aspect sentiment classification (Pontiki et al., 2014). During the first step, Aspect Category Detection (ACD), a system aims to detect a set of the pre-defined aspect categories that are described in the given text. For example, in the domain of restaurant review, the pre-defined set of aspects can be {Food, Price, Service, Ambience, Anecdotes/Miscellaneous} and the task is to detect {Price, Food} out of the text "This is not a cheap place but the food is worth to pay". During the second step, Aspect Category Polarity (ACP), a system aims to classify a text into one of sentiment polarity labels (i.e., Positive, Negative, Neutral, etc) given a pair of text and aspect categories. For example, the task to produce a set of pairs, such as {(Price, Negative), (Food, Positive)} *given the set of ground truth categories* {Price, Food} and the text.

There has been significant improvement in ABSA systems over the past few years thanks to the recent progress of deep neural network (DNN) based NLP models, (Ruder et al., 2016; Wang et al., 2016c; Xue and Li, 2018; Xu et al., 2019; Sun et al., 2019a). For example, Sun et al. (2019a) propose a Bidirectional Embedding Representations from Transformers (BERT) (Devlin et al., 2019) based ABSA system by casting an ABSA task as a sentence-pair classification task. Even though this sentence-pair approach shows the state-of-the-art performance by exploiting the expanded labelled data set with sentence-aspect conversion[1] (Sun et al., 2019a), it still requires a large amount of labelled data.

Later, Xu et al. (2019) propose a post-training to utilise unlabelled corpus datasets to further train a pre-trained model for ABSA. The proposed post-training exploits both the general-purpose corpus dataset (i.e., texts from Wikipedia) and task-related corpus dataset (i.e., reading comprehension dataset) for the end task (i.e., review reading comprehension). Xu et al. (2019) showed utilising multiple unlabelled corpus datasets can enhance the performance of the end task. Extensive studies on utilising unlabelled corpus for further pre-training showed that the importance of using domain-relevant data (Sun et al., 2019b; Gururangan et al., 2020). However, domain-related corpus datasets for further pre-training are possibly not available in some domain (e.g., healthcare) because of privacy issue[2].

4.2.2 Active Learning Algorithm

Active learning that aims to select the most informative data to be labelled has been extensive studied (Lewis and Gale, 1994; Lewis and Catlett, 1994; Dasgupta and Hsu, 2008; Settles, 2009). The core of active learning is a query function that computes score for each data point to be labelled. Existing approaches include uncertainty-based (Shelmanov et al., 2019; Dor et al., 2020), ensemble-based (Lakshminarayanan et al., 2017; Beluch et al., 2018), and expected model change-based methods (Settles, 2009). Thanks to their simplicity, uncertainty-based methods belong to the most popular ones. Uncertainty-based methods can use least confidence scores (Shen et al., 2017; Wu et al., 2020; Lewis and Gale, 1994), max margin scores (Balcan et al., 2007; Gonsior et al., 2020), or max entropy scores (Shannon, 1948) for querying.

[1] As it is described in the original paper (Sun et al., 2019a), a sentence s_i in the original data set can be expanded into multiple sentence-aspect pairs (s_i, a_1), (s_i, a_2), $\cdot \cdot \cdot$, (s_i, a_N) in the sentence pair classification task, with aspect categories a_n where $n \in \{1, 2, .., N\}$.

[2] For example, General Data Protection Regulation (GDPR) includes the purpose limitation principle mentioning that personal data be collected for specified, explicit, and legitimate purposes, and not be processed further in a manner incompatible with those purposes (Article 5(1)(b), GDPR).

One of the earliest studies of active learning with DNN is by Wang et al. (2016b) for image classification. They proposed a Cost-Effective Active Learning (CEAL) framework that uses two different scores for querying. One is an uncertainty score to select samples to be manually labelled. And the other is a certainty score to select samples to be labelled with pseudo-labels which are their predictions. Both scores are computed based on the output of DNN. Wang et al. (2016b) showed that the proposed CEAL works consistently well compared to the random sampling, while there is no significant difference in the choice of uncertainty measures, among the least confidence, max-margin, and max entropy.

However, other researchers claim that using the output of DNN to model uncertainty could be misleading (Gal and Ghahramani, 2016; Gal et al., 2017). To model uncertainty in DNN, Gal and Ghahramani (2016) proposed Monte Carlo (MC) dropout as Bayesian approximation that performs dropout (Srivastava et al., 2014) during inference phase. Later, Gal et al. (2017) incorporated uncertainty obtained by MC dropout with Bayesian Active Learning by Disagreement (BALD) (Houlsby et al., 2011) to demonstrate a real-world application of active learning for image classification. Also, Shen et al. (2017) applied BALD to an NLP task and experimentally showed that BALD slightly outperforms the traditional uncertainty method that uses the least confidence scores. The results from the large-scale empirical study by Siddhant and Lipton (2018) also showed the effectiveness of BALD for various NLP tasks. Even though BALD outperforms the random sampling method, the differences between BALD and active learning methods with the traditional uncertainty scores (i.e., least confidence, max margin, and max entropy) are marginal (Shen et al., 2017; Siddhant and Lipton, 2018). Also, BALD is computationally more expensive than the traditional methods because it requires multiple forward passes. Therefore, the traditional uncertainty scores are reasonable options when deploying active learning in a real-world setting.

Practical concerns on how to implement active learning in real-world settings include the issue that a model can perform poorly when the amount of labelled data is minimal (Reker, 2020). This issue is referred to as the cold-start issue. Ideally, active learning could be most useful in low-resource settings. In practice, however, it is more likely that the model might work poorly with the limited number of labelled data at the beginning of active learning (Yuan et al., 2020). Therefore, introducing a component to ensures a certain level of performance with the limited labelled data is important to address the cold-start issue.

Table 4.1: An example of aspect-based sentiment analysis based on the free-text user review of a health-related program.

	Example
Free-text	I noticed that I was losing weight, but I missed the mid-afternoon caffeine boost most days. I slogged my way through work in the afternoon hours and missed the caffeine then, although I did sleep better.
Aspect	Energy: Negative Missing caffeine: Negative Sleep quality: Positive

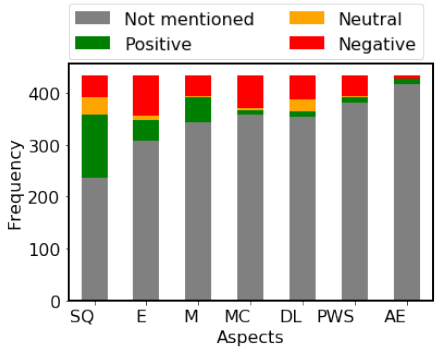
4.3 Aspect-Based Sentiment Analysis for Health-Related Program Reviews

This section describes a mobile-based health-related program use-case that we call Caffeine Challenge. To conduct aspect-based sentiment analysis on the reviews of Caffeine Challenge, an experimental dataset is collected and annotated. The next subsections explain the details of the use-case, data collection protocol, and data labelling scheme with the initial data analysis result.

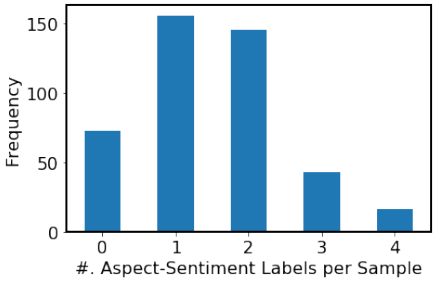
4.3.1 Caffeine Challenge Use-Case

In this study, we introduce a health-related program that is designed to help people improve their sleep quality by restricting behaviour that might negatively affect their sleep quality. Having caffeinated beverage or desserts during the late afternoon and evening is selected as a target behaviour for this study. A challenge rule is restricting a caffeine intake after 13:00 for two weeks. During the program, participants use a mobile application to log their progress and receive notifications and recommendations of relevant information. At the end of the program, an in-app chatbot (conversational agent) asks about challenge experience and the participants are allowed to provide answers in free-text sentences. Our goal is to understand users' sentiments towards different aspects of the program by analysing the review data. To this end, we aim to develop an automated ABSA system for health-related program reviews as illustrated in Table. 4.1 where a system detects opinions (sentiment polarity) expressed towards multiple aspects. Since the ABSA system can capture detailed user opinions, it can be used to tailor the health-related program to individual users.

4.3.2 Experimental Data collection



(a) Sentiment class distribution per aspect category. Due to limited space, we use the following abbreviations: Sleep Quality (SQ), Energy (E), Mood (M), Missing Caffeine (MC), Difficulty Level (DL), Physical Withdrawal Symptoms (PWS), and App Experience (AE). Green, yellow, red, and grey bars indicate the number of samples with *Positive, Neutral, Negative*, and *Not mentioned* labels, respectively.



(b) Distribution of the number of aspect-sentiment labels per text excluding *Not mentioned* labels. The number of aspect-sentiment labels per sentence indicates the number of aspect categories mentioned in the sentence.

Figure 4.2: Annotation result of the collected Caffeine Challenge dataset. Sentiment class distribution per aspect category (a) and the number of aspect-sentiment labels per text (b) are shown.

In the real-world machine learning application implementation process, multiple cycles on iterative development are often required: firstly, implementing a baseline model with experimental data and then gradually updating the model with real-world data. To develop the first version of the ABSA system, we conducted a pilot study with a semi-realistic dataset that is collected from an online survey via a crowd-sourcing platform (Amazon Mturk). At the beginning of the survey, an instruction containing details of the Caffeine Challenge (i.e., its purpose, goal, procedure, and consent form), is given to the survey participants. Then each participant has received a questionnaire regarding the experience of the Caffeine Challenge. Then the participants have requested to answer the questions by imagining that they have done this challenge. In total, we recruited 1,000 participants and collected 12,000 answers and examples of collected data are shown in Appendix 4.A.

4.3.3 Data Labelling

We annotated a random subset of the collected data for aspect-based sentiment analysis. Based on both health-related program and app development perspectives, seven different aspects are defined:

1. Sleep Quality (SQ)
2. Energy (E)
3. Mood (M)
4. Missing Caffeine (MC)
5. Difficulty Level (DL)
6. Physical Withdrawal Symptoms (PWS)
7. App Experience (AE)

Each aspect category is annotated with one of the sentiment values as follows: Positive, Neutral, Negative, and Not Mentioned. Not Mentioned class is introduced as a placeholder for an empty sentiment value. For example, when a sample does not describe any opinion towards a specific aspect, then it is labelled as Not Mentioned for that aspect category. A labelling scheme of each aspect category is given in Appendix 4.B.

Fig. 4.2 illustrates annotation results and Fig. 4.3 shows the example of annotated data point. As it is shown in Fig. 4.2a, the majority of sentiment label within all aspect categories is an empty sentiment label (Not Mentioned). Some categories (Sleep Quality, Energy, and Mood) appeared more frequently compared to other categories (Missing Caffeine, Difficulty Level, Physical Withdrawal Symptoms, and App Experience). The former group is denoted as majority aspect categories and the latter group is denoted as minority aspect categories. Fig. 4.2b shows the distribution of the number of aspect-sentiment labels per text, excluding Not Mentioned labels. It is observed that most of the annotated texts have either one or two aspect-sentiment labels and only a few have more than three aspect-sentiment labels.

4.4 Label-Efficient Training Scheme for Aspect-Based Sentiment Analysis

We develop an automated ABSA system by utilising a pre-trained language model. Also, a label-efficient training scheme is proposed to reduce effectively

```
{
        'sentence':'I noticed that I was losing
            weight, but I missed the mid-afternoon
            caffeine boost most days. I slogged my
            way through work in the afternoon hours
             and missed the caffeine then, although
             I did sleep better.',
        'labels': {
                'sleep_quality': 'positive',
                'mood' : 'not_mentioned',
                'energy' : 'negative',
                'missing_caffeine': 'negative',
                'difficulty_level': 'not_mentioned'
                    ,
                'physical_withdrawal_symptoms': '
                    not_mentioned',
                'app_experience': 'not_mentioned',
        }
}
```

Figure 4.3: An example of annotated data. Each annotated data point includes free-text and labels which are pairs of aspect category and sentiment class.

manual labelling efforts. The following subsections will explain the ABSA system and the proposed label-efficient training scheme in detail.

4.4.1 Aspect-Based Sentiment Analysis System

Similar to the previous work by Sun et al. (2019a), we reformulate ABSA task as sentence-pair classification by using a pre-trained language model, BERT (Devlin et al., 2019). Fig. 4.4 illustrates a sentence-pair classification approach for ABSA. As shown in the figure, the proposed ABSA system produces the probability distribution over sentiment classes C, including polarised sentiment classes S (e.g., Positive, Neutral, Negative, etc) and an empty placeholder (e.g., Not Mentioned), for the given free-text sentence x_i and aspect category a_k. This formalisation allows a single model to perform aspect category detection and aspect sentiment classification at the same time. Also, adding an aspect category as the second part of input can be seen as providing a hint to the model where to attend for creating a contextualised embedding. Moreover, this formalisation allows expanding the training data set by augmenting labelled data, which will be explained in the following section (Sec. 4.4.2).

Formally, an input is transformed into a format of $\boldsymbol{x}_i^k = [$`[CLS]`$, \boldsymbol{x}_i,$ `[SEP]`$, \boldsymbol{a}_k,$ `[SEP]`$]$, where $\boldsymbol{x}_i = [w_i^1, w_i^2, ..., w_i^{n_i}]$ is the tokenised i-th free-text,

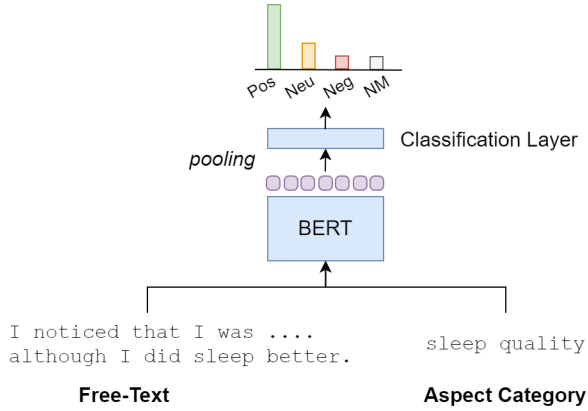


Figure 4.4: Illustration of aspect-based sentiment analysis (ABSA) as a sentence-pair classification by using Bidirectional Embedding Representations from Transformer (BERT).

$\boldsymbol{a}_k = [a_k^1, a_k^2, ..., a_k^{m_k}]$ is the tokenised k-th aspect category in K aspect categories, and `[CLS]` and `[SEP]` are special tokens indicating classification and separation, respectively. Then the input is fed to the BERT model (f_θ) that produces contextualised embeddings for each token by using multi-head attention mechanism (Devlin et al., 2019). The contextualised embedding vector $\boldsymbol{e}_i^k \in \mathbb{R}^{d \times 1}$, corresponding to the classification token `[CLS]`, is used as the final representation of the given input $\boldsymbol{x}_i^k$. Then a classification layer projects $\boldsymbol{e}_i^k$ into the space of the target classes:

$$\boldsymbol{e}_i^k = f_\theta(\boldsymbol{x}_i^k) \tag{4.1}$$

$$\hat{\boldsymbol{y}}_i^k = \text{softmax}(\boldsymbol{W} \cdot \boldsymbol{e}_i^k + b) \tag{4.2}$$

where $\hat{\boldsymbol{y}}_i^k \in [0,1]^{|C|}$ is the estimated probability distribution over the sentiment classes C for the given free-text sample x_i and aspect category a_k pair, and f_θ, $\boldsymbol{W} \in \mathbb{R}^{|C| \times d}$, and $b \in \mathbb{R}^{|C|}$ are trainable parameters.

4.4.2 Label-Efficient Training Scheme

One of the bottlenecks in developing an ABSA system with a pre-trained language model is to create a large-scale labelled task-specific dataset for fine-

tuning which requires a labour-intensive manual labelling process. To mitigate this issue, we propose a **L**abel-**E**fficient **T**raining **S**cheme, which we refer as *LETS*. The proposed LETS consists of three elements to effectively reduce manual labelling efforts by utilising both unlabelled and labelled data. Fig. 4.1 illustrates the overview of the proposed LETS. The first element is task-specific pre-training to exploit the unlabelled task-specific corpus data. The second element is label augmentation to maximise the utility of the labelled data. The third element is active learning to efficiently prioritise the unlabelled data for manual labelling. The followings will describe the details of each element.

Task-specific pre-training

Task-specific pre-training is used to exploit the unlabelled task-specific corpus data. We adopt a novel pre-training strategy of Masked Language Modelling (MLM) from BERT (Devlin et al., 2019) to train an Attention-based model to capture bidirectional representations within a sentence. More specifically, during the MLM training procedure, the input is formulated with a sequence of tokens that are randomly masked out with a special token `[MASK]` at a certain percentage p. Then the training objective is to predict those masked tokens. Since ground truth labels are original tokens, MLM training can proceed without manual labelling.

Label augmentation

Label augmentation is proposed to not only address the cold-start issue in active learning but also to maximise the utility of the labelled data. The proposed label augmentation algorithm multiplies the labelled data set by replacing aspect categories with similar words. This might look similar to common data augmentation techniques proposed by Wei and Zou (2019) that includes synonym replacement, random insertion, random swap, and random deletion. Our method, however, modifies only the second part of the input (i.e., aspect category) while keeping the original free-text part. The proposed label augmentation technique is applied to pre-defined aspect categories with polarised sentiment classes S (e.g., Positive, Neutral, Negative, etc). Algorithm 2 summarises the proposed label augmentation technique.

Active learning

Active learning is used to prioritise the unlabelled data points to be manually labelled and added to the training pool. The core of active learning is a query

Algorithm 2: Label augmentation

Data: Labelled training set D_l, a dictionary of similar words per aspect category $Dict$, polarised sentiment classes S
Result: Augmented training set $\hat{D}_l$

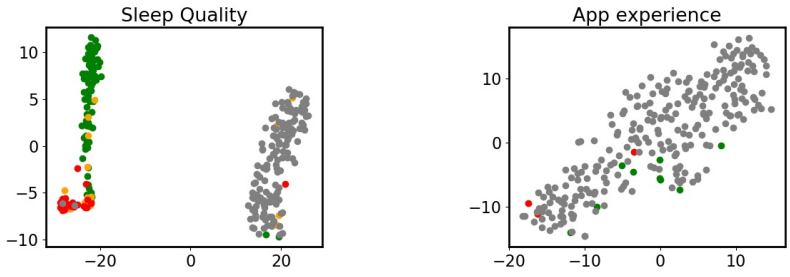
$\hat{D}_l \leftarrow D_l$
for $d_l \in D_l$ **do**
  $txt \leftarrow$ getFreeText(d_l)
  $asps \leftarrow$ getAspects(d_l)
  for $asp \in asps$ **do**
    $senti \leftarrow$ getSentimentLabel(d_l, asp)
    if $senti \in S$ **then**
      $\hat{asps} \leftarrow Dict(asp)$
      for $\hat{asp} \in \hat{asps}$ **do**
        $\hat{d}_l \leftarrow (txt, \hat{asp}, senti)$
        $\hat{D}_l \leftarrow$ addData($\hat{d}_l$)
      end
    end
  end
end

function that scores the data points to use a labelling budget effectively in terms of performance improvement.

Even though most of the existing active learning methods consider balanced datasets, one typical feature of a real-world dataset is that it can be imbalanced (Ertekin et al., 2007). As it is shown in Sec. 4.3.3, the collected dataset is also highly imbalanced: there are majority aspect categories that more often appear in the training set and minority aspect categories that less often appear in the training set. We observe that a fine-tuned ABSA model performs differently towards majority and minority aspect classes. For example, Fig. 4.5 illustrates the vector representations before the final classification layer[3] plotted into 2-dimensional space by using a dimensionality reduction algorithm (Van der Maaten and Hinton, 2008). From the figure, it is observed that the fine-tuned model can create distinctive representations between sentiment labels within the Sleep Quality aspect category, while the model fails to learn to differentiate data points among sentiment classes and empty sentiment class within the App Experience aspect category. This shows that a fine-tuned ABSA model performs relatively well towards majority aspect categories and its prediction is reliable,

[3]The fine-tuned model at the initial step of active learning experiment (Sec. 4.5.4) is used.

whereas a model works poorly towards minority aspect categories and it tends to fail to even detect the aspect categories.



(a) (b)

Figure 4.5: The final vector representations of inputs plotted in 2-dimensional space for Sleep Quality (a) and App Experience (b) aspect categories. Green, yellow, red, and grey colour indicate inputs with *Positive, Neutral, Negative*, and *Not Mentioned* sentiment labels, respectively. All data points were not used during the training phase.

Therefore, we propose two uncertainty measures for majority aspect categories and minority aspect categories, respectively:

$$u_{major} = 1 - Pr(\hat{y}_i^k = \arg\max_{c \in \mathcal{C}}(\hat{\boldsymbol{y}}_i^k) | \boldsymbol{x}_i^k) \tag{4.3}$$

$$u_{minor} = 1 - |Pr(\hat{y}_i^k = nm | \boldsymbol{x}_i^k) - \sum_S (Pr(\hat{y}_i^k = s | \boldsymbol{x}_i^k))| \tag{4.4}$$

$$= 1 - |1 - 2Pr(\hat{y}_i^k = nm | \boldsymbol{x}_i^k)| \tag{4.5}$$

where $Pr(\hat{y}_i^k = \arg\max_{c \in \mathcal{C}}(\hat{\boldsymbol{y}}_i^k) | \boldsymbol{x}_i^k)$ is the highest probability in the estimated probability distribution over sentiment classes given $\boldsymbol{x}_i^k$, nm refers *Not Mentioned*, and S refers a polarised sentiment classes set (e.g., Positive, Neutral, Negative, etc). u_{major} is the traditional least confidence score and u_{minor} is the margin of confidence score between an empty placeholder (i.e., Not Mentioned) and sum of other sentiment classes. As it is shown in (4.5), u_{minor} treats the model's prediction as binary classification result (i.e., *Not Mentioned* or *Mentioned*) producing high uncertainty scores when $Pr(\hat{y}_i^k = nm | \boldsymbol{x}_i^k)$ is close to 0.5. The intuition of introducing u_{minor} is allowing a model to focus on detecting whether the aspect category is mentioned or not. The proposed two uncertainty measures allow the model to search different boundaries during

active learning: the boundaries where the model is uncertain about its aspect category sentiment classification result towards majority classes is described by u_{major}. And the boundary where the model is uncertain about aspect category detection result towards minority classes is described by u_{minor}.

Algorithm 3 shows the proposed LETS that integrates three elements. Firstly, a pre-trained model is further pre-trained with an unlabelled task-specific corpus data set. Then the task-specific pre-trained model is used for initialisation during active learning iterations. Active learning is repeated t times and each time a model is fine-tuned with the labelled data set that is augmented by the proposed label augmentation technique. At the end of each iteration step, n samples are queried from the unlabelled set for manual labelling. For querying, each Query function Q_{major} and Q_{minor} select $n/2$ samples where u_{major} and u_{minor} are the highest, respectively.

Algorithm 3: Label-efficient training scheme (LETS)

Data: Pre-trained model M_{pt}, unlabelled task-specific corpus data set D_c, initial training set D_l, unlabelled training set D_u, total iteration t, labelling budget n, query function for majority categories Q_{major}, query function for minority categories Q_{minor}

Result: Fine-tuned model M_t, Labelled data set D_t

$M_{tspt} \leftarrow$ task-specificPre-train(M_{pt}, D_c,)
$i = 0$
$D_i \leftarrow D_l$
while $i < t$ & $|D_u| > 0$ **do**
  $D'_i \leftarrow$ augmentLabel(D_i)
  $M_i \leftarrow$ fineTune(M_{tspt}, D'_i)
  $d_{major} \leftarrow Q_{major}(D_u, M_i, n/2)$
  $d_{minor} \leftarrow Q_{minor}(D_u, M_i, n/2)$
  $D_{i+1} \leftarrow D_i$
  $D_{i+1} \leftarrow$ addData(addLabels($d_{major} \cup d_{minor}$))
  $D_u \leftarrow D_u - \{d_{major} \cup d_{minor}\}$
  $i+ = 1$
end

4.5 Experiments

4.5.1 Datasets

We evaluate the proposed method on two datasets. One is the custom dataset that we collected for the Caffeine Challenge use-case. The other is SemEval-2014 (Pontiki et al., 2014) task 4 dataset[4] that is the most widely used benchmark dataset for aspect-based sentiment analysis.

Custom dataset: Caffeine Challenge

The custom dataset, which is described in Sec. 4.3, is named as a Caffeine Challenge dataset. We annotate a random subset of the Caffeine Challenge dataset with 7 different aspect categories (i.e., Sleep Quality, Energy, Mood, Missing Caffeine, Difficulty Level, Physical Withdrawal Symptoms, App Experience) and 3 sentiment labels S ={Positive, Neutral, Negative} and an empty placeholder (i.e., Not Mentioned). The aspect categories distribution of the Caffeine Challenge dataset is imbalanced as described in Sec. 4.3. Aspect categories are divided into subgroups of majority and minority aspect categories based on the frequency in a training set: {Sleep Quality, Energy, Mood} as majority aspect categories and {Missing Caffeine, Difficulty Level, Physical Withdrawal Symptoms, and App Experience} as minority aspect categories.

The unlabelled corpus data set are used for task-specific pre-training and the annotated data set is used for fine-tuning. Table 4.2 summarises the sizes of the Caffeine Challenge dataset used for the experiments. For task-specific pre-training, sentences from the unlabelled corpus data set are used. For the fine-tuning, 5-fold cross-validation splits are created at the sentence level and sentence-aspect pairs are used for training.

Benchmark dataset: SemEval

The SemEval-2014 task 4 dataset contains restaurant reviews annotated with 5 aspect categories (Food, Price, Service, Ambience, Anecdotes/Miscellaneous) and 4 sentiment labels S ={Positive, Neutral, Negative, Conflict[5]}. Since the SemEval dataset is also imbalanced, as illustrated in Appendix. 4.C, we define majority and minority categories: {Food, Anecdotes/Miscellaneous}

[4]https://alt.qcri.org/semeval2014/task4/

[5]The conflict label applies when both positive and negative sentiment is expressed about an aspect category (Pontiki et al., 2014)

Table 4.2: Size of Caffeine Challenge dataset used for the experiments. Sentences from the unlabelled corpus data set used as the task-specific corpus data for task-specific pre-training. S-A pairs indicate sentence-aspect pairs and sentence-aspect pairs from the training set are used for fine-tuning.

Data set	Sentence	S-A pairs
Unlabelled corpus	22,577	-
Training	325	2,275
Test	87	609
Total Fine-tuning	412	2,884

Table 4.3: Size of SemEval dataset used for the experiments. Sentences from the training set are used as the task-specific corpus data for task-specific pre-training. S-A pairs indicate sentence-aspect pairs and sentence-aspect pairs from the training set are used for fine-tuning.

Data set	Sentences	S-A pairs
Training	2,435	12,175
Test	609	3,045
Total	3,044	15,220

and {Service, Ambience, Price} as majority and minority aspect categories, respectively.

We used the original SemEval train set for the experiments to create 5-fold cross-validation splits. Table 4.3 summarises the size of SemEval dataset used for the experiments. For task-specific pre-training, sentences from the training set are used. For the fine-tuning, sentence-aspect pairs are created with an empty placeholder (Not Mentioned) for the sentences that do not contain a sentiment label towards specific aspect categories.

4.5.2 Experimental Settings

Task-specific pre-training and fine-tuning

We use the pre-trained uncased BERT-base model as the pre-trained model (PT). The task-specific pre-trained model (TSPT) is created by further training the pre-trained model on the task-specific corpus data with the masked-language modelling (MLM) objective with masking probability $p = 0.15$. The TSPT is used to initialise the proposed method and the PT is used to initialise other methods during the active learning process. For fine-

tuning, the final classification layer is added and entire model parameters are updated. More detailed implementation and hyperparameter settings are given in Appendix. 4.D.

Label augmentation

Label augmentation multiplies the amount of labelled data by generating synthesised pairs of sentence and aspect categories by replacing aspect categories with similar words. The pre-defined dictionary containing a list of similar words is used for label augmentation and label augmentation is applied to the only minority aspect categories to avoid inefficient augmentation. The pre-defined dictionaries are given in Appendix 4.E.

Active learning

Active learning experiments are repeated 5 times with 5-fold cross-validation splits. At each fold, the initial labelled data set (i.e., seed data) is randomly selected from the training set at the sentence level and transformed into sentence-aspect pairs. For the Caffeine Challenge dataset, 20% of the training set (n=455) is used as seed data (D_l) and the remaining data is used as unlabelled data (D_u). For the SemEval dataset, 10% of the training set (n=1,220) is used as seed data (D_l) and the remaining data is used as unlabelled data (D_u). Active learning is iterated with 10 steps with a fixed labelling budget (n=$|D_u|/10$). At the initial iteration step (t=0), a model is trained on the seed data. During active learning steps, more data are iteratively added to the training set by selecting unlabelled data.

For comparison, we implemented BALD by using MC dropout (Gal and Ghahramani, 2016), Cost-Effective Active Learning (CEAL) (Wang et al., 2016b), least confidence scores, and random sampling. For BALD, we use the same approximation by Siddhant and Lipton (2018) to compute uncertainty score as the fraction of models which disagreed with the most popular choice. The number of stochastic forward passes for BALD is set to 10. For CEAL, the least confidence score is used for calculating uncertainty and the threshold for pseudo-labelling is set to 0.05 with a decay rate of 0.0033. Since pseudo-labels are not included in the labelling budget, the active learning with CEAL can be terminated early when there is no more data for manual labelling. More details of these methods can be found in the original papers (Siddhant and Lipton, 2018; Wang et al., 2016b).

Table 4.4: Types of error used to compute aspect category sentiment classification (ACSC) scores. TP, NA, FN1, FN2, FP refer to true positive, not applicable, false negative type 1, false negative type 2, false positive, respectively. TARG and PRED refer to a target sentiment class and a predicted sentiment class where S is a set of polarised sentiment classes (e.g., Positive, Neutral, Negative, etc).

Error type	Target	Prediction	Comparison
TP	TARG $\in S$	PRED $\in S$	TARG = PRED
NA	Not Mentioned	Not Mentioned	TARG = PRED
FN1	TARG $\in S$	Not Mentioned	TARG $\neq$ PRED
FN2	TARG $\in S$	PRED $\in S$	TARG $\neq$ PRED
FP	Not Mentioned	PRED $\in S$	TARG $\neq$ PRED

4.5.3 Evaluation Metrics

In this paper, we used two different metrics to evaluate the performance of an ABSA system. One metric is aspect category detection (ACD) and the other metric is aspect category sentiment classification (ACSC). Aspect category detection (ACD) is proposed by Pontiki et al. (2014) and limited to evaluating aspect category detection ignoring the performance of aspect category sentiment classification. Aspect category polarity (ACP) metric is proposed to assess the sentiment classification performance of a system (Pontiki et al., 2014). However, as it is mentioned in the previous study by Brun and Nikoulina (2018), the ACP metric presumes the ground truth aspect categories and cannot be used to correctly evaluate an ABSA system end-to-end. To address this issue, we introduce a new metric of aspect category sentiment classification (ACSC) which is the modified version of ACP taking into account false aspect category detection results.

Aspect category detection (ACD)

ACD is used to evaluate how a system accurately detects a set of aspect categories mentioned in the input text. F_1 score is used which is defined as:

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R}$$

where precision (P) and recall (R) are:

$$P = \frac{|E \cap G|}{|E|}, R = \frac{|E \cap G|}{|G|}$$

where $|*|$ denotes the cardinality of a set *, E is the set of aspect categories that a system estimates for each input, and G is the set of the target aspect categories. Micro-F_1 scores are calculated at sentence-level and averaged over all inputs and macro-F_1 scores are calculated and averaged at aspect category-level.

Aspect category sentiment classification (ACSC)

ACSC is used to evaluate the performance of an ABSA system end-to-end. Since the proposed ABSA system produces multiple sentence-pair predictions for a single text input, the predictions are aggregated to compute (aspect, polarity) pairs at sentence-level while eliminating the pairs that contain Not Mentioned as a target as well as a predicted sentiment class. F_1 scores are calculated on the (aspect, polarity) pairs at aspect-level following:

$$P = \frac{TP}{TP + FP}, R = \frac{TP}{TP + FN1 + FN2}$$

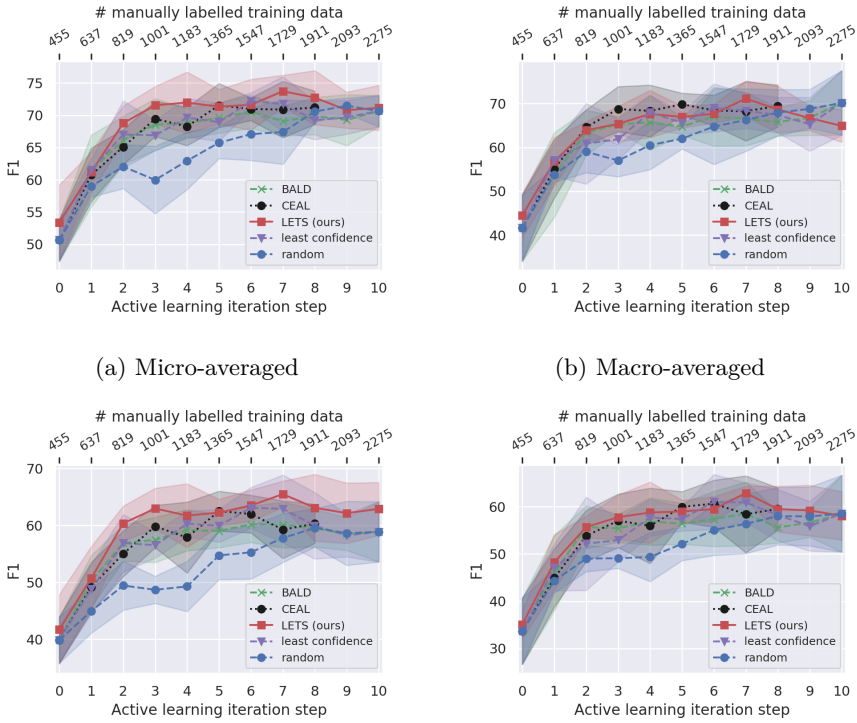
where TP, FP, FN1, and FN2 are defined as in Table 4.4. Similar to ACD, both micro- and macro-averaged F_1 are used.

4.5.4 Results and Analysis

Exp 1: Caffeine Challenge

Fig. 4.6 illustrates the active learning results with the Caffeine Challenge dataset. Active learning results in ACD metrics are illustrated in Fig. 4.6a and Fig. 4.6b. All active learning methods show better performance improvement than random sampling. It is observed that all models achieve much lower performances in macro-averaged scores than micro-averaged scores. These results show that the models perform worse towards minority aspect categories in the Caffeine Challenge dataset. In micro-averaged ACD score, LETS outperforms other active learning methods in general. In macro-averaged ACD score, CEAL achieves slightly better performance than LETS. However, the ACD metrics are incomplete because they ignore sentiment classification results.

ACSC metric is proposed to address the limitation of the ACD metric and correctly evaluate the ABSA system end-to-end. Fig. 4.6c and Fig. 4.6d illustrate



(a) Micro-averaged

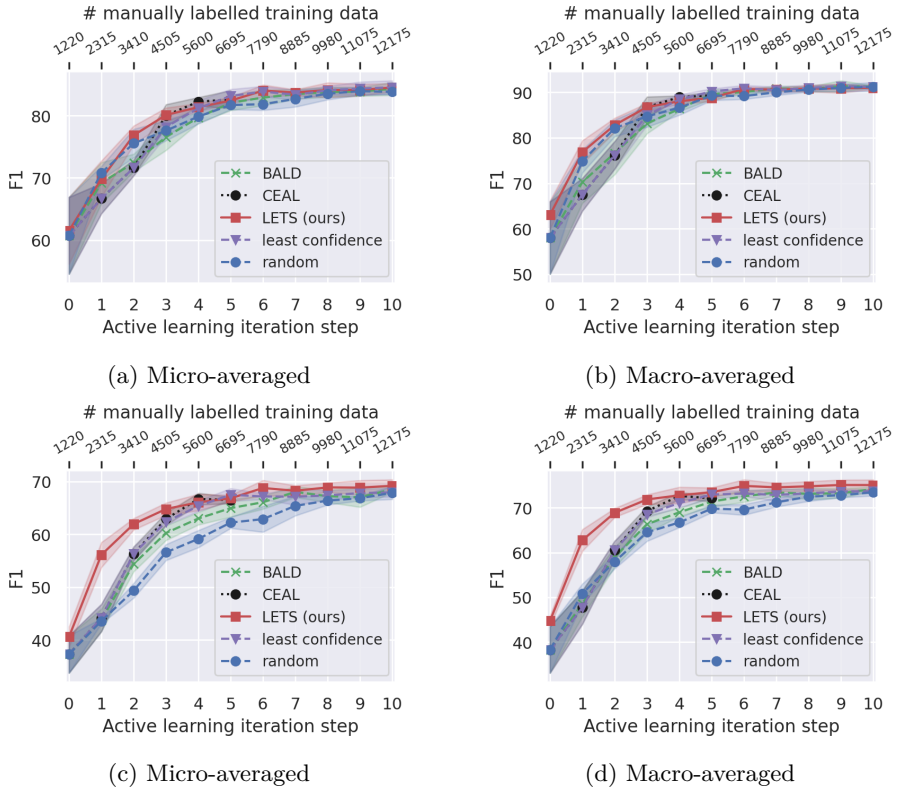
(b) Macro-averaged

(c) Micro-averaged

(d) Macro-averaged

Figure 4.6: Active learning results with the Caffeine Challenge dataset. Aspect category dectection (ACD) scores ((a), (b)). Aspect category sentiment classification (ACSC) scores ((c), (d)). Each line indicates averaged 5-fold results with standard deviation as shade. The bottom X-axis indicates the active learning iteration step and the top x-axis indicates the number of manually labelled training data. Y-axis indicates the performance score.

active learning results with the respect to the ACSC metrics. From the figures, it is observed that the performances of all models decrease compared to the observations from the ACD metrics. Similar to the results with the ACD metrics, LETS shows better performance improvement compared to other active learning methods. Specifically, from iteration step 0 to 1, the performance of LETS increases from 35.1% to 48.2%, while other method increase from 33.7% up to 47.1% in macro-averaged ACSC metric. The most significant difference is observed between LETS and random sampling. For example, random sampling achieves a similar performance of 48.2% at iteration step 2-4. Moreover, the



(a) Micro-averaged

(b) Macro-averaged

(c) Micro-averaged

(d) Macro-averaged

Figure 4.7: Active learning results with the SemEval dataset. Aspect category dectection (ACD) scores ((a), (b)). Aspect category sentiment classification (ACSC) scores ((c), (d)). Each line indicates averaged 5-fold results with standard deviation as shade. The bottom X-axis indicates the active learning iteration step and the top x-axis indicates the number of manually labelled training data. Y-axis indicates the performance score.

difference between LETS and random sampling increases over iteration steps. The random sampling method at iteration step 6-7 and LETS at iteration 2 show similar performances in terms of macro-average ACSC metric. These results suggest that LETS can reduce manual labelling efforts 2-3 times better compared to the random sampling method. Also, LETS slightly outperforms other active learning methods at the beginning of the iteration step with the respect to the ACSC metrics. This result shows that the task-specific and the proposed label augmentation can contribute to better generalisability with the Caffeine Challenge data set.

Performance differences between LETS and random sampling method are statistically significant (Wilcoxon signed-rank test with $p < .05$) from iteration step 1 to 7 and iteration step 2 to 5 in micro-and macro-averaged ACSC metrics, respectively. However, performance differences between LETS and active learning methods are not statistically significant ($p > .05$) throughout the entire iteration steps. In general, all methods show high variances of performances.

One interesting observation is CEAL achieves lower performances than LETS in terms of micro-averaged ACSC metric, especially in the later iteration steps. This is different from the observation from the micro-averaged ACD metric. A possible explanation for this is as follows: CEAL uses pseudo-labels. These pseudo-labels might not correct in terms of sentiment classes and errors might propagate throughout the iteration steps. Since the ACD metrics ignore sentiment classification results, this error might not be detected. Results with the macro-averaged ACSC metric show similar trends to the results with the macro-averaged ACD metric. These results suggest LETS slightly outperforms CEAL in terms of end-to-end evaluation metric.

Exp 2: SemEval

Fig. 4.7 illustrates the active learning results with SemEval benchmark dataset. Compared to the results with the Caffeine Challenge dataset, it is observed that the results with the SemEval dataset show less fluctuated learning curves in general. It is potentially because the SemEval dataset contains fewer aspect categories with more training data.

As illustrated in Fig. 4.7a and Fig. 4.7b, LETS shows slightly faster learning curves compared to other methods in terms of the ACD metrics. The random sampling method shows better learning curves compared to other active learning methods (i.e., BALD, CEAL, least confidence) in the ACD metrics. However, this does not imply that the random sampling method outperforms other active learning methods because the ACD metrics ignore sentiment classification results.

Fig. 4.7c and Fig. 4.7d show the active learning results in terms of the ACSC metrics. It is observed that the performances of all models decrease compared to the observations from the ACD metrics because the ACSC metrics consider sentiment classification results. From the figures, we can also see that the random sampling method achieves slower learning curves compared to the active learning methods. These results are opposite from the results with the ACD metrics and imply that the model trained with randomly sampled data tends to more misclassify sentiment labels.

In the ACSC metrics, it is observed that LETS substantially outperforms other active learning methods and random sampling method by showing fast performance improvement. For example, from iteration step 0 to 1, the performance of LETS substantially increases from 45.5% to 61.6%, while the performances of other methods only increase from 38.3% to around 50.8% in macro-averaged ACSC metric. Other methods achieve a similar performance of 61.6% at iteration step 2-3, which means that LETS can reduce manual labelling effort 2-3 times better with the SemEval dataset. Moreover, it is worth mentioning that LETS achieves significantly (Wilcoxon signed-rank test with $p < .05$) better performances than other methods at the beginning and the end of iteration thanks to the task-specific pre-training and label augmentation. Similar trends are also observed in the micro-averaged ACSC metric. Similar to the result with the Caffeine Challenge dataset, this result shows that the task-specific and the proposed label augmentation can also contribute to better generalisability with the SemEval dataset.

Performance differences between LETS and random sampling method are statistically significant ($p < .05$) throughout entire iteration steps in both micro- and macro-averaged ACSC metrics. Also, performance differences between LETS and other active learning methods are statistically significant ($p < .05$) from iteration 0 to 4 for BALD and from iteration step 0 to 2 for CEAL and least confidence methods, respectively, in both micro and macro-averaged ACSC metrics.

4.5.5 Discussion

The proposed LETS integrates multiple components, including task-specific pre-training, label augmentation, and active learning. To investigate the effect of task-specific pre-training with label augmentation separately, we further analyse the performances of a pre-trained model (PT) and task-specific pre-trained model (TSPT) by ablating the label augmentation (LA) component. Fig. 4.8 and Fig. 4.9 summarise the ablation study with the Caffeine Challenge dataset and the SemEval dataset, respectively. Note that all models use the proposed active learning method.

From the Fig. 4.8 and Fig. 4.9, it is observed that each task-specific pre-training and label augmentation provides performance improvement in the ACSC metrics. Nonetheless, more consistent improvement is observed when both components are applied together. For example, the results from the Caffeine Challenge dataset, as illustrated in Fig. 4.8, show that task-specific pre-training can contribute to performance improvement and label augmentation can further provide performance boost, especially in early iteration steps.

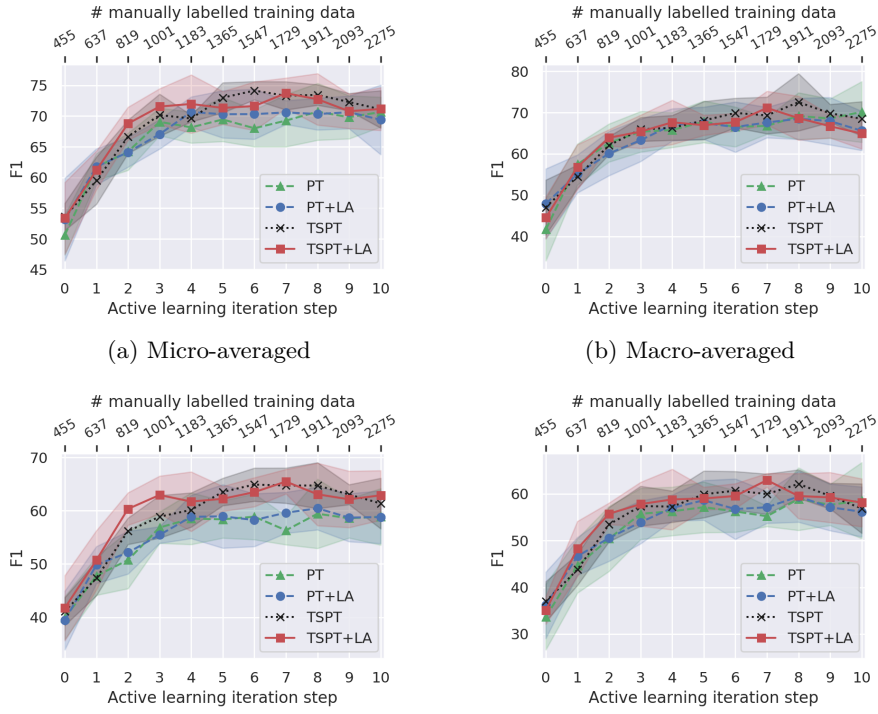
Similar trends are also observed in the results from the SemEval dataset as illustrated Fig. 4.9. The major differences are the results from the SemEval dataset are more stable throughout the iteration steps. The results from the Semeval dataset, as illustrated in Fig. 4.9, show significant differences ($p < .05$) between the task-specific pre-trained model with label augmentation (TSPT+LA) and the pre-trained model (PT) from iteration step 0 to step 4. This suggests that the combination of task-specific pre-training and label augmentation can contribute statistically significant performance improvement for the SemEval dataset, in early iteration steps. Interestingly, each task-specific pre-training and label augmentation can also contribute to the similar performance improvement of combining both of them. This suggests that applying either ask-specific pre-training or label augmentation can be also beneficial for the SemEval dataset.

4.6 Limitations and future studies

Even though we show the effectiveness of the proposed method by validating with two different datasets, some points can be further studied. Firstly, the Caffeine Challenge dataset is semi-realistic and not collected from actual users of a mobile application. This is mainly because the goal of this paper was to conduct a pilot study of developing an aspect-based sentiment analysis system for the healthcare domain prior to having a mobile application available. Therefore, further study is needed to collect real-world data and conduct experiments to validate the developed system. Since the real-world data are not labelled and the main contribution of this paper is proposing a label-efficient training scheme, we argue that the proposed method can be used to efficiently label the real-world data to further train the system.

The second limitation is the handcrafted rules of the proposed methods. The majority and minority classes were defined based on the frequency in the training sets. Further study could explore an algorithmic approach to distinguish between majority and minority classes. For example, in the active learning setting, minority classes can be dynamically defined based on the labelled data set of the previous iteration step. Also, the proposed label augmentation uses handcrafted dictionaries. A synonym search algorithm by using a lexical database, such as WordNet (Miller, 1995), or a knowledge graph, such as ConceptNet (Speer et al., 2017), could be used for automatically creating dictionaries for the proposed label augmentation.

Thirdly, a remaining difficulty in applying this work is to know when to start and when to stop active learning iterations. For example, in our experiments



(a) Micro-averaged

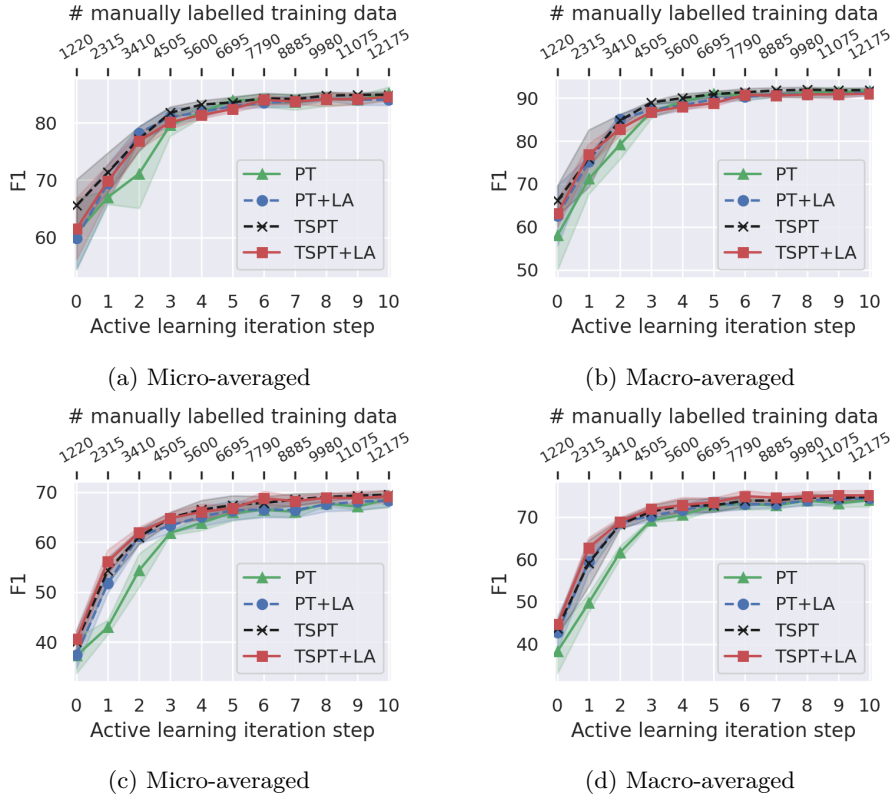
(b) Macro-averaged

(c) Micro-averaged

(d) Macro-averaged

Figure 4.8: Compared active learning results for ablation study with the Caffeine Challenge dataset. Aspect category dectection (ACD) scores ((a), (b)). Aspect category sentiment classification (ACSC) scores ((c), (d)). Each line indicates averaged 5-fold results with standard deviation as shade. The bottom X-axis indicates the active learning iteration step and the top x-axis indicates the number of manually labelled training data. Y-axis indicates the performance score. PT and TSTP refer to the model with pre-training and task-specific pre-training, respectively. Masked language modelling is used for task-specific pre-training objective. +LA indicates that label augmentation is applied during the active learning process. All models use the proposed active learning method.

(Sec. 4.5), the size of seed data is set to 20% of the training set for the Caffeine Challenge dataset while it is set to 10% of the training set for the SemEval dataset. It is decided based on heuristics and future studies could investigate the optimal size of the seed data. Also, even though the proposed method achieves fast performance improvements at the beginning, it reaches a plateau in the middle of the active learning process. This is because we consider a



(a) Micro-averaged

(b) Macro-averaged

(c) Micro-averaged

(d) Macro-averaged

Figure 4.9: Compared active learning results for ablation study with the SemEval dataset. Aspect category dectection (ACD) scores ((a), (b)). Aspect category sentiment classification (ACSC) scores ((c), (d)). Each line indicates averaged 5-fold results with standard deviation as shade. The bottom X-axis indicates the active learning iteration step and the top x-axis indicates the number of manually labelled training data. Y-axis indicates the performance score. PT and TSTP refer to the model with pre-training and task-specific pre-training, respectively. Masked language modelling is used for task-specific pre-training objective. +LA indicates that label augmentation is applied during the active learning process. All models use the proposed active learning method

pool-based active learning scenario, which assumes a large amount of unlabelled data at the beginning of the process and the active learning iteration ends when there is no more data to be labelled. To avoid unnecessary iteration steps, a stopping strategy is needed. Potentially, stopping strategy can be defined based on the stabilisation of predictions (Bloodgood and Vijay-Shanker, 2009) or the

certainty scores of predictions (Zhu et al., 2010).

4.7 Conclusion

In this paper, we introduce a new potential application of ABSA applied to health-related program reviews. To achieve this, we collected a new dataset and developed an ABSA system. Also, we propose a novel label-efficient training scheme to reduce manual labelling efforts. The proposed label-efficient training scheme consists of the following elements: (i) task-specific pre-training to utilise unlabelled task-specific corpus data, (ii) label augmentation to exploits the labelled data, and (iii) active learning to strategically reduce manual labelling.

The effectiveness of the proposed method is examined via experiments with two datasets. We experimentally demonstrated the proposed method shows faster performance improvement and achieves better performances over existing active learning methods, especially in terms of the end-to-end evaluation metrics. More specifically, experimental results show that the proposed method can reduce manual labelling effort 2-3 times compared to labelling with random sampling on both datasets. The proposed method also shows better performance improvements than the existing state-of-the-art active learning methods. Furthermore, the proposed method shows better generalisability than other methods thanks to the task-specific pre-training and the proposed label augmentation.

As future work, we expect to collect actual user data from a mobile application and implement the developed ABSA system with the proposed label-efficient training scheme. Moreover, we will investigate a stopping strategy to terminate the active learning process to avoid unnecessary iteration steps.

Appendix

4.A Examples of the Collected Data

Table 4.5 shows examples of the collected data used for experiments.

Table 4.5: Example of question and answers. This example shows 12 different responses from a single participant.

Imagine you **successfully finished** the challenge.
Q1: How was your experience with this challenge and why?
Answer (pos): My experience was great. I felt that my experience was personalized and I really was able to fall asleep faster and stay asleep longer by giving up caffeine after 1pm. It was a lot easier than expected.
Answer (neu): It was okay. While I did find it helpful to give up caffeine after 1pm to help with my sleep, it was difficult for me to give up and almost felt as if I were detoxing from caffeine.
Answer (neg): My experience was not very good. While I was able to give up caffeine after 1pm, it gave me a headache as I must have been going through withdrawals and in turn, these headaches kept me up later than I would have wanted.
Q2: Could you tell me how reducing caffeine affected you?
Answer (pos): Reducing caffeine really affected me positively. I was easily able to give the caffeine up after 1pm and in turn, I fell asleep much faster and didn't wake up throughout the night as I normally would.
Answer (neu): It was an okay experience. While I slept better, it was difficult for me to give up the caffeine, especially chocolate when I crave a snack after work.
Answer (neg): My experience was not very good. Because I gave up caffeine, I think my wellbeing was negatively affected because I then had a headache which made getting to sleep difficult. I think I actually lost sleep due to this.
Imagine you **was not able to complete** the challenge.
Q1: How was your experience with this challenge and why?
Answer (pos): While it was difficult for me to give up my afternoon and evening caffeine so I could not complete the challenge, I still had a positive experience as I did sleep better on the nights that I did successfully complete the challenge for the day.
Answer (neu): The experience was just okay for me. Because I did not successfully complete the challenge, I am not sure that I saw all of the benefits. I would like to try again in the future.
Answer (neg): I didn't like having to give up the caffeine. I kept getting headaches and for that reason I went back to the caffeine and did not successfully complete the challenge.
Q2: Could you tell me how reducing caffeine affected you?
Answer (pos): Reducing caffeine affected me by allowing me to go to sleep earlier and stay asleep longer. Therefore, I felt better and more refreshed when I woke up in the morning.
Answer (neu): It affected me in an okay way. While my sleep did tend to be better, I struggled with actually giving up the caffeine. This is something I would have to work at.
Answer (neg): It affected me negatively because while I was giving up the caffeine, I actually saw an increase in headaches and because of this, I also saw a lack of sleep.

Table 4.6: Explanation and examples of aspect categories.

Aspect	Explanation/Examples
Sleep quality	Impact on sleep quality. Positive: Sleep quality has improved. Negative: Sleep quality has been worsened.
Mood	Experience related to mental state. Positive: became calm or relaxed. Negative – felt nervous/anxious, experienced mental drain, or negative thoughts.
Energy	Impact on energy and concentration. Positive: Had/felt ore energy during day. Negative: Tired during the day or couldn't concentrate at work.
Missing caffeine	Feeling of caffeine deprivation. Positive: Did not miss caffeine products. Negative: Missed the taste of caffeine or didn't like decaffeinated alternatives.
Difficulty level	Difficulty of the challenge. Positive: Challenge was easy/easier than thought. Negative: Too difficult to change the habit.
Physical withdrawal symptoms	Impact on physical state. Positive: Physical state has improved. Negative: Experienced headache, stomach aches, or any other physical withdrawal symptoms.
App experience	Experience with app. Positive: App was supportive or reminder/recommender was helpful. Negative: User experience of app was bad or the reminder was annoying.

4.B Explanation of Aspect Categories

Table 4.6 summarises the explanation and examples of aspect categories used in the paper.

4.C Aspect Category Distribution of the SemEval Dataset

Fig. 4.10 illustrates the aspect category distribution of the training set from the SemEval dataset used for the experiments. As it is shown in the figure, the SemEval dataset is imbalanced and we define {Food, Anecdotes/Miscellaneous} and {Service, Ambience, Price} as majority and minority aspect categories, respectively.

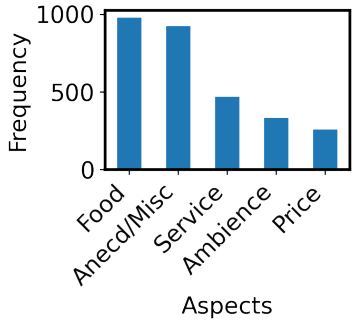


Figure 4.10: Aspect category distribution of the training set from the SemEval dataset. Anecd/Misc refers Anecdotes/Miscellaneous aspect category.

Table 4.7: Detailed implementation specification.

Item	**Specification**
CPU	Intel®Xeon®W-2123 CPU @ 3.60 GHz
GPU	NVIDIA GeForce GTX 1080 ti, 11 GB memory
Graphic driver	NVIDIA graphic driver version 416.34
CUDA	Version 10.0
OS	Windows 10, 64-bit
Python	Version 3.6.6
Pytorch	Version 1.5.1

4.D Implementation and Training Settings

All experiments were performed on the Windows 10 operating system and the detailed specification of hardware and software is summarised in Table 4.7. For model implementation, PyTorch version of BERT with the pre-trained weights (`bert-base-uncased`) (Wolf et al., 2019) was used as the pre-trained model (PT). During task-specific pre-training, the pre-trained model is further trained on the end task corpus. For task-specific pre-training, we adopt masked language modelling (Devlin et al., 2019) with masking probablity $p = 0.15$. During task-specific pre-training, randomly sampled 10% of training data is used as a validation set for early-stopping.

For fine-tuning, 5-fold cross validation splits are created by using K-Folds

cross-validator function from scikit-learn library[6]. Also, a final dense layer with softmax function is added and cross entropy loss is used. Since the focus of this paper is active learning experiments, we did not conduct hyperparameter tuning experiments but used hyperparameter values based on the recent study (Sun et al., 2019b) as summaries in Table 4.8.

Table 4.8: Hyperparameters for task-specific pre-training (top) and fine-tuning (bottom).

Hyperparameter	**Assignment**
training epoch	4
batch size	32
learning rate	$2e-5$
drop out	0.1
optimizer	AdamW
training epoch	4
batch size	32
learning rate	$2e-5$
drop out	0.1
optimizer	AdamW
classification layer	feed-forward network

4.E Pre-defined Dictionaries for Label Augmentation

Pre-defined dictionaries were used for label augmentation. For the Caffeine Challenge dataset, the list of minority aspect categories and the list of similar words for each aspect categories are defined as:

- Missing caffeine: [Missing caffeine, Dislike decaffeine, Need caffeine, Caffeine addiction]
- Difficulty level: [Difficulty level, Hard to finish, cannot complete, Too difficult]
- Physical withdrawal symptoms: [Physical withdrawal symptoms, Headache, Pain, Jitter]

[6] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

- App experience: [App experience, UI, UX, Design]

For SemEval dataset, the list of minority aspect categories and the list of similar words for each aspect categories are defined as:

- Service: [Service, Staff][7]
- Ambience: [Ambience, Atmosphere, Decor]
- Price: [Price, Bill, Quality[8]]

[7]During experiments, we observed that adding more labels for Service aspect category harms the performance.

[8]Quality is not a similar word for price but it is used because the training data set contains reviews mentioning price-quality relationship.

Chapter 5

Monitoring: Temporal Information Extraction and Normalisation

This chapter was previously published as:

Shim, H., Lowet, D., Luca, S., & Vanrumste, B. (2021, November). Synthetic Data Generation and Multi-Task Learning for Extracting Temporal Information from Health-Related Narrative Text. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, The Seventh Workshop on Noisy User-generated Text*, pp. 260-273.

This chapter includes a study on a free text sleep diary tool for monitoring sleep behaviour. The goal of the free text sleep diary tool is to extract structured temporal information from unstructured sleep diaries. To achieve this, a system can not only extract temporal expressions (e.g., "8 in the evening") but also normalise them into standard formats (e.g., 20:00 (HH:MM)). Temporal expression understanding requires both numeracy skills (e.g., "half past nine" = 09:00 + 00:30 = 09:30) and language understanding skills (e.g., "half past nine in the *evening*" = 21:00 + 00:30 = 21:30) which is not trivial. On one hand, a rule-based approach works well with the standard expressions but cannot handle noisy data (e.g., misspelt expressions, uncommon expressions, etc) and ambiguous expressions (e.g., "I went to bed at 10" –> 10:00 or 22:00). On the other hand, a language model can generalise well at extraction task but

it lacks numeracy that is required for normalisation. Therefore, we aim to combine the power of the rule-based approach and language model. Specifically, we investigate how to enhance the numeracy of a language model with a low-resource setting when there is no large dataset for training a model for temporal information extraction.

For this, we study how to utilise synthetic data to inject knowledge into a language model. To utilise a language model, we reformulate temporal information extraction and normalisation tasks as a question and answering task. The main hypothesis is that we can inject a numeracy skill into a language model for this temporal information extraction task by utilising synthetic data generated by the rules. To achieve this, we propose a data augmentation technique that uses a set of regular expressions for generating synthetic data. Also, we propose multi-task learning that includes an auxiliary task to receive more training signals from the existing training data. We empirically evaluate the proposed methods on a custom dataset of free text sleep diaries. Experimental results show the effectiveness of using synthetic data and the multi-task approach can contribute to performance improvement when it utilises the synthetic data for training.

This chapter studies the following research questions:

RQ1. How can we fine-tune a pre-trained language model when only a small-sized training set for the target task is available?

RQ3. Can we exploit other resources (e.g., knowledge, databases, et cetera) during fine-tuning to improve the performance of a pre-trained language model?

5.1 Introduction

Extracting temporal information from text is important linguistic skill to process health-related text. Also, there are a lot of potential applications of temporal information extraction in the health-related domain, including forecasting treatment effect (Choi et al., 2016), early detecting diseases (Khanday et al., 2020), and tracking treatment progress (Demner-Fushman et al., 2021). With the recent trends of telehealth, an automated system that can extract temporal information from the health-related narrative text can provide benefits to not only healthcare professionals but also recipients enabling active engagement, such as self-monitoring.

I went to bed around 11 pm. Used the phone for around 15 minutes and after that switched the light off. It took around 30 minutes to fall asleep. My sleep was disturbed at 5:45 am. and I spend in the bed for other 45 minutes to get sleep. I got off the bed around 6:30 am. Overall the sleep was normal. I felt refreshed.

Event	Time
Bed time	23:00
Lights off time	23:15
Sleep time	23:45
Sleep disturbance	05:45
Duration of disturbance	00:45
Out of bed time	06:30

Figure 5.1: Example of free-text sleep diary (top) and the extracted temporal information (down).

In this paper, we consider the use-case of a sleep diary, which is a summary of sleep designed to gather information about daily sleep patterns (Carney et al., 2012). A typical sleep diary consists of a series of close-ended questions to record the time. By writing sleep diaries, people can keep track of sleep, monitor sleep habits, and document sleeping problems which can be shared with their sleep therapists. We focus on extracting temporal information from a free-text sleep diary, To achieve this, a system should extract temporal expressions from the unstructured user-generated text and normalise the extracted temporal expressions into a standard format, as illustrated in Figure 5.1.

Temporal information extraction from user-generated text is a challenging task. First of all, it requires processing not only text but also numbers (e.g., 11pm or 23:00). But recent pre-trained language models (Devlin et al., 2019; Yang et al., 2019) have difficulty in processing numbers (Saxton et al., 2018; Ravichander et al., 2019; Dua et al., 2019) because these language models are pre-trained with language modelling objectives. Even though there have been recent studies on training language models to process numerical information (Andor et al., 2019; Geva et al., 2020), the remaining challenge is how to obtain a large amount of training data.

A second challenge is that there are various ways of describing the same normalised time. For example, the normalised time 23:00 can be expressed as 11, 11 pm, 23:00, eleven o'clock, etc. This issue is, even more, severe when dealing with user-generated text that is typically noisy: the user-generated text is prone to spelling errors and grammatical errors and contains a lot of

abbreviations (Petz et al., 2013). To address this, a sufficient amount of training dataset containing pairs of various temporal expressions and normalised time values is required.

A third challenge is that there are different types of temporal expressions which of each is difficult to extract. For example, temporal expressions include not only standalone times (e.g., 23:00) but also relative times (e.g., 5 minutes after), counts (e.g., 3 times), duration (e.g., for an hour), and frequencies (e.g., once per hour). For relative time expressions, the challenge is how to annotate temporal expressions and model dependencies. For count time expressions, the challenge is to deal with ambiguous terms, such as '*several times*' and '*a few times*'.

The last challenge is how to collect large-scale data while developing a proof-of-concept model to validate the hypothesis. Especially for health-related data, the data collection requires rigorous process of considering privacy and ethical aspects, which might result in a slow process. Moreover, typical machine learning development process includes the multiple cycles of collecting a new dataset and updating a model to improve the performance of model. Therefore, the challenge is how to train a machine leanring model when only a low very low amount of training data is available.

Therefore, the main research question of this paper is how to extract temporal information from user-generated noisy text with the limited number of training data. To this end, we propose a synthetic data generation algorithm to augment the size of training data. We also propose a multi-task model and investigate whether the multi-task learning strategy is beneficial to the target task by exploiting additional training signals from the existing training data. The main contributions of this paper include the followings:

- A new custom dataset has been collected to demonstrate the success of the free-text sleep diary use-case (Section 5.3).
- The temporal information extraction and normalisation tasks are reformulated as a question and answering task (Section 5.4.1).
- A novel model that can extract temporal expressions from unstructured text and normalise them into the standard format is proposed (Section 5.4.2).
- Experimental results show that utilising synthetic data and multi-task learning can be beneficial to performance improvement (Section 5.5.5).
- We also provide further analysis on experimental results to reveal insights of the model behaviours (Section 5.6).

5.2 Related Work

There are two lines of approach in temporal information processing. One is rule-based and the other is machine learning-based. Generally, rule-based systems achieve high performances in a normalisation task (Chang and Manning, 2012). However, rule-based systems have difficulties in dealing with ambiguous phrases or relative expressions (Verhagen et al., 2010; Chang and Manning, 2012).

Another line of approach is machine learning-based approaches. Previous works have focused on detecting temporal links between entities and classify the temporal relations between them (Ning et al., 2017; Meng and Rumshisky, 2018) rather than predicting the exact time of events. Recently, Leeuwenberg and Moens (2020) propose a system that can directly extract start and end-points for events from the text. However, the remaining gap is that it is not entirely end-to-end: Leeuwenberg and Moens (2020) used the text with ground truth event spans and normalized temporal expressions as inputs. Moreover, even though machine learning models show promising results, the fundamental challenge is how to obtain data. Not only data acquisition can be difficult but also data labelling can be time-consuming and expensive.

5.3 Sleep Diary Analysis

This section describes the dataset collected for experiments. The following subsections explain the details of use-case definition, data collection protocol, data labelling scheme, and an initial data analysis result.

5.3.1 Use-Case Definition

A sleep diary is a summary of sleep designed to gather information about daily sleep patterns. A typical sleep diary consists of a series of close-ended questions to record the time (i.e., the time people went to bed last night, woke up, etc), factors that may have influenced the way people slept, and how people felt when they woke up. In this study, we introduce free-text sleep diary use-case that allows people to describe their nights' of sleep in text. The goal of this study is to extract structured information from unstructured sleep diaries, as described in Figure 5.1.

Table 5.1: The list of sleep-related event entities used in this study.

Event entity	Explanation
bed time	The time when the participant went to bed/bedroom.
lights off	The time when the participant switched off the lights and began trying to fall asleep.
sleep time	The time when the participant fell asleep
sleep latency	The amount of time it took for the participant to fall asleep after deciding to go to sleep.
sleep disturbance	The times when the participant's sleep was disturbed.
wake up	The time when the participant woke up from their sleep.
out of bed	The time when the participant finally got out of bed to start their day.
sleep duration	The total duration of time the user was asleep.

5.3.2 Data Collection Protocol

We conducted an online survey via Amazon's Mechanical Turk (MTurk) to collect experimental data. At the beginning of the survey, the participants were given a questionnaire with a brief background of the study purpose. Then the participants were asked to provide information about their sleep of the previous night via an open-ended question (i.e., *"Please describe, in a few lines, your sleep last night."*). The details of data subject selection criteria and examples of responses are given in Appendix 5.A. In total, 600 participant inputs are collected and used for the experiments.

5.3.3 Data Labelling Scheme

To annotate the collected data, several sleep-related event entities are defined based on sleep study (Carney et al., 2012) as summarised in Table 5.1. Each event entity text was annotated with its span (i.e., start and end positions in the text), entity label, expression type (i.e., standalone, relative[1], count, frequency[2]), and normalised time value. Expression types are used to assign a specific type value: None, +/-, *, t for standalone, relative, frequency, count, respectively. A normalised time value includes a type value and 4 digits indicating HH:MM, except for a count type: for an entity with a count type, a normalised time

[1]Relative type includes both relative time (e.g., after 5 minutes) and duration (e.g., for 5 minutes).

[2]Frequency type includes expressions of events occurring periodically (e.g., every 1 hour)

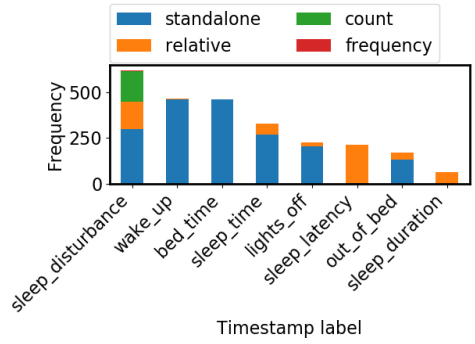


Figure 5.2: Distribution of the annotated data over the event entities.

value is a cardinal number padded with leading zeros (e.g., 1 becomes 0001). Also, we set rules for ambiguous expressions. For example, 'a lot of times', 'several times', and 'many times' are annotated as five times (t0005) and 'a few times', 'a couple times' are annotated as two times (t0002). The example of an annotated data point is illustrated in Appendix 5.B.

Figure 5.2 shows the distribution of the collected data set over the sleep-related event entities. It is observed that `sleep_disturbance` entity appears more than other entities. This is because different types of sleep disturbance are often mentioned together (i.e., "*I woke up 1 time to use restroom at midnight*"). Meanwhile, some entities (e.g., `lights_off`, `sleep_latency`, `out_of_bed`, `sleep_duration`) are often missing in sleep diary entries. In general, `bed_time` and `wake_up` entities appear once per each sleep diary entry.

5.4 Multi-Task Temporal Information Extraction Model

5.4.1 Task Formulation

We formulate a temporal information extraction and normalisation task similar to a question and answering task. Therefore, each data point is transformed into <entity, text, answer> where the entity is a sleep event entity label, the text is sleep diary text, and the answer is a normalised time with a type value and 4 digits. For example, a system is expected to predict a list of answers [None, 2, 2, 3, 0] given input <bed time, I went to bed at half past 10...>.

Formally, an entity $q_j \in Q$, where Q is the set of the sleep-related event entities decribed in Table 5.1, is tokenised[3] with m_j tokens $q_j = [q_j^1, ..., q_j^{m_j}]$ and sleep diary text is tokenised with n_i tokens $p_i = [p_i^1, ..., p_i^{n_i}]$. Then the task is to predict an answer $a_{ij} = [a_{ij}^{type}, a_{ij}^{t1}, a_{ij}^{t2}, a_{ij}^{t3}, a_{ij}^{t4}]$ given a sequence of tokens `[[CLS]` q_j `[SEP]` p_i, `[SEP]]`, where `[CLS]` and `[SEP]` are special tokens for classification and separation, respectively. a_{ij}^{type} is the ground truth label for the type value of the normalised time and a_{ij}^{t1}, a_{ij}^{t2}, a_{ij}^{t3}, and a_{ij}^{t4} is the the ground truth labels for the each digit of the normalised time.

5.4.2 Model Architecture

We propose a multi-task model that utilises a pre-trained language model with specific heads, motivated by recent works (Andor et al., 2019; Geva et al., 2020). The overview of the proposed model is illustrated in Figure 5.3.

Firstly, the model computes $e_{ij} = [e_{ij}^{cls}, .., e_{ij}^{l_{ij}}]$ which are contextualised representations for the $l_{ij} = m_j + n_i + 3$ input tokens ([CLS] q_j [SEP] p_i, [SEP]) by using a pre-trained language model BERT (Devlin et al., 2019). The contextualised embedding vector $\boldsymbol{e}_{ij}^{cls} \in \mathbb{R}^{d \times 1}$, corresponding to the classification token `[CLS]`, is fed to the type classification head (H_{type}) that uses a fully-connected layer followed by a softmax to compute distributions over the type values {None, +, -, *, t}. Then the remaining sequence of contextualised embedding vectors $[e_{ij}^2, .., e_{ij}^{l_{ij}}]$ is used to create pooled embedding $e_{ij}^{pool} \in \mathbb{R}^{d \times 1}$ by using average pooling. Then the pooled embedding e_{ij}^{pool} is passed to normalised time value heads ($H_{t1}, H_{t2}, H_{t3}, H_{t4}$). H_{t1} head computes a distribution over the number {0, 1, 2}[4] by using a fully-connected layer followed by a softmax layer. Similarly, H_{t2}, H_{t3}, and H_{t4} heads compute distributions over the numbers {0, ..., 9}.

The contextualised embedding vectors $[e_{ij}^2, .., e_{ij}^{l_{ij}}]$ are fed to additional answer span heads, H_{start} and H_{end}, to compute a score for each token, corresponding to whether that token is the start or the end of the answer span, respectively. The start and end probability for each token is computed as follow:

$$p_{ij}^{start} = \text{softmax}(H_{start}(e_{ij}^2), ..., H_{start}(e_{ij}^{l_{ij}})) \tag{5.1}$$

$$p_{ij}^{end} = \text{softmax}(H_{end}(e_{ij}^2), ..., H_{end}(e_{ij}^{l_{ij}})) \tag{5.2}$$

[3]Underbars are replaced by whitespace characters during tokenisation.

[4]Since the answer is formulated as a standard time format (e.g., HH:MM), the first digit is limited to {0, 1, 2}.

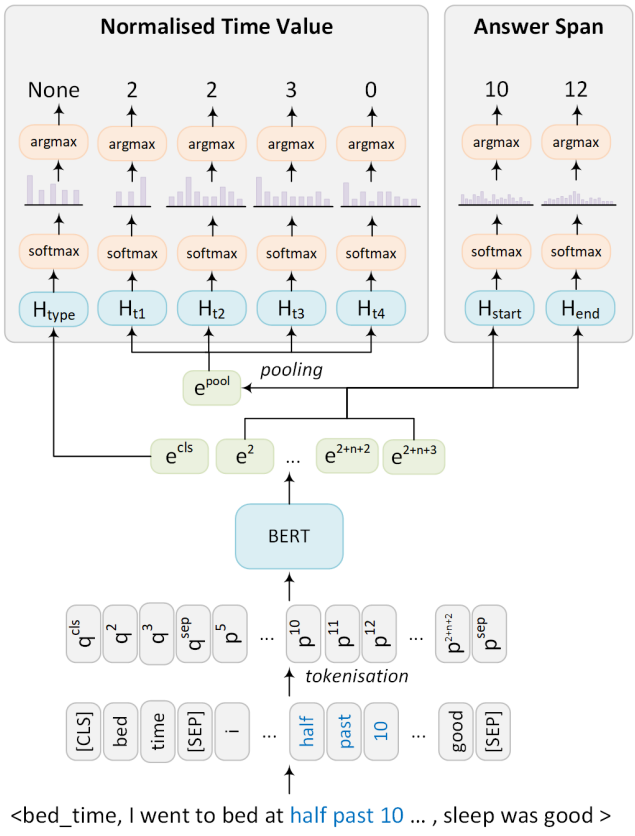


Figure 5.3: Illustration of the proposed model.

Normalised time prediction loss For normalised time prediction, cross-entropy between the target answer and the model estimation is used to compute $\mathcal{L}_{t_1}, \mathcal{L}_{t_2}, \mathcal{L}_{t_3}, \mathcal{L}_{t_4}$, and $\mathcal{L}_{type}$ which is a loss function for each head, respectively. Then the time loss function ($\mathcal{L}_{\text{time}}$) is defined as the linear combination of the each loss function, i.e.

$$\mathcal{L}_{\text{time}} = \alpha \mathcal{L}_{\text{type}} + \beta(\mathcal{L}_{t_1} + \mathcal{L}_{t_2} + \mathcal{L}_{t_3} + \mathcal{L}_{t_4}) \tag{5.3}$$

Answer span detection loss For answer span detection, we follow the previous work on a question and answering task by using a pre-trained language model (Devlin et al., 2019) to compute cross-entropy losses for the start $\mathcal{L}_{\text{start}}$ and end

$\mathcal{L}_{\text{end}}$. Then the span loss function ($\mathcal{L}_{\text{span}}$) is defined as the sum of the start loss and the end loss:

$$\mathcal{L}_{\text{span}} = \mathcal{L}_{\text{start}} + \mathcal{L}_{\text{end}} \tag{5.4}$$

Multi-task loss The final multi-task loss function ($\mathcal{L}_{\text{multi}}$) is defined as the linear combination of the normalised time prediction and answer span detection loss functions:

$$\mathcal{L}_{\text{multi}} = \mathcal{L}_{\text{time}} + \gamma\mathcal{L}_{\text{span}} \tag{5.5}$$

5.4.3 Synthetic data generation

To augment the size of annotated data containing temporal information, we propose a simple yet effective rule-based synthetic data generation algorithm. Figure 5.4 illustrates the proposed algorithm. The first step is to create a template by masking out labelled event entities from the annotated data and replacing them with placeholders. The second step is to generate random entity types and corresponding normalised time values. Then the randomly generated normalised time values are translated into texts by using regular expressions. The last step is to replace placeholders with the translated texts. Details of regular expressions and examples of generated texts are given in the Appendix 5.C.

5.5 Experiments

5.5.1 Dataset

The collected dataset from the Section 5.3 was used for the experiments. We randomly split the collected data ($n = 600$) into train, validation, and test sets with the ratio of 0.8, 0.1, and 0.1. After splitting data sets, we dropped the data points that do not contain any event entities. During pre-processing, we lowercased and tokenised data sets by using a WordPiece algorithm Schuster and Nakajima (2012). Numbers and punctuation symbols were not removed during pre-processing because they play important role in temporal expressions. Table 5.2 shows the statistics of each data set after pre-processing.

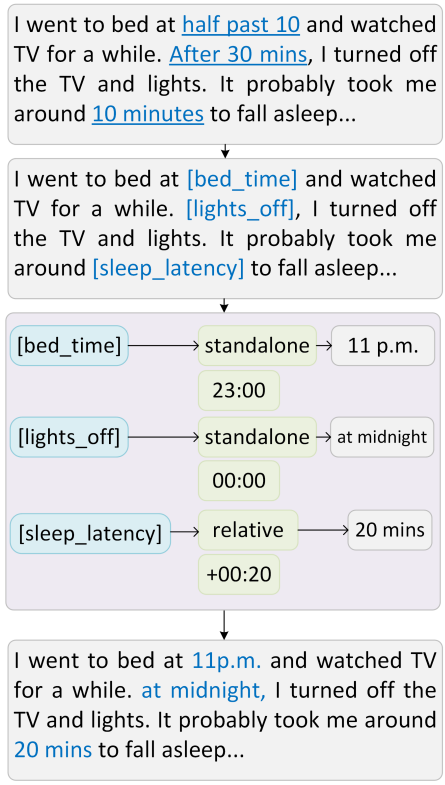


Figure 5.4: Illustration of the proposed synthetic data generation algorithm to augment the size of training data.

Table 5.2: Data set statistics across the different splits

Statistics	**Train**	**Valid**	**Test**
# sleep diaries	467	56	57
Total # tokens	197,695	24,007	23,528
Unique # tokens	1,687	604	585
Avg. # tokens/diary	95.9	101.7	95.3
Total # entities	2061	236	247
Avg. # entities/diary	4.4	4.2	4.3

The proposed synthetic data generation algorithm is applied to the train set to augment the size by a factor of k. The validation set is used to check the training progress and perform early stopping. The test set is used to evaluate the performance of the trained model.

5.5.2 Settings

We use a pre-trained BERT to implement the proposed model. All heads are implemented as fully-connected layers with dropout followed by softmax functions. Also, to investigate the effect of multi-task learning, we implement a baseline model (BASE) that uses only normalised time prediction loss ($\mathcal{L}_{\text{time}}$) and a multi-task model (MULTI) that uses multi-task loss ($\mathcal{L}_{\text{multi}}$), as defined as:

$$\mathcal{L}_{\text{time}} = \alpha\mathcal{L}_{\text{type}} + \beta(\mathcal{L}_{t_1} + \mathcal{L}_{t_2} + \mathcal{L}_{t_3} + \mathcal{L}_{t_4}) \tag{5.6}$$

$$\mathcal{L}_{\text{span}} = \mathcal{L}_{\text{start}} + \mathcal{L}_{\text{end}} \tag{5.7}$$

$$\mathcal{L}_{\text{multi}} = \mathcal{L}_{\text{time}} + \gamma\mathcal{L}_{\text{span}} \tag{5.8}$$

α, β, and γ are set to 0.25, 1, and 0.25, respectively. See Appendix 5.D for more system and training details.

5.5.3 Configuration

To configure the input and output of a model, each data point is expanded into multiple <entity, text, answer> triples. If a single data point contains the same entity more than once, the ordinal number is added to an entity label from the second occurrence. For example, `second_sleep_disturbance` will be used for the second `sleep_disturbance` event in a data point. For `sleep_disturbance` entity with a count type, the type is also added to an entity label, i.e., `count_sleep_disturbance` will be used. For output, a special character of a normalised time value (i.e., None, +, -, *, t) is used as a ground truth type value and each digit of a 4 digit normalised time is used as a ground truth normalised time value. For the multi-task model, the start and end position of entity text are used as ground truth value of start and end position, respectively.

Table 5.3: The performances of the baseline model (BASE) and the multi-task model (MULTI) on Normalised time prediction (NTP) and Answer span detection (ASD) tasks. k refers the augmentation factor. * and ** indicate that this result is significantly different (approximate randomisation test (Dror et al., 2018)) from the result without the synthesised data (the first row in that column) with p-value < 0.05 and < 0.01, respectively. Best performances are boldfaced.

	BASE		**MULTI**			
	NTP		NTP		ASD	
k	micro-EM	macro-EM	micro-EM	macro-EM	EM	F1
-	82.0	83.9	66.9	62.9	64.9	82.9
$\times 2$	83.3	**87.2***	86.9**	88.0**	**66.9**	85.1
$\times 3$	**85.3***	85.1	**88.6****	**91.9****	66.1	**86.7***
$\times 5$	81.6	81.6	86.1**	86.8**	64.5	84.1
$\times 8$	82.0	83.2	78.4**	80.8**	60.8	79.2*
$\times$ 10	72.2**	74.7**	68.6	71.2*	53.5**	75.3**

5.5.4 Evaluation Metrics

Normalised time prediction We use Exact Match (EM) as an evaluation metric. EM considers only when a model predicts both a correct type value and correct 4 digits of normalised time values as a correct prediction. Since the experimental data set has an imbalance over event entities, both micro-and macro-averaged scores are used: the micro-EM score is computed by taking the average over inputs and the macro-EM score is computed by taking the average at the entity level.

Answer span detection Following Rajpurkar et al. (2016), we use Exact Match (EM) and F1 score for answer span detection: EM measures the percentage of predictions that match the ground truth answers exactly. F1 score measures the average overlap between the prediction and ground truth answer. We treat the prediction and ground truth as bags of tokens, and compute the F1 score per entity and average over all of the entities. Both metrics consider articles, numbers, and punctuation symbols.

5.5.5 Results and Analysis

Normalised time prediction Table 5.3 summarises the experimental results. As expected, models trained on the synthetic training data generally achieve

Table 5.4: Normalised time prediction results per entity label. +SD indicates that synthetic data are used. * and ** indicate that this result is significantly different from the best result in that row (bolded) with p-value < 0.05 and < 0.01, respectively.

	BASE		**MULTI**	
	-	+SD	-	+SD
sleep disturb.	66.1	80.6	50.0	83.9
bed time	95.1	97.6	95.1	95.1
wake up	86.5	86.5	65.4	88.5
sleep time	75.0	69.4	72.2	75.0
lights off	100.	93.3	93.3	100.
sleep latency	95.2	100.	66.7	100.
out of bed	78.6	78.6	35.7	92.9
sleep dur.	75.0	75.0	25.0	100.
micro-EM	82.0**	85.3*	66.9**	**88.6**
macro-EM	83.9**	85.1*	62.9**	**91.9**

Table 5.5: Normalised time prediction results per expression type. +SD indicates that synthetic data are used. Best performances are boldfaced.

	BASE		**MULTI**	
	-	+SD	-	+SD
Standalone	86.8	86.3	72.0	**90.1**
Relative	78.7	78.7	55.3	**83.0**
Count	37.5	**93.8**	43.8	87.5

higher performances. Results show that the benefit of using synthetic data for training shows a peak at $k = 3$ for both models and the improvements are statistically significant. However, the benefit of using synthetic data decreases afterwards. It can even harm the performance of the baseline model when $k = 10$.

To further investigate the effects of using synthetic data for training, we calculate the performance per event entity label as shown in Table 5.4 and the performance per expression type as shown in Table 5.5. The models trained on synthetic data with the factor of $k = 3$ are used for comparison. Table 5.4 shows that using synthetic data generally improves the performance of almost all event entities. From both models, the biggest improvements are observed at `sleep_disturbance` entity, which is the most frequent entity label in the training set. However, as shown in Table 5.5, the biggest improvements in terms of expression type are observed at the count type, which is one of the least

frequent expression types in the training set. It is worth mentioning that the count type is only included in `sleep_disturbance` entity label, as illustrated in Figure 5.2. These results imply that using synthetic data can be the most beneficial to both models in terms of predicting normalised time values with the count type.

Answer span detection The answer span detection results of the multi-task model are also summarised in Table 5.3. Similar to the normalised time prediction results, the performances tend to increase till $k = 3$ and decrease afterwards. It is observed that the utilising synthetic data with the augmentation factor $k = 3$ can provide the significant improvement in terms of F1 measure. But the effect to the EM measure is not statistically significant (p>.05). It is also observed that using synthetic data with augmentation factor $k = 10$ can significantly harm the performances.

5.6 Discussion

Effects of using synthetic data The first row of Table 5.3 shows that the multi-task learning model achieves lower normalised time prediction performances than the baseline model when no synthetic data is used for training. However, when the multi-task model utilises an appropriate amount of synthetic data ($k = 3$), as it is shown in the last two rows in Table 5.4, the multi-task model significantly outperforms ($p < .01$) the baseline model without synthetic data. These results imply two things: 1) multi-task learning can be beneficial to improve the target performances of normalised time prediction. However, training the multi-task model may require a larger training set; and 2) the proposed synthetic data generation algorithm can mitigate this issue to a certain degree. Also, as shown in the last two rows in Table 5.4, when the same amount of synthetic data ($k = 3$) are used for both models, the multi-task model significantly outperforms ($p < .05$) the baseline model in terms of normalised time prediction. This result may not be so surprising since the multi-task model receives additional training signals during training.

Effects of using multi-task learning To get a further understanding of the effect of multi-task learning, we conduct a qualitative analysis. In Table 5.6, we highlight some examples of the predictions of the proposed models. In the first example, it is observed that both models can process the combination of number and text (*9 pm*) and an ambiguous expression (*10*), correctly predicting corresponding the normalised time values (21:00 and 22:00). It is also observed

Table 5.6: Qualitative examples showing the outputs of the proposed models. Underline indicates temporal expressions and red colour indicates wrong predictions. Due to limited space, we use the following abbreviations: sleep disturbance (dstb.), the count of sleep disturbance (cnt. dstb.), and the second occurrence of events (2nd).

***Sleep Diary**: I went to bed about 9 pm. we sleep with the lights on for my toddler who co-sleeps. I was asleep about 10. I woke up a lot of times in the night to blow my nose or to try and get comfortable. I got out of bed at 2:30 am. Sleep was terrible. I feel exhausted today.*

	BASE	**MULTI**		**Ground Truth**	
	time	time	text	time	text
bed	21:00	21:00	9 pm	21:00	9 pm
sleep	22:00	22:00	about 10	22:00	about 10
cnt. dstb.	t0005	t0002	a lot of times	t0005	a lot of times
out of bed	02:30	02:30	2:30 am	02:30	2:30 am

***Sleep Diary**: I turned the lights off at 9:30 layed down in bed at 10 pm. I fell alseep around 11pm I woke up at 1am to turn on my other side. I fell back to sleep until 4 am to use the rest room went back to sleep until 5:30 am.*

	BASE	**MULTI**		**Ground Truth**	
	time	time	text	time	text
lights off	21:30	21:30	9:30	21:30	9:30
bed	22:00	22:00	10 pm	22:00	10 pm
sleep	23:00	23:00	11pm	23:00	11pm
dstb.	01:00	01:00	1am	01:00	1am
2nd dstb.	01:00	04:00	4 am	04:00	4 am
wake up	05:00	05:30	5:30 am	05:30	5:30 am

***Sleep Diary**: I went to bed around 10p. m. I read for about 30 minutes. Put my kindle away and was asleep by 10:30. I woke up at 1a. m. and it took me around 20 minutes to fall back asleep. I woke up at 2:45 and used the bathroom. I went back to sleep until 4 which is when I normally get up.*

	BASE	**MULTI**		**Ground Truth**	
	time	time	text	time	text
bed	22:00	22:00	10p. m.	22:00	10p. m.
sleep	22:30	22:30	10:30	22:30	10:30
dstb.	01:00	01:00	1a. m	01:00	1a. m
2nd sleep	01:00	00:20	1a. m. and it took me around 20 minutes	+00:20	20 minutes
2nd dstb.	01:00	01:45	1a. m.	02:45	2:45
wake up	02:45	04:45	-	04:00	until 4

that the baseline model correctly predicts a normalised time value of a cardinal number (t0005) from the text-only temporal expression (*a lot of times*). The multi-task model fails at predicting the correct normalised time value but extracts the correct answer span (*a lot of times*). Based on this observation, it seems that answer span detection results can be useful to decide how to synthesise training data, such as generating pairs of <'a lot of times', t0005>.

In the second example, it is observed that the baseline model correctly predicts only the first occurrence of `sleep_disturbance` entity, predicting the identical timestamps for the second occurrences. Meanwhile, the multi-task model correctly predicts both occurrences with correct answer spans. We found that the multi-task model generally performs better on extracting normalised time values that occur multiple times in the text. However, in general, both models have difficulties in dealing with entities that occur several times in a single sleep diary. It is also observed that the normalised time value heads ($H_{type}, H_{t1}, H_{t2}, H_{t3}, H_{t4}$) and the answer span heads (H_{start}, H_{end}) of the multi-task model are not fully aligned: as shown in the third example, the multi-task model estimates the second sleep disturbance as 01:45 while extracting the answer span as '1a.m'. This error is challenging because the current model is a black box model so that we do not know where the error occurs and how the error propagates. To address this issue, one interesting area for future work may be in investigating shared information between the normalised time value heads and the answer span heads.

Limitations and Future Work Even though we show the effectiveness of the proposed method by validating on the collected dataset, some points can be further studied. First of all, the proposed models estimate a normalised time value conditioned on an input which is a pair of sleep diary text and sleep-related event entity label. However, since most sleep diary texts do not contain all the event entities, an additional module is required to detect which entities are mentioned in the given text and how many times each entity is mentioned in the given text. Similar to the previous work by Liu et al. (2020), the answer span detection head of the proposed multi-task model can be used as a detection module.

Secondly, even though the proposed models can handle relative expressions by using specific type values (i.e., +, -), a linking algorithm is currently missing. A potential solution is to add a head that can predict a starting point, similarly to the previous work by Leeuwenberg and Moens (2018).

The third limitation is that the proposed models process only temporal information. To completely analyse sleep diary, extracting contextual and qualitative information is required (Ibáñez et al., 2018). In the future study, we

will train a model to extract both temporal and other information from text data. To achieve this, we will collect more data that are longer and contain rich information about the context of the night and sleep.

5.7 Conclusions

In this paper, we propose a model that can extract temporal information from health-related narrative text. We conducted experiments to investigate how to utilise synthetic data and multi-task learning to improve the performance of normalised time prediction. Experimental results show that utilising synthetic data for training can contribute to performance improvement the most when the augmentation factor is set to 3. The results also show that when multi-task learning is used with synthetic data appropriately, the performance can be significantly improved. In the future study, we will extend the current work to extract not only temporal information but also contextual and qualitative information from text.

5.8 Ethical Considerations

Table 5.7 summarises ethical and privacy considerations of the data collection in this study.

Appendix

5.A Details of Data Collection Protocol

Participants were recruited through Amazon's MTurk-service. We selected data subjects who meet the following criteria:

- People who are 18 years or older
- People who are USA residents

When participants selected the study, they received a link to the web page hosting the survey. During the survey, participants were asked to answer an open-ended question (i.e., *"Please describe, in a few lines, your sleep last night."*)

Table 5.7: Ethical and privacy considerations for the data collection. Vulnerable groups include military veterans, terminally ill, educationally or socioeconomically disadvantaged, employees, students who could be unduly influenced, individuals with lack of or loss of autonomy due to immaturity or through mental disability that might suggest their consent is not of free will, etc.

Question	Answer
Are children under the age of 18 involved as test subjects in the study?	No
Are test subjects over the age of 65 involved in the study?	Yes
Do the test subjects belong to vulnerable groups?	No
Does the study induce harm or discomfort to the test subjects?	No
Is there any doubt on the test subjects' freedom in deciding on their participation?	No
Collection of any personal data	No
Collection of data by means of audio recording	No
Collection of data by means of video recording or photographs	No
Collection of data by means of observation of test subjects and logging in written format	No
Collection of data by means of filling-in questionnaires/surveys/interviews	Yes

with the guidance of the following sentences: "*While describing your answer, please include the following information: sleep-related events (e.g., the time you went to bed, the time you switched off to go to sleep, the time it took you to fall asleep, the number of times you woke up and the time at those moments, the time you woke up, the time you got out of bed) and the overall sleep evaluation or how you refreshed after you woke up.*"

At any moment, a participant was allowed to end her/his participation in the study. In this case, the test participant was not replaced. Furthermore, every participant was received informed consent, on the landing page that participants enter when following the link from MTurk. Only answers from the participants who gave their own consent to the study were used for experiments in this study. Table 5.8 shows examples of the collected data used for experiments.

Table 5.8: Examples of responses to the open-ended question regarding the previous night's sleep.

ID	Answers
#1	I went to bed at 11 pm. I switched off the lights and lay down around 15 minutes before falling asleep. It was a deep sleep and i wake only 1 time to witch off ceiling fan. I had couple of dreams that I remember partially not scary. I wake up at 6 am. lay on bed for 15 minutes more and got up.
#2	I turned off the lights around 9:45 PM. I closed my eyes and went to sleep around 10 PM. I did not wake up at all during the night. I slept straight through. I woke up at 6 AM. I lied in bed for a few minutes before actually getting up around 6:05 AM. I felt fairly well rested.
#3	I went to bed at 10:00 and immediately turned off the light. I fell asleep in just a few minutes. I slept without waking until 5:00. I immediately got out of bed when I woke up and felt great.

5.B Example of Annotated Data

Figure 5.5 shows the exmample of an annotated data point.

5.C Examples of Synthetic Data

```
{'text':'I went to bed at 10'o clock and I switched
     off the light at 11'o clock. After that I fall
     asleep in 30 minutes as my guess. I woke up 3
    times and I felt restless. I was awake up to 3
    hours. I woke up at 7'o clock in the morning
    and got out of the bed at 7:45 a.m. My overall
    sleep is up to 5 hours and it was not a sound
    sleep. If I awake during night then it causes
    me heavy headache and drowsiness. But today I
    din't get any of the above symptoms even though
     I woke up in the night. I felt fresh in the
    morning.',
        'labels': [
        { 'text': '10'o clock',
                'span': (17, 27),
                'entity': 'bed_time',
                'type': 'standalone',
                'norm_time': 2200},
        { 'text': '11'o clock',
                'span': (60, 69),
                'entity': 'lights_off',
                'type': 'standalone',
                'norm_time': 2300},
        { 'text': 'in 30 minutes',
                'span': (97, 110),
                'entity': 'sleep_latency',
                'type': 'relative',
                'norm_time': +0030},
        { 'text': '3 times',
                'span': (134, 141),
                'entity': 'sleep_disturbance',
                'type': 'count',
                'norm_time': t0003},
        {'text': '3 hours',
                'span': (181, 188),
                'entity': 'sleep_disturbance',
                'type': 'relative',
                'norm_time': +0300},
        {'text': '7'o clock',
                'span': (203, 212),
                'entity': 'wake_up',
                'type': 'standalone',
                'norm_time': 0700},
        {'text': '7:45 a.m',
                'span': (254, 261),
                'entity': 'out_of_bed',
                'type': 'standalone',
                'norm_time': 0745},
        {'text': '5 hours',
                'span': (290, 297),
                'entity': 'sleep_duration',
                'type': 'relative',
                'norm_time': +0500}]}
```

Figure 5.5: Example of annotated data point containing free-text sleep diary and labels of event entities.

Table 5.9: Regular expression patterns to translate timestamps into texts based on the given entity type and format.

Regular Expression	Entity type	Format	Example	
			Timestamp	Text
?(around|at|until|by|) t_1t_2[:,.]t_3t_4 ?(((a.?m.?)|(A.?M.?))| (hrs|hours|hour|hr))	standalone	$t_1t_2t_3t_4$	1130	at 11:30 am
?(around|at|until|by|) (t_1t_2|t_1t_2-12)[:,.]t_3t_4 ?(((p.?m.?)|(P.?M.?))|(hrs|hours|hour|hr))	standalone	$t_1t_2t_3t_4$	2230	by 10:30 PM
?(around|at|until|by|) t_1t_2 ?((o['']clock)|((a.?m.?)| (A.?M.?))|(hrs|hours|hour|hr)) ?(o(''| |)clock|[:,.]00)	standalone	$t_1t_2$00	0600	at 6 o'clock
?(around|at|until|by|) (t_1t_2|t_1t_2-12) ?((o['']clock)| ((p.?m.?)|(P.?M.?))|(hrs|hours|hour|hr)) ?(o(''| |)clock|[:,.]00)	standalone	$t_1t_2$00	2200	22 o'clock
?(after|within|) t_3t_4 ?(min|mins|minute|minutes) ?(later)	relative	+00t_3t_4	+0010	10 min later
t_3t_4 ?(min|mins|minute|minutes) ?(before|prior)	relative	-00t_3t_4	-0005	5 mins before
every t_3t_4 ?(min|mins|minute|minutes)	frequency	*00t_3t_4	*0010	every 10 mins
every (t_1t_2|t_1t_2+1) ?(hour|hours)	frequency	*$t_1t_2$00	*0100	every 1 hour
every (t_1t_2|t_1t_2+1) and ?(hour|hours) t_3t_4 ?(min|mins|minute|minutes)	frequency	*$t_1t_2t_3t_4$	*0115	every 1 hour and 15 minutes
((one|1) (times| time)|once)	count	t000t_4	t0001	once
((two|a couple of|a few|few|2) (times| time)|twice)	count	t000t_4	t0002	twice
(several|many|a lot of|multiple|5) (times| time)	count	t000t_4	t0005	several times
(t_4|t_4 or t_4+1|t_4 to t_4+1|t_4-t_4+1) ?(time|times)	count	t000t_4	t0003	3 times

Table 5.10: Detailed implementation specification.

Item	Specification
CPU	Intel Xeon W-2123 CPU(3.60 GHz)
GPU	NVIDIA GeForce GTX 1080 ti, 11 GB memory
Driver	NVIDIA graphic driver ver. 416.34
CUDA	Version 10.0
OS	Windows 10, 64-bit
Python	Version 3.6.6
Pytorch	Version 1.5.1

Table 5.11: Hyperparameters for fine-tuning.

Hyperparameter	Assignment
α	0.25
β	1.
γ	0.25
max training epoch	9
batch size	32
learning rate	$4e-5$
dropout rate	0.1
optimizer	AdamW

A synthetic data set was generated by the following steps: 1) Template generation; 2) Random timestamps generation; and 3) Rule-based timestamps-to-texts translation. Table 5.9 summarises a set of rules used for timestamp-to-text translation and the examples of generated texts.

5.D Experimental Settings

The detailed specification of hardware and software is summarised in Table 5.10. For model deployment, PyTorch version of BERT with the pre-trained weights `bert-base-uncased` (Wolf et al., 2019) was used. Table 5.11 summarises hyperparameter values used for the experiments. All hyperparameters are obtained based on non-exhaustive experiments. During the inference phase, we followed the settings from the original paper (Devlin et al., 2019) to compute the scores of a candidate span.

Chapter 6

Medical Code Prediction: Multi-Label Classification

This chapter was previously published as:

Shim, H., Lowet, D., Luca, S.,& Vanrumste, B. (2022, July). An exploratory data analysis: the performance differences of a medical code prediction system on different demographic groups. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, the 4th Clinical Natural Language Processing Workshop*, pp. 93–102.

From the previous chapters, we have observed how data-efficient methods, either data augmentation with semi-supervised learning (Chapter 3) or active learning (Chapter 4) and synthetic data generation with multi-task learning (Chapter 5), can improve the performance of NLP models in low-resource settings. So far we haven't explored the option of using external resources, such as prior knowledge about the data or existing knowledge bases. To further investigate the opportunities of utilising external resources, we move into a clinical domain, where domain-specific knowledge plays a critical role. For this, we conducted an exploratory data analysis study to identify the gap in the clinical NLP (Ch 6.1) and proposed methods to fill the identified gap (Ch 6.2).

In Section 6.1, we study a clinical benchmark dataset for medical code prediction which is often formulated as a multi-label classification problem. This study has inspired a series of papers studying fairness concerns of biased ML systems

from various application domains. We were intrigued by this and investigated whether a clinical benchmark dataset contains underlying bias, such as class imbalance, and its effect on a model's performance. Data analysis results indicate that the benchmark dataset and showed that the benchmark dataset is imbalanced in terms of label classes and demographics. Analysis results also reveal the performance differences of the benchmark-trained model in different demographic groups. Finally, we found a negative correlation between label distribution distance and performance. This result implies that the trained model performs poorly in the group that contains data whose label set is different from the global label distribution of the entire data.

Section 6.2 presents a study that aims to address the performance differences in different demographic groups. This work is motivated by the findings from the previous study that the model tends to perform differently across demographic groups. Specifically, we focus on age groups where the performance differences are most pronounced. The goal is to build a model that performs equally well across different groups. To this end, we propose two approaches. The first method is to build an ensemble model, which consists of multiple group-specific models. Further, we propose a novel weighted loss function that utilises the prior knowledge of data distributions. The proposed loss function encourages the model to use per-class weights decided based on the group-specific label distribution known from the training dataset. The second method formulates the medical code prediction task as a binary classification problem and proposes a novel binary classification architecture. The proposed architecture takes a document text and label information (e.g., a label name) as inputs. We also propose a data augmentation method for the proposed binary classification architecture. The proposed data augmentation method replaces the label name in the input with its synonyms extracted from the knowledge database. Results show that the ensemble approach with the proposed loss function improves global performance scores and the binary classification approach performs equally well across different age groups achieving high fairness scores. Experimental results also indicate the limitation of the proposed data augmentation method that degenerates performance.

This chapter studies the following research questions:

RQ1. How can we fine-tune a pre-trained language model when only a small-sized training set for the target task is available?

RQ3. Can we exploit other resources (e.g., knowledge, databases, et cetera) during fine-tuning to improve the performance of a pre-trained language model?

6.1 Data Analysis Study

6.1.1 Introduction

Medical coding is the process of assigning standard codes, such as The International Classification of Diseases (ICD) codes, to each clinical document for documenting records and medical billing purposes. Even though medical coding is an important process in the healthcare system, it is expensive, time-consuming, and error-prone (O'malley et al., 2005).

Researchers have investigated approaches for automated ICD coding systems and there has been great progress with neural network architectures (Kalyan and Sangeetha, 2020). However, current state-of-the-art models still suffer from data imbalance issues: since the benchmark dataset is imbalanced in terms of assigned ICD codes, the model performances differ across ICD codes (Mullenbach et al., 2018; Li and Yu, 2020; Kim and Ganapathi, 2021; Vu et al., 2021; Ji et al., 2021). Moreover, a recent study argues that the performances of models tend to decrease when the ICD codes have fewer training instances (Ji et al., 2021).

Based on this observation from the literature (i.e., imbalanced ICD code distribution results in the performance imbalance between the ICD codes), the goal of this paper is to investigate the effect of the imbalance of different demographic groups in the training data set on the performances of the demographic groups. More specifically, we study the following questions: 1) Is a benchmark dataset for medical code prediction imbalance in terms of the data subject's demographic variables (i.e., age, gender, ethnicity, socioeconomic status)?; 2) If so, would it result in performance differences between demographic groups? To answer these questions, we analyse the benchmark dataset, reproduce one of the state-of-the-art models (Li and Yu, 2020), and analyse the performance of the model. To the best of our knowledge, this is the first attempt to study the demographic imbalance of the medical code prediction benchmark dataset and analyse the performance differences between demographic groups.

Our contribution is three-fold. Firstly, we analysed the medical code prediction benchmark dataset to investigate the underlying imbalance in the dataset (Section 6.1.4) and reproduced one of the state-of-the-art medical code prediction models proposed by Li and Yu (2020). Secondly, we propose sample-based evaluation metrics (Section. 6.1.3) to identify potential biases inside a model and potential risk of the bias (Section. 6.1.4). Thirdly, we propose a simple label distance metric to quantify the differences in the label distribution between each group and the global data (Section. 6.1.3) and found that the label distance metric is strongly correlated with the performance negatively (Section. 6.1.4).

We expect that these analytic results could provide a valuable insight to the natural language processing (NLP) research community working for clinical applications.

6.1.2 Data

This section includes the information on the benchmark dataset used and the details of pre-processing steps taken for preparing data for the experiments. Note that we followed the previous approach to reproduce the result from the literature. More details are explained in the following subsections.

MIMIC-III dataset

We used Medical Information Mart for Intensive Care (MIMIC-III v1.4.) dataset (Johnson et al., 2016)[1] for the experiments. MIMIC-III is the benchmark dataset that has been widely used to build a system for automated medical code prediction (Shi et al., 2017; Mullenbach et al., 2018; Li and Yu, 2020; Kim and Ganapathi, 2021). For medical code prediction, discharge summary texts[2] are used as inputs and corresponding ICD-9 codes[3] are used as output of a system. In other words, the medical code prediction is formulated as a multi-label classification where the ground truth of the given input includes one or more ICD-9 codes.

For benchmarking purposes, Mullenbach et al. (2018) provides script codes that pre-process the discharge summary text data and splits the dataset by patient IDs into training, validation, and testing sets[4]. Also, Mullenbach et al. (2018) creates two benchmark sets, with full ICD codes as well as with the top 50 most frequent ICD codes, which are denoted as `MIMIC-III full` and `MIMIC-III 50`, respectively. The `MIMIC-III full` dataset contains 52,728 discharge summaries with 8,921 unique ICD codes and the `MIMIC-III 50` dataset contains 11,366 discharge summaries with 50 unique ICD codes.

In this paper, we only consider the `MIMIC-III 50` dataset. Following the previous works (Li and Yu, 2020; Kim and Ganapathi, 2021; Vu et al., 2021), we used Mullenbach et al. (2018)'s scripts to split the data which results in 8,066 discharge summaries for training, 1,573 for validation, and 1,729 for testing. Additionally, we extracted patients' demographic information from the

[1] https://physionet.org/content/mimiciii/1.4/

[2] A discharge summary is a note that summarises information about a hospital stay

[3] MIMIC-III dataset includes both diagnoses and procedures which occurred during the patient's stay

[4] https://github.com/jamesmullenbach/caml-mimic

MIMIC-III dataset, including gender, age, ethnicity, and insurance type as a socioeconomic proxy.

Data pre-processing

Discharge Summary texts One of our objectives is to reproduce the results by Li and Yu (2020) and analyse the performance. Therefore, we followed the Li and Yu (2020)'s pre-processing steps which are the same as the work by Mullenbach et al. (2018). Data cleaning and pre-processing include the following steps: the discharge summary texts were tokenized, tokens that contain no alphabetic characters were removed, and all tokens were lowercased. All documents are truncated to a maximum length of 2500 tokens. More details can be found in the original paper (Mullenbach et al., 2018).

Demographic data In the MIMIC-III dataset, each unique hospital visit for a patient is assigned with a unique admission ID. Therefore we used admission ID to extract the demographic information of patients. The following steps were taken to pre-process the demographic data: firstly, age values are computed based on the date of birth data and the admission time data[5]. Secondly, the four most frequent values in ethnicity data, including 'WHITE', 'BLACK', 'ASIAN', 'HISPANIC', are being kept, whereas the remaining values are combined into one group and labelled as 'OTHER'. Thirdly, the three most frequent values in insurance type data, including 'Medicare', 'Private', 'Medicaid', are being kept, whereas the other values are combined into one group 'Other'.

6.1.3 Methods

Data analysis

We analysed the size, as well absolute as relative, of each group and investigated relationships between variables. Also, we analysed the length of discharge summary notes and the number of assigned ICD codes per note to investigate relationships between the length of notes and demographic variables and between the number of ICD codes per note and demographic variables. We also calculate the differences in the ICD code label distributions between the entire data and each group.

[5]The date of birth data of patients older than 89 have been shifted and the original values cannot be recovered. Therefore, we assigned the same age value of 90 to all patients who are older than 89.

Label distribution distance metric

To calculate the differences in the ICD code label distributions between the entire data and each group, we used cosine distance[6] between ICD code label representations, each of which is a multi-hot vector $\mathbb{R}^{1\times 50}$. Specifically, we compute the average distances between the globally averaged label vector and the label vector of each data point in groups, which is defined as:

$$\text{distance}_g = \frac{1}{N_g}\sum_{i=1}^{N_g} 1 - \frac{\boldsymbol{u}\cdot\boldsymbol{v_i}}{||\boldsymbol{u}||_2||\boldsymbol{v_i}||_2} \tag{6.1}$$

where $\boldsymbol{u}$ is the globally averaged label vector of the entire data and $\boldsymbol{v_i}$ is a label vector of a single data point in the group D_g that contains N_g of data points. A low distance score means the group contains patients whose label set is close to the global label distribution of the entire data.

Medical code prediction model

In this study, we study one of the state-of-the-art medical code prediction models proposed by Li and Yu (2020). There are three important architectural details in Li and Yu (2020)'s model: firstly, it uses a convolutional layer with multiple filters where each filter has a different kernel size (Kim, 2014). This multi-filter convolutional layer allows a model to capture various text patterns with different word lengths. Secondly, residual connections (He et al., 2016) are used on top of each filter in the multi-filter convolutional layer. This residual convolutional layer enlarges the receptive field of the model. Thirdly, the label attention layer (Mullenbach et al., 2018) is deployed after the multi-filter convolutional layer. More details on the model architecture can be found in the original paper (Li and Yu, 2020). For implementation, we re-trained a model by using a script[7] and followed the same hyperparameter setting except the early-stopping setting: we used a macro-averaged F1 score as an early-stopping criterion with a patience value 10.

Evaluation metrics

Performance metrics To evaluate the model's performance, micro-and macro-averaged F1 scores are widely used in the literature (Shi et al., 2017; Mullenbach

[6]We used cosine distance because it is widely used to calculate the similarity between high-dimensional vectors and the distance is always normalised between 0 and 1.

[7]https://github.com/foxlf823/Multi-Filter-Residual-Convolutional-Neural-Network

et al., 2018; Li and Yu, 2020). Micro-averaged scores are calculated by treating each <text input, code label> pair as a separate prediction. Macro-averaged scores are calculated by averaging metrics computed per label. For recall, the metrics are computed as follows:

$$\text{Micro-R} = \frac{\sum_{k=1}^{K} \text{TP}_k}{\sum_{k=1}^{K} \text{TP}_k + \text{FN}_k} \tag{6.2}$$

$$\text{Macro-R} = \frac{1}{|K|} \sum_{k=1}^{K} \frac{\text{TP}_k}{\text{TP}_k + \text{FN}_k} \tag{6.3}$$

where TP_k and FN_k, denote true positive examples and false negative examples for a specific ICD-9 code label k, respectively. Since we use `MIMIC-III 50` dataset, $|K|$ equals 50

Since we focus on performance differences in terms of data subject's demographics, we additionally use sample-averaged F1 scores. Sample-averaged scores are calculated by computing scores at the instance level and averaging over all instances in the data set. For sample-averaged recall, the metric is computed as follows:

$$\text{Sample-R} = \frac{1}{|N|} \sum_{i=1}^{N} \frac{|\boldsymbol{y}_i \cap \hat{\boldsymbol{y}}_i|}{|\boldsymbol{y}_i|} \tag{6.4}$$

where $\boldsymbol{y}_i$ and $\hat{\boldsymbol{y}}_i$ denote the ground truth labels and the predicted labels for the i-th test example, respectively and N denotes the total number of test samples. Precision is computed in a similar manner.

For statistical analysis, we conducted the Kruskal-Wallis tests to investigate differences between the average performance scores of each group. Also, we computed the Pearson correlation coefficient and p-value for testing the correlation between the training data size of the group and the model performance on the group and between label distance of the group and the model performance on the group. All statistical tests were done by using sample-F1 scores.

Error metrics Following previous studies (Hardt et al., 2016; Chouldechova, 2017), we consider two metrics to quantify the error of a trained model: false negative rate (FNR) and false positive rate (FPR) in the sample level. FNR is the fraction of ICD codes that are failed to be predicted by a system but included in a ground truth label set. FPR is the fraction of ICD codes that are

Table 6.1: Error types for computing FNR and FPR.

Error type	Target	Prediction
$\text{FN}_{i,k}$	$y_{i,k} = 1$	$\hat{y}_{i,k} = 0$
$\text{FP}_{i,k}$	$y_{i,k} = 0$	$\hat{y}_{i,k} = 1$
$\text{TN}_{i,k}$	$y_{i,k} = 0$	$\hat{y}_{i,k} = 0$
$\text{TP}_{i,k}$	$y_{i,k} = 1$	$\hat{y}_{i,k} = 1$

erroneously predicted by a system but not included in a ground truth label set. High FNR scores imply low recall scores and high FPR implies high probability of false alarms. Two metrics are computed as follows:

$$\text{FNR} = \frac{1}{|N|} \sum_{i=1}^{N} \frac{\text{FN}_i}{\text{FN}_i + \text{TP}_i} \tag{6.5}$$

$$\text{FPR} = \frac{1}{|N|} \sum_{i=1}^{N} \frac{\text{FR}_i}{\text{FP}_i + \text{TN}_i} \tag{6.6}$$

where $\text{FN}_i = \sum_{k=1}^{K} \text{FN}_{i,k}$, $\text{FP}_i = \sum_{k=1}^{K} \text{FP}_{i,k}$, $\text{TN}_i = \sum_{k=1}^{K} \text{TN}_{i,k}$, and $\text{TP}_i = \sum_{k=1}^{K} \text{TP}_{i,k}$. Table 6.1 summarises how $\text{FN}_{i,k}$, $\text{FP}_{i,k}$, $\text{TP}_{i,k}$, and $\text{TN}_{i,k}$ are computed for each data point x_i and its ground truth label set $y_i = \{y_{i,1}, ..., y_{i,K}\}$.

To assess the risk of errors, we use the worst-case comparison method (Ghosh et al., 2021). Also, we conducted Mann–Whitney U tests to investigate the differences between the error scores of the best and the error scores of the worst models.

6.1.4 Results

Data analysis results

Table 6.2 summarizes the sample sizes of the data set. It is shown that only gender variables are well-balanced. For age groups, patients who are 50-89 take up to 71.2% of the data. Also, the data set includes more White patients than patients from other ethnic groups. Also, more than half of the entire patients in the data set are patients with Medicare insurance and only 8.8% of patients are with Medicaid insurance.

Figure 6.1 shows the relationship between insurance types, Medicare and Medicaid, and ethnicity variables. It is observed that insurance type has

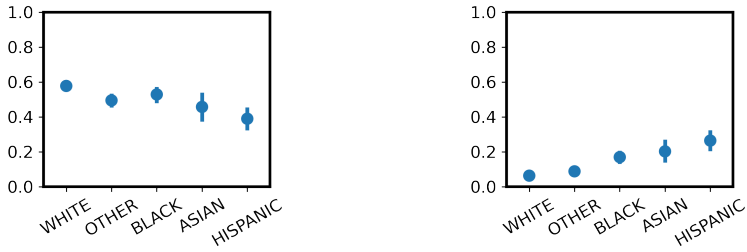
Table 6.2: Sample size (absolute and relative) of the groups of gender, age, ethnicity, and insurance type.

	Count (n)	**Percentage (%)**
Total	8066	
Gender		
F	3593	44.5
M	4473	55.5
Age		
0-17	440	5.5
18-29	300	3.7
30-49	1148	14.2
50-69	2931	36.3
70-89	2817	34.9
90+	430	5.3
Ethnicity		
WHITE	5651	70.1
OTHER	1097	13.6
BLACK	799	9.9
HISPANIC	311	3.9
ASIAN	208	2.6
Insurance		
Medicare	4440	55.0
Private	2636	32.7
Medicaid	709	8.8
Other	281	3.5

a certain relationship with the patient's race: 57.7% of White patients are paying with Medicare, whereas 38.9% of Hispanic patients are paying with Medicare. On the other hand, 26.4% of Hispanic patients are paying with Medicaid, whereas only 0.63% of White patients are paying with Medicaid.

Figure 6.2 illustrate the age distribution of each insurance type. Medicare and Medicaid are two separate, government-run insurance in the United States. Medicare is available for people age 65 or above and younger people with severe illnesses and Medicaid is available to low-income individuals under the age of 65 and their families. Because of the eligibility criteria for Medicare, Medicare includes more older patients compared to other insurance types, as we can see from the Figure 6.2.

Figure 6.3a and Figure 6.3b show the distribution of the length of a discharge summary note and the number of ICD codes assigned per note, respectively.



(a) Percentage of Medicare within each ethnic group

(b) Percentage of Medicaid within each ethnic group

Figure 6.1: Relationship between insurance and demographic variables. 95% confidence intervals are illustrated by lines.

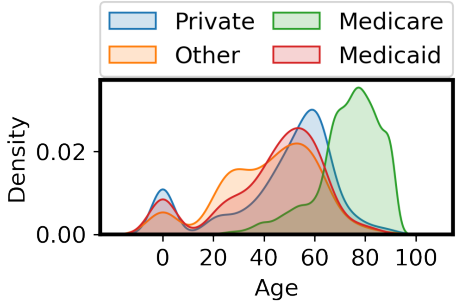
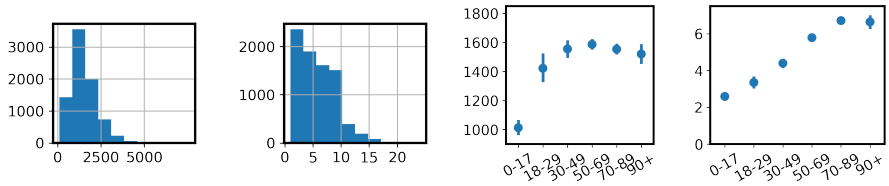


Figure 6.2: Kernel density estimate plot for visualising the age distribution of each insurance type

The average length is 1529.7 (std=754.9) and the average number of codes per note is 5.7 (std=3.3). Figure 6.3c and Figure 6.3d illustrate relationship between patients age and the length of note and the number of codes per note, respectively. From Figure 6.3c, it is observed that the length of note tends to increase until age group 50-69 and starts to decrease afterwards. From Figure 6.3d, positive correlations between age and the number of ICD codes per note are observed. Other noticeable patterns are not observed in other demographic variables (i.e., gender, insurance, ethnicity) with the respect to the length of a discharge summary note and the number of ICD codes assigned per note.

Figure 6.4 illustrates ICD code distributions. Figure 6.4a shows the entire data



(a) #. tokens/note (b) #. codes/note (c) #. tokens/note (d) #. codes/note

Figure 6.3: The distribution of the length of a discharge summary note (a) and the number of ICD codes assigned per note (d). Relationship between the length of notes and age groups (c) and between the number of ICD codes per note and age groups (d). X-axes indicate the average number of tokens in a note (a, c) and the average number of ICD codes per note (b,d). 95% confidence intervals are illustrated by lines.

Table 6.3: Average label distribution distances between each group and the global data. Standard deviations are added in parentheses.

	Distance
Gender	
F	0.613 (0.137)
M	0.615 (0.133)
Age	
0-17	0.737 (0.097)
18-29	0.746 (0.111)
30-49	0.684 (0.133)
50-69	0.610 (0.129)
70-89	0.564 (0.116)
90+	0.560 (0.118)
Ethnicity	
WHITE	0.610 (0.135)
OTHER	0.607 (0.131)
BLACK	0.633 (0.135)
HISPANIC	0.646 (0.135)
ASIAN	0.626 (0.143)
Insurance	
Medicare	0.579 (0.124)
Private	0.653 (0.135)
Medicaid	0.658 (0.136)
Other	0.691 (0.139)

set has long-tail distribution. Between female and male patient groups, no

noticeable difference between the label distributions is not observed. In terms of insurance type and ethnicity, each group shows slightly different ICD code distributions. Clear differences are observed between age groups: patients whose ages are younger than 30 (0-17, 18-29) show less spread ICD code distributions with fewer ICD codes than other age groups. The label distribution distances between each group and the global data are summarised in Table 6.3. Similar to the observations from Figure 6.4, age groups 0-17 and 18-29 have the bigger distance scores.

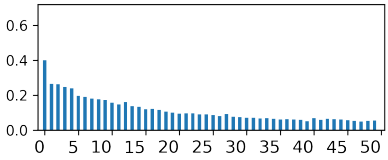
Performance & error analysis results

Table 6.4 summarises the prediction results on the test set. It is observed that a re-trained model slight underperforms compared to the original model (Li and Yu, 2020). The different early-stopping settings might cause this difference. Both models achieve higher scores in micro-averaged metrics than macro-averaged metrics, which means the model's performance on rare labels is worse than on frequent labels. The sample-averaged metrics are higher than macro-averaged metrics but lower than micro-averaged metrics.

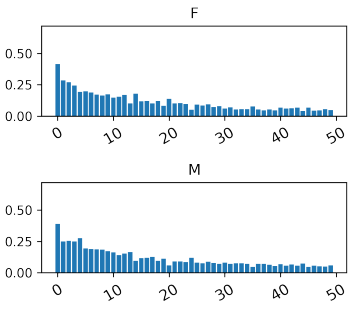
Noticeable performance differences are observed between age groups, especially between patients younger than 30 years (18-29) and older than 90 (90+). The percentages of both groups in the training set are low but patients younger than 30 years get distinctively worse predictions in terms of all F-1 scores. Between different ethnic groups, it is observed that Hispanic and Asian patients get worse predictions compared to other patients. Between insurance types, it is also observed that patients with other types of insurance and Medicaid insurance get worse predictions compared to patients with Medicare and Private insurance in sample-averaged F-1 scores.

As the result of the Kruskal-Wallis test, we found statistically significant differences in sample-averaged F1 scores according to age group ($H(4)=46.57$, $p<0.001$) and insurance type ($H(3)=18.58$, $p<0.001$), separately. Close to being statistically significant is found according to gender ($H(1)=3.65$, $p=0.056$) and no statistically significant difference is found according to ethnicity ($H(4)=2.657$, $p=0.657$).

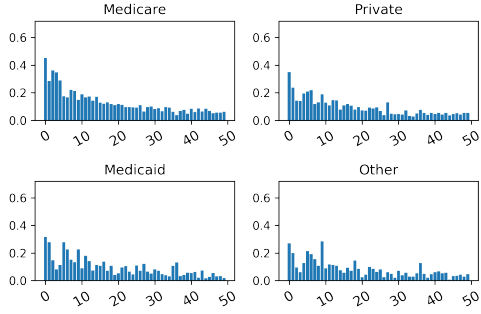
Error metrics per group are summarised in Table 6.5. Error metrics between groups show a similar trend as the performance metrics: differences between age groups are the most pronounced. It is observed that FNR scores tend to decrease as age increases. However, the largest difference between age groups is not significant ($p=0.06$). FPR also tends to increase as the age increases in the age groups under 90 and the largest difference between the younger group (18-29) and the older group (70-89) is significant ($p<0.001$). Patients with



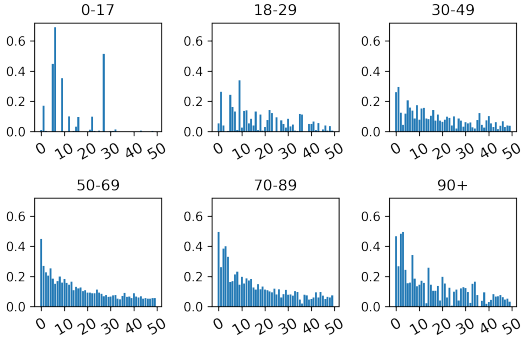
(a) Entire data set



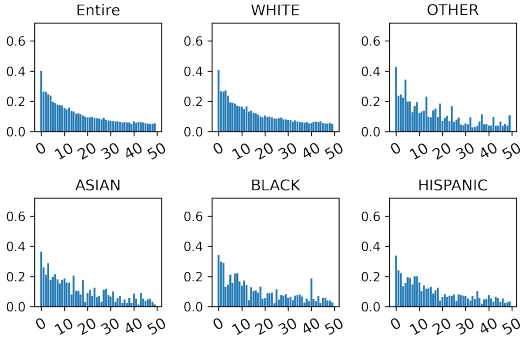
(b) Gender



(c) Insurance type



(d) Age group



(e) Ethnicity

Figure 6.4: ICD code distribution. X-axis indicates the sorted ICD code class label and Y-axis indicate the percentage of labels observed in the training set.

Table 6.4: Performances on the MIMIC-III 50 test set. † indicates performances reported in the paper by Li and Yu (2020). Other results are obtained from a reproduced model. The percentage of training samples (%) is added in parentheses after the group labels. Best performances are boldfaced and worst performances are underlined.

	F-1 (%)		
	Micro	**Macro**	**Sample**
Total			
Li and Yu (2020)	67.3†	60.8†	-
Reproduced	64.4	59.2	60.6
Gender			
F (44.5)	<u>63.2</u>	<u>58.1</u>	<u>59.7</u>
M (55.5)	**65.3**	**59.4**	**61.4**
Age			
18-29 (3.7)	<u>53.9</u>	<u>36.1</u>	<u>48.2</u>
30-49 (14.2)	58.9	58.2	52.4
50-69 (36.3)	64.2	57.7	60.9
70-89 (34.9)	65.6	**59.2**	63.6
90+ (5.3)	**67.1**	55.9	**65.0**
Ethnicity			
WHITE (70.1)	64.3	59.2	60.8
OTHER (13.6)	64.3	**60.9**	60.7
BLACK (9.9)	**66.2**	60.2	**61.7**
HISPANIC (3.9)	<u>62.0</u>	54.6	<u>56.0</u>
ASIAN (2.6)	64.7	<u>51.2</u>	59.3
Insurance			
Medicare (55.0)	**65.3**	58.4	**62.5**
Private (32.7)	63.4	58.8	59.0
Medicaid (8.8)	62.9	**59.3**	57.8
Other (3.5)	<u>56.0</u>	<u>49.3</u>	<u>50.5</u>

other types of insurance take significantly worse scores compared to Medicare patients in terms of FNR scores. Interestingly, FPR shows different patterns. For example, patients with Medicare get the worst FPR scores and patients with Private insurance get the best FPR scores.

Correlation test result

As the result of correlation tests, we found a weak positive correlation (0.43, p=0.09) between training set size and performance. This result shows that

Table 6.5: Errors on the MIMIC-III 50 test set. The percentage of training samples (%) is added in parentheses. Best performances are boldfaced and worst performances are underlined. * and *** indicate the error of the worst model is greater than the error of the best with statistical significance of p=0.05 and p=0.001 (Mann–Whitney U test), respectively.

	FNR (%)	**FPR (%)**
Total	40.6	3.8
Gender		
F (44.5)	39.7	4.3
M (55.5)	**38.0**	**4.2**
largest diff. (↓)	1.7	0.1
smallest ratio (%) (↑)	95.8	98.2
Age		
18-29 (3.7)	46.2	**2.9**
30-49 (14.2)	45.9	3.3
50-69 (36.3)	39.5	3.9
70-89 (34.9)	35.7	5.0
90+ (5.3)	**34.1**	4.4
largest diff. (↓)	12.2	2.1***
smallest ratio (%) (↑)	73.7	57.7
Ethnicity		
WHITE (70.1)	38.7	4.2
OTHER (13.6)	39.3	4.5
BLACK (9.9)	**37.0**	4.2
HISPANIC (3.9)	42.5	4.2
ASIAN (2.6)	40.3	**3.8**
largest diff. (↓)	5.4	0.8
smallest ratio (%) (↑)	87.2	83.3
Insurance		
Medicare (55.0)	**37.0**	4.7
Private (32.7)	40.7	**3.4**
Medicaid (3.5)	41.0	3.6
Other (8.8)	46.9	4.2
largest diff. (↓)	9.8*	1.3***
smallest ratio (%) (↑)	79.0	71.5

even though the model performs well for groups with more training data in general, the relationship is not statistically significant. Contrary to this result, we found a very strong negative correlation (-0.95, p<0.001) between label distance and performance. This result implies that the model performs poorly

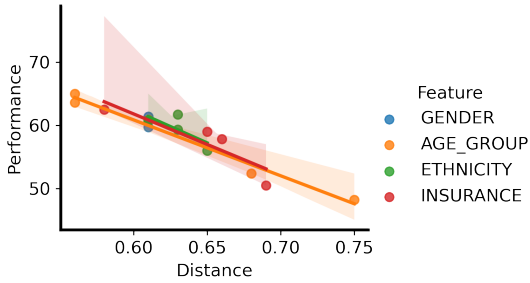


Figure 6.5: Label distance of each group and the model performance on each group. Linear relationships are illustrated by lines determined through linear regression.

in the groups containing many patients whose label set is different from the global label distribution of the entire data. The group-specific correlations between label distances and the performances are illustrated in Figure 6.5. It is observed that the negative correlation is much more pronounced between different age groups than in other groups.

6.1.5 Discussion

Impact of the study. The MIMIC-II dataset for medical code prediction provides opportunities to develop and benchmark models and facilitates natural language processing research in the clinical domain. Since it is one of the most frequently used benchmark datasets for medical code prediction, it has a huge impact on the quality of the developed models. For example, previous studies (Mullenbach et al., 2018; Li and Yu, 2020; Kim and Ganapathi, 2021; Vu et al., 2021; Ji et al., 2021) have shown that the ICD code distribution in the MIMIC-III dataset is imbalanced and it results in performance differences between ICD codes. In this study, we investigated the data imbalance of the MIMIC-III 50 data, in terms of the data subject's demographic factors, and its effect on the model performance for ICD code prediction.

Evaluation metrics for fairness. In this paper, we proposed metrics that can correctly evaluate the model's performance in terms of individual patients' benefits and potential harms. Especially, we formulated the medical code prediction task as a multi-label classification task. From a machine learning perspective, sample-based metrics and label-based metrics are used to evaluate

the performance of a model in a multi-label classification task (Zhang and Zhou, 2013). Sample-based and label-based metrics focus on different aspects of model performance, one in sample-wise performance and the other in label-wise performance. However, label-based metrics are more frequently used in the literature (Xiao et al., 2018; Mullenbach et al., 2018; Li and Yu, 2020; Kim and Ganapathi, 2021; Vu et al., 2021; Ji et al., 2021). Considering a healthcare application setting where all patients are expected to receive an equal quality of service, we argue that using sample-based metrics is required to evaluate the model performance. Also, we propose to use disaggregated metrics (Barocas et al., 2021), which are metrics evaluated on each group of data, to ensure that a model is equally accurate for patients from different demographic groups (Rajkomar et al., 2018; Gichoya et al., 2021).

Correlation between demographic variables We analysed the MIMIC-III dataset to identify the underlying data imbalance of demographic variables. Our data analysis results show that the MIMIC-III dataset is imbalanced in terms of the data subject's demographics. However, we also found a correlation between demographic variables. For example, age is correlated with insurance type: patients older than 65 are likely to be insured with Medicare. This confounding factor across demographic variables makes it complicated to interpret the main effects of the data subject's demographics on the model performance.

Correlation between label distance and performance Based on the previous study arguing the performances of models tend to decrease when the ICD codes have fewer training samples (Ji et al., 2021), we hypothesised that the performance of the model on a demographic group is correlated with the number of data of that group in the training data set. However, the analysis results do not support this hypothesis: even though the performance differences are observed across some demographic groups (i.e., across age groups and insurance types), the correlation between the number of training data of the group and the performance of the group is weak. Instead, we found that the label distance of the group is negatively correlated with the performance of the group. This result suggests that when the group contains patients whose label set is different from the global label distribution of the entire data, it is likely that the model performs poorly in that group.

In terms of machine learning perspective, this issue can be seen as a label shift: the train and test label distribution is different while the feature distribution remains the same (Lipton et al., 2018; Guo et al., 2020). To address this issue, one interesting area for future work may be in re-training the classifier with adjusted training sample weights (Lipton et al., 2018) or adapting the predictions

of a pre-trained classifier (Saerens et al., 2002; Du Plessis and Sugiyama, 2014; Alexandari et al., 2020).

Limitations and future directions There are several limitations to this study. Firstly, we used a subset of MIMIC-III data with the top 50 most frequent ICD codes to simplify the analysis. Since the full MIMIC-III dataset contains more than 47,000 ICD codes, further study is required. Secondly, we only studied the model proposed by (Li and Yu, 2020). One potential direction is to investigate the performance of models using pre-trained language models (Zhang et al., 2020; Ji et al., 2021). Thirdly, we found an issue of confounding factors across demographic variables, which makes it complicates the interpretation of the main effects of the data subject's demographic factors on the model performance. To address this issue, further analysis of multiple intersectional groups or causal analysis is required. In future work, we will also investigate how to build a model that can perform equally well on across all demographic groups.

6.1.6 Conclusion

In this study, we performed an empirical analysis to investigate the data imbalance of the MIMIC-III 50 dataset and its effect on the model performance for ICD code prediction. We found that demographic imbalance exists in the MIMIC-III 50 dataset and a medical code prediction model performs differently across some demographic groups. Interestingly, the correlation between the number of training data of the group and the performance of the group is weak. Instead, we found a negative correlation between the label distance of the group and the performance of the group. This result suggests that the model tends to perform poorly in the group whose label distribution is different from the global label distribution. Potential future research direction includes further analysis of the main effects of the data subject's demographic factors on the model performance and investigation of building a robust and fair model that can perform equally well across demographic groups with different label distributions.

6.2 Model Development Study

6.2.1 Introduction

Researchers have been working on applying NLP technologies for clinical use cases, such as a medical code assignment task. For example, medical code assignment is formulated as a multi-label text classification task. The goal is to train a system that can assign standard medical codes, such as the International Classification of Diseases (ICD) code (Organization et al., 1978), to a given clinical document (i.e., discharge summary). This task is often called medical code prediction which is not trivial because clinical texts contain a lot of medical terms and abbreviations. Clinical texts are also generally very long and text fragments that contain linked information are scattered over documents (Vu et al., 2021). Further, medical knowledge is required to analyse text (Prakash et al., 2017). Moreover, there is a large number of codes with long-tail distribution: some codes are frequently observed in the dataset but others are less frequent and may only have a few instances in the dataset (Shi et al., 2017; Xie et al., 2019). This data imbalance results in the performance differences across ICD code classes (Ji et al., 2021) which can also create a biased model that performs differently across demographic groups (Shim et al., 2022).

We aim to address the challenges in the development of an NLP system for medical code assignment. Specifically, we focus on the performance differences across demographic groups caused by imbalanced data (Shim et al., 2022) and investigate how to mitigate this. To this end, we propose two approaches: the first approach is to create group-specific models by further fine-tuning the model on group-specific data with a novel weighted loss function. The second approach is to formulate a multi-label classification task as a binary classification task by providing label information (e.g., label name) as an input. Moreover, we propose a data augmentation method that replaces the label name in the input with its synonyms extracted from the knowledge database.

The first approach is motivated by two observations: firstly, the traditional weighted binary cross entropy loss function applies a positive weight to the loss from positive samples by trading off between precision and recall. Generally, this weight is defined as the number of positive samples over the number of negative samples in the data. Secondly, in our previous study (Chapter 6.1), we show that each group has different label distribution. For example, some labels are more frequent in a group than others. By combining these two observations, we propose a loss function that defines class-specific weights based on the group-specific label distributions which we refer to as distribution-aware weighted loss

(DAWL) (Section 6.2.3). In experiments, we compare the proposed DAWL to the binary cross-entropy, weighted binary cross-entropy, focal loss (Lin et al., 2017), and asymmetric loss (Ridnik et al., 2021).

In the second approach, we formulate a multi-label classification task as a binary classification task by utilising label information as an input (Section 6.2.3). We refer to this approach "As Binary Classification (ABC)" approach. For example, a model takes a clinical document and a specific ICD code as an input and predicts whether the given ICD code has been assigned or not. Since this approach utilises the label information, it can create better label-specific feature representations. This approach has been proved to be effective, especially when used with the attention-based model (i.e., BERT) because providing a label as an input can be seen as providing a hint to the model where to focus (Sun et al., 2019a; Halder et al., 2020; Shim et al., 2021). It also has the effect of increasing the size of the training dataset by expanding a sentence into multiple sentence-label pairs. This binary classification approach can also be used with other model architectures, such as CNN-based Siamese networks (Koch et al., 2015). Together with the ABC formulation, we propose a data augmentation method that replaces a label name in a input with its synonyms. To this end, we utilise a database, Universal Medical Language System (UMLS)(Bodenreider, 2004), that contains information related to ICD label classes (e.g., ICD code names, disease descriptions, synonyms, etc) for augmenting the training data. We compare the proposed ABC approach to multi-label classification approaches and investigate how ABC formulation and data augmentation affect performance.

Experimental results indicate that performance improves when using the knowledge of the group-specific label distribution for weighting losses. Further, results demonstrate that the ABC approach can achieve equally good performances across demographic groups compared to the multi-label classification model which performs unequally across demographic groups creating a large demographic disparity. Error analysis reveals interesting model behaviours: the model with the proposed weighted loss function and ABC approach improve the missing rate (false negative rate) substantially while marginally increasing the false alarm rate (false positive rate) compared to the baseline models. Since there are extremely many medical codes and human coders are prone to missing assigning codes, we expect that a system that achieves a low missing rate could support human coders.

6.2.2 Related Works

Loss functions

Weighted Binary Cross-Entropy Loss Binary cross entropy loss (L_{BCE}) is defined as Eq. 6.7. Since BCE loss treats positive samples and negative samples equally, the losses from negative samples overwhelm the losses in positive samples when the training dataset contains more negative samples than positive samples. To mitigate this, typically weighted binary cross entropy loss (L_{WBCE}) is used to trade off recall and precision by adding a positive weight w:

$$L_{BCE} = -\sum_{i=0}^{N}(y_i log(\hat{y_i}) + (1 - y_i) log(1 - \hat{y_i})) \tag{6.7}$$

$$L_{WBCE} = -\sum_{i=0}^{N}(w y_i log(\hat{y_i}) + (1 - y_i) log(1 - \hat{y_i})) \tag{6.8}$$

where y_i and $\hat{y_i}$ are the ground truth label and the model output (i.e., sigmoid output) for ith sample, respectively. N is the number of samples. For simplicity, we only consider a binary loss from a label class l_k in an entire label set $l_k \in \{l_1, ..., l_K\}$ for multi-label classification. As it is shown in Eq. 6.8, weighted binary cross entropy adds a positive weight w to change the importance of misclassifying positive samples. If $w > 1$, the model more to increases the recall. On the other hand, if $w < 1$, the model increases the precision.

For simplicity, we will use the following expressions:

$$L_{BCE} = -L_+ - L_- \tag{6.9}$$

$$L_{WBCE} = -wL_+ - L_- \tag{6.10}$$

Focal Loss Instead of scaling the losses from positive samples, focal loss (L_{FL}) is proposed to deal with the class imbalance issue by down-weighting the losses assigned to well-classified examples (Lin et al., 2017). Focal loss is defined as:

$$L_{FL} = -(1 - \hat{y})^{\gamma} L_+ - \hat{y}^{\gamma} L_- \tag{6.11}$$

which includes a modulating factor ($(1-\hat{y})^{\gamma}$ for positive samples and $\hat{y}^{\gamma}$ for negative samples) with a focusing value $\gamma \geq 0$. By setting a large focusing value γ, the contribution of easy examples can be decreased in the loss function resulting in focusing more on difficult samples during training. When using $\gamma = 0$, the loss function equals the original binary cross-entropy loss.

However, the downside of focal loss is that it also down-weights the learning signals from positive samples. In a multi-label classification setting, especially with imbalanced data, it is important to keep the learning signals from positive samples even though the signals are from easy samples.

Asymmetric Loss Motivated by focal loss, asymmetric loss (L_{ASL}) is proposed for a multi-label classification setting where the number of negative samples per category is much higher than the number of positive samples (Ridnik et al., 2021). The main difference between focal loss and asymmetric loss is that asymmetric loss introduces asymmetric focusing, which means that two different focusing factors are used for positive and negative samples, respectively, as defined as:

$$L_{ASL} = -(1-\hat{y})^{\gamma_+} L_+ - \hat{y}^{\gamma_-} L_- \tag{6.12}$$

where $\gamma_- > \gamma_+$ to more focus on the contribution of positive samples than negative samples.

Even though asymmetric loss addresses the imbalance between positive and negative samples, the remaining gap is when the degree of imbalances changes across labels in a multi-label setting. In other words, asymmetric loss equally modulates losses from negative samples without differentiating frequent labels and rare labels.

Binary Classification Formulation

Binary classification formulation is a way of translating a multi-label classification problem into a binary classification problem. A traditional multi-label classification system takes a document as an input and predicts one or more labels. When building a multi-label classification system with neural networks, the final classification layer is employed, which contains multiple neurons with their sigmoid activation functions. Binary classification formulation approach reframes a multi-label classification problem as a binary classification problem by providing label information (i.e., label name) as a part of the input. The benefit of this binary classification formulation is three-fold: firstly, this approach

lessens the burden of a classifier by providing label information as an input. Secondly, this approach narrows down multi-label classification search space (i.e., multiple labels) into binary decision space (either True or False). Thirdly, it increases the size of training data. For example, an input sentence s_i is expanded into multiple sentence-label pairs $(s_i, l_1), (s_i, l_2), .., (s_i, l_K)$ with label category l_k is a label set $l_k \in \{l_1, ..., l_K\}$.

Binary classification formulation can be applied to various tasks. For example, Sun et al. (2019a); Shim et al. (2021) reframe aspect-based sentiment analysis task as a binary classification problem by providing an aspect label as a part of the input. Similarly, Koch et al. (2015); Halder et al. (2020) employ binary classification formulation for few-shot learning. For a model that has an attention mechanism (Bahdanau et al., 2015; Luong et al., 2015), such as Transformer (Vaswani et al., 2017), binary classification formulation allows the model to create better representation by providing the label information as a hint for attention. One drawback of using an attention model is that it requires multiple forward passes which is not practical for applications that deal with a large number of labels, such as medical coding (Ziletti et al., 2022). Another drawback of Transformer-based models is that computational complexity increases quadratically as the input length increases. Because clinical documents are typically long, complexity grows quadratically resulting in a huge computational burden. Therefore, binary classification formulation with a model that has separate architectures for text input and labels input, such as a Siamese network (Koch et al., 2015) [8], is more efficient when dealing with lengthy clinical documents.

6.2.3 Methods

In this study, we propose two approaches to address the performance difference issue. As the first approach, we propose a novel weighted loss that uses prior knowledge of class distribution. In the second approach, we reformulate a multi-label classification task as a binary classification task and propose a novel model architecture. Further, we propose a data augmentation method that uses an external knowledge base for the binary classification model.

[8] A typical Siamese Neural Network contains two identical sub-networks that share parameters and weights. A Siamese Neural Network produces two output vectors based on two different inputs. And the final layer of the network compares the two output vectors. Generally, one of the output vectors is precomputed, and forming a baseline against which the other output vector is compared.

Distribution-Aware Weighted Loss

We propose a distribution-aware weighted loss (DAWL), designed to address the data imbalance of the multi-label datasets for group-specific training. For this approach, we extend the traditional weighted binary loss function to a multi-label setting. The proposed a distribution-aware weighted loss function:

$$L_g = -w_g^k L_{g+} - L_{g-} \tag{6.13}$$

where $k \in \{1, .., K\}$ is a label class, $g \in \{1, .., G\}$ is a demographic group class, and w_g^k is the label-specific weight value that we calculate per group. In a multi-label setting with a highly imbalanced dataset, defining the weight value based on the number of minority samples over the number of majority samples could result in overwhelmingly large losses from extremely rare samples. To avoid this, we define a weight $w_g^k \in [1, 2]$ as follow:

$$w_g^k = 1 + (1 - N_g^{l_k=\text{pos}}/N_g) \tag{6.14}$$

where N_g is the size of the group-specific data set D_g and $N_g^{l_k=\text{pos}}$ is the number of samples that contains positive label for l_k within a group data set D_g, where the entire dataset D consists of group-specific subsets $D = \{D_1, ..., D_G\}$.

As Binary Classification Approach

Motivated by the previous works (Koch et al., 2015; Sun et al., 2019a; Halder et al., 2020; Shim et al., 2021), we reformulate the medical code prediction task, which is multi-label classification, as binary classification, which we refer it "As Binary Classification (ABC)" approach. The main idea of the ABC approach is providing a label name (i.e., in our case, disease name) to a model as an input. The proposed method is similar to the previous label attention model (Vu et al., 2021) that consists of $|K|$ different binary classification layers, which is a one-vs-all approach (Rifkin and Klautau, 2004). However, the main difference is that the proposed method uses a unified classification layer and leverages the label information which is given as an input.

Figure 6.6 illustrates the proposed ABC model. There are four main parts to the proposed ABC model: a document encoder, a label encoder, an attention module, and a classification layer. The document and label encoders produce document and label representations based on the given document embeddings and label embeddings, respectively. Then the attention module takes both

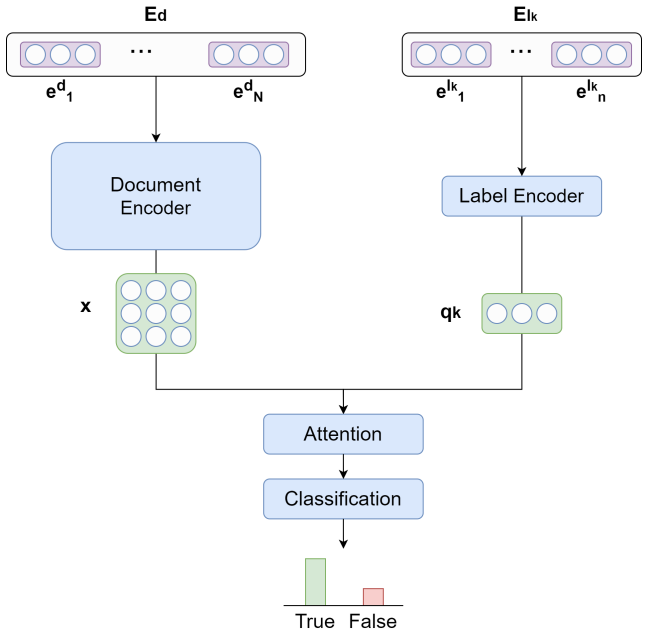


Figure 6.6: Illustration of "as binary classification (ABC)" approach.

representations to create the final label-specific document representations that are used for final classification. The final classification result is binary, either True (i.e., the given label has been assigned to the given document) or False (i.e., the given label has not been assigned to the given document). In the following parts, we describe each module in detail.

Embedding Layer The embedding layer takes a sequence of words from a clinical document $\boldsymbol{w^d} = [w_1^d, w_2^d, ..., w_N^d]$ and a sequence of words from a label name $\boldsymbol{w^{l_k}} = [w_1^{l_k}, w_2^{l_k}, ..., w_n^{l_k}]$, where N and n denote the length of the clinical document and the label name of l_k[9], respectively. A pre-trained embedding layer, such as word2vec (Mikolov et al., 2013), maps each word into vector space resulting in a sequence of vector representations $\boldsymbol{E_d} = [\boldsymbol{e_1^d}, ..., \boldsymbol{e_N^d}] \in \mathbb{R}^{N \times d^e}$ and $\boldsymbol{E_{l_k}} = [\boldsymbol{e_1^{l_k}}, ..., \boldsymbol{e_n^{l_k}}] \in \mathbb{R}^{n \times d^e}$ where d^e is the dimension of each word embedding.

[9]We tested both labels name consists of a few words and label descriptions with 1-2 sentences and better results were observed at a model with label names. Therefore we use label names for label representation in this study.

Document Encoder The document encoder projects the document embeddings $\boldsymbol{E_d}$ into a document representation $\boldsymbol{x}$:

$$\boldsymbol{x} = f_\theta(\boldsymbol{E_d}) \tag{6.15}$$

where f_θ is a document encoder model. We use the MultiResCNN model proposed by Li and Yu (2020) as the document encoder. MultiResCNN consists of convolutional layers with m different filters. The document encoder outputs $\boldsymbol{x} \in \mathbb{R}^{N \times (m \times d^p)}$ where d^p indicates the out-channel size of a convolutional filter. More details of model architecture can be found in the appendix (Appendix 6.A) or the original paper (Li and Yu, 2020).

Label Encoder The label encoder maps the label embeddings $\boldsymbol{E_{l_k}}$ into a label representation $\boldsymbol{q}_k$:

$$\boldsymbol{q}_k = g_\theta(\boldsymbol{E_l}) \tag{6.16}$$

where g_θ is a single-layer feed-forward neural network[10]. The label encoder projects a series of input embeddings into a single vector representation $\boldsymbol{q}_k \in \mathbb{R}^{d^e}$.

Attention layer We use the document and label representations to compute label-specific document representation by using the attention mechanism (Bahdanau et al., 2015; Luong et al., 2015). We take the linear transformation of the document representation $\boldsymbol{x}$ to create a latent vector. The projected latent vector and the label representation $\boldsymbol{q}_k$ are used to calculate attention scores $\boldsymbol{a} \in \mathbb{R}^N$. Formally:

$$\boldsymbol{a} = \text{softmax}(\boldsymbol{q}_k \tanh(\boldsymbol{W_h}\boldsymbol{x}^T + \boldsymbol{b_h})) \tag{6.17}$$

$$\boldsymbol{v} = \boldsymbol{a}\boldsymbol{x} \tag{6.18}$$

where $\boldsymbol{W_h} \in \mathbb{R}^{d^e \times (m \times d^p)}$ and $\boldsymbol{b_h} \in \mathbb{R}^{d^e \times N}$ are trainable parameters. The computed vector $\boldsymbol{v} \in \mathbb{R}^{(m \times d^p)}$ is the final label-specific document representation.

[10]We also tested the same architecture of the document encoder but the result with a one-layer network was better.

Classification layer Finally, the label-specific document representation is fed into a classification layer to produce a label-specific prediction:

$$z = \boldsymbol{W_c} \boldsymbol{v}^T + b_c \tag{6.19}$$

$$\hat{y}_d^k = \sigma(z) \tag{6.20}$$

where $\boldsymbol{W_c} \in \mathbb{R}^{(m \times d^p)}$ and b_c are trainable parameters and $\sigma(\cdot)$ is a sigmoid function. The training objective is to minimise the binary cross entropy loss between the prediction $\hat{y}_d^k$ and the ground truth y_d^k.

Data Augmentation

Motivated by the previous work (Shim et al., 2021), we propose a data augmentation technique that replace a label name in a input with its synonyms by using an external knowledge base. The proposed binary classification model ABC_θ takes document $d \in D_g$ and label name as an input l_k and makes a prediction $\hat{y}_d^k = \text{ABC}_\theta(d, l_k)$. To augment the training data, we extract a set of synonyms $\bar{l}_{k,s}$ of each label l_k, where $s \in \{1, ..., s_k\}$. Then we replace the label name l_k in the document-label pair (d, l_k) data with its synonyms $\bar{l}_{k,s}$ to create augmented dataset $\{(d, \bar{l}_{k,1}), ..., (d, \bar{l}_{k,s})\}$.

6.2.4 Experiments

Data

Following the previous studies (Mullenbach et al., 2018; Li and Yu, 2020; Shim et al., 2022), pre-processing steps are as follows: firstly, the discharge summary texts were tokenised and lowercase. Then, tokens that contain no alphabetic characters were removed. All documents are truncated to a maximum length of 2500 tokens. We extract demographic information: age values are computed based on the date of birth data and the admission time data. We follow the same data pre-processing steps of the previous study (Ch. 6.1) and more details on data and pre-processing can be found in Ch. 6.1.2.

Table 6.6 summarises the size of data sets used in the experiments. Following the previous study (Ch 6.1), we use the `MIMIC-III 50` dataset (Johnson et al., 2016) which contains 11,366 discharge summaries with 50 unique ICD codes. Specifically, we focus on building a medical coding system that performs

Table 6.6: Sample size of the train, validation, test set (top) and the size of age group-specific training sets (bottom).

	Train	**Valid**	**Test**	**Total**
Entire	7626	1571	1729	10966
Subset				
D_{18-29}	300	46	54	400
D_{30-49}	1148	209	228	1585
D_{50-69}	2931	634	693	4258
D_{70-89}	2817	593	642	4052
D_{90+}	430	89	112	631

equally well across different age groups. For this, we create subsets of data by splitting the entire training data based on the age of data subjects, where $D_{\text{subsets}} = \{D_{\text{18-29}}, D_{\text{30-49}}, D_{\text{50-69}}, D_{\text{70-89}}, D_{\text{90+}}\}$.

Settings

In the experiments, we compare a baseline model, ensemble models with different loss functions, and a binary approach without and with data augmentation. Following the previous study (Ch. 6.1), we use a state-of-the-art medical code prediction model proposed by Li and Yu (2020) as a baseline architecture. For a baseline model (`Baseline`), we train a single model with the entire training dataset.

For ensemble models, we create group-specific models by continuing fine-tuning the trained baseline model (`Baseline`) on subsets of group-specific training data. A baseline ensemble model (`Ensemble-Baseline`) is trained by using a standard binary classification loss. We compare the baseline ensemble model and other ensemble model with different loss functions, including the proposed distribution-aware weighted loss (`Ensemble-DAWL`), focal loss (Lin et al., 2017) (`Ensemble-FL`), and asymmetric loss (Ridnik et al., 2021) (`Ensemble-ASL`).

Finally, we train a unified model by formulating a problem as binary classification (`ABC`). Additionally, we train ABC model with augmented data (`ABC-aug`) by utilising an external medical knowledge database (UMLS (Bodenreider, 2004)). For data augmentation, we extract synonyms of each ICD code label (i.e., disease name) from the database and create augmented training samples by pairing an input text with the extracted synonyms. We create two different augmented data sets: firstly, we augment all data from the original dataset. Secondly, we only augment data with infrequent label classes. Infrequent label classes are defined as the label classes that are observed less than the median

Table 6.7: Hyperparameter settings.

Hyperparameter	**Assignment**
Kernel sizes	{3, 5, 9, 15, 19, 25}
Focal loss γ	1
Asymmetric loss γ_+	0
Asymmetric loss γ_-	2
max training epoch (baseline)	200
max training epoch (ensemble, as binary classification)	20
patience p	10
batch size	150
learning rate	$1e-4$
dropout rate	0.2
optimizer	Adam (Kingma and Ba, 2015)

frequency of entire labels in the training set. Then we compare a model trained with a training set containing original data and augmented data from all labels (`ABC-aug-all`), and a model trained with a training set containing original data and augmented data of infrequent labels (`ACB-aug-inf`).

Table 6.7 summarises hyperparameter settings for the experiments. During the training epoch, the validation sets are used to compute loss at every epoch for monitoring the training progress. When there is no improvement for $p = 10$ times, training is early-stopped. The hyperparameters are decided based on the previous studies (Li and Yu, 2020; Lin et al., 2017; Ridnik et al., 2021) and non-exhaustive experiments.

Evaluation metrics

As global performance metrics, we use the F1 scores, including micro-, macro-, and sample-averaged F1 scores following the previous works (Shi et al., 2017; Mullenbach et al., 2018; Li and Yu, 2020; Shim et al., 2022). Micro-averaged scores are calculated by treating each <text input, code label> pair as a separate prediction. Macro-averaged scores are calculated by averaging metrics computed per label. Sample-averaged scores are calculated by computing scores at the instance level and averaging over all instances in the data set. For recall, the global metrics are computed as follows:

$$\text{Micro-R} = \frac{\sum_{k=1}^{K} \text{TP}_k}{\sum_{k=1}^{K} \text{TP}_k + \text{FN}_k} \tag{6.21}$$

$$\text{Macro-R} = \frac{1}{|K|} \sum_{k=1}^{K} \frac{\text{TP}_k}{\text{TP}_k + \text{FN}_k} \tag{6.22}$$

$$\text{Sample-R} = \frac{1}{|N|} \sum_{i=1}^{N} \frac{|\boldsymbol{y}_i \cap \hat{\boldsymbol{y}}_i|}{|\boldsymbol{y}_i|} \tag{6.23}$$

where TP_k and FN_k, denote true positive examples and false negative examples for a specific ICD-9 code label k, respectively. Since we use `MIMIC-III 50` dataset, $|K|$ equals 50. $\boldsymbol{y}_i$ and $\hat{\boldsymbol{y}}_i$ denote the ground truth labels and the predicted labels for the i-th test example, respectively and N denotes the total number of test samples. Precision is computed similarly.

To evaluate group-specific performances, group-averaged sample F1 scores are used:

$$\text{Group-F1} = \frac{1}{|G|} \sum_{g}^{G} \text{Sample-F1}_g \tag{6.24}$$

where Sample-F1_g is computed by averaging F1 scores computed per sample in a age group $g \in G = \{G_{\text{18-29}}, G_{\text{30-49}}, G_{\text{50-69}}, G_{\text{70-89}}, G_{\text{90+}}\}$.

We also consider fairness metrics, such as false negative rate (FNR) and false positive rate (FPR) in the sample level (Hardt et al., 2016; Chouldechova, 2017; Shim et al., 2022). FNR is the fraction of ICD codes that failed to be predicted by a system but are included in a ground truth label set. FPR is the fraction of ICD codes that are erroneously predicted by a system but not included in a ground truth label set. High FNR scores imply low recall scores and high FPR implies a high probability of false alarms. Two metrics are computed as follows:

$$\text{FNR} = \frac{1}{|N|} \sum_{i=1}^{N} \frac{\text{FN}_i}{\text{FN}_i + \text{TP}_i} \tag{6.25}$$

$$\text{FPR} = \frac{1}{|N|} \sum_{i=1}^{N} \frac{\text{FP}_i}{\text{FP}_i + \text{TN}_i} \tag{6.26}$$

Table 6.8: Experimental results on the entire test set.

Model	Micro-F1	Macro-F1	Sample-F1
Baseline	64.4	59.2	60.7
Ensemble-Baseline	64.4	58.5	60.6
Ensemble-DAWL	**64.7**	59.5	**60.8**
Ensemble-FL	64.1	58.	60.3
Ensemble-ASL	64.6	**60.1**	60.8
ABC	62.5	56.5	59.2
ABC-aug-all	55.3	49.5	51.3
ABC-aug-inf	57.9	52.7	54.1

More details on how to compute evaluation metrics and error metrics can be found in Ch. 6.1.3.

6.2.5 Results

Global performances

Table 6.8 summarises the experiment results. It is observed that training group-specific models (`Ensemble-Baseline`) does not improve the global performance metrics. The ensemble model trained by using the proposed loss (`Ensemble-DAWL`) slightly outperforms the baseline models (`Baseline`, `Ensemble-baseline`) but the differences are not significant. The ABC model (`ABC`) achieve slightly lower performances across all globally averaged F1 scores than the baseline model (`Baseline`). The ABC model trained with the augmented data of infrequent labels (`ABC-aug-inf`) achieves slightly better performances than the ABC model with the entire augmented data (`ABC-aug-all`). However, both models achieve lower performances than the ABC model (`ABC`) trained without augmented data. These results imply that the proposed data augmentation method harms performance.

Group-specific performances

Group-specific performances are summarised in Table 6.9. Contrary to the previous results with the global performance metrics, it is observed that `Ensemble-baseline` slightly outperforms the `Baseline` in terms of group-averaged F1 scores and fairness scores. `Ensemble-DAWL` outperforms `Baseline` and other ensemble models in terms of group-averaged F1 scores but shows lower fairness metrics. It is worth mentioning that ABC models achieve

Table 6.9: Performances per age group. Diff. and Ratio refer to largest difference (the lower the better) and smallest ratio (the higher the better), respectively. Sample-averaged F1 scores are used for group-averaged scores.

		Ensemble					ACB-aug	
Age	Base	Base	DAWL	FL	ASL	ABC	all	inf
18-29	48.2	48.9	47.4	48.7	47.7	**62.1**	47.2	53.5
30-49	52.4	52.6	53.4	52.7	52.1	**60.3**	51.9	56.
50-69	60.9	60.8	**62.0**	60.4	61.3	59.2	51.9	54.2
70-89	63.6	63.4	**63.7**	63.4	63.4	59.2	51.6	53.6
90+	65.0	65.1	**65.4**	65.2	65.0	56.4	46.1	52.6
Average	58.0	58.2	58.4	58.1	57.9	**59.4**	49.7	54.
Diff. (↓)	16.8	16.2	18.	16.5	17.3	5.7	5.8	**3.4**
Ratio (↑)	74.2	75.1	72.5	74.7	73.4	88.8	86.9	**93.9**

similar performances across all age groups resulting in better fairness metrics compared to multi-label classification models. Similar to the previous results, data augmentation harms the performance of the ABC model. The ABC model trained with the augmented data of infrequent labels (`ABC-aug-inf`) achieve a lower group-averaged F1 score than the ABC model without data augmentation (`ABC`).

Error analysis

Table 6.10 summarises the error analysis results. In general, all models show higher FNR than FPR which means that the probability that mentioned ICD codes will be missed by the systems is high (low recall). One interesting observation is that all ensemble models and ABC models achieve lower FNR scores but higher FPR scores than the baseline model (`Baseline`). Especially `Ensemble-DAWL` and `ABC` improve FNR scores considerably while degrading FPR scores marginally. The ABC model trained with the augmented data of infrequent labels achieves higher FNR scores than the ABC model without data augmentation (`ABC`).

6.2.6 Discussion

Group-specific ensemble approach In this study, we propose two approaches to build a fair model that performs equally well across different demographic groups. In the first approach, we create an ensemble of group-specific models. Even though the experimental results show that ensemble approaches improve

Table 6.10: Errors analysis results. Because of space limitation, only a subset of results is reported. More results can be found at Table 6.11 in Appendix 6.B.

	Baseline		Ens-Base		Ens-DAWL		ABC		ABC-aug-inf	
	FNR	FPR	FNR	FPR	FNR	FPR	FNR	FPR	FNR	FPR
Total	40.6	**3.8**	39.4	4.0	**34.**	5.5	36.1	5.8	42.7	5.7
18-29	46.2	2.9	45.6	2.8	44.4	3.5	33.2	4.8	41.5	6.4
30-49	45.9	3.3	45.8	3.2	40.5	4.3	34.3	6.3	39.9	6.1
50-69	39.5	3.9	40.1	3.7	34.	5.3	37.1	5.8	43.	5.6
70-89	35.7	5.0	36.6	4.8	31.2	6.4	35.8	5.6	43.6	5.4
90+	34.1	4.4	34.1	4.3	31.7	5.	37.2	6.4	43.	6.1

the performances in terms of global F1 scores, the improvements are marginal. Similarly, ensemble models `Ensemble-baseline` and `Ensemble-FL` improve fairness scores, which means the models achieve relatively equal performances across all groups, except for the ensemble approach with a data distribution-aware weighted loss and asymmetric loss. Especially, `Ensemble-DAWL` achieves higher group-averaged F1 scores than the baseline approaches but shows worse fairness scores by increasing performance differences between the groups. Overall, ensemble approaches provide marginal performance improvement but they require more resources, including more computation for fine-tuning and memory for group-specific models. Therefore, building an ensemble model could be beneficial when there is enough computational resource and high performances (e.g., high recall) are desirable.

As binary classification approach From the experiment results, it is observed that the ABC models slightly underperform compared to the multi-label classification models in terms of globally averaged F1 scores. One interesting observation is, however, that the ABC approach substantially improves the fairness metrics by achieving relatively equal performances across age groups. Similarly, error analysis results show that multi-label classification approaches show low FNR scores and high FPR toward old patient groups (70-89, 90+) which implies that the models predict many false positives for the old patient groups (70-89, 90+). This is potentially caused by label co-occurrence (Zhang and Zhou, 2013) because older patients groups contain more labels in the training set (Figure 6.3 in Ch 6.1.4). Meanwhile, ABC approaches do not show this behaviour because this label co-occurrence information is lost when formulating a multi-label problem as binary classification. This information loss results in lower performance in terms of global performance metrics. Some researchers show that incorporating label co-occurrence information from the training data can improve the performances (Huang and Zhou, 2012; Yu et al.,

2014; Zhu et al., 2017). However, it is not a trivial question whether we should train a model to learn label co-occurrence or not for a clinical application. On one hand, label co-occurrence captures information about comorbidity that can be useful for a medical coding task. On the other hand, it can result in a biased model that fitted to the training data set but learned how to perform a target task. Therefore we argue that more careful consideration is required when building a clinical application and it is important that a user (human coder) is aware of the behaviour of a system, such as when and where a system fails.

Limitations and future directions In this study, we aim to address the performance differences between demographic groups. In general, ensemble approaches marginally improve global and group-averaged performances. However, the ensemble model with the proposed distribution-aware weighted loss results in lower fairness scores. On the other hand, binary classification formulation degenerates performance scores but substantially improves group-averaged scores and fairness scores.

Experimental results indicate that the proposed data augmentation method harms performance. This result is counter-intuitive because similar data augmentation techniques have proven to be useful with transformer-based models (Halder et al., 2020; Shim et al., 2021). Potential reasons are that the baseline architecture used in the experiments is CNN-based or the augmented data, which are synonyms of disease names, are not diverse enough to help a model generalise better. Further study is required, which we leave for future works.

6.2.7 Conclusion

In this study, we investigate how to address the problem of performance differences across demographic groups. For this, we propose two approaches including an ensemble model utilising the prior knowledge of data distributions for a novel weighted loss function and formulating the problem as binary classification. Results demonstrates that the ensemble approach with the proposed loss function can improve global performance. It is observed that the binary classification approach can improve group-averaged scores and fairness scores by performing equally well across different age groups. Potential future research direction includes a further study on data augmentation for performance improvement and an investigation of building a robust and fair model for a clinical application.

Appendix

6.A Multi-Filter Residual Convolutional Neural Network

Multi-Filter Residual Convolutional Neural Network (MultiResCNN) (Li and Yu, 2020) for a medical code prediction task. MultiResCNN is built based on TextCNN (Kim, 2014), Residual Network (He et al., 2016) and CAML (Mullenbach et al., 2018). The key idea of MultiResCNN is to use multiple CNN filters with varying window lengths to capture various text patterns with different lengths. The model also uses residual convolutional layers to enlarge the receptive field.

Figure 6.7 illustrates the architecture of the MultiResCNN. Input of MultiResCNN is a sequence word embeddings $\boldsymbol{E} = [\boldsymbol{e}_1, ..., \boldsymbol{e}_N]$, where N is the length of input. MultiResCNN consists of m different convolutional filters $f_1, ..., f_m$ with different kernel sizes $k_1, ..., k_m$ to capture text patterns with different lengths. Each convolutional filter sets padding and stride as floor$(k_i/2)$ and 1, respectively, to make the same output dimension $H_i \in \mathbb{R}^{N \times d^f}, i \in \{1, ..., m\}$ when kernel sizes are odd numbers[11]. d^f indicates the out-channel size of a convolutional filter.

On top of each convolutional layer, there are p residual blocks. Each residual block consists of three convolutional filters with residual connections. The first $(r^1_{i,j})$ and the second $(r^2_{i,j})$ convolutional filters in each residual block $(R_{i,j}, j \in \{1, ..., p\})$ have same kernel size k_i with the corresponding convolutional filter f_i in the multi-filter convolutional layer. The kernel size of the third convolutional filter $(r^3_{i,j})$ in each residual block is 1. Each p-th residual block outputs $H_{i,p} \in \mathbb{R}^{N \times d^p}$. The final output $H \in \mathbb{R}^{N \times (m \times d^p)}$ is a concatenation of the outputs of m residual blocks $H = [H_{1,p}; ...; H_{m,p}]$.

6.B Error analysis results

In Ch 6.2.5, we analyse errors to understand model behaviour. We use false negative rate (FNR) and false positive rate (FPR) as error metrics. All error analysis results can be found in Table 6.11.

[11] Since we employ 1-dimension convolutional filter, the output size of the filter w_{out} is computed as $w_{\text{out}} = (w_{\text{in}} - k + 2p)/s + 1$ where k,p,s indicate kernel size, padding size, and stride size, respectively.

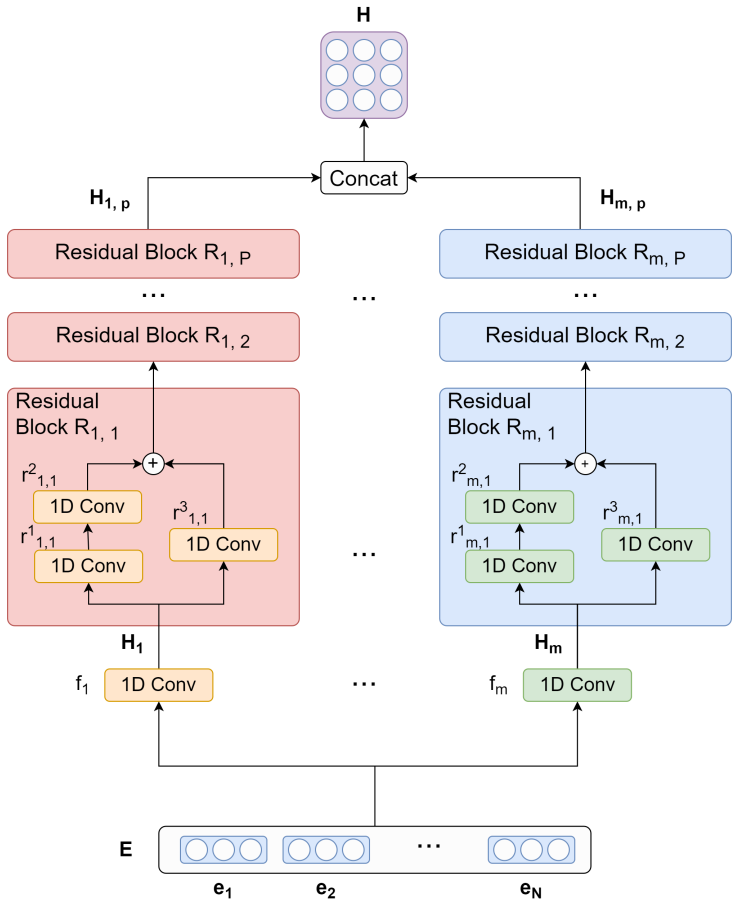


Figure 6.7: The architecture of the Multi-Filter Residual Convolutional Neural Network (MultiResCNN) model was used in this study.

Table 6.11: Entire error analysis results.

	Baseline		Ens-Base		Ens-DAWL		Ens-FL		Ens-ASL		ABC		ABC-aug-all		ABC-aug-inf	
	FNR	FPR	FNR	FPR	FNR	FPR	FNR	FPR	FNR	FPR	FNR	FPR	FNR	FPR	FNR	FPR
Total	40.6	3.8	39.4	4.0	34.	5.5	39.7	4.	35.7	5.1	36.1	5.8	42.7	7.4	42.7	5.7
18-29	46.2	2.9	45.6	2.8	44.4	3.5	45.6	2.8	45.6	3.	33.2	4.8	46.7	7.1	41.5	6.4
30-49	45.9	3.3	45.8	3.2	40.5	4.3	45.8	3.2	44.8	3.6	34.3	6.3	41.5	7.7	39.9	6.1
50-69	39.5	3.9	40.1	3.7	34.	5.3	40.8	3.6	35.6	5.	37.1	5.8	42.6	7.4	43.	5.6
70-89	35.7	5.0	36.6	4.8	31.2	6.4	37.	4.6	32.1	6.1	35.8	5.6	42.3	7.3	43.6	5.4
90+	34.1	4.4	34.1	4.3	31.7	5.	34.1	4.3	34.1	4.4	37.2	6.4	46.6	8.3	43.	6.1

Chapter 7

Conclusion

In this thesis, we investigated the hypothesis that deep neural networks with data-efficient algorithms outperform their counterparts in data- and label-scarce settings. To this end, we proposed multiple data-efficient methods for different NLP tasks and validated their effectiveness. This chapter recapitulates how the proposed methods addressed the research questions outlined at the beginning and discuss their contributions and limitations (Ch 7.1), provides an outlook for future research (Ch 7.2), and multidisciplinary challenges of applying NLP technologies to healthcare (Ch 7.3). The final section introduces the valorisation opportunities of the research discussed in this dissertation (Ch 7.4).

7.1 Revisiting the research questions

RQ1. How can we fine-tune a neural NLP model when only a small-sized training set for the target task is available?

A large-scale annotated training dataset is required to fine-tune a pre-trained language model for a downstream task. Since a large pre-trained language model consists of hundreds of millions of parameters, it is challenging to fine-tune with a small amount of training data. Therefore, we investigated how to fine-tune a neural NLP model when only a small-sized training set is available for the target task. Our contributions are summarised below:

Increasing the Size of Dataset

To address this question, we mainly focused on maximising the utility of existing training data and proposed methods to increase the size of a labelled dataset. For example, we proposed data augmentation methods by augmenting an input text (Ch 3) and label text (Ch 4). Experimental results show that the proposed methods can significantly improve performance in data-scarce and label-scarce settings (Ch 3, Ch 4).

Similarly, formulating multi-label classification problems as binary classification with label information is helpful because it increases the size of the training data[1] (Ch 4). Also, we found that generating synthetic data can be seen as a data augmentation technique. In Chapter 5, we proposed a rule-based synthetic data generation algorithm for a temporal information extraction task. The proposed method augments the training data by utilising human knowledge of the structure of temporal expressions. Experimental results show that the method can improve the model's performance.

Throughout the thesis, we observed that the effects of data augmentation techniques were especially pronounced when dealing with imbalanced datasets, which contain minority classes with a small amount of data in a training set (Ch 3.6.3). The proposed data augmentation methods over-sample data points by perturbating them, which results in less specific decision regions for minority classes. In other words, the proposed data augmentation methods allow classification models to learn larger, more general regions. Since minority classes tend to have small, specific decision regions, whereas majority classes tend to have large, generalised decision regions, the effect of data augmentation is more significant in minority classes, as illustrated in Figure 7.1. This observation is also aligned with the previous literature on data augmentation by generating synthetic data (Chawla et al., 2002) that is particularly effective for the minority classes.

Providing Explicit Hints

Another approach we proposed to address the small-sized training set problem is to provide explicit cues about a task. For example, in Chapter 4, we modified an input and output configuration by appending label information (e.g., aspect category name) to inputs and formulated an aspect-based sentiment classification

[1]When a multi-label classification problem is formulated as a binary classification, a sentence s_i in the original dataset can be expanded into multiple sentence-label pairs (s_i, l_1), $\cdots$, (s_i, l_N) with label categories l_n where $n \in \{1, 2, .., N\}$.

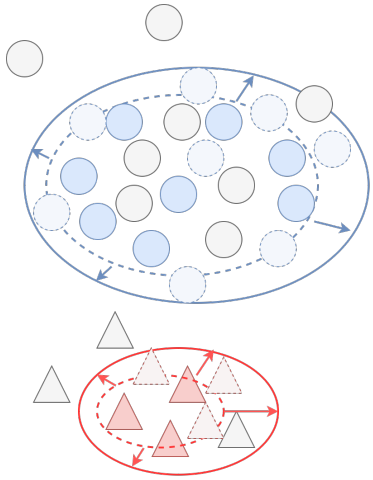


Figure 7.1: The effect of data augmentation. Circles and triangles with solid and dashed lines represent labelled and augmented data, respectively. Grey-coloured circles and triangles with solid lines represent unlabelled data. Solid ovals indicate updated decision regions and dashed ovals indicate original decision regions.

task as a sentence-pair classification problem (i.e., <input, aspect category> -> sentiment class) and results showed substantial performance improvement.

There are a couple of possible interpretations of why this approach improves performances: firstly, this input and output configuration changes a problem space from multi-label classification to binary classification lessening a burden for a classification layer. Secondly, the class name appended to an input guides a model where to focus for a target task. Because of this reason, this approach works well with a model that uses an attention mechanism (Bahdanau et al., 2015), such as BERT (Devlin et al., 2019). Moreover, using a second part of the input to perform a target task is similar to the sentence-pair classification, which is one of the pre-training tasks of BERT.

Adding Auxiliary Tasks

Lastly, we proposed multi-task learning that allows a model to jointly learn a target task and an auxiliary task related to the target task (Ch 5). Experimental results indicate that multi-task learning can improve the model's performance by utilising additional training signals with the existing training data (Ch 5).

There are a few plausible explanations for why the proposed multi-task learning improves performance. Firstly, introducing auxiliary tasks increases the training signals for training a model. Therefore, this can be seen as a type of data augmentation (Ruder, 2017). Secondly, multi-task learning helps a model generalise better by reducing the risk of overfitting (Tu et al., 2020). Since the model learns multiple tasks simultaneously while sharing the hidden layers, the model learns to find generalised representations, rather than be overfitted to one task. Regarding this, Baxter (1997) already showed that an increase in the simultaneous learning tasks decreases the risk of overfitting. In our case, we introduced an auxiliary task related to the target task so that it not only provides additional learning signals to a model but also helps the model generalise better for a target task. Thirdly, multi-task learning helps a model eliminate irrelevant features and learn discriminative features for individual tasks (Bi et al., 2008). Especially when dealing with high-dimensional data, it is easy for a model focuses on spurious features. Multi-task learning teaches a model to differentiate between relevant and irrelevant features for each task by providing additional training signals.

Limitations

Throughout this dissertation, we have demonstrated the effectiveness of the proposed data-efficient methods to address the first research question (**RQ1**) on the data scarcity problem. To improve on the proposed methods, several aspects can be further studied.

Firstly, the effect of the proposed data augmentation methods is marginal when training data are sufficiently large. For example, for the sleep issue classification task in Chapter 3, the performance gain is less than 1% when the entire dataset is used for training. Similarly, for the aspect-based sentiment analysis task in Chapter 4, the proposed label augmentation method contributes to a negligible performance improvement when a full dataset is used for training. This is potentially caused by the proposed methods augmenting the data by slightly perturbating labelled samples to get the neighbouring unlabelled samples. Even though these approaches expand decision regions by synthesising similar samples but the coverage is limited. Therefore, a model cannot handle well new samples from unknown distribution.

Secondly, the label augmentation method might not be applicable to some use cases or other model architectures. In Chapter 4, we showcased the effectiveness of the label augmentation method by validating it on a custom dataset containing user reviews of a sleep coaching programme and a benchmark dataset containing user reviews of restaurants. Following the success of providing class names as

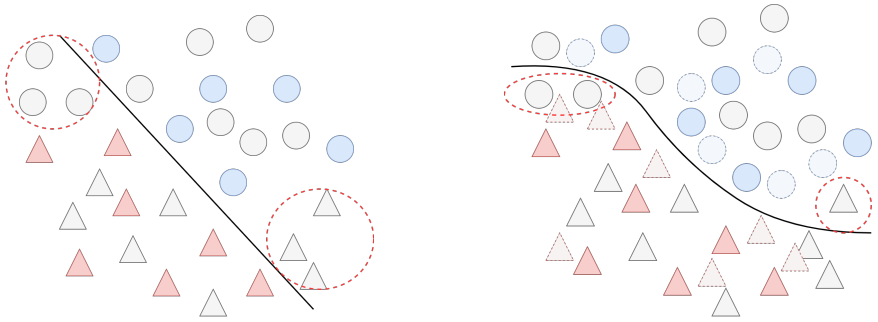
inputs with a BERT-based model (Ch 4), we applied a similar approach to a CNN-based model and observed that the proposed method is less effective (Ch 6.2). It could be because of different model architectures (i.e., the proposed CNN-based model consists of two separate pipelines combined in a later stage by using an attention mechanism.[2]) or relevancy to pre-training task. For example, the proposed label augmentation method is similar to the next sentence prediction task, which is one of the pre-training tasks of BERT, in terms of forcing a model to learn features based on two separate inputs. Another possible explanation is because of different domain data (i.e., user-generated texts vs. clinical texts).

Lastly, the multi-task learning approach and synthetic data generation method proposed in Chapter 5 require additional human effort. For example, the proposed multi-task learning that trains a model on a target task and an auxiliary task related to the target task can be beneficial because it can utilise additional training signals from the same amount of data. Even though this approach can mitigate a data scarcity issue, it requires more annotations, which are not available in a label-scarce setting. Furthermore, the proposed synthetic data generation method is based on handcrafted regular expressions designed by a human programmer who has knowledge (i.e., on the structure of temporal expressions) related to a target problem (i.e., temporal information extraction). Therefore, it requires human expertise that cannot be automatically transferred when dealing with a new use case (e.g., temporal information extraction in another language). Moreover, the proposed method uses only one auxiliary task that is similar to a target task. Therefore, the auxiliary task could provide limited complimentary information.

RQ2. How can we train a machine learning model when only a small subset of the target dataset is labelled?

Training data should be labelled for supervised learning; however, manually annotating textual data is costly and not scalable. Therefore, the second research question asked how to solve the label-scarcity issue. This is a critical issue when domain experts need to annotate data (e.g., in the clinical domain) or build an application with a labelling scheme that might be changed during the development process (e.g., a new label class needs to be added). Our contributions are summarised below:

[2]Firstly, the label names are used to create label representations and documents are used to create document representations, separately. Then the label representations and document representations are used to compute label-specific document representations.



(a) Original decision boundary.

(b) Updated decision boundary after retraining with augmented data.

Figure 7.2: Illustrations of retraining a model with augmented data. Circles and triangles with solid and dashed lines represent labelled and augmented data, respectively. Grey-coloured circles and triangles with solid lines represent unlabelled data. Solid lines indicate decision boundaries. Areas with dashed lines indicate misclassification.

Semi-Supervised Learning

In this dissertation, we mainly investigated how to reduce manual labelling efforts required for supervised learning while minimally compromising performance. Firstly, we studied how to leverage unlabelled data without additional manual labelling. For this, we used the model's predictions on unlabelled data (i.e., pseudo labels) as additional training signals via semi-supervised learning. From experiments, we found that the semi-supervised learning approach is particularly useful when combined with the data augmentation method to deal with imbalanced data (Ch 3).

From the experiments, we observed essential requirements for the proposed semi-supervised learning method. Firstly, the effect of semi-supervised learning relies on the performance of an initial model. This is because the proposed method depends on a pre-existing model's predictions (referred to as pseudo-labels) on an unlabelled dataset, and retrains a model with the pseudo-labels. Secondly, the proposed semi-supervised learning method that uses pseudo-labels works well with the data augmentation method. This is because the proposed data augmentation method provides regularisation effects. For example, the data augmentation method creates synthetic data by perturbating samples. As illustrated in Figure 7.2, retraining a model on these synthetic data encourages a model to have consistent predictions on neighbouring samples and provides the effect of regularisation.

Active Learning

We then focused on an active learning scenario to efficiently reduce manual labelling in Chapter 4. For this, we proposed a label-efficient training scheme to maximise the utility of unlabelled data and already labelled data. The proposed method employs task-specific pre-training, which is self-supervised learning, and label augmentation, which is data augmentation, in an active learning framework. Experimental results show that the proposed method can use only half of the labelled data to achieve comparable performance to a model without active learning. Moreover, the proposed method outperforms other active learning methods. Moreover, a model trained with the proposed method generalises better than models without the proposed method by achieving higher performance scores with the same amount of data at the beginning and the end of active learning iterations due to task-specific pre-training and label augmentation.

From the experiments, we found two critical points in the active learning scenarios: the first is an initial model and the second is uncertainty thresholds. Since the proposed method is uncertainty-based active learning, an initial model selects samples near decision boundaries that are most likely moving the decision boundaries. However, we observed that the initial model is over-confident about its wrong predictions towards minority classes and rarely selects minority class samples and it results in no performance improvements in terms of minority classes. As we showed in Figure 4.5 in Chapter 4, this is because the initial model is not fully trained to select minority class samples. Because of this reason, we proposed separate uncertainty scores for minority classes to lower uncertainty thresholds so that a model can select minority class samples. With this strategy, we prioritised the chance of increasing recall for minority, which is critical when dealing with an imbalanced dataset.

Limitations

Based on the research in this dissertation to address the second research question (**RQ2**) on label scarcity, we identified a few points that can be further addressed.

The first limitation is unstable semi-supervised learning in a data-scarce setting. For the study on semi-supervised learning with pseudo-labels for sleep issue classification in Chapter 3, we found that the pseudo-labelling approach might not work well with a small dataset. For example, experimental results show that when only 10% of the training dataset is used, the model is not sufficiently trained to select samples of minority classes with high confidence scores. In other words, the model is overconfident about its wrong predictions (i.e., false

negatives) and fails at detecting samples belonging to minority labels. Since the pseudo-labelling approach utilises the trained model's predictions to train the next model, the error could propagate. Our study in Chapter 3 shows that the initial model achieves low performance concerning the minority classes, and the pseudo-labelling method fails at selecting data with minority labels. As a result, no additional training data with minority label classes were added to the training set, resulting in negligible performance improvement (2%) (Ch 3.6.3).

The second limitation concerns self-supervised learning with small-sized unlabelled datasets. For task-specific pre-training in Chapter 4, we used self-supervised learning on unlabelled data. Generally, self-supervised learning is used to pre-train language models by using a massive corpus. The benefit of self-supervised learning is that it can be deployed with unlabelled corpus data. However, we found that the effect of self-supervised learning is limited when the training corpus is relatively small. For example, the experimental results (Ch 4.5.5) show that the impacts of task-specific pre-training and label augmentation are similar. Considering that task-specific pre-training requires additional pre-training steps before fine-tuning, the proposed task-specific pre-training is not computationally efficient when the training corpus is small.

RQ3. Can we exploit other resources (e.g., structured information, prior knowledge, etc) to improve the performance of a model?

Deep neural networks require a large-scale training set. A model might not be fully trained if there is a limited number of training data. Collecting more training data could be the most intuitive solution; however, it is sometimes difficult to collect this data, or the data may not be available. To this end, we hypothesised that utilising other resources, such as knowledge or structured information, could be an alternative to collecting more data.

Utilising Knowledge of Data Structure and Dataset

Firstly, we studied how to utilise human knowledge about the data structure to generate synthetic data for supervised learning. In Chapter 5, we proposed a synthetic data generation algorithm to address the lack of text data for temporal information extraction. The proposed algorithm uses handcrafted regular expressions which are built based on human knowledge, such as common structures of temporal expressions (e.g., HH:MM) to augment training data by generating synthetic data. Experimental results show that using synthetic data for training can improve the performance of a temporal expression normalisation

task, which requires both natural language understanding skills and numeracy skills to translate a temporal expression (e.g., "ten to eleven in the evening") into a normalised format (e.g., 22:50), when there is no large corpus containing temporal expressions.

The proposed method injects a numeracy skill into a pre-trained language model. Pre-trained language models typically lack numeracy because they are trained on a large corpus extracted from books or scrapped from web texts with language modelling objectives. It is not enough for tasks that require numerical reasoning. Moreover, the proposed method improves performance by generating synthetic data. Recent literature showed that language models can obtain numeracy but they also showed that the model's generalisability is limited to the coverage of the given training set (Wallace et al., 2019). Since collecting a dataset that covers all potential numbers is practically impossible, applying data augmentation is a straightforward solution. For example, one recent literature also applied data augmentation techniques to generate synthetic data and showed the effectiveness of data augmentation for numerical tasks (Geva et al., 2020). Therefore, the main contribution of the proposed method is that it leverages human knowledge of the structure of data to generate synthetic data to improve the coverage of a training dataset.

Secondly, we investigated how to exploit knowledge about data or label information, especially for a knowledge-intensive domain. Chapter 6 introduced a clinical use case, such as medical code prediction, and we studied how to utilise domain knowledge. We proposed a novel weighted loss function that uses information about the label distribution of a training dataset. The proposed loss function utilises label distribution to dynamical weight loss. Experimental results show that a model trained using the proposed weighted loss achieves higher performance scores compared to a model trained without the proposed weighted loss.

The proposed weighted loss can improve performances by using different label distributions among demographic groups. From the data analysis study (Ch 6.1), we demonstrated that each demographic group has a different label distribution and found that a model performs poorly in the demographic groups that contain data whose label set is different from the global label distribution. To address this, we proposed ensemble approaches with group-specific weighted losses by using group-specific label distribution. From experiments, we observed that the proposed group-specific weight loss approach can improve group-averaged scores. However, we also observed that the proposed method improves the performances of large demographic groups but harms the performances of other demographic groups. This is potential because the small groups contain too small sample sizes to take an advantage of information from group-specific label distributions.

Thirdly, we proposed a binary classification approach to utilising label information. In Chapter 6, we provided a label name (in our case, a disease name) as input to guide a model to perform binary classification (to determine whether the given disease is mentioned in the given clinical document or not). Additionally, we explored how to utilise domain knowledge. For this, we used the medical knowledge database that contains synonyms of disease names and augmented training data by replacing label names with their synonyms. To do this, we applied the data augmentation method that we proposed for other use cases (Ch. 4). We found that the binary classification approach can achieve better fairness scores and perform equally well across different demographic groups.

The proposed binary classification approach is similar to the label augmentation method that we proposed for the first research question (**RQ1**): it can be seen as providing explicit hints to a model and simplifying a target task as binary classification. However, the effect of using label information as inputs is less pronounced in Chapter 6.2 compared to Chapter 4. This is because the model proposed in Chapter 6.2 applies an attention mechanism after extracting document features and label features separately, whereas the model proposed in Chapter 4 uses multiple attention layers from the beginning to extract deeply contextualised document and label features.

Limitations

We studied opportunities for exploiting other resources to improve supervised learning in a data-scarce setting, and there were some limitations that should be further addressed.

The proposed synthetic data generation algorithm based on human knowledge in Chapter 5 is rule-based. Therefore, it requires human expertise and handcrafting, which cannot be reapplied to another use case. Furthermore, since this approach includes designing rules and training a model, combining rule-based and learning-based methods, it is less efficient than either a purely rule-based approach or a learning-based approach. In other words, to update the model, researchers must change the rules for generating synthetic data and re-train the model by using the newly generated data.

In Chapter 6, we proposed a method to utilise prior knowledge of data, such as label distribution of a training dataset. Since the proposed method uses label distribution for weighting a loss function for supervised learning, the trained model cannot handle data from a different distribution. In other words, the proposed method assumes that test data always comes from the same data distribution of the training dataset. Therefore, the trained model cannot be

used in a setting where data distribution might shift as time changes. Moreover, since prior knowledge of label distribution is required, the proposed method cannot be applied to a new domain where researchers have no prior knowledge of data.

In the same chapter (Ch. 6), we explored using an external database containing domain-specific knowledge. For example, we used a medical knowledge database that contains synonyms of disease names to augment the training data. To this end, we reformulated a problem as binary classification by providing a label name as an input and replaced a label name with its synonyms while augmenting data. The proposed data augmentation method is similar to what we proposed and validated for other use cases in Chapter 4. However, experimental results indicate the proposed data augmentation is not effective for the clinical use case (Ch. 6.2). As we mentioned earlier, this could be because of the different model architectures. In Chapter 6.2, we use a model consisting of separate feature extraction pipelines with late, shallow attention, which is different from the early, deep attention used in Chapter 4. Another possible reason is the limited synonym diversity of the used external database. These results imply that directly applying data augmentation methods that are validated for other use cases does not always work.

7.2 Future Directions

7.2.1 Addressing Data Scarcity Problem

There are a few potential approaches to mitigating the limitations of the proposed methods for data scarcity problem. Firstly, to address the limitation of the proposed data augmentation methods, which augments texts at a token/word level, researchers can increase the diversity of augmented texts by augmenting texts at a sentence level. One potential way to do so is to utilise separate machine translation models or pre-trained language models to generate synthetic data. For example, back translation (Sennrich et al., 2016) is a method that can introduce paraphrased sentences or different sentence structures while keeping the original meaning[3]. Moreover, recent studies have proposed prompt engineering (Liu et al., 2021b) that uses large generative models (e.g., GPT-3 (Brown et al., 2020) or BART (Lewis et al., 2020)) for data augmentation (Wang et al., 2022; Chintagunta et al., 2021). The key idea is to make use of trained

[3]Back translation consists of two steps: the first step is to translate a source sentence into another language by using a machine translation model. The second step is to translate the translated sentence back into the original language. These steps generate a slightly different version of the original sentence while preserving the original meaning.

generative models to generate texts conditionally. These model-based data augmentation methods are powerful because they can generate synthetic data, resulting in more diverse augmented data compared to the proposed methods (Ch 3, Ch 4), which are based on text-editing augmentation. Nonetheless, the drawback of these model-based data augmentation methods is that they require separate models, and the performances depend on the separate models.

Another potential approach is to reduce the model complexity rather than increasing the size of the data. For example, model pruning is an ongoing research topic that focuses on reducing the size of a deep learning model while not compromising on test performance (Michel et al., 2019; Hoefler et al., 2021). The key idea is to eliminate unimportant weights in a trained model to create a smaller and sparser model[4]. Even though previous studies have mainly focused on fast inference with a pruned model, recent works have produced promising results showing that pruning can also improve performance when fine-tuning with a small dataset (Liu et al., 2021a; Chen et al., 2021). Similarly, *lottery ticket hypothesis* is an emerging approach to training a smaller version of the model without sacrificing performance. The main idea is to identify sparse, small subnetworks (winning tickets) from an original model that can reach higher test accuracy (Frankle and Carbin, 2018; Chen et al., 2020a). The benefit of reducing model complexity is that it requires less computational power during inference. Therefore, we expect that pruning has the potential to build data-efficient and resource-efficient models.

7.2.2 Addressing Label Scarcity Problem

There are several promising solutions for overcoming the limitations of the methods proposed in this thesis. One potential approach to address the issue of unstable semi-supervised learning because of the model's overconfidence is to calibrate a trained model. As other researchers have pointed out, neural networks are generally overconfident about their predictions (Guo et al., 2017), and it is not reliable to interpret the model's predictions (e.g., softmax outputs or sigmoid outputs) as confidence scores (Gal and Ghahramani, 2016). To mitigate this, Guo et al. (2017) propose several methods to calibrate overconfident predictions. For example, temperature scaling is a simple yet effective calibration method that divides the logits ($e^{(z/T)}/\sum_i e^{(z_i/T)}$) by a scalar parameter $T > 0$. Further, using soft pseudo-labels could be a potential solution to mitigating the overconfidence issue. Unlike hard pseudo-labels that use the predicted output class (i.e., one-hot vectors), soft pseudo-labels use the predicted output

[4]Contrary to dropout that randomly deactivate nodes during training (Srivastava et al., 2014), pruning remove nodes that are unimportant. Therefore, pruning works as a type of model compression.

distribution (i.e., softmax outputs or sigmoid outputs). In general, soft pseudo-label approaches outperform hard pseudo-label approaches by addressing the noisy nature of pseudo-labels (Tanaka et al., 2018; Arazo et al., 2020; Zou et al., 2020). Therefore, we expect that incorporating the soft pseudo-label method could improve the proposed semi-supervised learning method.

Another potential approach is to combine data augmentation and semi-supervised learning. Data augmentation methods are mainly applied to labelled data in supervised learning settings, as shown for the **RQ1**. However, data augmentation methods can also be applied to unlabelled data in semi-supervised learning settings. For example, consistency regularisation approaches, also known as consistency training, are famous methods that use data augmentation for semi-supervised learning (Sajjadi et al., 2016; Tarvainen and Valpola, 2017). The consistency training assumes that small perturbations of data points should not modify model predictions given the same input. Thus, data augmentation is applied to unlabelled data to create perturbations, and a consistency regularisation term, defined as the mean squared error or Kullback–Leibler divergence metrics, is added to a loss function to encourage a model to produce robust predictions for noisy inputs. In other words, consistency training methods train a model to minimise the prediction difference between the original unlabelled sample and its corresponding perturbed version. Therefore, we expect that combining the proposed data augmentation methods and semi-supervised learning has the potential to further improve model performance with a smaller quantity of labelled data.

Similarly, another interesting future research direction is to combine the data augmentation and self-supervised learning. One potential approach is to use contrastive learning, which is a technique that aims to teach a model to learn discriminative features between different class samples and similar features for the same class samples. A recent study showed that a model trained on contrastive learning objectives with data augmentation learns better visual representations, which results in performance improvements (Chen et al., 2020b). Therefore, we expect that combining the proposed data augmentation methods in a pre-training phase with contrastive learning is a promising research direction for data-efficient training.

7.2.3 Utilising Knowledge and External Resources

There are a few potential areas of research for addressing the limitations of the proposed methods of using knowledge and external resources. First of all, a promising area that has been less explored is the methods for automatically generating appropriate synthetic data. As we discussed earlier for the first

research question (**RQ1**), using generative models or pre-trained language models could be an approach to automatically generating synthetic data for supervised learning. For example, we can use pre-trained language models to randomly substitute words in an original sentence based on the prediction of pre-trained models within a masked language prediction setting (Wu et al., 2019; Kumar et al., 2020; Chen and Yang, 2021) or to generate synthetic data through prompting (Liu et al., 2021b). However, as mentioned earlier, these approaches require additional resources for large models. Therefore, these approaches are useful for application domains that have enough computational resources but lack data.

Another potential area for future research is knowledge-enhancing approaches. In this dissertation, we proposed methods that utilise knowledge, such as human knowledge and knowledge database, to augment data and improve supervised learning. However, there are other approaches that focus on enhancing the knowledge of neural models for knowledge-intensive applications, such as medical code prediction use cases (Agarwal et al., 2019; Teng et al., 2020; Chang et al., 2020). These approaches utilise knowledge graphs to extract relevant knowledge and represent them into vectors, often called graph embeddings (Perozzi et al., 2014; Grover and Leskovec, 2016; Wang et al., 2016a), demonstrating that incorporating graph embeddings can improve performance and explainability. Therefore, we expect that combining graph embeddings and word embeddings is a promising approach to utilising knowledge, especially for application domains requiring domain-specific knowledge.

Finally, one interesting direction is to combine neural networks and symbolic, logic-based approaches. Neural networks are powerful at extracting and learning meaningful features from data but lack other capabilities, such as capturing relations and compositional structure or reasoning (Johnson et al., 2017). Instead, logic-based methods can capture these abilities due to their nature. Therefore, the neural-symbolic approach aims to combine the strength of two complementary approaches. Moreover, one of the major advantages of the neural-symbolic approach is its data efficiency because it utilises symbolic knowledge (d'Avila Garcez et al., 2019; De Raedt et al., 2020). Therefore, the integration of neural and symbolic approaches could be a key to building advanced and data-efficient NLP systems.

7.3 NLP in Healthcare: Multidisciplinary Challenges

Finally, we want to discuss the remaining legal, international, and ethical challenges in utilising NLP systems in healthcare. Even though these

multidisciplinary challenges are not extensively studied in this thesis, we believe that more multidisciplinary collaborations are required and hope our experiences from the HEART project can provide some insights.

The first multidisciplinary challenge is how to collect a large dataset, which is difficult not only practically but also legally. Especially in light of the General Data Protection Regulation (GDPR), each step of data acquisition and model development should follow data protection laws and privacy regulations. Moreover, healthcare companies like Philips typically apply higher legal standards to safeguard their customers. As a result, exploiting personal data from real customers is highly restricted. For example, Philips has an Internal Committee for Biomedical Experiments (ICBE) that reviews potential privacy, legal, and ethical issues of using data for all studies and controls data processing activities within the company. This constrains access to large, personal data for building machine learning models. This is the main reason why we used crowd-sourced data to train machine learning models in this thesis. The drawback is that crowd-sourced data are different from real-world data and the trained model requires iterative updates based on real-world data. This is partially addressed in Chapter 4 where we propose active learning to reduce the required number of labelled data. However, the proposed approach is still a long way from being applied in a real-world situation. Therefore, we expect more future work to mitigate legal and privacy issues of collecting data for machine learning model development, especially for healthcare applications to safeguard potential users.

The second challenge is how to scale internationally, which is critical when building NLP systems for different languages. When building multilingual NLP systems, rule-based approaches are limited because they require language experts who know the target language to program rules. Therefore, rule-based approaches are not scalable, require a lot of resources, and increase time to market when expanding to a new language. In this thesis, we partially address this multilingual challenge by proposing learnable systems. Learnable systems can scale better because they learn from data. Moreover, the proposed data-efficient algorithms reduce the amount of required data so that they can support fast development when building a model for a new language. However, our works focused on NLP systems for English-language text only and roughly 80% of the world population does not speak English (Crystal, 2008). This implies that the usability of proposed NLP systems is significantly limited and developing NLP systems for non-English languages or multilingual NLP systems is crucial. Leveraging multilingual language models or multilingual data for under-resourced languages would be an interesting research direction to address the multilingual, scalability challenge (Xue et al., 2021).

The third challenge is how to build a fair machine learning model. Fairness

is one of the top priorities in developing machine learning applications in healthcare to ensure that models perform ethically. This is also applicable to datasets used for training machine learning models. For example, if the datasets used to train machine learning models are imbalanced, they could result in building biased models and reinforcing systemic health disparities for minority populations (Röösli et al., 2022). In Chapter 6, we show how imbalanced data affect the performance of machine learning models and propose a method to address this challenge. However, the proposed method is a case study on a public benchmark dataset and our analysis is limited to the machine learning perspectives. Therefore, we advocate a close collaboration between machine learning societies and medical practitioners to monitor data collection protocols and assess the fairness of models when building healthcare applications.

7.4 Valorisation Plan

This section introduces the valorisation opportunities of the research discussed in this dissertation. Valorisation refers to the utilisation of scientific knowledge in practice creating economic and societal value. Examples include not only the commercialisation of scientific knowledge but also the dissemination of research results making scientific knowledge accessible to broader audiences. Throughout this dissertation, we propose novel data-efficient methods to address the data and label scarcity issues. The proposed methods include neural NLP models, data augmentation techniques, and learning strategies and we experimentally prove the effectiveness of the proposed methods.

In the following subsections, we will introduce potential domains that can benefit from the research results, including application domains and general audiences. Firstly, we explain how the developed NLP technologies can be applied to the personal healthcare domain (Ch. 7.4.1) and clinical domain (Ch. 7.4.2). For each potential application, we describe what additional steps are needed for real-world implementation. Lastly, we describe the borader impact of the developed NLP technologies (Ch. 7.4.3).

7.4.1 Personal Healthcare Applications

In this dissertation, we considered the business context of Philips Research and introduced potential applications for personal healthcare services. Specifically, we focused on a sleep coaching programme and potential applications in the three stages of the programme, including assessment, coaching, and monitoring.

Dialogue-based Sleep Triage System

The first step of a sleep coaching programme is a triage step to determine whether the patient is applicable for the coaching programme. Typically, this triage can be done by using long questionnaires, which is tedious and not engaging. To overcome this, one potential solution is to design a dialogue system that guides the user toward identifying their major sleep problems and the related causes. The dialogue system leverages NLP technologies to identify sleep problems from the users' complaints about their sleep. The benefit of using NLP technology is that users have more freedom to describe sleep-related issues in their own words. The developed sleep issue classifier proposed in Chapter 3 is a proof-of-concept version of a key NLP module in this application.

To deploy the developed sleep issue classifier for a real-world triage application, several steps are required. The first requirement is to integrate a classifier with clinical knowledge. At a high level, this can be done while defining the output label classes of a classifier. For example, we consulted with sleep experts to identify major sleep issue categories. At a low level, it is still needed to make a classifier grounded in clinical knowledge. Utilising a medical knowledge graph could be a potential approach to building a knowledge-grounded triage system (Li et al., 2020). Regarding the required infrastructure for this system, a possible solution is to deploy it on an edge device (e.g., a mobile) on a user side or a cloud server on a service provider side. When deploying the system on an edge device, privacy is protected by nature because data stay at the user side. The additional requirement, however, is that a classification model should be lightweight to be deployed and run on an edge device. On the other hand, the benefit of using the cloud is that it has more computation power for a large model. But the drawback is that it requires additional privacy-preserving functions. Additionally, when implementing the proposed classifier for a real-world application, a short response time should be considered to ensure that dialogue can flow without latency.

Motivational Interviewing Analysis

Another potential application of the developed system in Chapter 3 is analysing user feedback on healthcare devices to support usage. In this use case, people use not only a hardware device but also a mobile app. For example, Philips offers a continuous positive airway pressure (CPAP) machine to treat sleep apnea disorders, such as obstructive sleep apnea (OSA). Along with the hardware devices, including the CPAP machine and CPAP mask, Philips also offer a software app (DreamMapper) to help users stay motivated for the treatment by allowing them to check their progress, as illustrated in Figure 7.3. The desired

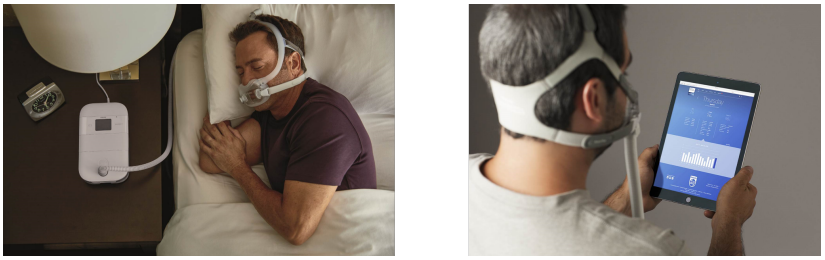


Figure 7.3: CPAP devices (left) and a mobile app (right).

app feature is to make people to be motivated for using the devices because CPAP compliance is important to treat OSA (Weaver and Grunstein, 2008). One potential solution is to use the motivational interviewing method, which encourages people to change their behaviour by focusing on their motivation (Aloia et al., 2013).

For this, a mobile app could allow users to describe why they are motivated to use CPAP or what blocks them from using it. Therefore, a potential valorisation option is to apply the developed NLP technology to analyse motivational interviewing data[5]. As a proof-of-concept, we built a classifier to analyse the motivational interviewing data by applying the data augmentation and semi-supervised learning methods proposed in Chapter 3. Since the developed methods are generic and reusable, we could apply them to a different application and achieve similar results[6].

There are several requirements to implement this application in the real world. Firstly, it is needed to identify root causes. For example, if user responses describe why users are not motivated because of specific issues, a system is required to differentiate issues between ones related to devices (e.g., a noisy machine, an uncomfortable mask) and ones related to personal circumstances (e.g., travelling, being sick). This can be satisfied by hierarchical classification, where the output labels are organized into a class hierarchy, such as a tree structure (Silla and Freitas, 2011). Furthermore, one of the important requirements is to allow the classifier to detect new emerging classes. For example, when a new issue could be reported by users, it is required to detect new labels. To fine-tune a model with new labels, an easy model update process

[5]A data collection protocol consists of three steps: firstly, mobile app users are asked to rank their motivation for using a device, on a scale of 1 to 10. Secondly, follow-up questions ask users to explain why they gave specific values, rather than a lower or a higher value, in their own words. Lastly, the collected responses are annotated with pre-defined labels. Then the annotated data are used for building a multi-label classification system.

[6]Because of the business circumstance, we do not report the details of data and experimental results in this dissertation.

is required. Lastly, it is required to integrate device data (i.e., usage data, sensor signals) with user inputs for monitoring compliance status. Regarding infrastructure, the classifier can be deployed in an edge device or a cloud server, similar to the sleep issue classifier.

Behaviour Change Challenge App

In Chapter 4, we introduced a potential mobile sleep coaching app that aims to support people to keep a healthier lifestyle for better sleep health. One of the desired features is the "Challenge" feature which helps people change their behaviour to improve their sleep quality. For example, the "caffeine challenge" asks people to stop drinking coffee in the late afternoon because it can negatively affect sleep quality. To achieve this, we developed an NLP system that can extract meaningful information from user inputs that they provide at the end of the challenge. Also, we developed an active learning framework to reduce manual labelling efforts and validated it with the semi-realistic user data obtained from a crowdsourcing platform.

For this application, there are a few requirements. The first required step is to validate the developed active learning framework on real-world datasets that are collected from real app users. It is also needed to validate the developed NLP system for other behaviour challenges[7] to verify the performance of the developed system. Moreover, it requires an additional feature that can detect the strength of expressed sentimental values to fully understand the user experience. For example, when a user expresses emotion towards a specific aspect (e.g., "*I cannot survive without coffee. I'm dying!*"), it is required to detect not only the sentimental value (i.e., negative) but also the strength of the sentiment (i.e., very strong). Similar to other applications, it is also required to deploy the developed system either on an edge device or a cloud server.

Free-Text Sleep Diary Tool

A sleep diary is a tool for monitoring sleep activity and assessing the quality of sleep. A typical sleep diary includes a series of questions to record the time of sleep-related events, such as bedtime and wake-up time (Carney et al., 2012). The main drawback of a current sleep diary tool is that it uses a structured questionnaire and does not allow people to report their feelings towards their sleep or provide additional information. Because of this limitation, the structured sleep diary misses an opportunity to get other possible important information, such as what happened throughout the night and how people

[7]Examples of other behaviour challenges include diet challenge and meditation challenge.

feel/perceive their sleep experience. To address these limitations, we introduced a free-text sleep diary use case that allows people to describe their sleep in their words and developed a temporal information extraction model (Ch. 5). Additionally, we have submitted an invention disclosure[8] about a free text NLP system that extracts both objective and subjective information and performs analytics on the extracted results. Figure 7.4 illustrated the examples of the proposed NLP system for free text sleep diary analysis.

Several requirements are needed to implement the proposed free text sleep diary for sleep monitoring. The first requirement is temporal reasoning ability to calculate sleep metrics, such as the total sleep time and sleep efficiency[9]. The second requirement is the additional functionality of extracting subjective information from free text sleep diary because we focused on extracting only temporal information in this dissertation. Regarding the required infrastructure for this system, a possible solution is to integrate with another internet of things (IoT) for health monitoring, such as smart watches, to add additional health-related temporal information automatically (e.g., exercise time).

7.4.2 Clinical Applications

In Chapter 6, we discussed the opportunity of utilising NLP technology for a clinical application. We proposed two NLP models that formulate medical coding as multi-label classification and binary classification, respectively. A potential valorisation option is to apply the proposed methods to the medical coding process for supporting human experts by predicting mentioned ICD codes from the given clinical documents as illustrated in Figure 7.5. The benefit of this potential application is to lessen the burden of human experts by automating partial steps in the manual coding process. For example, the developed NLP system can provide a candidate list of ICD codes and human experts can verify whether each prediction is correctly mentioned in the clinical document. We expect that this work could be of interest to a wide audience from the clinical and healthcare community.

However, there are special requirements to use the developed NLP technologies for clinical application. Firstly, it is required that the system achieves high recall scores because output label space could be extremely large. For example, the full MIMIC-III benchmark dataset (Johnson et al., 2016) contains almost 9,000 unique ICD codes. Secondly, the system needs to be validated on other datasets, including a large dataset (`MIMIC-III full`) and other datasets obtained

[8] Intellectual property rights belong to Philips Research and patent filling is in progress.

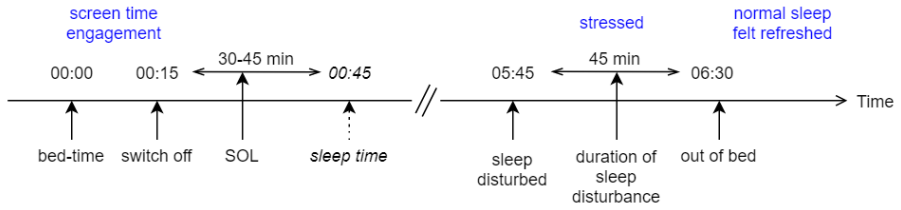
[9] Sleep efficiency is calculated by dividing the time a person is asleep by the total time in bed (Sleep Efficiency (%) = Total Sleep Time/Total Time in Bed) $\times$ 100.

I went to bed around 12'o clock. Used the phone for around 15 minutes and after that switched the light off. It took around 30 or 45 minutes to fall asleep. My sleep was disturbed at 5:45 am. and I spend in the bed for other 45 minutes to get sleep. I got off the bed around 6:30 am. Overall the sleep was normal. I think stress caused trouble in sleep. I felt refreshed.

(a) An example of free text sleep diary

Item	Value
Bedtime	00:00
Switch off	00:15
SOL	30-45 min
Sleep disturbed	5:45
Duration of disturbance	45 min
Out of bed time	6:30
Habit	Screen time
Quality	Normal
Issue	Stress
Feeling	Refreshed

(b) An example of extracted information



(c) Example of timeline.

Figure 7.4: Examples of free text sleep diary (a), the extracted information (b), and the visualised timeline (c). Underlined texts indicate and blue-coloured texts indicate temporal expression and additional information, respectively. SOL refers to sleep onset latency which is the time it takes a person to fall asleep after turning the lights out.

from different settings. Thirdly, it is required that domain experts evaluate the system and verify whether it is safe to use in a real-world application. Specifically, it is critical to include domain experts in development process to assess the quality of a system and identify when and how the system makes errors. Lastly, it is required to improve the explainability and causality of the system. Since the proposed systems are based on the neural networks model, which is a black box model[10], explainability is needed to understand why the system makes certain predictions. Also, enhancing the causality of the system is required for the target use case. For example, if a system predicts ICD codes based on other spurious correlations (e.g., frequently co-occurred ICD codes, a

[10]Block box model refers to a system that produces output without revealing any details about how it works internally.

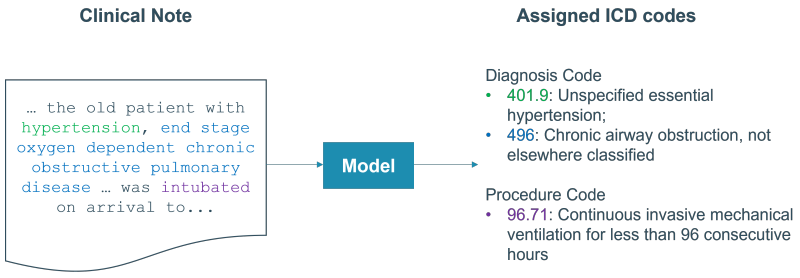


Figure 7.5: Illustration of a medical coding system.

correlation between demographics and certain ICD codes), it might create fatal issues. In our study (Ch. 6.2), we have discussed this problem and potential future directions.

7.4.3 Broader Impact

Additionally, we expect that our research can create societal values, including supporting healthcare professionals and empowering healthcare recipients. In this dissertation, we have investigated the possibilities of developing NLP applications for a sleep coaching programme. These NLP applications aim to provide a user-friendly interface for healthcare recipients and to support health professionals by focusing on two things: 1) applying NLP technologies to allow people to provide free-text inputs for a natural conversation; and 2) adding an analytic feature that automates processing free-text inputs to support decision making. As a result, this project could also contribute to the healthcare field by supporting health professionals to lessen the burden and allowing healthcare recipients to actively engage in the process of healthcare service. Especially in a post-pandemic era, when the possibility of telehealth service became more important than before, applying machine learning technologies for data analysis can play a key role in a healthcare domain (Lepore et al., 2022). Therefore, the developed NLP technologies for healthcare applications within out-of-hospital settings can provide useful insights to a healthcare community creating potential societal impact.

To exploit the proposed data-efficient methods and NLP models, it is required to validate them with other datasets. Further, the effectiveness of the proposed data-efficient methods is needed to be validated when using different neural models other than BERT. Moreover, when the proposed methods are applied to clinical use cases, further consideration is required to avoid undesirable model behaviour, such as learning from spurious cues in a training dataset (McCoy

et al., 2019) or performing differently between majority and minor labels (Shim et al., 2022). Therefore, an additional tool is required to test when a model makes errors or understand model behaviour.

Bibliography

Amina Adadi. A survey on data-efficient algorithms in big data era. *Journal of Big Data*, 8 (1):1–54, 2021.

Khushbu Agarwal, Tome Eftimov, Raghavendra Addanki, Sutanay Choudhury, Suzanne Tamang, and Robert Rallo. Snomed2vec: Random walk and poincaré embeddings of a clinical knowledge base for healthcare analytics. *arXiv preprint arXiv:1907.08650*, 2019.

Amr Alexandari, Anshul Kundaje, and Avanti Shrikumar. Maximum likelihood with bias-corrected calibration is hard-to-beat at label shift adaptation. In *International Conference on Machine Learning*, pages 222–232. PMLR, 2020.

Mark S Aloia, J Todd Arnedt, Matthew Strand, Richard P Millman, and Belinda Borrelli. Motivational enhancement to improve adherence to positive airway pressure in patients with obstructive sleep apnea: a randomized controlled trial. *Sleep*, 36(11):1655–1662, 2013.

Daniel Andor, Luheng He, Kenton Lee, and Emily Pitler. Giving bert a calculator: Finding operations and arguments with reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5947–5952, 2019.

Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. January 2015. 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015.

Maria-Florina Balcan, Andrei Broder, and Tong Zhang. Margin based active learning. In *International Conference on Computational Learning Theory*, pages 35–50. Springer, 2007.

Solon Barocas, Anhong Guo, Ece Kamar, Jacquelyn Krones, Meredith Ringel Morris, Jennifer Wortman Vaughan, W Duncan Wadsworth, and Hanna Wallach. Designing disaggregated evaluations of ai systems: Choices, considerations, and tradeoffs. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 368–378, 2021.

Jonathan Baxter. A bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine learning*, 28(1):7–39, 1997.

William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9368–9377, 2018.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

Jinbo Bi, Tao Xiong, Shipeng Yu, Murat Dundar, and R Bharat Rao. An improved multi-task learning approach with applications in medical diagnosis. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 117–132. Springer, 2008.

Michael Bloodgood and K Vijay-Shanker. A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 39–47, 2009.

Olivier Bodenreider. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270, 2004.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. doi: 10.1162/tacl_a_00051. URL https://aclanthology.org/Q17-1010.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Caroline Brun and Vassilina Nikoulina. Aspect based sentiment analysis into the wild. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 116–122, 2018.

Colleen E Carney, Daniel J Buysse, Sonia Ancoli-Israel, Jack D Edinger, Andrew D Krystal, Kenneth L Lichstein, and Charles M Morin. The consensus sleep diary: standardizing prospective sleep self-monitoring. *Sleep*, 35(2):287–302, 2012.

Angel Chang and Christopher D Manning. Sutime: A library for recognizing and normalizing time expressions. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 3735–3740, 2012.

David Chang, Ivana Balažević, Carl Allen, Daniel Chawla, Cynthia Brandt, and Andrew Taylor. Benchmark and best practices for biomedical knowledge graph embeddings. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing*, pages 167–176, 2020.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16: 321–357, 2002.

Jiaao Chen and Diyi Yang. Simple conversational data augmentation for semi-supervised abstractive dialogue summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6605–6616, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.530. URL https://aclanthology.org/2021.emnlp-main.530.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems*, 33:15834–15846, 2020a.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020b.

Xuxi Chen, Tianlong Chen, Yu Cheng, Weizhu Chen, Zhangyang Wang, and Ahmed Hassan Awadallah. Dsee: Dually sparsity-embedded efficient tuning of pre-trained language models. *arXiv preprint arXiv:2111.00160*, 2021.

Bharath Chintagunta, Namit Katariya, Xavier Amatriain, and Anitha Kannan. Medically aware GPT-3 as a data generator for medical dialogue summarization. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 66–76, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.nlpmc-1.9. URL https://aclanthology.org/2021.nlpmc-1.9.

Kyunghyun Cho, B van Merrienboer, Caglar Gulcehre, F Bougares, H Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 2014a.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014b.

Edward Choi, Mohammad Taha Bahadori, Joshua A Kulas, Andy Schuetz, Walter F Stewart, and Jimeng Sun. Retain: an interpretable predictive model for healthcare using reverse time attention mechanism. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 3512–3520, 2016.

Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163, 2017.

Michael Chui, James Manyika, and Mehdi Miremadi. What ai can and can't do (yet) for your business. *McKinsey Quarterly*, 2018.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

David Crystal. Two thousand million? *English today*, 24(1):3–6, 2008.

Sanjoy Dasgupta and Daniel Hsu. Hierarchical sampling for active learning. In *Proceedings of the 25th international conference on Machine learning*, pages 208–215, 2008.

Artur S. d'Avila Garcez, Marco Gori, Luís C. Lamb, Luciano Serafini, Michael Spranger, and Son N. Tran. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *FLAP*, 6(4):611–632, 2019. URL https://collegepublications.co.uk/ifcolog/?00033.

Luc De Raedt, Sebastijan Dumancic, Robin Manhaeve, and Giuseppe Marra. From statistical relational to neuro-symbolic artificial intelligence. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 4943–4950. ijcai. org, 2020.

Dina Demner-Fushman, Noémie Elhadad, and Carol Friedman. Natural language processing for health-related texts. In *Biomedical Informatics*, pages 241–272. Springer, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

Hai Ha Do, PWC Prasad, Angelika Maag, and Abeer Alsadoon. Deep learning for aspect-based sentiment analysis: a comparative review. *Expert Systems with Applications*, 118:272–299, 2019.

Barbara Snell Dohrenwend. Some effects of open and closed questions on respondents' answers. *Human Organization*, 24(2):175–184, 1965.

Liat Ein Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. Active learning for bert: An empirical study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, 2020.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, 2018.

Marthinus Christoffel Du Plessis and Masashi Sugiyama. Semi-supervised learning of class balance under class-prior change by distribution matching. *Neural Networks*, 50:110–119, 2014.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, 2019.

Seyda Ertekin, Jian Huang, Leon Bottou, and Lee Giles. Learning on the border: active learning in imbalanced data classification. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 127–136, 2007.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2018.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192. PMLR, 2017.

Mor Geva, Ankit Gupta, and Jonathan Berant. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 946–958, 2020.

Avijit Ghosh, Lea Genuit, and Mary Reagan. Characterizing intersectional group fairness with worst-case comparisons. In *Artificial Intelligence Diversity, Belonging, Equity, and Inclusion*, pages 22–34. PMLR, 2021.

Judy Wawira Gichoya, Liam G McCoy, Leo Anthony Celi, and Marzyeh Ghassemi. Equity in essence: a call for operationalising fairness in machine learning for healthcare. *BMJ Health & Care Informatics*, 28(1), 2021.

Julius Gonsior, Maik Thiele, and Wolfgang Lehner. Weakal: Combining active learning and weak supervision. In *International Conference on Discovery Science*, pages 34–49. Springer, 2020.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2016.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

Jiaxian Guo, Mingming Gong, Tongliang Liu, Kun Zhang, and Dacheng Tao. Ltf: A label transformation framework for correcting label shift. In *International Conference on Machine Learning*, pages 3843–3853. PMLR, 2020.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, 2020.

Kishaloy Halder, Alan Akbik, Josip Krapac, and Roland Vollgraf. Task-aware representation of sentences for generic text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3202–3213, 2020.

Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9 (8):1735–1780, 1997.

Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *J. Mach. Learn. Res.*, 22(241):1–124, 2021.

Neil Houlsby, Ferenc Huszar, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *CoRR*, abs/1112.5745, 2011. URL http://arxiv.org/abs/1112.5745.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1031. URL https://aclanthology.org/P18-1031.

Sheng-Jun Huang and Zhi-Hua Zhou. Multi-label learning by exploiting label correlations locally. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 949–955, 2012.

Huggingface. Pytorch - bert. https://github.com/huggingface/pytorch-transformers.

Vanessa Ibáñez, Josep Silva, and Omar Cauli. A survey on sleep assessment methods. *PeerJ*, 6:e4849, 2018.

Shaoxiong Ji, Matti Hölttä, and Pekka Marttinen. Does the magic of BERT apply to medical code assignment? A quantitative study. *Computers in Biology and Medicine*, 139, 2021. ISSN 18790534. doi: 10.1016/j.compbiomed.2021.104998.

Alistair EW 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*, 3(1):1–9, 2016.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017.

Katikapalli Subramanyam Kalyan and Sivanesan Sangeetha. Secnlp: A survey of embeddings in clinical natural language processing. *Journal of biomedical informatics*, 101:103323, 2020.

Akib Mohi Ud Din Khanday, Syed Tanzeel Rabani, Qamar Rayees Khan, Nusrat Rouf, and Masarat Mohi Ud Din. Machine learning based approaches for detecting covid-19 using clinical text data. *International Journal of Information Technology*, 12(3):731–739, 2020.

Byung-Hak Kim and Varun Ganapathi. Read, Attend, and Code: Pushing the Limits of Medical Codes Prediction from Clinical Notes by Machines. *Proceedings of Machine Learning Research*, 149:1–12, 2021. URL http://arxiv.org/abs/2107.10650.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1181. URL https://aclanthology.org/D14-1181.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Sosuke Kobayashi. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2072. URL https://aclanthology.org/N18-2072.

Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, page 0. Lille, 2015.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, 2020.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pages 6402–6413, 2017.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. Mixout: Effective regularization to finetune large-scale pretrained language models. In *International Conference on Learning Representations*, 2020a. URL https://openreview.net/forum?id=HkgaETNtDB.

Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 2, 2013.

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, 2020b.

Artuur Leeuwenberg and Marie Francine Moens. Temporal information extraction by predicting relative time-lines. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1237–1246, 2018.

Artuur Leeuwenberg and Marie-Francine Moens. Towards extracting absolute event timelines from english clinical reports. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2710–2719, 2020.

Dominique Lepore, Koustabh Dolui, Oleksandr Tomashchuk, Heereen Shim, Chetanya Puri, Yuan Li, Nuoya Chen, and Francesca Spigarelli. Interdisciplinary research unlocking innovative solutions in healthcare. *Technovation*, page 102511, 2022.

David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine learning proceedings 1994*, pages 148–156. Elsevier, 1994.

David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *SIGIR'94*, pages 3–12. Springer, 1994.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://aclanthology.org/2020.acl-main.703.

Fei Li and Hong Yu. ICD coding from clinical text using multi-filter residual convolutional neural network. In *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, volume 34, pages 8180–8187. AAAI press, apr 2020. ISBN 9781577358350. doi: 10.1609/aaai.v34i05.6331. URL https://ojs.aaai.org/index.php/AAAI/article/view/6331.

Linfeng Li, Peng Wang, Jun Yan, Yao Wang, Simin Li, Jinpeng Jiang, Zhe Sun, Buzhou Tang, Tsung-Hui Chang, Shenghui Wang, et al. Real-world data medical knowledge graph: construction and applications. *Artificial intelligence in medicine*, 103:101817, 2020.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and correcting for label shift with black box predictors. In *International conference on machine learning*, pages 3122–3130. PMLR, 2018.

Bingyan Liu, Yifeng Cai, Yao Guo, and Xiangqun Chen. Transtailor: Pruning the pre-trained model for improved transfer learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8627–8634, 2021a.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651, 2020.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *CoRR*, abs/2107.13586, 2021b. URL https://arxiv.org/abs/2107.13586.

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.

Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1166. URL https://aclanthology.org/D15-1166.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6294–6305, 2017.

Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1334. URL https://aclanthology.org/P19-1334.

Oren Melamud, Jacob Goldberger, and Ido Dagan. context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61, 2016.

Yuanliang Meng and Anna Rumshisky. Context-aware neural model for temporal information extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 527–536, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1049. URL https://www.aclweb.org/anthology/P18-1049.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013. URL http://arxiv.org/abs/1301.3781.

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38 (11):39–41, 1995.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. Explainable prediction of medical codes from clinical text. In *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, volume 1, pages 1101–1111, 2018. ISBN 9781948087278. doi: 10.18653/v1/n18-1100.

Qiang Ning, Zhili Feng, and Dan Roth. A structured learning approach to temporal relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1027–1037, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1108. URL https://www.aclweb.org/anthology/D17-1108.

Kimberly J O'malley, Karon F Cook, Matt D Price, Kimberly Raiford Wildes, John F Hurdle, and Carol M Ashton. Measuring diagnoses: Icd code accuracy. *Health services research*, 40 (5p2):1620–1639, 2005.

World Health Organization et al. *International classification of diseases:[9th] ninth revision, basic tabulation list with alphabetic index*. World Health Organization, 1978.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014a. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014b.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1756–1765, 2017.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://aclanthology.org/N18-1202.

Gerald Petz, Michał Karpowicz, Harald Fürschuß, Andreas Auinger, Václav Stříteskỳ, and Andreas Holzinger. Opinion mining on the web 2.0–characteristics of user generated content and their impacts. In *International Workshop on Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data*, pages 35–46. Springer, 2013.

Jason Phang, Thibault Févry, and Samuel R. Bowman. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *CoRR*, abs/1811.01088, 2018. URL http://arxiv.org/abs/1811.01088.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland, August 2014. Association for Computational Linguistics. doi: 10.3115/v1/S14-2004. URL https://www.aclweb.org/anthology/S14-2004.

Aaditya Prakash, Siyuan Zhao, Sadid A Hasan, Vivek Datla, Kathy Lee, Ashequl Qadir, Joey Liu, and Oladimeji Farri. Condensed memory networks for clinical diagnostic inferencing. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

Guo-Jun Qi and Jiebo Luo. Small data challenges in big data era: A survey of recent progress on unsupervised and semi-supervised methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018. URL https://openai.com/blog/language-unsupervised/.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Anton Ragni, Katherine Mary Knill, Shakti P Rath, and Mark John Gales. Data augmentation for low resource languages. pages 810–814, 2014. URL http://www.isca-speech.org/archive/interspeech_2014/i14_0810.html.

Alvin Rajkomar, Michaela Hardt, Michael D Howell, Greg Corrado, and Marshall H Chin. Ensuring fairness in machine learning to advance health equity. *Annals of internal medicine*, 169(12):866–872, 2018.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, 2016.

Abhilasha Ravichander, Aakanksha Naik, Carolyn Rose, and Eduard Hovy. Equate: A benchmark evaluation framework for quantitative reasoning in natural language inference. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 349–361, 2019.

Daniel Reker. Practical considerations for active machine learning in drug discovery. *Drug Discovery Today: Technologies*, 2020.

Tal Ridnik, Emanuel Ben-Baruch, Nadav Zamir, Asaf Noy, Itamar Friedman, Matan Protter, and Lihi Zelnik-Manor. Asymmetric loss for multi-label classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 82–91, 2021.

Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *The Journal of Machine Learning Research*, 5:101–141, 2004.

Filipe Rodrigues and Francisco Pereira. Deep learning from crowds. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020. doi: 10.1162/tacl_a_00349. URL https://aclanthology.org/2020.tacl-1.54.

Eliane Röösli, Selen Bozkurt, and Tina Hernandez-Boussard. Peeking into a black box, the fairness and generalizability of a MIMIC-III benchmarking model. *Scientific Data*, 9(1), 2022. ISSN 20524463. doi: 10.1038/s41597-021-01110-7. URL https://doi.org/10.1038/s41597-021-01110-7.

Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

Sebastian Ruder, Parsa Ghaffari, and John G Breslin. A hierarchical model of reviews for aspect-based sentiment analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 999–1005, 2016.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

Marco Saerens, Patrice Latinne, and Christine Decaestecker. Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. *Neural computation*, 14(1): 21–41, 2002.

Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. *Advances in neural information processing systems*, 29, 2016.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations*, 2018.

Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, 2016.

Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

Artem Shelmanov, Vadim Liventsev, Danil Kireev, Nikita Khromov, Alexander Panchenko, Irina Fedulova, and Dmitry V Dylov. Active learning with deep pre-trained models for sequence tagging of clinical and biomedical texts. In *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 482–489. IEEE, 2019.

Yanyao Shen, Hyokun Yun, Zachary C Lipton, Yakov Kronrod, and Animashree Anandkumar. Deep active learning for named entity recognition. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 252–256, 2017.

Haoran Shi, Pengtao Xie, Zhiting Hu, Ming Zhang, and Eric P. Xing. Towards automated ICD coding using deep learning. *CoRR*, abs/1711.04075, 2017. URL `http://arxiv.org/abs/1711.04075`.

Heereen Shim, Dietwig Lowet, Stijn Luca, and Bart Vanrumste. Lets: a label-efficient training scheme for aspect-based sentiment analysis by using a pre-trained language model. *IEEE Access*, 9:115563–115578, 2021.

Heereen Shim, Dietwig Lowet, Stijn Luca, and Bart Vanrumste. An exploratory data analysis: the performance differences of a medical code prediction system on different demographic groups. In *Proceedings of the 4th Clinical Natural Language Processing Workshop*, pages 93–102, Seattle, WA, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.clinicalnlp-1.10. URL `https://aclanthology.org/2022.clinicalnlp-1.10`.

Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

Aditya Siddhant and Zachary C Lipton. Deep bayesian active learning for natural language processing: Results of a large-scale empirical study. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2904–2909, 2018.

Carlos N Silla and Alex A Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1):31–72, 2011.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Mohammad S Sorower. A literature survey on algorithms for multi-label learning. *Oregon State University, Corvallis*, 18:1–25, 2010.

Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL `http://jmlr.org/papers/v15/srivastava14a.html`.

Chi Sun, Luyao Huang, and Xipeng Qiu. Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 380–385, 2019a.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer, 2019b.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5552–5560, 2018.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017.

Fei Teng, Wei Yang, Li Chen, LuFei Huang, and Qiang Xu. Explainable prediction of medical codes with knowledge graphs. *Frontiers in Bioengineering and Biotechnology*, 8:867, 2020.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics*, 8:621–633, 2020.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. Semeval-2010 task 13: Tempeval-2. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 57–62, 2010.

Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Michal Walczak, Julius Pfrommer, Annika Pick, et al. Informed machine learning-a taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

Thanh Vu, Dat Quoc Nguyen, and Anthony Nguyen. A label attention model for icd coding from clinical text. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3335–3341, 2021.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. Do NLP models know numbers? probing numeracy in embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5307–5315, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1534. URL https://aclanthology.org/D19-1534.

Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1225–1234, 2016a.

Keze Wang, Dongyu Zhang, Ya Li, Ruimao Zhang, and Liang Lin. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(12):2591–2600, 2016b.

William Yang Wang and Diyi Yang. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563, 2015.

Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Comput. Surv.*, 53(3), jun 2020. ISSN 0360-0300. doi: 10.1145/3386252. URL https://doi.org/10.1145/3386252.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615, Austin, Texas, November 2016c. Association for Computational Linguistics. doi: 10.18653/v1/D16-1058. URL https://www.aclweb.org/anthology/D16-1058.

Yufei Wang, Can Xu, Qingfeng Sun, Huang Hu, Chongyang Tao, Xiubo Geng, and Daxin Jiang. Promda: Prompt-based data augmentation for low-resource nlu tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4242–4255, 2022.

Terri E Weaver and Ronald R Grunstein. Adherence to continuous positive airway pressure therapy: the challenge to effective treatment. *Proceedings of the American Thoracic Society*, 5(2):173–178, 2008.

Jason Wei and Kai Zou. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1670. URL https://www.aclweb.org/anthology/D19-1670.

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*, pages arXiv–1910, 2019.

Jian Wu, Victor S Sheng, Jing Zhang, Hua Li, Tetiana Dadakova, Christine Leon Swisher, Zhiming Cui, and Pengpeng Zhao. Multi-label active learning algorithms for image classification: Overview and future promise. *ACM Computing Surveys (CSUR)*, 53(2): 1–35, 2020.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. Conditional bert contextual augmentation. In *International conference on computational science*, pages 84–95. Springer, 2019.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016. URL `http://arxiv.org/abs/1609.08144`.

Yuxiang Wu and Baotian Hu. Learning to extract coherent summary via deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Cao Xiao, Edward Choi, and Jimeng Sun. Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review. *Journal of the American Medical Informatics Association*, 25(10):1419–1428, 2018.

Xiancheng Xie, Yun Xiong, Philip S Yu, and Yangyong Zhu. Ehr coding with multi-scale feature attention and structured knowledge graph propagation. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 649–658, 2019.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1242. URL `https://aclanthology.org/N19-1242`.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR, 2015.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, 2021.

Wei Xue and Tao Li. Aspect based sentiment analysis with gated convolutional networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2514–2523, 2018.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763, 2019.

Ying Yu, Witold Pedrycz, and Duoqian Miao. Multi-label classification by exploiting label correlations. *Expert Systems with Applications*, 41(6):2989–3004, 2014.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7935–7948, 2020.

Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2013.

Ye Zhang and Byron C Wallace. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 253–263, 2017.

Zachariah Zhang, Jingshu Liu, and Narges Razavian. Bert-xml: Large scale automated icd coding using bert pretraining. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, pages 24–34, 2020.

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.

Xiaojin Jerry Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005.

Yue Zhu, James T Kwok, and Zhi-Hua Zhou. Multi-label learning with global and local label correlation. *IEEE Transactions on Knowledge and Data Engineering*, 30(6):1081–1094, 2017.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.

Angelo Ziletti, Alan Akbik, Christoph Berns, Thomas Herold, Marion Legler, and Martina Viell. Medical coding with biomedical transformer ensembles and zero/few-shot learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*, pages 176–187, Hybrid: Seattle, Washington + Online, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-industry.21. URL `https://aclanthology.org/2022.naacl-industry.21`.

Yuliang Zou, Zizhao Zhang, Han Zhang, Chun-Liang Li, Xiao Bian, Jia-Bin Huang, and Tomas Pfister. Pseudoseg: Designing pseudo labels for semantic segmentation. In *International Conference on Learning Representations*, 2020.

FACULTY OF ENGINEERING TECHNOLOGY
DEPARTMENT OF ELECTRICAL ENGINEERING
ESAT-STRADIUS
Andreas Vesaliusstraat 13 box 2600
3000 Leuven