# Efficient Large Language Models for the NHS and Psychiatry


**Niall Taylor**

St Hugh's College

University of Oxford

A thesis submitted for the degree of

*Doctor of Philosophy* in *Health Data Science*

Trinity term 2024

# Abstract

Electronic Health Records (EHRs) contain large volumes of unstructured clinical text, produced for a large variety of reasons. The clinical notes within EHR can be full of technical jargon and have a low signal-to-noise ratio. This poses a challenge for healthcare providers like the UK National Health Service (NHS), as making use of these texts to produce insights can be labour-intensive and requires domain expertise. This thesis explores methods for developing efficient, cost-effective, bespoke Large Language Models (LLMs) to understand NHS clinical text, enabling improved patient management and treatment. An enhanced pretraining regime, using contrastive loss on NHS clinical data, enabled the creation of NHS-specific LLMs within a day on a single GPU. These models outperformed open-source LLMs, facilitating faster adaptation to downstream clinical NLP tasks.

Traditional LLM fine-tuning is computationally expensive and challenging with larger models. Efficient adaptation methods, such as prompt learning, were developed and employed, reducing computational and storage requirements by up to 98% while maintaining state-of-the-art performance on several clinical NLP tasks. The bespoke NHS LLMs and efficient adaptation methods were applied to a digital triage system for secondary mental health referrals. This system aimed to improve transparency, accuracy, and efficiency in routing patients to appropriate care pathways based on their clinical information. The resulting model processed variable-length patient referral text and produced triage team recommendations with an explainability tool to enhance interpretability. Crucially, the triage model

remained cost-effective and feasible in resource-constrained environments. This work evaluates the feasibility, utility, and potential benefits of developing specialized LLMs for NHS clinical text processing, discussing implications for enhancing patient care and clinical decision support.

# Acknowledgements

It is not possible to finish a DPhil without some scientific and emotional help. I would like to dedicate this space to thank all the people who were supporting me during this time.

First, I'd like to thank my supervisors, Alejo Nevado-Holgado and Andrey Kormilitzin, who provided extensive expertise and knowledge to guide me and my DPhil along the correct path. They were constant sources of support and exhibited tremendous patience with my rather forward working style. Alejo as my primary supervisor took on the most responsibility and kept me on track, with a calm and relaxed working environment.

I'd like to next make special thanks to Dan W Joyce, my clinical and unintentional career consult throughout. Without his expertise and guidance, this work would have lacked direction and importance.

I would not have been able to finish my DPhil without the help of my family. They were always available for a telephone call after a stressful week and gave me a place to stay when a change of scene was needed. My parents could not have been more supportive, not just during the DPhil, but in everything that has led me here.

The biggest thank-you goes to Anna-Sophia, without whom I would have thrown in the towel long ago. She truly kept me sane and put up with my frequent complaining, pessimism and impatience (far more than anyone should). She was

there for me during any difficulties that arose, often putting my worries before her own and I can't thank her enough. Beyond that, she just makes me happy and I couldn't ask for more.

Finally, I want to thank all my friends here in Oxford, back home, and wherever in the world they have ended up. My friends were able to keep life satisfaction and goals in perspective. They provided a much-needed avenue for laughter and fun to escape the woes of PhD life.

Without the encouragement, guidance, and unwavering love from all those mentioned above, this accomplishment would not have been possible. This thesis is a testament to the power of collaboration, resilience, and the invaluable contributions of others.

# Originality

**Statement** The writing of this thesis is my original work, and is either:

- my original work either with or without collaborators,
- relevant prior or concurrent work included for reference, so as to provide a survey of the field.

**Publications** This thesis contains material from the following papers on the efficient use of LLMs for clinical applications and secondary mental health triage, organised in reverse chronological order:

- Niall Taylor, Andrey Kormilitzin, Isabelle Lorge, Alejo Nevado-Holgado, and Dan W. Joyce. "Bespoke Large Language Models for Digital Triage Assistance in Mental Health Care." arXiv preprint arXiv:2403.19790 (2024). Code available at: https://github.com/NtaylorOX/OHFT_triage.
- Niall Taylor, Upamanyu Ghose, Omid Rohanian, Mohammadmahdi Nouriborji, Andrey Kormilitzin, David Clifton, and Alejo Nevado-Holgado. "Efficiency at Scale: Investigating the Performance of Diminutive Language Models in Clinical Tasks." arXiv preprint arXiv:2402.10597 (2024). Code available at: https://github.com/nlpie-research/efficient-ml
- Niall Taylor, Dan Schofield, Andrey Kormilitzin, Dan W. Joyce, and Alejo Nevado-Holgado. "Developing Healthcare Language Model Embedding Spaces."

arXiv preprint arXiv:2403.19802 (2024). Code available at: https://github.com/NtaylorOX/Healthcare_LLM_Embeddings

- Niall Taylor, Zhang, Yi, Joyce, Dan W, Gao, Zimming, Kormilitzin and Nevado-Holgado, Alejo, "Clinical Prompt Learning With Frozen Language Models." in *IEEE Transactions on Neural Networks and Learning Systems*, doi: 10.1109/TNNLS.2023.3294633. Code available at: https://github.com/NtaylorOX/Public_Clinical_Prompt.

- Niall Taylor, Lei Sha, Dan W. Joyce, Thomas Lukasiewicz, Alejo Nevado-Holgado, and Andrey Kormilitzin. "Rationale production to support clinical decision-making." arXiv preprint arXiv:2111.07611 (2021).

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **AI** | Artificial Intelligence |
| **AUC** | Area Under the Curve |
| **BERT** | Bidirectional Encoder Representation Transformer |
| **CMHT** | Community Mental Health Team |
| **CPU** | Central Processing Unit |
| **DL** | Deep Learning |
| **ED** | Eating Disorders |
| **EE** | Event Extraction |
| **EHR** | Electronic Health Record |
| **EIP** | Early Intervention for Psychosis |
| **FLOPs** | Floating Point Operations |
| **FSL** | Few-shot Learning |
| **GB** | GigaBytes |
| **GDPR** | General Data Protection Regulation |
| **GPT** | Generative Pretrained Transformer |
| **GPU** | Graphical Processing Unit |
| **ICD-9** | International Classification of Diseases |
| **IE** | Information Extraction |
| **LD** | Learning Difficulties |
| **LGBTQ** | Lesbian Gay Bisexual Transgender Queer |

| | |
|---|---|
| **LLM** | Large Language Model |
| **LoS** | Length of Stay |
| **LSTM** | Long Short Term Memory |
| **MHT** | Mental Health Team |
| **ML** | Machine Learning |
| **MoE** | Mixture of Experts |
| **MP** | Mortality Prediction |
| **NER** | Named Entity Recognition |
| **NHS** | National Health Service |
| **NLP** | Natural Language Processing |
| **NN** | Neural Network |
| **MDD** | Major Depressive Disorder |
| **MDT** | Multi Disciplinary Team |
| **OHFT** | Oxford Health Foundation Trust |
| **PEFT** | Parameter Efficient Fine Tuning |
| **PD** | Perinatal Psychiatry |
| **PL** | Prompt Learning |
| **PLM** | Pre-trained Language Model |
| **RNN** | Recurrent Neural Network |
| **ROC** | Receiver Operator Characteristic |
| **SoTA** | State of the Art |
| **SPoA** | Single Point of Access |
| **TRE** | Trusted Research Environment |
| **t-SNE** | t-Distributed Stochastic Neighbour Embedding |

| | |
|---|---|
| **UMLS** | Unified Medical Language System |
| **VRAM** | Video Random Access Memory |
| **ZSL** | Zero-shot learning |

Chapter 1

# Introduction

Clinical notes within Electronic Health Records (EHRs) are integral for patient management and treatment but are produced in massive volumes, prioritizing documentation over human interpretability (*1*). Natural Language Processing (NLP) and deep learning have become powerful tools for dealing with these large volumes of text by creating numerical representations. NLP is now dominated by Large Language Models (LLMs), with state-of-the-art (SoTA) performance across many domains, tasks, and languages (*2–4*). LLMs have been popularised through the proliferation of closed-source chatbot APIs such as Chat-GPT (*5*), Gemini (*6*), and Claude-2/3 (*7*). Moreover, several companies, such as Meta with its Llama models (*3*) are routinely releasing powerful, and very large open-source LLMs for everyone to use (given sufficient computational resources).

However, public LLMs remain limited in understanding complex clinical text from EHRs, due to data governance, confidentiality, and safety concerns. Even massive LLMs like GPT-4 (*5*) require enhancing clinical knowledge through further training or alignment (*8–12*) to make sense of unseen clinical notes.

**Building efficient bespoke NHS language models** To produce an LLM capable of understanding and using NHS clinical text requires an LLM trained using NHS clinical text, of which none are publicly available. Following the creation of an LLM, one has to further adapt and fine-tune these models to specific downstream tasks and applications, leading to domain and task-specific LLMs. LLMs vary dramatically in scale today, with smaller LLMs such as BERT (*2*) having 108 million model parameters, and much larger models, such as PaLM (*13*, *14*) having 540 billion. The largest LLMs remain out of reach for most research groups, especially those working in resource-limited research environments. Even with the smaller LLMs, the combined pre-training, fine-tuning and inference stages are computationally expensive and time-demanding. They require costly Graphical Processing Units (GPUs), or reliance on cloud-based providers such as Google Cloud or Amazon Web Services.

Nevertheless, the potential power of LLMs for different NHS trusts, healthcare entities and research is likely worth the investment, but with careful considerations and guidelines in place. Given the massive volume of free text in different NHS and healthcare entities, bespoke and domain-specific LLMs offer a powerful tool for text processing. A single NHS trust that can train its own LLM can provide a foundation for different research avenues and downstream applications.

The first goal of my DPhil was to develop and showcase, the feasibility and utility of training bespoke NHS LLMs in a resource-limited Trust Research Environment (TRE). On top of this, I explored how improved language model pre-training methods can further enhance the LLM's adaptability to subsequent downstream tasks (covered in Chapter 3).

Once the NHS trust LLMs were trained, the next objective was to efficiently adapt them to new tasks, improving upon fully fine-tuning a separate LLM for each task, such as text classification (e.g., mapping documents to classes - film reviews labelled "positive"). A major driving force behind training LLMs is the use for downstream tasks, and in my case, potential clinical applications within secondary mental health. A common downstream task using LLMs is classification, or clinical outcome prediction using clinical notes. Adapting LLMs to predict specific class labels typically involves supervised training of all model weights on that dataset and task, using what is known as traditional fine-tuning. The main problem with this approach is the computational resource required to update LLMs is substantial and time-consuming. Moreover, task-specific LLMs may lose some language modelling ability due to the training shifting away from the original pre-training objective. Consequently, a diverse set of domains and tasks yields an equally diverse set of specific, non-interchangeable LLMs. Multiple LLMs can be problematic due to LLM size and considerable storage requirements (4GB for a small BERT model in full precision, or >300GB for larger models).

I investigated methods to enhance the efficiency of task-specific fine-tuning of LLMs by lowering the number of model parameters updated for each task, thus reducing the compute and storage requirements for task-specific LLMs. In turn, this would improve the re-usability of the base LLMs for many different tasks. Utilising several popular publicly accessible clinical NLP datasets I conduct a large volume of experiments to uncover the best performing, efficient, and usable downstream adaptation methods (covered in Chapter 4). The results will then inform the design of an end-to-end modelling pipeline applied to a real-world secondary

mental health triage problem in Chapter 5.

**Secondary Mental Health Digital Triage** Within the UK, the referral process within secondary mental health involves a team of clinicians reviewing a patient's referral documents and associated information that is typically stored in free text and routing them to appropriate teams or care pathways. Unfortunately, the referral and triage process is not always transparent, accurate, or efficient, leading to a problem of a very time-consuming process. Moreover, there is the potential for patients to be routed to inappropriate teams or rejected entirely - with little insight into the decision-making involved.

To assist the triage process, I sought to combine the trained NHS LLM from the earlier chapters with efficient task adaptation methods to produce a prototype triage referral recommendation system using data from the local NHS Oxford Health Foundation Trust (OHFT). First, by automatically creating numerical document representations of clinical notes using the LLM, then training a triage team recommendation model. On top of this, I introduce an explainability component that provides a view of the inner workings of the underlying model outputs (covered in Chapter 5). The triage tool prototype serves as a research piece only but can provide a starting point for future research and the further development of a deployable pipeline.

## 1.1 Thesis Aims

The research questions I hope to answer in this thesis are:

- How to create feasible and effective bespoke NHS trust LLMs be developed in TREs
- What are the benefits of generating specific NHS trust LLMs and how can they be evaluated?
- How can traditional fine-tuning be made more efficient and how can this be quantified?
- Is it possible to derive a model for secondary mental health triage support using LLMs?

I will start my thesis with a thorough review of relevant literature and seminal papers that provide the foundation for this work, followed by an overview of some general methods that are common across the subsequent experimental chapters. Whilst I presume readers will have a knowledge of deep learning and clinical Natural Language Processing (NLP), and do not endeavour to detail exactly how neural networks work to their most basic form, I do provide a gentle introduction of key methods and concepts in my preliminaries section 2.5. Each experimental chapter (3, 4, and 5) will have a structure of Introduction Methods Results and Discussion (IMRAD). In my Chapter 6, I will discuss the collective results, future directions and ethical considerations of my work and beyond.

Chapter 2

# Literature review

I start with a brief overview of the history of LLMs and how they transformed the NLP field. Then, I review the growing desire and need for UK-specific clinical (healthcare) LLMs through domain adaptation and transfer learning. and in particular why this is particularly important for the UK and the NHS. The penultimate section will focus on the efficiency problem associated with adapting LLMs to new domains and tasks and avenues to improve these. The final section of the literature review will focus on a specific case study in secondary mental health and psychiatry, whereby I propose the potential value of LLMs in assisting with the triage and referral process.

Following the literature review, I will present some technical details of the types of LLMs, methods and concepts used throughout this work.

## 2.1 Introduction to LLMs

LLMs have become a phenomenon of global prominence recently in 2023/2024, as a major component of the generative artificial intelligence (AI) era. Their roots are formed in the wider field of NLP, which has long sought to provide computers with the ability to represent human-written text.

To understand how LLMs work, it is essential to grasp the basic principles of

neural networks. They consist of interconnected layers of nodes called neurons. A neuron is a fundamental computational unit that performs a simple mathematical operation on its inputs to produce an output. It is inspired by the biological neurons in the human brain, which receive signals from other neurons, process them, and then transmit the processed signal to other connected neurons. In reality, these are objects in computer memory called tensors. These neurons pass information along weighted connections, with the weights determining how each neuron's output influences other neurons. Neural networks are trained by providing input data and desired outputs, enabling the network to make predictions and calculate errors. Through a process called back-propagation and optimization techniques like gradient descent (*15*), the weights between neurons are iteratively adjusted to minimize prediction errors. As training progresses over numerous iterations, the network "learns" patterns between inputs and outputs by optimizing the weights and neuron activations to reproduce the desired outputs accurately. Once trained, the network can make predictions for new inputs it was not exposed to during training. This training forms the fundamental basis for LLMs and other neural network models, with variations in architectures and training objectives. Discussed below.

### 2.1.1 Natural Language Processing

The field of NLP began in the mid-twentieth century (*16*) intending to teach machines to perform useful tasks and solve problems involving human language. This requires developing algorithms that process linguistic data - recognizing patterns that convey meaning, much like humans do. Initially, NLP efforts were heavily

focused on Information Extraction (IE) and text mining. IE is the automatic extraction of structured information or specific entities from text. This involves identifying and extracting relevant pieces of information, such as named entities (e.g., people, organizations, locations), relationships between entities, events, attributes, and facts from text data. As the field progressed, more complex goals were pursued, such as machine translation (translating from one human language to another, e.g., English to French), which encouraged the emergence of language modelling approaches. I do not think it serves a purpose to provide a detailed history of all milestones in NLP that led to modern LLMs. I will rather provide an overview of the most recent decade that produced the first LLMs.

### 2.1.2 Language models and Pretrained LLMs

Generally speaking, a language model is a statistical model that learns the patterns and probabilities of how words and sequences of words are used in a particular language[1] (or tokens, for the definition of tokens see section 2.5.2.2). The common objective of LMs is to predict the next word given a history of text, or context. LMs are a family of different architectures and algorithms with varying complexity and use. LMs today are a little more abstract as model complexity and language modelling objectives evolved. A key driving force of change was the development of neural networks. Initial language models in the early 2000s relied on comparatively simple neural networks, such as a feed-forward neural network with just two linear layers paired with non-linear activation functions (*17*). Next, was the

[1]LLMs technically interact with tokens, which are not always direct mappings of words. Casually one can say LLMs process words, and there will be times I use these terms interchangeably

increasing size and depth of neural networks, with models emerging with tens or hundreds of layers, the so-called deep learning models. For example, the Recurrent Neural Network (RNN) became a powerful choice for LMs from 2010 onwards (*18*, *19*). RNNs work by processing words in a sequence, one by one - generating, maintaining and combining numerical representations of each word. RNNs are capable of processing any length of sequence, and in theory, maintain a memory of the earlier parts. This proved powerful for language modelling. However, they did not fare well with longer sequences, as the information and influence of the beginning of the sequence diminished over time and were effectively lost by the end of the sequence. This effect is known as vanishing gradients (*20*).

To circumvent this problem, Long-short Term Memory (LSTM) models were introduced with a gating mechanism, which routed earlier representations forward via skip connections. This better maintained early sequence representations and mitigated the gradient vanishing problem (*21*, *22*). LSTMs remained popular, especially with the introductions of bidirectional LSTMs and attention mechanisms (*23*, *24*). One problem with both RNNs and LSTMs is that they rely on processing one word or element of the sequence at a time and essentially passing this information along. This generally meant long training times, and whilst the vanishing gradient problem had been improved - it was not resolved.

A lasting concern for the aforementioned LMs was the slow training and use of static word embeddings. A major goal of LM models is to produce meaningful numerical representations of words, typically represented as numerical vectors or embeddings (described in greater detail later in section 2.5.3.1). An LM will produce representations for each word in its known vocabulary, much like humans

do. Typically these embeddings are randomly initialised numbers and updated through long training regimes. Researchers realised that creating or pre-training good word embeddings was a possible route to speed up the LM training process. For instance, Word2vec (*25*) was a popular model, a simple neural network trained to create vector representations based on context. The model was trained by feeding in passages of text and tasked with predicting a target word given the surrounding words, or the reverse, by using a target word to predict the surrounding context. Given a large and varied training set, these pre-trained Word2Vec representations could be used as the starting word vectors (embeddings) in other neural networks, such as the LSTMs. This leads to improvements across many NLP tasks and datasets (*17*) - and importantly reduces the training time.

The Word2Vec-derived vectors are however static, in that each word has one vector representation, regardless of the sentence or context it appeared in. For example, the sentences *"They were having a picnic by the river* ***bank****"* and *"The robbers attempted to break into the* ***bank****"* would have the same word embedding for **"bank"**, which is not ideal given the very different contextual meanings. From this arose a desire to improve the representation of words based on their context.

In 2017, perhaps the most influential neural network architecture and seminal paper for modern AI was released: the transformer (*26*). The transformer's architecture will be covered in greater detail in the preliminaries Section 2.5.3. A crucial component introduced by the transformer is positional awareness, self-attention and multi-head attention mechanisms that operate on entire sequences in parallel. This transformed these static word embeddings into more nuanced, contextualised embeddings, where word embeddings were directly influenced by the rest of the

sequence. In the original paper (*26*), the transformer architecture was used for neural translation whereby a sequence of English text was passed as input and the output was the direct translation to another language, such as French, with great success. The transformer model paved the way for language models, and the last 7 years have been dominated by the training and release of transformer-based large Pretrained Language Models - more popularly known as LLMs[2].

Modern LLMs, much like older LMs, are produced via training on massive amounts of free text data: As an example of how large these text datasets have become, a recent LLM from Microsoft, Phi-3 (*27*) was trained on 3.3 trillion tokens (words). To put that into perspective - in 2024 the entirety of Wikipedia is estimated to have 4.5 billion words. The pre-training involves a choice of different language modelling objectives which influence the possible use cases of LLMs. One popular objective is Masked Language Modelling (MLM), introduced in Bi-Directional Encoder Representation from Text (BERT) (*2*). MLM can be viewed as a "fill in the gap" task, where parts of real human written text are corrupted (removed or replaced) and the model has to predict what should have gone there. MLM models are powerful for generating useful embeddings and are used extensively for traditional NLP tasks, like text classification, and information extraction. A second objective is Autoregressive (causal) language modelling (ALM), popularised by the Generative Pre-trained Transformer (GPT) (*9*, *28*), which involves predicting the next word in a sequence given the previously seen words. ALM models are particularly useful for text generation and are the backbone of many recent Chatbot services (ChatGPT (*5*) for example). Finally, there is Sequence-to-Sequence (Seq2Seq),

[2]The terms PLM and LLM are often used interchangeably. Throughout my thesis I will be using the term LLM

used in the original transformer paper (*26*), which maps an input sequence of text to a specific output sequence - such as with English to French translations. These LM objectives are explained in more detail in the preliminaries Section 2.5.2.1.

With these LLMs one can adapt them to new domains and downstream tasks with relative ease through a process known as fine-tuning, often resulting in state-of-the-art results on several popular benchmark datasets and tasks (*4*, *17*, *26*, *29*, *30*). Fine-tuning is a crucial process when using LLMs on new datasets and tasks, whereby the already pre-trained LLM is further trained. The main purpose of fine-tuning is to update the LLM model weights to perform a new task, likely different to the pre-training objective. One such task is text or sequence classification, where a piece of text has an associated class label e.g. a clinical discharge summary labelled as discussing a certain health condition (see section 2.5.7). Adapting the LLM to achieve the classification of the text requires changes to the LLM by adding classification heads or additions of new models. Fine-tuning involves training the base LLM model alongside the new additional *classification head* using the new dataset and labels to achieve the task (see section 2.5.5 to learn more about classification heads).

### 2.1.3 Foundational LLMs

Through increases in model size and training volume, we now have many so-called foundational LLMs from large tech companies emerging from mid-2021 onwards. These LLMs are intended to serve as general-purpose models. To give the reader a context of how large they have become, I am listing just a few that are currently popular in research in 2023/2024: Llama-2/3 models (*3*) from Meta, and Mistral

or Mixtral from MistralAI (*31*) range from 7 to 80 billion parameters. Gemini, PaLM, or Bard from Google have sizes up to 540 billion parameters (*6*, *32*). GPT-3 and 4 from OpenAI have estimated sizes of several hundred billion parameters (although this is undisclosed (*5*)). Consider early LLMs like BERT in 2018 only had approximately 100 million parameters.

**Open vs Closed-source LLMs** There is a key distinction to be made in the world of LLMs between closed- and open-source models:

Closed-source LLMs, such as those developed by major technology companies like Google and OpenAI, are trained on vast datasets in a centralized and controlled manner. The training data, model architecture, and fine-tuning processes are closely guarded trade secrets and the resulting models are not publicly released. Moreover, their financial backing and computing resources are incredible. OpenAI, for example, has now raised around $12 billion, with its main product being LLMs and other generative AI models. This funding allows training typically unattainable by public institutions such as universities and government bodies (University of Oxford, Dept. Psychiatry had a total grant value of £70 million in 2023 (*33*)). The closed models are typically utilized within the confines of the company's products and services, with limited external access or customization options. Popular examples include: GPT-3.5/4 (powering Chat-GPT) from OpenAI (*5*), BARD and Gemini from Google (*6*), Claude-2/3 from Anthropic (*7*) and Grok from xAI (*34*). There are comparatively few closed-source LLMs but they command a massive reputation and dominate the conversation.

In contrast, open-source LLMs are developed through collaborative efforts, often

involving academic institutions, research organizations, and open-source communities such as HuggingFace (*35*). The training data, model architectures, and fine-tuning procedures are openly shared to foster transparency and reproducible research. These models are freely available for public use, modification, and further research to enable a wide range of applications and innovations. Popular open LLMs include the Llama models from Meta (*3*, *36*) and the Phi models from Microsoft (*37*).

The open paradigm promotes the democratization of AI technologies and allows researchers, developers, and enthusiasts to contribute to and benefit from these powerful language models. However, closed LLMs may have advantages in terms of computational resources, access to proprietary data, and specialized hardware and can potentially yield superior performance in certain domains or tasks. The choice between open and closed LLMs ultimately depends on the specific requirements, priorities, and ethical considerations of the intended application or research endeavour.

For my work, which requires transparency, offline functionality, cost-effectiveness and to be ethical, I relied on open-source LLMs. I will discuss this in greater detail in Chapter 6.

Hereby, I will state clearly that the research presented in this thesis does not make any attempt to utilise all possible LLMs and associate technology. It remains critical to utilise LLMs that can be paired with private and secure datasets. The landscape for utilising popular APIs such as ChatGPT (*5*) for healthcare data is in its infancy, with data governance and laws not properly formed for their use with private data such as NHS patient data used in my work. For this reason, I

also do not make any attempt to use any closed LLMs.

## 2.2 Clinical LLMs

Although LLMs have shown tremendous abilities in processing and understanding human-written text, they have limitations. A notable limitation, especially of smaller LLMs, is the drop in performance when directly applied to text data different from the domain the model was pre-trained on. A domain refers to the nature or topic area of the text, and some examples are finance, law, biomedical, or clinical. Most LLMs have been trained on text sourced from the public domain and scraped from web pages. For example, one popular LLM training dataset is *The Pile* (*38*) - consisting of 825 GB of text from webpages. These datasets are proposed to cover many different domains and writing styles, including creative writing or more formalised coding languages. However, we still see that these models often underperform in biomedical and clinical text, especially specialist text found in EHRs (*9*, *39–43*). For example, a recent study by Rohanian et al. (2023) explored open-source and closed-source LLMs (including Chat-GPT (*5*) and Claude-2 (*7*)) for biomedical NLP tasks, including NER, sequence classification, and relation extraction. They found that without fine-tuning, the models performed quite poorly (*43*). They showed that a comparatively small 108 million BioClinicalBERT (*44*) model matched the performance of a significantly larger 13 billion parameter Llama-2 model (*3*). Thus, it appears that foundational LLMs cannot be used directly to obtain peak performance on clinical NLP tasks without domain or task adaptation.

**Biomedical versus clinical text** Two common domains that align with my interests are the biomedical and the clinical[3] domain. The distinction between these two may not be immediately apparent but can be surmised as follows: Biomedical text in the LLM literature and training data typically consists of scientific texts in bio-medicine and life sciences, e.g. PubMed literature. These usually follow more standard writing styles and plain English alongside technical jargon. In contrast, clinical texts are unstructured notes collected routinely by healthcare professionals, often describing patient encounters with varying writing styles (*45*). Within EHRs, a distinction exists between unstructured and structured texts. Unstructured texts are narrative-style notes, noisy and voluminous, typically from nurses, clinicians, and administrators without enforced rules or syntax, making them challenging. Structured texts adhere to data types and entry rules, containing categorical information such as demographics, lab results, and quantitative measurements, easily retrievable from databases. This thesis focuses almost entirely on unstructured clinical data.

There is a plethora of biomedical and clinical LLMs. However, the majority are trained on a limited amount of clinical data and the source of these data is typically from a single EHR system i.e. MIMIC-III (*46*). The MIMIC-III dataset (The Medical Information Mart for Intensive Care III) is used throughout this thesis and contains a large volume of EHR records and clinical notes for a US-based healthcare provider (explained in more detail in Section 2.5.1.1). Popular biomedical and clinical BERT-style models (encoder-only MLM LLMs) include BioBERT (*47*), ClinicalBERT (*48*), ClinicalBioBERT (*49*), PubMedBERT (*50*), BioLinkBERT

[3] I typically refer to text within healthcare settings as clinical and healthcare text interchangeably

(*51*), DRAGON (*52*). Popular GPT style models (decoder-only autoregressive) include BioGPT (*53*), BioMedLM (*54*), Meditron (*55*), and MedPaLM (*56*).

The clinical BERT models have led to promising improvements upon the generally trained base versions in traditional clinical NLP tasks, with State of The Art (SoTA) performance on several datasets (*11*, *45*, *47*, *57*). Ling et al. (2023) found that BioClinicalBERT significantly outperformed the standard BERT model, with an 11% increase in F1 score on a drug review sentiment classification task. However, there are cases with a less clear performance gain. For instance, Belkadi et al. (2023) (*58*) used ClinicalBERT for clinical NER tasks to identify certain medical events within text and found that the pre-trained BERT model performed very similarly with no obvious gains from the pre-training. This may highlight a more general difficulty in aligning LLMs pretraining with the downstream task, as opposed to a problem with domain adaptation.

The generative GPT style LLMs have also been used to great effect for certain styles of NLP tasks. One prominent study by Singhal et al. (2023) (*56*) used Google's 540 billion parameter model, Med-PaLM, to achieve SoTA on several medical question-answer and information extraction datasets. They found that careful instruction-based fine-tuning allowed the model to answer questions like trained clinicians. However, certain NLP tasks such as text classification and NER remain difficult for generative LLMs (*41*, *59*).

Clinical NLP research with LLMs has revolved around heavily curated datasets and specific US-based clinical datasets. The application of current clinical LLMs to UK EHRs and secondary mental health is much less known.

### 2.2.1 LLMs in EHRs and mental health

EHRs contain a longitudinal series of electronically recorded patient information (*60*). Unstructured free-text notes within EHRs have been challenging to process automatically due to the complex language used, including regional/speciality-specific nomenclature, overuse of abbreviations, and lack of grammar rules (*1*, *10*, *60*, *61*). In addition, the clinical text is large in volume, contains redundant information, and may contain irrelevant information. These issues are compounded by differences between healthcare institutions and countries, such as diagnostic ontologies (e.g., ICD-9/10 in the US vs. UMLS/SNOMED-CT in the UK) and medication naming conventions (*62*, *63*). Consequently, training NLP models or LLMs on US clinical text may not directly apply to UK data without domain adaptation.

Due to the noisy nature of clinical text, rule-based information extraction techniques were better suited for specific applications, such as extracting medical concepts from ontologies like Unified Medical Language System (UMLS) (*64*, *65*), or symptom extraction for depression and suicide ideation (*66*). A prominent UK study from the "Cogstack" team, developed a medical concept annotation tool for UMLS/SNOMED-CT ontologies called MedCAT (*64*). MedCAT was trained using annotated EHR data for extraction of UMLS concepts, linking words and text spans to these target concepts and codes with a high level of performance in their dataset. Another study, Med-7 (*65*) developed a Convolutional Neural Network (CNN)-based IE model using the NLP library SpaCy (*67*) with both the MIMIC-III dataset (*46*) and a subset of a UK NHS dataset. Med-7 could extract drug

names, routes of administration, frequency, dosage, strength, form, and duration, with decent performance on unseen test sets.

Now similar works have begun to use LLMs instead, with varying degrees of success. For example, MedTem utilised different neural language models, including a BERT-based one, to extract medication and temporal event relations from clinical text (*68*). The findings of that study did not show a major benefit to using BERT above a simpler Bi-LSTM model. Another study used BioBERT on three text mining tasks (*69*), achieving SoTA in some cases. Kulkarni et al. (2024) (*70*) used a BERT model trained on a private EHR dataset to achieve the extraction of 6 depressive symptoms from clinical text. They found their domain-adapted BERT model outperformed the base version, highlighting the impact of the domain adaption. More recently, several papers have investigated the clinical information extraction capabilities of recent LLMs such as GPT-3/4 (*5*). However, the findings are quite mixed. For example, one study reports superior performance using GPT-3 above BioClinicalBERT on various clinical NLP tasks, including NER, clinical sense disambiguation and medical event classification (*71*). However, other studies report an inferior performance of GPT-3/4 on related or similar clinical NLP tasks; Hu et al. (*72*) used GPT-4 to extract medical problems, treatments, and tests from clinical notes and found that it was beaten by a fine-tuned BERT model on the same tasks.

Other important clinical NLP tasks for LLMs relate to the embedding representation and sequence classification of whole clinical documents. For example, predicting different labels or clinical outcomes from admission and discharge notes: ICD-9 diagnosis codes (*73*, *74*), readmission risk of patients (*48*), length of stay and mor-

tality (*57*). One study also combined structured and unstructured EHR data using LLM and improved performance over either modality alone in three tasks: 30-day readmission prediction, ICD-9 diagnosis code classification, and medication recommendations (*75*). Other work has attempted to generate patient embeddings by modelling a patient's entire visit as a sequence of temporal events (*76*). The resulting model, called Deep Patient, extracts features of the EHR into structured formats, such as ICD-9 codes and medications, and represents patients as a timeline. Another study used raw clinical note texts to create a vector representation (embedding) in what they called Doc2Vec (*77*), which could be used as features for classification models. One study used an LLM with adaptations for time components, named Time-aware patient EHR representation (TAPER), to create embeddings for individual patients based on their corresponding notes (*78*). TAPER proved effective in clinical outcome prediction, surpassing the performance of baseline models with no time component. An impressive study using UK NHS data appeared in 2022 from the same Cogstack group that introduced MedCAT mentioned above, where they used MedCAT to produce a temporal sequence of structured information from the raw text and then fed this as a new sequence to a transformer-based LM architecture (*79*). They named their approach "Foresight" and used the model to predict future procedures and disorders based on the preceding timeline of extracted information. One major caveat with the foresight work is the reliance on their MedCAT tool and the restructuring of unstructured data, rather than ingesting and representing raw clinical notes directly - which, whilst powerful, does limit the potential of this approach to specific use cases.

LLMs are emerging as a standard tool in clinical NLP, with EHR data becoming

an increasingly sought-after target. To that end, there are now several EHR pre-trained LLMs that use structured data (lab results, age, diagnosis codes, etc.), such as BEHRT (*80*), MedBERT (*81*) and EHR-BERT (*82*), but unstructured text data remained less utilised in training these EHR focused LLMs (*60*). Much of the discussed work has focused mainly on physical health and the number of available EHR datasets to the research community is limited, with one dominant US-based dataset providing for most of the cited works - The Medical Information Mart for Intensive Care III (MIMIC-III) (*83*) (outlined in detail in my preliminaries Section 2.5.1.1). For my work, the major focus is the applicability and use of LLMs for mental health and psychiatry (discussed further below in Section 2.4) - where it has been shown that there is a lack of research utilising LLMs in place of traditional ML models (*84*).

In attempts to address the lack of mental health-focused LLMs, a team at the University of Manchester developed: MentalLlama (initialised from Llama-2-7b LLM from Meta (*3*)) and MentalBERT (initialised from BERT (*2*))(*85*). These LLMs were fine-tuned on social media datasets (Reddit and Twitter/X) where certain mental health topics were discussed. The objective was to derive a question-answer or chat-bot style LLM for mental health topics, which could then be used for the detection of specific disorders in the social media posts themselves. The authors did make a great effort to clinically validate aspects of the data through expert review. A similar achievement was also produced by a UK-based NLP group that developed a GPT-style LLM using NHS disease definitions named OpenGPT (*86*). The main caveat in these works is the data does not reflect real clinical data contained within EHRs. The direct application of LLMs in mental health, and

especially EHR data is very much in its infancy.

Aligning LLMs with NHS EHR datasets is non-trivial and carries risks. A driving force behind the sparsity of open LLM research with mental health EHR data is the difficulty in accessing the data, pairing it with powerful computing resources, and the inability to share these datasets or trained models publicly. Patient-level data is incredibly sensitive, requiring strict data governance and confidentiality practices. LLMs are trained to capture the structure, syntax, and semantics of the written language, leading to a form of memory of the training data. Probing refers to attempts to reveal information encoded by the LLM during training and can be achieved by re-using the original format of the pre-training objective. For example, as the LLM was trained to predict the upcoming or missing words in a sequence, given a sentence such as "The patient named ... in Oxford presented with symptoms of...", the LLM could correctly predict the gaps with the real training data. Research has shown LLMs are prone to adversarial attacks, revealing sensitive patient information through probing (*87*) or extracting training data through queries (*88*). Adversarial attacks can also disrupt LLMs' capabilities or remove safeguard training, with implications for patient safety if these trained LLMs are used (*89*). While much of this LLM safety research has focused on generative GPT-style LLMs, the concern is that any LLM can retain and make accessible training data. As I am exploring the use of LLMs with very sensitive patient-level data, these concerns are important to consider, especially for any deployment or sharing of these LLMs in a real-world setting. A full exploration of the de-identification of data and privacy concerns with LLMs is beyond the scope of the thesis. Still, I will discuss the implications of this in my final chapter.

## 2.3 Efficiency of using LLMs

An important aspect that I had to consider during my work was the efficiency in terms of computing resources required to train and use LLMs, for both practical and ethical reasons associated with a public entity such as the NHS. The practical reasons relate to restrictions imposed by working in TREs with potentially identifiable patient data, with limited resources available. Ethical reasons pertain to the environmental impact of using LLMs, the safety of patient data, and the possible risks associated with using AI for clinical decision support (I will discuss these issues in greater detail in Chapter 6).

When working with large neural networks, such as LLMs, one can expect to need large suites of GPUs and many of the most cost-effective solutions are cloud computing or API services (*90*). The acquisition of large suites of GPUs to enable both training and deployment of LLMs is not cheap, financially or concerning energy use. Smaller LLMs, like GPT-2 (*28*) cost around 0.0011 US cents per response on a suitably large NVIDIA A100 GPU but the larger GPT-3 (*30*) through OpenAI's API service is around 1.10 cents per example (*91*)[4]. It is estimated to cost OpenAI $\$700,000$ per day to run ChatGPT (*92*). To run reasonably sized LLMs locally is also a considerable cost, with appropriate GPUs costing several thousand British pounds each (a popular NVIDIA A100 with 80 GB VRAM can cost approximately £40 000 each from the University of Oxford suppliers (*93*). Keep in mind these costs are associated with just using the LLMs for inference. The initial pre-training costs of LLMs are considerably higher than inference, with the

[4]The pricing of OpenAI's services may have changed since the time of writing

estimated costs of training LLMs like BLOOM (176 Billion parameter LLM) in the order of millions of dollars over months of compute time (*94*). Fortunately, this pre-training cost is typically only incurred once. Although, as I will discuss in detail later, LLMs require some form of continued pre-training or fine-tuning to align with certain domains and tasks. This cost problem is of utmost importance when considering LLMs for public sector applications, such as the NHS, which has tight budgetary controls and where funding for speculative AI tools is limited (*95*). For technology such as LLMs to be useful in healthcare settings, we must understand the burden associated with using them. It is unclear how well LLMs address real-world problems, and careful thought is required to determine if LLMs are worth the cost. If the answer is indeed that they are, then in their present form LLMs are very difficult to scale to large organizations on a budget.

### 2.3.1 Scale of LLMs

The relative cost of different LLMs differs drastically and correlates directly with their size. The size of a transformer-based LLM equates to the number of parameters, consisting of the weight and bias values that comprise the transformer architecture's various components. As a baseline we can use the early and small BERT LLM (base version released in 2019) (*2*), which has approximately 108 *million* model parameters and amounts to approximately 1.3 GigaBytes (GB) of physical disk space in full 32-bit precision. We now have much larger LLMs such as Llama-2-70b (released in 2023) (*3*), which has approximately 70 billion model parameters, amounting to over 200 GB to store. Beyond this we have LLMs with considerably more parameters: PaLM (released in 2022) (*13*) from Google with

540 billion parameters, and estimates of closed source LLMs such as GPT-4 having over 1 *trillion* parameters (although this model is closed source, and we do not know its exact size). The increase in size was driven by scaling laws, whereby simply increasing the size of the LLM models and training datasets predictably improves performance on NLP tasks (*14*, *96*).

The larger sizes of LLMs were typically produced outside of academia, and many recognised how infeasible it is to operate with such large models, especially outside commercial settings. There has been a push for improving access to LLMs by reducing model size. For instance, Phi-2 from Microsoft has approximately 2.7 billion parameters, but through training on more data and for longer, yielded performance comparable to much larger models (*37*). Or with sparse Mixture of Expert (MoE) models, such as Mixtral (*31*) and the switch transformer (*97*), using gating and router mechanisms to selectively pick a subset of the full model's parameters (the experts) for each input. This allows a massive model size but with a constant compute through only a portion of model parameters used for each input. It is now common to see people distinguish Small Language Models (SLMs) and LLMs. Ultimately these are all LLMs but due to the vast differences in model sizes present today, the field has begun to separate LLMs based on size - although there is no agreed-upon size. This can appear confusing and for clarity, I will almost always use the term LLM to refer to any transformer-based language model, regardless of the size, as I believe this will make it easier to follow and link to more recent research.

### 2.3.2 Difficulties in adapting LLMs

As discussed earlier in section 2.2, developing viable LLMs for specialist text typically requires continued training through transfer learning and domain adaptation (see section 2.5.5). Moreover, to then adapt any LLM to a specific downstream task, such as sequence classification, requires further fine-tuning of the LLM model parameters through standard neural network training. With the increasing scale of these LLMs, this workflow becomes incredibly expensive and essentially impractical as it requires large suites of GPU to perform any model training (*12*, *98*), or even just to load the model for inference. Traditionally one would adapt a model to a new domain or task by updating all model parameters with respect to a new loss function, which would effectively create a whole new LLM. This can cause catastrophic forgetting, whereby training on new domains and tasks leads to forgetting or diminished performance on the previous training domain/data. This can render an LLM unable to perform well on the original LM objective, and thus inhibit the ability to re-use the LLM for other domains or tasks. A problem still prevalent with LLMs today.

#### 2.3.2.1 Efficient domain adaptation

A recent review explored the literature on domain adaption of NLP and LLMs for EHR data and found that whilst models such as BioClinicalBERT (*44*) showed improvements upon their generally trained base model, they do not always transfer well to specific healthcare datasets (*84*). The cited reason is the major differences in writing style, language, and jargon used in healthcare datasets (*99*), as discussed

earlier in Section 2.2.

Several studies have sought to alter the pre-training stage of biomedical or clinical LLMs to better align with the domain and common NLP tasks of interest. For instance, BioLinkBERT (*51*) combined the standard MLM objective with a novel citation link classification task during pre-training. This citation link task was derived from hyperlinked PubMed articles and based on whether one article cited another. The idea behind this link was to capture knowledge that spans across different documents, almost a proxy for a knowledge graph or database. The presumption was that research articles would cite relevant literature and the topics of interest would be related and encourage the model to capture this in some form. The same authors released another model, Deep Bidirectional Language-Knowledge Graph Pre-training (DRAGON) (*52*), which utilised a knowledge graph (KG) derived from the UMLS to create a KG representation of the input text itself, as paired input with the raw text. The main goal of these works was to enhance the pre-training stage to embed additional knowledge or structural information in the LLM to align better with the desired clinical domain.

The recurring gap in this research area is the application to UK datasets, and especially EHR datasets in mental health. In Chapter 3, I explore alternative methods for improving the pre-training stage to improve the ability to adapt LLMs to healthcare datasets, and especially NHS text, with subsequent improvements to downstream tasks and applications.

#### 2.3.2.2 Efficient downstream adaptation

The problems of further adapting LLMs to specific tasks are quite similar to those aligned for domain adaptation, with the added issue of altering the loss objective entirely. For example, with domain adaptation one typically seeks to maintain the language modelling objective used during the original pre-training phase, whereas many downstream tasks will require a different format, as discussed below:

**Traditional fine-tuning and prompt learning** As discussed earlier in Section 2.1, adapting LLMs to a new NLP task, such as sequence classification (see Section 2.5.7), involves a fine-tuning process (explained in more detail in Section 2.5.5). Fine-tuning of LLMs for downstream tasks often results in SoTA performance on many datasets and benchmarks (*17*). Different adaptations of fine-tuning have appeared, such as multi-task training, where several tasks are trained in parallel (*100*), or intermittent fine-tuning where a model is first trained on a similar task before being trained on the target task (*101*). However, they still follow the process of using the base LLM to produce contextualised representations of the text as features for a classifier on top. Researchers recognised that forming the tasks in such a way deviates from the original pre-training objective LLMs were trained to do - which means the newly fine-tuned model can no longer act as a base LLM for other tasks.

A major shift for autoregressive LMs was introduced by Brown et al.'s 2020 GPT-3 paper (*102*), where all tasks were treated as text-to-text tasks. Instead of using LLMs as encoders producing text embeddings for classification heads, the task was embedded into natural language. For example, sentiment analysis on "That movie

was wonderful, and the casting was perfect!" would traditionally fine-tune the LLM to produce a sentence embedding, passed to a classification head outputting positive, neutral, or negative. Brown et al. altered the input to "That movie was wonderful, and the casting was perfect! What is the sentiment of this movie review? Given the following response choices. Positive, neutral or negative". The LLM's objective remained unchanged - predict the next words. In effect, the LLM is instructed to complete the task.

Originally, Brown et al. did not perform model updates, instead relying on few-shot in-context learning by providing completed examples as initial input and appending an uncompleted instance. However, in-context learning's performance was typically inferior to fine-tuning and unstable based on examples and prompts used (prompt engineering) (*4*, *102*). The problem with fine-tuning is that with such large LLMs, the resources required are substantial and incur problems of catastrophic forgetting.

**Parameter Efficient Fine-tuning and Prompt learning** To reduce the need to fully fine-tune LLMs for new tasks, solutions have emerged that reduce the number of LLM parameters updated during training, reducing computational resources required - an area referred to as Parameter-Efficient Fine-Tuning (PEFT). One early approach was lightweight fine-tuning, which froze different LLM layers or modules during fine-tuning to preserve core knowledge while only fine-tuning later task-specific layers (*103*). The difficulty arose in selecting which sections to freeze. Pfeiffer et al. (*104*) introduced adapters - small trainable parameters trained on a specific task and fused into the main LLM, reducing trainable parameters

by 95% while approaching full fine-tuning performance. Subsequent works combined prompting with the adapter notion, allowing greater flexibility. Prefix tuning (*105*) introduced trainable vectors alongside input text at every transformer layer. Prompt tuning (*4*) replaced prompt tokens with trainable "soft prompts" prepended/appended to the input. Concurrently, Low-Rank Adaptation (LoRA) (*106*) represented weight updates as low-rank matrices, constraining updates to a low-dimensional subspace. LoRA approximated full fine-tuning weight updates while retaining comparable performance on NLP benchmarks. Initially quiet in 2022, LoRA has since become popular for fine-tuning larger LLMs and will be discussed in detail in Chapter 4.

**Quantisation and precision reduction** Additional strategies exist to address the issue of model size and fine-tuning. Pruning redundant weights for given downstream tasks has been effective in certain cases (*107*). A more dominant recent approach has been reducing model size in terms of floating-point precision, bits, and physical memory needed to store weights through quantization (*108*, *109*). Quantization reduces the precision of model weights by compressing standard high-precision 32-bit floating-point numbers (typical of neural network frameworks like PyTorch (*110*)) to lower precision formats like 16-bit float, 8-bit integers, or even 1-bit tertiary values (*111*). Bits are the basic units of digital information, with only two possible values (0 or 1), used to represent all digital data, including floating-point numbers. While reducing bits reduces the amount of information stored, potentially truncating or compressing the language representation learned during training, it leads to performance degradation. However, the major benefit is reduced computation requirements; converting from 32-bit to 16-bit floating-point

numbers effectively halves the memory and GPU requirements.

The efficient downstream adaptation is the major focus of Chapter 4, where methods will be explained and explored in great detail.

## 2.4 Psychiatry - secondary mental health triage

With the development of LLMs and their ability to adapt to new domains, it is possible to utilize these models for specific use cases in mental health research and assist clinicians through digital support applications. Psychiatry in the UK is a medical speciality focused on the diagnosis, treatment, and prevention of mental health disorders. As a field of study, it combines aspects of medicine, psychology, neuroscience, and social sciences. Mental health problems are a massive contributor to the Global Burden of Disease (GBD), which has been steadily increasing over the past 30 years (*112*). The study and treatment of mental health disorders is complex and long underfunded. The advancement of ML and AI has the potential to improve the speed of research, better understand and potentially improve treatment outcomes for various mental health-related illnesses. As I will discuss throughout this work, within the UK, a large source of information related to an individual's mental health journey with healthcare providers is captured in text. A primary goal of this thesis was to investigate the use of LLMs in Psychiatry in the context of a particular component of mental health triage within the local NHS trust in Oxfordshire, detailed below.

In the UK, mental healthcare is stratified into primary (General Practice (GP)), secondary (community and hospital NHS Trusts), and tertiary services. Most

people (96%) requiring specialist (secondary) mental healthcare are referred to and treated by community mental health teams (CMHTs) (*113*). This referral and triage process is time-consuming, prone to subjective interpretation, and often repeated for the same patient across different teams. Assisted triage, where AI is deployed, could enhance the efficiency of extracting relevant clinical data and assist in allocating and justifying triage decision-making.

The majority of available information related to the referral and triage process is contained within written documents, typically collated in the NHS trust's respective EHR system. Collectively, this is a large pool of millions of individual documents related to hundreds of thousands of patients and written by various practitioners involved in the referral pipeline. The ultimate objective of clinical NLP and LLMs is to facilitate and potentially automate the use of these texts, for use cases such as triage decision support.

### 2.4.1 Representing clinical notes with LLMs

As discussed in Section 2.2, EHR data can be viewed as a clinical information sequence. The referral process relies on clinician collaboration, with the primary patient information source being the EHR's unstructured clinical text. A single patient can have hundreds of associated clinical notes, which clinicians must manually sift through to find critical information. Developing NLP methods to assist has been a prominent objective, traditionally relying on labour-intensive annotations and information extraction pipelines. Previous work like MedCAT, MedGPT (*114*), and Foresight (*79*) showcased how NLP can enhance clinical note representation. Foresight used the MedCAT information extraction and annotation tool to

transform unstructured text into a structured form. A patient's history was effectively transformed into a sequence of symptoms, medications, and treatments to predict patient trajectories. A powerful tool, but requires bespoke tools and large annotated data volumes, primarily focusing on isolated NHS trusts, excluding the NHS OHFT dataset used in this work.

Direct LLM use for mental health research is rare and non-existent for the outlined triage problem. Recent healthcare LLM work has focused on chat APIs like GPT-4 (*5*) and Claude-2/3 (*7*) for question-answering systems or chatbots for well-being and psycho-education (*115*). One study explored GPT-4 (*5*) and Llama-2 (*3*) for clinical decision support similar to triage but found diagnostic performance relatively poor and far from safe for use (*116*). The major issue is the misinterpretation of LLMs like GPT-4 outputs as factual statements, due to their training to act as incredibly helpful and confident chat-bots. This means these models would likely provide a clinical opinion, and regardless of whether or not it is correct, appear entirely confident about it. Therefore, determining the truthfulness of LLMs still requires a domain expert to review the response. Overall, the application of larger, recent LLMs to secondary mental health remains difficult, primarily due to data alignment issues and their recentness.

### 2.4.2 Developing a triage support model

The use of ML and AI for triage support in the health domain is active but traditionally has focussed on emergency departments and physical health. For example, one study utilised different binary classification models (admitted or discharged), such as logistic regression, to predict hospital admission at emergency department

triage (*117*). Across three datasets, totalling approximately half a million unique patient visits, their models could classify admissions with an AUC of 0.9 using a mixture of historical and triage data. A large portion of admissions to ED are related to cardiovascular and respiratory disease, and in their work, they showed the health measures related to cardiovascular health were the most influential predictors. A recent systematic review found across eleven studies, ML models could reasonably predict outcomes in emergency health settings, including; mortality, admissions, and care outcomes (*118*). These studies highlight the potential benefits of integrating ML into the triage setting, although the type of data available for mental health triage differs dramatically from that for emergency departments.

The application of AI to mental health triage is limited, and the use of LLMs is virtually non-existent. One study used a "cascading classifier" with individual random forest models per triage category as a screening tool in initial psychiatric assessments, using manually extracted features from Initial Psychiatry Evaluation (IPE) documents (*119*). While highlighting the viability of machine learning for triage, the study had limitations such as a relatively small dataset, organized IPE documents, and reliance on handcrafted feature extraction rather than raw unstructured data. Whilst not triage specifically, recent work from a London-based NHS research group investigated the identification of mentions of pain within text using a fine-tuned BERT model. They found that using a domain-specific BERT model outperformed simpler machine learning models substantially (*120*). This type of research highlights the potential to use LLMs to aid in UK clinical NLP tasks, which in turn can influence work on triage applications. Commercial medical technology companies like Holmusk are targeting mental health EHR data for

triage purposes using LLMs, recently using a BERT model with a contrastive loss function to identify Major Depressive Disorder (MDD) symptoms (*70*). This approach aligns well with the pre-training method developed in Chapter 3. The use of ML and AI models like LLMs for mental health triage is relatively new, with unclear real-world usefulness (*121*). Studies on pre-primary care triage errors found digital symptom checkers prone to errors, with accuracy decreasing for urgent symptoms (*122*). The Australian Project Synergy initiative faced complications, with only 1 out of 500,000 potential users deemed eligible (*123*), highlighting the challenges of translating AI models to real-world settings. A 2023 narrative review concluded that much of the research has described potential benefits to the use of mental health triage tools, but there is limited evidence of their use (*124*).

While not attempting to create a full prototype digital triage tool during this DPhil, I explored LLMs' ability to process clinical notes and assist triage referrals. I will discuss recommendations and future steps to align this work with real-world settings and clinical practices.

## 2.5 Preliminaries

### 2.5.1 Datasets

Throughout my DPhil projects, I utilised three main clinical or healthcare datasets with varying degrees of access, detailed below:

#### 2.5.1.1 MIMIC-III

MIMIC-III (*83*), was developed by the MIT Lab for Computational Physiology, and is a de-identified EHR dataset associated with 38,597 critical care patients and 58,976 intensive care unit (ICU) admissions at the Beth Israel Deaconess Medical Center between 2001 and 2012. Data includes demographics, vital signs, laboratory tests, medications, caregiver notes, imaging reports, and mortality. MIMIC-III is the most commonly used EHR dataset in clinical NLP due to its relative ease of access, high standard, and active releases. An important categorical feature available is the International Classification of Diseases (ICD-9) diagnosis and procedure codes. These are used to assign certain health symptoms and disorders to a unified classification system, or ontology. MIMIC-III at the time of writing had data linked by the ICD-9 system, which is now outdated. Now, many EHR systems utilise ICD-10 or ICD-11 coding (*125*, *126*), which are quite different in scope, with a larger set of unique codes. This means the work presented throughout this thesis utilising MIMIC-III data, may not align well with the more recently adopted ICD classifications.

It is accessible to researchers via a data agreement and relevant governance training obtained through the data providers (*46*). De-identification utilises structured data to remove all eighteen patient identifiers listed by the US Health Insurance Portability and Accountability Act (HIPAA).

#### 2.5.1.2 NHS Patient Safety Incident Reports - PSIR

The NHS England National Reporting and Learning System (NRLS) hold annually collected data related to country-wide patient safety incident reports (PSIR) (*127*) that occur in different care settings. These data include categorical features such as incident type and location, alongside the free-text document describing the incident in detail. A subsample of approximately 2.3 million de-identified reports was produced in the financial year 2019/2020. The NHS PSIR dataset is not strictly an EHR dataset and does not relate to patient-level care specifically, so it should not contain sensitive patient data. However, it is not an open dataset and was only accessible during an internship with NHS England (the former NHS-X branch) between June and November 2022.

#### 2.5.1.3 NHS Oxford Health Foundation Trust (OHFT)

NHS Oxford Health Foundation Trust (OHFT) is the local secondary mental health provider, covering Oxfordshire and Buckinghamshire's population of around 1.2 million people. From OHFT's EHR, access was granted to historical data for approximately 200 thousand patients spanning over a decade, with a total of around 8 million de-identified, pseudonymized clinical notes. The data from OHFT are de-identified and obtained through the Clinical Record Interactive Search (CRIS) system powered by AkriviaHealth (*128*), which provides a secure data environment and platform for processing de-identified clinical case notes. Access to and use of these de-identified patient records from the CRIS platform have been granted an exemption by the NHS Health Research Authority (March 2020) for research reuse

of routinely collected clinical data. The Oversight Committee of OHFT and the Research and Development Team reviewed and approved the project in November 2022.

## 2.5.2 Deep learning and LLMs

### 2.5.2.1 Key concepts

Below are several key terms and concepts in deep learning that will be useful to know when reading this thesis:

- **Tensor** - A generalization of matrices to an arbitrary number of dimensions. It is a multi-dimensional array of numerical values. Scalars (single numbers) can be thought of as 0-dimensional tensors, while vectors are 1-dimensional tensors, and matrices are 2-dimensional tensors.
- **Batch size** - The number of samples that are propagated through the neural network at one time before updating the model weights. A smaller batch size allows for more frequent updates to the model. This has important implications for the loss function and GPU memory consumption.
- **Epoch** - One full cycle through the entire training dataset. Generally, training is done over multiple epochs, as the model weights are updated after each epoch until the model converges or a set number of epochs is reached.
- **Learning rate** - A configurable hyperparameter that controls how much the weights of the neural network are updated during training after each

batch and optimization step. Higher learning rates cause larger weight updates, which can lead to faster convergence but potentially overshoot optimal weights.

- **Optimizer** - An algorithm that updates the weights of the neural network based on the computed gradients from the loss function during batched training. The optimizer determines exactly how the gradients are used to update the weights at each optimization step. Common optimizers include stochastic gradient descent (SGD), Adam (*129*), and Adam with Weight decay (AdamW) (*130*). They often help accelerate and stabilize the learning process compared to using only basic gradient descent. For reference, I use AdamW throughout my experiments as it has been shown effective for training and fine-tuning LLMs (*35*).

- **Hyperparameter tuning** - Refers to the process of finding the best set of values for the various configuration settings (hyperparameters) of a deep learning model to achieve optimal performance on a given task. Hyperparameters are the settings or variables that are specified before the model training process begins and are not learned from the data during training. Examples of hyperparameters include the learning rate, batch size, number of layers, number of neurons per layer, dropout rate, regularization strength, and so on.

#### 2.5.2.2 Tokenization

As LLMs will be used throughout the experimental chapters, understanding how they operate is crucial. Text needs to be converted into numerical representations, known as embedding vectors for LLMs. Tokenization assigns unique identifiers to words or subwords from the training corpus. Different tokenization algorithms exist with varying granularity levels. While character-level tokenization is an option, word or subword-level tokens are generally better for language modelling. Whole word tokenizers assign unique labels to words split by whitespace but can lead to poor subword representation and many unknown tokens. The LLMs used throughout my work utilize Byte-Pair Encoding (BPE), a hybrid character-subword approach that reduces unknown tokens by iteratively splitting unknown words into known subword pieces (*131*). For example, consider the string: "thequickbrownfoxjumpsoverthelazydog".

Initial tokenization:

```
"thequickbrownfoxjumpsoverthelazydog"
```

First BPE iteration, merging frequent pair "the":

```
["th","e","q","u","i","c","k","b","r",...]
```

The algorithm iterates, merging frequent pairs until the desired token count. Final tokenized output:

```
["th", "e", "quick", "brown", "fox", "jump", "sove", "rthe",
"lazy", "dog"].
```

In summary, BPE iteratively merges frequent character pairs into tokens, controlling granularity by the number of merge operations.

### 2.5.3 Transformers

The dominant neural architecture in today's LLMs utilizes components from the transformer architecture (*26*), introduced in 2017 with over 100,000 citations by 2024. Its breakthrough was parallelizing sequential data processing. Unlike previous architectures like RNNs (*132*) and CNNs (*133*) that processed sequences step-by-step, the transformer uses attention to consider the entire input sequence simultaneously. It calculates attention weights for each input part based on relevance, directly modelling distant relationships without regard to sequence distance. Specifically, the transformer employs multi-headed self-attention layers. Each neuron attends to all others, its past state, and aggregated input context. Multiple attention weight sets (heads) focus on different patterns. Their outputs are aggregated and processed for prediction. By leveraging attention, transformers build contextual representations of full input sequences without recurrence. The attention weight patterns encode important global sequence relationships. A transformer model consists of a given number of stacked transformer layers sequentially (see Figure 1).

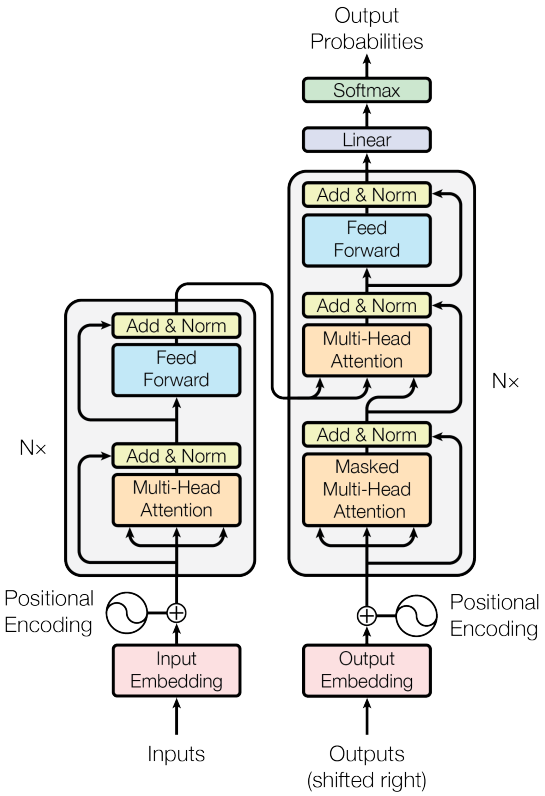


**Figure 1** Transformer architecture - adapted from (*26*). The original diagram shows a full encoder-decoder model architecture, with the *encoder* on the left in red, and the *decoder* on the right in blue.

#### 2.5.3.1 Embeddings

Perhaps the most fundamental component of LLMs is the word (token) embeddings, which are essentially the learned numerical representations of the text. Each word (or token more precisely) will be linked to its unique embedding, which is a $d_h$-dimensional vector, where $d_h$ can be any number, and as an example one of the most popular LLMs, BERT (*2*) uses an embedding dimension of 768. In simple terms, when sending input (text) to an LLM, it is first converted to a set of embeddings dependent on the word or token (and with one embedding per word/token). These embeddings are then passed through the LLM model as can be seen in Fig

1.

#### 2.5.3.2 Positional encoding

To represent the position of tokens within a sequence, *positional encodings (PE)* are created and combined with the token embeddings. These PEs are vectors of real numbers with the same dimension as the token embeddings, allowing them to be summed and match the sequence length. The encodings can be learned or predefined: in the original transformers paper (and often the case with other models since), sine and cosine functions of different frequencies were used:

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{\text{model}}})$$

where $pos$ is the position and $i$ is the dimension. The authors state these functions form wavelengths with a geometric progression dependent on the token embedding dimension ($i$) and the sequence position ($pos$). That is, each dimension of the positional encoding corresponds to a sinusoid. The wavelengths form a geometric progression from $2\pi$ to $10000 \cdot 2\pi$. They hypothesised it would allow the model to easily learn to attend by relative positions, since for any fixed offset $k$, $PE_{pos+k}$ can be represented as a linear function of $PE_{pos}$.

#### 2.5.3.3 Encoder and decoder

The original transformer paper (*26*) used an encoder-decoder configuration for language translation. As shown in Fig 1, it has $N$ transformer layers on the left (*encoder*) and right (*decoder*). The encoder generates a contextual representation of the input sequence, used by the decoder as context for producing the output sequence. The difference lies in the attention mechanism and which input parts can be attended to. The encoder calculates attention between all input positions, allowing each position to incorporate relevant context from the full input. The decoder originally used masked or causal attention, altering attention calculation to only produce scores for previous tokens, intending to predict the next output token.

#### 2.5.3.4 Attention

The attention mechanism in transformers involves three vectors calculated for each position in the input sequence (in the case of text, this would be every token in the input sequence):

- Key vector - Encodes the position's context to be compared to other positions
- Query vector - Encodes what context should be focused on to represent the position
- Value vector - Contains input information at the position to contribute to the representation

For each query position, attention weights are then calculated between its query vector and the key vectors of all positions, typically via the dot-product as the compatibility function. This determines the relevance of other positions to the query (*26*):

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{2.1}$$

Where:

$Q$ = matrix of query vectors for each position

$K$ = matrix of key vectors for each position

$V$ = matrix containing value vectors for each position

$d_k$ = dimensionality of the key vectors

The $softmax(Q * K^T)$ term calculates and normalizes the similarity between the query and each key to obtain attention weights. This determines which positions to pay attention to. The attention weights are then multiplied by $V$ to calculate a representation for the query position, which is a weighted summation of the value vectors from all input positions. This determines what context contributes most. By calculating attention in this way across all query positions, each position's representation captures the relevant global context from the entire sequence.

To increase the complexity of the model, and potentially the diversity of the representations learnt by the model, multiple attention heads are used, each contributing its attention-weighted output. These heads mirror one another with regard

to attention architecture but have different initialised weights. Multiple heads can provide multiple attention views, see Figure 2. The separate attention heads compute their output in parallel and the results are concatenated to form a final representation of each token that is passed forward to the next layer(s).

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Where the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$.

The authors argue that having different attention heads allows the model to learn different representations of each sub-space of the input sequence, whereas having only a single attention head would rely on averaging and inhibit this.

#### 2.5.3.5 Maximum sequence length

An important aspect of transformer attention is the impact of maximum token sequence length. Attention scales quadratically with sequence length, as attention scores need to be computed between every pair of positions, resulting in a

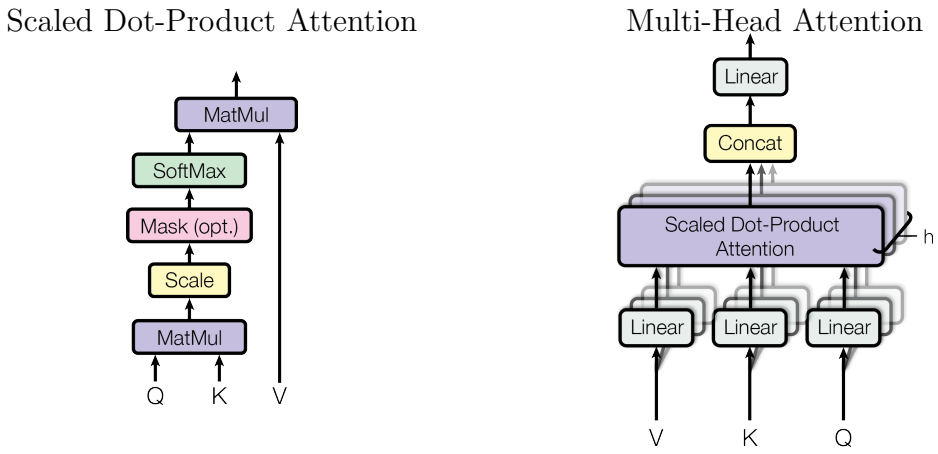


**Figure 2** (left) Scaled dot-product attention. (right) Multi-head attention consists of several attention layers running in parallel.

complexity of $\mathcal{O}(n^2 \cdot d)$, where:

$\mathcal{O} =$ big-O notation for time complexity.

$n =$ sequence length.

$d =$ dimensionality of embeddings (query, key, value vectors).

Practically, this limits standard transformer LLMs' ability to handle very long sequences due to increasing GPU memory requirements. Original LLMs like BERT (*2*) had a limited maximum sequence length of 512 tokens, while later LLMs like Llama-2-7b (*3*) have a maximum of 2048 tokens. Architectural advances improve LLMs' ability with long sequences, but I will not cover them in detail here as it would detract from the main goal. Select architectures for handling longer sequences will be mentioned in Chapter 5 when relevant to a specific use case.

## 2.5.4 Language modelling objectives

Two key language modelling objectives relevant to this thesis are:

### 2.5.4.1 Masked Language Modelling

MLM, introduced by BERT (*2*), is a "fill in the gap" task where parts of the input text are masked with a special token. The model predicts the masked token positions, with loss calculated on these positions, illustrated in Figure 3. RoBERTa

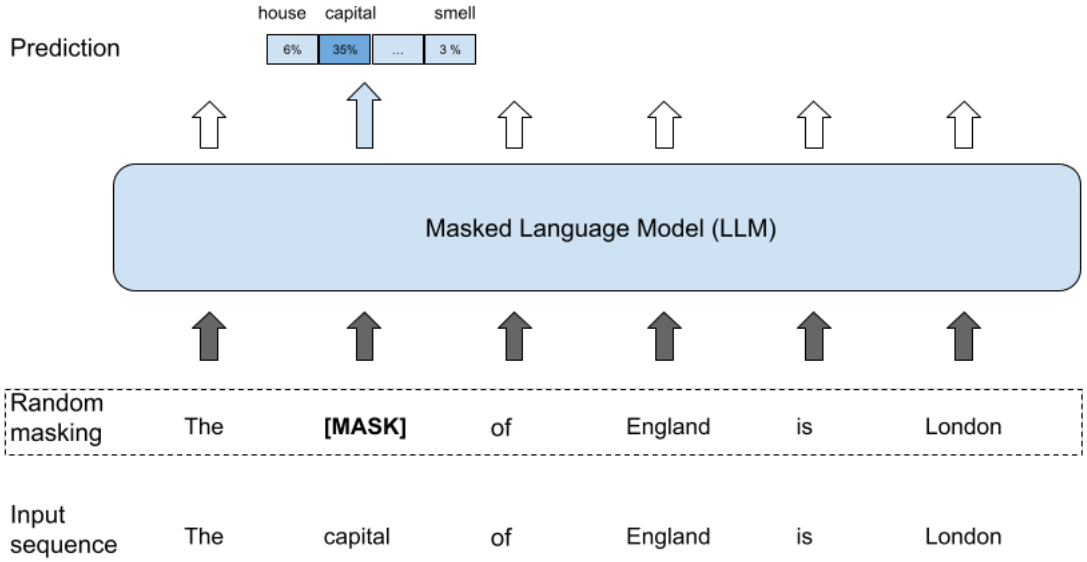


**Figure 3** Masked Language Modelling schematic.

(*134*) improved upon BERT by using dynamic masking during training and removing the next-sentence prediction task. RoBERTa is crucial for this work, often used as a base model. MLM LLMs excel at tasks like text classification, Named Entity Recognition, semantic search, and embedding extraction (detailed in Section 2.5.7).

#### 2.5.4.2 Autoregressive Language Modelling

ALM uses a decoder-only setup, where the attention mechanism only attends to previous positions, and the objective is to predict the next token in the sequence, as shown in Figure 4. Today, ALM models dominate generative AI, powering

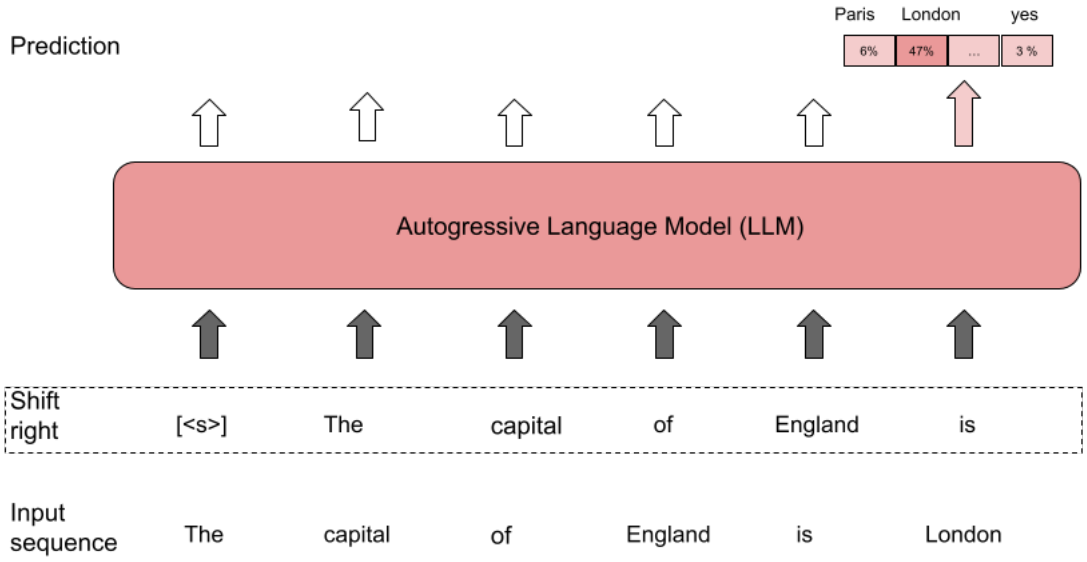


**Figure 4** Autoregressive Language Modelling schematic.

LLMs like GPT-3/4 (*5*) and Claude 2/3 (*7*), albeit with further training on the APIs.

### 2.5.5 Domain and task adaptation

To align LLMs with specialist domains like clinical or biomedical, domain adaptation can be employed. This involves either training an LLM from scratch on the target domain or continuing pre-training of an existing LLM on the domain of interest. The latter approach is more common, as training from scratch is ex-

pensive and limited by smaller domain-specific text availability (*8*). Continued pre-training uses the same language modelling objective as the base model (e.g., MLM) but initializes the model weights with those from the original pre-training, as shown in Figure 5.

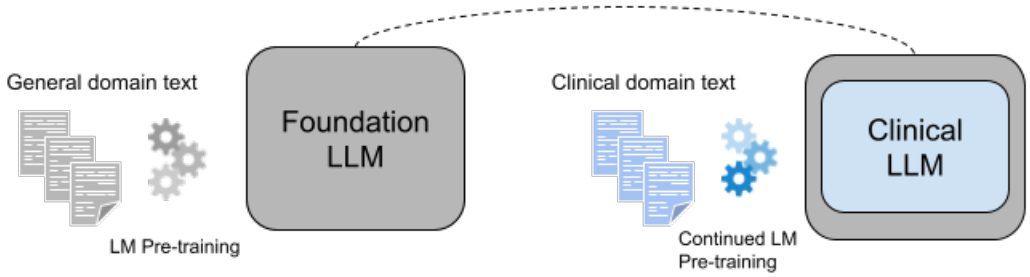


**Figure 5** Domain adaptation schematic.

This continued pre-training approach for domain transfer is vital to the work presented in this thesis and enabled the creation of different NHS-trained LLMs. After pre-training, an LLM can be used as a base model for various downstream NLP tasks, such as Named Entity Recognition (NER), text classification, sentiment analysis, information extraction, translation, summarization, topic modelling, and generation. Traditionally, adapting the LLM to a new task required further training with additional model parameters and a new loss function. For example, in text classification (e.g., classifying a film review as positive or negative), the LLM can be used to produce a numerical representation of the text, which is fed into additional neural network layers or other machine learning models, as shown in Figure 2.5.5.

As adapting LLMs to downstream sequence classification tasks constitutes the majority of my experiments, I will explain the traditional fine-tuning approach for

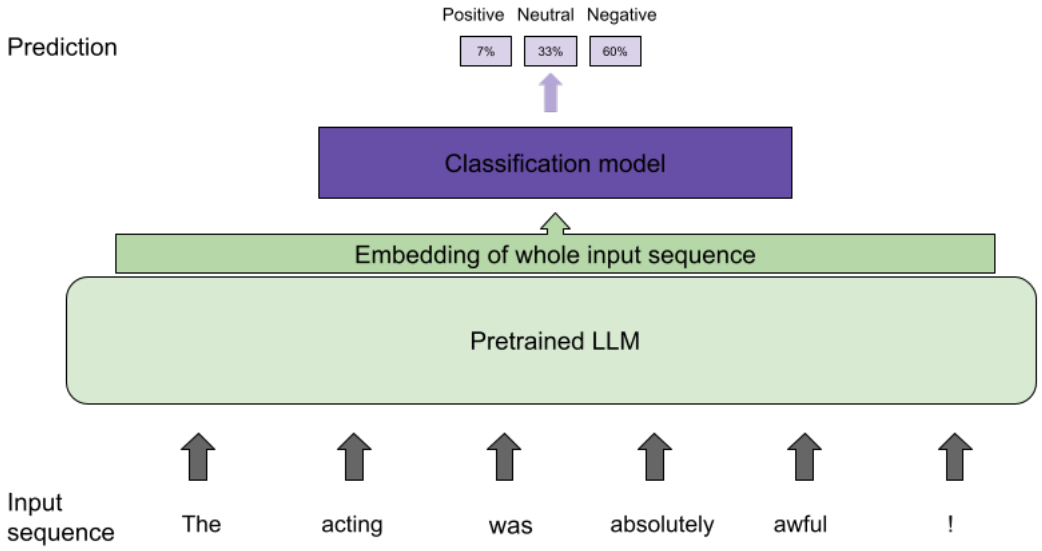


**Figure 6** Downstream adaptation of LLM for a text classification task.

this task type in more formal detail here, which subsequent experimental chapters will refer back to.

Fine-tuning for sequence classification involves introducing additional layers on top of the LLM in the form of a classification head. The LLM is fed with an input sample $\boldsymbol{x}$, which consists of a sequence of $n_w$ words converted to tokens using the LLMs tokenizer which has a fixed vocabulary $\mathcal{V}$. Mathematically, we say that $\boldsymbol{x} \in \mathcal{V}^{n_v}$, where $n_v$ is the total number of unique tokens.

This LLM, the encoder, $f_{enc}(\cdot)$ then transforms the input $\boldsymbol{x}$ into a list of embeddings $\mathbf{E}_i$, one per token $i$. Each embedding is a numerical representation of the corresponding token, and it consists of a $d_h$-dimensional vector:

$$f_{enc}(\boldsymbol{x}) = [\mathbf{E}_1, \mathbf{E}_2, ..., \mathbf{E}_{nw}] \tag{2.2}$$

where $f_{enc}(\cdot) : \mathcal{V}^w \rightarrow \mathbb{R}^{n_w} \times d_h$, $\mathbf{E}_i \in \mathbb{R}^{d_h}$ for all $i$, and [...] represents concatenation of vectors.

Next, the output of the LLM is fed to a pooling function $g(\cdot)$. Typically, this is the mean, which averages all the embeddings along the word dimension to obtain a single vector representation $\mathbf{H}$ of the whole input[5]:

$$\mathbf{H} = g\left([\mathbf{E}_1, \mathbf{E}_2, ..., \mathbf{E}_w]\right) \tag{2.3}$$

where $g(\cdot) : \mathbb{R}^{n_w \times d_h} \rightarrow \mathbb{R}^{d_h}$, and $\mathbf{H} \in \mathbb{R}^{d_h}$.

Finally, the output of the pooling function is fed to the classification head $f_{head}(\cdot)$, which has the task of calculating the logits $y_j$ of each of the possible $c$ classes $j \in \mathbb{C}$. A softmax operation is applied to the logits to produce a normalized probability score that $\boldsymbol{x}$ belongs to each of $c$ the possible classes. Mathematically, for one sample with vector representation $\mathbf{H}$, the probability of the sample belonging to class $j$ is:

$$\begin{aligned} f_{head}(\mathbf{H}) &= [y_1, y_2, ..., y_c]\,, \\ P(j) &= softmax([y_1, y_2, ..., y_c]) = \frac{exp(y_j)}{\sum_{k=1}^{c} exp(y_k)} \end{aligned}$$

where $f_{head}(\cdot) : \mathbb{R}^{d_h} \rightarrow \mathbb{R}^c$, and $y_j \in \mathbb{R}$ for all $j$. Represented diagrammatically in Fig. 7.

In this example, the sentence *"Patient is complaining of severe chest pain"* is fed

[5] different methods for obtaining a single vector representation of a sequence exist

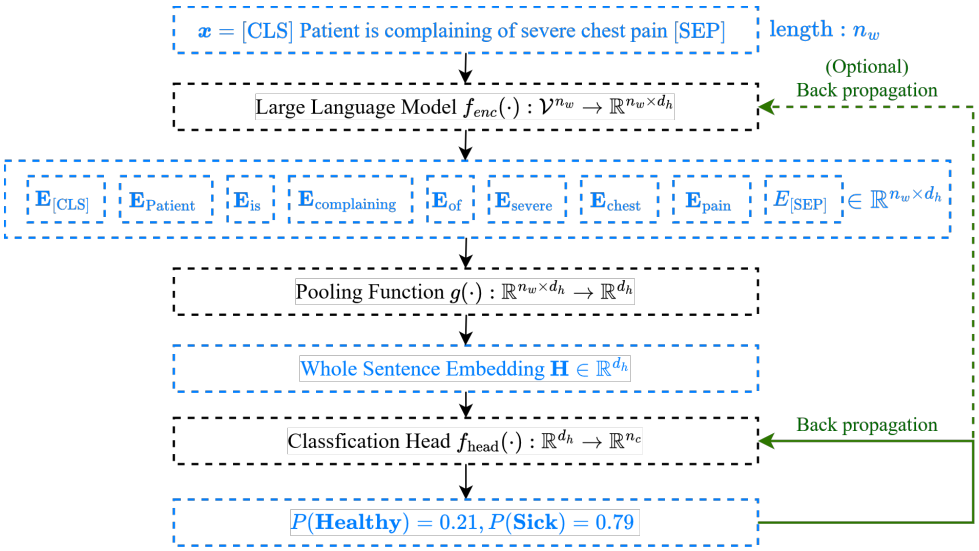


**Figure 7** Illustration of the traditional fine-tuning method with a LLM + classification head. The option to freeze the LLM is shown in dotted lines. Here [CLS] and [SEP] tokens are special tokens for BERT-based models added to the beginning and end of sequences, respectively.

to the LLM, with an embedding representation produced for each of the individual words, which are then pooled via an averaging function to produce a single embedding of the whole sentence. This sentence embedding is then used as the feature for the classification head and an output decision of *"Healthy"* or *"Sick"* is produced. This approach can be applied to any document and label pair and remains the standard approach for using LLMs for sequence classification.

### 2.5.6 Learning paradigms

All of the statistical, machine and deep learning models attempt to provide a model of an underlying data distribution and interactions between variables. To automatically learn from data, modern machine and deep learning techniques use unsupervised and supervised learning.

#### 2.5.6.1 Supervised learning

Supervised learning aims to learn a mapping from input data to output labels using labelled training examples (*135*). The algorithm adjusts model parameters to minimize a loss function, applicable in regression (predicting continuous values, such as blood biomarkers) and classification (assigning categorical labels, such as the presence of pneumonia in lung x-ray images) tasks.

#### 2.5.6.2 Unsupervised learning

Unsupervised learning focuses on finding patterns and structures within unlabelled data. Unlike supervised learning, where the goal is to learn a mapping from input data to labelled outputs, unsupervised learning algorithms aim to discover inherent relationships, similarities, or representations within the data without relying on predetermined labels or targets (*136*). Common unsupervised learning tasks are clustering, and dimensionality reduction (e.g. Principal Components Analysis PCA)).

#### 2.5.6.3 N-shot learning paradigm

N-shot learning includes zero-shot learning (ZSL) and few-shot learning (FSL), which train models with limited or no samples. ZSL tests models without prior training, while FSL uses a limited number of training samples. These approaches are valuable in clinical fields where large labelled datasets are challenging to obtain (*137*). ZSL is increasingly used with large foundational LLMs for generalization to unseen tasks, while FSL is crucial for building bespoke, targeted models with

limited resources and annotated data.

FSL is crucial for my work, where I am trying to build bespoke NHS trust or location-specific models with limited resources and annotated data available.

A table summarising the key characteristics of each learning paradigm is provided in Table 1.

| **Learning Paradigm** | **Description** | **Key Characteristics** | **Typical Applications** | **Data Requirements** |
|---|---|---|---|---|
| Supervised Learning | Learns mapping from input to output labels | • Uses labelled data<br>• Minimizes loss function<br>• Produces predictive model | • Regression<br>• Classification | Large amounts of labelled data |
| Unsupervised Learning | Discovers patterns in unlabeled data | • No predetermined outputs<br>• Finds inherent structures | • Clustering<br>• Dimensionality reduction | Large amounts of unlabeled data |
| Zero-Shot Learning (ZSL) | Tests performance without task-specific training | • No task-specific training<br>• Uses pre-existing knowledge | • New task generalization<br>• Cross-domain applications | No task-specific training data |
| Few-Shot Learning (FSL) | Trains with limited samples | • Uses few labelled examples<br>• Generalizes from limited data | • Rare event detection<br>• Personalized models | Small labelled dataset (e.g., 1-10 samples/class) |

**Table 1** Summary of Learning Paradigms

### 2.5.7 NLP tasks of interest

As described, LLMs can be adapted to different downstream tasks, and those relevant to this thesis are outlined below:

**Information Extraction (IE)** IE focuses on automatically extracting structured information from unstructured or semi-structured text. Common tasks in the clinical field include: Named Entity Recognition (NER), Relation Extraction (RE), and Event Extraction (EE). Below is an example made-up sentence:

```
"The patient was prescribed Warfarin to treat his atrial
fibrillation."
```

- NER - involves labelling words in a text that refer to types such as person, organization, place, date, etc. Example: Identifying "Warfarin" as a drug in the example sentence above.
- RE - identifies relationships, and typically the direction of relationship between entities or events. For the same example, the relation could be *treatment* between *Warfarin* and the *atrial fibrillation*.
- EE - similar to relation extraction but more concerned with extracting details of events, and the when, who, where and how involved.

**Sequence classification** The task of assigning a classification label/category to a sequence of text, like a sentence, paragraph or document, E.g., sentiment analysis (e.g. identifying if a movie review is positive or negative) and topic classification (assigning topics like "sports", "politics", etc to articles). With the example sentence above, the task could be to classify the health area being discussed in a clinical note. In this case, it could be *Cardiovascular*.

**Clustering** Clustering is the unsupervised task of grouping unlabeled items such that items in the same cluster are more similar than those across different clusters. In NLP, clustering algorithms can group together words/documents with similar meanings, contexts, or linguistic properties, without predefined labels. With LLMs, embeddings of words with similar meanings in a given context tend to be close in the embedding space. For example, in "Patient presented with heart palpitations and high bp, suggested to be seen by the cardiovascular team immediately...", embeddings for "heart" and "cardiovascular" would be more similar than other word pairings. This extends to the document level, where an LLM's representations of whole documents cluster based on their contents and purpose. Clustering will be explained in greater detail in relevant experimental chapters for specific tasks of interest.

### 2.5.8 Implementation details

**Hardware** TREs allow researchers to work with sensitive data securely while protecting confidentiality. Access requires rigorous approval and controls on data outputs. Examples include UK Biobank (*138*), SeRP (*139*), and SAIL Databank

(*140*).

Each of the NHS datasets required TREs, and the available compute resources and GPUs differed across datasets (see 2.5). Important factors are Random Access Memory (RAM), GPU, and Video RAM (VRAM). GPUs handle computer graphics/visual processing and are optimized for parallel mathematical operations required in deep learning. Table 2 summarizes the hardware details.

| Dataset | Provider | RAM | GPU model | VRAM (GB) |
|---|---|---|---|---|
| MIMIC-III | local | 80 | RTX 3080 | 24 |
| NHS Patient Safety | NHS England via Microsoft Azure | 32 | Tesla T4 | 15 |
| NHS OHFT | Akrivia via AWS | 32 | Tesla T4 | 15 |

**Table 2** The hardware provider and details available for each dataset

**Text pre-processing** For LLMs, minimal data cleaning is required, as tokenization and contextualized representations benefit from preserving original input. Pre-processing steps included removing carriage returns, tabs, extra whitespace, and poorly encoded characters. Acronyms, jargon, and noise were intentionally kept to encourage LLMs to learn from real clinical texts.

Chapter 3

# Developing Bespoke Healthcare LLMs

## 3.1 Disclaimer

This chapter and its contents are based on one paper available as a preprint and currently under review. I am the first author, with author contributions to the paper provided after the publication details below. Parts of the work were completed during an internship with NHS England in the summer of 2022:

- Niall Taylor, Dan Schofield, Andrey Kormilitzin, Dan W. Joyce, and Alejo Nevado-Holgado. "Developing Healthcare Language Model Embedding Spaces" arXiv preprint arXiv:2403.19802 (2024). N.T, D.S, A.K, and A.N.J conceptualised this work. N.T and D.S curated the datasets. N.T developed pre-processing, experiment running and analysis code. N.T. performed experiments across the three datasets. D.S performed extension experiments on the patient safety data. N.T and D.S explored and evaluated experimental results. N.T drafted the manuscript. D.S, D.W.J, A.K, and A.N.H revised and edited the manuscript.

## 3.2 Introduction

LLMs like BERT (masked language modelling) and GPT (autoregressive language modelling) (*26*, *29*, *141*) offer state-of-the-art performance on many NLP benchmarks (*26*, *29*, *142*, *143*). BERT-like LLMs excel at embedding tasks like classification and retrieval (*143*), while generative LLMs excel at generation and question answering - forming the basis for AI chatbots today.

However, as discussed in Chapter 2.2, LLMs experience performance drops on domain-specific data like healthcare (*39–42*, *49*, *59*, *144*). This is particularly important when looking at UK healthcare datasets, for which no publically available pre-trained LLMs exist. I believe there is a real benefit to having UK and NHS-specific LLMs (as opposed to relying on open LLMs), to capture the writing style, ontologies, and jargon used within NHS trusts.

Training LLMs involves a choice of architecture and pre-training objectives that align with the intended use cases and the compute budget, resources and expertise of an organisation (i.e. the NHS). Pre-training from scratch can be very costly, taking hundreds or thousands of GPU hours, with costs of several hundred thousand pounds or more (*94*). What is considerably cheaper and achievable is the continuation of pre-training. In effect, you initialise an LLM that has already been pre-trained on general open text datasets, such as RoBERTa (*134*), and continue LM training on your own dataset for a short period of time. Even with this approach, larger LLMs of today such as Llama-2-7b (*3*) require substantial hardware, needing approximately 56 GB of GPU VRAM to train all the parameters in full precision. Recall that the GPU available for the NHS data in this thesis has only

15 GB of VRAM - meaning smaller LLMs must be used. To enhance the efficiency and cost of LLM adaptation, I sought to emulate previous work by restricting the training time to just one 24-hour window and a single GPU (*145*).

Furthermore, LLMs must be fine-tuned on downstream tasks after pre-training (as discussed in section 2.3.2.2), risking catastrophic forgetting and necessitating the creation of multiple specialist LLMs. An intermediate approach to mitigate these problems is to pre-train language models to produce good text embeddings by integrating additional knowledge that aligns well with possible downstream tasks. With well-aligned and good embeddings, the amount of subsequent fine-tuning required may be reduced. In turn, this will improve training efficiency and reduce the time and financial burden of adapting LLMs to specific tasks.

In this chapter, I explore the efficacy of altering the pre-training of smaller, BERT-like LLMs to align with the healthcare[1] domain and downstream tasks of interest. The utility of small and resource-efficient LLMs for the healthcare domain is especially attractive where the budget for computing may be limited and training on private data is required. Linked to this problem is the inability to utilise many cloud-based APIs with private data due to data governance laws and issues of confidentiality and security.

[1]I used the term healthcare to cover the more general NHS dataset used in this work that is not strictly an EHR

### 3.2.1 Motivation and related work

Recall that LLM embeddings (section 2.5.3.1) are the learned word representations. With standard language modelling objectives (e.g., MLM), the embeddings are learned through token-level losses. While performant for pre-training and token-level tasks such as NER, many downstream tasks require document-level embeddings, e.g., sentiment analysis, sequence classification, retrieval, and clustering. Using LLMs for such tasks requires aggregating token embeddings into document embeddings, commonly through mean pooling or a linear projection (*29*, *131*, *146*).

However, these crude approximations may be suboptimal, as the pre-training objective did not encourage useful document embeddings directly. The fact that these approaches seem to work sufficiently well does not negate the underlying issue, and there are likely improvements to be made. Several approaches align LMs to produce better sentence embeddings through contrastive learning, clustering similar sentences/documents together (*146–148*). These incorporate a document-level loss, encouraging token embeddings that directly influence the resultant averaged embedding space. A difficulty with contrastive learning is the reliance on class labels for positive/negative pairs. Most contrastive loss functions require labels to distinguish class membership, restricting training to labelled datasets (*146*).

Large labelled healthcare text datasets are rare. For the datasets used in this thesis, there are no extensive or completely labelled data. For this reason, I focus on methods that can use readily available structured metadata for my datasets or use an unsupervised training regime. Both approaches would not require extra

label annotations - which in turn should allow these methods to be more easily applied to other datasets.

The metadata available for each dataset in this chapter differ but typically relate to the purpose of the note. For example, within secondary mental health and the OHFT NHS trust dataset specifically, there are multidisciplinary teams (MDTs) composed of different professionals. A patient may be assessed or examined by different clinicians (doctors, nurses, social workers, psychologists). Each professional is educated and trained to focus on and deliver different aspects of patient care, and this will be reflected in the language recorded in the patient's EHR. For example, doctors record reasoning for diagnosis and management, while nurses understand the patient's current needs in the context of relatives and emotional needs (*149*). Capturing this type of metadata through a contrastive loss function may enhance the resultant LLM embedding space. In turn, these enriched LLM embeddings could assist efficient downstream task adaptation, and thus improve the usability and cost of using the LLMs for different applications.

Improving general domain LLM embeddings has been extensively studied and these studies have influenced this work directly. In particular, DeCLUTR (*147*), sentence transformers (*146*), and SimCSE (*150*) show promise in improving BERT-style embeddings through contrastive methods. Similar works augment pre-training with external knowledge graphs like BioLinkBERT (*51*), Dragon (*52*), and SAP-Bert (*151*) and positively impact biomedical downstream tasks. Though their healthcare applicability is unclear.

Recent work shows task-specific contrastive losses with sentence transformers can adapt LLMs to downstream tasks with few training samples (*39*). Holmusk en-

hanced BERT embeddings with contrastive loss for depression symptom extraction from EHRs (*70*), with strong results despite limited samples. However, their work required extensive annotation. These works strongly support using contrastive learning to enhance LLM embedding spaces.

The key contributions of my work are the introduction of a note category (metadata) pre-training objective for EHR datasets, the development of LLMs for different NHS datasets, and the exploration of resource-constrained pre-training and downstream adaptation. These resultant LLMs will also form the base models for Chapter 5.

## 3.3 Methods

### 3.3.1 Datasets and downstream tasks

In this chapter, I utilised all three datasets outlined in Section 2.5: MIMIC-III, NHS PSIR, and NHS OHFT.

To explore the effect of different pre-training methods on LLMs, I evaluated them on dataset-specific downstream tasks focused on document-wide embeddings. I derived document (sequence) classification tasks using available categorical metadata for each dataset.

The specific tasks for each dataset are outlined below and the data distributions and tasks are overviewed in Table 3. Note that these downstream tasks were not intended for clinically meaningful applications, but rather to investigate discrim-

inative performance differences between pre-trained LLMs as a proxy for understanding the influence of pre-training on the underlying embedding spaces.

### 3.3.2 MIMIC-III

The MIMIC-III dataset (*46*) is a collection of approximately 2.31 million unique clinical notes related to intensive care unit (ICU) admissions at the Beth Israel Deaconess Medical Center between 2001 and 2012.

**ICD-9 Triage** A commonly used categorical variable for MIMIC-III is the ICD-9 diagnosis codes associated with discharge summaries. Originally, a massive multi-class problem with over 2,000 unique codes (with a very skewed distribution) has proven difficult (*57*). An issue with simply classifying ICU discharge notes into specific ICD-9 codes is the lack of validity as a clinical decision support task - ICD-9 codes are typically only used for billing purposes. In collaboration with clinicians, we designed a novel task more similar to the patient flow decision-making process during ICU discharge. ICU patients are "stepped down" to another ward/team when they no longer require ICU care. The ICD-9 codes were grouped into "teams" that reflect the post-ICU triage destination decision found in hospitals. As a proof of concept, the top 20 most frequent MIMIC-III ICD-9 codes (comprising $\sim 80\%$ of the dataset) were selected. A clinician provided triage groups based on the likely team to continue care post-ICU discharge. The ICD-9 codes were mapped to 7 discharge destination teams: Cardiology, Obstetrics, Respiratory Medicine, Neurology, Gastroenterology, Acute/Internal Medicine, and Oncology. The resulting dataset consists of 15,000 clinical notes across these 7 triage categories.

### 3.3.3 NHS OHFT

NHS OHFT is a regional UK-based provider of specialist mental healthcare covering Oxfordshire and Buckinghamshire (covered in more detail in Section 2.5.1.3). The dataset contains full historical EHR data for approximately 200 000 patients spanning over a decade and gives access to around 8 million de-identified clinical notes.

**Triage team association** In the UK, mental health services are structured into primary, secondary, and tertiary levels. Most patients (96%) needing specialist care are referred to and treated by CMHTS (*113*). Referrals contain information written by the referring doctor or professional and are triaged by the receiving CMHT into:

- accept to the team for assessment
- reject due to insufficient information
- route to a sub-specialty team if warranted

Structured EHR data on referral and discharge dates establishes which team accepted the patient, though local administrative variations required developing supplementary heuristics in collaboration with OHFT clinicians. With assistance from clinicians, a classification task to identify the accepting triage team was made for each given referral. Specifically, given a random clinical note, the task is to determine which referral team it likely belongs to based solely on its free text contents. There were a total of 5 sub-specialty teams;

- Eating Disorders (ED)
- Learning Difficulties (LD)
- Early Intervention for Psychosis (EIP)
- Older-adults Community Mental Health Team (OACMHT)
- Perinatal psychiatry (PD)

A diagrammatic overview of the referral and triage process is provided in Figure 8. This is a task that will become part of a more specific triage classification task in Chapter 5, where it is detailed further.

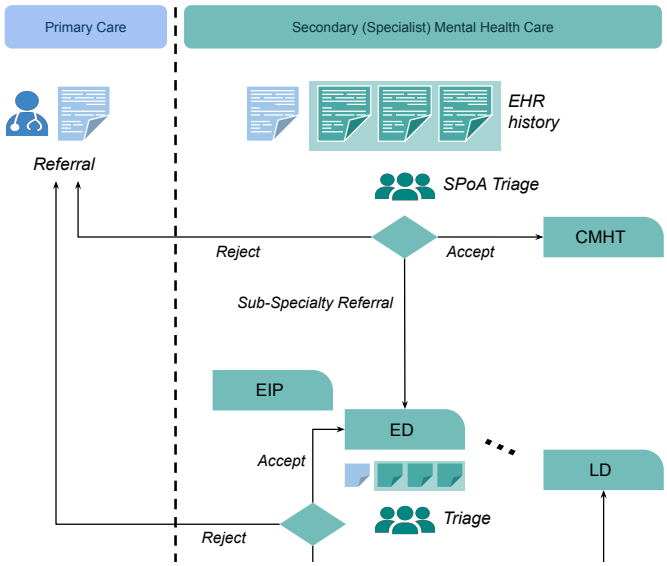


**Figure 8** Schematic description of the OHFT triage process using referral and historical EHR documents. Referrals are received and result in an accept/reject decision. Accepted referrals are typically routed to subspecialty teams, but can be subsequently re-assigned to another team. Single-point-of-access (SPoA) refers to the common location that receives all referrals and where many clinicians will perform triage.

### 3.3.4 NHS PSIR

The PSIR dataset consisted of approximately 2.3 million de-identified patient safety incident reports produced in the financial year 2019/2020. Patient safety incidents are any unintended or unexpected incident which could have, or did, lead to harm for one or more patients receiving healthcare. The NHS use these data to recognise any emerging problems and inform safety regulations, training and policies.

Two categorical variables (metadata) were provided alongside the free text as detailed below:

**IN05 level 1 - Incident category** The labels of incident category at the first level are a standard field provided alongside each incident report which is used to detail the type of the incident, e.g. fall, dosage error, waiting time. There are 13 unique categories.

**PD09 - Incident degree of harm** The degree of incident severity categories is an ordinal scale ranging from no harm (1) to death (5) collected for all incidents. Similar to the MIMIC-III triage task outlined above and with the guidance of the patient safety team who use these data routinely, we derived task related to incidence severity prediction in the following way: the 1-5 incident severity labels given with the free text reports are skewed, with the vast majority being attributed a 1 (or no harm). As this collective dataset is very large, and training with all samples is well beyond the purpose of this work, a smaller and balanced binary

classification dataset, which bins incidence labels into low (severity categories 1-3) and high (severity categories 4-5) severity, was created.

### 3.3.5 Metdata - All datasets

The various origins and purposes of these clinical notes are partially captured by their note category assignment (or associated metadata). These categories can be seen as a form of structured knowledge pertaining to the organization of the healthcare system in focus. This is a commonly captured field in healthcare datasets and while the exact use and meaning may differ between datasets, each dataset contains this field to record the professional role who authored the note or the departmental origin. I expect that note category may be a valuable signal for the pre-training itself because it might be:

- a note entered by a social worker will contain terminology and describe concepts relevant to a patient's social circumstances
- a note entered by an occupational therapist will emphasise functioning and the patient's capacity for everyday tasks
- a note entered by a physician will emphasise clinical state, examinations and treatment plans
- a note entered by a care coordinator will reflect progress on executing a management/care plan for the patient

Note category captures contextual differences in language use and focus across

professional roles and thus provides a meaningful signal for enhancing LLM embeddings during pre-training.

For both MIMIC-III and OHFT, the note category reflects the purpose of the document in relation to the profession of the individual making the note, e.g. Nursing, Doctor or social worker. Whereas for PSIR, the category variable (RP02 on the database)[2] reflects the care setting in which the patient safety incident occurred, such as Acute or general hospital, various community groupings, and GP.

### 3.3.6 Data splits

For each healthcare dataset, there are over 2 million clinical documents available. Due to resource constraints, I developed and trained initial LM pre-training pipelines at various smaller scales. I created a random subsample of 250,000 documents as the LLM pre-training split for each dataset. This was large enough for the purposes of this study. The remaining data for each dataset was utilized for downstream tasks.

Utilizing patient identifiers, I ensured no individual's data was present in both the training and evaluation sets, avoiding explicit data leakage between the two. No data was used for both pre-training and downstream evaluation. An overview of the downstream task data splits is provided in Table 3.

[2]This category variable for PSIR is not the same as the P-Cat label in Table 3

| Dataset | Task (Acronym) | # labels | # train samples | # test samples |
|---|---|---|---|---|
| Mimic-III | Note category (M-Cat) | 8 | 1,600 | 4,000 |
| Mimic-III | ICD-9 Triage (M-Tri) | 7 | 1,400 | 3,150 |
| OHFT | Note category (O-Cat) | 10 | 10,000 | 2,500 |
| OHFT | Referral team relation (O-Tri) | 5 | 6,250 | 2,500 |
| PSIR | IN05: Category (P-Cat) | 13 | 26,000 | 2,600 |
| PSIR | PD09: Severity (P-Sev) | 2 | 14,000 | 14,000 |

**Table 3** Downstream classification dataset statistics. Task names are also given shorthand codes used throughout this chapter.

**Length of documents** An important dataset feature is the number of words (tokens) in each document. Transformer LMs like RoBERTa (*134*) can only handle up to 512 tokens due to the quadratic complexity of self-attention calculations (*152*) (see Section 2.5.3.5). The average document length and distribution varies across the three datasets, though a large portion fits within the 512 token limit, as shown in Table 4. Documents exceeding this are truncated, which is standard practice. There are approaches to mitigate this issue, such as the longformer with sliding attention windows (*152*), key-value caching (*153*), and flash attention (*154*). However, I focus on the standard transformer attention as it best aligns with previous research and covers the majority of the documents sufficiently.

| Dataset | Mean | Percentiles (25:50:75:90) |
|---|---|---|
| MIMIC-III | 410 | 105 : 223 : 424 : 1007 |
| OHFT | 189 | 58 : 115 : 220 : 409 |
| PSIR | 81 | 29 : 55 : 101 : 170 |

**Table 4** Descriptive statistics about the number of tokens across clinical notes related to the LM dataset splits

The MIMIC-III dataset has the longest documents on average, with a sizable proportion reaching or above the 512 token limit. The OHFT and PSIR documents

are shorter, with most under the 512 token limit.

### 3.3.7 Language modelling pre-requisites

The choice of language model architecture was driven by the downstream tasks of interest for clinical documents: document classification and embeddings/representation learning which facilitate other NLP tasks (sequence classification, embedding analysis, clustering). Furthermore, I was seeking to train models in a resource-constrained environment where access to suites of GPUs is unlikely and impractical. For this reason, I opted to use the RoBERTa (*134*) LLM as the starting model. RoBERTa is a variation of the original BERT model (*2*) where the pre-training setup has undergone optimisation to improve on the original choice of training hyperparameters.

Building upon the preliminaries section 2.5.4, I provide a more formalized description of standard LM pre-training: Given a corpus of $D$ text documents $D = [d_1, d_2, ..., d_{n_d}]$, where $n_d$ is the total number of documents. Each document $d$ consists of a sequence of $n_w$ tokens, which are each passed to the LLM $f_{enc}$ with the language modelling function $f_{lm}$ to give a contextualised token vector $\mathbf{E} = [e_1, e_2, e_3...e_{n_w}]$, where $\mathbf{E} \in \mathbb{R}^{n_w \times d_h}$ with $d_h$ being the dimension of the vectors. $f_{head}$ then uses $\mathbf{E}$ for the chosen self-supervised pre-training task and subsequent downstream fine-tuning tasks. To produce a single numerical representation of the whole sequence of tokens, a pooling function $g$ takes the mean of all token embeddings $H = g(f_{enc}(E))$, as has been done previously (*146*, *147*).

#### 3.3.7.1 Continued MLM

The first pre-training method explored is the standard MLM objective, described in section 2.5.4.1. MLM randomly replaces a proportion of input tokens with a special "<MASK>" token, corrupting the original input. The model's objective is to predict the masked token values, based on the context surrounding them.

The standard MLM loss function is given as follows:

$$\mathcal{L}_{mlm}(X, Y) = -\sum_{n=1}^{N_c} W_n(\sum_{i=1}^{|V|} Y^{n_i} \ln(f_{lm}(X)_i^{n_w})) \tag{3.1}$$

where $X$ is the input of the model, $Y$ denotes MLM labels which is a collection of $N_c$ one-hot vectors each with the size of $|V|$, where $|V|$ is the size of the vocabulary of the model and $n_w$ is the number of input tokens[3] and $W_n$ is 1 for masked tokens and 0 for others. This ensures that only masked tokens will contribute to the loss. $f_{enc}$ represents the encoder model with a language modelling head, whose output is a probability distribution vector with the size of the vocabulary ($|V|$) for each token.

#### 3.3.7.2 Contrastive loss pre-training

In classification, contrastive loss functions aim to maximize similarity between examples of the same class (intra-class) and minimize similarity between different classes (inter-class) (*155*). For sequence/document classification, it assumes a set of paired examples $\mathcal{P} = (X_i, X^+j)i, j = 1^P$, where $P$ are the number of samples,

[3]Note that one-hot vectors for non-masked tokens are zero vectors.

and documents $X_i$ and $X_j$ are semantically related documents to be close in the embedding space. Deriving pairs typically requires knowledge of classes or labelled data. As discussed, this is difficult with healthcare datasets, and instead I first utilised a self-supervised method as follows:

**DeCLUTR** (Deep Contrastive Learning for Unsupervised Textual Representations) (*147*) utilizes InfoNCE (Noise-Contrastive Estimation), a self-supervised contrastive loss for identifying positive pairs among samples containing negatives. InfoNCE learns representations by maximizing mutual information between different views/modalities of the same data. In my case, this approach generates spans of text from the same document as the different views of the same data. During training, a batch of $P$ anchor-positive span pairs is taken from document $d$. Each span is encoded by the LLM $f_{enc}$ and token embeddings are pooled using an aggregation function like mean $g(\cdot)$ to create a single embedding per span, $s \in \mathbb{R}^{P \times d_h}$. The cosine similarity between anchor $s_i$ and positive $s_j$ embeddings is maximized while minimizing the similarity of $P^2 - P$ non-matching pairs. For a batch, the cosine similarities calculate the probability a pair of spans inside that batch is a class match:

$$P(s_i, s_j; \tau) = \frac{exp(s_i \cdot s_j/\tau)}{\sum_{k \neq i,j} exp(s_i \cdot z_k/\tau)} \tag{3.2}$$

where $\tau$ is a trainable temperature parameter. This results in the InfoNCE loss symmetrically maximizing match similarity and minimizing non-match similarity.

$$L_{\text{InfoNCE}} = -\frac{1}{2}\left[\frac{1}{P}\sum_{i,j=0}^{P} \log P(s_i, s_j; \tau) + \frac{1}{P}\sum_{i,j=0}^{P} \log P(s_j, s_i; \tau)\right] \tag{3.3}$$

An overview of the entire DeCLUTR learning paradigm is provided in Fig. 9.

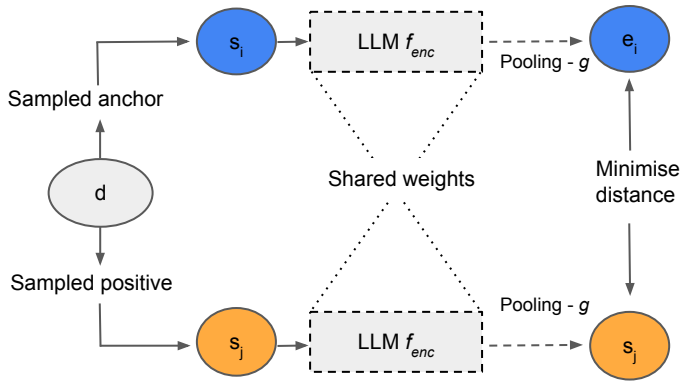


**Figure 9** Adapted from (*147*). Overview of the DeCLUTR training process. Anchor spans $s_i$ and positive spans $s_j$ are sampled from each document $d$ in a mini-batch of size $N$. For simplicity, the figures show an example with $A = P = 1$, where $A$ and $P$ are the number of anchors and positives per document. The spans are encoded by $f_{enc}()$ and pooled by $g(\cdot)$ to get embeddings $e_i = g(f(s_i))$ and $e_j = g(f(s_j))$. The encoder and pooler are trained to minimize the distance between positive span pairs while maximizing the distance to negatives (omitted for simplicity).

As per the original paper (*147*), the minimum number of tokens for a sampled document is: $2 \times A \times S_{max}$ where $A$ is the number of anchors sampled, and $S_{max}$ is the maximum span length. For example, with 2 anchors and a maximum span length of 512, the minimum document length would be 2048. The span length is a key parameter when training using the DeCLUTR regime, with a large search space (many possible values for span length, number of anchors and positives), making exhaustive exploration infeasible. Instead, I selected a set of different span lengths to try which align with the average document length in each dataset. To generate spans of text from a document, a simple sampling strategy of randomly selecting adjacent anchor and positive spans of non-overlapping text with a maximum length, $S_{max}$. To do so, an anchor length, $L_{anchor}$, is first sampled from a beta distribution (with a maximum length of $S_{max}$). Following this, a random

starting position is sampled and the span is extracted. A similar process is used to extract the positive spans, but restricting the starting index to not overlap with the anchor. Sample distributions based on the minimum document length for each dataset are presented in Appendix A.40.

**Note category as a pre-training signal** For the final pre-training method, I utilised the "note category" metadata variables outlined earlier in Section 3.3.5.

Instead of using an explicit contrastive loss function like with DeCLUTR, I formulated this task as a replacement of the original *next sentence prediction* task used in BERTs implementation (*29*). Effectively, this is achieved much the same as standard sequence classification. As outlined earlier (section 2.5.5) the document is passed through the LLM $f_{enc}$ and classification head $f_{head}$ to calculate the softmaxed logits $y_j$ of each of the possible $c$ classes where $p_{c_i}$ is the probability per class (note categories in each dataset).

Following this, the loss for the note category pre-training can be defined by the standard cross-entropy formula:

$$\mathcal{L}_{note} = -\sum_{c=1}^{N} y_c \log(p_c) \tag{3.4}$$

In principle, the cross-entropy loss function operates similarly to the contrastive loss function outlined for DeCLUTR. It will effectively push members of the same class together in the embedding space to enable the separation of the classes to improve the discriminative performance. Consequently, this will lower the loss. A diagrammatic overview is provided in Figure 10.

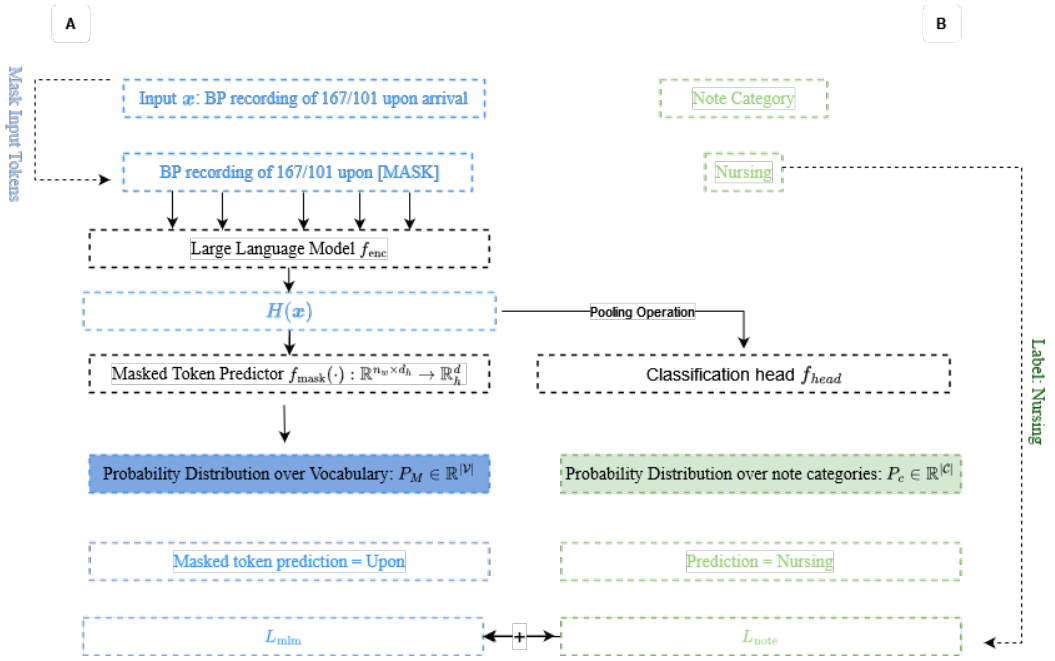


**Figure 10** Overview of the note category pre-training approach. On the left side (**A**) shows the flow of the input sequence (**x**) through the standard MLM pipeline, and on the right side (**B**) shows the integration of the associated note category label in parallel. The MLM and note category classification objectives are jointly optimised with each document.

With both the DeCLUTR and Note contrastive pre-training objectives outlined above, their respective losses are combined with standard MLM to form a joint loss function as shown in Equation 3.5. Where $L_{\text{contrastive}}$ is either $L_{\text{InfoNCE}}$ (for DeCLUTR) or $L_{\text{note}}$ (for note contrastive). For the note contrastive pre-training specifically, I applied a reduced weighting $w$ to the loss to avoid overfitting to the classification loss alone.

$$L = L_{\text{MLM}} + w(L_{\text{contrastive}}) \tag{3.5}$$

### 3.3.8 Evaluation of LLMs

After pre-training with the different loss objectives, the LLMs were evaluated across two facets: the classification performance on their respective classification tasks (Section 3.3.1) and document embedding space analysis as outlined below:

#### 3.3.8.1 Classification performance

Each of the downstream classification tasks are standard sequence classification tasks, as outlined in Section 2.5.5.

**LLM Freezing and Few-shot Training** To explore the potential of using the different LLM embeddings without further fine-tuning for a given downstream task, I conducted experiments in various settings:

- Full fine-tuning of all LLMs and classification head parameters
- Keeping the LLM body frozen and only training the classification head
- Freezing different numbers of LLM layers to reduce trainable parameters and to potentially mitigate catastrophic forgetting
- Altering the number of training samples per class

These approaches allowed me to evaluate the embedding's effectiveness across different levels of task-specific adaptation.

To evaluate the viability of approaches with limited training data, which is particularly relevant in healthcare where labelled data can be scarce, I compared full

and few-shot learning setups. For few-shot experiments, sample size refers to the number of samples per class for training, i.e., $N = s \times c$, where $N$ is the total training samples, $s$ is the sample size per class, and $c$ is the number of classes[4]. To observe a trend in increasing the number of training samples provided per class, I chose various few-shot learning sizes between 16 and 1250. All evaluation results are on the full held-out test sets for each task, with varying training sample sizes.

#### 3.3.8.2 Document embeddings analysis

Beyond downstream classification performance, I explored a selection of different metrics of the LLMs embedding space to discern any clear differences produced by the varied pre-training objectives used:

**Uniformity and alignment** Following a similar analysis plan to SimCSE (*150*), I probed the embedding space quality through measures of alignment between in-class pairs and uniformity across the entire space.

Alignment calculates the expected distances between paired instances, which were embeddings of documents within the same class (a proxy for positive pairs).

$$\ell_{\text{align}} \triangleq \underset{(x,x^+)\sim p_{\text{pos}}}{\mathbb{E}} \|f(x) - f(x^+)\|^2 \tag{3.6}$$

Conversely, uniformity measures how well the embeddings for all documents are uniformly distributed, regardless of class. For each metric, a lower score is argued

[4]In some cases, not all classes could fill the sample size, leading to class imbalance in few-shot experiments.

to be more desirable by the original authors and together, these metrics help illuminate how the embedding spaces are represented.

$$\ell_{\text{uniform}} \triangleq \log \underset{\substack{x,y \overset{i.i.d.}{\sim} p_{\text{data}}}}{\mathbb{E}} e^{-2\|f(x)-f(y)\|^2} \tag{3.7}$$

**Cosine similarity, network analysis and clustering** Cosine similarity is a common distance metric for determining how close together two points are in an embedding space. For my analysis, I opted to look at cosine similarity within and between known classes for each dataset, where:

$$Cosine(x, y) = \frac{x \cdot y}{|x||y|} \tag{3.8}$$

Additionally, I used cosine similarity to construct a graph network analysis. In this approach, documents are represented as nodes in a graph. The edges connecting nodes were determined by the cosine similarity between documents according to a threshold. The connectivity of the graph and the formation of connected subgraphs can reveal insights about how notes are positioned in embedding space.

## 3.3.9 Implementation details

### 3.3.9.1 LLM model setups

The different base LLM models and pre-training combinations are provided in Table 5.

| Clinical Dataset | Domain Pre-training | Model size | Model Name |
|---|---|---|---|
| None | None | 124.6 | RoBERTa-base |
| Mimic-III | MLM | 124.6 | RoBERTa-mimic |
| Mimic-III | DeCLUTR + MLM | 124.6 | RoBERTa-mimic-DeCLUTR |
| Mimic-III | Note + MLM | 125.1 | RoBERTa-mimic-note |
| OHFT | MLM | 124.6 | RoBERTa-OHFT |
| OHFT | DeCLUTR + MLM | 124.6 | RoBERTa-OHFT-DeCLUTR |
| OHFT | Note + MLM | 125.1 | RoBERTa-OHFT-note |
| PSIR | MLM | 124.6 | RoBERTa-PSIR |
| PSIR | DeCLUTR + MLM | 124.6 | RoBERTa-PSIR-DeCLUTR |
| PSIR | Note + MLM | 125.1 | RoBERTa-PSIR-note |

**Table 5** Overview of the different datasets and LLM pre-training. Domain pre-training refers to whether this model has been explicitly pre-trained using the related clinical dataset. We will use this table to define the model names that will be used throughout the results section. Each model uses RoBERTa-base as the base model.

I conducted training, evaluation, and hyperparameter exploration with the use of a single local GPU, with a maximum time of training for one model of 24 hours (emulating previous works on efficient LLM pre-training (*145*)). For more information on the hardware setups in each case, please refer back to Section 2.5.8.

**Hyperparameter choices** Table 6 reports the hyperparameters for the different pre-training methods.

| Parameter | MLM | DeCLUTR | Note Contrastive | Downstream |
|---|---|---|---|---|
| Batch size | 16 | [16, 32, 64, 128] | 16 | 16 |
| Gradient accumulation steps | 4 | 1 | 1 | 1 |
| Embedding dimension | 768 | 768 | 768 | 768 |
| Learning rate | $1 \times 10^{-5}$ | $1 \times 10^{-5}$ | $1 \times 10^{-5}$ | $1 \times 10^{-4}$ |
| Optimiser | Adam W | Adam W | Adam W | Adam W |
| Span length | - | [16, 64, 512, 1024] | - | - |
| Contrastive loss weight | - | - | [0.1, 0.3, 1.0] | - |
| Epochs | 3 | 3 | 3 | 5 |

**Table 6** Overview of hyperparameters used in the pre-training experiments. All training regimes utilised a linear scheduler with warm-up.

An important note on the DeCLUTR approach is that the batch size directly affects the difficulty of the contrastive loss objective used. In the contrastive loss paradigms, for any given anchor, the model needs to make the correct match out of the $N$ documents in the batch. The probability of successfully finding the correct match by chance decreases with the batch size. The selection of the optimal span length also has a direct impact on the possible batch size, with smaller span lengths allowing a much larger batch size. A batch size of 128 on the single-GPU machine was the maximum possible across all span length options tried. Additionally, the original authors of DeCLUTR found that using 2 anchors was optimal for training, whereas the number of positives had little effect.

Combining MLM and the note category losses was optimal for the note contrastive models. However, when given equal weighting, the note category loss often dominated due to it being new to the already MLM pre-trained model. This would lead the model to perform very well on the classification task but lead to catastrophic forgetting of the MLM pre-training. Thus, lowering the weighting of that loss function improved the subsequent downstream performance and ability to function as an LM.

Due to the different batching approaches required for each pre-training approach, it was not possible to keep certain parameters consistent. Instead, I forced each training regime to complete a maximum of 3 epochs in an attempt to keep the number of training updates received during pre-training relatively consistent.

The optimal span length differed between each of the datasets: 1024 for MIMIC-III with a batch size of 16, and 64 with a batch size of 128 for OHFT and Patient Safety. I chose to select these parameters to align with the average length of

documents per dataset - although the differences in performance between span lengths were not large enough to warrant any further investigation. The optimal contrastive loss weight used for all note contrastive models was 0.1.

## 3.4 Results

### 3.4.1 Pre-training loss

The training losses of each of the pre-training methods for the OHFT and MIMIC-III datasets are presented in Fig. 11. I was unable to reproduce the same plots for the PSIR dataset due to a lack of access to PSIR data. Each of the model's losses was consistently reduced over the training steps.

I separated the loss functions of the note category pre-training methods, and we can see both losses are reducing and contributing to the overall loss effectively.

### 3.4.2 Downstream classification performance

The performance evaluation for each pre-training method across the different datasets and tasks is presented in Table 7. The DeCLUTR models performed best when the LLM remained frozen during fine-tuning. In the fully fine-tuned setting, MLM models generally perform marginally better. Across all tasks and fine-tuning settings, the models with no domain pre-training achieve the lowest performance. For brevity, I only report the F1 macro-averaged score here. For all individual performance metrics, please see Appendix A.2.

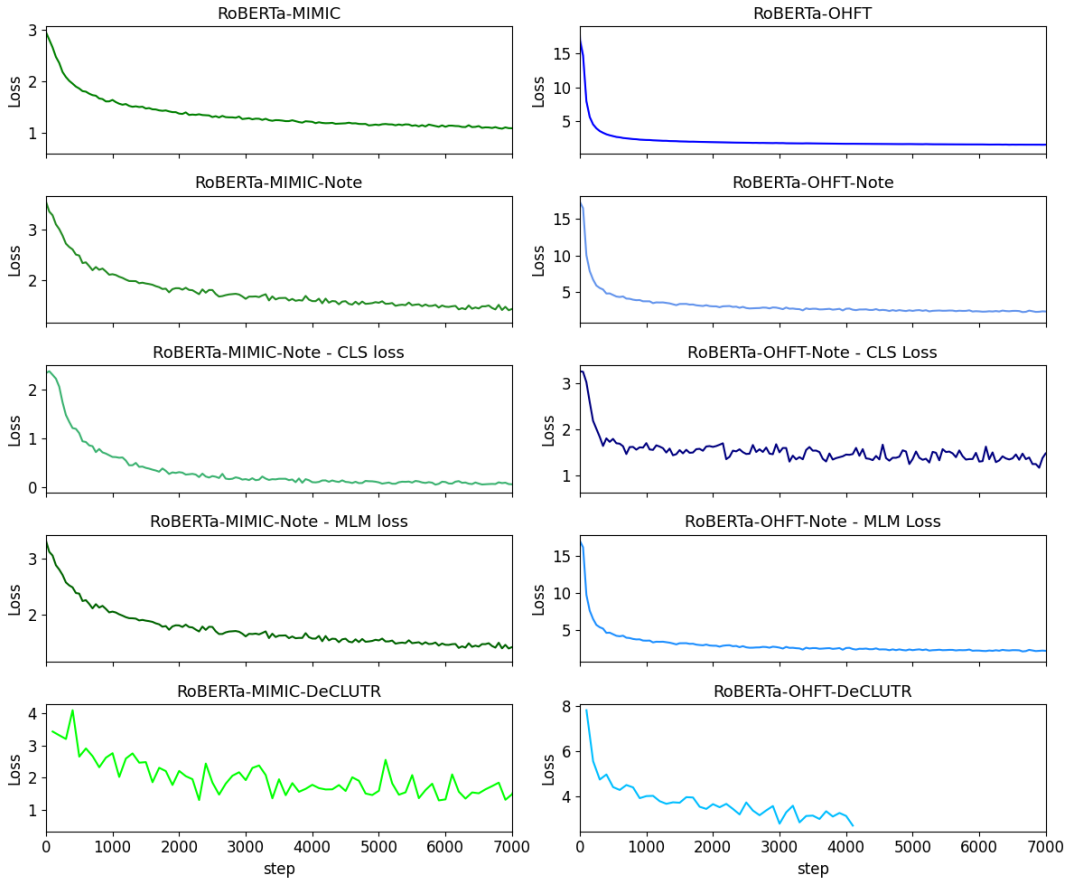


**Figure 11** LM training loss curves for the OHFT and MIMIC-III datasets. MLM loss stands for the masked language modelling loss, and CLS loss stands for the note category classification loss. The training parameters for each approach are given in Table 5.

The results show that the LLMs given no domain pre-training have the lowest performance in both the frozen and fine-tuned settings, with the DeCLUTR models performing best when the LLM remains frozen during the downstream training. In the fine-tuned setting the MLM models generally perform a little better than the others, except on the PSIR dataset tasks (P-Cat, P-Sev) where the note contrastive models had a slight advantage.

| LLM | Domain pre-training | M-Cat | M-Tri | O-Cat | O-Tri | P-Cat | P-Sev |
|---|---|---|---|---|---|---|---|
| Frozen | None | 0.766 | 0.314 | 0.319 | 0.423 | 0.384 | 0.65 |
| | MLM | **0.867** | 0.439 | 0.355 | 0.552 | 0.498 | 0.739 |
| | DeCLUTR + MLM | 0.859 | **0.711** | **0.411** | **0.601** | **0.555** | **0.748** |
| | Note + MLM | - | 0.471 | - | 0.546 | 0.450 | 0.714 |
| Fine-tuned | None | 0.991 | 0.827 | 0.593 | 0.766 | 0.655 | 0.837 |
| | MLM | **0.991** | **0.846** | **0.629** | **0.779** | 0.660 | 0.842 |
| | DeCLUTR + MLM | 0.988 | 0.844 | 0.613 | 0.765 | 0.653 | 0.844 |
| | Note + MLM | - | 0.836 | - | 0.757 | **0.663** | **0.847** |

**Table 7** $F1$ macro across all datasets and classification tasks based on domain pre-training received. We report the maximum $F1$ macro achieved over 5 epochs of training per model. Task acronyms are used: MIMIC-III note category (M-Cat), and ICD-9 triage (M-Tri). OHFT note category (O-Cat) and triage team (O-Tri). PSIR incident category (P-Cat), and incident severity (P-Sev).

#### 3.4.2.1 Few-shot learning

The results of the few-shot learning experiments are presented in Fig. 12.

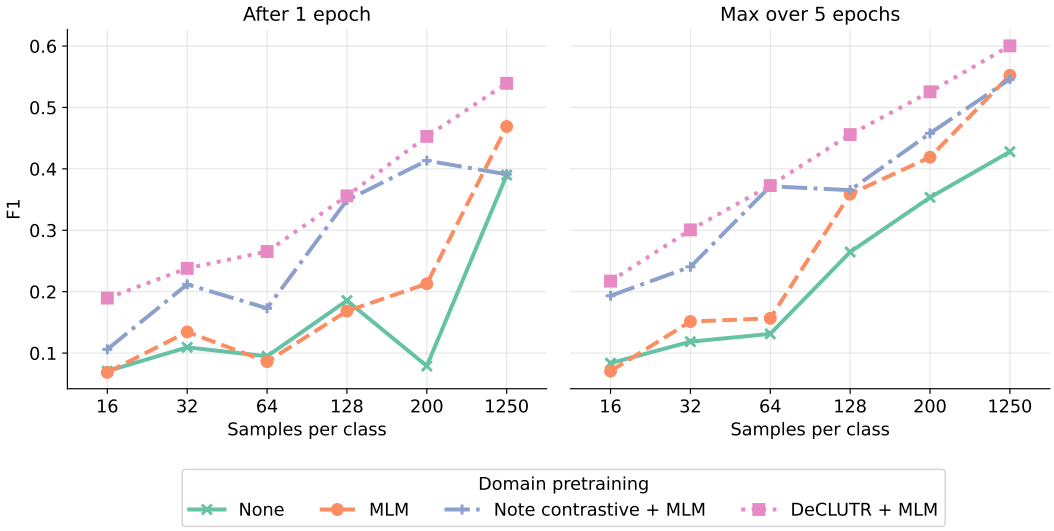


**Figure 12** F1 macro score on the evaluation set for the OHFT Accepted Triage task (Section 3.3.3) with frozen LLMs trained with different sample sizes per class.

There is a clear correlation with the number of training samples per class and performance, with the DeCLUTR model maintaining an advantage across 5 epochs of training. With lower sample sizes, the note category models perform slightly

better than the MLM only and no pre-training models. The LLM with no domain pre-training lags behind in performance across all sample sizes. The results of few-shot learning for the remaining datasets and tasks are provided in Appendix A.2.1.

#### 3.4.2.2 Effect of freezing layers

The effect of freezing an increasing number of consecutive layers of the LLM on the evaluation performance is presented in Fig. 13.

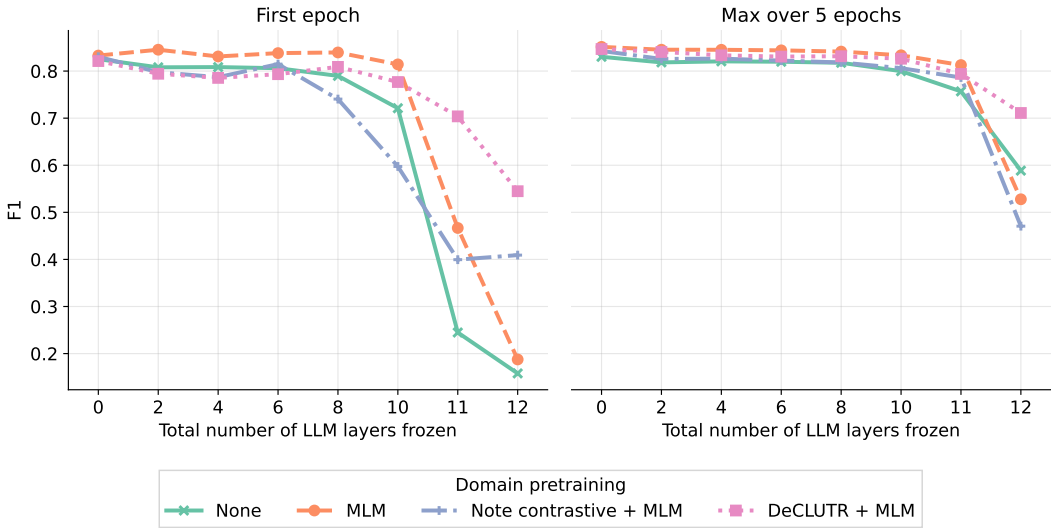


**Figure 13** F1 macro score on the evaluation set for the MIMIC-III ICD-9 Triage task with varying transformer layers frozen (all models utilised a 12-layer RoBERTa architecture as the base).

The greater the number of transformer layers frozen, the worse the performance. However, the degradation in performance is lesser in the DeCLUTR models, especially when only given 1 epoch of training.

### 3.4.3 Document embeddings analysis

I present the exploration of the embedding spaces of the different LLMs for the MIMIC-III dataset and ICD-9 triage task below. Unfortunately, due to the limited TREs required for the other datasets (section 2.5.8), it was not possible to run this analysis on all datasets.

#### 3.4.3.1 Cosine similarity

The LLMs which utilised a contrastive loss function (RoBERTa-mimic-DeCLUTR and RoBERTa-mimic-note) exhibit a much greater separation of embeddings, with a much wider range of cosine similarity values. However, the differences between and within class members revealed a similar pattern for all models, see Fig. 14.

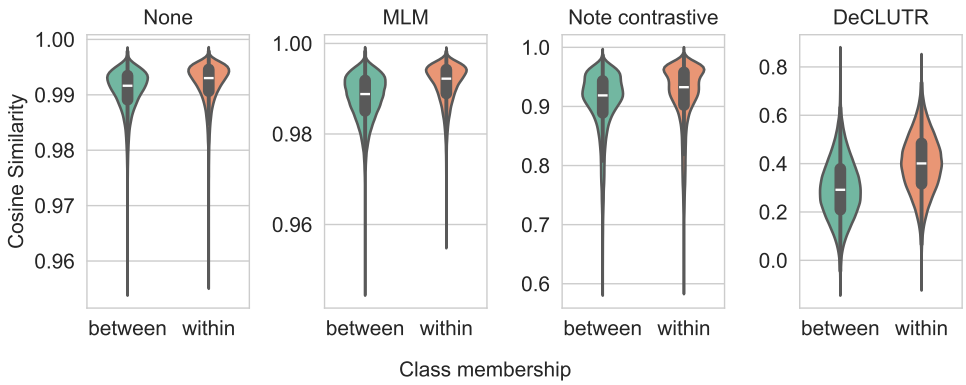


**Figure 14** Cosine similarity of document embeddings within and between classes for the MIMIC-III ICD-9 triage dataset. Note the y-axis scales are separate for each subplot, this is due to the large differences in value ranges between models.

These findings suggest that the separation of documents in the embedding space produced by the contrastive loss functions, in particular for the DeCLUTR approach was relatively non-specific. The DeCLUTR uses a sampling approach that

is effectively self-supervised and derives its within-class membership, instead of using any known class labels. Therefore, the approach may inadvertently result in samples of different note category class labels together. This would then result in a loss function that pushes mixed classes closer together, rather than apart as intended.

#### 3.4.3.2 Alignment and uniformity

An example of comparing uniformity versus alignment is presented in Fig. 15. The DeCLUTR models appeared to have produced an embedding space with a high diversity, with low uniformity between all classes. However, they also had a high alignment score, implying that within-class embeddings remain relatively far apart.

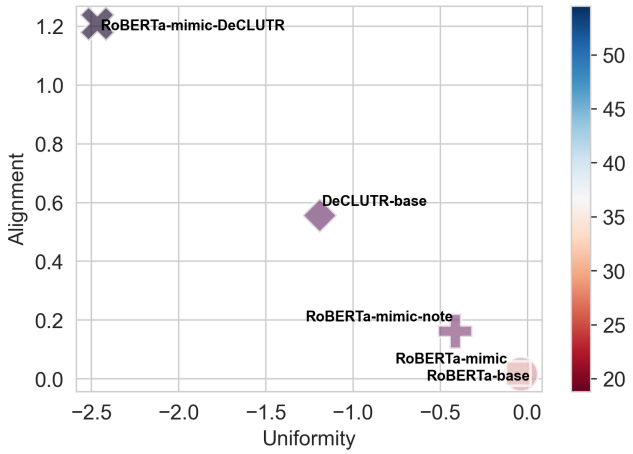


**Figure 15** Uniformity vs. alignment for the different LLM setups for the MIMIC-III ICD-9 task embeddings. The colour bar represents the corresponding F1 score on the ICD-9 triage classification.

This analysis further supports the large separation and closeness of embeddings introduced by the DeCLUTR algorithm and the note category loss function to a

lesser degree.

#### 3.4.3.3 Network analysis

A simple graph network analysis is provided in Table 8. Here, the nodes are documents, and edges are the cosine similarity between document embeddings. A simple approach of setting different thresholds for how similar (or close in embedding space) two documents need to be to in order to be deemed *connected* and observe the change in the graph structure.

| Model name | Cos. threshold | # Components | Avg. degree |
|---|---|---|---|
| RoBERTa-base | 0.995 | 1 | 317.896 |
| | 0.996 | 4 | 54.355 |
| | 0.997 | 8 | 24.128 |
| | 0.997 | 42 | 46.628 |
| RoBERTa-mimic | 0.994 | 2 | 160.150 |
| | 0.995 | 7 | 168.512 |
| | 0.996 | 16 | 82.579 |
| | 0.997 | 45 | 45.913 |
| RoBERTa-mimic-DeCLUTR | 0.538 | 1 | 317.300 |
| | 0.617 | 3 | 54.242 |
| | 0.699 | 24 | 26.233 |
| | 0.742 | 50 | 44.607 |
| RoBERTa-mimic-note | 0.959 | 3 | 115.873 |
| | 0.967 | 8 | 127.278 |
| | 0.973 | 23 | 56.543 |
| | 0.977 | 34 | 38.244 |

**Table 8** Quantitative analysis of graph properties for different models. Cosine similarity thresholds are derived from the 90th, 95th, 98th, and 99th percentiles. The number of components reflects the count of connected components given the threshold chosen, and the average degree refers to the graph degree across the top 3 connected components.

The network analysis highlighted that whilst the cosine distances between embeddings for the LLMs that used a contrastive loss during pre-training are greater, they retain a similar graph network structure to other LLMs. The number of components (sub-graphs) can be considered to be a good indicator of the diversity of the whole graph and correlate strongly with the cosine similarity threshold.

#### 3.4.3.4 Cluster analysis

A simple analysis using the K-Means clustering technique is provided below in Table 9. The following clustering metrics are reported:

- **David Bouldin index (DBi)** - (*156*) The DBi determines the average similarity measure of each derived cluster with its most similar cluster, with similarity defined as the ratio of within-cluster distance to between-cluster distance. Far-apart clusters with little dispersion result in a better, lower score.
- **Calinski Harabaz Index (CHi)** - CHi is the ratio of the sum of between-cluster and within-cluster dispersion, with higher scores indicating more separable clusters.
- **Silhouette score (SS)** - The SS assesses the overlap of clusters and ranges from -1 to 1 with 1 being optimal.

| Model | DBi score (<) | CH score (>) | Silhouette score (>) | Optimal K |
|---|---|---|---|---|
| RoBERTa-base | 2.103 | 478.670 | 0.147 | 5 |
| RoBERTa-base-DeCLUTR | 2.848 | 300.310 | 0.114 | 4 |
| RoBERTa-mimic | 1.663 | 911.911 | 0.235 | 4 |
| RoBERTa-mimic-note | 1.582 | 755.344 | 0.230 | 6 |
| DeCLUTR-base | 3.031 | 284.237 | 0.096 | 4 |

**Table 9** K-means cluster analysis results for each pre-trained LLM related to the Mimic-III ICD-9-Triage dataset. The sign next to each metric indicates the direction of good performance.

The cluster analysis showed that RoBERTa-mimic (MLM domain pre-training only) appeared to perform best, with DeCLUTR models the worst. This again implies that the contrastive loss function used by DeCLUTR has forced a very

distinct embedding space, that is now more difficult to re-organise using k-means clustering.

### 3.4.4 Ablation results

To determine the effect of the contrastive loss components during note category pre-training, I investigated isolating each of the MLM and note category losses. I analysed the MIMIC-III and OHFT datasets only, due to access constraints with the PSIR data.

| LLM | Domain pre-training | M-Tri | O-Tri |
|---|---|---|---|
| Frozen | Note + MLM | 0.471 | 0.546 |
| | Note MLM only | 0.465 | 0.565 |
| | Note CLS only | 0.273 | 0.518 |
| Fine-tuned | Note + MLM | 0.836 | 0.757 |
| | Note MLM only | 0.839 | 0.765 |
| | Note CLS only | 0.741 | 0.761 |

**Table 10** Classification results for the note category pre-trained LLMs with different components of the loss function removed. I report the maximum $F1$ macro achieved over 5 training epochs per model. Task acronyms are used: MIMIC-III ICD-9 triage (M-Tri). OHFT triage team (O-Tri).

As shown in Table 10, for the MIMIC-III ICD-9 triage task (M-Tri), MLM-only models performed similarly to the combined loss model, with a 0.05 drop in F1 macro score in the frozen setting and a marginal 0.03 increase in the fine-tuned setting. However, the note category loss-only model substantially affected downstream task performance, with a 0.2 and 0.1 drop in F1 macro in the frozen and fine-tuned settings, respectively. With the OHFT Accepted triage team task (O-Tri), there was little difference: The MLM-only loss model had a subtle increase of 0.02 and 0.01 in frozen and fine-tuned settings. Note-only loss led to a 0.05 and

0.005 drop in frozen and fine-tuned settings, respectively.

### 3.4.5 Performance of open LLMs

In Table 11 I show further evaluation results using similarly sized open LLMs pre-trained on biomedical and clinical text, including models trained on all MIMIC-III notes (recall I only used a subsample of 250,000). Namely, BioLinkBERT and BioClinicalBERT, which have previously achieved near SoTA when applied to open medical NLP datasets (*44*, *47*, *144*). The presented results are for the MIMIC-III triage task (M-Tri) and the OHFT accepted triage task (O-Tri).

The performance of the open LLMs on the MIMIC dataset is similar to the models produced in this work. Whilst the open LLMs lag behind the DeCLUTR model in the frozen setting, when fully fine-tuned, there is little difference. This is likely due to the popularity of the MIMIC-III dataset in the research literature, with certain open models exposed to snippets of the data itself.

For the OHFT dataset, there is a larger gap between the domain pre-trained and the open models, across both frozen and fine-tuned settings. This is a crucial finding, as it implies there is a distinct difference in the domains used for pre-training. The open LLMs that have achieved SoTA on MIMIC-III data, i.e. BioClinicalBERT and BioLinkBERT, do not achieve the same performance on the NHS OHFT dataset.

| Model | LLM | Accuracy | F1 | AUC | Precision | Recall |
|---|---|---|---|---|---|---|
| RoBERTa-mimic* | Frozen | 0.711 | 0.528 | 0.943 | 0.641 | 0.581 |
| RoBERTa-mimic-DeCLUTR* | Frozen | 0.842 | 0.711 | 0.964 | 0.721 | 0.758 |
| RoBERTa-mimic-note* | Frozen | 0.688 | 0.471 | 0.899 | 0.515 | 0.528 |
| BioLinkBERT-base | Frozen | 0.782 | 0.612 | 0.950 | 0.672 | 0.634 |
| Bio-ClinicalBERT | Frozen | 0.796 | 0.615 | 0.926 | 0.640 | 0.631 |
| RoBERTa-mimic* | Fine-tuned | 0.928 | 0.846 | 0.991 | 0.813 | 0.902 |
| RoBERTa-mimic-DeCLUTR* | Fine-tuned | 0.922 | 0.844 | 0.991 | 0.818 | 0.898 |
| RoBERTa-mimic-note* | Fine-tuned | 0.919 | 0.836 | 0.989 | 0.803 | 0.894 |
| BioLinkBERT-base | Fine-tuned | 0.942 | 0.870 | 0.988 | 0.846 | 0.911 |
| Bio-ClinicalBERT | Fine-tuned | 0.921 | 0.844 | 0.992 | 0.818 | 0.904 |

**(a)** MIMIC-III ICD-9-triage task (M-Tri)

| Model | LLM | Accuracy | F1 | AUC | Precision | Recall |
|---|---|---|---|---|---|---|
| RoBERTa-OHFT* | Frozen | 0.566 | 0.552 | 0.846 | 0.586 | 0.566 |
| RoBERTA-OHFTs-DeCLUTR* | Frozen | 0.607 | 0.600 | 0.858 | 0.617 | 0.607 |
| RoBERTa-OHFT-note* | Frozen | 0.550 | 0.546 | 0.812 | 0.553 | 0.550 |
| BioLinkBERT-base | Frozen | 0.481 | 0.476 | 0.769 | 0.499 | 0.481 |
| Bio-ClinicalBERT | Frozen | 0.443 | 0.435 | 0.742 | 0.450 | 0.443 |
| RoBERTa-OHFT* | Fine-tuned | 0.778 | 0.779 | 0.947 | 0.786 | 0.778 |
| RoBERTA-OHFT-DeCLUTR* | Fine-tuned | 0.764 | 0.765 | 0.938 | 0.774 | 0.764 |
| RoBERTa-OHFT-note* | Fine-tuned | 0.756 | 0.757 | 0.938 | 0.766 | 0.756 |
| BioLinkBERT-base | Fine-tuned | 0.703 | 0.700 | 0.915 | 0.714 | 0.703 |
| Bio-ClinicalBERT | Fine-tuned | 0.715 | 0.714 | 0.922 | 0.718 | 0.715 |

**(b)** OHFT accepted triage team task (O-Tri)

**Table 11** Evaluation metrics for text classification tasks. The reported scores are the maximum obtained over 5 epochs of training. * represents models produced in this work. LLM here refers to whether the base LLM was frozen during downstream fine-tuning or fully fine-tuned with the classification head.

### 3.4.6 Pre-training effects on token classification

One aspect of this work was to determine how pre-training affected the document-level embeddings. Additionally, I investigated the impact on word-level embeddings, expecting the contrastive loss functions to have shifted the LLM's objective from the original MLM objective, affecting token-level tasks. I evaluated each pre-trained model from the MIMIC-III dataset on three NER tasks from various I2B2 challenges (*157–159*), with gold standard ground truths. Micro-averaged F1 scores for each task are provided in Table 12.

The i2b2 2010 and 2012 tasks involve temporal relation extraction based on discharge summaries from Partners Healthcare and the Beth Israel Deaconess Medical Center. Discharge summaries were annotated by clinicians for mentions of clinical concepts, clinical departments, evidentials, and occurrences. The i2b2 2014 task is a deidentification task, whereby the model is tasked with identifying different protected health information (PHI) identifiers, such as name, address and postcode. These tasks will be explored further in Chapter 4, Section 4.3.1.2.

| Model | LLM | i2b2 2010 | i2b2 2012 | i2b2 2014 |
|---|---|---|---|---|
| RoBERTa-base | Frozen | 0.052 | 0.042 | 0.639 |
| RoBERTa-base-DeCLUTR | | 0.083 | 0.092 | 0.551 |
| RoBERTa-mimic | | 0.041 | 0.08 | 0.636 |
| RoBERTa-mimic-note | | 0.006 | 0.067 | 0.616 |
| DeCLUTR-base | | 0.043 | 0.065 | 0.651 |
| RoBERTa-base | Fine-tuned | 0.847 | 0.83 | 0.975 |
| RoBERTa-base-DeCLUTR | | 0.854 | 0.837 | 0.977 |
| RoBERTa-mimic | | 0.854 | 0.847 | 0.976 |
| RoBERTa-mimic-note | | 0.855 | 0.841 | 0.979 |
| DeCLUTR-base | | 0.844 | 0.833 | 0.976 |

**Table 12** Token classification results. Due to the relative label imbalance in the presented NER tasks, I report F1 micro to represent the global performance of the models.

Results showed no major differences between models, and keeping the LLM frozen left models unable to learn the 2010 and 2012 tasks. The 2014 de-identification task did exhibit some model performance differences, with a slight decrease in performance for the DeCLUTR and note category models. The 2014 de-identification task is potentially less clinical in nature and was easier for the base LLMs to achieve. Overall, the NER results suggest that these approaches may have produced an embedding space less applicable to direct NER tasks. However, in the fine-tuned setting, all models converged on similar performance, in line with other studies (*45*). Therefore the fine-tuning still appears to allow the contrastive loss

models to achieve NER to a reasonable degree.

## 3.5 Discussion

Pre-training of LLMs is crucial for producing useful embedding spaces for downstream tasks like sequence classification, document retrieval, clustering, and semantic search. I evaluated various pre-trained LLMs on sequence classification tasks in frozen and fine-tuned settings. Models pre-trained with contrastive loss functions generally outperformed other approaches across each of the datasets, requiring fewer training samples for reasonable performance. When the LLM remained frozen during downstream fine-tuning, the DeCLUTR models achieved the highest F1 macro scores, although they remained considerably worse than any of the fine-tuned models. The pre-training had a clear influence on the embedding space usability for tasks where classification boundaries matter. Integrating structured metadata based on note categories during pre-training didn't improve classification performance over standard MLM training but yielded better clustering metrics. The purpose of the metadata for each dataset did differ, and the relation to the downstream tasks was not always evident. In particular, the ICD-9 triage task for the MIMIC-III dataset utilises discharge summaries only, which themselves are a single distinct note category. Meaning, the note category metadata during pre-training may be of little use for that task in particular. Metadata usefulness varied across datasets, with the OHFT dataset and would require further investigation and development of other downstream tasks.

The embedding space analysis revealed a relative lack of separation between doc-

ument embeddings in the MIMIC-III dataset based on the note category classes when no contrastive loss was used. The models with no domain adaptation or MLM only yielded embeddings very close together based on cosine distance. Whereas, the DeCLUTR-based LLMs produced distinct embedding spaces with a large separation between documents and classes. However, the separation of document embeddings within the same known note category classes was also high, suggesting misalignment with known classes. The note category loss pre-training produced a wider spread of embeddings than MLM only, but again with a similar pattern of cosine similarity regardless of class membership. The uniformity and alignment measurements further support these findings, with a clear distinction between contrastive loss models and the others. The graph network analysis revealed surprising consistency across LLMs, regardless of cosine distance differences. This may suggest that whilst the embedding spaces may be more spread out in DeCLUTR models, the overall organisation is similar. Embeddings as direct features are sought after in resource-constrained settings where fine-tuning is difficult. Domain adaptation of open LLMs to the UK clinical domain remains important, with performance gaps between general LLMs and UK NHS dataset-trained LLMs. LLMs trained with OHFT data performed consistently better on the OHFT triage task compared to open biomedical/clinical LLMs, reinforcing the notion that UK healthcare datasets benefit from LLMs pre-trained and adapted to this domain.

The LLM training and adaptation pipeline presented can be used for other NHS trusts to produce specific LLMs for representing large bodies of text. The LLMs can then serve as the foundation for a given downstream task and aid classification tasks directly.

The resource efficiency of domain adaptation through pre-training is not straightforward. Continued pre-training of LLMs requires at least a single GPU and 24 hours or more. While no adaptation of LLMs is resource-friendly, the potential performance gain on downstream tasks is valuable and likely worth the cost.

### 3.5.1 Limitations

**DeCLUTR sampling** The optimal DeCLUTR sampling parameters varied across datasets based on document length. Shorter documents were eliminated, restricting the training subsample compared to other methods. DeCLUTR's unsupervised sampling approach makes ascertaining anchor-positive pair correctness difficult, which potentially accounts for the relative lack of separation between in-class and between-class samples in its cosine similarity embeddings.

**Note category pre-training loss** Exploring different contrastive loss setups or hyperparameters was limited. Differential sample selection for the pre-training objective was not explored. Further work could utilize modern transformer adaptations for larger batch sizes (*143*). Alternative approaches to integrating metadata into LLMs may also offer more value.

**Strict training settings** To showcase different pre-training methods on a budget, training was limited to a single low-end GPU and a strict time limit in line with budget-oriented LLM research (*145*). Greater flexibility in training regimes and algorithms could extend this work.

**Alternative embedding approaches** Recent studies have explored enhancing generative LLMs' embedding and generation abilities using instruction tuning, embedding loss functions (*143*), and graph neural networks for emotion representation in mental health texts (*160*). Applying these to NHS datasets could be interesting for further research.

### 3.5.2 Conclusion

This chapter underscores the importance of pre-training in aligning LLMs with tasks relying on document representations. Contrastive, self-supervised objectives proved most effective across sequence classification tasks, outperforming masked language modelling when training data was limited or the LLM was kept frozen. Incorporating structured metadata via note category during pre-training produced distinct embedding spaces with implications for sub-tasks like classification and clustering. The resource efficiency of this adaptation is non-trivial but the low-resource approaches confer valuable gains. Future work should explore modern architectures and objectives, optimized contrastive learning sampling, enhanced metadata use, and applications beyond classification. Broader hyperparameter tuning may unveil further gains. This research validates pre-trained clinical language models' utilities, provides best practices, and motivates specialized adaptation.

Domain adaptation to UK NHS data remains critical, with specialized models substantially improving over general ones. Beyond the presented analysis, these new LLMs could be used as a tool for clinical text processing for the associated healthcare trusts. For example, the OHFT LLMs produced here would serve as

a bespoke LLM for generating numerical representations of clinical documents within the EHR system for use in wider research. This work can also provide a framework and protocol for efficiently developing other NHS-specific LLMs within TREs. With these LLMs, the hope is that they can facilitate any number of text-processing tasks and possibly reduce the manual workload of admin and clinical staff. The capability of such LLMs for targeted clinical tasks in Psychiatry remains unknown, and I will explore this further in Chapter 5.

Collecting and pre-training models on larger UK healthcare datasets is an important goal to reduce the performance gap observed between general and in-domain models. As clinical NLP expands, rigorous evaluation of pre-training techniques on real-world deployment metrics will clarify best practices for embedding model development. By systematically assessing how different pre-training choices impact model usability across datasets, tasks, and institutional contexts, we can further optimize language models to power clinical decision support.

The next chapter will introduce and evaluate methods for improving the efficiency in the fine-tuning of LLMs for downstream tasks.

Chapter 4

# Efficient fine-tuning of LLMs

## 4.1 Disclaimer

This chapter and its contents are based on two published works, of which I am the first author. I provide the publication details, followed by author contributions.

- Niall Taylor, Zhang, Yi, Joyce, Dan W, Gao, Zimming, Kormilitzin and Nevado-Holgado, Alejo,"Clinical Prompt Learning With Frozen Language Models" in IEEE Transactions on Neural Networks and Learning Systems, doi: 10.1109/TNNLS.2023.3294633. N.T drafted the manuscript in full and conducted all experiments, analysis and coding with contributions and editing of methods sections and code by Y.Z. Certain experiments were conducted by Z.G, but under my supervision and code base. Critical review was provided by all other authors.
- Niall Taylor, Upamanyu Ghose, Omid Rohanian, Mohammadmahdi Nouriborji, Andrey Kormilitzin, David Clifton, and Alejo Nevado-Holgado. "Efficiency at Scale: Investigating the Performance of Diminutive Language Models in Clinical Tasks." arXiv preprint arXiv:2402.10597 (2024) - under

peer review in AI for Medicine. N.T, and O.R conceptualised this work. N.T curated the datasets. N.T and U.G developed pre-processing and experiment running and analysis code. O.R and M.N provided select dataset pre-processing scripts. N.T drafted the manuscript, and U.G revised and drafted the methods section. O.R, D.C, A.K, and A.N.H revised and edited the manuscript.

## 4.2 Introduction

Both sets of experiments in this chapter aim to improve the efficiency of adapting LLMs to new downstream tasks in terms of storage, computing, and data (assisting the LLMs developed in Chapter 3). Below, I present a common introduction and methods section, followed by separate methods, results, and discussion sections for each work. Finally, a general discussion synthesizing all results and their use for Chapter 5 is provided. Due to computational requirements, experiment volume, hyperparameter tuning, and the desire to contribute to open-source, only the openly available MIMIC-III (see also Section 2.5) (*46*) clinical dataset was used.

Fine-tuning general LLMs often performs poorly on clinical text, a domain with limited openly available corpora. As discussed in the literature review in Section 2.3, adapting LLMs to downstream tasks different from pre-training can require substantial resources. Despite domain adaptation improving performance (Chapter 3), fine-tuning clinical LLMs typically requires retraining all parameters, which is computationally expensive, time-consuming, and unsuitable for resource-

constrained environments. Fine-tuning can also lead to catastrophic forgetting of pre-trained knowledge, overfitting to smaller fine-tuning datasets, and lower generalizability to unseen samples (*161–163*). This lack of generalizability has also been observed when fine-tuning on American English clinical text and then validating on British English clinical text (*164*).

### 4.2.1 Motivation and related work

To mitigate the problems associated with traditional full fine-tuning, I distinguish two lines of work. First is prompt learning, which alters the formatting of adapting LLMs to downstream NLP tasks and effectively reduces the number of parameters updated during fine-tuning. Secondly, a set of alternative Parameter Efficient Fine-tuning (PEFT) methods (technically prompt learning can also be seen as a PEFT method, but for clarity, I will refer to these two as separate and distinct). PEFT methods also seek to improve efficiency by introducing a small number of trainable weights, whilst approximating full fine-tuning. I sought to explore and apply select prompt learning and PEFT methods to the healthcare/clinical text domain.

### 4.2.2 Prompt learning

Prompt learning is an alternative to traditional fine-tuning, stemming from research converting NLP tasks into text-to-text problems. The T5 LLM (*100*) treated NLP tasks as natural language instructions (e.g., "Classify the sentiment of the following text..") with outputs as natural language responses (e.g., "posi-

tive", or "negative"). Brown et al. (*30*) used few-shot learning to provide GPT-3 with several task examples and completed answers as part of the prompt, with the test sample with no label as the final part. This effectively "taught" the model the structure of the task and how the answers should be given. The performance of this approach was reasonable, with performance similar to fine-tuning smaller models on select tasks. Without parameter updates, prompting relied on pre-trained knowledge representation, but minor prompt alterations substantially influenced performance (*4*). Few-shot learning only worked well if pre-training was similar enough to the downstream task. Improving the prompt structure has been a popular topic, with improvements seen through enhancing the instruction. Chain-of-Thought (CoT) (*165*) for example encouraged step-by-step thinking, improving few-shot but not zero-shot performance. Reflexion (*166*) involved LLMs evaluating their responses, showing promise but again, with inconsistency across datasets and higher cost. Manual prompting became costly trial-and-error, with no pre-defined solution for all problems. As a solution, soft prompt tuning was introduced. The idea is to avoid handcrafting prompts and inject trainable embeddings as soft prompts that the model learns - known as prompt tuning (*4*). Prompt tuning typically outperforms few-shot prompting and even matches full fine-tuning, while keeping most LLM parameters frozen. A remaining issue for manual prompt learning was the mapping of the LLM outputs to class label words for the task, which requires domain knowledge and can itself be a large search space. To make this process learnable, subsequent work extended soft prompts to model outputs via verbalizers - trainable label embeddings that replace manually selected labels (*167*, *168*). This verbalizer approach is detailed in Section 4.4.

In my work, I investigated several prompting and verbalizer setups with small BERT-style LLMs for select clinical NLP tasks.

### 4.2.3 Parameter Efficient Fine-tuning (PEFT)

The goal of PEFT is to drastically reduce the number of trainable weights to fine-tune LLMs to new tasks, with early work highlighting how different layers of the transformer-based LLMs appear more or less useful for certain tasks (*169*). Notable PEFT methods include: Prefix tuning (*105*), Low Rank Adaptation (LoRA) (*106*), and Inhibit Activations ($IA^3$) (*170*). I will cover these in detail in the relevant methods section 4.7. PEFT methods fall into different categories:

- **Selective**, whereby subsets of the base LLM are chosen to be updated during fine-tuning.
- **Reparametrisation** utilises intrinsic properties of the LLM's weights to alter them during fine-tuning.
- **Additive**, add additional trainable parameters during fine-tuning.

Unlike prompt learning, each of these types is usually combined with the traditional format of fine-tuning, whereby a task-specific head is added and updated. Prompt tuning (*4*), which I have linked to prompt learning, can be used as a PEFT method with traditional fine-tuning too. Prefix tuning (*105*) is an additive PEFT method which injects a set of trainable weights in each layer of the LLM. This is similar to prompt tuning described above, but whilst prompt tuning only adds soft prompts to the input itself as virtual tokens, prefix tuning is applied to every layer

of the LLM. The result is still a large reduction in trainable parameters compared to full fine-tuning, with only a small reduction in task performance. More recently, a reparametrisation technique, LoRA (*106*), made use of the intrinsic properties of fine-tuning weight updates to improve efficiency. The authors of LoRA recognised that standard fine-tuning typically yielded a set of weight update matrices that are intrinsically low in rank. This means there is a large amount of highly correlated information within these matrices that can be well represented by much smaller ones. Effectively, the model can be updated to the same effect with a considerably smaller set of values and reduce the computation required. In their paper, LoRA outperformed other PEFT methods in terms of maintaining performance equivalent to full fine-tuning. However, it remains relatively unexplored with smaller LLMs and the clinical domain. Inhibit Activations ($IA^3$) (*170*) inserts trainable linear layers on top of the activations of certain LLM components, such as the self-attention or the feed-forward layers, which the authors suggest effectively scales these activations. Again, the initial findings were comparable to full fine-tuning, with a large reduction in the number of introduced parameters.

While LLM applicability in the clinical domain is well researched (*56*, *171*), few studies explore PEFT utility for traditional NLP tasks, focusing instead on large-scale models (*106*, *108*, *172*) except (*11*). One group utilised prompt learning for clinical NLP, Cui et al. (*173*) used prompt learning to classify temporal events within discharge summaries. Prompt learning was effective but did slightly underperform traditional fine-tuning. However, in their work, the LLMs were not frozen, which is a major component of the prompt learning approach I opted to use (detailed in section 4.4). Another group has recently investigated PEFT for

clinical NLP tasks with larger Llama-2 models (*3*), finding that certain PEFT methods were comparable to fine-tuning and worked well across several different tasks (*174*). The major distinction in my work is the emphasis on the efficiency of these methods and their applicability to much smaller LLMs, and how this translates to time and cost demands.

Beyond prompt learning and PEFT methods, other strategies exist to address the difficulty in fine-tuning large LLMs. Quantization reduces model size in terms of floating-point precision, bits, and memory needed to store weights, enabling full fine-tuning of moderately sized models (*108*). Pruning model parameters to reduce redundant weights for given downstream tasks has also been effective in certain cases (*107*). Many of these strategies present avenues for future work, especially for very large LLMs, not used in this work. For this thesis, the focus was on prompt learning and PEFT methods well-suited for smaller models and datasets.

The goal of this chapter is to improve the efficiency and ease of applying LLMs to specific tasks and establish the feasibility of doing so in resource-constrained environments. The prompt learning work was a study focused on the applicability of using prompt and verbalizer setups for LLMs in the clinical domain, enabling a direct comparison with traditional fine-tuning. The PEFT learning work has similar objectives, although I was more focused on the interaction of LLM size and domain pre-training with different PEFT methods.

It is important to note when reading this chapter that the clinical NLP tasks presented are intended to serve as a benchmark for the different downstream adaptation methods and models, rather than produce clinically validated prediction models for wider use.

## 4.3 General methods

### 4.3.1 Common Datasets

The derived tasks for this chapter use the MIMIC-III dataset (outlined in section 2.5.1.1) and the I2B2 NLP research datasets (*175*). The I2B2 datasets (also referred to as the National NLP Clinical Challenges research datasets (n2c2)) are based on deidentified notes from the Research Patient Data Registry at Partners Healthcare System in Boston.

There are several derived tasks from these datasets; covering NER, sequence classification and relation extraction, in line with the prompt learning work and other clinical NLP research (*176*, *177*). Both my works using prompt learning and PEFT methods use the sequence classification tasks derived from the MIMIC-III dataset. However, due to the difficulty of aligning prompt learning with NER tasks, I was only able to utilise the other PEFT methods for NER.

Details of each of the available tasks are provided below:

#### 4.3.1.1 Sequence Classification Tasks

**ICD-9 50** Within the MIMIC-III dataset, there are up to around 2,000 unique ICD-9 diagnosis codes, and whilst previous studies have attempted to predict all codes, it is often with quite poor performance and biased towards the common occurring ICD-9 codes. Tackling all ICD-9 codes with a classification model is not simple, and I felt it would not serve the purposes of this work. Instead of

trying to classify all unique ICD-9 codes, the top 50 most frequent were selected, as has been done previously (*57*, *178*, *179*). The main purpose of this task was to test the prompt learning approach for a relatively extreme multi-class problem, which I predicted it would struggle with due to the granular nature of the labels. Unfortunately, the nature of the MIMIC-III data at the time of these experiments means the ICD-9 coding system is now outdated, and as discussed earlier in Section 2.5.1.1, it is not possible to translate findings using ICD-9 diagnosis codes to later ICD-10/11 codes. Therefore I remind the reader that this work serves as a sensible benchmark for the time, and I encourage future work to utilise the later coding systems.

**ICD-9 Triage task** I re-use the MIMIC-III ICD-9 Triage task from Chapter 3, described in detail in section 3.3.2. As a reminder, this task utilised the ICD-9 diagnosis codes associated with each ICU visit and grouped these unique codes into seven post-ICU destination teams: *Cardiology*, *Obstetrics*, *Respiratory Medicine*, *Neurology*, *Gastroenterology*, *Acute or Internal Medicine*, and *Oncology*. The resulting dataset consists of 15,000 clinical notes across the 7 triage categories.

**Mortality Prediction (MP)** A frequent benchmark clinical support task used with the MIMIC-III dataset is the prediction of whether a patient will survive their hospital episode or not. Within the MIMIC-III database are structured data relating to the mortality status of a patient. Only notes before the mortality flag are considered and some simple regular expression rules were used to filter any notes that had explicit mentions of a patient's death, similar to that of previous work (*57*, *180*).

**Length of stay in ICU (LoS)** Predicting how long a patient will require treatment in the ICU is of significant value to hospitals that aim to optimise the flow of patients in resource-limited settings (that is, there are usually very few ICU beds compared to the hospital's overall bed capacity). We model this as a four-way classification task, binning length of stay in the following categories: Under 3 days, 3 to 7 days, 1 week to 2 weeks, and more than 2 weeks (*57*). These are given class labels of $0, 1, 2, 3$ respectively.

For the class label distributions for these tasks see appendix figure 34.

**I2B2 2010 Relation Extraction** From the I2B2 series - the 2010 medical relation extraction dataset (*157*) aims to classify text based on the apparent medical relationship being described in a discharge summary, with the following derived labels:

1. Treatment improves medical problem (TrIP)
2. Treatment worsens medical problem (TrWP)
3. Treatment causes medical problem (TrCP)
4. Treatment is administered for medical problems (TrAP)
5. Treatment is not administered because of medical problem (TrNAP)
6. Test reveals medical problem (TeRP)
7. Test conducted to investigate medical problem (TeCP)
8. Medical problem indicates a medical problem (PIP)

9. No Relations

#### 4.3.1.2 NER tasks

For the PEFT experiments, I also used a set of clinical NER tasks (for a reminder of what NER tasks are, see Section 2.5.7). All NER tasks use the Inside-Outside-Beginning (IOB) tagging format (*181*), whereby **B** represents the beginning of an entity, **I** represents tokens inside an entity span, and **O** represents tokens outside of an entity span.

**I2B2 - 2010 and 2012** The two NER tasks used in this work involved classifying text spans related to temporal relations (*157*, *158*) within discharge summaries, as delineated by expert annotations. The classification is based on four primary categories: clinical concepts, clinical departments, evidentials, and occurrences. These categories are further broken down into more specific entities concerning the following: *medical problem (PR)*, *medical treatment (TR), medical test (TE)*, *clinical department (CD), evidential (EV)*, *occurrence (OC)*, and *none (NO)*. Both datasets are relatively balanced across their respective entity classes and have served as a standard clinical NER benchmark in the field (*49*).

**I2B2 - 2014** A deidentification task, whereby spans of text within clinical notes are classified according to different protected health information (PHI) identifiers such as name, address, and postcode (*159*). This is a more granular dataset, with a skewed distribution across the unique identifiers, with some occurring less than ten times.

I followed the same pre-processing procedure outlined in previous works (*44*, *45*), which involves converting XML and text documents into a more suitable format for LLMs. For full details of the tasks, please refer to the original papers (*157–159*).

For NER class distributions, see Appendix B.2.

### 4.3.2 Data splits

I follow a traditional approach to splitting the datasets into a train, validation, and held-out test set. The ratio is 80:10:10 and evaluation metrics reported are on the test set unless otherwise stated. The number of training and evaluation samples for each dataset are outlined in Table 13.

| Dataset | Task Type | # labels | # train samples | # eval samples |
|---|---|---|---|---|
| MIMIC-III MP | Seq. CLS | 2 | 33,954 | 9,822 |
| MIMIC-III LoS | Seq. CLS | 3 | 30,421 | 8,797 |
| MIMIC-III ICD-9 Triage | Seq. CLS | 7 | 9,559 | 3,172 |
| MIMIC-III ICD-9 50 | Seq. CLS | 50 | 14,360 | 9,447 |
| I2B2 2010 RE | Seq. CLS | 9 | 22,256 | 43,000 |
| I2B2 2010 | NER | 7 | 6726 | 27,626 |
| I2B2 2012 | NER | 13 | 6797 | 5,664 |
| I2B2 2014 | NER | 42 | 45974 | 32,586 |

**Table 13** Dataset details for the sequence classification (Seq. CLS) and NER tasks. # is used to indicate the number of that item.

### 4.3.3 Full and few-shot training

As with Chapter 3, I used both few-shot learning and full training set setups to probe the ability of different prompt learning and PEFT methods to handle limited training samples (for reference, see Section 2.5.6.3).

### 4.3.4 Hardware Details

For the core experiments, the HuggingFace (*141*), Openprompt (*168*) and Parameter Efficient Fine-tuning (PEFT) (*109*) libraries were used. For all approaches, I used the AdamW (*182*) optimizer with a linear warm-up scheduler. All experiments utilised a single NVIDIA RTX 3090 graphics card with 24GB of VRAM for consistency and equal footing between model types.

## 4.4 Prompt learning methods

As discussed already, prompt learning encapsulates several approaches to producing LLM outputs via changes to the input with prompts. Below, I outline the approach I took to prompt learning with LLMs, which utilised an open framework called OpenPrompt (*168*).

Prompt learning modifies the input text $\mathbf{x}$ to a prompt format $\boldsymbol{x'} = f_p(\boldsymbol{x})$, where $f_p$, often called a template, prepends, appends, or inserts additional tokens and a <MASK> token. $\boldsymbol{x'}$ is fed into the LLM to predict the masked token, following the MLM objective (section 2.5.4.1). Let $\mathbf{E} = [e_1, e_2, \ldots, e_{n_w}] \in \mathbb{R}^{n_w \times d_h}$ denote the

embedding of input $\boldsymbol{x}$ of length $n_w$ with dimension $d_h$, where $e_{n_w}$ is the embedding per token (similarly, $\mathbf{E}'$ for $\mathbf{x}'$). A verbalizer $g : \mathcal{V} \mapsto \mathcal{C}$ maps tokens in the LLM's vocabulary $\mathcal{V}$ to class labels $\mathcal{C}$. The LLM's function $f_{lm} : \mathbb{R}^{n_w \times d_h} \to \mathbb{R}^{d_h}$ maps the hidden <MASK> representation to tokens in the vocabulary through a linear layer to produce output probabilities for each possible token in $V$. Through the verbalizer, output probabilities are restricted to defined class labels, $C$ rather than the entire vocabulary. For example, to classify heart disease with labels 'sick' and 'healthy', a prompt could be: "<clinical text> Patient is <MASK>", with a verbalizer mapping 'Healthy': 'fine', 'Sick': 'unwell'. The LLM predicts the <MASK> token, which is mapped to the corresponding class.

Within the broad prompt learning framework, there are two key design choices between manual templates and verbalizers, and soft templates and verbalizers, detailed below:

#### 4.4.0.1 Manual prompt learning

Manual prompt learning uses entirely handcrafted prompt templates and verbalizer label mappings. A manual template has the form:

$$\boldsymbol{x'} = \{[P_0, P_1, \ldots, P_j], \boldsymbol{x}, [P_{j+1}, P_{j+2}, \ldots, P_k], [\text{MASK}]\}$$

where for $j \in \{0, 1, \ldots, k\}$, $P_j$ denotes the token of the template. The set of words in the verbalizer for each class is denoted as $y \in \mathcal{C}$ to be $\mathcal{V}^y$, which means each class may have multiple assigned words. The verbalizer maps each word to each class $g' : \mathcal{V} \mapsto \mathcal{V}^C$, where $\mathcal{V}^C$ is the set of classes with their assigned words. The

probability of each class given the input $\boldsymbol{x}$ and its prompt $\boldsymbol{x'}$ is thus:

$$P(y \mid \boldsymbol{x}) = \frac{\exp\left(\frac{1}{|\mathcal{V}^y|}\sum_{t\in\mathcal{V}^y} P_M\left(t \mid \boldsymbol{x'}\right)\right)}{\sum_{i=1}^{|\mathcal{C}|}\exp\left(\frac{1}{|\mathcal{V}^i|}\sum_{t\in\mathcal{V}^i} P_M\left(t \mid \boldsymbol{x'}\right)\right)}.$$

Manual templates and verbalizers are discrete and bound to the LLM's pre-trained vocabulary, so there are no additional parameters to train, although fine-tuning the base LLM is still possible.

An illustration of manual prompt learning is given in Figure 16.

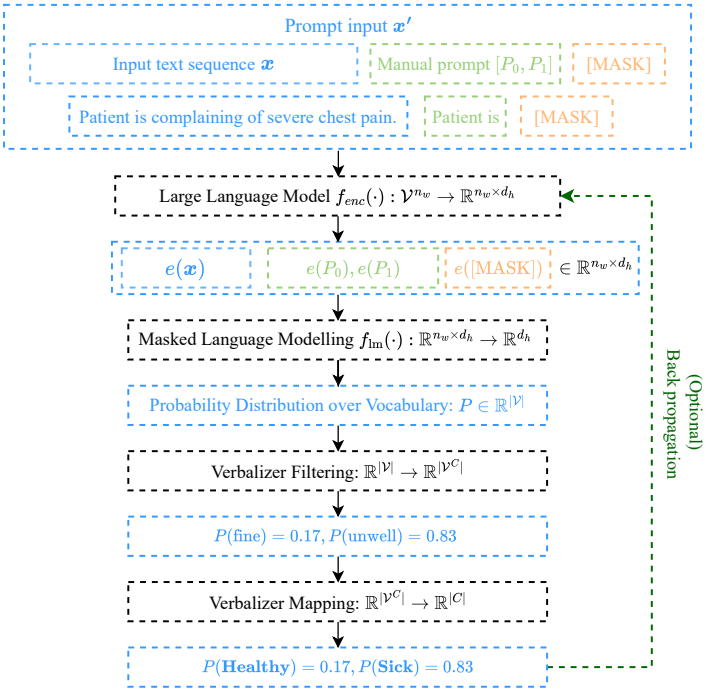


**Figure 16** Illustration of the manual template and verbalizer components in the prompt learning framework.

#### 4.4.0.2 Soft prompt learning

Soft prompt learning operates similarly to the manual approach but replaces fixed manual components with trainable or soft embeddings (continuous vectors) of the same dimension as the original LLM embeddings. The error from the downstream task can then be back-propagated to update only these new embeddings of the soft template and verbalizer. The soft prompt template is achieved by using trainable embeddings via the following form:

$$\boldsymbol{x'} = [S_0, S_1, \ldots, S_j], \boldsymbol{x}, [S_{j+1}, S_{j+2}, \ldots, S_k], [\text{MASK}]$$

where for $j \in 0, 1, \ldots, k$, $S_j$ denotes the token of the soft template. As $\boldsymbol{x'}$ is fed to the LLM to get $\mathbf{E}'$, the prompt tokens $S_i$ are also mapped to the embedding space, where one can assume $e(S_i)$ to be optimized during training and such tokens are denoted as <[soft]> in the template format. Optionally, $S_i$ can be initialized from an existing word token embedding from the LLM, like in a manual template, or a random vector with the same dimension.

A template where all tokens are <[soft]> is called a soft template, while a template with a mixture of manual and <[soft]> tokens is called a mixed template. An illustration of the data flow for a mixed template and soft verbalizer setup is presented in Figure 17.

The soft verbalizer can be interpreted as replacing assigned class label words from the manual verbalizer with trainable vectors (embeddings) for each class. Therefore, when using the soft verbalizer, there is no need to build the mapping from vocabulary $\mathcal{V}$ to class labels $\mathcal{C}$. A caveat of soft verbalizers is that these trainable

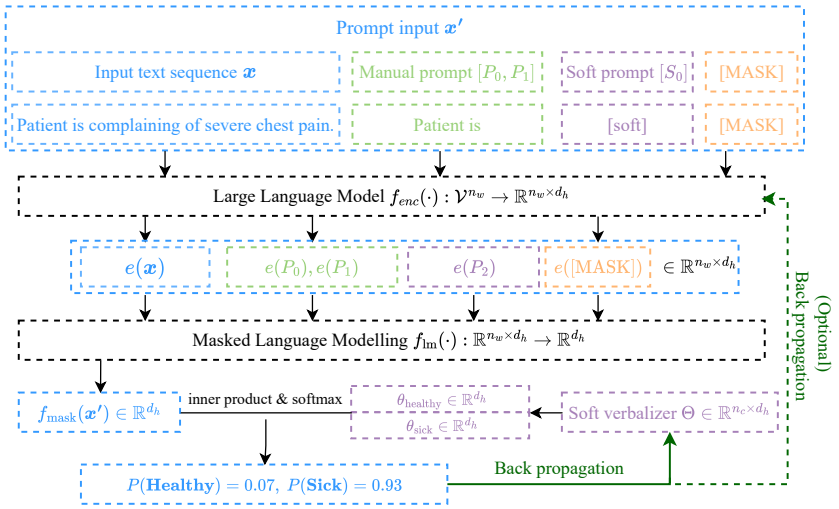


**Figure 17** Illustration of a mixed template and soft verbalizer in the prompt learning framework. If the <[soft]> token $S_0$ is not defined manually in advance, the embedding $e(S_0) \in \mathbb{R}^{d_h}$ will be randomly initialized.

vectors do not implicitly have semantic meaning, as they were not trained through a language modelling objective. The resulting verbalizer then becomes a matrix operator $\Theta \in \mathbb{R}^{n_c \times d_h}$, where $n_c$ represents the number of classes. The $i$-th row of $\Theta$ is represented by $\theta_i$ for each trainable embedding of class $i$. Then the output is processed with $f_{\text{lm}} : \mathbb{R}^{w \times m} \to \mathbb{R}^{d_h}$, where $n_w$ is the sequence length of $\boldsymbol{x'}$. Therefore, the probability of class $y$ given the input $\boldsymbol{x}$ and its prompt $\boldsymbol{x'}$ can be calculated by

$$P(y \mid \boldsymbol{x}) = \frac{\exp\left(\theta_y^\top f_{\text{lm}}(e(\boldsymbol{x'}))\right)}{\sum_{i=1}^{n} \exp\left(\theta_i^\top f_{\text{lm}}(e(\boldsymbol{x'}))\right)}.$$

### 4.4.1 Pre-trained LLMs

I chose the base model for the majority of the prompt learning experiments to be BioClinicalBERT (*47*, *49*): a clinical LLM pre-trained on a large collection of PubMed abstracts and full articles (*47*), followed by further training on all MIMIC-III notes.

The selection was intended to focus on smaller, efficient LLMs, whilst using an already domain-adapted model for clinical text. Therefore, this model should be well suited to the downstream tasks and allow a fair comparison between prompt learning and traditional fine-tuning.

In an extended analysis, I also explore alternative LLMs with prompt learning in section 4.5.6.

### 4.4.2 Datasets and tasks

I use several sequence classification tasks derived from the MIMIC-III dataset, covered in the general methods Section 4.3.1. As a reminder, these are:

- ICD-9 50 - assign each document with the prevalent ICD-9 diagnosis associated with the hospital episode
- ICD-9 Triage classification - deciding a triage team pathway from discharge summaries
- Mortality Prediction (MP) - predicting the mortality status of a patient based on discharge summaries

- Length of Stay prediction (LoS) - predicting the patient's length of stay binned into short, medium, long and very long

### 4.4.3 Prompt examples

Examples of different prompt methods are shown below. For each task, I show one manual prompt template and one mixed template. The <[soft]> token represents the trainable continuous vector or embedding of the mixed template that has been initialised from the LLMs vocabulary. Thus, <[soft]>:"This" indicates a soft embedding initialised from the LLMs pre-trained embedding for the token "This".

#### ICD-9 triage

- <clinical note> Best department is <[MASK]>.
- <clinical note> <[soft]>: "This" patient should <[soft]>:"go to this medical team based on symptoms of their illness" <[MASK]>.

#### Mortality prediction

- <clinical note> Patient is on the path to <[MASK]>.
- <clinical note> <[soft]>: "This" patient <[soft]>:"on path to" <[MASK]>.

#### ICD-9 50

- <clinical note> Patient has diagnosis <[MASK]>
- <clinical note> <[soft]>: “This” patient <[soft]>:“has diagnosis” <[MASK]>.

#### Length of stay prediction

- <clinical note> The patient will be at hospital with a <[MASK]> length.
- <clinical note> <[soft]>: “This” patient <[soft]>:“will be in hospital for a” <[MASK]> length.

The selection of the manual prompt text, the prompt engineering, was not explored to any great extent as the number of possible formats is vast. Instead, I focused on simplicity and ease of use, leaving the exploration of alternative prompting strategies for future work.

## 4.5 Prompt Learning Results

### 4.5.1 Comparison of different prompt learning setups

The number of possible prompt and verbalizer combinations in the prompt learning framework is vast. So, previous research guided the most suitable setups for this work. Evaluation results on the MIMIC-III ICD-9 Triage task comparing six prompt learning combinations are provided in Table 14. The setups were combinations of manual, mixed, or soft templates with manual or soft verbalizers.

| PLM params | Prompt combination(T,V) | Balanced accuracy |
|---|---|---|
| Fine-tuned | (manual, manual) | 0.8765 |
| | (manual, soft) | 0.8818 |
| | (mixed, manual) | 0.8817 |
| | (mixed, soft) | 0.8824 |
| | (soft, manual) | 0.8860 |
| | **(soft, soft)** | **0.8954** |
| Frozen | (manual, soft) | 0.7524 |
| | (mixed, manual) | 0.8474 |
| | (mixed, soft) | 0.8724 |
| | (soft, manual) | 0.8591 |
| | **(soft, soft)** | **0.8900** |

**Table 14** Table comparing different prompt learning setups on ICD-9 Triage task. PLM params refers to whether the PLM body was frozen or fine-tuned during training. For prompt combinations (T,V): *T* represents template and *V* represents verbalizer.

The prompt template and verbalizer combinations' evaluation performance was similar when fine-tuning the LLM, but large differences emerged when freezing it. This relates partly to the varying trainable parameter counts introduced by each setup (manual components introduce none) and the known prompt engineering limitation where subtle prompt changes impact performance (*183*). While the soft template and soft verbalizer combination performed best overall, the mixed template and soft verbalizer were chosen as the prompt learning benchmark going forward. The mixed template allows injecting interpretable domain-specific prompt tokens while leveraging trainable soft token embeddings. See section 4.4.3 for mixed template examples used.

### 4.5.2 Prompt learning versus classification head

For simplicity, I will refer to traditional fine-tuning as the classification head throughout. Each framework utilises the same LLM, and results for both fine-tuning and freezing the entire LLM are presented. In the case of the frozen LLM, only the parameters introduced by the classification head or prompt learning components are updated during training. Evaluation results across each task and approach are provided in Table 15, followed by a further comparison of these same setups against training sample sizes in Fig. 18.

| Method | LLM | ICD-9 50 | ICD-9 Triage | LoS | Mortality |
|---|---|---|---|---|---|
| LLM + Classification head | Fine-tuned | **0.663** | 0.885 | **0.712** | **0.831** |
| LLM + Prompt learning | Fine-tuned | 0.639 | **0.888** | 0.699 | 0.799 |
| LLM + Classification head | Frozen | 0.166 | 0.492 | 0.634 | 0.752 |
| LLM + Prompt learning | Frozen | <u>0.619</u> | <u>0.876</u> | <u>0.647</u> | <u>0.773</u> |

**Table 15** Prompt learning results for CLS tasks. The reported metric for each task is as follows: weighted F1 score for ICD-50, macro averaged F1 score for ICD-9 Triage, and macro averaged Receiver Operator Characteristic Area Under the Curve (ROC AUC) for the MIMIC-III LoS and Mortality tasks. The best-performing setup for each task is **boldfaced**, and the best-performing frozen setup is <u>underlined</u>

It can be seen that prompt learning can match or improve on using a classification head, with a much smaller gap in performance between the frozen and fine-tuned LLM setting across few-shot and full training setups.

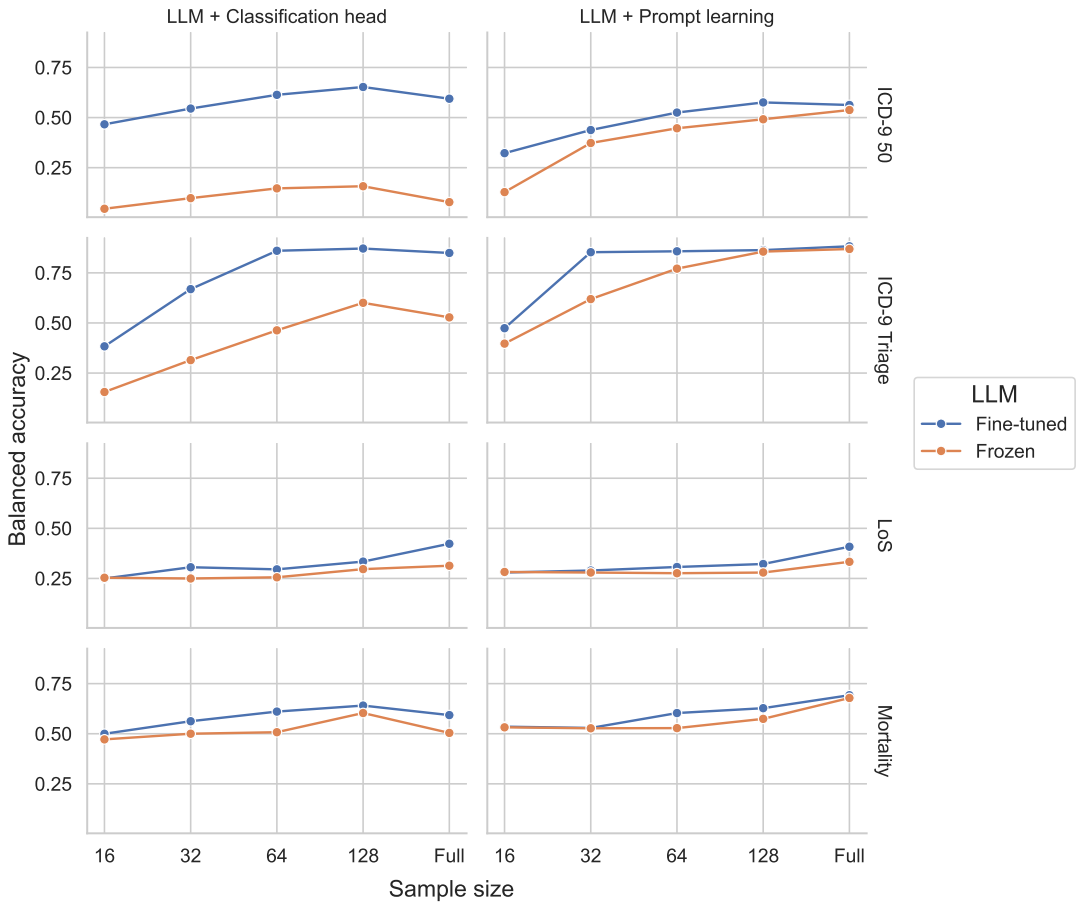


**Figure 18** Balanced accuracy for training the LLM with prompt learning (left) and a classification head (right) across the four clinical tasks given by each row. The "Full" sample size refers to a full training dataset which varies in size from task to task. The colour of the line refers to whether the underlying LLM was frozen or fully fine-tuned during training.

### 4.5.3 Training hyperparameter search

Prompt learning and traditional fine-tuning have large neural network hyperparameter spaces, in terms of choice of learning rate, optimizer, batch size and so on. However, with prompt learning, there are additional modelling choices concerning prompt engineering, including the prompt and verbalizer setups etc. Initial exper-

iments used sensible hyperparameters and prompt setups from previous research (*35*, *168*), achieving similar performance when fine-tuning the LLM. However, with frozen LLMs, prompt learning appeared superior despite substantially fewer task-specific trained model parameters. To ensure performance differences were not due to poor hyperparameter choices, a hyperparameter search within single GPU capabilities was conducted on the ICD-9 Triage task, with the search space provided in Table 16.

| **Parameter** | **Search space** |
|---|---|
| classifier learning rate | log.uniform$[1 \times 10^{-5}, 3 \times 10^{-1}]$ |
| batch size | $[4, 8, 16]$ |
| gradient accumulation steps | range$[2, 10]$ |
| dropout | range$[0.1, 0.5]$ |
| optimizer | categorical[adamw, adafactor] |
| prompt learning rate | log.uniform$[1 \times 10^{-5}, 3 \times 10^{-1}]$ |
| verbalizer learning rate | log.uniform$[1 \times 10^{-5}, 1 \times 10^{-1}]$ |

**Table 16** Hyperparameter search space used for optmization of prompt learning and classification head fine-tuning

The results of the subsequent optimized models for the ICD-9 Triage task are presented in Table 17. Further details of the hyperparameter search and results are presented in Appendix B.4. With these optimised parameters it is clear that both methods, prompt learning and traditional fine-tuning, improved and reduced the gap in performance between them. Prompt learning maintains an improvement above traditional fine-tuning and can learn with much fewer model parameters. This may be due to the better alignment of the task through prompting to the LLMs pre-training, in contrast to the use of classification heads in traditional fine-tuning.

| Method | LLM | Balanced accuracy | F1 weighted | AUC |
|---|---|---|---|---|
| LLM + Classification head | Frozen | 0.8162 | 0.8919 | 0.9811 |
| LLM + Prompt learning | Frozen | **0.8698** | **0.9246** | **0.9889** |

**Table 17** Evaluation metrics for the ICD9 triage task after training with derived optimal hyperparameters using prompt learning or a classification head with frozen LLMs

### 4.5.4 Sensitivity analyses

A possible explanation for the performance drop with traditional fine-tuning may be overfitting to the training data, incurred by the larger number of parameters (introduced by the classification head). To probe this, I varied the number of additional parameters introduced and compared the performance on the ICD-9 Triage task, see Figure 19. Concretely, adjusting the number of trainable parameters for traditional fine-tuning involves adjusting the number of layers and hidden dimension size of the classification head. For prompt learning, adjusting the number of trainable parameters requires just changing the number of soft template tokens and whether to include a soft verbalizer. A training set of 128 samples per class was used as this approached peak performance without requiring a full training run.

Note that prompt learning with the fewest trainable parameters (N params = 1,536) achieves comparable performance to the traditional fine-tuning model with 1000 times the number of trainable parameters (N params = 1,552,007). This is a large improvement in efficiency and can vastly reduce the storage requirements for adapting the LLM to new tasks.

For prompt learning, the mixed template was a crucial component. To determine whether mixed templates benefit from a common sense or domain-specific tem-

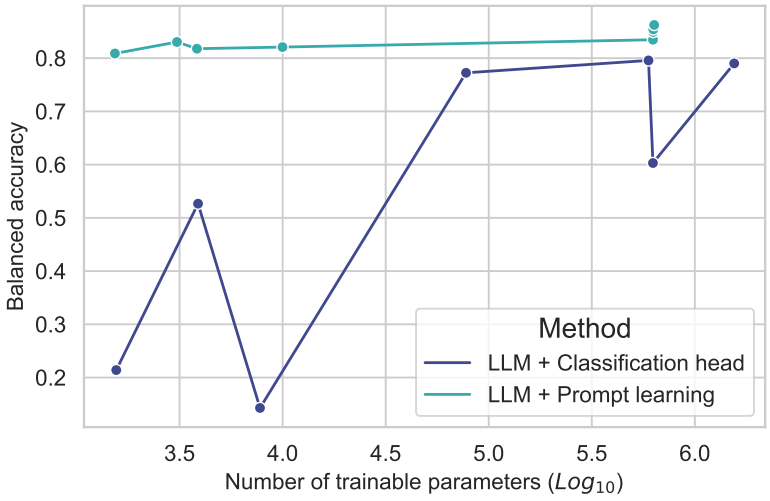


**Figure 19** Balanced accuracy for prompt learning versus traditional fine-tuning with a classification head, each with an increasing number of trainable parameters on top of frozen LLMs. For readability, a logarithmic scale is used on the $x$-axis.

| **Prompt text** | **Balanced accuracy** |
|---|---|
| <[soft]>: "This" <[MASK]> | 0.8195 |
| <[soft]>: "This" patient <[soft]>:"should go to" <[MASK]>. | 0.8539 |
| <[soft]>: "This" patient should <[soft]>:"go to" <[MASK]>. | 0.8491 |
| <[soft]>: "This" patient should <[soft]>:"go to this medical team based on symptoms of their illness" <[MASK]>. | **0.8624** |
| random words here <[soft]>:"random" <[MASK]>. | 0.8346 |

**Table 18** Balanced accuracy for different mixed prompt templates for the ICD-9 Triage task with the LLM frozen.

plate, I compared the performance of different templates. Results are shown in Table 18. We can see that the more domain-specific prompts have higher performance, and a single soft token or random tokens lead to a drop in performance.

This suggests that the LLM is partly dependent on the prompt template and is linked to the domain pre-training the LLM received.

### 4.5.5 Efficiency analysis

The additional parameters introduced by prompt learning and the classification head require storage an order of magnitude smaller than the LLM itself. When keeping the LLM frozen during training for downstream tasks, these additional parameters can remain task-specific and portable while preserving a static LLM. This enables efficient re-use of the same base model for different tasks and datasets. Table 19 compares the number of introduced trainable parameters for traditional fine-tuning and prompt learning with comparable evaluation performance.

| Trained params | # trained params | MB | Train time(secs) | GPU vRAM(GB) |
|---|---|---|---|---|
| Soft prompt | 7144 | 0.012 | 83 | 4.6 |
| Classification head | 595,975 | 2.26 | 90 | 5.2 |
| LLM + soft prompt | $108.3 \times 10^6$ | 415.7 | 110 | 11.2 |
| LLM + classification head | $108.9 \times 10^6$ | 475.5 | 115 | 11.6 |

**Table 19** Memory storage, training times in Mega Bytes (MB) and GPU vRAM required for parameters introduced by prompt learning and a classification head used for fine-tuning BioClinicalBERT on the ICD-9 triage task. Trained params refer to the set of model parameters that are updated with respect to the downstream task during training. Training times are based on 1 epoch with batch size of 8

The computational efficiency is highest for the soft prompts, whilst outperforming the classification head only setup by a reasonable margin (as shown in Fig 19). This implies that even when tuning a similar number of parameters, prompt learning encourages better learning of the task.

### 4.5.6 Alternative LLM results

The final set of results in Table 20 showcases the performance of alternative LLMs not used in the main study results. The purpose was to determine how transferable the prompt learning or traditional classification head approaches were and are not an exhaustive set of results.

| Model | Traditional fine-tuning | Prompt learning |
|---|---|---|
| ClinicalBioBERT* (*44*) | 0.8612 | 0.8698 |
| BERT-base-uncased (*2*) | 0.2541 | 0.6301 |
| RoBERTa-base (*134*) | 0.3451 | 0.7989 |
| GPT-2 (*184*) | 0.3812 | 0.8613 |
| OPT-125m (*185*) | 0.3233 | 0.7231 |

**Table 20** Balanced accuracy for prompt learning and traditional fine-tuning with frozen non-clinical PLMs on the ICD-9 Triage task. The best-performing prompt learning setup from the main experiments of a mixed template with a soft verbalizer was used. These results used a sample size of 128 per class for training and evaluated using the whole test set. * represents the best-performing results using ClinicalBioBERT model from the main experiments. Citations are in the brackets.

Following this, I provide results using much larger LLMs including Llama-2-7b (*3*) and Tiny-Llama-1b (*186*). Results for each of the MIMIC-III classification tasks are presented in Table 21.

The larger LLMs surpass the performance of the smaller BioClinicalBERT model - but with substantial increases in the model size: Tiny-Llama-1b has 1.1 billion, and Llama-2-7b has 6.7 billion versus the 108 million parameters of BioClinicalBERT. It must be noted that it is also difficult to determine the exact training samples used for the pre-training of the Llama models due to the vast size of their training datasets, which may include examples from the MIMIC-III dataset.

| Model | LLM | ICD-9 50 | ICD-9 Triage | LoS | Mortality |
|---|---|---|---|---|---|
| ClinicalBioBERT* | Frozen | 0.619 | 0.876 | 0.647 | 0.773 |
| Llama-2-7b | Frozen | 0.644 | 0.886 | 0.701 | 0.803 |
| Tiny-Llama-1b | Frozen | 0.634 | 0.868 | 0.689 | 0.805 |

**Table 21** Prompt learning results for CLS tasks. The reported metric for each task is as follows: weighted F1 score for ICD-50, macro averaged F1 score for ICD-9 Triage, and macro averaged Receiver Operator Characteristic Area Under the Curve (ROC AUC) for the MIMIC-III LoS and Mortality tasks.

## 4.6 Prompt learning discussion

The experiments directly compared adapting an LLM to a downstream clinical task using prompt learning or a traditional classification head. The objective was to assess if prompt learning's performance on general domain text translates to the clinical domain with potential efficiency improvements.

Four clinical sequence classification tasks using the MIMIC-III dataset related to patient outcomes were presented in full training and few-shot setups with varying trainable parameters. Prompt learning could typically match and occasionally outperform traditional fine-tuning. Notably, when the LLM was frozen, prompt learning outperformed traditional fine-tuning with considerably fewer trainable parameters (Fig. 19).

The selection of a manual prompt had a large impact on performance, with even subtle changes to the prompt yielding different performances. A mixed template was able to introduce the capacity for the model to learn soft prompts more flexibly, whilst retaining an element of domain expertise. However, even with a mixed template, there were fluctuations in performance that limited the ability to apply

templates to multiple tasks. A combination of entirely soft or trainable prompts and verbalizers allows the most flexibility, with the loss function driving the performance entirely. However, the interpretability of the soft prompt learning setup is more challenging as the so-called prompts no longer map to a known vocabulary.

The exact reason prompt learning performs better by injecting trainable parameters within the LLM's input rather than a classification layer is unclear. One postulation is that formatting the downstream task in a prompting setup better aligns with the LLM's pre-training objective. Further research is required to investigate this extensively.

The trade-off between performance, training efficiency, portability, and LLM reusability is important in resource-light environments where data may change frequently, making training and storing only task-specific parameters more desirable than retraining the entire LLM.

### 4.6.1 Limitations

**Pre-training data leakage** The choice of BioClinicalBERT (*49*) as the clinical LLM for the reported MIMIC-III tasks, pre-trained on MIMIC-III notes, may have made the tasks easier or inflated the evaluation performance. This is especially true for prompt learning, which reframes the downstream objective similarly to the pre-training objective. Nevertheless, prompt learning still outperformed traditional fine-tuning with other clinical or non-clinical LLMs where data leakage is not an issue.

**Task performance variance** The four presented clinical tasks derived from MIMIC-III notes data achieved results in line with previous research (*57*). However, the hyperparameters used for prompt learning were incredibly important and the prompts and verbalizer components directly influenced task performance. This is why soft prompts and verbalizers are generally more desirable, as they have flexibility and can be trained for the task directly. However, the interpretability of these soft prompts and verbalizers is more difficult as they are not mapped to known vocabulary tokens. The sensitivity to hyperparameter changes is also present for traditional fine-tuning, but there are arguably fewer working components to consider compared to prompt learning.

**Task type** A major limitation of standard prompt learning was the difficulty of handling token classification tasks like NER, primarily due to smaller LLMs' limited context windows not allowing multiple prompts per input. For sequence classification with one target label, the prompt requests one target response. For NER, each token requires a prompt and label mapping. Even larger LLMs struggle when simply prompted to generate a list of found entities (*40*, *43*). Recent works have sought to improve LLMs' NER capabilities through prompting, with contrastive loss functions (*187*, *188*) or finding suitable prompts/instructions (*189*).

### 4.6.2 Prompt learning conclusion

Overall, this work showed that prompt learning outperformed a traditional fine-tuning approach when the LLMs are frozen during training on the downstream task. Prompt learning also requires fewer trainable parameters to achieve superior

performance when compared to training a classifier head with a frozen LLM.

Next, I will present the alternative PEFT methods and results.

## 4.7 PEFT methods

### 4.7.1 Prefix tuning

This approach (*105*) prepends a prefix to the input and every layer of the encoder stack. Given a model $LLM_\phi$, that is parameterised by $\phi$, the activation of any time step $i$ can be considered as $h_i = [h_i^1; h_i^2; \ldots; h_i^j], \in \mathbb{R}^d$, where the concatenation of activation's $h_i^j$ from each layer $j$ of the encoder at time step $i$. For any given time step, $h_i$ can be obtained as $LLM_\phi(x_i, h_{<i})$, a function of input $x_i$ and the activation's $h_{<i}$, from all time steps before $i$. After prepending the prefix, the hidden state activation's $h_i$ of the model are modified to

$$h_i = \begin{cases} P_\theta[i,:] & \text{if } i \leq p \\ LLM_\phi(x_i, h_{<i}) & \text{otherwise} \end{cases}$$

where, $P_\theta \in \mathbb{R}^{p \times d}$ is the trainable matrix that forms the prefix, parameterised by $\theta$, and $p$ is the length of the prefix.

### 4.7.2 Prompt tuning

Within the PEFT setup, prompt tuning (*4*) is very similar to prompt learning, but without the verbalizer nor a mixed format of manual prompts alongside soft prompts. The method uses a matrix of soft prompts, $P_e \in \mathbb{R}^{p \times d_h}$, that are concatenated with the embedded representation of the input, $X_e \in \mathbb{R}^{n_w \times d_h}$. Here, $p$ and $n_w$ represent the length of the soft prompt and the input respectively, and $d_h$ represents the embedding dimension. The effective input to the model becomes $[P_e; X_e] \in \mathbb{R}^{(p+n) \times d_h}$ and only $P_e$ is updated. Conceptually, prompt tuning is similar to prefix tuning, with the exception being that the soft prompts are introduced only in the input layer and not in each encoder layer like in the case of prefix tuning.

### 4.7.3 P-tuning

P-tuning (*190*) is an extension of prompt tuning that also adds a prompt encoder, $f$, which would map the initial prompt embedding $f : [P_e] \rightarrow \mathrm{h}$ (we found an LSTM or simple MLP worked well). This added flexibility and non-linearity appeared to be more robust than the standard prompt tuning approach. Unlike prefix tuning, the soft prompts can be inserted anywhere in the sequence.

### 4.7.4 Low-Rank Adaptation (LoRa)

LoRA (*106*) is a reparametrisation technique that works by injecting two trainable matrices ($A$ and $B$) that act as an approximation of a singular value decomposition (SVD) of the weight update $\Delta W$ for any weight matrix $W \in \mathbb{R}^{d \times k}$ in the LLM. The approximation works as $\Delta W = BA$, where $B \in \mathbb{R}^{d \times r}$, $A \in \mathbb{R}^{r \times k}$ and $r \ll min(d, k)$ is the rank of the LoRA matrices, which is a tunable parameter. The new forward pass is updated from $h = Wx$ (where $x$ is the input embedding to the layer/operation and $h$ the output embedding) to $h = (W + \Delta W)x = (W + AB)x = Wx + ABx$. While it is possible to introduce the LoRA matrices in any layer of the LLM, it is common practice to introduce them as weight update approximations for the key, query and value matrices. The underlying assumption is that the weight updates in LLMs intrinsically have a lower rank than their dimensions, and thus can be well approximated by their SVD. Additionally, once fully trained, the LoRA matrices can be integrated into the model as $W_{updated} = W_0 + BA$, thereby introducing no inference latency. With LoRA the original weight matrices of the LLM remain frozen during the fine-tuning phase.

### 4.7.5 $IA^3$

Infused Adapter by Inhibiting and Amplifying Inner Activation ($IA^3$) shares similarities with other adapter methods that introduce new parameters to scale activations using learned vectors (*170*). While these learnable vectors can be applied to any set of activations, applying them to the keys and values in the relevant attention mechanism and the intermediate activation of the position-wise feed-

forward networks was found to be both efficient and sufficient. For a transformer based architecture, we have a query $Q \in \mathbb{R}^{d_q}$, key $K \in \mathbb{R}^{d_k}$, value $V \in \mathbb{R}^{d_v}$, and a position-wise feed-forward network with hidden dimension $d_{ff}$. $IA^3$ introduces learnable vectors $l_k \in \mathbb{R}^{d_k}$, $l_v \in \mathbb{R}^{d_v}$ and $l_{ff} \in \mathbb{R}^{d_{ff}}$ and modifies the attention and feed-forward calculation as follows:

$$softmax\left(\frac{Q(l_k \odot K)}{\sqrt{d_k}}\right)(l_v \odot V) \tag{4.1}$$

$$(l_{ff} \odot \gamma(W_1 x))W_2 \tag{4.2}$$

where $\odot$ represents the element-wise product, and $\gamma$, $W_1$ and $W_2$ are the activation function and weight matrices of the feed-forward network. $W_1$ is a matrix of dimension $d_{ff} \times d_v$, and $W_2$ is of dimension $d_v \times d_{ff}$. The equations use Numpy's 'broadcasting notation' (*191*) where the (i, j)th entry of l⊙x is $l_j.x_{i,j}$. Similar to LoRA, the learnable vectors can be merged into the model as $l \odot W$ because any operation $l \odot Wx$ is equivalent to $(l \odot W)x$. Hence, this method does not introduce any inference latency either. Once again, with $IA^3$ the original weight matrices of the LLM remain frozen during fine-tuning.

### 4.7.6 Model architectures

I evaluated the performance of PEFT methods across various transformer-based LLM's architectures of differing sizes; including TinyBERT (*192*), MobileBERT (*193*), DistilBERT (*194*) and standard BERT (*2*). These are comparatively small LLMs (some would now refer to these as SLMs), for optimal efficiency and aligned with the LLMs used in previous chapters. However, I did run additional ex-

periments with the much larger LLM Llama-2-7b (*3*) to probe the efficiency-performance-cost trade-off. A table of relevant architecture details is provided in Table 22.

| Model architecture | # Params (mil) | GPU (VRAM GB) | FLOPs |
|---|---|---|---|
| Tiny-BERT | 13.87 | 0.052 | $3.66 \times 10^7$ |
| Mobile-BERT | 24.58 | 0.092 | $1.62 \times 10^8$ |
| Distil-BERT | 65.78 | 0.245 | $3.41 \times 10^8$ |
| BERT | 108.31 | 0.403 | $6.81 \times 10^8$ |
| Llama2-7b | 6607.34 | 24.6 | $5.18 \times 10^{10}$ |
| Llama2-7b (bfloat16) | 6607.34 | 12.37 | $5.18 \times 10^{10}$ |

**Table 22** Model architectures and their associated number of parameters, Video Random Access Memory (VRAM), and Floating Point Operations (FLOPs). FLOPs were based on a random sample of 10 tokens.

The reported VRAM relates to the amount of memory required of the GPU to load the LLM model weights into memory to allow any input to be passed through the model in inference mode. FLOPs are a related metric of the total number of floating point operations performed during a forward pass.

### 4.7.7 Downstream fine-tuning

To keep these PEFT methods separate from prompt learning, I used the traditional fine-tuning approach with a classification head, as detailed in section 2.5.5. This approach is common practice when applying PEFT methods for sequence and token classification tasks and remains the most suitable across all model architectures and aligns with previous research (*11*, *45*).

### 4.7.8 Domain pre-training

In addition to the size of the LLM, I examined three domain variants for each from previous research where model checkpoints have been released and are available to download via HuggingFace (*35*):

- **General:** Original, unadapted models (*2*, *192–194*).
- **Biomedical:** Models pre-trained or distilled with biomedical literature (*195*)
- **Clinical:** Models pre-trained with clinical EHR data, in this case MIMIC-III2.5 (*45*)

The objective was to probe the potential benefits to efficiency gained from domain pre-training and uncover the interplay between domain pre-training, model size, and the chosen PEFT methods - which should provide a holistic view of efficiently adapting LLMs to downstream tasks.

### 4.7.9 Datasets and tasks

For the PEFT work, I utilised the following tasks, outlined in detail in the general methods Section 4.3.1. For clarity, I will briefly summarise the tasks used here:

### Sequence classification tasks

- ICD-9 Triage classification - deciding a triage team pathway from discharge summaries
- Mortality Prediction (MP) - predicting the mortality status of a patient based on discharge summaries
- Length of stay prediction - predicting the patient's length of stay binned into short, medium, long and very long
- I2B2 Relation Extraction - classifying sentences or documents based on the medical relationship

### NER

- I2B2 2010 and 2012 - Identifying certain medical and temporal events with documents
- I2B2 2014 - A de-identification task, whereby spans of text within clinical notes are classified using different protected health information (PHI) such as name, address, and postcode

### 4.7.10 Hyperparameters

The default hyperparameters used for the main PEFT experiments are provided below in Table 23. Later, I present an analysis after optimisation of hyperparameters for select PEFT methods.

| PEFT | Hyperparameter | Value |
|---|---|---|
| Prefix-Tuning \| P-Tuning | dropout | 0.1 |
| | learning rate | $3e-4$ |
| | # virtual tokens | 20 |
| | layers | all |
| Prompt-Tuning | dropout | 0.1 |
| | learning rate | $3e-4$ |
| | # virtual tokens | 20 |
| | layers | 1 |
| LoRA | r | 8 |
| | alpha | 8 |
| | dropout | 0.1 |
| | learning rate | $3e-4$ |
| | target modules | [key, value] |
| | layers | all |
| $IA^3$ | dropout | 0.1 |
| | learning rate | $3e-4$ |
| | target modules | [key, value, feed-forward] |
| | layers | all |

**Table 23** The default hyperparameters for LoRA and $IA^3$ were used in all experiments prior to the hyperparameter optimisation. For full fine-tuning, the same learning rate ($3e-4$) and dropout (0.1) were used.

## 4.8 PEFT results

Initial experiments focused on the LLMs of different sizes that had received biomedical domain pre-training, as opposed to clinical domain pre-training. I chose this to avoid having models that were potentially pre-disposed to some of the

clinical texts contained with the downstream tasks. For instance, BioClinicalBERT (*44*) was pre-trained on all MIMIC-III notes.

### 4.8.1 All PEFT methods

A comparison of all PEFT methods for each model size is provided in Table 24.

| Model name | PEFT method | ICD9-Triage | MIMIC LoS | MIMIC MP | i2b2-2010-RE |
|---|---|---|---|---|---|
| BioBERT | Full | 0.876 | 0.582 | 0.828 | 0.934 |
| BioBERT | $IA^3$ | 0.733 | 0.628 | 0.767 | 0.877 |
| BioBERT | LoRA | **0.833** | **0.685** | **0.819** | **0.918** |
| BioBERT | Prefix | 0.289 | 0.574 | 0.637 | 0.818 |
| BioBERT | Prompt | 0.269 | 0.584 | 0.638 | 0.789 |
| BioBERT | P-tuning | 0.487 | 0.617 | 0.736 | 0.865 |
| BioDistilBERT | Full | 0.858 | 0.612 | 0.670 | 0.877 |
| BioDistilBERT | $IA^3$ | 0.825 | 0.668 | 0.780 | 0.899 |
| BioDistilBERT | LoRA | **0.874** | **0.694** | **0.812** | **0.924** |
| BioDistilBERT | Prompt | 0.709 | 0.637 | 0.737 | 0.867 |
| BioDistilBERT | P-tuning | 0.847 | 0.665 | 0.778 | 0.889 |
| BioMobileBERT | Full | 0.879 | 0.690 | 0.824 | 0.928 |
| BioMobileBERT | $IA^3$ | 0.789 | 0.668 | 0.760 | 0.878 |
| BioMobileBERT | LoRA | **0.850** | **0.670** | **0.805** | **0.913** |
| BioMobileBERT | Prompt | 0.409 | 0.581 | 0.645 | 0.806 |
| TinyBioBERT | Full | 0.749 | 0.674 | 0.812 | 0.901 |
| TinyBioBERT | $IA^3$ | 0.417 | 0.603 | 0.611 | 0.787 |
| TinyBioBERT | LoRA | 0.601 | **0.645** | **0.759** | 0.881 |
| TinyBioBERT | Prefix | **0.819** | 0.638 | 0.738 | **0.888** |
| TinyBioBERT | Prompt | 0.153 | 0.581 | 0.608 | 0.789 |
| TinyBioBERT | P-tuning | 0.404 | 0.608 | 0.704 | 0.810 |

**Table 24** PEFT results for all sequence classification tasks using biomedical models. Micro-averaged F1 scores are reported for the i2b2-2010-RE. Macro-averaged Receiver Operating Characteristic area under the curve ($ROCAUC$) is used for MIMIC-LoS and MP tasks, while macro-averaged F1 scores are reported for the ICD-9 triage task. **Bold** results indicate best PEFT performance, and values underlined are top performance across all fine-tuning methods.

Certain PEFT methods were incompatible with particular LLM architectures: DistilBERT models were incompatible with prefix-tuning, and MobileBERT models were incompatible with P-tuning or prefix-tuning. These methods underperformed others and struggled with token classification tasks. Therefore, only the top-

performing PEFT methods, LoRA and $IA^3$, were kept for the remaining experiments as they worked for all models and easily adapted to token classification.

### 4.8.2 Model size vs PEFT

In Table 25, I present the evaluation results across the select PEFT methods and model sizes. Smaller LLMs ($<$ 100 million parameters) can be prone to instability during fine-tuning. Thus, for the next experiments, I ran 3 separate training runs with different random seeds for PyTorch. This will effectively randomly initialise certain weights of the models and training regime.

The results demonstrate that LoRA consistently outperforms $IA^3$ across the majority of models and tasks, often approaching the performance of full fine-tuning.

A comparison of the number of trainable parameters as a function of the different fine-tuning methods is shown in Figure 20. An efficient, well-performing model would be low on the x-axis but high on the y-axis (perfect would be the top left of the respective plot). There is a clear correlation between the number of trainable parameters and performance, and LoRA appears to provide larger models with an advantage over fully fine-tuned smaller models.

### 4.8.3 LoRA rank vs model size

Given LoRA's superior performance, the impact of its rank hyperparameter across models of varying sizes was evaluated. LoRA uses matrix rank reduction to approximate weight updates of the frozen LLM, with rank being a primary parameter

| Model name | PEFT | ICD9-Triage | i2b2-2010-RE | MIMIC-LoS | Mimic-MP |
|---|---|---|---|---|---|
| BioBERT | Full | 0.864 (0.002) | 0.935 (0.004) | 0.709 (0.002) | 0.819 (0.020) |
| | $IA^3$ | 0.703 (0.19) | 0.896 (0.004) | 0.634 (0.001) | 0.769 (0.005) |
| | LoRA | **0.827** (0.002) | **0.925** (0.001) | **0.697** (0.002) | **0.828** (0.002) |
| BioDistilBERT | Full | 0.862 (0.010) | 0.927 (0.003) | 0.706 (0.003) | 0.825 (0.006) |
| | $IA^3$ | 0.792 (0.008) | 0.906 (0.002) | 0.677 (0) | 0.797 (0.001) |
| | LoRA | **0.855** (0.005) | **0.928** (0.003) | **0.702** (0.001) | **0.825** (0.001) |
| BioMobileBERT | Full | 0.851 (0.004) | 0.932 (0.003) | 0.704 (0.004) | 0.819 (0.011) |
| | $IA^3$ | 0.744 (0.012) | 0.897 (0.003) | 0.639 (0.001) | 0.774 (0.002) |
| | LoRA | **0.808** (0.004) | **0.918** (0.002) | **0.671** (0.004) | **0.798** (0.002) |
| TinyBioBERT | Full | 0.727 (0.012) | 0.910 (0.005) | 0.684 (0.001) | 0.802 (0.001) |
| | $IA^3$ | 0.390 (0.035) | 0.852 (0.002) | 0.588 (0.003) | 0.607 (0.003) |
| | LoRA | **0.599** (0.008) | **0.895** (0.003) | **0.649** (0.006) | **0.764** (0.003) |

**(a)** Sequence classification task results

| Model name | PEFT | i2b2-2010-NER | i2b2-2012-NER | i2b2-2014-NER |
|---|---|---|---|---|
| BioBERT | Full | 0.819 (0.003) | 0.824 (0.001) | 0.967 (0.001) |
| | $IA^3$ | 0.473 (0.002) | 0.485 (0.006) | 0.850 (0.001) |
| | LoRA | **0.696** (0.003) | **0.753** (0.001) | **0.935** (0) |
| BioDistilBERT | Full | 0.803 (0.003) | 0.795 (0.006) | 0.967 (0.001) |
| | $IA^3$ | 0.498 (0.003) | 0.503 (0.001) | 0.883 (0) |
| | LoRA | **0.718** (0.008) | **0.729** (0.006) | **0.940** (0.001) |
| BioMobileBERT | Full | 0.796 (0.003) | 0.772 (0.006) | 0.966 (0) |
| | $IA^3$ | 0.515 (0.003) | 0.515 (0.003) | 0.908 (0) |
| | LoRA | **0.638** (0.010) | **0.650** (0.004) | **0.941** (0.001) |
| TinyBioBERT | Full | 0.655 (0.004) | 0.705 (0.008) | 0.906 (0.003) |
| | $IA^3$ | 0.328 (0.009) | 0.381 (0.003) | 0.715 (0.002) |
| | LoRA | **0.438** (0.007) | **0.561** (0.009) | **0.8051** (0.013) |

**(b)** NER task results

**Table 25** PEFT results for all downstream tasks using biomedical models, with values representing the median from 3 distinct training runs under varied random seeds for PyTorch weight initialisations. Standard Deviation ($SD$) is provided in brackets. Micro-averaged F1 scores are reported for the i2b2-2010-RE and all NER tasks. Macro-averaged Receiver Operating Characteristic area under the curve ($ROCAUC$) is used for MIMIC-LoS and MP tasks, while macro-averaged F1 scores are reported for the ICD-9 triage task. **Bold** results indicate best PEFT performance, and values underlined are top performance across all fine-tuning methods.

affecting trainable weights introduced. Using the Optuna package (*196*), 50 hyperparameter optimization trials were conducted, holding LoRA rank constant at $r \in 8, 16, 32, 64, 128$ while tuning LoRA dropout ($d \in 0.1, 0.3, 0.5$), LoRA alpha ($\alpha \in 0.3, 0.5, 1.0$), and learning rate ($lr \in [10^{-5}, 10^{-3}]$). The range of LoRA ranks

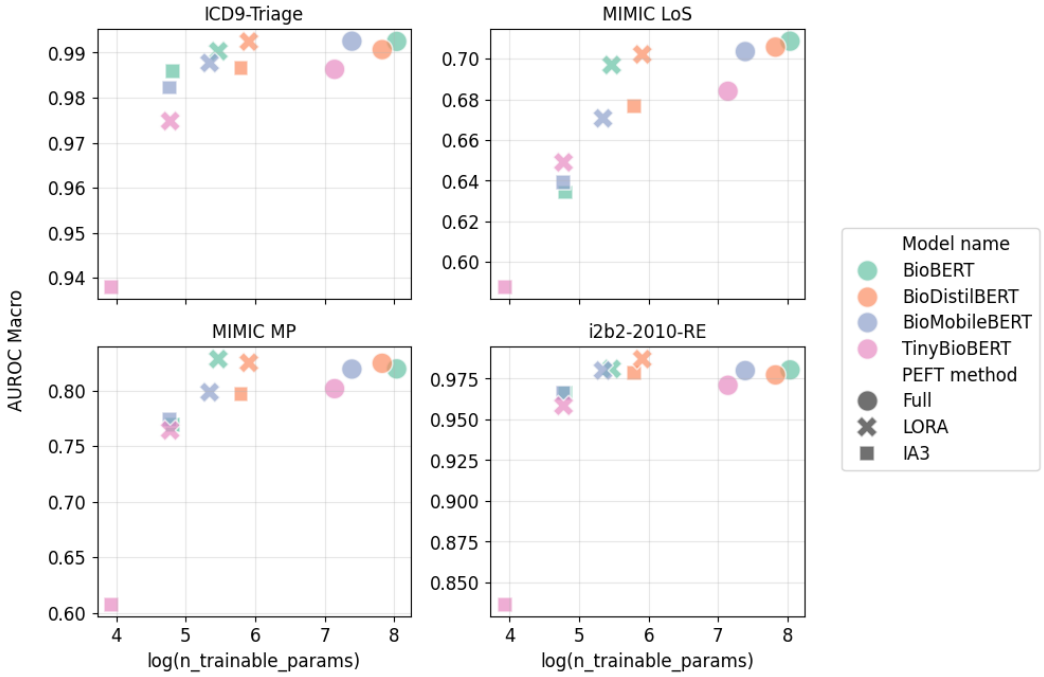


**Figure 20** Sequence classification performance across the different LLM model sizes and the associated number of trainable parameters (n_trainable_parameters) in the log scale. The colour of the markers reflects the model, and the shape of the marker reflects the fine-tuning method used (full or LoRa). For example, the orange circle is BioDistilBERT with full fine-tuning.

selected was in line with previous research for models of this size (*174*), and in this context covers a spread of values that maintain a substantial reduction in parameter update size. Increasing the rank beyond 128 would begin to lose value in terms of compute efficiency and were not explored. The Llama model was excluded due to its significantly larger size. Following the search, the optimal model for each $r$ value was selected to analyze a rank's effect on models with differing parameter counts.

Figure 21 shows increasing $r$ improved TinyBioBERT performance up to $r = 64$, then a slight decline at $r = 128$. BioDistilBERT followed a similar pattern, with a turning point at $r = 32$. BioMobileBERT's impact was more variable, poten-

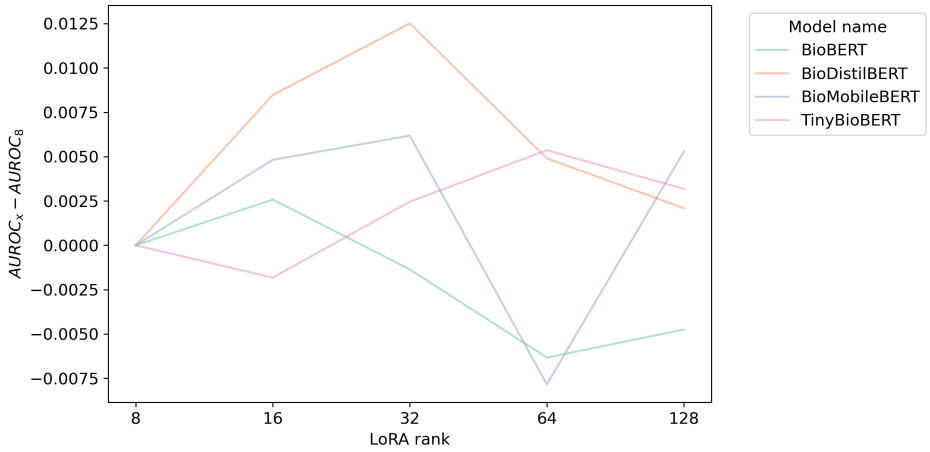


**Figure 21** Differential effect of LoRA rank on the performance of a model. The y-axis represents the difference in AUROC between the rank on the x-axis and rank=8.

tially due to its distinct architecture (*193*). For BioBERT, modest improvement occurred at $r = 16$, but performance decreased at higher ranks. For RoBERTa, enhancements were seen at $r = 32$ and $r = 128$, with no clear pattern. Despite fluctuations, the overall performance impact was minor, with the greatest AUROC increase being 0.0125 and the largest decrease 0.0078. Thus, even for varying parameter counts, the default LoRA rank of 8 was a good trade-off between tuning time and performance, though further tuning LoRA parameters may benefit tasks requiring small performance increases.

### 4.8.4 Effect of domain pre-training

I conducted direct comparisons between models pre-trained in general, biomedical, and clinical domains across the various model architectures. For the sake of brevity, I focus solely on the i2b2-2010 relation extraction task for the main result in Figure 22.

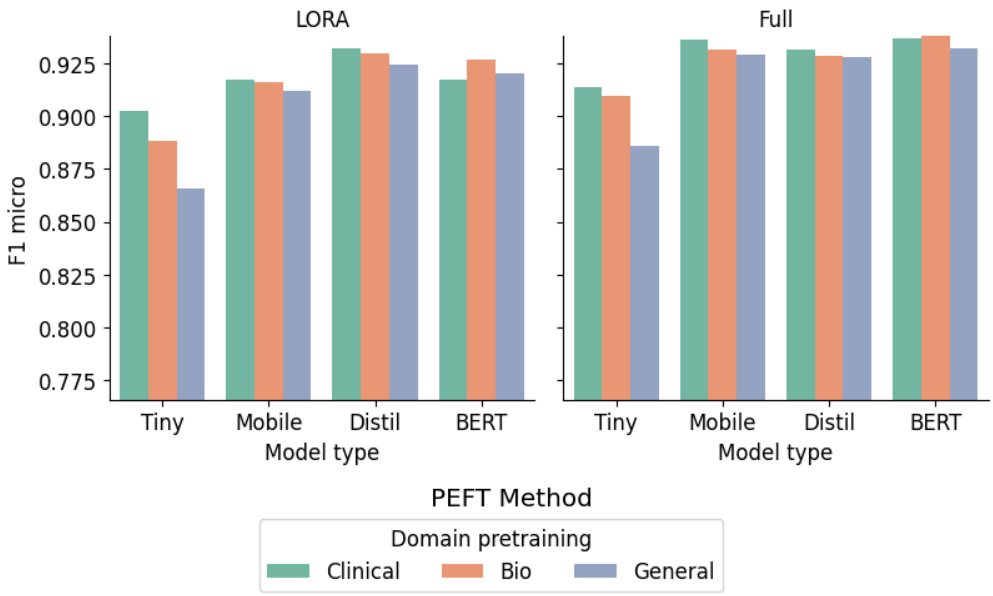


**Figure 22** Comparison of F1 micro scores on the I2B2 2010 relation extraction task dependent on whether the model received biomedical, clinical, or general domain pre-training. Model type refers to the base model (architecture) used: TinyBERT, MobileBERT, DistilBERT or standard BERT as described in Table 22.

The performance differences were greatest in the smaller models (<65 million parameters), with clinically pre-trained models generally performing best with a 1-4 percent improvement based on model size. For results across all tasks and their dependence on domain pre-training, please see Appendix B.3.

### 4.8.5 Budget

The primary advantage of employing PEFT methods lies in their ability to reduce training times, lower GPU memory demands, minimise storage requirements and enhance model re-usability (all of which lower financial burden). I examined the trade-offs among these aspects for the various model architectures, focusing on the most effective PEFT method identified in the initial experiments, LoRA. For

each defined budget, I used the MIMIC mortality prediction as the benchmark task and macro-averaged AUROC as the metric of evaluation. In addition to training the LoRA versions of each model, I also conducted full fine-tuning on each model to determine whether any budget level could achieve efficiency improvements comparable to those provided by PEFT approaches. The only exception was the Llama model, which was exclusively trained with LoRA due to computational constraints whereby fully fine-tuning a 7 billion parameter model would require over 100 $GB$ of VRAM.

#### 4.8.5.1 Time

A key measure of efficiency is the training time and the speed at which different models converge within a constrained period, particularly a relatively short one. An initial time limit of $2,000$ seconds (33 minutes) was used for all models. To evaluate the performance of the models that seemed to show an increasing trend in performance after the budget of $2,000$ seconds (Fig. 23), the time budget was increased to $6,000$ seconds (100 minutes). An exception was made for the Llama model, which remained under-trained even after $6,000$ seconds, necessitating an extension of the training period to approximately $21,500$ seconds (6 hours) to attain optimal performance.

Fully fine-tuned versions of the models, regardless of size, were quicker to converge than the LoRA versions, followed by eventually over-fitting. The LoRA versions of the models eventually converged to the performance (or close to the performance) of the fully fine-tuned models. This observation suggests that fully fine-tuning a model on a small time budget could theoretically obtain an efficiency gain similar

to the PEFT methods. However, from a practical standpoint, the LoRA version of all models converged to similar performance within ∼1 hour of training while being substantially more memory efficient. A caveat to this analysis is that the learning rates for LoRA and full fine-tuning were different due to drastic fluctuations in performance, whereby one approach would under- or over-fit massively. A more detailed analysis of the difference in cost efficiency between the methods is discussed in Section 4.8.5.4.

#### 4.8.5.2 Few-shot training

I explored few-shot training with sample budgets that ranged from 8 to 4096 samples, increasing incrementally by a factor of 2. As expected, there is a direct relationship between sample budget and model performance, regardless of the model type and training method used. While the fully fine-tuned models generally performed better than their LoRA counterparts for smaller sample budgets, the difference became negligible for higher budget values (Figure 23). The fully fine-tuned models on a budget of 4096 samples underperformed when compared against the LoRA versions on all samples. Hence, for the sample budget to be considered as an effective method for efficiency gain, there would need to be more than 4096 samples.

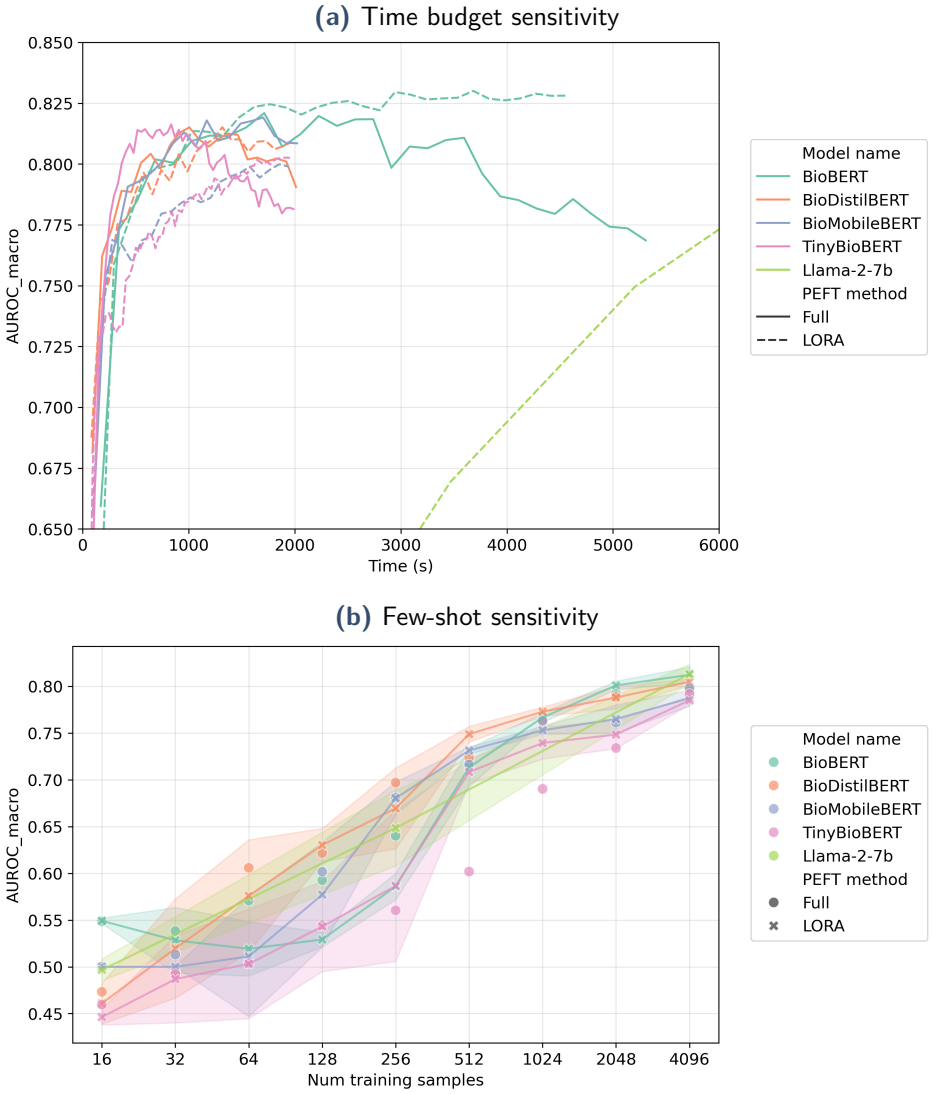


**Figure 23** Effect of training time (**a**) and few-shot sampling (**b**) on models of varying sizes, trained using full fine-tuning as well as LoRA. The connected points reflect the LoRA results to highlight the trend. The task used for this experiment was the MIMIC-III mortality prediction task and the highlighted regions show the standard deviation across 3 runs with different random seeds. The colour of the pointers reflects the model, and the shape of the marker reflects the fine-tuning method used (full or LoRa).

#### 4.8.5.3 Holistic efficiency

In an attempt to establish a unified metric of estimated efficiency (similar to previous works combining multiple training metrics (*8*)), I computed the average of the following normalised metrics: time taken to reach peak performance $T$, number of trainable parameters $P$ and total model parameters $S$[1]:

$$\text{Efficiency estimate} = \frac{T + P + S}{3} \tag{4.3}$$

For ease of interpretability, I scaled the final efficiency value to a range between 0 and 1, where 0 represents the least efficient model and 1 represents the most efficient [2]. The relationship between the estimated efficiency metric and performance is shown in Figure 24.

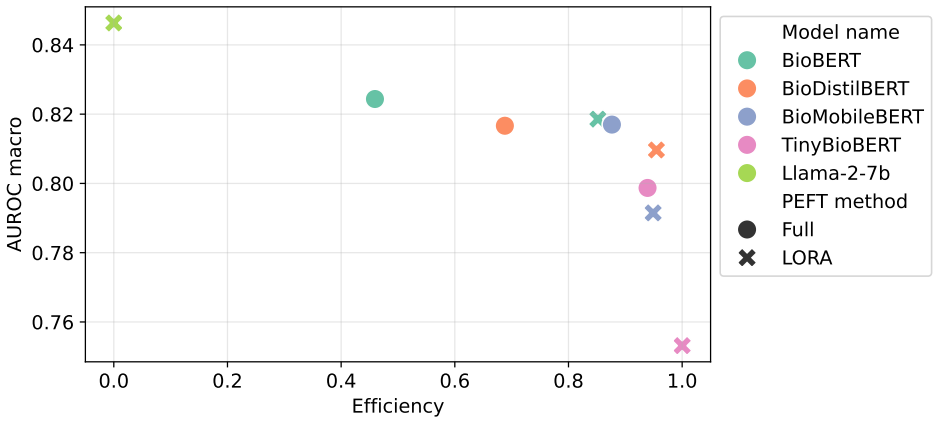


**Figure 24** Comparison of efficiency against performance on the validation set between models of different sizes. The colour of the pointers reflects the model, and the shape of the marker reflects the fine-tuning method used (full or LoRa).

[1]This is not a statistically robust or proven approach, but rather for illustrative purposes

[2]the efficiency estimate used here is crude and may not capture certain statistical properties of individual metrics

The holistic efficiency estimate shows a general negative correlation between efficiency and performance. However, the performance gap is relatively minor compared to the difference in efficiency between models. This highlights the balance of efficiency and performance, with LoRA arguably achieving the best compromise. The model size in this analysis is the largest influence though, and the larger Llama-2 model with LoRA does achieve the best absolute performance.

#### 4.8.5.4 Memory and cost

| Model name | PEFT Method | Train time (hr) | Inference time (hr) | Total cost (GBP) |
|---|---|---|---|---|
| Llama-2-7b | LoRA | 51.07 | 4.06 | 112.22 |
| BioBERT | Full | 2.51 | 0.22 | 5.56 |
| BioBERT | LoRA | 2.16 | 0.22 | 4.84 |
| BioMobileBERT | Full | 1.57 | 0.14 | 3.48 |
| BioMobileBERT | LoRA | 1.35 | 0.14 | 3.03 |
| BioDistilBERT | Full | 1.35 | 0.12 | 2.99 |
| BioDistilBERT | LoRA | 1.21 | 0.13 | 2.73 |
| TinyBioBERT | Full | 0.53 | 0.06 | 1.20 |
| TinyBioBERT | LoRA | 0.46 | 0.06 | 1.06 |

**Table 26** Costs for training each model on a task with approximately 30,000 training samples for 10 epochs, followed by running it in inference mode for 100,000 samples. The costs were estimated using AWS EC2 rates. The instances used for estimating training and inference costs were g5.16xlarge and g4dn.16xlarge, respectively.

The GPU and storage requirements for training differ massively between model types and fine-tuning methods. While performance generally increases with model size, there is a trade-off between performance and required computing, as well as training and inference speed. Table 22 provides model sizes and memory requirements, and Table 26 calculates the estimated training and storage costs for different model sizes. As observed previously, larger models like Llama-2-7b achieve higher performance on most tasks but at 20 and 94 times the monetary cost of models

like BioBERT and TinyBioBERT, respectively. If fine-tuning a model for multiple tasks, BioBERT and similar models can provide a good trade-off between monetary cost and performance.

## 4.9 PEFT discussion

I have explored using different-sized LLMs for various clinical downstream tasks, assessing both traditional fine-tuning and different PEFT methods. Across the sequence classification tasks, the LoRA PEFT method proved superior to other PEFT methods in terms of performance and consistency, leading to its selection as the preferred PEFT method for subsequent analysis. While full fine-tuning generally outperforms LoRA, in certain models and tasks, the performance is at least matched or even surpassed. Furthermore, LoRA worked well for all model sizes and task types, making it a flexible choice. This finding highlights the potential of utilizing PEFT methods with very small LLMs. The relative performance gap between full fine-tuning and LoRA appears to increase with smaller models, which was only partially mitigated by increasing the LoRA rank. Using PEFT methods for the clinical domain holds particular value when resources can be limited, or for rare diseases where labels and data are scarce.

### 4.9.1 Comparison of LLM size

Evaluating model sizes on a specific task within a fixed time frame, including the 7 billion parameter Llama-2, revealed significant learning capability differences. Numerous smaller LLMs completed 5 training epochs before Llama-2 achieved comparable performance. However, given sufficient time, Llama-2 reached the highest evaluation performance. Despite being ∼500 times smaller than Llama-2, smaller LLMs achieved reasonable performance much faster, with Llama-2 taking roughly ten times longer and over six hours to reach peak performance.

### 4.9.2 Holistic efficiency

According to the composite estimated efficiency metric, medium-sized LLMs are substantially more computationally efficient than the largest model for the given task, with only a minor performance drop. Deriving a true holistic efficiency representation is difficult without considering pre-training cost and other unknown facets, but this provides a reasonable overview of the model size and fine-tuning method interplay. Further profiling would quantify exact runtime improvements.

### 4.9.3 Domain pre-training

Pre-training LLMs proved important for performance on various clinical domain tasks, with biomedical and clinical LLMs generally outperforming their general counterparts. Clinical LLMs like ClinicalBioBERT were trained on MIMIC-III notes, giving them an unfair advantage. However, the potential for data leakage in

Llama-2 is difficult to ascertain. Developing specialized clinical LLMs through pre-training on relevant clinical language remains optimal for subsequent downstream task adaptation, in line with my previous chapter.

### 4.9.4 Limitations and future work

The PEFT methods investigated reflected the field's state at the time, but the research area has evolved since then. Indeed, since these experiments were conducted, the PEFT library (*109*) has introduced several new methods worth exploring. When comparing model sizes, training was limited to a single GPU to align with the thesis and overarching budget aim. This may disadvantage the larger models (Llama-2) where I employed weight quantisation to allow any training. This constraint also hindered the exploration of Llama-2 across all tasks and conducting hyperparameter optimization. Future work could explore this further, although the resources required are extensive and may yield diminished returns.

### 4.9.5 PEFT conclusion

Overall, this work highlights the power of PEFT methods for small LLMs and demonstrates how domain pre-training can create efficient clinical models. While larger LLMs' capabilities are evident, they come with significantly higher time and financial demands.

## 4.10 Overall conclusion

Multiple methods were presented to efficiently adapt LLMs of different sizes to various downstream clinical NLP tasks, successfully reducing computational and storage requirements for good performance. Both prompt learning and other PEFT methods were able to reduce the trainable parameters needed by approximately 99 %. The subsequent storage requirements and compute resources required can drastically reduce the financial burden of training and using LLMs in specialist domains such as healthcare. Effectively a single LLM can be re-used for any number of downstream tasks and adapted using small adapters produced via prompt learning or PEFT methods, instead of re-training the whole LLM for each task. While prompt learning was initially powerful, it suffered from a more complicated prompt and verbalizer ecosystem than the superior PEFT method, LoRA, which is easy to integrate into any LLM-based architecture. Prompt learning was more sensitive to hyperparameters, and it was unclear why one prompt setup worked better than others. Moreover, prompt learning in its presented format is difficult to align with NER tasks, which are prevalent in healthcare NLP research. PEFT is part of a booming ecosystem with a fully maintained code library intended for transformer-based language models and HuggingFace (*35*, *109*), making open-source sharing substantially easier.

This chapter also carried through the same importance of domain pre-training as seen in Chapter 3, whereby smaller LLMs benefit from initial pre-training on clinical texts when adapting them to downstream tasks. Therefore, combining efficient pre-training with efficient downstream adaptation methods provides the

ultimate value. Combining the two paradigms can lower the barrier to training bespoke LLM models in resource-limited settings and TREs.

The applicability of prompt learning and different PEFT methods to smaller LLMs is very promising, providing a large increase in efficiency across multiple metrics. However, the modern Llama-2-7b (*3*) LLM yielded boosts to absolute task performance that cannot be ignored and gives reason to consider their use for future healthcare research. The major caveat of large LLMs is the substantially larger associated time and financial burden, rendering them difficult to apply wholesale to many tasks while remaining budget-friendly. However, there is active research into the scaling down of LLMs, and quantisation methods and hardware improvements may begin to reduce the financial and computational burden of recent LLMs.

The value of this work to the healthcare research community is in the much-improved efficiency and feasibility of adapting increasingly large LLMs to specific datasets and tasks. Now, with a reasonable GPU and ample storage, a single LLM can be stored and adapters such as LoRA trained for any number of downstream tasks. Future work could seek to improve the ease of interchanging different LoRA weights dynamically and create a system capable of handling multiple domains and targets. Given the massive variety in the disease area, the department needs of different NHS trusts, and use cases for LLMs, LoRA is very exciting.

Based on the experiments presented throughout Chapter 3 and Chapter 4, I will seek to utilise and combine the NHS OHFT LLMs with LoRA in Chapter 5.

Chapter 5

# Psychiatry referral triage assistance

## 5.1 Disclaimer

This chapter and its contents are based on one paper available as a preprint and currently under review. I am the first author, with author contributions to the paper provided after the publication details below:

- Niall Taylor, Andrey Kormilitzin, Isabelle Lorge, Alejo Nevado-Holgado, and Dan W. Joyce. "Bespoke Large Language Models for Digital Triage Assistance in Mental Health Care." arXiv preprint arXiv:2403.19790 (2024). - under review at AI in Medicine, *LLMs in healthcare special issue*. N.T, D.W.J, A.K, and A.N.J conceptualised this work. N.T and D.W.J curated the datasets. N.T developed pre-processing and experiment running and analysis code. N.T and D.W.J drafted the manuscript. A.K, and A.N.H, I.L, and A.C revised and edited the manuscript.

## 5.2 Introduction

In this chapter, I combined components of the preceding chapters into a full pipeline prototype applied to digital triage support in secondary mental health.

Referrals to secondary mental healthcare are usually made via a written document describing the patient's symptoms, difficulties, and relevant risks (e.g., self-harm, suicide, risk to others). In most secondary care services, Community Mental Health Teams (CMHTs) provide a single point of access and triage, often organized geographically. CMHTs treat the whole spectrum of mental illness, but some areas have sub-specialist teams for specific conditions like eating disorders or first-episode psychosis. Depending on the referring professional (e.g., general practitioner, emergency department, social care), a patient may be referred directly to a subspecialty team if the referrer suspects a specific condition. This can lead to "referral bouncing" between teams when they disagree on the appropriate team (*197*). Upon receiving a referral, secondary care teams summarize or enter it into the patient's EHR, along with any historical data from previous treatment episodes. Based on the referral and historical EHR data, the triage team decides to either: accept the patient for assessment, reject the referral due to insufficient information or lack of need for secondary care, or route the referral to a more appropriate subspecialty team. This process is outlined in panel A of Figure 25.

The NHS maintains monthly digital audits of activity in mental health services (*198*) that show in each month of 2023, there were between 370,000 and 470,000 individual new referrals to these mental healthcare services. This triage process is time-consuming, requires clinical expertise, and occurs before seeing the patient.

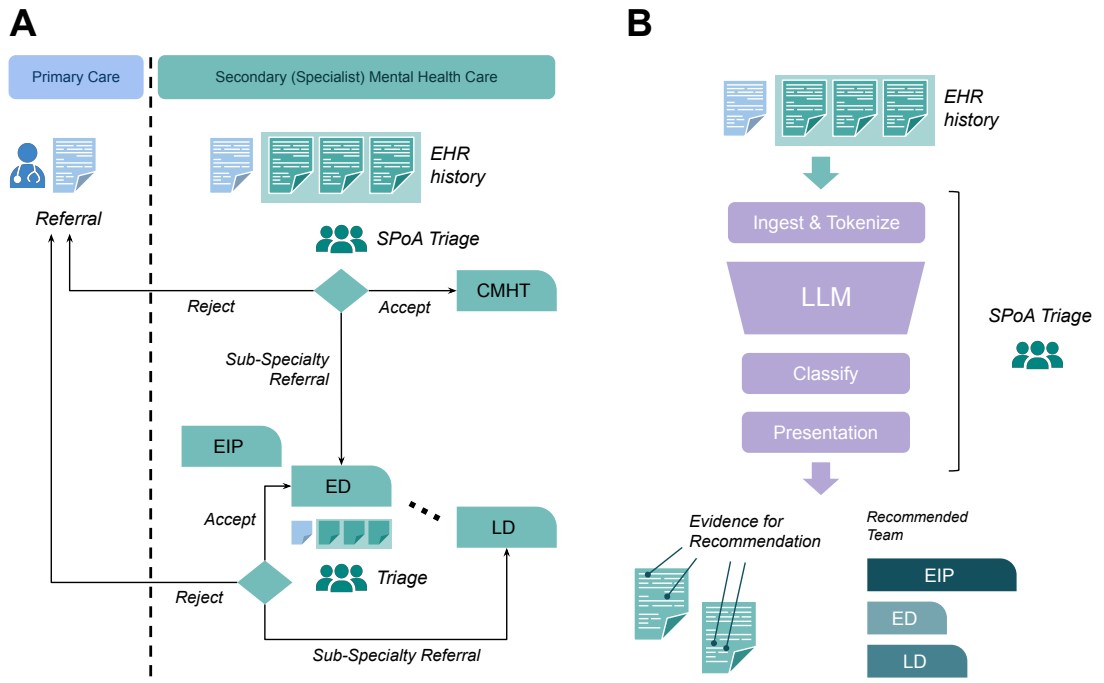


**Figure 25** Overview of the referral triage pipeline. **A**: Schematic description of the triage process using referral and historical EHR documents highlighting the accept/reject and subspecialty referral and re-triage process **B**: End-to-end ingestion of the same clinical data for assisted triage. Example subspecialty teams: EIP = early intervention for psychosis, ED = eating disorders, ID = intellectual disability

It has been criticized for lack of transparency, inconsistent criteria use, and referral bouncing leading to hidden waiting lists where patients deteriorate awaiting triage outcomes without receiving care (*199*).

In this chapter, I consider assisting clinicians make decisions about triaging patients to an appropriate specialist mental healthcare (MH) team by ingesting and representing the patient's EHR data through an LLM-based model, depicted in Figure 25. Mental healthcare makes extensive use of narrative recording of clinical data as unstructured free-text, making LLMs a particularly useful technology for assisting clinicians in parsing and making use of volumes of textual information. In turn, this assistance could improve the triage process by enhancing the extraction

efficiency and making relevant clinical data visible. I must also state clearly that this work serves as a prototype for developing a secondary mental health triage tool, and provides a foundation to build. This work does not attempt to model psychiatric outcomes based on triage recommendations and only focuses on a simple initial proof of concept for ingesting triage-related clinical documents to aid clinicians' processing. This is very early-stage work and would not be fit for use in a live clinical setting at this stage.

## 5.3 Motivation and related work

As covered in Chapter 3, specific clinical domains (e.g. neurology, psychiatry, respiratory medicine) are particularly difficult for general-purpose foundation LLMs. I hoped to have resolved this in part with the creation of the OHFT NHS-adapted LLMs from chapter 3. EHR data has been shown to contain a large portion of redundant information (*1*), and traditionally clinical NLP has heavily relied on information extraction, heuristics, and NER to locate salient information. These have proven powerful for numerous applications, such as NER for medications (*65*) and medical concept annotation (*64*) for delivering structured data from the original raw text.

In contrast, triaging clinicians can extract relevant signals from sequences of narrative clinical notes contained in the EHR. It is reasonable to postulate that supervised representation (feature) learning through LLMs might capture similar signal information that can assist triage tasks. Importantly, I investigated the ability to represent raw, narrative clinical notes using LLMs in an end-to-end fashion.

LLMs can ingest token sequences to deliver feature representations (embeddings) to serve a downstream classification task directly, as opposed to information extraction. Here, the embeddings of patients' referral information and histories (from their EHR) are used to determine (classify) which triage team they best align with.

To learn representations for assisting triage, an LLM must gracefully handle variable-length sequences of tokenized input, directly extracted from a patient's EHR. In the EHR, clinical information is recorded in a time-stamped sequence of documents (notes) containing narrative and largely unstructured free text. Each document or note can thus be of variable word length. For example, a short administrative note (acknowledging the receipt of a referral or recording a brief telephone contact with a patient) may be tens of words, whereas a clinical assessment can be thousands (illustrative descriptive statistics for the dataset are shown in Appendix C.1.2).

In addition, any given patient may have had one or more historical referrals or episodes of care with one or more different teams (see top panel, Fig. 28). These episodes will be accompanied by further EHR notes that describe the episode of care from referral to the point of discharge from services.

### 5.3.1 Desiderata for LLM assisted triage

Following the above, I had the following design prerequisites for the LLM and triage recommendation pipeline:

1. **End-to-end ingestion** of unstructured clinical EHR text to assist in triage, capable of gracefully handling variable-length inputs (e.g. at the document, referral and instance level) while maintaining classification performance

2. **Resource efficient** in GPU compute and memory requirements such that re-purposing of foundation LLMs is feasible for the specific clinical use-case

3. **Ability to interrogate models** to present users of assisted triage with evidence from the source EHR data that drives a triage recommendation – in accordance with guidance on people's right to explanation for a decision-making use of AI assistance (*200*)

4. **Facility to train at-scale** without the need for human expert annotation of e.g. entities, concepts or text thought to be relevant to a triage decision

Several studies have utilised structured EHR data to representation-learn patient embeddings (*78*, *81*, *201*) about acute or general hospital units. However, as discussed in section 2.4, there is a relative lack of research focussed on triage specifically. One notable study utilized a machine learning "cascading classifier" approach with individual models per triage category for initial psychiatric assessments. They used manually extracted features from Initial Psychiatry Evaluation (IPE) documents to predict triage severity (absent, mild, moderate, severe) (*119*). That study did not, however, use or ingest unstructured free text clinical notes and instead relied on available structured fields. Studies have utilised the MIMIC-III dataset to develop long sequence transformer-based approaches for free text discharge summaries to predict different clinical outcomes and ICD-9 diagnosis codes (*57*, *73*, *74*). As discussed in the earlier literature review, section 2.4, there are also some studies utilising mental health EHRs for medical information extraction with MedCAT, MedGPT (*114*), and Foresight (*79*). Recently, the South London and Maudsley (SLaM) NHS research group fine-tuned BERT to identify mentions

of pain within free text, achieving an F1-score of 0.97, outperforming simpler machine learning models substantially (*120*). There is a shift towards utilising recent LLMs for mental health research, but it is relatively lacking.

This chapter will serve as a framework for using LLMs for clinical decision support within secondary mental health, whilst emphasising compute efficiency and transparency of model outputs.

## 5.4 Methods

### 5.4.1 Dataset and triage recommendation

For this chapter, I used the OHFT EHR dataset outlined earlier in Section 2.5.1.3. As a reminder, this is a regional UK-based provider of specialist secondary mental healthcare to Oxfordshire and Buckinghamshire's population of approximately 1.2 million people; of which a subset of 200,000 patients was used spanning over a decade.

Within the EHR data are clinical notes alongside structured information like referral and discharge dates, with the triage team initially receiving the referral. Using structured fields, I developed a heuristic rule with OHFT clinicians to record if a patient was accepted or rejected. The heuristic for an accepted referral used a 14-day window after the referral date. If the discharge date occurred within this window, it was deemed a rejection by that receiving team. If the referral had not been closed and remained open after the 14-day window, it was deemed accepted. Each referral is treated independently, so a patient may be referred to another

| Dataset | # labels | # train samples | # eval samples |
|---|---|---|---|
| Accepted Triage Team Brute. | 5 | 65,000 | 157, 880 |
| Accepted Triage Team Concat. | 5 | 17,629 | 4,272 |

**Table 27** Dataset details. With the brute force approach, every single document is processed separately, hence the significantly higher numbers of individual samples. Both relate to the same number of referral instances. The training sample for the brute force approach was randomly subsampled to 13,000 samples per class to avoid excessive training times.

team (i.e., rejected from one, referred to another) and this is seen as two separate instances. If the referral date was within 14 days of the data extraction, they were removed due to the inability to determine team acceptance. Applying this heuristic produced training and evaluation samples over 5 sub-specialty teams (detailed below), with sample numbers in Table 27.

Using referral and discharge dates, I segmented patients' historical EHR data into collections of narrative clinical notes surrounding a referral (see Fig. 28, top panel). Such a collection over an episode – consisting of a referral and documents describing care up to the referral date – is an instance. A patient may have multiple unique instances in training or validation samples, but not both (avoiding data leakage). Crucially, an instance is a time-ordered sequence of variable-length documents (notes) before, or on, the referral date. This ensures the model is only trained on data before any clinician's referral decisions are documented. Effectively, the model will learn to use historical information to predict and recommend the triage team. For patients routed between teams, they are represented as separate instances.

With this data structure, a classification task for identifying the accepted triage team for any given instance was designed based on the available clinical notes alone. For training, I subset only the referral instances that resulted in an "accept"

decision. This downstream classification task assists representation learning of instance embeddings (in the LLM) that represent referral acceptance.

**Sub-specialty teams** The current single point of access (SPoA) triage process (**A** in Figure 25) presents technical and ethical difficulties. Technically, the OHFT community mental health teams (CMHTs) can act as a SPoA for triaging any referred patient. But they can also be a secondary care team, accepting patients not suitable for sub-specialist teams. Additionally, referrals were often directed to sub-specialty teams, bypassing the SPoA model, leading to possible confusion where 'general' CMHTs appeared to arbitrarily accept known sub-specialty patients (as shown in Figure 26). Essentially, referral bouncing can be a legitimate clinical decision or an opaque practice of teams declining patients due to disagreement over the most suitable team (detailed further in appendix C.1).

The ethical concern is that if a model was trained to act as a SPoA triage assistant, then I would need to additionally include a classification for "rejection" from any/all teams. This risks the model maladaptively learning to 'default to rejection', rather than learning the signal that drives being accepted by a team. Consequently, this risks recapitulating existing critiques of the triage system in mental health (*197*, *202*) and defeats the desirable property of being able to interrogate the triage system to understand why a particular patient has been recommended for acceptance by a specific team.

To this end, I present results on triaging assistance for well-defined sub-specialty teams present in adult mental health services across the United Kingdom, namely:

1. Eating Disorders - (**ED**),

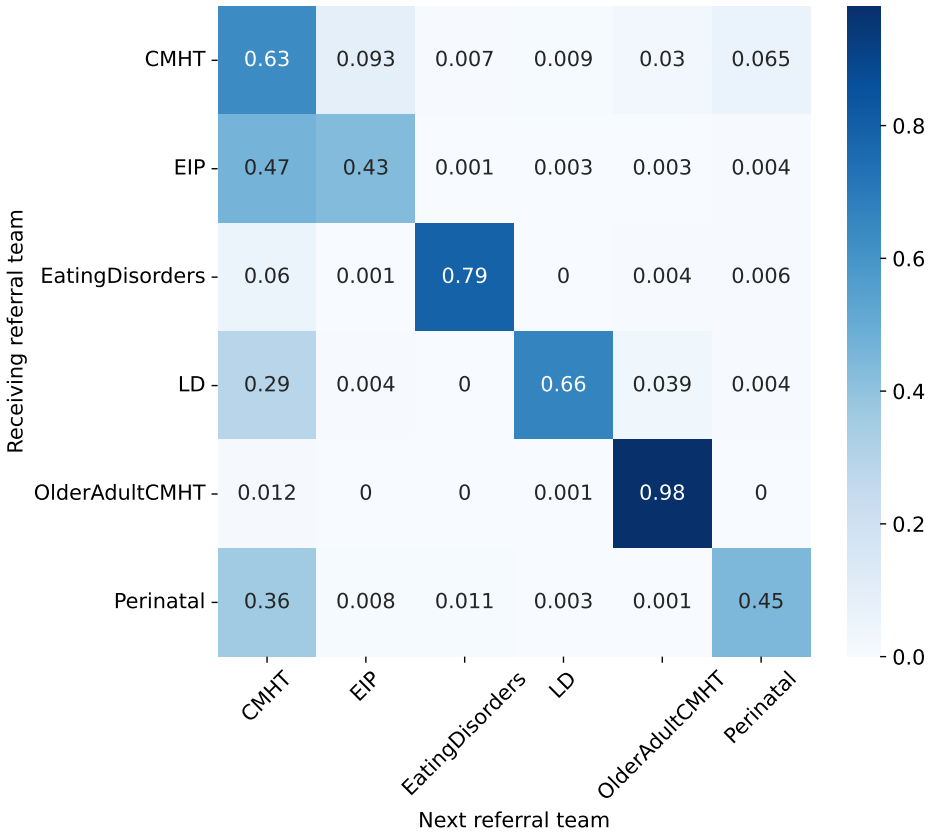


**Figure 26** Tabulation of the probability of the team first receiving a referral (Team A, rows) referring the patient to another team (team B, columns) within a 30-day window.

2. Mental health for people with intellectual disability - (**LD**)

3. Older-adults (including memory and dementia services) (**OACMHT**)

4. Early intervention for psychosis (**EIP**)

5. Peri-natal psychiatry (**PD**).

Teams not included were the crisis resolution and home-treatment teams (because the referral processes and criteria are very different, representing the urgency and acuity of these patients), teams that do not provide community-based assessment

and treatment (for example, psychological medicine or general hospital liaison psychiatry), and specialist research-led clinics (that will not be available in a majority of NHS trusts). The number of referrals per each of the subteams was highly imbalanced; for example, teams serving older adults (usually, those over the age of 65) were over-represented and this reflects the organisation of clinical services from which the data was extracted. Given the stated intention to develop an end-to-end triage assistance tool, I decided not to artificially balance the data set (for either training or testing). Figure 27 shows the distribution of referral instances accepted by each triage team.

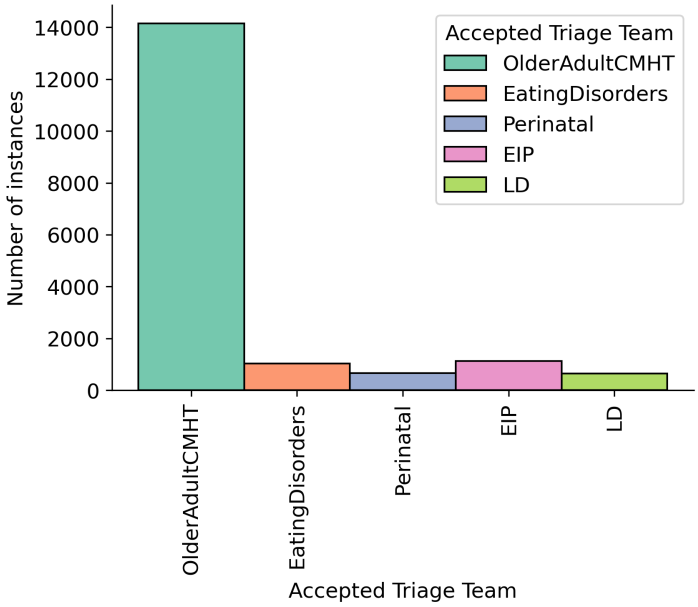


**Figure 27** Distribution of the number of individual referrals accepted to each triage subspeciality.

### 5.4.2 LLM choice

For a reminder of how LLMs process sequences of text, i.e. documents, please refer back to section 2.5.2.1 and the earlier chapters. The choice of LLM for this chapter was important and I opted to use open-source baseline models and the models produced from chapter 3 as follows:

1. RoBERTa-base (*134*) as a general domain LLM for baseline results
2. RoBERTa-base-OHFT, which was initialised from RoBERTa-base and further trained via MLM on a sub-sample of the OHFT clinical notes
3. Clinical Longformer (*73*, *203*), which is an LLM model with a sparse attention mechanism trained on clinical text, allowing a maximum token sequence length of 4096 (see appendix C.1.3 for details)

Note, that I have selected the OHFT NHS LLM domain adapted using MLM only, as opposed to the DeCLUTR or note category pre-trained LLMs. This was in part due to the slight improvement in fully fine-tuned classification performance for the MLM only model. Initial results for the triage task also showed the MLM only model performed best and thus I opted to only focus on this model. Another reason is the relative simplicity of MLM only when compared with the contrastive loss models, especially when considering the ability to replicate the model training for other NHS trusts.

To adapt these LLMs to a sequence classification task, I followed the same steps used in the previous chapter with a task-specific classification head: $f_{\text{head}}(\cdot)$ which

takes the sequence embedding output by the LLM, $\mathbf{E}$, as input and generates a $n_t$-dimensional vector, where $n_t$ is the number of triage teams (refer back to section 2.5.5). The exact algorithm for deriving the LLMs sequence output is dependent on the approach used, which is outlined below.

### 5.4.3 Representing referral instances

The number of documents and total volume of words for a given referral instance varied considerably in the OHFT dataset, ranging between approximately 300 and 50,000 (in extreme cases). Table 28 shows the summary statistics for the distribution of token numbers per individual document and when concatenated together to form an entire instance. The clinical information or 'signal' contained in each note will be highly variable, with a mixture of note categories produced by different members of the clinical team with widely varying purposes.

To mimic the clinician's perspective of viewing these historical clinical notes, notes are fed into the model in reverse chronological order, whereby most recent notes are at the beginning. The reasons are two-fold: clinicians will typically be presented the most recent documents in EHR user interfaces, and the models typically operate in a bidirectional manner and in certain scenarios will truncate very long sequences (i.e. the oldest notes) due to limitations of the architectures themselves.

| Type | Mean | Percentiles (25:50:75:90) |
|---|---|---|
| Per document tokens | 183 | 62 : 120 : 217 : 388 |
| Concatenated instance tokens | 6420 | 429 : 1323 : 3658 : 11427 |

**Table 28** Descriptive statistics about the number of tokens across clinical notes related to individual referral instances.


**Figure 28** Overview of the three different methods for using LLMs to embed clinical notes and classify the related triage team. **A** shows the evidence accumulation method, which combines classification decisions for individual documents using the most frequent class prediction or statistical mode (modal decision). **B** utilises the longformer to process as much of the concatenated clinical notes as a singular historical sequence embedding, with the red cross indicating very long sequences are still truncated ($>$ 4096 word tokens). **C** decomposes the long concatenated sequences into smaller chunks that are separately encoded and then combined via a label-aware attention mechanism, maintaining the original token sequence order and providing explicit weights between tokens and classification outputs.

The number of tokens per document and per entire instance is an important factor when utilising LLMs, which have limits on the sequence length (as discussed in section 2.5.3.5). To cope with variable-length EHR histories, I compare three different approaches to handling variable token sequence lengths (Figure 28). Denote the $n$th instance as the time-stamp ordered set of $n_d$ documents: $I_n^\star = (d_1, d_2, \cdots, d_{n_d})$, and the output triage recommendations as a multi-class probability vector $\Pr(\mathbf{y}|\bullet) = (y_1, y_2, \cdots, y_{n_t})$ where $n_t$ is the number of triage teams, and $y_1, y_2, \cdots, y_{n_t}$ the probabilities assigned to each team, $T$.

I compare three different approaches to ingest and process any instance, outlined in Fig. 28, with full details of their implementation below:

### 5.4.4 Brute force approach: A

A document-level **brute force** approach (**A** in Fig. 28): Each document $d_i \in I_n^\star$ is tokenised, truncated to a length of 512 tokens (the maximum for RoBERTa-based models) to give $d_i^\star$, which is passed to the LLM in order. For each document, the LLM with the classification head (as described earlier in Figure 6) $f_{\text{head}}$ delivers $\Pr(\mathbf{y}|d_i^\star)$ and the recommended triage team $j \in T$ for that document is a vote $v_i = \arg\max_{j \in T} \Pr(y_j|d_i^\star)$. After ingesting an entire instance $I_n^\star$, there is an ordered set of votes for each document, $V_{I_n^\star} = (v_1, v_2, \ldots, v_c)$ and the recommended team is the most frequently voted team (i.e. mode) over $V_{I_n^\star}$.

### 5.4.5 Truncated concatenated sequence approach: B

An instance-level **single concatenated sequence** approach (**B** in Fig. 28): The documents $d_1, d_2, \ldots d_c$ in the instance are concatenated (in order) and the resulting instance is tokenised before being truncated at the respective model's maximum token length. The resulting truncated instance $I_n^\star$ is fed through the LLM and its output is passed to the classification head $f_{\text{head}}$ and the recommended team is maximum probability value: $\arg\max_{j \in T} \Pr(y_j | I_n^\star)$. Here I compare the performance of RoBERTa-base and RoBERTa-OHFT with a maximum of 512 tokens, with the Clinical Longformer model with a maximum of 4096 tokens as described in Table 29.

### 5.4.6 Segment-and-batch: C

The **segment-and-batch** approach (**C** in Fig. 28) uses a form of sequence chunking and aggregation, paired with a label-aware attention mechanism based on previous work (*74*, *204*, *205*). This approach is unique because it handles variable sequence length documents by decomposing the sequence into ordered chunks shorter than the model's maximum sequence length. Given a concatenated instance $I_n^\star = [w_1, w_2 ... w_j, ... w_{n_w}]$, where $n_w$ is the total number of tokens, it is split into $n_s$ consecutive, non-overlapping segments of a fixed size $l$ number of tokens (I trialled segment lengths of 128, 256 and 512 tokens):

$$s_k = \{w_j \mid l \cdot k \leq j < l \cdot (k+1)\} \tag{5.1}$$

Individual segments are fed into the LLM separately to obtain hidden numerical representations for each token inside each segment, $\mathbf{S_k} = LLM(s_k)$, where $\mathbf{S} \in \mathbb{R}^{n_s \times l \times d_h}$, followed by the column-wise concatenation to produce a tensor of all token representations:

$$\mathbf{H} = [\mathbf{S_k}, ..., \mathbf{S_l}] \tag{5.2}$$

where $\mathbf{H} \in \mathbb{R}^{n_w \times d_h}$ preserves the original sequence order. Following this, a label-wise attention (LA) (*74*, *204*) mechanism converts $\mathbf{H}$ into label-specific representations $V \in \mathbb{R}^{d \times n_t}$, with $n_t$ as the number of triage teams. Here, $A = softmax(UZ)$, and $V = HA^{n_t}$ where $U \in \mathbb{R}^{n_t \times d_p}$ is a trainable embedding matrix, and $Z = \tanh(PH)$ where $P \in \mathbb{R}^{d_p \times d_h}$ is again a trainable matrix.

This approach provides direct attention between each token's hidden representation and the triage team classification decision. Similar to previous works, I utilised these label-aware attentions to aid the interpretability of classification decisions (*206*), see section 5.5.2.

**Integration of LoRA** As alluded to in the introduction to this chapter, I sought to bring certain components of Chapter 4 - namely LoRA - into this work in an attempt to improve the efficiency of the methods outlined above. Based on initial findings from each of the instance processing approaches outlined above, I combined LoRA (outlined in detail in section 4.7.4) with the segment-batch approach (the best-performing approach). The objective was to improve the efficiency in terms of trainable parameters and model size, whilst retaining performance.

The three main approaches offer different levels of granularity and efficiency, which

| Approach | Base Model | # Params | Max length | Infer. speed (SD) |
|---|---|---|---|---|
| Brute Force [**A**] | Roberta-base/OHFT | 125 mil. | 512 | 683 (0.41) |
| Concat truncated [**B**] | Roberta-base/OHFT | 125 mil. | 512 | 14 (0.01) |
| Concat Longformer [**B**] | Clinical-Longformer | 148 mil. | 4096 | 212 (0.2) |
| Segment-batch/LoRA [**C**] | Roberta-base | 125 mil./0.8 mil. | max* | 97 (0.69) |

**Table 29** Overview of the different sequence representation approaches and model setups with the number of trainable parameters, maximum sequence length and inference speed. The square brackets next to each approach refer to the methods outlined above in Fig. 28. I present the inference speed averaged across 500 instances from the evaluation data reported in seconds. The SD (standard deviation) represents variance over three repetitions. *The max length for the segment-batch approach is hardware-dependent.

have been summarised in Table 29.

As can be seen, each approach has a differing ability to handle long sequences, with clear differences in inference time. Approach **B** using RoBERTa-base is the fastest but is very limited in the number of tokens it can handle. The segment-batch approach is the second fastest but can handle the longest sequence.

The potential loss of information for each approach also differs massively. I calculated a crude metric of loss as the number of tokens truncated in the training data by each approach. Approach **A** retains the most, only losing on average 1.79% of tokens across an entire instance. Approach **B** with RoBERTa and Longformer lose 50.47% and 12.67% on average. Finally, approach **C** loses 5.18% on average.

### 5.4.7 Implementation details

**Hyperparameters** The different approaches (Table 29), Brute force, Concat truncated, Concat Longformer and Segment-batch, each have varying compute requirements. The hyperparameters used for each approach and model are presented below in Table 30 and were intended to best utilise the GPU, with the batch size

using the maximum possible for each. I chose sensible defaults based on previous chapters.

| Parameter | Brute force | Concat Trunc. | Concat Long. | Segment-batch |
|---|---|---|---|---|
| Batch size | 8 | 8 | 1 | 1 |
| Gradient accumulation steps | 2 | 2 | 16 | 16 |
| Embedding dimension | 768 | 768 | 768 | 768 |
| Learning rate | $1 \times 10^{-5}$ | $1 \times 10^{-5}$ | $1 \times 10^{-5}$ | $1 \times 10^{-4}$ |
| Optimiser | AdamW | AdamW | AdamW | AdamW |
| Chunk size | - | - | - | [128, 256, 512] |

**Table 30** Overview of hyperparameters used in experiments for each instance modelling approach. All training regimes utilised a linear scheduler with warm-up and early stopping with F1 score as the criteria and a patience threshold of 3

## 5.5 Results

Table 31 compares the three sequence representation approaches (A, B, and C in Fig. 28) on the triage team classification task. I also compare performance when utilising either the RoBERTa-OHFT model domain adapted to the OHFT data or general models RoBERTa-base and Clinical Longformer, without pre-training on the OHFT data. As expected, the RoBERTa-OHFT model generally provides a reasonable performance benefit irrespective of the sequence representation method used. This highlights the added benefit of domain pre-training received by the OHFT model when compared to the other models.

The segment-batch approach (**C**) is consistently the best method in absolute performance whilst remaining comparatively efficient. Using LoRA with the segment-batch approach to improve training efficiency (i.e. LoRA requires updating $< 1\%$ of base LLM's model parameters compared to fully fine-tuning the base LLM) incurs only a small degradation in F1 performance of 0.014. The difference in

| Model | Approach | Accuracy | F1 | Precision | Recall |
|---|---|---|---|---|---|
| RoBERTa-OHFT | Brute force [**A**] | 0.935 | 0.846 | 0.828 | 0.882 |
| RoBERTa-OHFT | Concat trunc. [**B**] | 0.974 | 0.922 | 0.927 | 0.917 |
| Clinical-Longformer | Concat Longformer [**B**] | 0.975 | 0.924 | 0.932 | 0.918 |
| RoBERTa-OHFT | **Segment** [**C**] | **0.981** | **0.938** | **0.942** | **0.933** |
| RoBERTa-OHFT | **Segment-LoRA** [**C**] | **0.968** | **0.924** | **0.927** | **0.919** |

**(a)** RoBERTa-OHFT

| Model | Approach | Accuracy | F1 | Precision | Recall |
|---|---|---|---|---|---|
| RoBERTa-base | Brute force [**A**] | 0.923 | 0.80 | 0.78 | 0.859 |
| RoBERTa-base | Concat trunc. [**B**] | 0.889 | 0.772 | 0.71 | 0.866 |
| RoBERTa-base | **Segment** [**C**] | **0.976** | **0.922** | **0.934** | **0.911** |

**(b)** RoBERTa-base

**Table 31** Accepted triage team classification metrics for each sequence representation approach. The square brackets next to the approach refer to the methods outlined in Fig 28.

performance between the Clinical-Longformer and the RoBERTa-OHFT model is relatively small, considering the much larger gap between the RoBERTa-base models. This potentially highlights the added benefit of the increased context window available for the longformer model. The RoBERTa-base model using the segment-batch approach did perform considerably better than when using the other methods, further suggesting the increased amount of information ingested drives performance to a degree.

### 5.5.1 Effect of sequence length on performance

To investigate the comparative performance of each approach across varying number of tokens per referral instance, I stratified the validation data into short (< 128 tokens), medium (> 128 and <= 512 tokens), long (> 512 and <= 4096 tokens), and extra long (> 4096 tokens) sequences.

Figure 29 reveals that for all three methods, the longer the instance (in tokens), the

better the classifier's F1 performance. Of note, the segment-and-batch approach is consistently as good, or better, than the other methods over all instance sequence lengths.

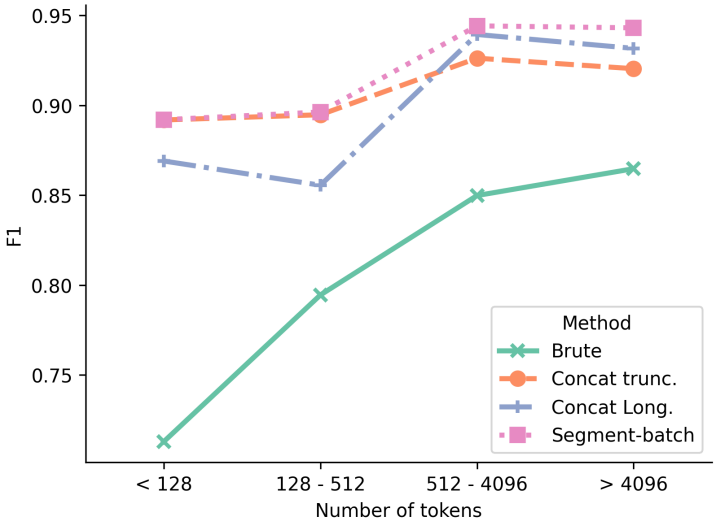


**Figure 29** Validation set F1 score for each method of handling variable sequence lengths as a function of instance token lengths: short ($< 128$ tokens), medium ($> 128$ and $<= 512$ tokens), long ($> 512$ and $<= 4096$ tokens), and extra long ($> 4096$ tokens). For details of each method, refer back to Fig 28 where *Brute* refers to method **A**, *Concat trunc.* and *Longformer* refer to **B** and *Segment* to method **C**.

### 5.5.2 Towards interpretable triage recommendations

A primary objective of this chapter was to deliver a model that provided interpretable outputs. It has been previously argued that it may not be possible to provide mechanistic or intrinsic interpretability for contemporary AI systems that make use of black box methods. A simple LLM with a downstream classification head is an example of such a use-case - the input is transformed into an output with little insight into the inner workings (*207*). Thus, authors have instead proposed that interpretability through presentation may provide a way to offer clinicians the

facility to interrogate decisions using such systems. In essence, this approach tries to capture how the overall system operates (from input to output) by exposing key stages or steps in the computational process as graphical intuitions. First, an instance is ingested and mapped to a location in the 768-dimensional embedding space. Second, a mapping is learned from this embedding space to the probability of being accepted by one of 5 subspecialty teams. To present this process to users:

- I use dimensionality reduction (*208*) to display a planar projection of the 768-dimensional embedding space of the training data. This effectively provides the user with a map of the population of instances (patient referrals) emphasising the clustering of similar referral instances (and the teams they were respectively triaged to). This gives the user the ability to visualise how similar the current query patient is to others known to have been accepted by the different teams.

- I exploit the label-aware attention weights (of the segment-and-batch approach to ingest and classify instances, Fig 28, panel **C**) to visually highlight which instance specific tokens (or groups of tokens) contributed most weight to that instance's classification output. This enables the user to inspect what source information is driving the triage recommendation that may, for example, be useful for quickly locating data that justifies the final clinical triage decision from a multidisciplinary team.

I propose that this gives users (clinicians) the ability to see how prototypical the patient is (concerning the recommended team) and to locate the clinical text (feature importance) that drives the recommendation. In what follows, I present a

prototype user interface and show how different types of clinical notes are handled by the assisted triage model. It is important to note that the example clinical notes displayed here were written by a clinician of OHFT to model typical examples of the kinds of notes seen in practice. They are fictional, and I therefore do not present any confidential patient data.

Four examples are presented, and constructed to illustrate the following kinds of clinical notes (as described by the clinician):

1. A **mental state examination** that strongly implies the patient is experiencing a psychotic episode. A reasonable triage recommendation would therefore be an early intervention for psychosis (EIP) team. The mental state examination is a summary of psychopathology (signs and symptoms) describing a snapshot or cross-section of the patient's clinical state at the time they were examined and is primarily used by doctors (e.g. psychiatrists). See Fig. 30.

2. A clinical note summarising the reason for referral, a brief statement about the patient's history and summarising the outcome of a clinical review (with salient or headline psychopathology highlighted in less formal language) written by a third-person. This would be typical of a clinical note recording or **summarising a multidisciplinary meeting** about the patient, and may be written by a healthcare professional or administrator. This example is presented as a likely referral to the early intervention psychosis (EIP) team. See Fig. 31.

3. An instance containing **three short summative notes from different**

**healthcare professionals** – 1) the first outlining a history of clinical and imaging findings and plan followed by 2) a summary of a referral for assessment by a neuropsychologist describing collateral history and summarising an initial assessment/examination and plan followed by 3) a summary of a domiciliary visit with observations from e.g. an occupational therapist. This instance would strongly imply that this patient should be recommended for an older adult team. See Fig 32.

4. A brief note containing data where there is evidence of historical episodes of care with a different team (a learning disability team) but where the focus suggests a need for occupational therapy and suggests frailty that might suggest the patient should be under the care of an older adult team. This represents an **administrative note** where superficially, one might expect the previous care team to see the patient again, as they had done previously. See Fig 33.

I emphasise that the example instances shown are very short instances (i.e. they represent at most three sequential clinical notes in the patient's EHR, presented without the context of a complete instance containing any historical notes). So the performance of the system shown in these test cases represents the bare minimum that can be expected. All results shown are from the segment-and-batch approach.

These examples show that the model can highlight clinically salient pieces of information relating to a triage team classification decision. I included examples which were deemed clear team-specific examples: Early Intervention Psychosis (EIP) in Figure 31, where the greatest attention appears related to clear signs of schizophrenia. In Figure 32, an older adult community mental health team

**A**

App & Beh : moderate signs of self - neglect ( hair unwashed , unshaved , slightly malodorous ). Eye contact variable and appeared frequently distracted by extraneous environmental stimuli . Remained seated throughout interview , but frequently verbal ised need to leave the appointment " Is that it ? Are we finished now ?", No psychomotor abnormalities . Spe ech : normal rate and rhythm but lacking in prosody and monotonous . Some interruption of speech flow when appearing distracted ; required reorienting to topic on occasion . Content largely reflected pre occup ying thoughts ( see below ) Mood : reports reduced enjoyment for hobbies and activities , worse over past 3 months ... Perception : Does not endorse that he is experiencing internally generate stimuli however n (" I 'm not hearing anything , let alone voices which is what you mean right ?" ). During interview , appears distracted by extr aneous environmental stimuli and at times , appears distracted by , and responding to , auditory hallucinations . For example , appeared to be telling someone ( with a low volume , barely audible voice ) that " they don 't need to know that ". On balance , appears to display clear signs of responding to auditory hallucination super - imposed on distraction by extraneous environmental stimuli . In sight : for symptoms / behaviour -- unable to accommodate alternative explanations of beliefs around others interfering with objects including surveillance of mobile phones and internet devices . Holds these beliefs with almost certain conviction . for illness -- when asked if he thinks he may be experiencing a mental illness , emphatically denies this " I 'm not ill in the head you know ". For treatment -- explains he would like to talk to someone about his current difficulties , but thinks this is the police , rather than mental health professionals . Currently , does not want any treatment that would be indicated for someone experiencing a mental illness . Imp ression : On this assessment , signs of psychosis evident throughout interview and symptoms described are consistent with a psychotic episode

**B**

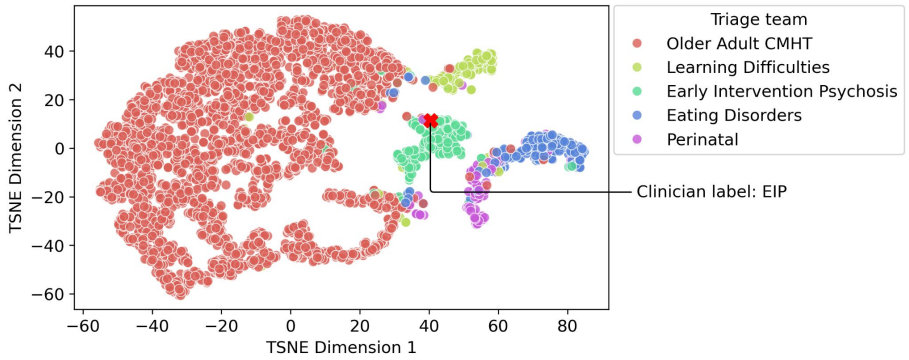


**Figure 30** Mental State Exam (MSE): **A** Visualisation of label-aware attention applied to the original synthetic text, where darker blue indicates *higher* soft-maxed attention scores driving the triage recommendation. **B** planar projection (via t-SNE) of the training data set instance embeddings, with the query instance shown as a red-cross. XXXX is used to mimic the de-identification masking typically used with the real EHR dataset.

**A**

Referral received from concerned consultant in adult mental health team . Collateral information to be obtained from the client 's social work team , particularly relating to onset of symptoms . Known to children social services as during proceedings a psychiatric opinon was sought independently . This opinion revealed a diagnosis of paranoid schizophrenia . This is however , the client 's first presentation of psychosis , but as detailed above , their appears to be a more chronic and insidious onset of symptoms . New pt assessment with Dr XXX X at outpatient department . Referred by GP with concerns about low mood , suicidality and symptoms that appear to be hallucinations ( auditory ). presenting complaint : increasingly suicidal with worsening mood over the past 10 days . Pt describes seeing and hearing things . Pt describes this has been happening for over 10 years , but hearing voices significantly worse in past 2 months to the extent he now cannot tolerate sleeping at home and has been sleeping rough in fields . In terms of risk , pt describes he stood on railway tracks for half an hour and " left it to fate " to determine if he would be killed by a passing train . A friend has reported him as missing to the police on one occasion last week

**B**

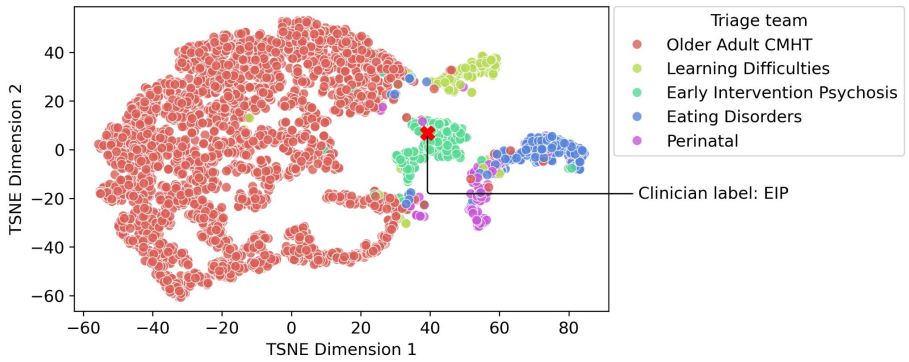


**Figure 31** Summary note from an MDT meeting or discussion. Figure details are as described in Fig. 30.

**A**

Patient came to clinic with their daughter who gave an account of the typical weekly routines for x xxx including visiting the shops and local attractions . Diagnostically , xxxx has strong risk factors for a vascular type dementia including hypertension and a history of cerebrovascular events including multiple TIAs .... I have told x xxx that she has neuroimaging showing brain changes consistent with dementia but framed this as difficulties with their memory rather than giving a diagnosis . Plan : follow up appointment with memory clinic consultant to formalise diagnosis and management / treatment plan . Referral for neuropsychology : background - 22 month history of progressive worsening of memory evidenced by struggling to recall events in the past week as well as problems recogn ising people even when well - known to them . Difficulty first noted in around 2014 after x xxx appeared to forget they had seen both their dentist and GP in the preceding 10 days . More recently , both xxxx daughter and husband noticed they appeared to " not recognise " them and spoke to them as if they were a friend or acquaintance ( rather than a close family member ). Further xxxx had some difficult recalling autobiographical memory of how they met their husband . Summary -- some recognition of the problems described by others ( daughter , husband ). On informal testing today , marked diff ic ul ites with recalling details of marriage ( date ) and children 's birth days . Word finding and verbal flu ency markedly impaired on this assessment . On working memory , learning and graded naming tests x xxx was below expected performance for their age . Plan : to continue neuropsych testing at next appoint ; I will request care coordinator input from older adult community team . Home visit : met with x xxx and her husband at home . No signs of neglect of house or environment . Observed some difficulties with using familiar objects including can opener . Family report her memory and ability to do everyday tasks ( e . g . cooking ) are variable - some days better than others - and they report that after a friend 's funeral recently , she was definitely less able to do AD L s and they wondered if she might be depressed as a result of bereavement . Memory tests will be useful in the next 3 or 4 months . No immediate safeguarding or carer burden risks . No clear role for social care input at present . I will discuss with the MDT

**B**

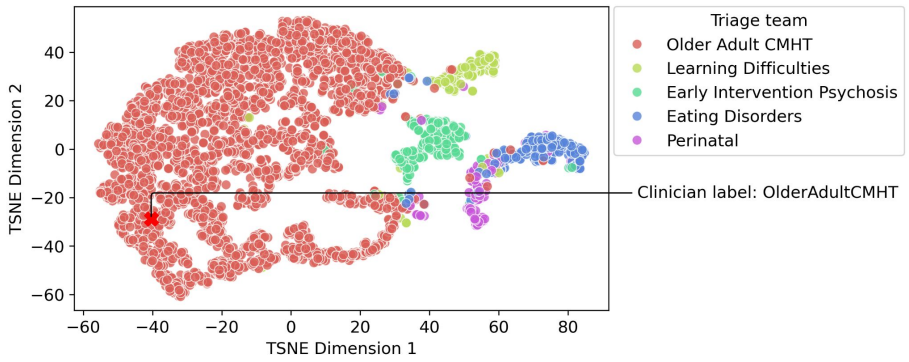


**Figure 32** A short instance summarising a patient from different healthcare professionals. Figure details are as described in Fig. 30.

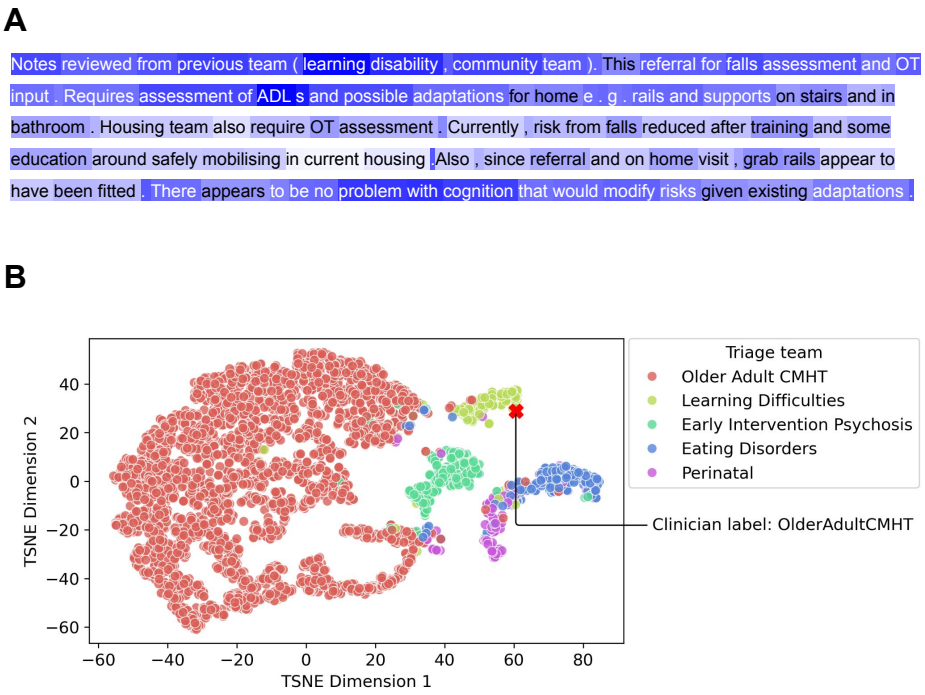


**Figure 33** An administrative note highlighting previous history with one team (learning disability) but where the content reflects needs appropriate to a different (older adult) team. Figure details are as described in Fig. 30.

(oaCMHT) example, where attention is given to a large portion of the note with an emphasis on the plan related to memory problems and next steps.

### 5.5.3 Qualitative error analysis

A qualitative analysis of the model's explained decisions revealed possible over-fitting to certain admin features of notes. For example, there were examples where attention was given to specific addresses or clinician names. This is a common problem of neural networks and language models in particular, where artefacts perfectly distinguish different classes whilst missing the important clinical content. There were examples of the segment-batch models that gave the most attention

to building names that contained keywords such as "memory". This may lead to a correct classification decision, but based on the wrong information. For example, in Oxfordshire, there is a brain imaging centre named *the Oxford Centre for Human Brain Activity (OHBA)*. This is a frequently used centre for older adults for cognitive assessments and imaging in relation to dementia - which will appear most frequently in clinical notes for patients referred to the older adults triage specialty team. The trained model could easily learn to associate documents with team-specific locations. Other errors occur due to more subtle differences and overlaps between the presenting symptoms and characteristics of different triage team referrals. For instance, *Learning Difficulties* and *Older Adult Community Mental Health* can receive patients with relatively similar trajectories.

## 5.6 Discussion

I provided a construction of LLMs to assist in triaging patient referrals to secondary care mental health teams by end-to-end ingestion of variable-length EHR data. The brute force approach showed the worst performance, likely suffering from making classification decisions for individual documents with little clinical information. The simplest approach of passing truncated clinical note histories to an LLM classification model yielded reasonable performance, with a very quick training and inference time. However, the RoBERTa-base models used are quite limited in the length of history that can be ingested (512 tokens) and perform worse for longer document histories. The clinical longformer model extended the context window to 4096 tokens and produced decent results, even with no domain

pre-training. The segment-batch approach using any LLM produced the best results, with the domain-adapted RoBERTa-OHFT achieving peak performance on the triage recommendation task.

Below I outline how the segment-batch approach aligned with the desired properties of an end-to-end system:

1. **End-to-end ingestion**: the segment-and-batch provides a plausible method for ingesting EHR data requiring only tokenisation, proceeding to create numerical representations of the data and then recommendation in a downstream classifier

2. **Resource efficient**: using LoRA and the segment-and-batch architecture, training and inference can be managed on a single GPU with fast inference (see Table 29)

3. **Ability to interrogate models**: while a full evaluation of the interpretability of the presented models is beyond the scope of my DPhil, I present initial experiments showing how the segment-and-batch model can be interrogated by using what others have previously described as interpretability through presentation (*207*)

4. **Facility to train at-scale**: with minimal human input, the models presented make use of a combination of administrative data (referral and discharge dates) and a heuristic based on the expected behaviour of clinical teams to deliver a training target for classifying any given instance. Most importantly, I utilise LLMs' ability to automatically learn a latent space of

token-sequence embeddings that can be used as the basis for downstream classification to deliver a triage recommendation.

As it stands, the segment-batch approach offers a prototype for automatically ingesting streams of clinical notes in sequence and providing a *interpretable* decision. The clinical utility of the presented model has not been validated in any real-world scenario at present, and I do not want to over-speculate its usefulness. In the current form, the pipeline can ingest a large volume of clinical notes concerning a presenting triage referral and output a recommendation alongside *interpreted features*. My vision for its future development and use would involve determining the use cases deemed acceptable by clinicians and developing some form of trial use. The intended use for a further developed model would be a clinical decision support tool that can speed up the clinician's processing of the clinical notes of their patients and have salient information brought to their attention. A benefit of the segment-batch approach is the use of small LLMs, enabling a quick and low-cost training regime that can easily be adapted to different health settings, and specialities. My work focussed on a set of sub-speciality teams in Psychiatry, but the holistic approach can be applied to any data in a similar format, making it quite flexible and generalisable.

### 5.6.1 Limitations

**Computational resource** I intentionally limited experiments to only consider model architectures that – at the time of writing – could be plausibly implemented on a modest, single GPU, with inference capabilities on a CPU. Consequently, this

means I was able to utilise smaller LLMs but I expect future work would seek to utilise more recent and larger LLMs (e.g. Llama-2 (*3*)). These limitations also meant that the segment-batch approach gracefully managed up to 12,000 tokens and this limit is a function of the GPU hardware capabilities.

**Gold Standard Labels** The derivation of which triage team a referral instance and its associated clinical notes belonged to was based on available administrative information (e.g. referral date, which is reliable in the OHFT data) and a simple heuristic for *acceptance*. Whilst this was developed with clinicians with knowledge of the clinical culture and practices of the source data (OHFT), I was unable to obtain any gold-standard decisions for a sample of clinical notes and future work will need to compare the performance, utility and alignment of the triage assistance system with clinical practice.

**Demographic Information** In this work, I did not have access to all structured variables associated with the raw clinical notes, and this included demographic information related to sex, age, ethnicity and socio-economic status. This limited the ability to discern possible biases related to the population the OHFT data relates to and ensure the triage model represents different demographics appropriately.

**Referral Bouncing and Novel Cases** One of the wider goals of this project was to provide a pipeline, model and prototype for assisting with the ingestion of clinical notes and triage team decisions. Whilst I believe this work serves as a valuable tool for representing clinical note histories for that purpose, there are issues regarding the point at which this proposed tool works and may not work. In its current

form, the envisaged use would be for clinicians to pair this with their usual triage decision-making cycle before arriving at a final decision of which team the patient should or should not be referred. This can work well if the model's decision ultimately aligns well with the clinician's own decision. But if the acting clinician, either with or without the triage model/tool, decides the current proposed team is the wrong one and essentially bounces the patient back, the next receiving team and the model are given the same clinical history and context for that patient, and the model will effectively return identical outputs and its utility changes.

Another problematic scenario would be where a presenting patient has symptoms and clinical features different to that of the clinical histories used to train the triage model. In this situation, the model would still output a likely erroneous triage recommendation with a corresponding probability value and interpretation. This is a common problem when deploying trained ML models in dynamic, real-world settings and strict guidelines are required to mitigate the risk of using the model outside of its intended use.

Therefore, I believe the utility of the work presented will not suit all scenarios, and careful considerations will need to be made and discussed with clinicians before any attempt to use it. Whilst the precision of the tool may be questionable, I think there is inherent value in the ability to meaningfully represent the clinical notes, combined with the visualisations presented in Section 5.5.2.

### 5.6.2 Conclusion

In this chapter, I have shown how to develop a bespoke and resource-efficient LLM-based pipeline for a specific healthcare task - combining components from all chapters. The segment-batch approach aimed to emulate the processes used by clinical teams in community outpatient settings within the NHS trusts within Oxfordshire. When a patient is referred, the referral details, along with the existing clinical history, are documented in the EHR system. This EHR data forms the basis for a triage decision, determining whether to accept the referral and invite the patient for a clinical assessment. Since 96% of secondary specialist mental health care is conducted in outpatient settings (*197*, *199*), delays and potentially erroneous triage decisions introduced during the referral process are recognized problems in NHS services. The envisioned use of this tool is to expedite the triage process and potentially enhance clinicians' decision-making.

Future work is required to test the acceptability and utility of the "interpretability by presentation" model (shown in 5.5.2) with clinicians trialling the triage assistance system using a mixed-methods study of clinician's behaviour when using the system. Furthermore, the ability to adapt this pipeline to different triage teams and domains would solidify the generalizability of the approach and enhance its usefulness to the clinical community.

Beyond its presented use, this work can serve as a foundation for other NHS trusts with a similar triage process.

Chapter 6

# General discussion and final thoughts

In this work, I have presented a chronological pathway for pre-training, adapting, and applying NHS-derived LLMs. The results of each line of work are as follows:

**Creating NHS LLMs** In Chapter 3, I presented a possible route for developing bespoke NHS LLMs on a budget, with adjustments to the pre-training objective yielding tangible benefits to downstream adaptation. I showed how contrastive loss functions can drastically influence the LLM's embedding space, with a direct impact on any subsequent application. Moreover, I have shown how open-source LLMs do not necessarily work well with the specialised NHS texts, with domain adaptation of smaller models remaining crucial to optimal performance on downstream tasks.

**Improving efficiency** In Chapter 4, I investigated a variety of PEFT methods for adapting these newly trained LLMs to different downstream tasks, with an impressive reduction in computation overhead without degradation in performance. Prompt learning was a powerful and resource-efficient approach but was more obtuse and inflexible than the likes of LoRA, which prevailed as the best-performing

PEFT method. The utility of PEFT methods for low-compute environments is particularly exciting in healthcare, where many specialised models are required to target specific disease areas and outcomes.

**Bringing it together - Psychiatry triage** Finally, in Chapter 5, the LLMs and LoRA methods were combined and applied to secondary mental health triage. I showed a reasonable implementation of an LLM-based pipeline for ingesting patient referral information and providing a triage team recommendation. A winning segment-batch approach was shown to distinguish between sub-speciality team referral instances reasonably well, and could also handle variable length note histories. Furthermore, this approach was accompanied by word-level attention visualisations that can be used to uncover the salient clinical information for potential use by clinicians.

Each of the models and methods developed in this work can serve as a starting point for many different avenues. The clinical utility of bespoke healthcare LLMs for specific NHS trusts and applications is clear and attainable in resource-limited environments. Based on the findings of this thesis, I have the following recommendations for tailoring LLMs for prospective healthcare or NHS trusts:

- Small LLMs can be cost-effective and powerful tools suitable to low compute environments
- The pre-training objective has a direct impact on downstream use, so consider the type of NLP task the LLM will be used for. BERT-style models remain SoTA for sequence classification and NER tasks.

- Prompt learning and PEFT methods such as LoRA are almost necessary when wanting to adapt larger LLMs for downstream tasks and using full fine-tuning is only attainable with smaller models. Using PEFT methods with much larger LLMs, such as Llama-3 (*3*) will likely be superior to fully fine-tuning smaller models.

- Developing interpretable LLM-based classification pipelines is difficult and requires adaptation to standard architectures and a high level of expertise. It is far from a solved problem, and relying on LLMs to output transparent and interrogatable decisions is of high risk.

## 6.1 Future directions

**Triage tool development** The triage referral models from Chapter 5 are certainly in the development stage and not ready for production. Whilst the classification performance of the segment-batch approach was good, there are still errors. Additionally, the interpretability of the pipeline is difficult to ascertain and its applicability to external datasets is not clear. The validation of the model would require a substantial amount of further work, with a need to develop a working prototype to be used by clinicians.

A proposed next step would be to build a functional prototype that can be used by clinicians alongside their usual workflow of interacting with the OHFT EHR system and conduct an A/B testing scenario, whereby clinicians either do or do not use the triage model for assistance during a referral triage process. One could then gather metrics of decision speed, interpretability of model outputs and triage

decisions, reliability and concordance. Importantly, the desire and acceptability of the use of patient data with an LLM-based model must be assessed.

**Alternative LLMs and architectures** The number of available LLMs in the open source community is vast and increasing, with over 600,000 downloadable models available on HuggingFace's hub today (22/04/2024) (*35*). Whilst the transformer architecture still remains dominant, there have been several improvements on certain components to reduce the computational burden of LLMs. For instance, Flash-attention (*154*) reduces the computation requirements of the transformer attention mechanism and thus the GPU requirements. There are also quantisation methods like Q-LORA (*108*), which reduce the bit-precision and storage requirement of model weights. In fact, I have already shown it is possible to fine-tune a 7 billion parameter Llama-2 model (*3*) on a single GPU by combining weight quantisation with LoRA in Chapter 4.

There are also emerging alternatives to the transformer architecture, such as Receptance Weighted Key Value (RWKV), a modified RNN network that introduces a time mixing element (*209*) and boasts the combination of positives from both transformers and RNNs. Together, this yields the parallelisable training present in transformers and inference speed benefits of RNNs. Another architecture is Mamba, a type of state space model that attempts to overcome the limitations of quadratic attention in transformers and improve performance on longer sequences (*210*). Both of these new architectures have now produced competitive LLMs, but the uptake by the community has not yet gained momentum. Nevertheless, all of my work could readily be adapted to use these newer architectures, with some

possible benefits in efficiency and performance to follow.

Perhaps the obvious next choice for healthcare data would be the latest generative AI models, (LLMs, text-to-image, multi-modal) developed by the financially well-off companies discussed earlier: GPT-3.5/4 (powering Chat-GPT) from OpenAI (*5*), BARD and Gemini from Google (*6*), Claude-2/3 from Anthropic (*7*) and Grok from xAI (*34*). We are already seeing research making use of these models for clinical use cases. One example is clinical text summary, where GPT-4 was found to be comparable to clinicians in summarising medical events from documents (*211*). Others have also found that Chat-GPT can provide preferable medical documentation to doctors in some settings (*212*), with an apparent empathetic style of writing that was agreeable to users. However, these are select studies which have primarily used social media-derived health datasets (from Reddit and twitter) or small, heavily curated datasets tailored for summarisation. Furthermore, there have also been numerous studies highlighting the potential harms of using generative AI models - in particular the tendency for the models to produce inconsistent outputs and provide responses regardless of correctness (often referred to as hallucinating). One particularly odd case had Chat-GPT encourage a man to end his life in order to help stop climate change (*212*).

That is to say, the realm of generative AI applied to healthcare, and especially mental health, is very much in its infancy and the data governance and legal landscape remains underdeveloped (*213*). I have no doubts that these tools have some genuine utility for healthcare applications and even the work presented in this thesis.

**Do we still need clinical LLMs?** My DPhil came at a time when NLP and AI have seen a massive shift in direction, with a rapid pace of model development and uptake from the research community. LLMs have become incredibly popular between 2022 and now (2024), and many are built to be a foundational model, with applicability to any domain. Now, one can wonder whether domain-specific LLMs, such as clinical LLMs, are still needed (*8*). Whilst my work with smaller, and now older, LLMs generally showed a specific clinical LLM was desirable, it stands to reason that the latest LLMs, or future LLMs (think Llama-3, or the rumoured GPT-5) will be trained on even more clinical text and might all but remove the need to domain adapt.

## 6.2 Ethical considerations

The majority of my DPhil, as well as the wider machine learning research community, tend to focus on performance and evaluation metrics. This tends to force us to utilise these big and complex deep learning models to stay competitive. However, this trend has certain ethical ramifications I would like to discuss below:

**Data privacy and patient safety** Research suggests that LLMs can sufficiently retain elements of their training data (*87*, *88*, *214*). Whilst this may not inherently diminish the usefulness of the models trained for the given OHFT NHS trust, it does mean the trained LLMs from my work cannot be readily shared outside the trusted research environment as it would pose a risk to data security. Therefore, to scale these approaches and apply them to external datasets, it would require

careful consideration and infrastructures in place to allow any model sharing.

Another fundamental issue when working with real-world big data is the inherent biases present in the training data - in particular demographic and algorithmic biases. In Chapter 5, I attempted to develop a decision support tool for triage using a dataset from Oxfordshire - where there is a large percentage of students and middle to upper-class residents of predominantly white ethnicity. The result of training any machine and deep learning models on such data has the potential to generalise poorly to other populations not well represented in the training data itself. For example, the triage classification model I produced may work reasonably well for the type of referrals commonly received - which in the OHFT NHS trust was older, white British adults[1]. The type of history and type of problems encountered by the typical person in this dataset may differ greatly from the experience of other demographics. Consequently, the triage model would not necessarily adapt to new types of patients appropriately without further training.

This demographic and algorithmic bias is a widely recognised problem in AI and research generally (*215–217*). A potential related problem that is unique to secondary mental healthcare is the differential treatment and reluctance to self-refer in certain communities. Surveys conducted in 2018 with LGBTQ+ communities found that susceptibility to stigmatisation and discrimination as reasons for not seeking care for mental health problems (*202*), and with this comes possible under-representation in the data leading to AI tools that inherit these biases.

[1]Unfortunately, I did not have access to the demographic information of the subset of data used in my work

**Transparency** A related ethical discussion revolves around the transparency of AI models. The public discourse around LLMs and the inevitable salesmanship presented by the large companies producing popular AI tools has made it quite difficult to discern the truth of how useful these tools are. The term black-box models is frequently used to describe large neural networks like LLMs. Black-box relates to the only visible components of a model's decision being the original input and the output - with little understanding of what transformations of the input actually take place to arrive at the decision. For low-risk use cases, such as movie recommendations, this may be satisfactory, but with healthcare applications, this is not. The field of psychiatry and precision medicine has a history of demanding model transparency as a crucial feature of any tool (*207*). Using my work as an example, the triage referral models were outputting a decision of which triage team a referral should go to based on the presented clinical notes. The triage model presented in Chapter 5, using the segment-batch approach, was my attempt to align with this ethos by providing a direct mapping of weights learnt by the model between the tokens in a document and the triage team classification decision. Whilst this provides a certain level of insight, it is not a fully transparent model as a large portion of the inner workings, i.e. the base LLM that produces the embeddings, remain hidden. It is a step in the right direction, and further work would be required to improve the understandability of the triage models' workings and outputs.

**Green AI** A final consideration for my work and prospective AI in healthcare is the environmental impact these tools inevitably have. The very recent Llama-3 set of models (*36*) released in April 2024 consisted of approximately 7.7 million GPU hours (thousands of GPUs are used in parallel so the actual time taken is just a matter of days or weeks), amounting to 2290 tonnes of $CO_2eq$ (Carbon dioxide equivalent) (*36*) - which they claim is offset by Meta's sustainability program, but this is difficult to confirm. To put this into perspective, an average US citizen emits around 21.8 tonnes $CO_2eq$ per year. Granted, the pre-training is typically the major computational burden of foundational LLMs, but the issue that has arisen is that foundational models are seemingly updated and replaced routinely. Llama-2 (*3*) was pre-trained with similar costings and released in July 2023, and within 9 months Meta released Llama-3. In fact, each of the major tech companies producing LLMs and other AI models is routinely releasing new models to stay up-to-date and, importantly, commercially competitive. This energy consumption does not stop once the models are released, as many of these models require GPUs to be adapted to new tasks or used in inference mode. Every time research is conducted using these models, or a person interacts with the likes of Chat-GPT, there will be an associated energy cost. My own work involved many fine-tuning experiments and hyperparameter tuning to obtain optimal performance, all of which required considerable time and resources: I have estimated the carbon emissions produced over the course of the DPhil using a crude estimate of 500 hours of GPU training time using an NVIDIA RTX 2080 Ti (TDP of 250W) at a rate of 0.432 $kgCO_2eq/kWh$ (*218*), resulting in approximately 75.6 kg of carbon emitted. This is roughly equivalent to driving 305 km in an average non-electric car, or 37.8 kgs of coal burned. This may seem small, but consider that I am only one student

in a field with many researching at a similar scale. With the large costs associated with using LLMs, one could wonder whether alternative models and tools are better suited to certain problems.

## 6.3 Closing remarks

Overall, AI and LLMs hold great promise for healthcare and psychiatry, but great care must be taken to ensure these tools are used only when there is real value, as the associated costs and risks remain high. This thesis has explored the development, adaptation, and application of NHS-derived LLMs, providing valuable insights into the potential and challenges of using these technologies in mental health care settings. The work presented here serves as a framework for other NHS trusts where the raw text contained within EHRS is used. We have demonstrated that it is possible to create bespoke, domain-specific LLMs for healthcare applications, even in resource-limited environments. The exploration of PEFT methods, particularly LoRA, has shown promising results for adapting these models to specific tasks without excessive computational overhead. Our application of these techniques to psychiatric triage demonstrates the potential for LLM-based systems to assist in clinical decision-making. However, it also highlights the challenges in developing truly interpretable and transparent AI systems for healthcare – a crucial consideration for any future implementations. As we look to the future, the rapid pace of development in AI and LLMs presents both opportunities and challenges. While larger, more general-purpose models may reduce the need for domain-specific pre-training, the ethical considerations surrounding data privacy,

algorithmic bias, and environmental impact will remain critical areas of concern. The possibility of extending this work with newer LLMs and applications is very attainable, and I am excited to see what follows in this space. Future research should focus on:

- Improving the interpretability and transparency of LLM-based clinical decision support tools.
- Addressing issues of bias and ensuring equitable performance across diverse patient populations.
- Exploring ways to reduce the environmental impact of AI in healthcare, possibly through more efficient architectures or training methods.
- Conducting rigorous clinical validation studies to assess the real-world impact and safety of these technologies.

In conclusion, while the potential of AI and LLMs in psychiatry is immense, their development and implementation must be guided by careful consideration of ethical implications, clinical utility, and patient safety. As we continue to advance in this field, collaboration between clinicians, data scientists, ethicists, and policymakers will be crucial to realizing the benefits of these technologies while mitigating their risks.

# Bibliography


(*1*) Searle, T., Ibrahim, Z., Teo, J., and Dobson, R. (2021). Estimating redundancy in clinical text. *Journal of Biomedical Informatics 124*, 103938.

(*2*) Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, arXiv:1810.04805 [cs], 2019.

(*3*) Touvron, H. et al. Llama 2: Open Foundation and Fine-Tuned Chat Models, arXiv:2307.09288 [cs], 2023.

(*4*) Lester, B., Al-Rfou, R., and Constant, N. The Power of Scale for Parameter-Efficient Prompt Tuning, arXiv:2104.08691 [cs], 2021.

(*5*) OpenAI GPT-4 Technical Report, arXiv:2303.08774 [cs], 2023.

(*6*) Team, G. et al. Gemini: A Family of Highly Capable Multimodal Models, arXiv:2312.11805 [cs], 2023.

(*7*) Claude 2, en.

(*8*) Lehman, E., Hernandez, E., Mahajan, D., Wulff, J., Smith, M. J., Ziegler, Z., Nadler, D., Szolovits, P., Johnson, A., and Alsentzer, E. Do We Still Need Clinical Language Models?, en, arXiv:2302.08091 [cs], 2023.

(*9*) Moradi, M., Blagec, K., Haberl, F., and Samwald, M. GPT-3 Models are Poor Few-Shot Learners in the Biomedical Domain, en, 2021.

(*10*) Leaman, R., Khare, R., and Lu, Z. (2015). Challenges in clinical natural language processing for automated disorder normalization. *Journal of Biomedical Informatics 57*, 28–37.

(*11*) Taylor, N., Zhang, Y., Joyce, D. W., Gao, Z., Kormilitzin, A., and Nevado-Holgado, A. (2023). Clinical Prompt Learning With Frozen Language Models. *IEEE Transactions on Neural Networks and Learning Systems*, 1–11.

(*12*) Taylor, N., Ghose, U., Rohanian, O., Nouriborji, M., Kormilitzin, A., Clifton, D., and Nevado-Holgado, A. Efficiency at Scale: Investigating the Performance of Diminutive Language Models in Clinical Tasks, arXiv:2402.10597 [cs], 2024.

(*13*) Chowdhery, A. et al. PaLM: Scaling Language Modeling with Pathways, arXiv:2204.02311 [cs], 2022.

(*14*) Hoffmann, J. et al. Training Compute-Optimal Large Language Models, arXiv:2203.15556 [cs], 2022.

(*15*) Hinton, G. E. How Neural Networks Learn from Experience, tech. rep., Issue: 3 Volume: 267, 1992, pp 144–151.

(*16*) Jones, K. S. In *Current Issues in Computational Linguistics: In Honour of Don Walker*, Zampolli, A., Calzolari, N., and Palmer, M., Eds.; Linguistica Computazionale; Springer Netherlands: Dordrecht, 1994, pp 3–16.

(*17*) Wang, H., Li, J., Wu, H., Hovy, E., and Sun, Y. (2023). Pre-Trained Language Models and Their Applications. *Engineering 25*, 51–65.

(*18*) Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences 79*, Publisher: Proceedings of the National Academy of Sciences, 2554–2558.

(*19*) Mikolov, T., Karafiát, M., Burget, L., Černocký, J., and Khudanpur, S. In *Interspeech 2010*, ISCA: 2010, pp 1045–1048.

(*20*) Pascanu, R., Mikolov, T., and Bengio, Y. On the difficulty of training Recurrent Neural Networks, arXiv:1211.5063 [cs], 2013.

(*21*) Graves, A. Generating Sequences With Recurrent Neural Networks, arXiv:1308.0850 [cs], 2014.

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

(*23*) Sak, H., Senior, A., and Beaufays, F. Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition, arXiv:1402.1128 [cs, stat], 2014.

(*24*) Bahdanau, D., Cho, K., and Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate, arXiv:1409.0473 [cs, stat], 2016.

(*25*) Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. DOI: 10.48550/ARXIV.1301.3781.

(*26*) Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. In *Advances in Neural Information Processing Systems*, ed. by Guyon, I., Luxburg, U. V., Bengio, S., Wallach,

H., Fergus, R., Vishwanathan, S., and Garnett, R., Curran Associates, Inc.: 2017; Vol. 30.

(*27*) Abdin, M. et al. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone, arXiv:2404.14219 [cs], 2024.

(*28*) Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving Language Understanding by Generative Pre-Training.

(*29*) Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. 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)*, Association for Computational Linguistics: Minneapolis, Minnesota, 2019, pp 4171–4186.

(*30*) Brown, T. et al. In *Advances in Neural Information Processing Systems*, ed. by Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., Curran Associates, Inc.: 2020; Vol. 33, pp 1877–1901.

(*31*) Jiang, A. Q. et al. Mistral 7B, arXiv:2310.06825 [cs], 2023.

(*32*) Manyika, J., and Hsiao, S. An overview of Bard: an early experiment with generative AI.

(*33*) psychiatry dept, d. Facts and figures about the Department of Psychiatry 2023, en.

(*34*) Open Release of Grok-1.

(*35*) Wolf, T. et al. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Association for Computational Linguistics: Online, 2020, pp 38–45.

(*36*) Introducing Meta Llama 3: The most capable openly available LLM to date, en.

(*37*) Hughes, A. Phi-2: The surprising power of small language models, en-US, 2023.

(*38*) Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S., and Leahy, C. The Pile: An 800GB Dataset of Diverse Text for Language Modeling, arXiv:2101.00027 [cs], 2020.

(*39*) Tunstall, L., Reimers, N., Jo, U. E. S., Bates, L., Korat, D., Wasserblat, M., and Pereg, O. Efficient Few-Shot Learning Without Prompts, en, arXiv:2209.11055 [cs], 2022.

(*40*) Gutiérrez, B. J., McNeal, N., Washington, C., Chen, Y., Li, L., Sun, H., and Su, Y. (2022). Thinking about GPT-3 In-Context Learning for Biomedical IE? Think Again. Publisher: arXiv Version Number: 3, DOI: 10.48550/ARXIV.2203.08410.

(*41*) Sun, X., Li, X., Li, J., Wu, F., Guo, S., Zhang, T., and Wang, G. Text Classification via Large Language Models, en, arXiv:2305.08377 [cs], 2023.

(*42*) Tang, R., Han, X., Jiang, X., and Hu, X. Does Synthetic Data Generation of LLMs Help Clinical Text Mining?, arXiv:2303.04360 [cs], 2023.

(*43*) Rohanian, O., Nouriborji, M., and Clifton, D. A. Exploring the Effectiveness of Instruction Tuning in Biomedical Language Processing, en, arXiv:2401.00579 [cs], 2023.

(*44*) Alsentzer, E., Murphy, J. R., Boag, W., Weng, W.-H., Jin, D., Naumann, T., and McDermott, M. B. A. (2019). Publicly Available Clinical BERT Embeddings.

(*45*) Rohanian, O., Nouriborji, M., Jauncey, H., Kouchaki, S., Group, I. C. C., Clifton, L., Merson, L., and Clifton, D. A. Lightweight Transformers for Clinical Natural Language Processing, en, arXiv:2302.04725 [cs], 2023.

(*46*) Johnson, A. E., Pollard, T. J., Shen, L., Lehman, L. W. H., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Anthony Celi, L., and Mark, R. G. (2016). MIMIC-III, a freely accessible critical care database. *Scientific Data 3*, Publisher: Nature Publishing Groups, DOI: 10.1038/sdata.2016.35.

(*47*) Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., and Kang, J. (2019). BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *CoRR abs/1901.08746*.

(*48*) Huang, K., Altosaar, J., and Ranganath, R. (2019). ClinicalBERT: Modeling Clinical Notes and Predicting Hospital Readmission.

(*49*) Alsentzer, E., Murphy, J., Boag, W., Weng, W.-H., Jindi, D., Naumann, T., and McDermott, M. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, Association for Computational Linguistics: Minneapolis, Minnesota, USA, 2019, pp 72–78.

(*50*) Gu, Y., Tinn, R., Cheng, H., Lucas, M., Usuyama, N., Liu, X., Naumann, T., Gao, J., and Poon, H. Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing, 2020.

(*51*) Yasunaga, M., Leskovec, J., and Liang, P. In *Association for Computational Linguistics (ACL)*, 2022.

(*52*) Yasunaga, M., Bosselut, A., Ren, H., Zhang, X., Manning, C. D., Liang, P., and Leskovec, J. Deep Bidirectional Language-Knowledge Graph Pretraining, arXiv:2210.09338 [cs], 2022.

(*53*) Luo, R., Sun, L., Xia, Y., Qin, T., Zhang, S., Poon, H., and Liu, T.-Y. (2022). BioGPT: Generative Pre-trained Transformer for Biomedical Text Generation and Mining. *Briefings in Bioinformatics 23*, arXiv:2210.10341 [cs], bbac409.

(*54*) BioMedLM: a Domain-Specific Large Language Model for Biomedical Text.

(*55*) Chen, Z. et al. MEDITRON-70B: Scaling Medical Pretraining for Large Language Models, arXiv:2311.16079 [cs], 2023.

(*56*) Singhal, K. et al. (2023). Large language models encode clinical knowledge. *Nature 620*, Publisher: Nature Publishing Group, 172–180.

(*57*) Van Aken, B., Papaioannou, J.-M., Mayrdorfer, M., Budde, K., Gers, F., and Loeser, A. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, Association for Computational Linguistics: Online, 2021, pp 881–893.

(*58*) Belkadi, S., Han, L., Wu, Y., and Nenadic, G. In *2023 IEEE International Conference on Big Data (BigData)*, IEEE: Sorrento, Italy, 2023, pp 3660–3669.

(*59*) Moradi, M., Blagec, K., Haberl, F., and Samwald, M. GPT-3 Models are Poor Few-Shot Learners in the Biomedical Domain, 2021.

(*60*) Tayefi, M., Ngo, P., Chomutare, T., Dalianis, H., Salvi, E., Budrionis, A., and Godtliebsen, F. (2021). Challenges and opportunities beyond structured data in analysis of electronic health records. *WIREs Computational Statistics 13*, DOI: 10.1002/wics.1549.

(*61*) NHS, U. Abbreviations you may find in your health records - NHS App help and support, en, 2021.

(*62*) Organization(WHO), W. H., *The ICD-10 classification of mental and behavioural disorders: Diagnostic criteria for research*; World Health Organization: 1993.

(*63*) SNOMED CT, eng, Product, Program, and Project Descriptions, Publisher: U.S. National Library of Medicine.

(*64*) Kraljevic, Z. et al. Multi-domain Clinical Natural Language Processing with MedCAT: the Medical Concept Annotation Toolkit, arXiv:2010.01165 [cs], 2021.

(*65*) Kormilitzin, A., Vaci, N., Liu, Q., and Nevado-Holgado, A. (2021). Med7: A transferable clinical natural language processing model for electronic health records. *Artificial Intelligence in Medicine 118*, 102086.

(*66*) Le Glaz, A., Haralambous, Y., Kim-Dufor, D.-H., Lenca, P., Billot, R., Ryan, T. C., Marsh, J., DeVylder, J., Walter, M., Berrouiguet, S., and Lemey, C. (2021). Machine Learning and Natural Language Processing in Mental Health: Systematic Review. *Journal of Medical Internet Research 23*, e15708.

(67) spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing, en-us, 2017.

(68) Tu, H., Han, L., and Nenadic, G. Extraction of Medication and Temporal Relation from Clinical Text using Neural Language Models, en, arXiv:2310.02229 [cs], 2023.

(69) Li, I., Pan, J., Goldwasser, J., Verma, N., Wong, W. P., Nuzumlalı, M. Y., Rosand, B., Li, Y., Zhang, M., Chang, D., Taylor, R. A., Krumholz, H. M., and Radev, D. (2022). Neural Natural Language Processing for unstructured data in electronic health records: A review. *Computer Science Review 46*, 100511.

(70) Kulkarni, D., Ghosh, A., Girdhari, A., Liu, S., Vance, L. A., Unruh, M., and Sarkar, J. (2024). Enhancing pre-trained contextual embeddings with triplet loss as an effective fine-tuning method for extracting clinical features from electronic health record derived mental health clinical notes. *Natural Language Processing Journal 6*, 100045.

(71) Agrawal, M., Hegselmann, S., Lang, H., Kim, Y., and Sontag, D. Large Language Models are Few-Shot Clinical Information Extractors, en, arXiv:2205.12689 [cs], 2022.

(72) Hu, Y., Chen, Q., Du, J., Peng, X., Keloth, V. K., Zuo, X., Zhou, Y., Li, Z., Jiang, X., Lu, Z., Roberts, K., and Xu, H. (2024). Improving large language models for clinical named entity recognition via prompt engineering. *Journal of the American Medical Informatics Association*, ocad259.

(73) Li, Y., Wehbe, R. M., Ahmad, F. S., Wang, H., and Luo, Y. Clinical-Longformer and Clinical-BigBird: Transformers for long clinical sequences, arXiv:2201.11838 [cs], 2022.

(74) Huang, C.-W., Tsai, S.-C., and Chen, Y.-N. In *Proceedings of the 4th Clinical Natural Language Processing Workshop*, Association for Computational Linguistics: Seattle, WA, 2022, pp 10–20.

(75) Liu, S., Wang, X., Hou, Y., Li, G., Wang, H., Xu, H., Xiang, Y., and Tang, B. (2023). Multimodal Data Matters: Language Model Pre-Training Over Structured and Unstructured Electronic Health Records. *IEEE journal of biomedical and health informatics 27*, 504–514.

(76) Miotto, R., Li, L., Kidd, B. A., and Dudley, J. T. (2016). Deep Patient: An Unsupervised Representation to Predict the Future of Patients from the Electronic Health Records. *Scientific Reports 6*, Publisher: Nature Publishing Group, 26094.

(77) Le, Q. V., and Mikolov, T. Distributed Representations of Sentences and Documents, arXiv:1405.4053 [cs], 2014.

(78) Darabi, S., Kachuee, M., Fazeli, S., and Sarrafzadeh, M. (2020). TAPER: Time-aware patient EHR representation. *IEEE Journal of Biomedical and Health Informatics 24*, Publisher: Institute of Electrical and Electronics Engineers Inc., 3268–3275.

(79) Kraljevic, Z., Bean, D., Shek, A., Bendayan, R., Hemingway, H., Yeung, J. A., Deng, A., Baston, A., Ross, J., Idowu, E., Teo, J. T., and Dobson, R. J. Foresight – Generative Pretrained Transformer (GPT) for Modelling of Patient Timelines using EHRs, arXiv:2212.08072 [cs], 2023.

(*80*) Li, Y., Rao, S., Solares, J. R. A., Hassaine, A., Ramakrishnan, R., Canoy, D., Zhu, Y., Rahimi, K., and Salimi-Khorshidi, G. (2020). BEHRT: Transformer for Electronic Health Records. *Scientific Reports 10*, Publisher: Nature Publishing Group, 7155.

(*81*) Rasmy, L., Xiang, Y., Xie, Z., Tao, C., and Zhi, D. (2021). Med-BERT: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *npj Digital Medicine 4*, Number: 1 Publisher: Nature Publishing Group, 1–13.

(*82*) Niu, H., Omitaomu, O. A., Langston, M. A., Olama, M., Ozmen, O., Klasky, H. B., Laurio, A., Ward, M., and Nebeker, J. (2024). EHR-BERT: A BERT-based model for effective anomaly detection in electronic health records. *Journal of Biomedical Informatics 150*, 104605.

(*83*) Johnson, A. E., Pollard, T. J., Shen, L., Lehman, L. W. H., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L. A., and Mark, R. G. (2016). MIMIC-III, a freely accessible critical care database. *Scientific Data 3*, DOI: 10.1038/sdata.2016.35.

(*84*) Laparra, E., Mascio, A., Velupillai, S., and Miller, T. (2021). A Review of Recent Work in Transfer Learning and Domain Adaptation for Natural Language Processing of Electronic Health Records. *Yearbook of Medical Informatics 30*, 239–244.

(*85*) Yang, K., Zhang, T., Kuang, Z., Xie, Q., Huang, J., and Ananiadou, S. MentaLLaMA: Interpretable Mental Health Analysis on Social Media with Large Language Models, arXiv:2309.13567 [cs], 2024.

(*86*) CogStack Large Language Model releases - CogStack, en-US, Section: News, 2023.

(*87*) Lehman, E., Jain, S., Pichotta, K., Goldberg, Y., and Wallace, B. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics: Online, 2021, pp 946–959.

(*88*) Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., Oprea, A., and Raffel, C. Extracting Training Data from Large Language Models, arXiv:2012.07805 [cs], 2021.

(*89*) Carlini, N., Jagielski, M., Choquette-Choo, C. A., Paleka, D., Pearce, W., Anderson, H., Terzis, A., Thomas, K., and Tramèr, F. Poisoning Web-Scale Training Datasets is Practical, arXiv:2302.10149 [cs], 2023.

(*90*) Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X., and Gao, J. Large Language Models: A Survey, en, arXiv:2402.06196 [cs], 2024.

(*91*) Howell, K., Christian, G., Fomitchov, P., Kehat, G., Marzulla, J., Rolston, L., Tredup, J., Zimmerman, I., Selfridge, E., and Bradley, J. The economic trade-offs of large language models: A case study.

(*92*) Chen, L., Zaharia, M., and Zou, J. FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance, arXiv:2305.05176 [cs], 2023.

(*93*) insights uk, u. Product, en, tech. rep.

(*94*) Workshop, B. et al. BLOOM: A 176B-Parameter Open-Access Multilingual Language Model, arXiv:2211.05100 [cs], 2023.

(*95*) NHS England » Our 2023/24 business plan.

(*96*) Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling Laws for Neural Language Models, arXiv:2001.08361 [cs, stat], 2020.

(*97*) Fedus, W., Zoph, B., and Shazeer, N. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity, arXiv:2101.03961 [cs], 2022.

(*98*) Zong, S., Seltzer, J., Jiahua, Pan, Cheng, K., and Lin, J. Which Model Shall I Choose? Cost/Quality Trade-offs for Text Classification Tasks, en, arXiv:2301.07006 [cs], 2023.

(*99*) Luo, Y., Kataoka, Y., Ostinelli, E. G., Cipriani, A., and Furukawa, T. A. (2020). National Prescription Patterns of Antidepressants in the Treatment of Adults With Major Depression in the US Between 1996 and 2015: A Population Representative Survey Based Analysis. *Frontiers in Psychiatry 11*.

(*100*) Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res. 21*, 140:1–140:67.

(*101*) Weller, O., Seppi, K., and Gardner, M. When to Use Multi-Task Learning vs Intermediate Fine-Tuning for Pre-Trained Encoder Transfer Learning, en, arXiv:2205.08124 [cs], 2022.

(*102*) Brown, T. B. et al. Language Models are Few-Shot Learners, arXiv:2005.14165 [cs], 2020.

(*103*) van Aken, B., Winter, B., Löser, A., and Gers, F. A. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*; arXiv:1909.04925 [cs], 2019, pp 1823–1832.

(*104*) Pfeiffer, J., Kamath, A., Rücklé, A., Cho, K., and Gurevych, I. AdapterFusion: Non-Destructive Task Composition for Transfer Learning, arXiv:2005.00247 [cs], 2021.

(*105*) Li, X. L., and Liang, P. Prefix-Tuning: Optimizing Continuous Prompts for Generation, arXiv:2101.00190 [cs], 2021.

(*106*) Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-Rank Adaptation of Large Language Models, en, arXiv:2106.09685 [cs], 2021.

(*107*) Frantar, E., and Alistarh, D. SparseGPT: Massive Language Models Can Be Accurately Pruned in One-Shot, arXiv:2301.00774 [cs], 2023.

(*108*) Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. QLoRA: Efficient Finetuning of Quantized LLMs, en, 2023.

(*109*) Mangrulkar, S., Gugger, S., Debut, L., Belkada, Y., Paul, S., and Bossan, B. PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods, https://github.com/huggingface/peft, 2022.

(*110*) Paszke, A. et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library, arXiv:1912.01703 [cs, stat], 2019.

(*111*) Ma, S., Wang, H., Ma, L., Wang, L., Wang, W., Huang, S., Dong, L., Wang, R., Xue, J., and Wei, F. The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits, arXiv:2402.17764 [cs], 2024.

(*112*) (2022). Global, regional, and national burden of 12 mental disorders in 204 countries and territories, 1990–2019: a systematic analysis for the Global Burden of Disease Study 2019. *The Lancet Psychiatry 9*, Publisher: Elsevier, 137–150.

(*113*) Digital, N. Mental Health Bulletin: 2019-20 Annual Report, tech. rep., 2020.

(*114*) Kraljevic, Z., Shek, A., Bean, D., Bendayan, R., Teo, J., and Dobson, R. MedGPT: Medical Concept Prediction from Clinical Narratives, arXiv:2107.03134 [cs], 2021.

(*115*) Li, H., Zhang, R., Lee, Y.-C., Kraut, R. E., and Mohr, D. C. (2023). Systematic review and meta-analysis of AI-based conversational agents for promoting mental health and well-being. *npj Digital Medicine 6*, Publisher: Nature Publishing Group, 1–14.

(*116*) Sandmann, S., Riepenhausen, S., Plagwitz, L., and Varghese, J. (2024). Systematic analysis of ChatGPT, Google search and Llama 2 for clinical decision support tasks. *Nature Communications 15*, Publisher: Nature Publishing Group, 2050.

(*117*) Hong, W. S., Haimovich, A. D., and Taylor, R. A. (2018). Predicting hospital admission at emergency department triage using machine learning. *PLOS ONE 13*, Publisher: Public Library of Science, e0201016.

(*118*) Sánchez-Salmerón, R., Gómez-Urquiza, J. L., Albendín-García, L., Correa-Rodríguez, M., Martos-Cabrera, M. B., Velando-Soriano, A., and Suleiman-Martos, N. (2022). Machine learning methods applied to triage in emergency services: A systematic review. *International Emergency Nursing 60*, 101109.

(*119*) Singh, V. K., Shrivastava, U., Bouayad, L., Padmanabhan, B., Ialynytchev, A., and Schultz, S. K. (2018). Machine learning for psychiatric patient triaging: an investigation of cascading classifiers. *Journal of the American Medical Informatics Association : JAMIA 25*, 1481–1487.

(*120*) Chaturvedi, J., Velupillai, S., Stewart, R., and Roberts, A. Identifying Mentions of Pain in Mental Health Records Text: A Natural Language Processing Approach, arXiv:2304.01240 [cs], 2023.

(*121*) Dawoodbhoy, F. M., Delaney, J., Cecula, P., Yu, J., Peacock, I., Tan, J., and Cox, B. (2021). AI in patient flow: applications of artificial intelligence to improve patient flow in NHS acute mental health inpatient units. *Heliyon 7*, e06993.

(*122*) Nguyen, H., Meczner, A., Burslam-Dawe, K., and Hayhoe, B. (2022). Triage Errors in Primary and Pre–Primary Care. *Journal of Medical Internet Research 24*, e37209.

(*123*) Allison, S., Bastiampillai, T., Kisely, S., and Looi, J. C. (2024). Unpeeling the onion: Digital triage and monitoring of general practice, private psychiatry, and psychology. *Australasian Psychiatry 32*, 118–120.

(*124*) Watson, T., Tindall, R., Patrick, A., and Moylan, S. (2023). Mental health triage tools: A narrative review. *International Journal of Mental Health Nursing 32*, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/inm.13073, 352–364.

(*125*) Cartwright, D. J. (2013). ICD-9-CM to ICD-10-CM Codes: What? Why? How? *Advances in Wound Care 2*, 588–592.

(*126*) ICD-11.

(*127*) England, N. NHS National patient safety incident reports, tech. rep., 2022.

(*128*) AkriviaHealth, tech. rep., 2024.

(*129*) Kingma, D. P., and Ba, J. Adam: A Method for Stochastic Optimization, arXiv:1412.6980 [cs], 2017.

(*130*) Loshchilov, I., and Hutter, F. (2019). DECOUPLED WEIGHT DECAY REGULARIZATION.

(*131*) Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. DOI: 10.48550/ARXIV.1907.11692.

(*132*) Rumelhart, D. E., and McClelland, J. L. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, Conference Name: Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations; MIT Press: 1987, pp 318–362.

(*133*) LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature 521*, 436–444.

(*134*) Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach, arXiv:1907.11692 [cs], 2019.

(*135*) Cunningham, P., Cord, M., and Delany, S. J. In *Machine Learning Techniques for Multimedia: Case Studies on Organization and Retrieval*, Cord, M., and Cunningham, P., Eds.; Cognitive Technologies; Springer: Berlin, Heidelberg, 2008, pp 21–49.

(*136*) Alloghani, M., Al-Jumeily, D., Mustafina, J., Hussain, A., and Aljaaf, A. J. In *Supervised and Unsupervised Learning for Data Science*, Berry, M. W., Mohamed, A., and Yap, B. W., Eds.; Unsupervised and Semi-Supervised Learning; Springer International Publishing: Cham, 2020, pp 3–21.

(*137*) Parnami, A., and Lee, M. Learning from Few Examples: A Summary of Approaches to Few-Shot Learning, arXiv:2203.04291 [cs], 2022.

(*138*) Sudlow, C. et al. (2015). UK Biobank: An Open Access Resource for Identifying the Causes of a Wide Range of Complex Diseases of Middle and Old Age. *PLOS Medicine 12*, Publisher: Public Library of Science, e1001779.

(*139*) Jones, K. H., Ford, D. V., Ellwood-Thompson, S., and Lyons, R. A. (2016). The UK Secure eResearch Platform for public health research: a case study. *The Lancet 388*, Publisher: Elsevier, S62.

(*140*) Ford, D. V., Jones, K. H., Verplancke, J.-P., Lyons, R. A., John, G., Brown, G., Brooks, C. J., Thompson, S., Bodger, O., Couch, T., and Leake, K.

(2009). The SAIL Databank: building a national architecture for e-health research and evaluation. *BMC health services research 9*, 157.

(*141*) Wolf, T. et al. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Association for Computational Linguistics: Online, 2020, pp 38–45.

(*142*) Lester, B., Al-Rfou, R., and Constant, N. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics: Online and Punta Cana, Dominican Republic, 2021, pp 3045–3059.

(*143*) Muennighoff, N., Su, H., Wang, L., Yang, N., Wei, F., Yu, T., Singh, A., and Kiela, D. Generative Representational Instruction Tuning, en, arXiv:2402.09906 [cs], 2024.

(*144*) Huang, K., Altosaar, J., and Ranganath, R. ClinicalBERT: Modeling Clinical Notes and Predicting Hospital Readmission, 2019.

(*145*) Geiping, J., and Goldstein, T. Cramming: Training a Language Model on a Single GPU in One Day, en, arXiv:2212.14034 [cs], 2022.

(*146*) Reimers, N., and Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, arXiv:1908.10084 [cs], 2019.

(*147*) Giorgi, J., Nitski, O., Wang, B., and Bader, G. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Association for Computational Linguistics: Online, 2021, pp 879–895.

(*148*) Carlsson, F., Gyllensten, A. C., Gogoulou, E., Hellqvist, E. Y., and Sahlgren, M. In *International Conference on Learning Representations*, 2021.

(*149*) Vreugdenhil, J., Somra, S., Ket, H., Custers, E. J. F. M., Reinders, M. E., Dobber, J., and Kusurkar, R. A. (2023). Reasoning like a doctor or like a nurse? A systematic integrative review. *Frontiers in Medicine 10*, 1017783.

(*150*) Gao, T., Yao, X., and Chen, D. SimCSE: Simple Contrastive Learning of Sentence Embeddings, arXiv:2104.08821 [cs], 2022.

(*151*) Liu, F., Shareghi, E., Meng, Z., Basaldella, M., and Collier, N. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp 4228–4238.

(*152*) Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The Long-Document Transformer, 2020.

(*153*) Pope, R., Douglas, S., Chowdhery, A., Devlin, J., Bradbury, J., Levskaya, A., Heek, J., Xiao, K., Agrawal, S., and Dean, J. Efficiently Scaling Transformer Inference, arXiv:2211.05102 [cs], 2022.

(*154*) Dao, T., Fu, D. Y., Ermon, S., Rudra, A., and Ré, C. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness, arXiv:2205.14135 [cs], 2022.

(*155*) Gunel, B., Du, J., Conneau, A., and Stoyanov, V. Supervised Contrastive Learning for Pre-trained Language Model Fine-tuning, arXiv:2011.01403 [cs], 2021.

(*156*) Davies, D. L., and Bouldin, D. W. (1979). A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-1*, Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence, 224–227.

(*157*) Uzuner, Ö., South, B. R., Shen, S., and DuVall, S. L. (2011). 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association : JAMIA 18*, 552–556.

(*158*) Sun, W., Rumshisky, A., and Uzuner, O. (2013). Evaluating temporal relations in clinical text: 2012 i2b2 Challenge. *Journal of the American Medical Informatics Association : JAMIA 20*, 806–813.

(*159*) Stubbs, A., Kotfila, C., and Uzuner, Ö. (2015). Automated systems for the de-identification of longitudinal clinical narratives: Overview of 2014 i2b2/UTHealth shared task Track 1. *Journal of Biomedical Informatics 58*, S11–S19.

(*160*) Cabral, R. C., Han, S. C., Poon, J., and Nenadic, G. (2024). MM-EMOG: Multi-Label Emotion Graph Representation for Mental Health Classification on Social Media. *Robotics 13*, Number: 3 Publisher: Multidisciplinary Digital Publishing Institute, 53.

(*161*) Chen, S., Hou, Y., Cui, Y., Che, W., Liu, T., and Yu, X. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics: Online, 2020, pp 7870–7881.

(*162*) Gururangan, S., Swayamdipta, S., Levy, O., Schwartz, R., Bowman, S., and Smith, N. A. In *Proceedings of the 2018 Conference of the North Amer-*

*ican Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, Association for Computational Linguistics: New Orleans, Louisiana, 2018, pp 107–112.

(*163*) Niven, T., and Kao, H.-Y. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics: Florence, Italy, 2019, pp 4658–4664.

(*164*) Hofer, M., Kormilitzin, A., Goldberg, P., and Nevado-Holgado, A. (2018). Few-shot learning for named entity recognition in medical text. *arXiv preprint arXiv:1811.05468*.

(*165*) Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models, arXiv:2201.11903 [cs], 2023.

(*166*) Shinn, N., Labash, B., and Gopinath, A. Reflexion: an autonomous agent with dynamic memory and self-reflection, arXiv:2303.11366 [cs], 2023.

(*167*) Hambardzumyan, K., Khachatrian, H., and May, J. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Association for Computational Linguistics: Online, 2021, pp 4921–4933.

(*168*) Ding, N., Hu, S., Zhao, W., Chen, Y., Liu, Z., Zheng, H.-T., and Sun, M. (2021). OpenPrompt: An Open-source Framework for Prompt-learning.

(*169*) De Vries, W., Van Cranenburgh, A., and Nissim, M. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, Association for Computational Linguistics: Online, 2020, pp 4339–4350.

(*170*) Liu, H., Tam, D., Muqeeth, M., Mohta, J., Huang, T., Bansal, M., and Raffel, C. Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning, en, arXiv:2205.05638 [cs], 2022.

(*171*) Kweon, S., Kim, J., Kim, J., Im, S., Cho, E., Bae, S., Oh, J., Lee, G., Moon, J. H., You, S. C., Baek, S., Han, C. H., Jung, Y. B., Jo, Y., and Choi, E. Publicly Shareable Clinical Large Language Model Built on Synthetic Clinical Notes, arXiv:2309.00237 [cs], 2023.

(*172*) Ding, N. et al. (2023). Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence 5*, Number: 3 Publisher: Nature Publishing Group, 220–235.

(*173*) Cui, Y., Han, L., and Nenadic, G. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, Association for Computational Linguistics: Toronto, Canada, 2023, pp 160–183.

(*174*) Gema, A. P., Daines, L., Minervini, P., and Alex, B. Parameter-Efficient Fine-Tuning of LLaMA for the Clinical Domain, arXiv:2307.03042 [cs], 2023.

(*175*) i2b2 dataset, d. i2b2: Informatics for Integrating Biology & the Bedside, tech. rep.

(*176*) Meehan, A. J., Lewis, S. J., Fazel, S., Fusar-Poli, P., Steyerberg, E. W., Stahl, D., and Danese, A. (2022). Clinical prediction models in psychiatry: a systematic review of two decades of progress and challenges. *Molecular Psychiatry 27*, Number: 6 Publisher: Nature Publishing Group, 2700–2708.

(*177*) Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., and Kang, J. (2020). BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics 36*, Publisher: Oxford University Press, 1234–1240.

(*178*) Yuan, Z., Tan, C., and Huang, S. Code Synonyms Do Matter: Multiple Synonyms Matching Network for Automatic ICD Coding, 2022.

(*179*) Wang, S., McDermott, M. B. A., Chauhan, G., Ghassemi, M., Hughes, M. C., and Naumann, T. In *Proceedings of the ACM Conference on Health, Inference, and Learning*, Association for Computing Machinery: Toronto, Ontario, Canada, 2020, pp 222–235.

(*180*) Boag, W., Doss, D., Naumann, T., and Szolovits, P. (2018). What's in a note? unpacking predictive value in clinical note representations. *AMIA Summits on Translational Science Proceedings 2018*, 26.

(*181*) Ramshaw, L. A., and Marcus, M. P. Text Chunking using Transformation-Based Learning, arXiv:cmp-lg/9505040, 1995.

(*182*) Loshchilov, I., and Hutter, F. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, OpenReview.net: 2019.

(*183*) Ma, R., Zhou, X., Gui, T., Tan, Y., Li, L., Zhang, Q., and Huang, X. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics: Seattle, United States, 2022, pp 5721–5732.

(*184*) Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language Models are Unsupervised Multitask Learners.

(*185*) Zhang, S. et al. OPT: Open Pre-trained Transformer Language Models, arXiv:2205.01068 [cs], 2022.

(*186*) TinyLlama/TinyLlama-1.1B-intermediate-step-1431k-3T · Hugging Face, 2023.

(*187*) Chen, Y., Zheng, Y., and Yang, Z. Prompt-Based Metric Learning for Few-Shot NER, arXiv:2211.04337 [cs], 2022.

(*188*) He, K., Mao, R., Huang, Y., Gong, T., Li, C., and Cambria, E. (2023). Template-Free Prompting for Few-Shot Named Entity Recognition via Semantic-Enhanced Contrastive Learning. *IEEE Transactions on Neural Networks and Learning Systems*, Conference Name: IEEE Transactions on Neural Networks and Learning Systems, 1–13.

(*189*) Ashok, D., and Lipton, Z. C. PromptNER: Prompting For Named Entity Recognition, en, arXiv:2305.15444 [cs], 2023.

(*190*) Liu, X., Zheng, Y., Du, Z., Ding, M., Qian, Y., Yang, Z., and Tang, J. GPT Understands, Too, arXiv:2103.10385 [cs], 2023.

(*191*) Harris, C. R. et al. (2020). Array programming with NumPy. *Nature 585*, Publisher: Nature Publishing Group, 357–362.

(*192*) Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., and Liu, Q. TinyBERT: Distilling BERT for Natural Language Understanding, arXiv:1909.10351 [cs], 2020.

(*193*) Sun, Z., Yu, H., Song, X., Liu, R., Yang, Y., and Zhou, D. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics: Online, 2020, pp 2158–2170.

(*194*) Sanh, V., Debut, L., Chaumond, J., and Wolf, T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, arXiv:1910.01108 [cs], 2020.

(*195*) Rohanian, O., Nouriborji, M., Kouchaki, S., and Clifton, D. A. (2023). On the effectiveness of compact biomedical transformers. *Bioinformatics 39*, btad103.

(*196*) Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. Optuna: A Next-generation Hyperparameter Optimization Framework, arXiv:1907.10902 [cs, stat], 2019.

(*197*) Chew-Graham, C., Slade, M., Montana, C., Stewart, M., and Gask, L. (2007). A qualitative study of referral to community mental health teams in the UK: exploring the rhetoric and the reality. *BMC health services research 7*, Publisher: BMC Health Serv Res, DOI: 10.1186/1472-6963-7-117.

(*198*) Digital, N. Mental Health Services Monthly Statistics Dashboard, tech. rep., 2024.

(*199*) Hidden waits force more than three quarters of mental health patients to seek help from emergency services, en.

(*200*) Explaining decisions made with AI, en, Publisher: ICO, 2024.

(*201*) Wu, X., Zhao, Y., Yang, Y., Liu, Z., and Clifton, D. A. A Comparison of Representation Learning Methods for Medical Concepts in MIMIC-IV, en, Pages: 2022.08.21.22278835, 2022.

(*202*) Joyce, D. W., Kormilitzin, A., Hamer-Hunt, J., James, A., Nevado-Holgado, A., and Cipriani, A. CHRONOSIG: Digital Triage for Secondary Mental Healthcare using Natural Language Processing - rationale and protocol, en, preprint, Health Informatics, 2021.

(*203*) Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The Long-Document Transformer, arXiv:2004.05150 [cs], 2020.

(*204*) Vu, T., Nguyen, D. Q., and Nguyen, A. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*; arXiv:2007.06351 [cs], 2020, pp 3335–3341.

(*205*) Edin, J., Junge, A., Havtorn, J. D., Borgholt, L., Maistro, M., Ruotsalo, T., and Maaløe, L. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Association for Computing Machinery: New York, NY, USA, 2023, pp 2572–2582.

(*206*) Mullenbach, J., Wiegreffe, S., Duke, J., Sun, J., and Eisenstein, J. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Association for Computational Linguistics: New Orleans, Louisiana, 2018, pp 1101–1111.

(*207*) Joyce, D. W., Kormilitzin, A., Smith, K. A., and Cipriani, A. (2023). Explainable artificial intelligence for mental health through transparency and interpretability for understandability. *npj Digital Medicine 6*, Number: 1 Publisher: Nature Publishing Group, 1–7.

(*208*) Maaten, L. v. d., and Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research 9*, 2579–2605.

(*209*) Peng, B. et al. RWKV: Reinventing RNNs for the Transformer Era, arXiv:2305.13048 [cs], 2023.

(*210*) Gu, A., and Dao, T. Mamba: Linear-Time Sequence Modeling with Selective State Spaces, arXiv:2312.00752 [cs], 2023.

(*211*) Van Veen, D. et al. (2024). Adapted Large Language Models Can Outperform Medical Experts in Clinical Text Summarization. *Nature Medicine 30*, arXiv:2309.07430 [cs], 1134–1142.

(*212*) Blease, C., and Torous, J. (2023). ChatGPT and mental healthcare: balancing benefits with risks of harms. *BMJ Mental Health 26*, e300884.

(*213*) Wang, C., Liu, S., Yang, H., Guo, J., Wu, Y., and Liu, J. (2023). Ethical Considerations of Using ChatGPT in Health Care. *Journal of Medical Internet Research 25*, Company: Journal of Medical Internet Research

Distributor: Journal of Medical Internet Research Institution: Journal of Medical Internet Research Label: Journal of Medical Internet Research Publisher: JMIR Publications Inc., Toronto, Canada, e48009.

(*214*) Carlini, N., Ippolito, D., Jagielski, M., Lee, K., Tramer, F., and Zhang, C. Quantifying Memorization Across Neural Language Models, arXiv:2202.07646 [cs], 2023.

(*215*) Fry, A., Littlejohns, T. J., Sudlow, C., Doherty, N., Adamska, L., Sprosen, T., Collins, R., and Allen, N. E. (2017). Comparison of Sociodemographic and Health-Related Characteristics of UK Biobank Participants With Those of the General Population. *American Journal of Epidemiology 186*, 1026–1034.

(*216*) Norori, N., Hu, Q., Aellen, F. M., Faraci, F. D., and Tzovara, A. (2021). Addressing bias in big data and AI for health care: A call for open science. *Patterns (New York, N.Y.) 2*, 100347.

(*217*) Gross, N. (2023). What ChatGPT Tells Us about Gender: A Cautionary Tale about Performativity and Gender Biases in AI. *Social Sciences 12*, Number: 8 Publisher: Multidisciplinary Digital Publishing Institute, 435.

(*218*) Lacoste, A., Luccioni, A., Schmidt, V., and Dandres, T. (2019). Quantifying the Carbon Emissions of Machine Learning. *arXiv preprint arXiv:1910.09700*.

Appendix A

# Chapter 3

## A.1 NLP evaluation metric definitions

In my experimental chapters, I focused on several NLP task types and evaluation metrics. Here I provide definitions of the common metrics that will appear throughout. Note, that the metrics below refer to binary classification problems where there are only 2 possible outcomes.

**Accuracy** is the ratio of correctly classified instances to the total number of instances. It is defined as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{A.1}$$

where TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, respectively.

**Precision** is the ratio of correctly classified positive instances to the total number of instances classified as positive. It is defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{A.2}$$

**Recall** (also known as Sensitivity or True Positive Rate) is the ratio of correctly classified positive instances to the total number of actual positive instances. It is defined as:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{A.3}$$

**F1 Score** is the harmonic mean of precision and recall, and it provides a balanced measure of both metrics. It is defined as:

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{A.4}$$

**AUC ROC** The Area Under the Receiver Operating Characteristic curve (AUC-ROC), is a performance metric that measures the trade-off between true positive rate (TPR) and false positive rate (FPR) at various classification thresholds. It is defined as:

$$\text{AUC ROC} = \int_0^1 \text{TPR}(\text{FPR})\, \text{dFPR} \tag{A.5}$$

where the TPR and FPR are computed as:

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{A.6}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{A.7}$$

The following can be seen as general guidance on the quality of a classification model based on the AUC metric.

- AUC = 1: Perfect classifier
- AUC > 0.9: Excellent classifier
- AUC = 0.8 - 0.9: Good classifier
- AUC = 0.7 - 0.8: Fair classifier
- AUC = 0.6 - 0.7: Poor classifier
- AUC = 0.5: Random classifier
- AUC < 0.5: Classifier performs worse than random guessing

The choice of metric depends on the specific problem, the relative risk of different types of errors, and the desired trade-off between precision and recall.

## A.2 Extended Results

**PSIR** The evaluation results for the PSIR severity classification task are presented in Table A.32 and for the incident type classification task in Table A.33.

| Model | LLM | Accuracy | F1 | AUC | Precision | Recall |
|---|---|---|---|---|---|---|
| RoBERTa-PSIR | Frozen | 0.704 | 0.687 | 0.787 | 0.679 | 0.704 |
| RoBERTa-base | Frozen | 0.644 | 0.643 | 0.718 | 0.643 | 0.644 |
| **RoBERTa-PSIR-DeCLUTR** | Frozen | **0.786** | **0.765** | **0.869** | **0.752** | **0.786** |
| RoBERTa-PSIR-note | Frozen | 0.699 | 0.678 | 0.773 | 0.669 | 0.699 |
| RoBERTa-PSIR | Finetuned | 0.866 | 0.835 | 0.942 | 0.816 | 0.866 |
| RoBERTa-base | Finetuned | 0.847 | 0.636 | 0.926 | 0.640 | 0.646 |
| **RoBERTa-PSIR-DeCLUTR** | Finetuned | **0.870** | **0.850** | **0.944** | **0.836** | **0.870** |
| RoBERTa-PSIR-note | Finetuned | 0.863 | 0.833 | 0.936 | 0.814 | 0.863 |

**Table A.32** Evaluation metrics for the PD09: severity classification pseudo-task after 1 epoch for various models in both frozen and full fine-tuned settings.

| Model | LLM | Accuracy | F1 | AUC | Precision | Recall |
|---|---|---|---|---|---|---|
| RoBERTa-PSIR | Frozen | 0.498 | 0.474 | 0.851 | 0.494 | 0.499 |
| RoBERTa-base | Frozen | 0.197 | 0.163 | 0.794 | 0.314 | 0.197 |
| **RoBERTa-PSIR-DeCLUTR** | Frozen | **0.580** | **0.549** | **0.900** | **0.559** | **0.580** |
| RoBERTa-PSIR-note | Frozen | 0.421 | 0.354 | 0.847 | 0.415 | 0.421 |
| RoBERTa-PSIR | Finetuned | 0.665 | 0.652 | 0.935 | 0.655 | 0.665 |
| RoBERTa-base | Finetuned | 0.646 | 0.636 | 0.926 | 0.640 | 0.646 |
| **RoBERTa-PSIR-DeCLUTR** | Finetuned | **0.670** | **0.659** | **0.935** | **0.667** | **0.670** |
| RoBERTa-PSIR-note | Finetuned | 0.660 | 0.647 | 0.933 | 0.652 | 0.660 |

**Table A.33** Evaluation metrics for the IN05: incident category classification pseudo-task after 1 epoch for various models in both frozen and full fine-tuned settings.

**MIMIC-III** Evaluation of the different LLMs for the MIMIC-III note category task are presented in Table A.34.

Evaluation of the different LLMs for the MIMIC-III ICD-9 traige task are presented in Table A.35.

| Model | LLM | Accuracy | F1 | AUC | Precision | Recall |
|---|---|---|---|---|---|---|
| RoBERTa-mimic | Frozen | 0.448 | 0.353 | **0.979** | 0.434 | 0.448 |
| RoBERTa-base | Frozen | 0.433 | 0.347 | 0.949 | 0.406 | 0.433 |
| **RoBERTa-mimic-DeCLUTR** | Frozen | **0.778** | **0.770** | 0.962 | **0.791** | **0.778** |
| RoBERTa-mimic | Finetuned | 0.978 | 0.978 | 0.999 | 0.979 | 0.978 |
| RoBERTa-base | Finetuned | 0.979 | 0.979 | 0.999 | 0.980 | 0.979 |
| **RoBERTa-mimic-DeCLUTR** | Finetuned | **0.985** | **0.985** | 0.999 | **0.986** | **0.985** |

**Table A.34** Classification metrics for Frozen LLM models after 1 epoch for the MIMIC-III note category task. *Roberta-mimic-note models were pre-trained on the note category labels

| Model | LLM | Accuracy | F1 | AUC | Precision | Recall |
|---|---|---|---|---|---|---|
| RoBERTa-mimic | Frozen | 0.315 | 0.188 | 0.871 | 0.435 | 0.303 |
| RoBERTa-base | Frozen | 0.702 | 0.264 | 0.778 | 0.381 | 0.293 |
| **RoBERTa-mimic-DeCLUTR** | Frozen | **0.776** | **0.545** | **0.893** | **0.561** | **0.542** |
| RoBERTa-mimic-note | Frozen | 0.591 | 0.409 | 0.834 | 0.439 | 0.473 |
| RoBERTa-mimic | Finetuned | 0.912 | 0.824 | 0.990 | 0.797 | 0.884 |
| RoBERTa-base | Finetuned | 0.909 | 0.827 | 0.984 | 0.808 | 0.855 |
| **RoBERTa-mimic-DeCLUTR** | Finetuned | **0.917** | **0.831** | **0.990** | **0.794** | **0.890** |
| RoBERTa-mimic-note | Finetuned | 0.906 | 0.819 | 0.987 | 0.788 | 0.867 |

**Table A.35** Classification metrics for LLM models after 1 epoch of fine-tuning for the MIMIC-III ICD-9 triage task.

**OHFT** Classification results for the OHFT note category task are presented in Table A.36.

| Model | LLM | Accuracy | F1 | AUC | Precision | Recall |
|---|---|---|---|---|---|---|
| RoBERTa-base | Frozen | 0.212 | 0.171 | 0.624 | 0.156 | 0.212 |
| RoBERTa-OHFT | Frozen | 0.224 | 0.163 | 0.697 | 0.162 | 0.224 |
| **RoBERTA-OHFT-DeCLUTR** | Frozen | **0.292** | **0.255** | **0.709** | **0.272** | **0.292** |
| RoBERTa-OHFT-note* | Frozen | 0.565 | 0.547 | 0.886 | 0.599 | 0.565 |
| RoBERTa-base | Finetuned | 0.380 | 0.348 | 0.800 | 0.453 | 0.380 |
| **RoBERTa-OHFT** | Finetuned | **0.455** | **0.431** | **0.852** | **0.463** | **0.455** |
| RoBERTA-OHFT-DeCLUTR | Finetuned | 0.404 | 0.390 | 0.821 | 0.434 | 0.404 |
| RoBERTa-OHFT-note* | Finetuned | 0.568 | 0.571 | 0.899 | 0.607 | 0.568 |

**Table A.36** Classification metrics for LLM models after 1 epoch of fine-tuning for the OHFT note category task. *RoBERTa-OHFT-note has been pre-trained on the note category labels

Results for the OHFT Accepted Triage Team task are presented in Table A.37

| Model | LLM | Accuracy | F1 | AUC | Precision | Recall |
|---|---|---|---|---|---|---|
| RoBERTa-base | Frozen | 0.424 | 0.390 | 0.752 | 0.415 | 0.424 |
| RoBERTa-OHFT | Frozen | 0.495 | 0.469 | 0.830 | 0.572 | 0.495 |
| RoBERTa-OHFT-note | Frozen | 0.414 | 0.391 | 0.767 | 0.466 | 0.414 |
| **RoBERTa-OHFT-DeCLUTR** | Frozen | **0.557** | **0.539** | **0.831** | **0.565** | **0.557** |
| RoBERTa-base | Finetuned | 0.677 | 0.665 | 0.918 | 0.692 | 0.677 |
| **RoBERTa-OHFT** | Finetuned | **0.752** | **0.753** | **0.935** | **0.769** | **0.752** |
| RoBERTa-OHFT-DeCLUTR | Finetuned | 0.738 | 0.738 | 0.934 | 0.750 | 0.738 |
| RoBERTa-OHFT-note | Finetuned | 0.744 | 0.743 | 0.933 | 0.745 | 0.744 |

**Table A.37** Classification metrics for LLM models after 1 epoch of fine-tuning for OHFT Accepted triage task

### A.2.1 Extended few-shot learning results

The evaluation performance represented by F1 Macro for all datasets and tasks presented in chapter 3 are provided below in Tables A.38 and A.39.

## A.3 DeCLUTR extended results

The DeCLUTR sampling algorithm used enforces a minimum length of documents dependent on the span length, and the number of anchor spans to derive. The effect of the minimum length on the number of suitable samples for each dataset is provided in Table A.40.

More granular classification results for the incident report severity and incident category pseudo-tasks, varying numbers of epochs of pre-training with the DeCLUTR approach (i.e. both the MLM and contrastive loss objective), in Table A.41 and Table A.42.

| Domain pre-training | Sample size / Task | 16 | 32 | 64 | 128 | 200 |
|---|---|---|---|---|---|---|
| DeCLUTR | Mimic NC | 0.269 | 0.287 | 0.495 | 0.618 | 0.770 |
| | Mimic Triage | 0.146 | 0.095 | 0.315 | 0.431 | 0.683 |
| | OHFT NC | 0.120 | 0.097 | 0.194 | 0.248 | 0.255 |
| | OHFT Triage | 0.190 | 0.238 | 0.265 | 0.356 | 0.453 |
| | PS Severity | 0.435 | 0.519 | 0.361 | 0.506 | 0.494 |
| | PS Type | 0.024 | 0.031 | 0.020 | 0.065 | 0.119 |
| MLM | Mimic NC | 0.075 | 0.184 | 0.029 | 0.167 | 0.353 |
| | Mimic Triage | 0.098 | 0.212 | 0.349 | 0.196 | 0.510 |
| | OHFT NC | NaN | 0.021 | 0.081 | 0.108 | 0.163 |
| | OHFT Triage | 0.069 | 0.134 | 0.086 | 0.168 | 0.213 |
| | PS Severity | 0.430 | 0.271 | 0.440 | 0.196 | 0.431 |
| | PS Type | 0.015 | 0.011 | 0.031 | 0.013 | 0.038 |
| None | Mimic NC | 0.028 | 0.098 | 0.082 | 0.113 | 0.347 |
| | Mimic Triage | 0.028 | 0.130 | 0.207 | 0.013 | 0.492 |
| | OHFT NC | NaN | 0.028 | 0.015 | 0.042 | 0.171 |
| | OHFT Triage | 0.071 | 0.109 | 0.095 | 0.186 | 0.079 |
| | PS Severity | 0.466 | 0.430 | 0.196 | 0.430 | 0.196 |
| | PS Type | 0.022 | 0.011 | 0.011 | 0.011 | 0.011 |
| Note contrastive | Mimic Triage | 0.025 | 0.038 | 0.091 | 0.190 | 0.426 |
| | OHFT Triage | 0.106 | 0.212 | 0.173 | 0.349 | 0.413 |
| | PS Severity | 0.237 | 0.467 | 0.437 | 0.454 | 0.196 |
| | PS Type | 0.027 | 0.011 | 0.027 | 0.015 | 0.015 |

**Table A.38** F1 macro score on all tasks after 1 epoch of training with different numbers of samples per class. Base LLMs were frozen and only the classification head received updates.

| Domain pre-training | Sample size / Task | 16 | 32 | 64 | 128 | 200 |
|---|---|---|---|---|---|---|
| DeCLUTR | Mimic NC | 0.734 | 0.921 | 0.967 | 0.981 | 0.985 |
| | Mimic Triage | 0.294 | 0.493 | 0.727 | NaN | 0.831 |
| | OHFT NC | 0.237 | 0.247 | 0.307 | 0.328 | 0.390 |
| | OHFT Triage | 0.218 | 0.380 | 0.493 | 0.504 | 0.605 |
| | PS Severity | 0.204 | 0.450 | 0.464 | 0.391 | 0.401 |
| | PS Type | 0.051 | 0.037 | 0.099 | 0.399 | 0.452 |
| MLM | Mimic NC | 0.531 | 0.892 | 0.968 | 0.981 | 0.978 |
| | Mimic Triage | 0.282 | 0.367 | 0.596 | 0.802 | 0.824 |
| | OHFT NC | 0.223 | 0.245 | 0.367 | 0.388 | 0.431 |
| | OHFT Triage | 0.221 | 0.236 | 0.494 | 0.669 | 0.706 |
| | PS Severity | 0.451 | 0.435 | 0.404 | 0.442 | 0.197 |
| | PS Type | 0.022 | 0.012 | 0.017 | 0.443 | 0.557 |
| None | Mimic NC | 0.296 | 0.831 | 0.878 | 0.921 | 0.979 |
| | Mimic Triage | 0.050 | 0.023 | 0.748 | 0.732 | 0.827 |
| | OHFT NC | 0.121 | 0.165 | 0.238 | 0.316 | 0.348 |
| | OHFT Triage | 0.069 | 0.115 | 0.331 | 0.466 | 0.490 |
| | PS Severity | 0.196 | 0.223 | 0.196 | 0.196 | 0.412 |
| | PS Type | 0.011 | 0.012 | 0.011 | 0.057 | 0.397 |
| Note contrastive | Mimic Triage | 0.134 | 0.258 | 0.537 | 0.759 | 0.827 |
| | OHFT Triage | 0.232 | 0.210 | 0.409 | 0.539 | 0.618 |
| | PS Severity | 0.430 | 0.196 | 0.430 | 0.340 | 0.196 |
| | PS Type | 0.013 | 0.012 | 0.019 | 0.129 | 0.519 |

**Table A.39** F1 macro score on all tasks after 1 epoch of training with different numbers of samples per class. Models were fully fine-tuned.

| | Proportion of 250k LM training dataset | | |
|---|---|---|---|
| Min. document length | OHFT | PSIR | MIMIC-III |
| 16 | 0.97 | 0.88 | 0.99 |
| 32 | 0.89 | 0.69 | 0.95 |
| 64 | 0.72 | 0.41 | 0.97 |
| 128 | 0.46 | 0.15 | 0.92 |
| 256 | 0.21 | 0.04 | 0.81 |
| 512 | 0.07 | 0.02 | 0.57 |
| 1024 | 0.02 | 0.0 | 0.35 |
| 2048 | 0.002 | 0.0 | 0.11 |

**Table A.40** Sample distributions for different DeCLUTR sampling minimum document lengths for each of the datasets: OHFT, PSIR, and MIMIC-III. The proportion refers to the amount of samples that meet each of the minimum document length thresholds.

| Model | | | | Performance after 1 epoch | | | | |
|---|---|---|---|---|---|---|---|---|
| Name | PLM | Pre. Epochs | Trainable params.(m) | Accuracy | ROC AUC | $F\_1$ | Prec. | Recall |
| DeCLUTR-base-incident | Frozen | 2 | 0.5 | 0.758 | 0.842 | 0.725 | 0.713 | 0.758 |
| | | 3 | 0.5 | 0.763 | 0.847 | 0.719 | 0.709 | 0.763 |
| | | 5 | 0.5 | 0.763 | 0.848 | 0.75 | 0.741 | 0.763 |
| | | 10 | 0.5 | **0.771** | 0.855 | 0.752 | 0.74 | **0.771** |
| | | 25 | 0.5 | 0.769 | **0.859** | **0.756** | **0.746** | 0.769 |
| | | 50 | 0.5 | 0.755 | 0.840 | 0.717 | 0.705 | 0.755 |
| RoBERTa-incident-DeCLUTR | Frozen | 2 | 0.5 | 0.775 | 0.858 | 0.750 | 0.736 | 0.775 |
| | | 3 | 0.5 | **0.786** | 0.869 | 0.765 | 0.752 | **0.786** |
| | | 5 | 0.5 | 0.780 | **0.871** | **0.769** | **0.760** | 0.780 |
| | | 10 | 0.5 | 0.782 | 0.865 | 0.751 | 0.731 | 0.762 |
| | | 25 | 0.5 | 0.776 | 0.866 | 0.752 | 0.740 | 0.776 |
| | | 50 | 0.5 | 0.765 | 0.851 | 0.734 | 0.720 | 0.765 |
| DeCLUTR-base-incident | Fine-tuned | 2 | 125 | 0.859 | 0.936 | 0.833 | 0.816 | 0.859 |
| | | 3 | 125 | **0.861** | **0.937** | 0.823 | 0.803 | **0.861** |
| | | 5 | 125 | 0.858 | 0.936 | 0.841 | 0.828 | 0.858 |
| | | 10 | 125 | 0.859 | 0.934 | 0.827 | 0.808 | 0.859 |
| | | 25 | 125 | 0.854 | 0.934 | **0.846** | **0.835** | 0.854 |
| | | 50 | 125 | 0.851 | 0.931 | 0.792 | 0.767 | 0.851 |
| RoBERTa-incident-DeCLUTR | Fine-tuned | 2 | 125 | **0.870** | **0.944** | **0.850** | **0.836** | **0.870** |
| | | 3 | 125 | 0.868 | 0.942 | 0.838 | 0.819 | 0.868 |
| | | 5 | 125 | 0.867 | 0.941 | 0.831 | 0.811 | 0.867 |
| | | 10 | 125 | 0.858 | 0.938 | 0.825 | 0.807 | 0.858 |
| | | 25 | 125 | 0.863 | 0.939 | 0.833 | 0.815 | 0.863 |
| | | 50 | 125 | 0.857 | 0.934 | 0.831 | 0.815 | 0.857 |

**Table A.41** Evaluation metrics for the PD09: severity classification pseudo-task for various models in both frozen and full fine-tuned settings of DeCLUTR models at various different total epochs of further pre-training for performance after one epoch of training on the task.

| Model | | | | Performance after 1 epoch | | | | |
|---|---|---|---|---|---|---|---|---|
| Name | PLM | Pre. Epochs | Trainable params.(m) | Accuracy | ROC AUC | $F_1$ | Prec. | Recall |
| DeCLUTR-base-incident | Frozen | 2 | 0.5 | 0.548 | 0.878 | 0.511 | 0.525 | 0.548 |
| | | 3 | 0.5 | 0.542 | 0.883 | 0.503 | 0.509 | 0.542 |
| | | 5 | 0.5 | 0.559 | 0.886 | 0.527 | 0.531 | 0.559 |
| | | 10 | 0.5 | 0.562 | 0.888 | 0.532 | 0.534 | 0.562 |
| | | 25 | 0.5 | **0.577** | **0.893** | **0.544** | **0.557** | **0.577** |
| | | 50 | 0.5 | 0.550 | 0.877 | 0.518 | 0.538 | 0.550 |
| RoBERTa-incident-DeCLUTR | Frozen | 2 | 0.5 | 0.555 | 0.883 | 0.521 | 0.527 | 0.555 |
| | | 3 | 0.5 | 0.568 | 0.892 | 0.531 | 0.541 | 0.568 |
| | | 5 | 0.5 | **0.580** | **0.900** | **0.549** | **0.559** | **0.580** |
| | | 10 | 0.5 | 0.578 | 0.894 | 0.544 | 0.551 | 0.578 |
| | | 25 | 0.5 | 0.574 | 0.892 | 0.545 | 0.545 | 0.574 |
| | | 50 | 0.5 | 0.572 | 0.883 | 0.548 | 0.552 | 0.572 |
| DeCLUTR-base-incident | Fine-tuned | 2 | 125 | 0.648 | 0.930 | 0.641 | 0.651 | 0.648 |
| | | 3 | 125 | **0.661** | 0.930 | **0.651** | **0.657** | **0.661** |
| | | 5 | 125 | 0.649 | **0.931** | 0.632 | 0.641 | 0.649 |
| | | 10 | 125 | 0.658 | 0.931 | 0.647 | 0.647 | 0.658 |
| | | 25 | 125 | 0.651 | 0.931 | 0.640 | 0.643 | 0.651 |
| | | 50 | 125 | 0.645 | 0.929 | 0.631 | 0.635 | 0.645 |
| RoBERTa-incident-DeCLUTR | Fine-tuned | 2 | 125 | **0.670** | **0.935** | 0.659 | 0.667 | **0.670** |
| | | 3 | 125 | 0.668 | 0.933 | **0.660** | **0.672** | 0.668 |
| | | 5 | 125 | 0.660 | 0.934 | 0.646 | 0.660 | 0.660 |
| | | 10 | 125 | 0.658 | 0.932 | 0.644 | 0.653 | 0.658 |
| | | 25 | 125 | 0.659 | **0.935** | 0.649 | 0.654 | 0.659 |
| | | 50 | 125 | 0.653 | 0.934 | 0.644 | 0.654 | 0.653 |

**Table A.42** Evaluation metrics for the IN05: incident category classification task for various models in both frozen and full fine-tuned settings of DeCLUTR models at various different total epochs of further pre-training for performance after one epoch of training on the task.

Appendix B

# Chapter 4

## B.1 MIMIC-III task class distributions

The distribution of the classes for each of the MIMIC-III sequence classification tasks utilised in Chapter 4 are presented in Figure 34.

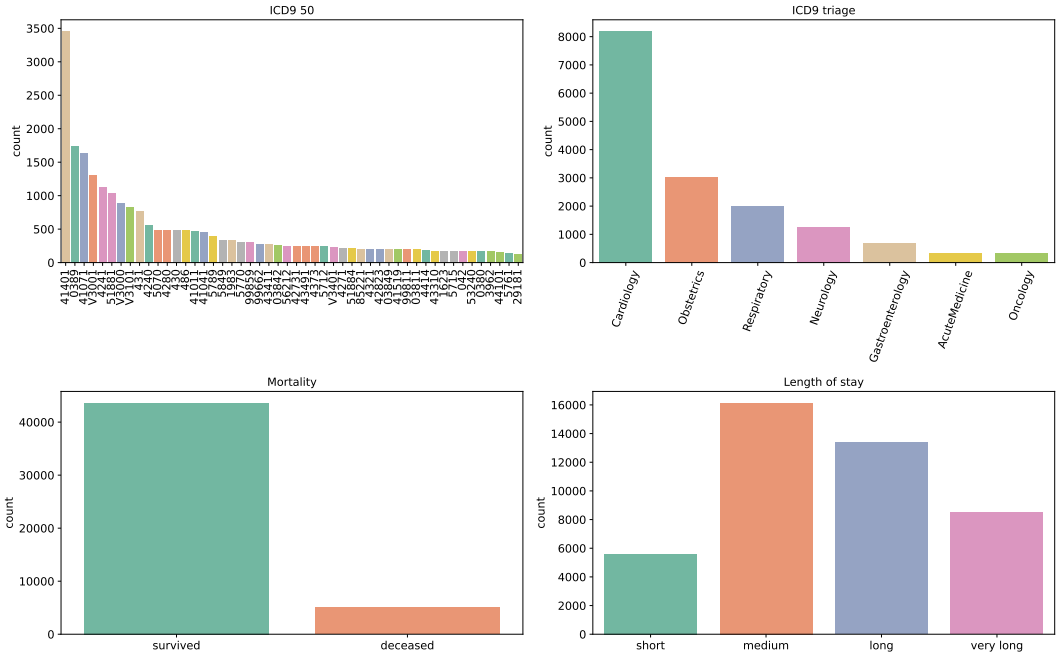


**Figure 34** Distribution of class labels across each of the MIMIC-III sequence classification tasks: ICD9-50, ICD9-Triage, Mortality Prediction, and Length of Stay.

## B.2 I2B2 NER task class distributions

The distribution of the i2b2 NER tasks is provided in Figures 35, 36, and 37.

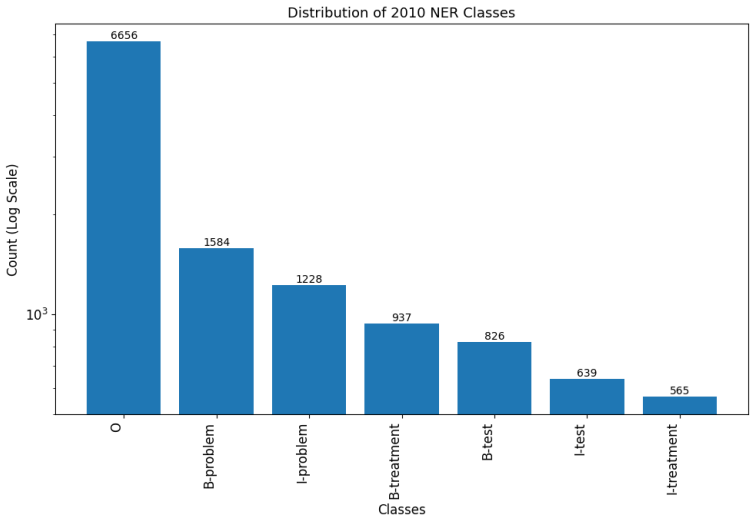


**Figure 35** Distribution of entity class labels for the I2B2 2010 NER task

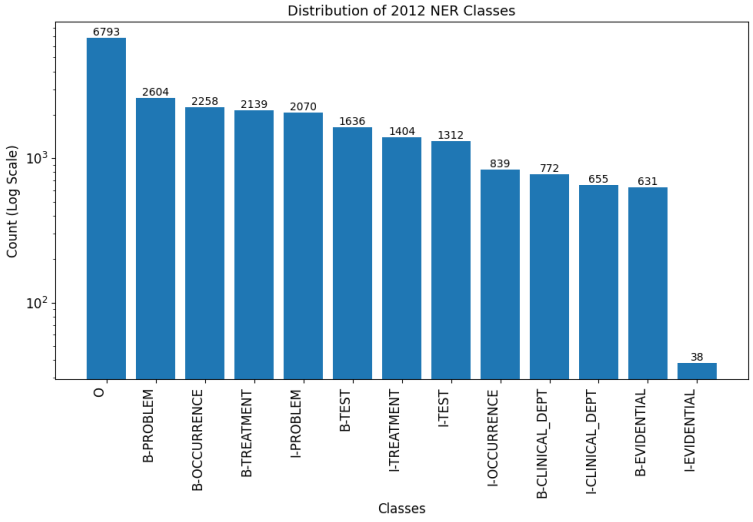


**Figure 36** Distribution of entity class labels for the I2B2 2012 NER task

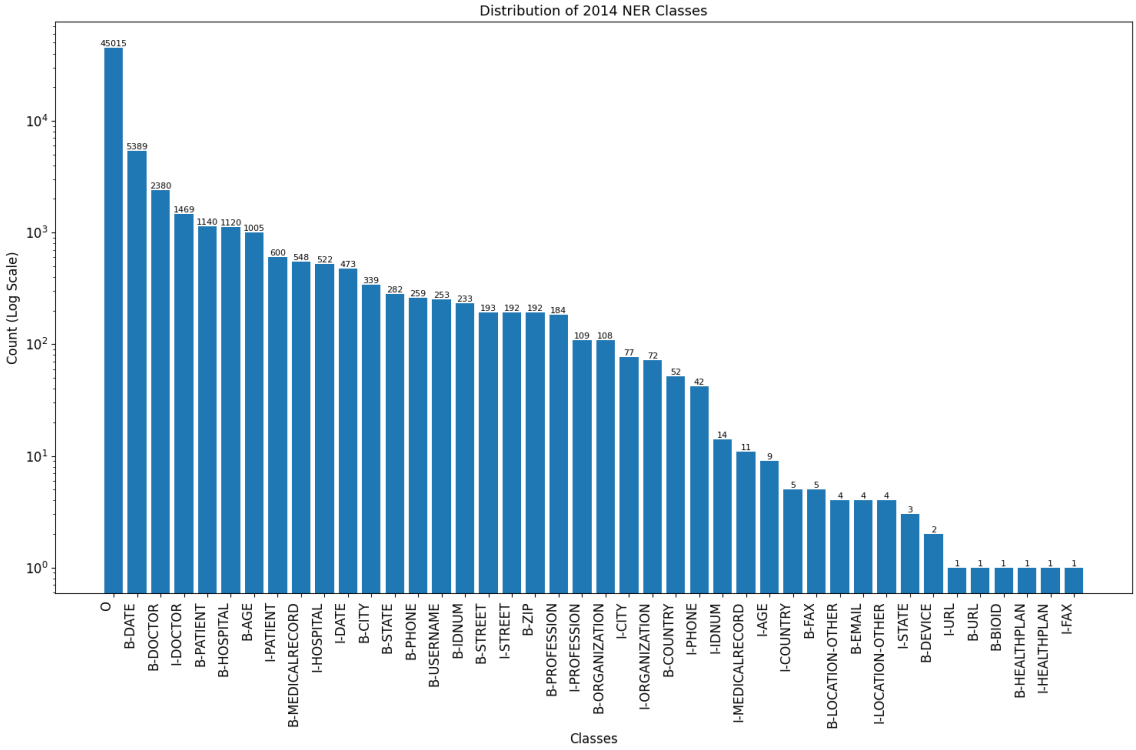


**Figure 37** Distribution of entity class labels for the I2B2 2014 NER task

## B.3 Extended results

The results for all tasks using the domain pre-trained small LLMs are provided below in Table B.43.

## B.4 Prompt learning hyperparameter search

Table B.44 shows the derived optimal hyperparameters for each training paradigm based on the hyperparameter search. The search consisted of 100 training runs using randomly generated hyperparameters from the search space shown in Table

| Model name | PEFT | ICD9-Triage | i2b2-2010-RE | MIMIC-LoS | Mimic-MP |
|---|---|---|---|---|---|
| BERTbase | Full | 0.991 | 0.975 | 0.702 | 0.799 |
| BERTbase | LORA | 0.983 | 0.980 | 0.679 | 0.811 |
| BioBERT | Full | 0.991 | 0.982 | 0.711 | 0.812 |
| BioBERT | LORA | 0.991 | 0.985 | 0.697 | 0.828 |
| BioClinicalBERT | Full | 0.993 | 0.978 | 0.697 | 0.793 |
| BioClinicalBERT | LORA | 0.990 | 0.981 | 0.701 | 0.822 |
| BioDistilBERT | Full | 0.992 | 0.979 | 0.697 | 0.803 |
| BioDistilBERT | LORA | 0.993 | 0.988 | 0.704 | 0.822 |
| BioMobileBERT | Full | 0.992 | 0.980 | 0.697 | 0.809 |
| BioMobileBERT | LORA | 0.987 | 0.982 | 0.670 | 0.792 |
| ClinicalDistilBERT | Full | 0.994 | 0.980 | 0.697 | 0.822 |
| ClinicalDistilBERT | LORA | 0.995 | 0.989 | 0.710 | 0.836 |
| ClinicalMobileBERT | Full | 0.995 | 0.983 | 0.720 | 0.826 |
| ClinicalMobileBERT | LORA | 0.994 | 0.982 | 0.690 | 0.824 |

**(a)** Sequence classification task results

| Model name | PEFT | i2b2-2010-NER | i2b2-2012-NER | i2b2-2014-NER |
|---|---|---|---|---|
| BERTbase | Full | 0.806 | 0.792 | 0.974 |
| BERTbase | LORA | 0.673 | 0.697 | 0.951 |
| BioBERT | Full | 0.822 | 0.823 | 0.969 |
| BioBERT | LORA | 0.713 | 0.757 | 0.935 |
| BioClinicalBERT | Full | 0.846 | 0.820 | 0.960 |
| BioClinicalBERT | LORA | 0.704 | 0.746 | 0.920 |
| BioDistilBERT | Full | 0.809 | 0.794 | 0.965 |
| BioDistilBERT | LORA | 0.704 | 0.726 | 0.939 |
| BioMobileBERT | Full | 0.794 | 0.774 | 0.966 |
| BioMobileBERT | LORA | 0.649 | 0.654 | 0.938 |
| ClinicalDistilBERT | Full | 0.816 | 0.817 | 0.961 |
| ClinicalDistilBERT | LORA | 0.671 | 0.740 | 0.920 |

**(b)** NER task results

**Table B.43** PEFT results for sequence classification and NER tasks dependent on domain pre-training received.

16. Due to relatively limited computational resources, this was only performed for the ICD-9 Triage task and a sub-sample of the training data was used, similar to

that of our few-shot experiments with 128 samples per class.

| hp | Traditional fine-tuning | Prompt learning |
|---|---|---|
| learning rate | 0.0048 | 0.0121 |
| batch size | 8 | 4 |
| gradient accumulation steps | 4 | 3 |
| dropout | 0.382 | 0.1536 |
| optimizer | adamw | adafactor |
| verbalizer learning rate | n/a | 0.007 |

**Table B.44** Optimized hyperparameters for each training paradigm

Appendix C

# Chapter 5

## C.1 Triage Team Referral Bouncing

The most common pathway for patients to be assessed and then treated in specialist secondary care is for another professional (often, a primary care general practitioner/family physician) to send a referral to a local secondary care organisation. Assume Team A receives a referral, but decides that another service – Team B – would be better positioned to assess or treat the patient. Team A will then forward the referral to Team B. On receipt of the referral, Team B might conclude that either i) another service, Team C, is better suited to care for the patient or ii) Team A should have accepted the patient's referral in the first place. This results in cycles of what is termed "referral bouncing" and is an unfortunate result of service pressures and – as described by (*197*) – "capricious" and "opaque" decision-making that in reality reflects arbitrary application of referral criteria in the interests of the team (rather than the patient). Of course, there are legitimate clinical reasons for referral forwarding and bouncing; in some clinical services, a team might function as a single point of access for a geographical region (see Figure 25) in addition to having an assessment/treatment function for the same cohort of patients.

Liaising with clinicians working in the secondary care system from which our data

originated, it was clear that some teams (notably, community mental health teams designated "CMHTs") have both these functions. Therefore, it is not uncommon for a referral to appear in the EHR as a referral to the CMHT and then quickly, as another referral to a different team. Figure 26 showed a descriptive analysis of these initial first- and second-referral patterns in our data set. As a concrete example; CMHTs frequently receive referrals for patients in crisis, for which they (reasonably) refer to CRHTT (crisis resolution and home treatment) teams. CRHTTs are specifically designed to be disorder agnostic to help patients within hours of being referred and to specifically address and manage crisis presentations that are not (in general) the remit of CMHTs or other secondary care teams. This is reflected in the first column of Figure 26. Similarly, by examining the diagonal of the same figure, it can be seen that for some teams (notably, teams working with older adults) there is a high probability that on first being referred, that referral will remain with that team. The asymmetry in referral patterns is equally revealing. For example, a patient referred to a sub-specialty team for Early Intervention in Psychosis (the EIP row in 26) has a probability of 0.47 and 0.43 of being referred to a "general" CMHT and CRHTT respectively. The former may suggest an inappropriate referral (i.e. the patient does not present meet the team's criteria of having a first episode of a psychotic disorder) while the latter reflects that many crisis presentations have features that would (in the absence of crisis) have been suggestive of a psychotic illness that EIP teams are specifically designed to help.

Finally, using LLMs and classification in assisting triage is fundamentally an inductive learning problem using a discriminative model – attempting to learn the probability of a team accepting a referral (from data contained in the EHR) given

the learned patient representations (from free-text, narrative documentation in the EHR). The data contained in the EHR does not explicitly describe the clinical reasoning that leads to an acceptance (or rejection) of a referral in a way that can be interrogated or exploited – so at the point of referral (e.g. within a specific instance – the fundamental unit of input to the triage assistance system – see Figure 28) it is not possible to conclude that a referral was bounced for a clinical reason (e.g. the patient was in crisis and appropriately forwarded to the crisis team), the referral was incorrectly sent to that team (i.e. the receiving CMHT was not the correct team for that patient because of their geographical location) or if the forwarding (bouncing) of a referral results from opaque clinical reasoning/decisions.

### C.1.1 Justifying the Acceptance Heuristic

As noted in Chapter 5, using local knowledge of the NHS and the specific clinical services available in Oxford Health NHS Foundation Trust, we were able to derive a heuristic to yield target labels for which team accepted a referral for any given instance in the training and evaluation data. Recall that in the EHR data, structured referral date fields are generally populated and reliable, but rejection date fields are unreliable with discharge date fields often as a proxy for referral rejection. To evidence the basis for this "14-day rule", I present the distribution of instance durations, based on the provided *referral date* and *discharge date* structured administrative data in Figure 38. It can be seen that there is a large proportion of referrals that are discharged within the same day, and a slight peak around 14 days.

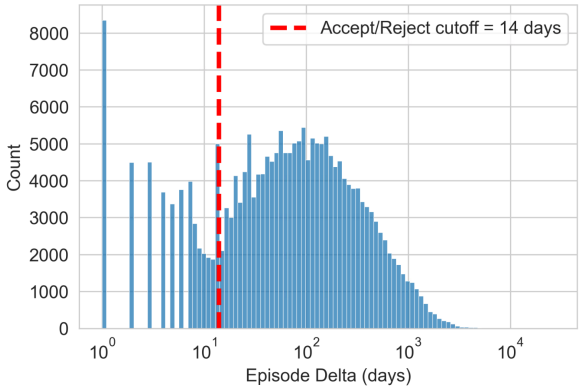


**Figure 38** Histogram of the number of days individual referral instances remain open based on the available referral date, and discharge date structured fields within the OHFT EHR data.

### C.1.2 Referral Instance and Document Statistics

In Figure 39 is the distribution of token numbers per instance as a function of whether the referral instance was deemed to have been accepted or rejected, according to our heuristic. There is no relevant difference between those instances that were accepted versus rejected, with a median of 1463 and 1367 tokens respectively.

### C.1.3 Longformer details

The Longformer model, introduced by Beltagy et al. in 2020(*203*), addresses the limitation of full attention described in section 2.5.3.5 by employing a combination of windowed local attention and global attention. Instead of attending to the entire sequence, each token attends only to a window of nearby tokens. This window size is a hyperparameter that can be adjusted. This local attention reduces the computational complexity to linear in the sequence length. While local attention

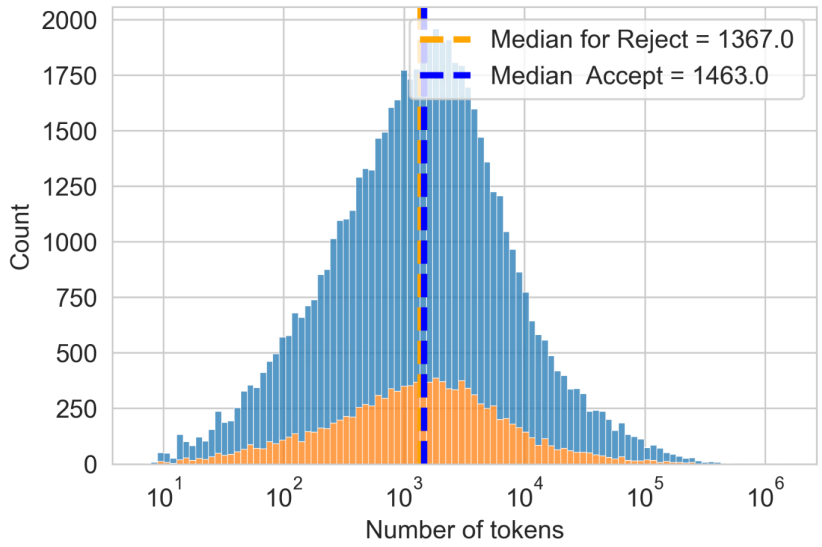


**Figure 39** Histogram of the number of tokens per referral ($log10$ scale) instance based on reject or accept label

captures local context, global attention is needed to incorporate long-range dependencies. The Longformer model allows attending to a subset of tokens globally, in addition to the local window. These globally attended tokens can be specified manually or learned during training.

By combining windowed local attention and global attention, the Longformer model strikes a balance between capturing long-range dependencies and computational efficiency, making it suitable for processing long sequences, such as those encountered in natural language processing tasks like document summarization, question answering, and language modelling. The attention mechanism in the Longformer model can be summarized as follows: for each token, attend to its local window of tokens using local attention, and also attend to a set of globally important tokens using global attention. This adaptation of the attention mechanism enables the Longformer model to handle input sequences much longer than

what was feasible with the standard transformer architecture, while still capturing relevant long-range dependencies.