IMPROVING CLINICAL TEXT CLASSIFICATION USING LARGE LANGUAGE MODELS GUIDED BY SEMANTIC KNOWLEDGE

by

Graham Scott

A thesis submitted in partial fulfillment
of the requirements for the Master of Science
degree in Electrical And Computer Engineering in the
Graduate College of
The University of Iowa

December 2024

Thesis Committee:
Kishlay Jha, Thesis Supervisor
Tyler Bell
Hans Johnson

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor Dr. Kishlay Jha for his extensive help with writing this thesis and navigating through graduate school. I would also like to thank Dr. Samuel Zetumer at University of Iowa Hospitals and Clinics for his collaboration on this project, the creation of his heart failure rating algorithm, and for the vital medical expertise he was able to provide my work. Lastly, I would like to thank my family and friends, without whom I would not have been able to last until the completion of this body of work.

ABSTRACT

Current state of the art models for performing clinical text analysis do not yet represent technologies that can be incorporated into tools for live use by medical professionals and practitioners in hospitals due to the discrepancy between the data used in research and the data created in and used by hospitals. Public datasets utilized by natural language processing (NLP) research groups are heavily processed before use in research both by necessity (removal of sensitive personal information) and to improve the ability of language-processing models to extract information.

This thesis explores the aspects of unprocessed hospital text that add unwanted noise, and using the knowledge gained of the syntax and semantics of these documents, proposes a novel model architecture that incorporates measures for addressing undesirable anti-patterns that are common in hospital patient notes with the final goal of creating a model that can be used directly on hospital medical data without any intermediate human processing.

Traditional machine learning models exhibit little capacity to cope with the intricacies of natural language processing. The introduction of deep learning architectures like recurrent neural networks (RNNs) and transformers have made NLP possible by allowing models to capture both local and global entities in text. Transformers in particular address key challenges through mechanisms like self-attention, enabling models to weigh the importance of different tokens in a sequence without requiring an explicitly ordered dependency. However, the flexibility that allows transformers to handle the complexities of human language also makes the highly sensitive to noise and unwanted patterns in the data they are trained on. We combat this by leveraging the semantic knowledge that we have gained to create software that reduces the intensive manual data curation that would normally be necessary into model hyperparameters that can be tuned to account for the anti-patterns of similar patient document datasets.

PUBLIC ABSTRACT

The models that are currently the best for analyzing medical text can't yet be used in actual hospitals because of the difference between the data used in research and the data created and used by hospitals. Most medical text datasets used by researchers are heavily processed before use for many reasons, and a model would have to work even without said processing in order to be effective in a live hospital setting.

This thesis explores the aspects of unprocessed hospital notes that make models less accurate and slower to train, and using the knowledge gained of the semantics of these documents, proposes a novel model architecture that addresses the most common of those problems to create a model that can be used directly on hospital medical data without any intermediate human processing.

Traditional machine learning models have trouble processing human language. The invention of transformers have made NLP possible by allowing models to understand connections between ideas in text even when they're far apart. However, the flexibility that allows transformers to handle the complexities of human language also makes the highly sensitive to things in the data they are trained on that are undesirable, like typos. We combat this by using the semantic knowledge we have gained about the noise present in raw hospital notes to create software that reduces the intensive manual data curation that would normally be necessary to feed hospital data into large language models into model hyperparameters that can be tuned to account for the anti-patterns of similar patient document datasets.

TABLE OF CONTENTS

LIST OF TABLES viii

LIST OF FIGURES ix

CHAPTER 1: INTRODUCTION 1

Classification in the Domain of Text 2

Binary Classification 3

Cohort Selection: Definition and Motivation 3

Extreme Multilabel Text Classification 4

ICD Codes: Definition and Motivation 4

Expert Knowledge and Testimony 6

Challenges of Medical Data 7

CHAPTER 2: BACKGROUND 8

Related Works in NLP and LLMs for Clinical Text 8

Bag of Words 8

Recurrent Neural Networks 9

Attention Mechanisms 9

Transformers and LLMs 10

LoRAs 11

Previous Works in Medical NLP 12

CorNet 12

MeSHProbeNet 13

AttentionXML 14

General Text Datasets Used 15

AmazonCat 15

EUR-Lex 15
Medical Text Datasets Used 16
PubMed 16
MIMIC-III & MIMIC-IV 17
University of Iowa Hospital and Clinic Data 18
Challenges of working with raw hospital data 18
Review of Previous Research 20
CHAPTER 3: METHODOLOGY 24
Extreme Multi-label Text Classification 24
Problem Formulation 24
Model Architecture 24
Cohort Selection 25
Problem Formulation 25
Fine-Tuning LLMs 27
Hospital Cohort Selection 28
Design of CohortLang 28
Model Design 28
Database Export 29
Characterset Simplification 29
Note type filtering 30
Text Subsegmentation 32
Segment Prompt Formatting 34
Prompt Tokenization 35
Prompt Inference 36
Mixture of Experts Aggregation 36
Threshhold Decision 37

Training 37

CHAPTER 4: EXPERIMENTS 39

CorNet 39

Re-creation 39

Transfer to medical text 39

Hospital Data 40

Baseline results 40

Large Language Models 42

CHAPTER 5: RESULTS 44

XMTC 45

Re-creation of CorNet 45

Transfer to medical text 46

Cohort Selection 48

Baseline results 48

LLM Results 48

Preprocessing Evaluation 51

CHAPTER 6: CONCLUSION 53

Recommendations for Future Research 53

Summary 54

REFERENCES 55

LIST OF TABLES

Table 1. Number of unique characters treated in different ways 29

Table 2. Summary of number of note sections per patient 30

Table 3. Document length statistics 32

Table 4. AttentionXML performance metrics on EUR-Lex text 45

Table 5. XMTC model performance metrics on MIMIC-IV text 47

Table 6. Decision Tree performance metrics on Hospital Data 48

Table 7. Performance of MedLlama3 49

Table 8. Performance metrics of Microsoft Phi 2 after being fine-tuned on pre-processed hospital data 50

Table 9. Performance for each semantic knowledge preprocessing step combination using best-performing model (Phi 2) 51

LIST OF FIGURES

Figure 1. Diagram of the MeSHProbeNet module architecture[43] 13

Figure 2. Diagram of the AttentionXML model architecture[46] 14

Figure 3. Diagram of CohortLang pipeline. 28

Figure 4. Character counts before and after 31

Figure 5. Training and Validation loss of models trained on EUR-Lex 46

Figure 6. Sensitivity and Specificity of Logistic Regression after training on Hospital Data 47

Figure 7. Sensitivity and Specificity of MedLlama3 after training on Hospital Data 49

Figure 8. Sensitivity and Specificity of Microsoft Phi 2 after training on Hospital Data . . . 50

CHAPTER 1: INTRODUCTION

Classification, language processing, and neural networks are all topics common within the domain of machine learning, and any problem setting involving all three introduces unique challenges. These challenges are more significant in the setting of medical text, where outcomes can be life or death and the data is inherently complex. Medical texts are rich with domain-specific language and carefully-structured data that often escapes the already convoluted structure of human language, such as tables. On top of the complications inherent to medical text, there is an additional layer of complexity when working with raw, unprocessed data such as the text medical practitioners create and reference while interacting with patients. Many experiments with natural language processing (NLP) are performed on carefully-curated datasets that are checked by humans for accuracy, structural uniformity, and other factors for use with scientific experiments. **In order to make the jump from research to practical application, exploration must be done into where NLP models fall short when using raw data from actual patients, and how to overcome these shortcomings.** This paper aims to bridge the gap between theoretical advancements and practical application by investigating the efficacy of various neural network models and training techniques when performing common classification tasks in medical settings, and investigate what can be changed in models to facilitate higher performance on non-curated hospital data. We do so by:

- Examining the differences between unprocessed hospital data and curated data from a publicly-available research dataset.
- Creating a system that reduces manual data curation tasks to a series of automated preprocessing steps with hyperparameters that can be tuned to accommodate the differences between different hospitals' datasets.
- Creating a model which incorporates this automated processing system into its training and inference processes to improve overall performance and speed.

To evaluate its effectiveness in improving performance with cohort selection, we perform evaluation with multiple large language models and compare multiple evaluation metrics with and without our semantics-guided processing.

Our goal is to increase the speed and accuracy of medical information retrieval and decision support systems. Such improvements have the potential to increase the pace of medical research and directly improve the care given to patients. We show this potential by examining the problems of binary classification and extreme multi-label text classification in medical settings.

Classification In The Domain of Text

Classification is one of the oldest problems that scientists have applied neural networks to. From the early days of machine learning, the task of categorizing data into distinct classes has been central to developing intelligent systems. In the context of text, classification encompasses a broad range of applications, from spam detection in emails to sentiment analysis in social media posts. Early neural networks, such as simple feed-forward architectures, laid the groundwork for these tasks by learning to identify patterns and make predictions based on labeled examples. However, as text data grew in complexity and volume, more sophisticated models were required. The advent of recurrent neural networks (RNNs) and long short-term memory networks (LSTMs) marked significant progress, allowing for the handling of sequential dependencies and contextual information. More recently, transformer-based models like BERT have revolutionized text classification by leveraging self-attention mechanisms to capture intricate relationships within the data. This evolution underscores the dynamic nature of text classification and highlights the ongoing need for advancements in neural network architectures to address the challenges posed by increasingly complex and diverse textual data.

Classification in particular has unique opportunities when working with text due to the way the labels themselves often have text descriptions. Unlike numerical or categorical labels, text-based labels provide rich, contextual information that can enhance the classification process. This descriptive nature allows for a more nuanced understanding of the categories, facilitating the use of natural language processing techniques to better align model predictions with the semantic meaning of the labels. For instance, in medical text classification, labels such as ”Diabetes Type 2” or ”Hypertension” can be connected with definitions and examples of said conditions included in the training data. By incorporating these textual descriptions into the learning process, neural networks can leverage additional context to improve classification accuracy and relevance, even when working with labels that are not common in the data on which they are trained.

Binary Classification

Binary Classification is the simplest-possible classification task. The model must predict a value of 0 or 1 for each data point. This can be a representation of a model predicting something to be True or False, answering a question ”yes” or ”no”, or identifying whether or not the given data point should be assigned some category that the model has been trained to sort data into. Binary Classification, while simple, is the backbone of all other classification tasks. Tagging an image, for instance, can be modeled as a series of Binary Classification tasks (assigning 0 or 1 to each potential label). Cohort Selection, one of of the clinical tasks we trained models to perform, is also a Binary Classification task.

Cohort Selection: Definition And Motivation

Cohort Selection is a particular type of classification task often required of medical researchers. When conducting a study, it is necessary to select patients whose condition is relevant to whatever the study is being conducted to investigate. This often requires manual searching by researchers examining patient files one by one in order to determine whether each patient should be selected for the cohort of subjects studied.

As a repetitive task with an output determined entirely by the information in the text of a patient’s medical history, cohort selection is a prime opportunity for use of text-based classifications systems. A model developed to aid in the process of cohort selection would not need to out-perform a human on its own in order to make a valid contribution; it need only lighten the load on researchers. A model that merely rules out half of the patients that are certain not to be eligible for cohort selection is already saving potentially dozens of man-hours of work for medical researchers.

As we had the chance to work directly with medical professionals in this experiment, we focused on the domain of medicine in which they already had experience with cohort selection methods research: Heart Failure.

Extreme Multilabel Text Classification

Extreme Multilabel Text Classification is an advanced subset of text classification that addresses the challenge of classifying a text document with any number of a vast set of possible labels. Unlike traditional multilabel classification, where the number of labels is relatively small, XMTC deals with scenarios where the 'label space' is enormous, often comprising thousands or even potentially millions of categories. This complexity necessitates novel algorithms and techniques to efficiently handle the scalability and sparsity inherent in such problems. Key approaches in XMTC include leveraging sparse representations, hierarchical structures implicit in the labels themselves, algorithms that can handle high-frequency and low-frequency labels within the same set, and efficient approximation methods to manage the computational and memory constraints associated with such large label sets.

The development of XMTC has significant implications within various domains, including assigning tags or categories to text documents in databases, product recommendation systems, and patient diagnosis. By improving the ability to handle and predict a large number of labels with high precision, XMTC advances contribute to more effective and nuanced data analysis, ultimately enhancing the capabilities of systems that rely on complex text classification tasks.

ICD Codes: Definition And Motivation

The ICD code system [27], developed by the World Health Organization (WHO), is a tool in healthcare for categorizing diseases and health-related conditions. This code system organizes conditions into a structured hierarchy, facilitating precise coding and consistent documentation across various healthcare settings. The ICD codes are used for a multitude of purposes, including epidemiological research, health statistics, and clinical decision-making.

Containing a variety of codes large enough to effectively differentiate between patient conditions with nothing other than the single label assigned, ICD Codes are an archetypal XMTC task. Each ICD code can be assigned as a label, and patient documents can be assigned any number of labels/codes depending on how many conditions they have suffered from at any point during the period where the hospital recorded information on them.

The classification system undergoes periodic updates to incorporate new medical knowledge and advancements, ensuring its relevance and accuracy in reflecting the evolving landscape of health and disease. This means that any technology built to be incorporated into medical systems utilizing ICD code assignment must not only be configured to correctly understand current ICD codes, but also have the potential to be updated for further updates to the ICD schema. ICD coding problems are not subject to the usual machine learning rule of the current version being the worst to ever exist; If not updated, ICD-based models *will* lose their effectiveness even if competitors are not developed.

In machine learning research, it is common for researchers to use ICD codes as the labels for training classification models, enabling the development of predictive algorithms and diagnostic tools. By aligning medical data with ICD classifications, researchers can enhance the accuracy and interpretability of their models, as well as facilitate cross-study comparisons using the predicted ICD codes as a common ground for contrasting performance.

ICD codes are also often used for billing in hospitals in a bureaucratic pipeline that requires the recruitment and training of specialized workers trained to assign ICD codes to documents in the best case, and requires that doctors *themselves* take time away from treating patients to look up and assign ICD codes to their documents in the worst case. This means that any developed model capable of working with ICD labels has the potential to significantly reduce the time it takes for patient data to be processed by hospitals, and also potentially reduce lost time and money caused by human error in diagnostic paperwork.

Expert Knowledge And Testimony

We were given the opportunity to work directly with medical practitioners for the development of our model. In addition to the data on heart failure liklihoods provided by heart failure domain experts, we received testimony from several practicing doctors on the role that ICD code assignment plays in their job, and the impact that a software improvement could have on their practice.

ICD codes are a vital part of hospital billing, and require trained personnel to ensure correct and quick assignment of codes to patients based off patient data. This means an additional role that hospitals have to hire for, train, and pay, and any complications with those 3 responsibilities can significantly delay hospital processing of patient data, causing additional lag time in an already overtaxed medical system.

Medical practitioners at the Minneapolis Clinic of Neurology provided an alternative perspective coming from a smaller hospital. Working at a facility with fewer than 5 doctors as of the time of the interview, the clinic does not do enough business to afford staff who hold the explicit job of code assignment. The only employees on staff with enough training to assign codes to patient paperwork is the doctors themselves. This means that the task of code assignment increases the amount of time doctors need to spend on each patient, and reduces the number of patients doctors have time to see in a day. For smaller clinics like these, software that can accelerate the process of Code Assignment while maintaining accuracy has the opportunity to help alleviate the long-standing issue of smaller hospitals having extremely long wait times by increasing the number of patients doctors can afford to book in a day.

The testimony of medical personnel also helped to inform our approach to the problem of Cohort Selection. We were advised that training a model which selects too many patients for a cohort is preferable to a model that is more discerning, but leaves out too many HF-positive patients.

Challenges of Medical Data

Working with medical text presents unique challenges not present in more general text processing. One major difficulty is the highly specialized and jargon-heavy nature of medical language, which often includes complex terminologies, abbreviations, and acronyms that are specific to various medical fields and conditions. This specialized language can be a barrier for NLP systems, which need to be able to accurately interpret and classify medical terms and their relationship to each other to extract meaningful information. Additionally, medical texts frequently involve nuanced descriptions of symptoms, diagnoses, and treatments, which require a deep understanding of both medical knowledge and contextual nuances in addition to the capacity to determine the difference between highly-similar terms.

In contrast, legal texts and novels, while also complex, tend to present different types of challenges. Legal texts often involve formal, uniformly-structured language with specific jargon related to laws and regulations, which can be challenging due to its precision and context-dependent meanings. Novels, on the other hand, are characterized by their rich, narrative-driven language and varied writing styles, which can pose challenges in terms of stylistic variability and thematic analysis. However, medical texts are distinctive in their demand for a high level of domain-specific knowledge and the need to accurately capture the subtleties of medical information, making them particularly challenging for text classification and analysis in the context of machine learning and healthcare applications.

CHAPTER 2: BACKGROUND

Related Works In NLP And LLMs For Clinical Text

Our choice of architecture is informed by past research in machine NLP and medical research. In order to best utilize all knowledge available within medical text, we reviewed previous models and techniques to evaluate their suitability for our problem setting.

Bag of Words

The Bag of Words (BoW) approach is one of the oldest techniques in text processing and NLP that represents text data in a simplified, numerical format [17]. In BoW, a text document is transformed into a vector based on the frequency of words contained within it, completely disregarding the order and grammatical structure of the words. Each unique word in the text dataset is treated as a distinct feature, and the document is represented by a vector that counts the occurrences of these words. This method enables the conversion of text into a simple numerical format suitable for machine learning algorithms, allowing for straightforward implementation of classification, clustering, and other analytical tasks. Despite its simplicity, the BoW model has proven effective for various NLP applications, serving as a crucial building block in text analysis.

However, the Bag of Words approach also has many significant limitations, primarily related to its disregard for word order and context. By treating each word as an independent feature, BoW fails to capture syntactic and semantic relationships between words, which can be crucial for understanding the meaning of a text. Additionally, the size of the vectors is dependent on the number of different words present in the document, which can increase computational costs and pose challenges in terms of memory usage when working in a text setting involving many domain-specific terms.

To address these issues, many subsequent methods would represent text documents by reducing words (or atoms of words, called ’tokens’) to individual vectors and would represent the overall document as a sequence of these vectors in order to provide more nuanced representations of text, incorporating contextual information and reducing dimensionality. The first major example of this formulation of text processing was Recurrent Neural Networks.

Recurrent Neural Networks

Recurrent Neural Networks (RNNs) have been instrumental in advancing text processing by addressing the limitations of the Bag of Words (BoW) approach [33]. RNNs are designed to capture the ordered nature of text data. By processing text in a sequential manner and maintaining a hidden state that evolves over time, RNNs can model the positional relationships between words and their significance in-context. This ability to capture context and word order allows RNNs to generate more nuanced representations of text, which enhances the performance of NLP tasks. Variants of RNNs, such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs), have further improved the handling of long-range dependencies and mitigated issues like vanishing gradients, making RNNs even more effective for complex text processing applications.

Despite many advantages over previous approaches, RNNs come with certain drawbacks. One notable limitation is their computational inefficiency, particularly with long sequences, due to their sequential structure being incapable of parallel processing. This inefficiency can lead to longer training times and increased resource requirements. Additionally, while RNNs excel at capturing contextual information, they may still struggle with very long-term references and require careful tuning and architecture design to avoid overfitting and retain generalization capacity.

To address these challenges, newer architectures such as Transformers have been introduced, offering improved scalability and parallelization capabilities.

Attention Mechanisms

Attention mechanisms have revolutionized machine NLP particularly in conjunction with transformer-based model architectures. At their core, attention mechanisms enable models to dynamically weigh the significance of different input tokens when generating output representations. This is particularly crucial in processing sequences of varying lengths, as it allows the model to focus on relevant parts of the input while disregarding less pertinent information. The fundamental operation involves computing a set of attention scores that quantify the relationships between tokens, which are then normalized to produce a weighted sum of the input representations. This process not only enhances the model's capacity to capture contextual relationships but also facilitates parallel processing, bringing the computational cost of working with transformers back into the realm of feasibility for modern hardware.

Transformers And LLMs

Transformers are the advancement responsible for the modern revolution in the field of machine NLP, fundamentally altering how text data is processed and understood. Introduced by Vaswani et al. [39] in 2017, the Transformer architecture leverages 'self-attention' mechanisms to weigh the importance of different words in a sequence relative to one another. This enables the model to capture long-range dependencies and contextual relationships more effectively than previous architectures. Transformers eschew the sequential processing characteristic of RNNs, allowing for parallel processing of data, which significantly enhances training efficiency and scalability. The core components of the Transformer model include multi-headed self-attention layers and feed-forward neural networks, which collectively enable the model to generate rich, context-aware representations of text.

In transformer-based language models, often simply referred to as LLMs, attention mechanisms are implemented through multi-headed attention layers, which allow the model to simultaneously operate on different parts of the input. Each head learns to focus on various aspects of the input, contributing to a richer and more nuanced understanding of language. The self-attention mechanism enables each token to interact with every other token in the sequence, effectively capturing long-range dependencies without the limitations imposed by sequential processing found in earlier architectures like recurrent neural networks. This flexibility is pivotal for tasks requiring contextual comprehension, such as translation and summarization, as it empowers LLMs to generate outputs that are coherent and relevant within local context without sacrificing compatibility with text present outside the current context. The integration of attention not only enhances model interpretability by allowing researchers to examine which tokens influence specific outputs but also lays the groundwork for the very transfer learning that we take advantage of in our pipeline, where models pre-trained on more general text can be fine-tuned for a variety of downstream tasks, often with more setting-specific data.

Despite their transformative impact, Transformers come with the challenge of yet greater computational and memory demands, which can be substantial given the model's reliance on the self-attention mechanisms that scale quadratically with sequence length. This can lead to high costs in both training and inference steps, particularly with larger models and datasets. Additionally, while Transformers excel in capturing complex contextual relationships, they require large amounts of data to train effectively and avoid over-fitting. Being a relatively recent innovation, research remains focused on improving the efficiency and scalability of Transformer models through techniques such as sparse attention and model distillation/quantization to address these and more limitations while continuing to leverage their powerful capabilities in understanding and generating natural language.

LoRAs

LoRA, or Low-Rank Adaptation [18], is a technique designed to fine-tune large language models (LLMs) efficiently by introducing low-rank matrices into their architecture. By leveraging the idea that the weight updates during training can often be approximated as low-rank modifications, LoRAs allows researchers to adapt pre-trained models to specific tasks without incurring the high computational costs associated with full model retraining. This approach not only reduces memory usage and speeds up training but also maintains performance levels comparable to traditional fine-tuning methods, making it particularly attractive for resource-constrained environments.

Training LoRA involves augmenting the original model's weight matrices with additional low-rank matrices that capture the essential adaptations needed for a specific downstream task. During the training process, the updates to these low-rank matrices are learned while keeping the majority of the original model parameters frozen. This selective tuning enables practitioners to achieve rapid convergence with fewer training samples and significantly lower computational overhead. Additionally, LoRA's modular nature allows for easy integration into existing architectures, promoting flexibility and scalability in various applications, from sentiment analysis to more complex tasks such as machine translation.

The implications of LoRA in NLP are significant, as they turn a task that would involve training the entirity of a model into a task that can leverage larger models without requiring the full compute power of training them. Empirical studies have shown that LoRA can maintain or even enhance performance metrics while requiring only a fraction of the resources typically necessary for fine-tuning. As the field of machine learning continues to evolve, LoRAs stand out as a pivotal advancement that not only addresses efficiency concerns but also fosters innovation by making powerful ML capabilities more accessible.

Previous Works In Medical NLP

CorNet

CorNet [42] represents in interesting innovation in XMTC models. Instead of changing model architecture with the end-goal of better capturing information about the text provided to the network for classification, CorNet adds a new layer to an existing model (referred to as the backbone model) with the express purpose of learning how the labels of the dataset correlate with each other during training. Traditional deep learning approaches for XMTC often overlook the relationships between labels, treating each label as an isolated prediction. CorNet addresses this limitation by integrating a specially-constructed module - the CorNet module - at the output of the prediction layer, which learns and utilizes learned label correlations to refine and enhance the initial label predictions generated by the deep model and output refined predictions. This approach allows the model to make an initial prediction and then refine its label predictions based on what labels it has learned are correlated, leading to more accurate and contextually relevant label assignments. This also means that more labels lead to more information being provided to the CorNet module,

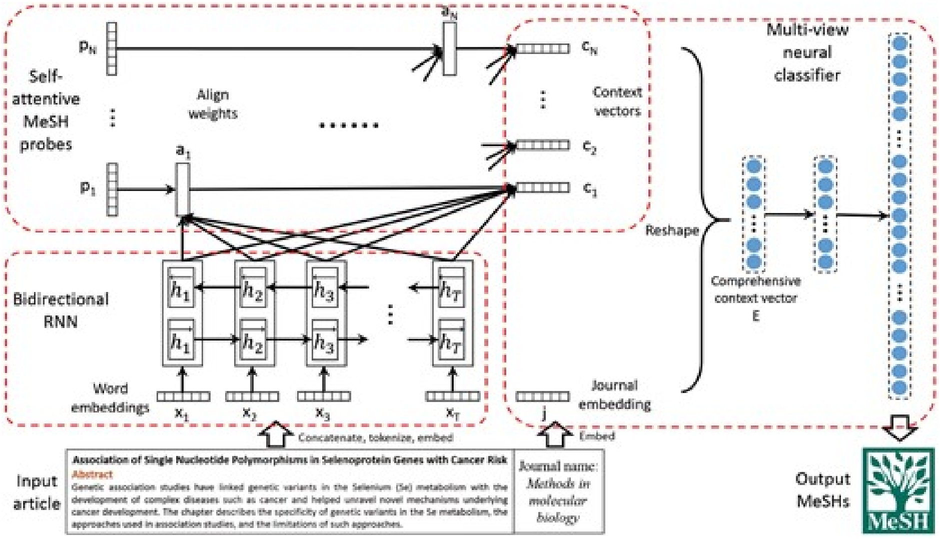

Self-attentive MeSH probes
p_N
p_1
Align weights
a_N
a_1
Context vectors
c_N
c_2
c_1
Multi-view neural classifier
Reshape
Comprehensive context vector E
Bidirectional RNN
h_1
h_2
h_3
h_T
Word embeddings
x_1
x_2
x_3
x_T
Journal embedding
j
Concatenate, tokenize, embed
Embed
Input article
Association of Single Nucleotide Polymorphisms in Selenoprotein Genes with Cancer Risk
Abstract
Genetic association studies have linked genetic variants in the Selenium (Se) metabolism with the development of complex diseases such as cancer and helped unravel novel mechanisms underlying cancer development. The chapter describes the specificity of genetic variants in the Se metabolism, the approaches used in association studies, and the limitations of such approaches.
Journal name: Methods in molecular biology
Output MeSHs
MeSH

Figure 1. Diagram of the MeSHProbeNet module architecture[43]

making CorNet both perfectly-suited for XMTC tasks and less effective on others.

By capturing and utilizing label correlations, CorNet has been shown to reliably enhance prediction quality at loss convergence [42]. The initial publication showed significant performance improvements in general text settings [8] using multiple backbone models to generate the initial predictions. The difference in performance as well as the final accuracy varied between different backbones and datasets, but in all cases CorNet showed either a consistent improvement in or equal performance. The fact that CorNet never once showed a decrease in performance makes it an appealing and nearly downside-free method of augmenting XMTC pipelines.

In our experiments, we used the following backbones for CorNet:

MeSHProbeNet

MeSHProbeNet [43] is an XMTC model specifically designed for the classification of biomedical texts. It employs a multi-channel convolutional neural network (CNN) that integrates both local and global context representations of text (shown in Figure 1), which gives it a method by which to capture intricate relationships among various MeSH terms. By leveraging the transfer learning that comes with with pre-trained embeddings, MeSHProbeNet enhances feature extraction from biomedical literature, thereby improving the model's ability to classify texts with complex

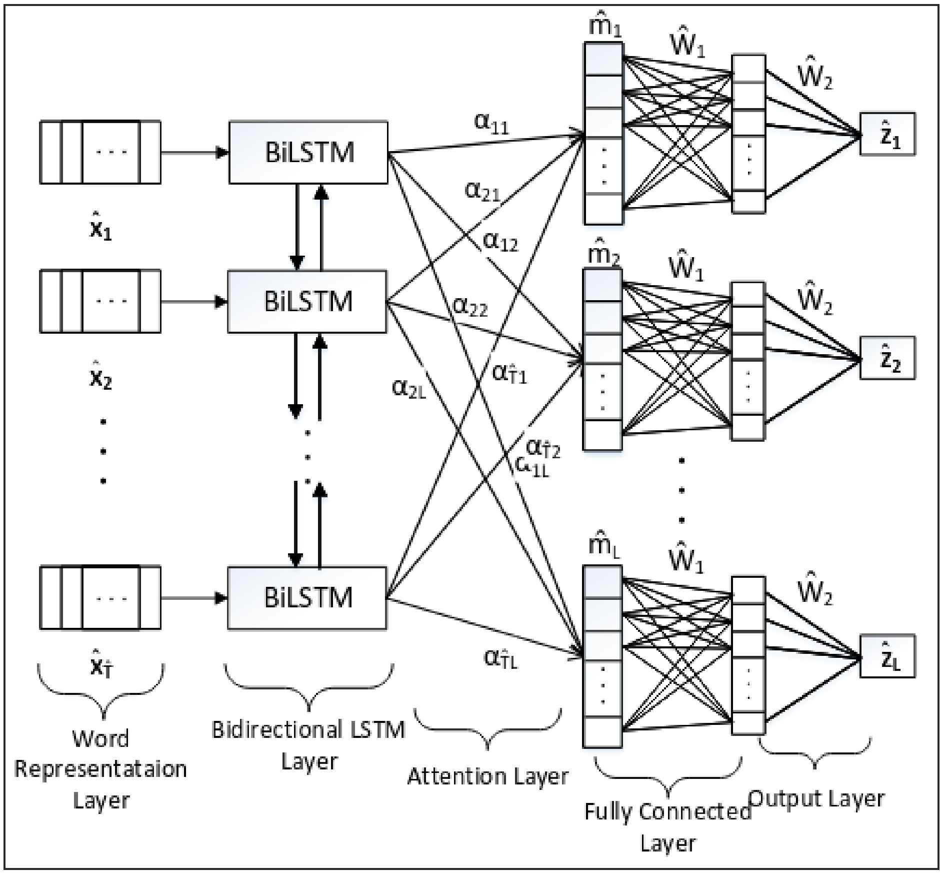


Figure 2. Diagram of the AttentionXML model architecture[46]

terminologies and hierarchical structures inherent in MeSH.

MeSHProbeNet was validated in its original publication on a variety of biomedical datasets, demonstrating its effectiveness in accurately assigning relevant MeSH terms to articles. The model outperforms traditional classification approaches and also showcases its robustness in handling datasets with varying label distributions, establishing MeSHProbeNet as a valuable tool for information retrieval in biomedical research. This work emphasizes the importance of specialized models for domain-specific tasks, contributing to advancements in automated literature classification and improving the accessibility of biomedical knowledge.

AttentionXML

AttentionXML [46] is a novel framework for multi-label classification that leverages attention mechanisms to efficiently handle large-scale label spaces. It uses a hierarchical attention model (shown in Figure 2) that captures both local and global label dependencies which enhances the learning process by focusing on the most relevant words and phrases for each label.

The experimental results indicate that AttentionXML achieves superior performance in both precision and recall compared to (at the time) existing multi-label classification methods across several benchmark datasets. The hierarchical attention mechanism effectively captures complex label correlations, which is critical for improving classification accuracy in medical settings. The model is also shown to be efficient, demonstrating its capability to reduce training time significantly while maintaining high-quality predictions. These results make the model a good candidate for backbone testing with CorNet in medical settings.

General Text Datasets Used

AmazonCat

AmazonCat is the name of a dataset provided by Manik Varma's Extreme Classification Repository[8]. AmazonCat contains the data necessary to train a model to perform the task of automatically predicting the category that should be assigned to a newly-listed product on Amazon based on the text content of the fields provided.

Amazon product descriptions and titles are written for a general audience in mind with the objective of appealing to as many customers as possible, and Amazon carries one of the widest variety of products of any store. This combinations of wide subject matter and simple language makes AmazonCat an ideal dataset for evaluating a model's capacity for generalized learning before evaluating it against more specialized datasets.

EUR-Lex

EUR-Lex [10] is an extensive collection of legal documents related to the EU, primarily focused on legal texts. This repository includes treaties, regulations, directives, and case law, providing a rich dataset for linguistic and legal analysis.

Legal texts are filled with specialized terminology and syntactic structures far more complex than common speech that can pose challenges for models. This means that the style of language present within EUR-Lex poses similar challenges to medical text and allows us to test the effectiveness of models against complex language while remaining distinct from the medical text we will contrast performance with.

Medical Text Datasets Used

PubMed

PubMed [32] is a comprehensive resource within the biomedical domain, offering an extensive repository of biomedical literature and research articles. It encompasses over 35 million citations from a diverse array of sources, including peer-reviewed journals, clinical studies, and systematic reviews. The dataset primarily includes abstracts and bibliographic metadata from articles, with a focus on research in medicine, life sciences, and related fields. Each entry typically contains important metadata such as titles, authors, publication details, MeSH (Medical Subject Headings) terms, and abstracts, which collectively provide a rich source of structured and unstructured text data. This wealth of information facilitates detailed exploration of biomedical topics and trends, allowing for sophisticated analyses and insights into medical research and practice.

In the context of natural language processing (NLP) and machine learning, PubMed's dataset offers significant potential for enhancing text comprehension and analysis capabilities. The structured metadata, including indexed terms and categorical classifications, provides a framework for developing and training algorithms that can parse, categorize, and interpret complex biomedical terminology and relationships. The unstructured textual data, particularly abstracts, presents challenges and opportunities for models aimed at understanding and generating human-like text. Effective NLP models must navigate domain-specific jargon and intricate scientific concepts, making PubMed a valuable resource for developing systems that require an advanced grasp of specialized language and contextual meaning.

Moreover, the diverse range of publications represented in PubMed enables the development of machine learning algorithms that can identify trends, correlations, and emerging topics within the biomedical field. By leveraging the dataset's depth and breadth, researchers can train models to perform tasks such as document classification, information retrieval, and semantic similarity assessments with greater precision. As such, PubMed not only serves as a critical dataset for advancing biomedical research but also contributes to the broader field of text comprehension, offering insights into how machines can achieve a nuanced understanding of complex, domain-specific language.

Our work with MeSH focused on using the text of the abstracts to classify documents by identifying the MeSH terms that should be assigned to them. MeSH terms provide a controlled vocabulary that systematically categorizes articles into predefined topics, facilitating the organization and retrieval of information. These hierarchical descriptors enable more accurate classification of documents by associating them with specific medical concepts, diseases, treatments, and research methodologies. When integrated with the textual content of abstracts, which offer detailed descriptions of study objectives, methodologies, and findings, MeSH terms augment the ability of machine learning models to categorize and interpret biomedical literature. This combination allows for refined classification strategies, where algorithms can leverage both the structured tagging provided by MeSH and the semantic richness of the abstracts. Consequently, models can achieve higher precision in tasks such as topic modeling, literature summarization, and trend analysis, ultimately leading to more effective and contextually relevant insights in the biomedical domain.

MIMIC-III & MIMIC-IV

The MIMIC-III [20] and MIMIC-IV [1] datasets are both extensive critical care databases that provide a wealth of de-identified patient notes, perfect for training models in medical NLP. MIMIC-III, which encompasses data from over 40,000 ICU admissions from 2001 to 2012, includes comprehensive clinical information such as vital signs, laboratory results, medications, and notes from healthcare providers. MIMIC-IV builds upon this foundation, extending the dataset through 2019 and incorporating additional features like updated diagnosis codes and more granular time-series data. Together, these datasets offer a rich and verbose perspective on patient care in intensive settings, enabling the development and validation of predictive models that can augment decision-making steps in critical care environments that are currently dependent entirely on human effort.

In comparison to other medical datasets, the MIMIC-III and -IV databases stand out for their scale, granularity, and temporal coverage. MIMIC's extensive and continuous data collection from ICU settings offers unique insights into the dynamics of critical illness and intensive care management. The detailed time-series data and the wide array of clinical variables included in MIMIC-III and MIMIC-IV facilitate sophisticated analyses of patient progression and personalized treatment plans. This level of detail and breadth makes the MIMIC datasets a critical resource for training models with the aim of lending a deeper understanding of how patient factors (diseases, medications, pre-existing conditions, etc) interact.

University of Iowa Hospital And Clinic Data

We were granted access to select files from the medical database used at University of Iowa Hospitals and Clinics for use with our experiments. This data gave us a unique opportunity to work with the kind of text that a machine-learning-based NLP model would actually need to process in a live-setting in order to make a meaningful impact on the ability of medical professionals to diagnose and treat patients in a timely and accurate manner.

Challenges of Working With Raw Hospital Data

One notable challenge of working with the total output of a hospital's database is the sheer scale of text to work with. As we treated each patient as a single data point and combined all of their medical documents for input into the model (discussed later), the per-patient text document length varied from thousands of tokens to hundreds of thousands of tokens. Document length is in itself a challenge, however developing a classification technique that can work equally well with documents that are powers of ten apart in length is another challenge unto itself.

Sourcing text directly from a real hospital without the aid of data scientists cleaning and correcting the dataset means that an approach robust against noise is also required. While MIMIC ([1]) was curated to be accurate in content and clean composition, the data from the hospital is not proofed against mistakes in spelling or medical accuracy. Piecemeal inspection of individual clinical notes found mundane typos and other entry errors such as inconsistent spelling of names, multiple spaces in a row in a sentence, periods in places that commas should be, etc. Use of shorthand text was also noted, such as incomplete sentences and grammatically-incorrect phrases containing only a few key words.

Errors on the part of the medical practitioner entering the text is not the only source of potential problems. Certain categories of clinical notes contained information that was technically relevant to the potential for heart failure diagnosis, but were exceedingly long without containing much information. An example of such a note would be the instructions for a dialysis machine. While it is meaningful and important to know that dialysis was a part of a patient’s treatment, the instruction manuals always have the same text. This means that the ’information density’ of such documents is low, taking thousands of tokens to convey meaning effectively equivalent to ”the patient was also on dialysis”. Attention mechanisms can dampen how much low-information-density text affects the output of the model, but these models still have limits on how much text they can process at once. Thousands of tokens of near-meaningless text can take up a large chunk of a model’s context window and hamper its ability to process long-range context by limiting how many other documents can be processed at once.

The semantic knowledge of the notes and the patterns of their content is what we seek to leverage in order to guide the LLMs to train on their contents more efficiently. Analyzing the difference between curated datasets provided to the public for training purposes and raw data that is completely unchanged from how it is originally entered is what we do to close the difference in performance between classifying the ’cleaned’ data and classifying the ’raw’ data.

The labels for the patients were provided by the heart failure domain experts we collaborated with on this project. They derived a regex-based method that would assign patients a Heart Failure Rating (HFRating) between 0 and 7 based on how many signs of heart failure they displayed. These ratings came from all patient information *except* the text of clinical notes to prevent a model trained on the notes and generated ratings from simply learning to re-create the regular expressions. The information the ratings were based on was comprised on data such as the administration time and dosage of medication and visit times. While the accuracy of patients rated at a 3-4 has not been fully calculated, manual investigation showed that none of the patients given a HFRating of 0 were positive for heart failure while all of the patients rated at 7 were considered positive by consulted medical professionals.

Review of Previous Research

Artificial Intelligence and Medicine are both areas of extreme interest in research, and the overlap between them is thusly extremely large in scale. In addition, with LLMs being a relatively new innovation, there is much unexplored territory. Every quarter there are many papers exploring every aspect of LLMs, from their training methodologies to their application in clinical settings, revealing both their potential applications and limitations [35] [26]. As these models become increasingly prevalent, understanding their capabilities and shortcomings is essential for their effective implementation in healthcare contexts [45] [19].

In order to develop AI-based technology for medical application, we must have a comprehensive and robust framework for evaluating all aspects of the technology we are implementing, from accuracy to efficiency to speed. However research suggests that traditional benchmarks often fall short in assessing the nuanced capabilities required for clinical reasoning [35] [29] [13]. Reliance on automated evaluations may overlook critical aspects of model functionality, leading to a misrepresentation of their true abilities. Therefore, there is a growing advocacy for both more flexible evaluations. Researchers have proposed everything from human-centered methods that incorporate open-ended questioning yielding insights into how well these models perform in real-world scenarios [35], to adaptive tests that change their approach over the course of testing to adapt to ongoing performance of the evaluated models [49], and evaluation utilizing other LLMs fine-tuned on the task of grading output quality [40].

There is also ongoing research into how n-lingual models perform differently in domain-specific tasks. Large-scale bi-lingual models sophisticated enough to attempt competition with state-of-the-art LLMs have already been trained [3] and show impressive performance, but there are many ways to handle mixtures of languages within a dataset, and the most effective way to train an LLM to handle multiple languages within a single context is not yet a solved problem [37]. With the increasingly-global nature of research and the staggering number of languages spoken globally, there is still ongoing research into when medical systems become more accurate simply by translating non-english medical text into english before being fed into a monolingual model [30]. Any research done into automated medical systems will have to consider how it will approach this language barrier problem before seeing truly wide-scale deployment.

Moreover, the capability of LLMs to adapt and provide accurate medical information can be significantly enhanced when evaluated against specialized datasets (10, 11). Studies indicate that models fine-tuned on specific medical data can outperform their generalist counterparts in tasks like radiology interpretation, highlighting the importance of context in evaluating model performance (12, 13). This tailored approach is not only essential for accurate medical advice but also for ensuring that LLMs are reliable tools in clinical practice.

The methodologies for training LLMs have evolved in response to the resource constraints often faced in healthcare settings. Training a model from scratch is not only costly in terms of computational resources [34] but also time-intensive. Research shows that fine-tuning existing models can lead to outcomes superior to training an LLM from scratch with medical text while being a more efficient method of training [25]. This approach allows healthcare organizations to leverage pre-trained models that have already learned from vast amounts of data, thereby conserving both time and energy. This approach has already been taken by several hospitals, choosing to use patient data local to their own practice to train LLMs for medical use [36], and in the process have shown that generalist monolingual english models can be fine-trained for use with other languages. It has also been shown that even without fine-tuning, generalist models show surprising efficacy at medical tasks even without fine-tuning on medical-data [26] [19]. Given this information, it is not yet clear what the best way to fine-tune a generalist model for medical use *is*, as this research seems to indicate that training a generalist model on medical text too much may eliminate the performance boost granted by the generalist foundation of the model.

In addition, training and/or fine-tuning models for medical use is significantly restricted by the availability of data, most often restricted specifically due to privacy. In addition to the simple solution of local hospital training mentioned previously [36], the use of federated learning techniques has emerged as a promising solution to address privacy concerns associated with sensitive medical data [29]. Research into optimal ways of augmenting training data without overstepping privacy boundaries remains ongoing.

As LLMs find their way into clinical decision-making processes, the demand for interpretability becomes increasingly urgent. Healthcare professionals need to trust the recommendations made by AI systems, which means understanding how these models arrive at their conclusions. Innovative approaches to model interpretability such as chain-of-thought reasoning frameworks aim to simulate the diagnostic processes of human practitioners [16] [21]. Training LLMs to solve problems via a chain of thought reminiscent of human reasoning has been shown to both increase performance [6] and also create a more transparent warrant for the conclusion at which the LLM arrives, allowing for more trust in an automated system and more effective human-AI collaboration [11].

Factual accuracy in LLM outputs is a problem of supreme importance. Techniques that augment LLMs with retrieval mechanisms have shown promise in enhancing the reliability of information provided by these systems [4]. By ensuring that LLMs can cross-reference their outputs with established medical knowledge, healthcare providers can feel more confident in the guidance offered by AI tools. This accuracy is particularly crucial in high-stakes environments where incorrect information can lead to severe consequences for patient health.

Capacity for medical reasoning is still being explored as both a problem and a solution. Training LLMs to process text with the direct goal of producing diagnoses accurately has been shown not to actually convey any deep understanding of medical concepts and reasoning ability to LLMs [21]. To solve this issue, many architectural and procedural changes have been proposed to enhance reasoning ability with the eventual goal of increasing the capacity of LLM to solve more complex multi-step problems, which are often necessary in medical settings. Often this is done through Chain-Of-Reasoning approaches (also called chain-of-thought or chain-of-diagnosis). This ‘chain’ approach has been shown to increase reasoning ability and problem solving capacity in many settings [47] [4] including medical.

Despite promising advancements, several challenges remain. Issues such as model biases and the potential for misalignment between LLM outputs and clinical standards remains to be addressed [41] [7]. While tests have found that fine-tuning generalist models often out-perform models trained on medical data from scratch, these generalist models also often have built-in “alignments” to keep models from generating content considered undesirable by either the company training the model or the country in which said company is based, and these alignments can often compromise the accuracy of model outputs [7]. We saw this alignment-accuracy problem with our own evaluations running against different models that had taken different measures to handle alignments.

CHAPTER 3: METHODOLOGY

Extreme Multi-label Text Classification

Problem Formulation

The formulation of the task of ICD code assignment is virtually identical to every other XMTC task; Given a superset of all valid labels L and a set of documents D, for each document $d_i \in D$ we must find the correct subset of labels $L_i \in L$ that should be assigned to that document, minimizing the number of incorrectly-assigned labels and missed labels $e_{error} = |(\hat{L}_i \cup L_i) - (\hat{L}_i \cap L_i)|$ between the predicted labels $\hat{L}_i$ and the true label set L_i.

This is often done by predicting a label 'confidence value' $\zeta_{l,i}$ for every label $l \in L$ and assigning each document the n labels with the highest predicted confidence values.

This means such classification algorithms must learn the function h where $\zeta_{i,l} = h(d_i, l)$.

Model Architecture

Re-creating the results of the initial CorNet paper [42] was one of our first experiments. Using the same list of backbone models and datasets as the original authors, we found consistent improvements on top of the backbone models alone. After reaffirming the soundness of the theory behind the architecture, we transitioned to medical text to test to what degree the improvement would transfer.

It stands to reason that as CorNet learns the correlation between models, it has potential to improve performance so long as the labels are not entirely independent. When labels represent medical topics, CorNet has the potential to assist in learning correlations that would appear obvious to casual observers (such as correlation between surface wounds and symptoms of blood loss stemming from the blood lost through the wound, etc) and previously-unknown connections between medical concepts that could lead to worthy research in their own right. For this reason the performance of CorNet was tested against both PubMed[32] and MIMIC IV[1].

ControlNet [48] and IPAdapters [44] are image generation technologies that were quickly adopted by end-users of image generation technology immediately after their use, and even as newer and much more advanced models have been released, ControlNet and IPAdapter compatibility [22] patches [14] have been made to use the comparatively old techniques with these newer models. This shows that less-novel techniques that augment instead of replacing current techniques have potential to make a more lasting impact than just an iterative replacement for the state of the art.

Cohort Selection

One of the primary goals of our research was to improve cohort selection for medical studies. We worked closely with doctors at University of Iowa Hospitals and Clinics at which we were granted access to the patient information database. While the database contains the sum of all medical information available regarding the patients health, our focus was on clinical notes recorded for each patient and the information that could be extracted from them using text models, namely large language models.

The medical experts we worked with were focused on automatically predicting whether a given patient suffered from heart failure based only on the text of the notes that were written about them during their hospital visits. This information could then be used to select patients for testing and studies automatically, a process which usually involves doctors going through patient files one by one, by hand.

This means that automation of this process could potentially speed up medical research significantly by eliminating the overhead associated with setting up a medical study. The initial experiment was focused on identifying heart failure, but given the widespread use of LLMs in general context [24] [15], there is indication that such a tool could be easily scaled to identify other traits given training and information.

Problem Formulation

At its largest scale, the task of cohort selection can be expressed as

$$C = \{p \in P : f(p) = true\}$$

where C is the set of patients to be included in the cohort, P is the set of all patients, and f is some predicate that decides whether a given patient should be included in a cohort.

For our purposes, it is convenient to define f as $f_t(p) = \zeta_p > t$, a simple predicate filtering patients based on whether some ”confidence rating” is above the given threshold t.

This allows researchers using this formula to adjust the threshold based on their needs, using a higher threshold when it is desirable to have as few patients wrongly-included in the cohort as possible, and using a lower threshold when it is desirable to miss as few of the patients that should be included as possible.

This variable ζ_p represents the label data provided to us by medical experts, and it is the confidence estimation function $h(p)$ such that $\zeta_p = h(p)$ which our model must learn to approximate.

In our experiments, we defined $h(p)$ as

$$h(p) = \square_{d \in D_p} h_{doc}(d)$$

Where we use $\square$ to represent some aggregation function (such as summation or averaging), and h_{doc} is a function that generates a confidence value for a given text document d coming from the set of documents D_p, all text documents associated with a patient.

Thus the full formulation of the problem is

$$C = \{p \in P : ((\square_{d \in D_p} h_{doc}(d)) > t)\}$$

where we trained and evaluated our LLMs to approximate h_{doc} for the hospital documents.

Fine-Tuning LLMs

Fine-tuning LLMs has become a major area of research given the proven efficacy of generalist text models. LoRAs provide an efficient approach to model adaptation by introducing low-rank matrices into the existing architecture of pre-trained models. This method capitalizes on the fact that many language tasks can be achieved with a very limited number of additional parameters, thereby reducing the computational burden associated with full model fine-tuning. By fixing the original weights of the pre-trained model and only training the added low-rank matrices, LoRAs allow researchers to achieve task-specific performance improvements while maintaining the integrity of the base model.

The core technical mechanism behind LoRAs involve the decomposition of weight updates into a low-dimensional space. Instead of modifying the entire weight matrix of a billion-parameter model, LoRAs insert trainable matrices A and B, where the rank of A and B is much lower than that of the original weight matrix W. This transformation effectively reduces the number of parameters that need to be tuned during training, which is especially advantageous when dealing with very large models. As a result, LoRAs accelerate the training process and partially mitigate the risk of overfitting by constraining the adaptation to a lower-dimensional subspace.

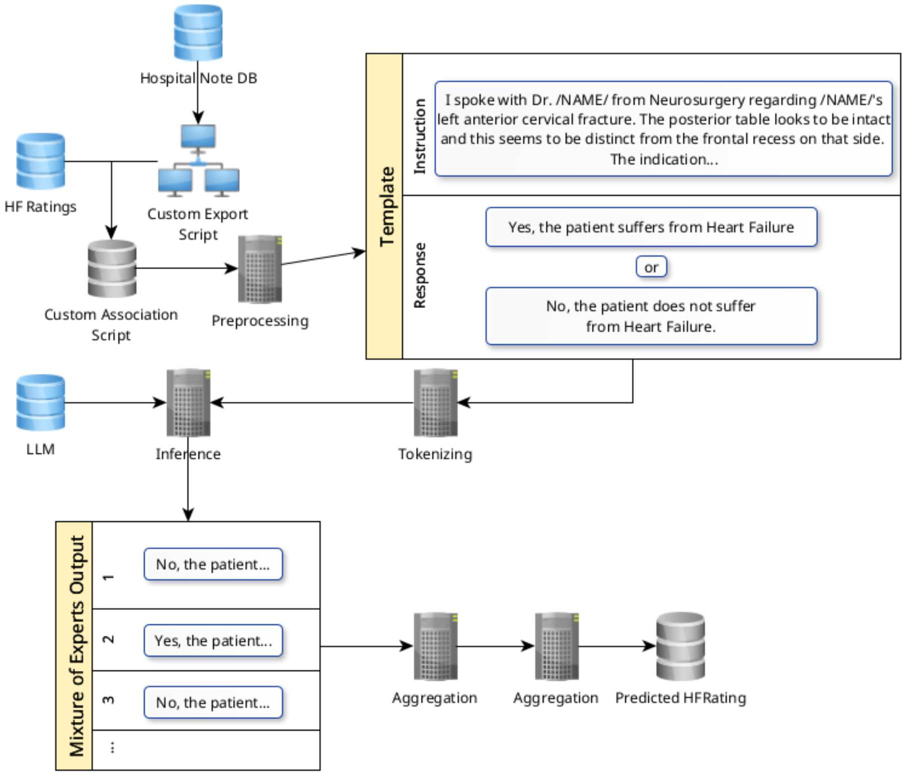
Additionally, a LoRA can be integrated with existing training frameworks and can be applied to various architectures, including transformers. This adaptability made it a compelling choice for our use with LLMs and the hospital data. The low-rank matrices can be introduced at multiple points within the model, allowing for flexible adjustments based on the task at hand. Moreover, the use of LoRA often leads to improved generalization, as the pre-trained model retains its ability to leverage knowledge acquired during initial training while adapting efficiently to new contexts.

The SFTTrainer provided by the Huggingface transformers library serves as a robust framework for fine-tuning LLMs using LoRAs. The trainer provides a simple interface for incorporating low-rank adaptations into the training pipeline, facilitating the configuration of hyperparameters specific to a LoRA such as rank size and learning rates for the added matrices. Additionally, SFT-

Trainer is designed to manage the intricacies of distributed training, making it scalable for large datasets and multiple GPUs of variable size. This solved the problem of ensuring the model converges appropriately while maintaining the integrity of the base and adapted components. This integration enhances the efficiency of the fine-tuning process to a degree that allowed us to fine-tune much larger and more complex language models than would otherwise be able to train within the time afforded for this experiment.

Hospital Cohort Selection

Design of CohortLang



I spoke with Dr. /NAME/ from Neurosurgery regarding /NAME/'s left anterior cervical fracture. The posterior table looks to be intact and this seems to be distinct from the frontal recess on that side. The indication...

Yes, the patient suffers from Heart Failure

No, the patient does not suffer from Heart Failure.

Figure 3. Diagram of CohortLang pipeline.

Action Taken	Number
Unchanged	96
Removed Entirely	3
Replaced by equivalent	88

Table 1. Number of unique characters treated in different ways

Database Export

The raw data is stored in a highly-separated SQL database, with each datapoint relative to our interests spread across multiple tables connected by multiple joins. There are many ways by which we could connect the notes together, especially accounting for the different times and dates associated with each of the notes. To establish a baseline evaluation of the fitness of LLMs to accomplish this task, we opted to concatenate all documents associated with each individual patient, and treat the entire histories of each patient as a single data point. While this means documents far apart in time are no longer separated, it puts all text associated with a given heart failure rating together as input to the system.

The heart-failure ratings were generated for individual hospital events, which equate roughly to a single hospital visit. Each hospital event has an associated key both for the patient and for the hospital event. We began preprocessing by finding all documents associated with the pair of event and patient key associated with each heart failure rating, and grouping them together.

Characterset Simplification

The most obvious source of complexity present in the notes was the wide variety of special UTF characters left in the exported text. On initial inspection it was not clear which of these characters were unintended errors and which held some important meaning. Manual inspection was required to determine when each of the special characters would occur, and if their occurrence had any meaning such that they could not be replaced by a character present within the tokenizer's lexicon.

After inspection and classification of non-standard-english characters, characters deemed meaningless were removed entirely. Groups of characters deemed to have equivalent meanings were replaced with a single character of their set (e.g. 7 different characters approximating the meaning of quotation marks were found. Most diacritics were replaced with undecorated letters. All common alphanumeric and grammatical characters were untouched. The breakdown of character treatment is shown in Table 1. Overall, this process rendered the text simpler in composition for easier processing in a way that did not greatly affect structure.

Note Type Filtering

Within the patient notes, we found many text artifacts that added complexity or length without adding information. Unnecessary complexity would serve only slow convergence of the model, spending time adapting the weights to properly process things that could have been removed automatically before model inference, and unnecessary length would simply add to training and inference time and increase the chances that CohortLang would de-emphasize earlier important information. Thus a decent amount of effort was dedicated to experimenting with preprocessing methods to see what allowed the model to learn faster without removing important knowledge from the patient notes.

Document Lengths Statistics, in number of Note Sections

Metric	
Average Section Num	536
STD	566
Min. Section Num	39
25 Percent	251
50 Percent	392
75 Percent	629
Maximum Section Num	29092

Table 2. Summary of number of note sections per patient

After the character set was simplified, the total document text *D* was split into sub-sections *s* along tokens deemed to be spacing tokens. Each of these subsections was assigned a score based on how many commonly-seen anti-patterns were recognized with it. These anti-patterns include typos, inconsistently-spelled names, long documents indicated by phrases like “toll-free” or “no

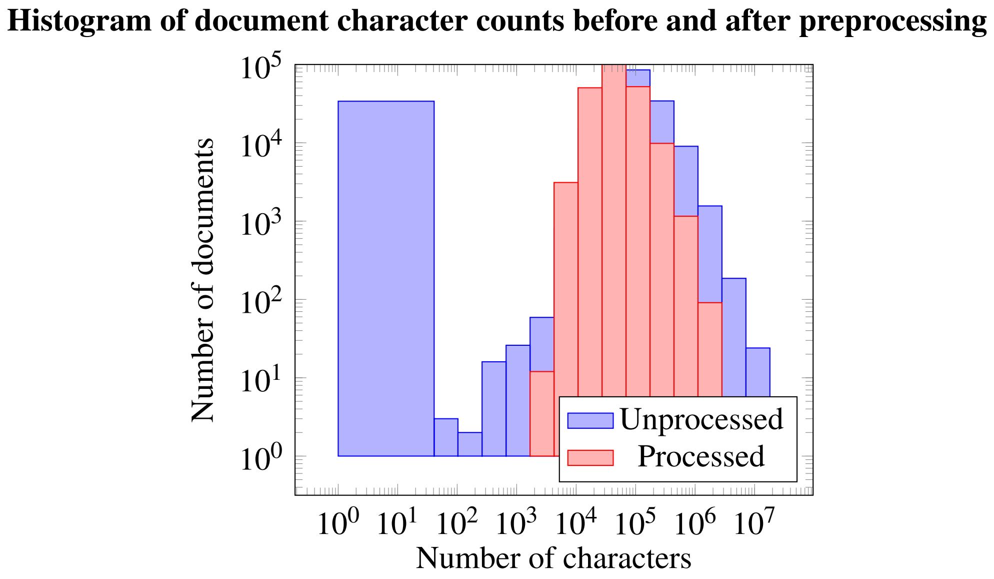


Figure 4. Character counts before and after

past". Patients with *no* usable documents were rejected from the dataset altogether, as it was unlikely the model could be trained on any meaningful information from them or be expected to accurately predict their likelihood for heart failure during evaluation. The number of document sub-sections per patient (after removing patients with no good documentation) can be found in Table 2.

The remaining text fragments were then re-combined into one using a spacer character. We tested two different spacer characters to denote separation between documents and/or sections; One was the "/SEP/" token used by some major LLMs, the other was a simple period and space. The SEP token saw better performance in models specifically designed to recognize it, but the standard sentence separator saw better overall performance. As models can learn to recognize the roles that previously-unseen tokens have within the documents they are trained on, it was assumed that even models untrained on the dedicated separator token would be able to infer the context and meaning of its use, but whatever degree to which they learned to understand it did not outweigh the training time they saved by not needing to interpret standard grammar symbols.

The final effect pre-processing had on the lengths of patients' notes can be found in Figure 4 and Table 3. In particular this shows how many patients had written medical history with under 100

Statistics on Document Lengths, in characters

Metric	Unprocessed	Processed
Number of Documents	266383	232057
Mean Length	119697	63767
Length STD	230216	75903
Minimum length	1	3640
25 percent	34478	28267
50 percent	67182	43963
75 percent	129934	72771
Max Length	18571322	4609261

Table 3. Document length statistics

characters before processing, and how processing lowered the maximum note length by a power of ten, allowing much more information to fit within the context window of the model.

Text Subsegmentation

All collected hospital documents were longer than the context length of the models used for evaluation. Multiple methods were used to process the text into a length usable by the rest of CohortLang.

- First-n truncation, removing all tokens after the context length of the model.
- Last-n truncation, taking the context length from the end of the text.
- First-Last-n/2 truncation, taking tokens from the beginning and end of the text and removing the middle.
- Length-bounded subdivision, splitting the text P_i into substrings P_{ij}. An optimal-split algorithm is used to reduce the superset of text to as few substrings as possible while maximizing the length of all individual substrings.

Length-bounded subdivision is the only method that leads to there being more than one document per patient, leading to a need for combining multiple document predictions into one patient prediction.

The Optimal Split algorithm is one we designed to divide a document into a set of groups of text sections or "splits" that preserve full sentences, maximize individual split length to take maximal advantage of LLM context windows, and ensure all splits fit within a given context window size.

Given a document D and a context length c, a set of sentences S is collected by splitting on the simple pattern of a period followed by a space. (". ")

From S, we calculate a list of integers L_S each representing the number of words a sentence.

$$L_S = \{|s| : s_i \in S\}$$

We calculate the number of words by splitting each sentence s_i on the space character. The word count is used to roughly approximate the number of tokens in each sentence.

While a tokenizer could be used to calculate the exact number of tokens, different tokenizers were used for different models, and it was deemed undesirable to split the text differently for each model trial.

We also calculate the total number of words in the document L_D.

$$L_D = \sum_{l \in L} l$$

To generate the initial splits, we divide D into a set of splits with an equal number of sentences in each split. We calculate the number of sentences in each split from L_D and c,

$$L_{splits} = |L_S|//(1 + L_D//c)$$

And then use that number to generate our set of splits P_{init}

$$P_{init} = \{S[w * L_{splits} : (w+1) * L_{splits}]\}$$

Once we have our starting splits, we calculate a score for the split set.

The score per split is calculated

$$\Phi_i = \begin{array}{cc} (c - |p_i|)^2 & p_i < c \\ b & p_i > c \end{array}$$

Where b is a large but not infinite constant. (10000) We use b instead of max int or infinity because it is beneficial to factor how many sentences are over c into the score, meaning there must be room for a larger number.

The score for all splits is calculated from the root of the sum of the scores.

$$\Phi = \sqrt{\sum_{\Phi_i \in \Phi} \Phi_i}$$

After calculating the starting score, we calculate the scores of all possible split-sets created by shifting a single sentence from one split to another. If any of the scores produced by said shifts are lower than the current score, that split becomes the current split. This process continues until the current split has a score lower or equal to all possible sets formed by a single-sentence shift.

Once a minimum score is found, the algorithm returns the set of sentences.

Segment Prompt Formatting

To direct the LLM chosen to output the information we need, the text of the patient note is inserted into a prompt template.

```
### INSTRUCTION
[ patient note text ]
Does this patient suffer from heart failure?
### RESPONSE
```

This prompt template matches the Instruction template used to train many popular LLMs, often referred to simply as Instruct models. This format was used because we found that even models not specifically trained on this Instruct format performed well with it.

During training, the model was trained to respond specifically with either “Yes, the patient suffers from heart failure” or “No, the patient does not suffer from heart failure”. This method of training it to respond with only “yes” or “no” trains it to respond in a way that facilitates binary classification, and training it to respond with full sentences in that exact style emphasized what “yes” and “no” answers mean in the context of the question and prevents it from responding with technically-correct but undesirable answers such as “There is insufficient information to answer this question” or “Additional testing is required to determine”.

The decision to select the ‘Yes’ or ‘No’ text template during training was made using the ground-truth HF-ratings provided by Dr. Zetumer. He advised that we consider any patient with an HF-rating of 2 or higher as being positive for heart failure.

Prompt Tokenization

Text requires tokenization before being fed into LLMs. Tokenization is a preprocessing step that transforms raw text into manageable unit - called tokens - that can be effectively processed by transformer architectures. By segmenting text into subword units or whole words which are assigned integer IDs, tokenization enables LLMs to handle a vast vocabulary while maintaining a manageable input size, thus facilitating efficient computation and memory usage. Moreover, tokenization allows partial mitigation of the out-of-vocabulary problem, allowing models to generate and/or understand rare or novel words by breaking them down into smaller, recognizable components. This adaptability not only enhances the model’s ability to learn from diverse linguistic datasets, but also improves its performance on downstream tasks.

As CohortLang is built to be used with pre-trained models, the models fine-tuned already have provided tokenizers for pre-processing use. We use the provided tokenizers to convert the formatted text of each split P_{text} into a sequence of token IDs P_{tokens}.

Prompt Inference

The tokens are fed to the model, which generates ε sets of n tokens. This is equivalent to asking the model to answer the question ε times. This is done because the model has a random element to its text generation, and may provide different answers for each seed. Multiple responses allows us to generate an integer confidence rating ζ representing how many times the model answers the question "yes".

A response is considered a "yes" if the token(s) for the word "yes" appear within the n tokens generated for a response. ζ is the number of responses given that are considered "yes" answers.

Mixture of Experts Aggregation

When evaluating a patient's notes for heart failure classification, the inference step outputs an integer number representing "yes". There are multiple methods for combining the responses into the binary positive/negative judgment we require. For each prompt P we can use a threshold based on either the number of "yes" responses ζ or the proportion of answers that were "yes" $\frac{\zeta}{\varepsilon}$. The problem of aggregating a mixture-of-experts answer into a single classification is not a new problem in and of itself. However this problem setting has the abnormal potential to have a variable number of experts/answers per sample depending on the length of the patient's document, which means that established mixture-of-experts methods aren't necessarily the most effective solution.

In this context, taking the sum of all ζ_j is better at catching all HF-positive patients but it is biased towards selecting patients with longer medical histories. This is not necessarily bad, as patients with concerning medical conditions are more likely to have more extensive medical histories. However this bias quickly becomes more of a problem if we want to select for a patient condition that is not likely to correlate with the number of visits a patient has made to the hospital.

Threshhold Decision

As the decision of what threshold value to use is an important decision to be made on a study-by-study basis by the researcher in question, we do not seek to decide on a threshold value ourselves, or train the model to find an 'ideal' single decision boundary. Instead, we evaluate the model at all threshold values and examine the sensitivity-to-specificity tradeoff as the treshold value changes.

Training

When first presented with the hospital data and the task of cohort selection, the first attempted solution was utilizing CorNet as it had already proven both the ability of the backbone models to acceptably parse the more intricate and complex medical terminology and discussions alongside the ability of the CorNet module itself to understand correlations between medical topics represented by the labels of the datasets.

This initial approach was largely unsuccessful, with the best-performing backbone producing results slightly worse than those of a simple logistic regression performed on a bag-of-words representation of the patient document. This wasn't surprising as patients were assigned a single diagnostic code per visit, meaning the vast majority of the patients were assigned a single label. This stands in stark contrast to the general-purpose datasets CorNet was originally designed for, where the average number of labels per document ranged from 3 to 5 between the datasets. If most patients are only assigned one label, this means that very few labels co-occur at all, and there is little correlation for CorNet to learn.

The next test was evaluating the efficiency of Transformer-based LLMs against the complex text of the patient notes.

After the data was cleaned of unneeded decoration characters and low-information documents, all notes were arranged into a simple Instruction and Response format used to train Instruct models, a variety of LLM specifically trained to take text descriptions of a task they are to perform, and respond with their completion of the task - such as asking it to write a CV. The instruction given was simple - after the text of the patient document, the question “Does this patient suffer from heart failure?” was inserted at the end of the text and CohortLang was prompted to write a response.

During the training, CohortLang was trained to respond with either “Yes, the patient suffer from heart failure.” or “No, the patient does not suffer from failure.” During initial training CohortLang was prompted to respond simply with “Yes” or “No”, but when not provided with sufficient context for the answers provided, CohortLang would occasionally respond with other words entirely or respond with sentences that began with “yes” or “no” as a part of a longer statement that did not properly answer the question, e.g. “No clear decision can be made without further information.” The longer answers were then provided to train CohortLang in a way that gave the use of “Yes” or “No” an explicit context that left no ambiguity as to the meaning of the answers.

To evaluate the accuracy of CohortLang in predicting heart failure, an approach akin to a Mixture of Experts structure is taken. In this setting, this means that CohortLang generates multiple responses to the provided question and the responses are combined to form the final prediction, leading to a numerical rating that resembles - but is not necessarily correlated with - the initial heart failure rating provided by Dr. Zetumer’s formula.

CHAPTER 4: EXPERIMENTS

CorNet

Re-creation

Before transferring CorNet[42] to the medical domain, we re-created the results of the original paper on our own hardware. This was done to establish that we were running CorNet correctly and to verify the improvement it is meant to provide over ”bare” models.

We used the same datasets as the original paper, EUR-Lex[8], AmazonCat-13K, and Wiki-500K. For the backbone models, we focused on the models specialized in medical text and speed, AttentionXML[46] and MeSHProbeNet[43].

Adaptation of the CorNet code required recreation of certain integral parts as the original paper used versions of provided packages that were not supported by our current laboratory hardware. In addition to re-writing the parts that referenced the outdated libraries, we took the opportunity to rewrite the structure of CorNet. Whereas the original paper designed and coded custom CorNet models for each backbone, we changed the implementation to express the functionality CorNet as a single wrapper class that can be constructed around any backbone that can take input and output data that CorNet is capable of recognizing.

Transfer To Medical Text

Once the soundness of CorNet on our lab hardware was confirmed, we integrated the MIMIC[1] data into the existing CorNet preprocessing code and formatted all the labels to be correctly interpreted by the system. Hyperparameters had to be tuned in order to accommodate documents that were longer than any in the previous datasets, but aside from requiring additional training time, this added no complications.

Hospital Data

Baseline Results

CountVectoriser

A Count Vectorizer [5] is a fundamental text preprocessing technique widely used in NLP to transform text data into a numerical vector format suitable for machine learning algorithms. This method operates by converting a collection of text documents into a matrix of word or token counts, where each row corresponds to a document and each column represents a unique word from the vocabulary of the entire corpus. The resulting matrix is sparse, indicating the frequency of each word in the respective documents. This representation allows algorithms to interpret the underlying structure of the data while retaining essential information about word occurrence, thereby facilitating subsequent analytical processes.

Unlike alternative preprocessing methods, such as TF-IDF (Term Frequency-Inverse Document Frequency), which weigh word importance based on frequency and document distribution, Count Vectorization maintains a straightforward count representation that can be more intuitive for certain models. For instance, in logistic regression, the simplicity of the count matrix aids in the direct interpretation of coefficients, making it easier to understand the influence of individual features. Similarly, decision trees can effectively leverage the count data to partition the feature space, potentially improving the model's interpretability. Therefore, while Count Vectorizers provide a more basic feature representation, they offer unique advantages in preserving interpretability and direct relationships within the data, which can be crucial for specific applications in machine learning.

Logistic Regression

A standard Logistic Regression model (provided by the commonly-used software package SKLearn [28]) was trained on the CountVectoriser embeddings on the hospital text as a control test.

Logarithmic regression is one of the most widely-used and common statistical modeling techniques [31] [12] [9] where the logarithm of the dependent variable is regressed against one or more independent variables. This approach is particularly useful when the relationship between variables exhibits exponential growth patterns, as it can stabilize variance and make patterns more interpretable. When applied to the output of a CountVectorizer[5], which was how we applied it for our purposes, logarithmic regression can be employed to model the relationship between the frequency of specific terms (features) and a continuous dependent variable, such as a rating or sales figure. By transforming the target variable using a logarithmic scale, this method can effectively capture multiplicative relationships and improve predictions, especially in contexts where the response variable is skewed or exhibits a wide range of values.

Decision Trees

A standard Decision Tree model (also provided by SKLearn[28]) was trained on the CountVectoriser embeddings on the hospital text as a control test.

Decision trees [5] are a model used for both classification and regression tasks. They operate by recursively splitting the data into subsets based on feature values, creating a tree-like structure where each internal node represents a feature, each branch signifies a decision rule, and each leaf node indicates an outcome or predicted class. In the context of text processing, decision trees can leverage the numerical representations generated by techniques like Count Vectorization. This allows the model to capture the relationships and patterns within the data effectively, making it suitable for various NLP applications such as sentiment analysis, topic classification, and spam detection.

Decision trees benefit from a straightforward feature representation that allows for quick and interpretable decision-making. The model can identify the most significant words that contribute to class distinctions, effectively determining thresholds that separate classes based on the presence or absence of specific terms. This interpretability is particularly valuable in text analysis, where understanding the reasoning behind classifications can be critical for end-users. Further-more, decision trees are capable of handling both binary and multi-class classification problems, making them a robust choice for text classification tasks. However, their susceptibility to overfitting, especially with high-dimensional data typical of text, necessitates careful tuning and potential integration with techniques like pruning or ensemble methods to enhance generalization and predictive performance.

Large Language Models

Llama 1 And 3

The Llama family of models [38] [15], developed by Meta, are a collection of LLMs that emphasize efficiency and adaptability in natural language processing tasks. Launched initially with Llama 1, the series introduced a range of architectures that optimize for both performance and accessibility, enabling researchers and developers to leverage state-of-the-art capabilities without the prohibitive resource demands typically associated with LLMs. The subsequent versions have continued to refine the training processes, improving not only the contextual understanding and coherence of generated text but also enhancing fine-tuning capabilities across diverse applications.

The iterative enhancements within the Llama family focus on addressing the nuanced challenges of language generation, including contextual relevance, reduced bias, and ethical considerations in deployment. By implementing new training methods and dataset curation practices, Meta has positioned the Llama models as competitive alternatives in the crowded LLM space. Their open-access framework makes using their various sub-versions an appealing choice for all uses, including experimentation within the research community.

MedLlama3

With the machinery used for our experiment, Llama3 [15] was too large to be properly fine-tuned with a LoRA [18] of any significant resolution. In order to evaluate the performance of the major new model, we used the MedLlama3 fine-tune released by the Yonsei University Medical

AI Laboratory. This allows us to evaluate how much fine-tuning on general medical text conveyed a capacity for medical question answering.

Microsoft Phi

Microsoft Phi 2 [24] is an LLM model developed by Microsoft Corporation. It is smaller than other LLMs with similar performance metrics, consisting of only 2.7 billion parameters when most similarly-performing current models are built with 7 billion [15] parameters. Its smaller size meant that we could train a higher-rank (larger, higher-resolution) LoRA, meaning more detail and nuance in the information it learned from the hospital data.

Before fine-tuning Phi, we also tested its base performance evaluating Heart Failure likelihood in order to determine how much fine-tuning increased its prediction accuracy.

Phi was then tested again after 3 epochs of training at a learning rate of 2×10^{-7} on 80% of the hospital data.

CHAPTER 5: RESULTS

We use multiple common metrics to evaluate the performance of these models: Micro-Precision[23], Micro-Recall[23], Micro-F1[23], and Normalized Discounted Cumulative Gain[42] (nDCG). The formulation for these follows:

Assuming a number of total labels K and number of total documents N, $y_i,\ \hat{y}_i \in 0, 1^K$,

$$Precision_{Micro} = \frac{\sum_{k=1}^{K} \sum_{i=1}^{N} y_i^k \cdot \hat{y}_i^k}{\sum_{k=1}^{K} \sum_{i=1}^{N} \hat{y}_i^k}$$

$$Recall_{Micro} = \frac{\sum_{k=1}^{K} \sum_{i=1}^{N} y_i^k \cdot \hat{y}_i^k}{\sum_{k=1}^{K} \sum_{i=1}^{N} y_i^k}$$

$$F1_{Micro} = \frac{2 \cdot Precision_{Micro} \cdot Recall_{Micro}}{Precision_{Micro} + Recall_{Micro}}$$

$$Specificity_k = \frac{\sum_{i=1}^{N} (1 - y_i^k) \cdot (1 - \hat{y}_i^k)}{\sum_{i=1}^{N} y_i^k}$$

$$nDGC_k = \frac{DCG_k}{\sum_{l=1}^{min(k,||z||_0)} log(l+1)^{-1}}$$

for a given integer k and

$$DCG_k = \sum_{l \in r_k(\hat{z})} \frac{z_l}{log(l+1)}$$

The Precision of a classification is a measure of how many predictions of that class were incorrect, or the ”impurity” of those predictions. Specifically, this means precision in XMTC settings will be lower for a label if that label is over-assigned to documents on which it does not belong. In the case of cohort selection, it means that heart-failure-negative patients were given a higher confidence score than they should have been.

The Recall of a classification is a measure of how many documents with that label were not

Comparing AttentionXML performance with and without CorNet on EUR-Lex

Metric	N=1	N=3	N=5
micro-Precision	0.7759	0.5226	0.6332
micro-Precision w/CorNet	0.8227	0.6870	0.5711
micro-Recall	0.1464	0.3586	0.4933
micro-Recall w/CorNet	0.1553	0.3890	0.5390
micro-F1	0.2464	0.4579	0.5075
micro-F1 w/CorNet	0.2612	0.4967	0.5546

Table 4. AttentionXML performance metrics on EUR-Lex text

predicted to have said label. This means recall in XMTC settings will be lower for a label if that label is not assigned to documents on which it belongs. In the case of cohort selection, it means that heart-failure-positive patients were not given the high confidence score they should have been.

As Precision will be 1.0 if all documents are assigned no labels (as there are thusly no wrongly-assigned labels) and Recall will be 1.0 if all documents are assigned all labels (as there will be no documents missing labels they should have), F1 represents a ”compromise” between the two metrics in an attempt to calculate a more accurate rating of model performance.

XMTC

Re-creation of CorNet

The re-creations showed similar performance to the original paper[42] with precision differing from the original by no more than 2% (as can be seen in Table 4, which compares the performance of AttentionXML[46] when predicting the labels associated with documents in EUR-Lex[8] with and without the assistance of CorNet).

Figure 5 demonstrates the acceleration in loss convergence shown in the original CorNet paper[42]. The difference between CorNet and non-CorNet models differs depending on the backbone, but in all tests, CorNet models would reach convergence at least 5 epochs before the non-CorNet versions. With the addition of CorNet increasing training times by under one second per epoch, this reinforces the motivation for adding CorNet layers to models in XMTC settings by showing a reliable decrease in overall training time.

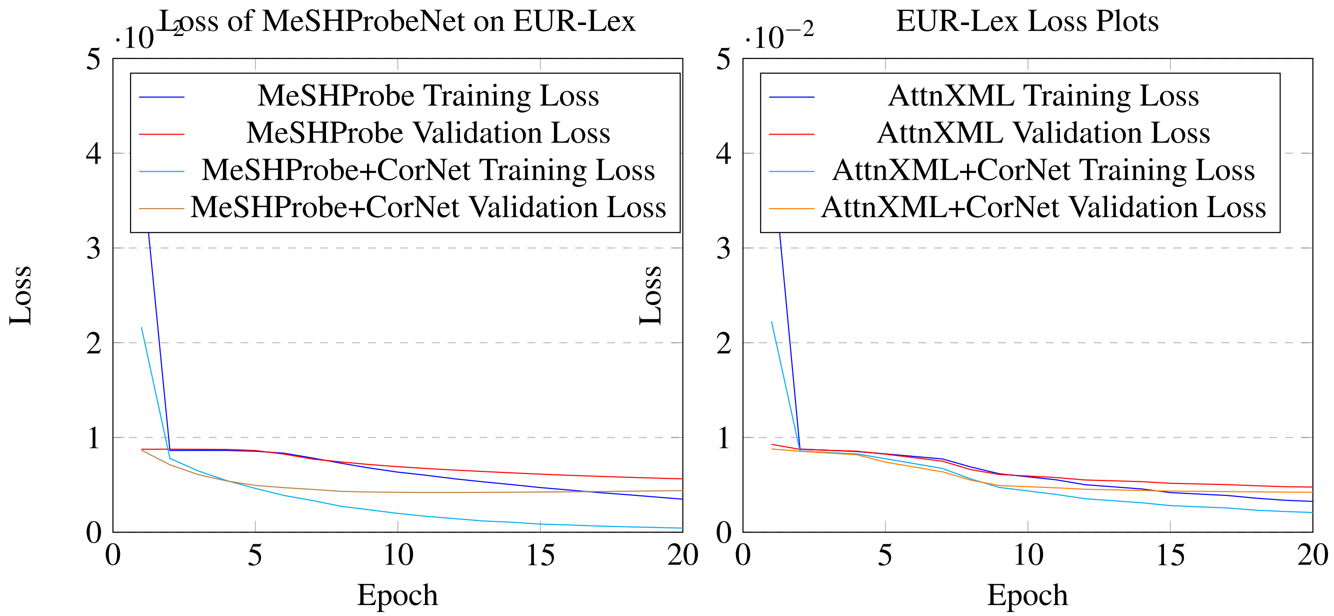


Figure 5. Training and Validation loss of models trained on EUR-Lex

Transfer To Medical Text

The results immediately showed that while performance varied significantly between different backbone models, the improvement provided by CorNet was not only still present, but consistent and even increased greater than previous datasets.

As shown in Table 4, backbone performance was much poorer on the MIMIC[1] dataset than the EUR-Lex dataset. However, the CorNet-augmented versions of the models (Table 4) show that CorNet retains its capacity to improve on the accuracy of predictions, even when the input predictions are less accurate.

Metric	N=1	N=3	N=5
AttentionXML			
Micro-Precision	0.5824	0.4769	0.4171
Micro-Precision (CorNet)	0.6245	0.5118	0.4498
Micro-Recall	0.0290	0.0712	0.1038
Micro-Recall (CorNet)	0.0311	0.0764	0.1120
Micro-F1	0.0552	0.1240	0.1663
Micro-F1 (CorNet)	0.0592	0.1330	0.1794
NDCG	0.5824	0.5013	0.4559
NDCG (CorNet)	0.6245	0.5382	0.4913
MeSHProbeNet			
Micro-Precision	0.5366	0.4466	0.3915
Micro-Precision (CorNet)	0.6689	0.5545	0.4822
Micro-Recall	0.0267	0.0667	0.0975
Micro-Recall (CorNet)	0.0333	0.0828	0.1201
Micro-F1	0.0509	0.1161	0.1561
Micro-F1 (CorNet)	0.0634	0.1441	0.1923
Micro-NDCG	0.5366	0.4686	0.4275
NDCG (CorNet)	0.6689	0.5829	0.5300

Table 5. XMTC model performance metrics on MIMIC-IV text

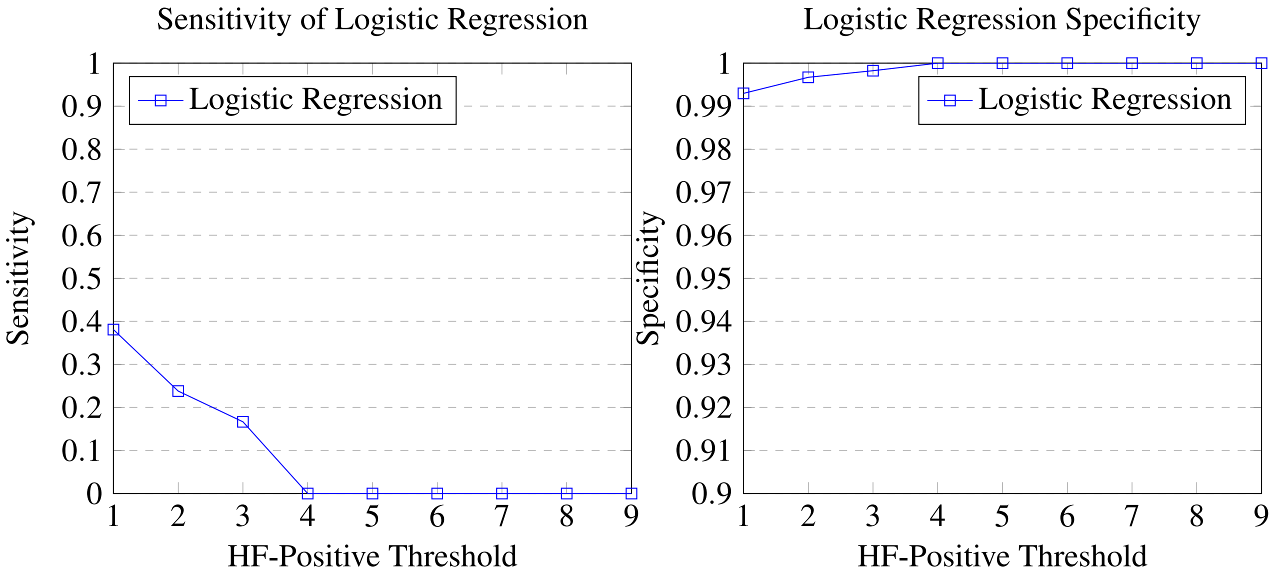


Figure 6. Sensitivity and Specificity of Logistic Regression after training on Hospital Data

Metric	Performance
F1	0.0129
Precision	0.5
Recall/Sensitivity	0.0065
Specificity	0.9998

Table 6. Decision Tree performance metrics on Hospital Data

Cohort Selection

Baseline Results

Logistic Regression

Logistic Regression[31] managed to perform much better than expected, successfully capturing 37.37% of heart-failure-positive patients at its lowest threshold, as shown in Figure 6. Performance quickly dropped, not managing to capture any patients

Decision Trees

Even with multiple trials, the Decision Tree quickly learned to classify all samples as HF-negative due to the overwhelming majority of the patients being negative for heart failure. Predictions were so skewed that the model would only predict patients as being heart-failure-positive when the threshold was at the absolute minimum value. This is why there is only one set of performance metrics in Table 6.

LLM Results

MedLlama3

Medllama 3 showed favorable results in terms of percentage of HF-positive patients recalled at the lowest threshold (Figure 7), however it only removed 39% of HF-negative patients from the output cohort. It had the highest recall of any baseline model we compared against, but with an F1 score not exceeding 0.15, its training on general medical text still leaves it falling short of models

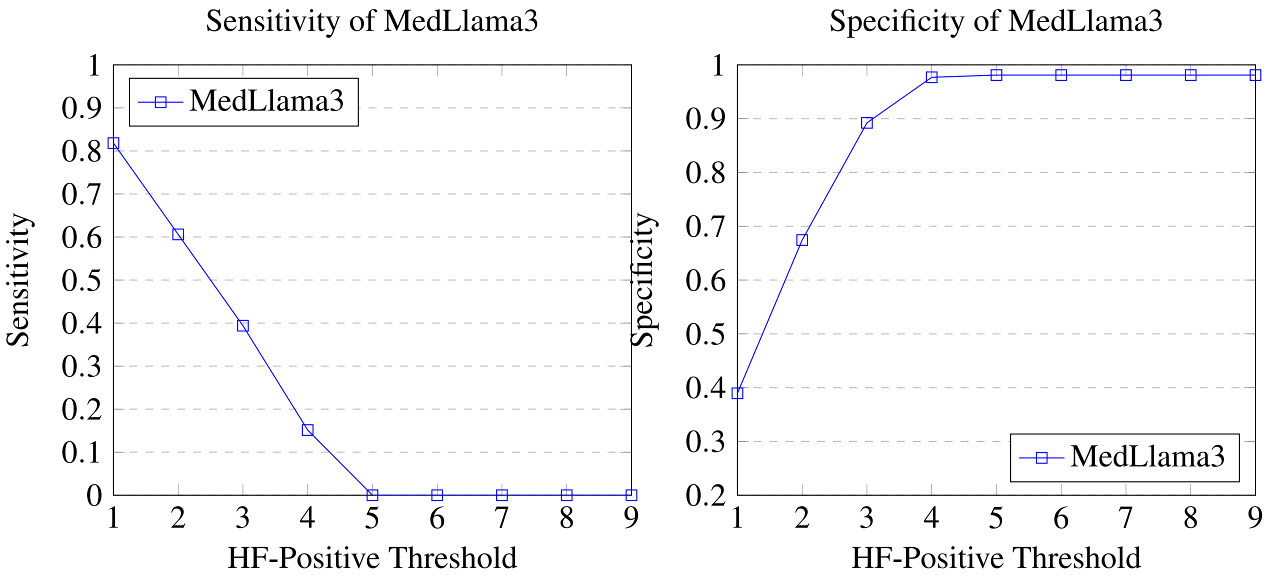


Figure 7. Sensitivity and Specificity of MedLlama3 after training on Hospital Data

Metric	Value @ Min Threshhold	Value @ Max Thredhhold
Precision	0.0329	0.1428
Recall/Sensitivity	0.8181	0.1515
F1	0.0633	0.1470
Specificity	0.3895	0.9769

Table 7. Performance of MedLlama3

trained on the pre-processed text and the provided heart-failure ratings.

Microsoft Phi

Microsoft Phi [24] was our best-performing model, capturing **75%** of all heart-failure-positive patients at its lowest threshold while removing **97%** of heart-failure-negative patients (Figure 8). If we model the amount of effort required by medical researchers to form a corpus as the number of heart-failure-negative patients they need to reject, researchers would normally need to manually reject 4552 out of the 4641 patients we removed from the training data for testing. Using this model, they would only need to manually remove the 145 negative patients that were wrongly included in the HF-positive set. This means that our fine-tuned Microsoft Phi 2 variant reduces the effort required by **97%** while losing 25% of HF-positive patients.

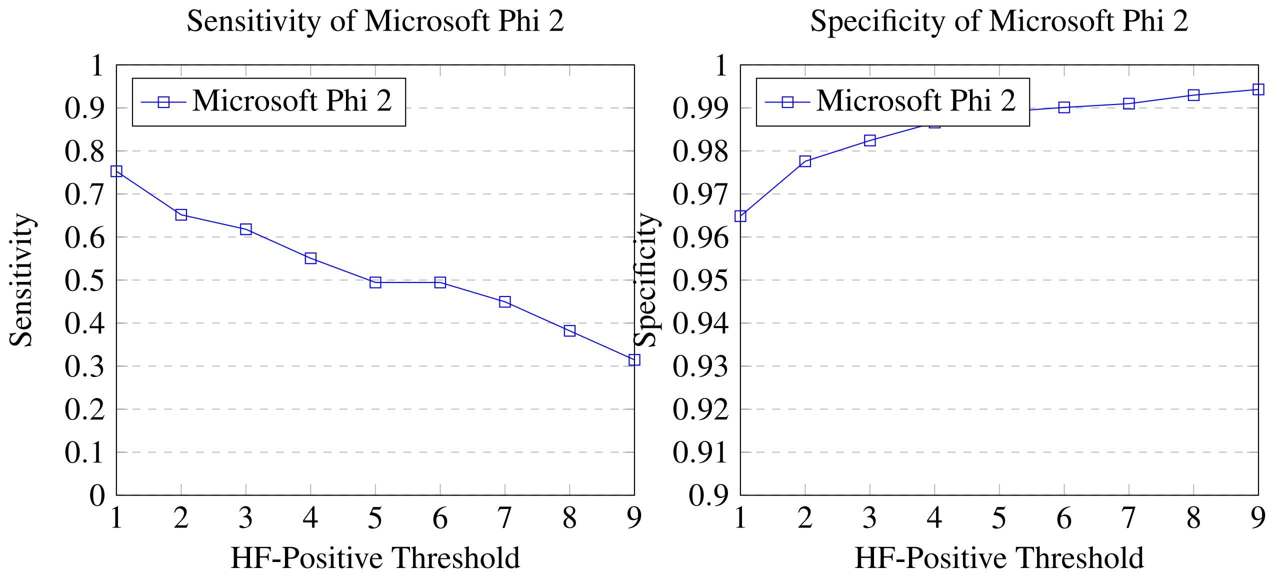


Figure 8. Sensitivity and Specificity of Microsoft Phi 2 after training on Hospital Data

Metric	Value @ Min Threshold	Value @ Max Threshold
F1	0.4240	0.3916
Precision	0.2951	0.5185
Recall/Sensitivity	0.7528	0.3146
Specificity	0.9648	0.9942

Table 8. Performance metrics of Microsoft Phi 2 after being fine-tuned on pre-processed hospital data

While a loss of 25% of patients that should be included is undesirable, a 97% reduction in time taken to collect a patient cohort means that a task which might require 10 hours would instead take roughly 30 minutes. Microsoft Phi 2 is a small model that was selected for its efficiency when working at the size we were able to accommodate on the hardware we had available to us. If further research was performed on a machine with a larger capacity for llm training, this process could be performed with a larger model without having to decrease the level of detail in the LoRA[18] trained. As seen by the difference between base medical models and their performance after fine-tuning, training on patient notes correlates to a greater number of HF-positive patients captured. This indicates that the 25% loss in HF-positive patients could be mitigated by further training and higher-capacity hardware.

Best-case performance on combinations of preprocessing steps

Character Linting	Part Filtering	Precision	Recall	F1
X	X	0.625	**0.515**	0.565
✓	X	0.563	0.485	0.521
X	✓	0.580	0.625	0.602
✓	✓	0.519	**0.753**	0.615

Table 9. Performance for each semantic knowledge preprocessing step combination using best-performing model (Phi 2)

Preprocessing Evaluation

We evaluated the performance of each step independently and in concert to determine how much each preprocessing task contributed to the overall improvement that resulted from our processing pipeline. Unfortunately the results were not as straightforward as expected. Notably, attempting to filter out documents deemed undesirable without properly filtering characters *reduced* the model's capacity to recall HF-positive patients. The predicate which partitions documents into used and undesirable sets performs better when working with a cleaned characterset, indicating that the preprocessing stages combined lend a greater performance increase than their individual performance impacts combined.

Comparing the performance with and without our preprocessing also showed a dramatic difference. We expected to an increase, however a 50% reduction in missed HF-positive patients was far more than we expected. One of the reasons we chose to use pre-trained large language models for this project was that they are understood to be somewhat robust against the sort of noise that our pre-processing removed. We designed our pipeline with the understanding that we were doing some of the language models' work for them in order to free up computational load to handle the complexity of the medical text. However, it was not expected that the measures taken would effectively halve the number of patients lost.

This implies that more work done on the pre-processing stage is necessary. Pre-processing was just one aspect of the pipeline we focused on in our experimentation, and time had to be balanced between making improvements to the pre-processing stage and other elements. In our opinion, the pre-processing stage seems like the best candidate for improvements in terms of the ratio between the potential degree of improvement and the effort necessary to achieve it.

CHAPTER 6: CONCLUSION

In conclusion, this thesis shows the value in further research into and eventual adoption of machine-learning-based NLP technologies in the medical sector. There are important and meaningful challenges that have a significant effect on the way medicine is practiced, and this thesis demonstrates that current machine-learning technology already has potential to be applied to these challenges.

We evaluated two different problem settings within the domain of Text Classification and showed that existing models and methods have the capacity to take advantage of the complex information found both in the text of documents as well as the information inherent in the correlations between labels. We also showed that the sophistication of current LLMs is not such that they are capable of learning the nuances of hospital database formatting more efficiently than a simplified version of the text, implying that whatever information exists within the highly-complex formatting of the hospital note text is not worth the computational effort needed to extract it, and that further research into distilling the large variety of characters and note types into a simpler encoding may further improve performance on all manor of downstream models.

While the architectures put forth in this paper are not performant enough on their own to warrant use in a real-world setting, they form a solid foundation for further research and are accurate enough that they could in their current state form a solid back-end for recommendation and double-checking systems.

Recommendations For Future Research

With the current renaissance in LLM research, any number of LLM-based innovations could be tested with our existing pipeline for Cohort Selection. However the largest hurdles we encountered in our research were the length of the patient notes, and the extensive characterset and vocabulary of their contents. Thus we recommend that further research start with context length improvements and more robust pre-processing techniques.

It would also be beneficial to see a prototype of the current models integrated into a suggestion system to showcase how they can be used to accelerate cohort selection and ICD Code assignment and evaluate to what degree human performance of each task can be accelerated with these models.

During the writing of this paper Microsoft has released a third version of their Phi series [24], Microsoft Phi 3 [2]. This represents another opportunity to explore how their improvements to minimizing a highly-performant model could be used for medical classification, and is a fairly trivial direction to explore further performance increases for CohortLang.

Summary

While not yet ready for real-world use in high-stakes medical settings, machine-learning NLP methods are already mature enough to benefit severe bottlenecks in hospital bureaucracy that are contributing to the congestion of the public health system.

Copies of fully-trained models regrettably cannot be provided as they have not been fully-evaluated for anonymity of patients whose data was used in training.

REFERENCES

[1] Johnson A., Bulgarelli L., Pollard T., et al. *MIMIC IV Clinical Database*. https://physionet.org/content/mimiciv/3.0/. 2023.

[2] Marah Abdin, Jyoti Aneja, Hany Awadalla, et al. *Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone*. 2024. arXiv: 2404.14219 [cs.CL]. URL: https://arxiv.org/abs/2404.14219.

[3] 01. AI, : Alex Young, et al. *Yi: Open Foundation Models by 01.AI*. 2024. arXiv: 2403.04652 [cs.CL]. URL: https://arxiv.org/abs/2403.04652.

[4] Akari Asai, Zeqiu Wu, Yizhong Wang, et al. *Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection*. 2023. arXiv: 2310.11511 [cs.CL]. URL: https://arxiv.org/abs/2310.11511.

[5] Sérgio Barreto, Ricardo Moura, Jonnathan Carvalho, et al. *Sentiment analysis in tweets: an assessment study from classical to modern text representation models*. 2021. arXiv: 2105.14373 [cs.AI]. URL: https://arxiv.org/abs/2105.14373.

[6] Asma Ben Abacha and Dina Demner-Fushman. "A question-entailment approach to question answering." In: *BMC Bioinformatics* 20.1 (Oct. 2019). ISSN: 1471-2105. DOI: 10.1186/s12859-019-3119-4. URL: http://dx.doi.org/10.1186/s12859-019-3119-4.

[7] Rishabh Bhardwaj and Soujanya Poria. *Language Model Unalignment: Parametric Red-Teaming to Expose Hidden Harms and Biases*. 2023. arXiv: 2310.14303 [cs.CL]. URL: https://arxiv.org/abs/2310.14303.

[8] K. Bhatia, K. Dahiya, H. Jain, et al. *The extreme classification repository: Multi-label datasets and code*. 2016. URL: http://manikvarma.org/downloads/XC/XMLRepository.html.

[9] Marcus Buckmann and Edward Hill. *Logistic Regression makes small LLMs strong and explainable tens-of-shot classifiers*. 2024. arXiv: 2408.03414 [cs.CL]. URL: https://arxiv.org/abs/2408.03414.

[10] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, and Ion Androutsopoulos. "Large-Scale Multi-Label Text Classification on (EU) Legislation." In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, 2019, pp. 6314–6322. DOI: 10.18653/v1/P19-1636. URL: https://www.aclweb.org/anthology/P19-1636.

[11] Junying Chen, Chi Gui, Anningzhe Gao, et al. *CoD, Towards an Interpretable Medical Agent using Chain of Diagnosis*. 2024. arXiv: 2407.13301 [cs.CL]. URL: https://arxiv.org/abs/2407.13301.

[12] Ying Chen, Peng Liu, and Chung Piaw Teo. *Regularised Text Logistic Regression: Key Word Detection and Sentiment Classification for Online Reviews*. 2020. arXiv: 2009.04591 [stat.ML]. URL: https://arxiv.org/abs/2009.04591.

[13] Davide Cifarelli, Leonardo Boiardi, and Alessandro Puppo. *Safurai 001: New Qualitative Approach for Code LLM Evaluation*. 2023. arXiv: 2309.11385 [cs.CL]. URL: https://arxiv.org/abs/2309.11385.

[14] *ControlNet with Stable Diffusion XL*. URL: https://huggingface.co/docs/diffusers/v0.20.0/en/api/pipelines/controlnet_sdxl.

[15] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, et al. *The Llama 3 Herd of Models*. 2024. arXiv: 2407.21783 [cs.AI]. URL: https://arxiv.org/abs/2407.21783.

[16] Shibo Hao, Yi Gu, Haotian Luo, et al. *LLM Reasoners: New Evaluation, Library, and Analysis of Step-by-Step Reasoning with Large Language Models*. 2024. arXiv: 2404.05221 [cs.CL]. URL: https://arxiv.org/abs/2404.05221.

[17] Z. S. Harris. "Distributional Structure." In: *Word* (1954).

[18] Edward J. Hu, Yelong Shen, Phillip Wallis, et al. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685 [cs.CL]. URL: https://arxiv.org/abs/2106.09685.

[19] Israt Jahan, Md Tahmid Rahman Laskar, Chun Peng, and Jimmy Huang. *A Comprehensive Evaluation of Large Language Models on Benchmark Biomedical Text Processing Tasks*. 2024. arXiv: 2310.04270 [cs.CL]. URL: https://arxiv.org/abs/2310.04270.

[20] A. Johnson, T. Pollard, and R. Mark. *MIMIC III Clinical Database*. https://physionet.org/content/mimiciii/1.4/. 2016.

[21] Taeyoon Kwon, Kai Tzu-iunn Ong, Dongjin Kang, et al. *Large Language Models are Clinical Reasoners: Reasoning-Aware Diagnosis Framework with Prompt-Generated Rationales*. 2024. arXiv: 2312.07399 [cs.CL]. URL: https://arxiv.org/abs/2312.07399.

[22] Black Forest Labs. *ControlNets, Depth, and Upscaler for FLUX.1-dev*. 2024. URL: https://fluxai.dev/blog/tutorial/2024-09/29-controlnets-depth-upscaler-flux1-dev.

[23] H. M. and Sulaiman M.N. "A Review On Evaluation Metrics For Data Classification Evaluations." In: *International Journal of Data Mining and Knowledge Management Process* 5 (2015), pp. 01–11. URL: https://api.semanticscholar.org/CorpusID:61877559.

[24] Microsoft. URL: https://huggingface.co/microsoft/phi-2.

[25] Aditi Mishra, Sajjadur Rahman, Hannah Kim, et al. *Characterizing Large Language Models as Rationalizers of Knowledge-intensive Tasks*. 2024. arXiv: 2311.05085 [cs.CL]. URL: https://arxiv.org/abs/2311.05085.

[26] Harsha Nori, Nicholas King, Scott Mayer McKinney, et al. *Capabilities of GPT-4 on Medical Challenge Problems*. 2023. arXiv: 2303.13375 [cs.CL]. URL: https://arxiv.org/abs/2303.13375.

[27] World Health Orgnization. *International Statistical Classification of Diseases and Related Health Problems (ICD)*. URL: https://www.who.int/standards/classifications/classification-of-diseases.

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, et al. "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[29] Le Peng, Gaoxiang Luo, sicheng zhou, et al. *An In-Depth Evaluation of Federated Learning on Biomedical Natural Language Processing*. 2023. arXiv: 2307.11254 [cs.CL]. URL: https://arxiv.org/abs/2307.11254.

[30] Sara Pieri, Sahal Shaji Mullappilly, Fahad Shahbaz Khan, et al. *BiMediX: Bilingual Medical Mixture of Experts LLM*. 2024. arXiv: 2402.13253 [cs.CL]. URL: https://arxiv.org/abs/2402.13253.

[31] S. Pradeepa, Elizabeth Jomy, S. Vimal, et al. "transforming review text classification with hypergraphs attention layer and logistic regression." In: *Scientific Reports* (). URL: https://doi.org/10.1038/s41598-024-70565-6.

[32] *PubMed*. URL: https://pubmed.ncbi.nlm.nih.gov/.

[33] Robin M. Schmidt. *Recurrent Neural Networks (RNNs): A gentle Introduction and Overview*. 2019. arXiv: 1912.05911 [cs.LG]. URL: https://arxiv.org/abs/1912.05911.

[34] Omar Shaikh, Jon Saad-Falcon, Austin P Wright, et al. *EnergyVis: Interactively Tracking and Exploring Energy Consumption for ML Models*. 2021. arXiv: 2103.16435 [cs.LG]. URL: https://arxiv.org/abs/2103.16435.

[35] Karan Singhal, Shekoofeh Azizi, Tao Tu, et al. *Large Language Models Encode Clinical Knowledge*. 2022. arXiv: 2212.13138 [cs.CL]. URL: https://arxiv.org/abs/2212.13138.

[36] Issey Sukeda. *Development and bilingual evaluation of Japanese medical large language model within reasonably low computational resources*. 2024. arXiv: 2409.11783 [cs.CL]. URL: https://arxiv.org/abs/2409.11783.

[37] Klaudia Thellmann, Bernhard Stadler, Michael Fromm, et al. *Towards Multilingual LLM Evaluation for European Languages*. 2024. arXiv: 2410.08928 [cs.CL]. URL: https://arxiv.org/abs/2410.08928.

[38] Hugo Touvron, Thibaut Lavril, Gautier Izacard, et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971 [cs.CL]. URL: https://arxiv.org/abs/2302.13971.

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762.

[40] Yidong Wang, Zhuohao Yu, Zhengran Zeng, et al. *PandaLM: An Automatic Evaluation Benchmark for LLM Instruction Tuning Optimization*. 2024. arXiv: 2306.05087 [cs.CL]. URL: https://arxiv.org/abs/2306.05087.

[41] Zijie J. Wang, Alex Kale, Harsha Nori, et al. "Interpretability, Then What- Editing Machine Learning Models to Reflect Human Knowledge and Values." In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Vol. 16. KDD '22. ACM, Aug. 2022, pp. 4132–4142. DOI: 10.1145/3534678.3539074. URL: http://dx.doi.org/10.1145/3534678.3539074.

[42] Guangxu Xun, Kishlay Jha, Jianhui Sun, and Aidong Zhang. *Correlation Networks for Extreme Multi-label Text Classification*. 2020.

[43] Guangxu Xun, Kishlay Jha, Ye Yuan, et al. "MeSHProbeNet: a self-attentive probe net for MeSH indexing." In: *Bioinformatics* 35.19 (Mar. 2019), pp. 3794–3802. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btz142. eprint: https://academic.oup.com/bioinformatics/article-pdf/35/19/3794/48975812/bioinformatics_35_19_3794.pdf. URL: https://doi.org/10.1093/bioinformatics/btz142.

[44] Hu Ye, Jun Zhang, Sibo Liu, et al. "IP-Adapter: Text Compatible Image Prompt Adapter for Text-to-Image Diffusion Models." In: (2023).

[45] Qichen Ye, Junling Liu, Dading Chong, et al. *Qilin-Med: Multi-stage Knowledge Injection Advanced Medical Large Language Model*. 2024. arXiv: 2310.09089 [cs.CL]. URL: https://arxiv.org/abs/2310.09089.

[46] Ronghui You, Zihan Zhang, Ziye Wang, et al. *AttentionXML: Label Tree-based Attention-Aware Deep Model for High-Performance Extreme Multi-Label Text Classification*. 2019. arXiv: 1811.01727 [cs.CL]. URL: https://arxiv.org/abs/1811.01727.

[47] Qingbin Zeng, Qinglong Yang, Shunan Dong, et al. *Perceive, Reflect, and Plan: Designing LLM Agent for Goal-Directed City Navigation without Instructions*. 2024. arXiv: 2408.04168 [cs.AI]. URL: https://arxiv.org/abs/2408.04168.

[48] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. *Adding Conditional Control to Text-to-Image Diffusion Models*. 2023.

[49] Yan Zhuang, Qi Liu, Yuting Ning, et al. *From Static Benchmarks to Adaptive Testing: Psychometrics in AI Evaluation*. 2024. arXiv: 2306.10512 [cs.CL]. URL: https://arxiv.org/abs/2306.10512.