Enhancing Medical Record Comprehensibility: Using Large Language Models to Produce Simplified Narratives of Image Reports in Electronic Medical Data

XUMOU ZHANG

Supervisor: Prof. Jinman Kim
Associate Supervisor: Prof. Adam Dunn

A thesis submitted in fulfilment of
the requirements for the degree of
Master of Philosophy

School of Computer Science
Faculty of Engineering
The University of Sydney
Australia

12 March 2025

Statement of Originality

I declare that the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes.

Abstract

The motivation of this thesis consists of three areas: clinical domain adaptation, long-context adaptation, and local language model implementation. This thesis investigates the use of language models in clinical applications where input documents are long and questions can be complex and require advanced reasoning. The research aims and objectives in this thesis contain two parts: one is to find and evaluate the appropriate solution that enhances the model performance under clinical settings; the other one is to find the solution to modify the models for better performance under this circumstance. Two approaches are developed to address the problem, and experimental results show that the approaches address the key challenges related to long contexts, complex questions, and the need to capture domain knowledge. Overall, this thesis aims to explore the possibility of replacing cloud-based large language models with local inference language models in specific professional domains.

In the first study, the RAPTOR framework extends a language model's ability to make sense of local and global information from long documents with its unique hierarchical tree structure datastore. The approach may be beneficial where cloud-based large language models (e.g. GPT-4o) cannot be used due to data privacy or reproducibility issues. Specifically, RAPTOR can be tailored to address clinical tasks including extracting critical patient information and summarizing clinical notes from long documents. The aim was to benchmark and optimize the RAPTOR framework using language models that can be implemented locally, making it more practical and accessible for users who are concerned about data privacy and want to use on-device models for application only. The study tested the RAPTOR framework on the QuALITY dataset and a novel Clinical Trial Question and Answer (CTQA) dataset, drawn from ClinicalTrials.gov, a registry that includes information about the design of more than 500,000 clinical trials. Experiments compared RAPTOR across multiple configurations, on simple questions and complex questions, two language models, four embedding models, and three chunking strategies. This study also introduced and included a novel modified

semantic and deep learning semantic chunking strategy that allows the text to be split based on the semantic text embedding dynamically. The evaluation experiment used the GPT-4o model as the baseline comparison to illustrate the larger LLMs. The performance results show that RAPTOR can outperform the GPT-4o model in complex questions of the CTQA dataset with the (smaller) local language model but not for the QuALITY dataset, suggesting that not all question complexity is the same. With the modified configurations in the study, the RAPTOR framework may be a practical solution for the long context problem even when constrained to locally implemented language models.

In the second study, I developed and tested an optimized language model that uses a continual pre-training process to incorporate domain knowledge with a Llama-3.1-8B language model, with a novelly collected, organized and preprocessed Clinical Trial registration dataset called CiTi. The dataset contains 358870 preprocessed clinical trial registration reports and 1401401 related publication abstracts. This study aimed to develop a language model that adapts clinical trial registration data as its specialty domain and has more understanding of this domain than general large language models. To demonstrate the performance improvement, I compared the original Llama-3.1-8B model, the CTLlama-8B-demo model, and the fully trained CTLlama-8B model on three publicly available medical and health domain datasets. The result shows that the continual pre-training process has improved the model's performance on average from 0.317 to 0.432 in terms of F1 score in the evaluation, indicating the importance of continual pre-training in building up a domain-specific model.

The two solutions evaluated in this thesis show that with the updated configuration, it is possible to achieve state-of-the-art performance using locally implemented language models. Future research should consider how specific configurations or auto-configurations better suit simple and complex questions.

Acknowledgements

I want to take this opportunity to thank Professor Jinman Kim and Professor Adam Dunn for all the help and support in my MPhil period and in the future. They are both terrific researchers on their own. And even more unnatural is that they are also excellent educational professionals and give me so much highly efficient support and guidance during my research studies and daily work.

I also want to thank my faithful partner Xiaotong Yu, who put up with my attitude when I was nervous during the research periods. It was hard for anyone to take me as a partner, but she did it and accepted me on both positive and negative sides.

Contents

List of Figures

CHAPTER 1

Introduction

1.1 Background

Electronic medical record data have brought up a significant shift in how healthcare professionals document, access, and analyze patient information. Studies have demonstrated that electronic medical record improves data management, communication, and healthcare delivery between doctors and patients. While electronic medical record data is essential for clinical care and provides valuable information, they often pose challenges for research due to regional privacy regulations like HIPAA and GDPR [107, 28], which limit access and require extensive de-identification [14, 20].

ClinicalTrials.gov is a registry of over 500,000 clinical trials [46]. It has commonalities with electronic medical record data in that it includes medical jargon and requires contextual medical knowledge to make sense of and answer questions about its content. It is an ideal dataset because ClinicalTrials.gov registrations are public records of trials that are recorded before the actual clinical trials start, and many also include summary results data after the trials are finished. The purpose of ClinicalTrials.gov is to ensure the transparency, accountability and accessibility of clinical trials. Compared to electronic medical record data, both systems store information critical to healthcare. Electronic medical records capture patient histories to support clinical care, while clinical trial registrations document study protocols, interventions, and outcomes to ensure research transparency and reproducibility.

While both electronic medical record and clinical trial registration data play crucial roles in the healthcare area, they also share similar challenges. Both electronic medical record and

clinical trial registration contain a large amount of information in a single report, and each report is fairly long in terms of size. Studies show that the length of the electronic medical record data has continued increasing over the years [88] at different rates for different kinds of notes. The average length of clinical notes in the MIMIC-IV dataset is stated as 2267 tokens [2]. Studies also indicate that 22% of patients did not know how to take medication and 48% of patients with low health literacy, which could cause poor health outcomes and health service costs [34]. Similar issues occur in the clinical trial registration data, where each report presented in the ClinicalTrials.gov is multiple pages long with contrasting information. Although the clinical trial registration data is stated as structured data, many columns are just text descriptions that need to be further extracted. It is important to find an appropriate solution to support and guide people in finding needed information from such text reports.

Recently, there has been an important breakthrough in the field of language models. Researchers find that by dramatically increasing the model size to a level of billions, the GPT model [82] seems to have emergent abilities of generalizability [111]. The resulting models are called large language models, and those models have demonstrated remarkable capabilities for summarizing text, information extraction, sentence completion and other tasks to process text information [67]. There have been some studies in the medical field as well. Early studies in the medical field are also promising. For example, researchers have evaluated the GPT-4 in the medical examination in the study, and the result shows it can reach the passing score of the United States Medical Licensing Examination [70]. Another study highlights the potential healthcare applications such as dialogue summarization, electronic health record generation, scientific research and more [108].

However, LLM integration in both clinical practice and academic research is not without challenges. One major issue is the model hallucination, where the LLM would generate incorrect or misleading information with blinded confidence. This is particularly concerning in clinical and academic areas, where false information can lead to catastrophic results. Another obvious challenge would be related to data privacy. The healthcare industry places a high priority on protecting patient information and often requires that the data can only be interpreted within local or protected environments. The related research databases are more

diverse. For example, the MIMIC database clearly states the guidelines for building models and creating datasets with strict privacy requirements. In a broader view, many countries and regions worldwide have proposed regional data protection regulations such as the GDPR [107] and HIPAA [28] to forbid and protect personal data and domain data from leaking out. Many companies worldwide also implement internal LAN networks to prevent data leakage and enhance security by limiting external access to sensitive information [29, 68].

Due to data security concerns and regional regulation constraints related to real patient data, the research community have not yet been able to fully leverage the model's performance in clinical applications. One recent study raised concerns about the LLM evaluation studies in the medical domain because only 5% of the studies implemented real patient care data for the evaluation [7]. This limitation will lead to bias and fairness issues; impacting the actual performance of LLM applications in clinical settings. Thus, implementing the local language model without privacy concerns can be a valuable opportunity to unlock the full potential of real patient data.

1.2 Research Contribution

The Large Language Models (LLMs) are growing to show remarkable capabilities in interpreting input context, processing complex logic and generating appropriate responses to support healthcare professionals. Cloud-based models such as GPT-4o and Gemini have gained significant attention because of their ability to solve problems across many application domains. [70, 89]. However, these models are not ideal solutions for health and medical domains with strict data privacy requirements and a focus on safety.

To address these gaps, I conducted an in-depth investigation of the local language models and built a pipeline based on the Retrieval-Augmented Generation (RAG)-based framework called Recursive Abstractive Processing for Tree-Organized Retrieval [90] (RAPTOR).

The RAPTOR framework constructed its datastore with a unique hierarchical tree where each node summarises its child nodes. It could show large improvements in questions that involve

complex reasoning because it could extract different summarized levels of information from the original document.

The experimental evaluation involved two datasets. The first is the QuALITY dataset, a multiple-choice question-answering dataset containing simple and complex questions in the normal literature domain, with an average length of around 5000 tokens [75]. It labelled the question as complex one if the human annotator cannot answer the question within 45 seconds. The second is the novel CTQA dataset. The data was collected from the ClinicalTrials.gov website, and the questions were constructed similarly to the QuALITY dataset format with simple and complex questions, where the question would be labelled as complex if it requires information retrieval and advanced reasoning. The data length in this dataset is ranging from 2000 to 6000 tokens. Both datasets divided the questions into simple and complex questions, aiming to test the model performance using different standards. This work could be a solution for healthcare professionals to use LLM-based applications in the local environment without relying on cloud-based servers. Besides that, I explore the practical performance and limitations of the local language models and find the potential solution to fill the gaps.

In another direction, I've also conducted training on a local language model, Llama-3.1-8B [17], and converted the model into a domain-specific LLM for clinical trial registration data that could be inferred locally. My contributions in the research include:

- **Benchmarking the RAPTOR framework on local environment:** A key limitation of the RAPTOR framework is that it had not been tested with local language model configurations, despite its strong performance on complex questions in previous evaluations. To address this gap, I've conducted a detailed evaluation of the RAPTOR framework on consumer-level hardware and the local language model, with a public-available QuALITY dataset and a novel CTQA dataset collected from ClinicalTrials.gov, demonstrating that it can perform effectively within the constraints of a local environment, offering meaningful and accurate text outputs to the preset questions from the dataset, without requiring extensive computational resources.
- **Optimizing semantic chunking strategy for RAPTOR framework:** The original RAPTOR framework employed a naive text chunking strategy, which limited its

performance by failing to capture the full semantic meaning in text chunks. To address this limitation, this research also developed a new semantic chunking strategy called recursive moving percentile semantic chunking, which allows the text to be split based on the semantic text embedding dynamically, to optimize the pre-processing step in the RAPTOR framework. The new chunking strategy has shown better performance than the naive text chunking strategy in all configurations tested across different testing datasets.

- **Developing New Model for Clinical Trial Domain Applications:** Although generalized large language models have demonstrated remarkable capabilities, their performance in clinical domains remains limited due to the lack of domain knowledge. To construct better domain uses of local language models, I've developed a new domain-specific model called CTLlama-8B, on top of the successful Llama-3.1-8B model by conducting a detailed continual pre-training process, to construct the domain-specific model with a better understanding of the clinical trial registration context. The training experiment used the CiTi dataset, which I novelly collected, organized, and preprocessed from ClinicalTrials.gov and PubMed for the continual pre-training process. I demonstrated the evaluation process using 3 datasets related to medicine and health domains. The result showed that the new model can outperform the original Llama-3.1-8B model in all 3 datasets.

1.3 Thesis Overview

This thesis explores the implementation and modification of using the RAPTOR framework with local language models in local environment, intending to enhance how the local language models can be practically applied in the healthcare domain. The work addresses both the opportunities and challenges that come with integrating these advanced technologies into everyday clinical practice.

Chapter 2 provides a comprehensive literature review on integrating LLMs into healthcare applications and academic research environments. It discusses the specific requirements

and reasons for deploying LLMs in healthcare, including the strategy of optimizing the current model using prompt engineering and RAG frameworks and modifying the model with continual pre-training and fine-tuning processes for better domain-specific task performance.

Chapter 3 dives into a RAG-based framework called RAPTOR. This framework could extend the language model's ability to understand long context documents efficiently with local and global information with its unique hierarchical tree structure datastore. In this chapter, I've conducted detailed benchmarking experiments to see how the RAPTOR framework could perform in local, cloud and hybrid settings and where the gap would be. I've also proposed a new semantic chunking strategy to fill the gap inside the RAPTOR framework. The result shows that the new chunking strategy improved the performance of the RAPTOR framework in both the QuALITY and CTQA datasets compared to the original one.

Chapter 4 addresses the challenges and solutions related to working with clinical trial registration data from ClinicalTrials.gov. In this chapter, I've demonstrated a new fine-tuned model called CTLlama, which contains complete domain knowledge from over 500,000 reports extracted from the website API and their related publications extracted from PubMed API. It offers insights into how the domain-specific model performs compared to the generalized ones with the minimum downstream task training process.

Chapter 5 explores the future potential of LLMs in clinical settings, considering ongoing developments and future directions. It also presents a detailed conclusion of the studies conducted, supported by evaluations and analytical results, providing a clear understanding of the impact and implications of this research.

CHAPTER 2

Literature review

2.1 Introduction to Large Language Models

The current mainstream large language models are developed based on the generative pre-trained transformer (GPT) model architecture [82]. The study has demonstrated that the GPT model architecture can offer serval benefits, including parallel processing, contextual understanding and scalability. Researchers found that the GPT-based models can "emerge" with unexpected performance when they scale up the model parameter size above 10 billion [8], and there are more follow-up studies discussing the "scaling-law" behind it [97, 42]. In summary, the studies show that the model could act surprisingly intelligent when the model size is large enough.

Thus, the development of the GPT-based language models encounters two obstacles: These language models are required to be trained using the really large scale of the collected text datasets, and these models are required to be trained on expensive hardware due to the expanding model size. For example, Llama models [104] were trained on an astonishing 1.4T tokens of unlabeled text data, and the most recent Llama-3 models [17] have kept pushing this limit to 15T tokens. The report also states that the Llama-3 model training requires 16,000 pieces of H100 GPUs.

LLMs are trained on publicly available content like websites, social media, and textbooks [58]. While there is a range of early examples of the use of modern LLMs in clinical tasks [70], there is no clear evidence that general-purpose language models can meet the requirements for solving complex clinical tasks like clinical decision-making and information retrieval and

summarization without modification, and researchers have expressed the safety concerns for the LLMs in the clinical domain [56, 32]. The potential solutions to such questions can be divided into 3 directions: prompt engineering, fine-tuning, and continual pre-training.

2.2 Prompt Engineering for Domain Adaptation

The prompt engineering is done by optimizing input prompts for the language model [8]. It serves as a straightforward method to adjust the language model for optimizing the performance in targeted domains and downstream tasks. Because it does not require adjustment to the model parameters, it potentially avoids the high cost and makes it a cost-effective method in model domain adaptation.

The general goal of prompt engineering is to provide clear, simple, and focused prompts to utilize the model's performance in downstream tasks. The original GPT-3 study suggested that, by training the model with the large-scale dataset, the model will emerge with the ability called "in-context learning" without modifying the model parameters. It indicates that the model will show extensive performance gain in the downstream tasks using examples in the input prompt. Other studies also contributed to the prompt engineering area and show that LLMs can greatly benefit from prompt-provided information.

As the GPT-3 paper states, the LLMs have a really impressive ability for in-context few-shot learning. However, it is time-consuming to build up the middle steps, and the traditional prompts do not perform well in the STEM-related questions and the tasks that require extensive logical reasoning. Chain-of-thought reasoning [110] (CoT), was born to fill this gap until today. It suggested that the users should add the description for the reasoning workflows in the few-shot examples. Each example should contain 3 parts: the question, the reasoning chain and the final answer. The model will emulate the given examples and generate its own version of the reasoning chain to answer the initial question. The whole idea is to simulate how humans would think and answer when facing a complex question. This study also pointed out that, the ability of CoT is directly related to the model size. The models contain more

than 100B parameters could have significant performance gain compared to the traditional few-shot prompting.

The follow-up study [45] dived deep into the CoT approach and proposed the zero-shot CoT. It is a really simple pipeline. By just saying, "Let's think step by step", the model would generate some thinking and reasoning process, encouraging the model to generate responses with more rationales. The author also tried on different sentences for the direct comparison on the GPT-3, and turns out the above one is the most optimal one for the GPT-3 model.

The ReAct framework [121] is more complex. Based on the few-shot CoT, it divides the few-shot example into 2 sections: Reasoning and Action (an additional section could be observation). By incorporating external database search results like Google search, the related reasoning is produced, and then action is taken based on the generated reasoning. The example prompt template could be: "Thought: xxx Action: Search[xxx] Observation: xxx". This idea demonstrates how LLMs could be working in the area of active AI assistants or robotics.

Microsoft has also proposed a prompt engineering framework called self-verification [27], which targets the clinical information extraction task. Based on the idea of few-shot prompting and CoT, it divides the task pipeline into 4 steps: original extraction, omission, evidence and prune. Each step will use the result from the last step to deliver the target result in the current step, and the model will eventually generate the final result after all steps are completed. Like the ReAct framework, the self-verification framework also shows the trade-off between the computational cost and targeting task performance. The existing studies in ReAct and self-verification explain that the researchers anticipate the model inference cost will decrease to an affordable level in the future.

2.3 Retrieval-Augmented Generation in Professional Domains

It is clear that the LLMs can perform in-context learning and generate better responses based on the external information from input [45], but this simple approach is not optimized when the users are trying to adapt more information externally.

The Retrieval-Augmented Generation (RAG) framework [52] takes a step further in this direction. Besides directly injecting knowledge into the model and modifying the parameters, RAG framework demonstrated an external solution to the area. The essential idea of RAG was proposed by Meta in early 2021, where the model could improve its performance using the retrieved related information in the external knowledge base. This idea suggests that the developers will not need to train the model from scratch for each task. They could attach an external knowledge base to the model, and improve the task accuracy. This idea is suitable for tasks that require a large amount of knowledge.

In conclusion, the RAG framework contains 2 steps:

- Using encoder models like SentenceBERT [84] to find the related documents or knowledge.
- Using the language model to generate the response based on the related context.

Previous study has defined this kind of RAG framework as naive RAG framework [25], distinguishing it from more advanced RAG-based frameworks.

The benefits of the naive RAG framework are as follows. One is to improve the accuracy of the results and reduce false information. Since it relies on the external knowledge base, it could be quickly swapped in and out with new information. Also since the model generates responses based on the context in the naive RAG framework, it could provide interoperability to the user. Therefore, it could give more control regarding data safety and privacy concerns.

There are also some concerns about it. Since the naive RAG framework does not change the model parameter, it cannot adapt to the new domain immediately. For example, if the

model is only been trained on the English contents, then it would be really hard for the model to understand French or Spanish documents. The researchers also find that the naive RAG frameworks could suffer from unrelated or incomplete context information while retrieving the knowledge base, leading to incorrect or unrelated responses. The real situation could be even more complex. For example, the data in the knowledge base could not be in the question-and-answering format at all and would require pre-processing, chunking and other processes to make it work. This leads to the advanced RAG frameworks.

The previous study have stated that the advanced RAG frameworks are designed to address the limitations of the naive RAG framework [25]. The improvements of advanced RAG frameworks generally follow the 2 directions: pre-processing and retrieval process.

The naive RAG framework only contains the most basic pre-processing processes. It splits the raw documents into text chunks with a maximum number of characters or tokens. This kind of method could cause incomplete semantic meaning in each text chunk. Larger chunks would contain more complete information, but they also contain more noise information that reduces the model performance in the downstream tasks. The ideal number of chunking lengths could vary for different domain tasks. One step up from that would be to split the texts via special characters like the newline character with the maximum character/token limit. However, it still cannot ensure the completeness of the information inside each text chunk.

To solve the issues, one study proposed a specific chunking strategy for the financial domain [122]. It introduces a method that chunks financial reports based on structural elements (e.g., tables, titles, narrative texts) rather than a standard paragraph or token-based naive chunking. This approach shows better results in document understanding and question-answering tasks for the RAG tasks of financial reports. The problem with this study is that it only focuses on the financial domain, and the proposed method clearly involves many rule-based processes. Additionally, a similar study has also been conducted in the legal domain [22], where the chunking process can be utilized based on the general structure of the legal document. These studies are focused on the chunking strategies in one particular type of text.

To chunk information semantically in a more general domain or within multiple domains, the developers begin to adapt the text embedding models into the chunking process called semantic chunking [1]. The essential idea is to find the "difference" breakpoints between sentences by comparing the text embeddings of each adjacent sentence, and the breakpoint occurs when the cosine distance goes over a human-set threshold value. This allows the chunking strategy to produce more precise text chunks semantically. The concern is that the human-set unified threshold value lacks scientific support, and the different text embedding models would have huge performance gaps. No studies are trying to solve this issue directly, but similar studies are in the text segmentation area. One study proposes the idea of using a BERT-based embedding model to convert sentences into sentence embeddings, then using those as inputs to a transformer model to identify breakpoints in the sentence embedding [61]. Another study proposes using a moving window inside the BERT model structure [124]. A PoNet-based model [99] extended the context length from 512 tokens to 4096 tokens.

The retrieval process is straightforward in the naive RAG framework. It compares the user query with each text data row stored in the knowledge base using a text embedding model and cosine distance calculation and keeps the most similar ones for reference. The goal of the studies in this direction generally focuses on retrieving the correct information that directly helps answer the question. A recent study proposed the CRAG framework [119]. It attached a lightweight T5 model after the retrieval process to identify whether the retrieved information is related to the question or not; and if the model is unsatisfied with the information, it would attempt to retrieve information from external data sources like Google Search via agent-based framework. Another study introduced the self-RAG framework [5] in a similar but different direction. Instead of taking all retrieved information simultaneously, the self-RAG framework takes one retrieved information per time and generates a sub-response, then asks an evaluator model to "critique" each piece of information; the model will then generate responses based on the best subset of all the retrieved information based on the "critique" evaluation.

The RAPTOR framework [90] has demonstrated a hybrid direction of utilizing both the chunking strategy and the retrieval process. It formulates a tree structure datastore with serval steps in the pre-processing. First, it splits the raw documents into text chunks (nodes) with

the naive chunking strategy. Then, it clusters similar nodes via text embeddings, summarizes the nodes within the cluster and produces a new parent text node. By repeating this process recursively, the RAPTOR framework will eventually get a root node with highly summarized text of all nodes. It would be obvious to see that the nodes in the higher layers will have more contrast summaries to the document information, whereas the nodes in the lower layers will contain more detailed information. This allows the RAG framework to retrieve different levels of information from different aspects and complete the context in the retrieval process.

The recent popular Graph-RAG framework [18] shows a similar story as well. It utilized the knowledge base by formulating a knowledge graph using LLM automatically, so it could retrieve highly relevant information in the retrieval process.

2.4 Continual Pre-training of LLMs on Domain-specific Data

Studies indicate that when the continual post-training process again cannot fulfil the requirement, we should consider the continual pre-training to the model as the starter point [115, 76, 106]. This is significantly important for the clinical domain models because normal generalized models are not trained on the clinical text, and therefore, there is a lack of knowledge, which can cause a decrease in the model performance in the clinical domain, or the tokenizer of the model would require extra tokens to formulate the clinical-domain vocabulary, which could cause unwanted additional computational cost.

The domain knowledge and information in the clinical domain tend to be relatively long in the context size. For example, the previous study indicated that the median of the patient record length was around 4300 words, whereas the mean in their collected dataset is 16826 words [95]. Another study also states that the average length of their EHR notes is around 1200 words [37]. These numbers indicated that the models should be capable of processing long text data to be implemented in clinical applications.

Large language models are not born to be capable of processing long inputs. For example, the original Llama models [104] are only capable of dealing with the text input within 2048 tokens, which is already really impressive at the time. The successors of the Llama models, the Llama-2 series [103] and the Llama-3 series [17], are expanding the context length for more and more urgent needs.

2.5 Optimizing LLMs with Supervised Fine-Tuning

As suggested in the GPT-3 paper, only a select few large-scale generative LLMs can fully harness the power of prompt engineering. In domain-specific studies, many works have expressed concerns, noting that generative LLMs like GPT-4o or the Gemini models do not perform optimally in professional medical and clinical settings [64, 38, 87]. In cases where prompt engineering techniques fail to meet the needs of domain tasks, model post-training becomes essential. Unlike prompt engineering, continual post-training of LLMs involves supervised fine-tuning and reinforcement learning, aiming to improve model performance by adjusting the model's parameters. While prompt engineering may represent the upper limit of LLMs, post-training can be viewed as a way to ensure a more reliable baseline.

Supervised fine-tuning (SFT) is the most common post-training process for the pre-trained models on the labelled dataset. In the LLM area, we often use instruction fine-tuning [112] to refer to the specific supervised fine-tuning process for the LLMs. This involves using questions and answers to group the training dataset and using special tokens from pre-trained base models to distinguish the question/instruction, the responses, and where to stop. The goal is to adapt the pre-trained model to better handle the new task, improving its performance on domain-specific tasks like medical diagnosis or sentiment analysis. Supervised fine-tuning allows the model to better generalize to tasks for which it was not explicitly trained during pre-training.

Studies have demonstrated that the supervised fine-tuning can enhance model performance in the clinical domain, and enable smaller models to compete with larger ones [120, 13, 26]. This fine-tuning process uses labelled clinical datasets—such as medical records, diagnosis

codes, or clinical notes—to teach the model to better handle healthcare-specific tasks like symptom analysis or diagnostic predictions. Instruction fine-tuning [112], a specific form of supervised fine-tuning, structures the training data using medical questions and answers. Special tokens from the pre-trained model are used to differentiate the question, response, and where to end the output. This process enables models like LLaMA-3 and GLM to better generalize to clinical tasks for which they were not explicitly trained, such as patient triage, disease classification, or generating treatment recommendations.

Many clinical and medical domain LLMs [116] [13] use clinical text data to instruct fine-tune base models like LLaMA-3 or GLM. Studies show that the models could achieve better clinical task performance with significantly less clinical training data. For the comparison, the original LLama-3 models were trained using over 15T tokens, whereas the MED42-V2 model only uses around 1 million rows of data to convert the original Llama-3 model into medical domain LLM.

It is worth noticing that different base models have different model sizes. In the conventional way of fine-tuning the model, developers need to train and modify all parameters of the model to produce their domain-specific fine-tuned model, and that is nearly irrational to implement for large-scale models like Llama-3-70B or Llama-3-400B in terms of the overall training cost. Recent studies has denoted a new strategy to avoid the out-of-control training cost called Parameter-Efficient Fine-Tuning (PEFT) [118]. It is a set of fine-tuning strategies which helps the developer fine-tune the base model with significantly less parameter update, and help the developers lower the training cost and time cost with minimum performance drop. There are two popular PEFT-related studies that are particularly useful: low-rank adaptation of large language models training (LoRA) [36] and quantized pre-trained language model into low-rank adapters training (QLoRA) [16].

As the name indicated, the essential idea behind the LoRA training strategy is to use fewer parameters for training to achieve similar performance as the full parameter tuning, with less computational cost in the end. In more detail, the LoRA training strategy freezes the pre-trained weights from the base model and only targets certain layers for modifications. It modifies the linear layers in the model by factorizing the weight updates into two smaller

matrices. If the original weight matrix has the dimensions of $m \times n$, LoRA represents the weight update as a product of two low-rank matrices, A and B, where A's size is $m \times r$ and B's size is $r \times N$. This reduces the number of trainable parameters while still capturing the important information needed to adapt the model. It is also might worth noticing that, the LoRA strategy adapts the idea of the "adaptor", which means, the model could choose whether it wants to use the domain-trained LoRA adaptor for the tasks. On top of that, it also provides the possibility to import multiple LoRA adaptors at the same time for cross-domain downstream tasks. Over the experiments, the LoRA strategy shows that the model could achieve over 95% of full-size training performance with only 10% parameters being trained in the instruction fine-tuning stage.

The QLoRA strategy builds on top of the vanilla LoRA framework with minor changes. For the vanilla LoRA-trained model that normally runs on 16-bit, if we want to further compress and quantify the model from BF16 to INT8 or INT4 numerical types, we would see the performance losses due to the datatype downgrading. The study of the QLoRA strategy is here to fill the gap and resolve the performance drop issue. It uses a 4-bit NormalFloat numerical type, which evolved from the FP4 and Int4 numerical types. In practice, the QLoRA will "expand" the parameters back to 16 bits when required in the training and inference but store and load it as 4 bits, which saves the computational cost as a result. The experiment shows that a model like Llama-65B could reduce the GPU memory consumption from 780 GB to 48 GB.

Another important aspect of the continual post-training process is the reinforcement learning from human feedback (RLHF) [73]. Generally speaking, the ultimate goals for LLMs are to be helpful, real, and harmless. Since the LLMs are pre-trained on relatively generalized text datasets compared to the target downstream tasks, they could fall short on all three goals at some levels; and the instruction fine-tuning could help with those shortages, but it cannot align the model response with human preference; and therefore, the purpose of RLHF is to align the LLM response with human-preferred reaction. In practice, almost all well-established large conversational language models on the market have been trained using RLHF, including OpenAI ChatGPT [72], Google Gemini [100], Anthropic Claude, and

more. The essential idea of RLHF revolves around training the reward model. Specifically, it uses human feedback to generate a human-preference dataset, which is then used to train a reward function that represents the desired outcomes for a specific task. This reward model is then used to iteratively improve the supervised fine-tuned model through reinforcement learning algorithms like Proximal Policy Optimization (PPO) [91] altering its internal text distribution to the sequences which are preferred by humans. The reward model serves to introduce "human preference bias" into the original supervised fine-tuned model. The resulting model would then generate responses that are closer to the human-preferred logic or format. The downside of implementing such a training process is the requirement of additional human-generated data that are expensive and difficult to produce in a large amount.

There are a few directions to further optimize the RLHF workflows. The first one is to replace the expensive human feedback with other trained models, which means using the model to generate human preference and guide the model for the training. This idea is what we call reinforcement learning from AI feedback (RLAIF) [49]. The RLAIF process consists of two main phases: supervised training and reinforcement learning. In the first stage, supervised training, the model initially responds to harmful questions, producing answers that may contain inappropriate content. The model then performs self-evaluation based on predefined principles, adjusting its responses iteratively until they meet the desired standards. This refined set of responses is used to fine-tune the model, resulting in the "SL-CAI" (Supervised Learning Constitutional AI) model, which is capable of providing both safe and useful answers. The second stage involves reinforcement learning, where the model generates pairs of responses to harmful questions. A feedback model selects the better response based on established principles, and these selections are used to train a reward model. The final step involves employing this reward model as the basis for reinforcement learning, further refining the model into the "RL-CAI" (Reinforcement Learning Constitutional AI) model, ensuring that it can appropriately handle harmful queries while delivering safer and more effective responses.

The second direction would be more feasible to implement. Instead of training the reward model and using the reward model for preference learning, the study of direct preference

optimization [83] proposes implementing a binary selection for the preferences, a positive sample and a negative sample, to replace the reward models and simplify the RLHF process. Like existing algorithms, DPO relies on a theoretical preference model (such as the Bradley-Terry model) to measure how well a given reward function aligns with empirical preference data. However, while current methods define preference loss based on the preference model to train the reward model and then optimize a policy using the learned reward model, DPO defines preference loss directly as a function of the policy through a change in variables. Given a dataset of human preferences for model responses, DPO can, therefore, optimize the policy using a simple binary cross-entropy objective without the need to explicitly learn a reward function or sample from the policy during training.

2.6 Summary

This chapter traces back to the beginning of the modern large language models and introduces the evolution from generalized models to ones that fit personalized requirements for different users. It highlights the essential feature of the modern GPT-based "large language models" - predicting the next token, and demonstrates that it is curial to optimize this feature in order to obtain better performance in downstream tasks.

This chapter then explores prompt engineering as a cost-effective strategy to adapt the generalized large language models for personalized requirements. And then it shift to retrieval-augmented generation (RAG) - a special prompt engineering framework that adopted the idea of auto configuration in the prompt engineering direction. Both ideas demonstrate that it is possible to optimize the performance of the generalized large language models by providing appropriate text inputs before asking model to generate the response.

Finally, the discussion shifts assuming the prompt engineering direction cannot fulfill the requirements. The focus in this direction is to adapt the models into specialized domains such as the clinical domain. By using efficient methods like LoRA and QLoRA, it is possible to modify only a small proportion of model parameters and achieve the domain adaptation.

CHAPTER 3

Strategies for Efficient Retrieval-augmented Generation in Clinical Domains with RAPTOR

3.1 Introduction

Language models have been used in various clinical applications [114, 71, 89, 48]. Challenges that are common but not exclusive to clinical applications include requirements related to data privacy [105, 15], robustness and requirements for generating stable and repeatable results [63], and the cost efficiency of using models as a service [10].

Due to these challenges, cloud-based language models may not be suitable for clinical applications where the data needs to be stored on a local machine and requires a local implementation of the language model [98, 89]. This also means that language models often need to be implemented on consumer hardware, limiting the size of the models that can be used. This sentiment is echoed by Mesko and Topol [63], who explain that locally hosted models enhance both the privacy and robustness of clinical language models. Language models such as Mistral-7B-Instruct-v0.2 [39] and Llama-2-7b-chat-hf [102] can be deployed on consumer hardware (Table 3.1) but are affected by the long context problem.

Several approaches have been developed to address the issues of handling longer documents in text summarisation and generation for local language models. LongLoRA [12] and Infini-attention [66] modify the attention mechanism and fine-tune the models to enhance the model capability of handling long texts up to 500K tokens in length. However, both of them require additional fine-tuning in the training phase and extra memory during inference for additional texts. Others including CFIC [81] and BGE Landmark Embedding [62], introduce special

layers or embedding models to select partial information without breaking down the long contents. Similar to the above, these two methods also require additional training for the model to run properly, and they are limited to either 32k token max length or the max model context length, respectively.

Retrieval-augmented Generation (RAG) framework [53], which works by setting up an external knowledge datastore to keep the long contents in chunks, and retrieving partial information from the preprocessed datastore as a reference text when generating a response. It could be adapted by both cloud-based and local language models since it is an external framework that works outside the model inference. It could reduce the overall cost by token counts for cloud-based language models and increase the capability for relatively smaller local language models that are focusing on. While it is cost-effective and able to deal with long content for the local language models, it comes with the obvious drawback of potential information losses during the text retrieval process.

Recursive Abstractive Processing for Tree-Organized Retrieval (RAPTOR) [90] extends from RAG and is an efficient method for language models to handle long text contents and reduce model hallucination without any internal model modifications. Different from the naive RAG framework [53], it implements a tree-structured datastore to preprocess and retrieve both relevant local and overview information from the datastore to support model generation. It can be more reliable when the user query requires information from a distance, e.g., the question requires the patient information located in the first section of a 6k token-long discharge summary, the Laboratory results in the middle, and admission medication in the last two sections to answer the question.

Other tree-based RAG-based frameworks include T-RAG [23] and RAGAR [44]. The RAPTOR framework implements the tree structure directly, and the retrieved text can contain complete overview information as support. In contrast, T-RAG implements the tree structure to modify the user query for organisational-related entities, and RAGAR uses the tree structure to generate sub-questions for the original user query. None of them are capable of dealing the overview information loss like RAPTOR does.

RAPTOR has been shown to achieve state-of-the-art performance on the QuALITY dataset [75], which is a question-answering dataset aiming for long-context scenarios. While RAPTOR has achieved strong results, it was evaluated on the cloud-based GPT-4 model. To the knowledge, its ability to work with smaller language models like Mistral-7B-Instruct-v0.2 has not been validated.

The aim of this study was to benchmark the RAPTOR framework and design the pipeline for clinical question-answering applications, using language models that can be implemented locally. The contributions of this study are as follows:

- Benchmarking the response accuracy of the RAPTOR framework using efficient locally implemented language models and evaluating the result with the cloud-based model, investigating differences between simple and complex questions.
- Constructing a new semantic chunking strategy that makes use of semantic relationships between texts and evaluating the performance changes to the RAPTOR framework.

3.2 Background and related work

3.2.1 RAPTOR framework

The processing procedure of the RAPTOR framework comprises the following steps to produce a tree structure:

- The document text is split into a set of chunks.
- An embedding model is used to convert each chunk into sentence embedding, and Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP) [51] is used to reduce the dimensionality.
- Unsupervised clustering is used to cluster the chunks using the Gaussian Mixture Model [86] with Bayesian information criterion.

- Chunked text in each cluster is concatenated, and a language model is used to generate a detailed summary.
- The above steps are followed recursively until the final root node is reached.

Once the tree structure has been constructed, the model is able to either follow the tree structure to retrieve relevant information at any step or collapse the tree structure to gather relevant information directly. This means the RAPTOR framework can potentially use both overview and local information as a reference during the inference process when generating a response to a query.

3.2.2 Chunking strategies in the RAPTOR framework

In the RAPTOR framework, methods for splitting the document into chunks can affect the performance of the model. Chunking methods include character chunking and semantic chunking.

3.2.2.1 Chunking by character

The standard RAG chunking strategy uses a regular expression to split text using characters in the text (e.g. "." or ";") under the preset maximum length limit. This is the standard chunking strategy used in the RAPTOR framework. This approach is efficient in both execution time and computational cost. Because it does not consider semantic context, it can split semantically similar sections of text into multiple chunks, which may result in information loss.

3.2.2.2 Chunking by semantic information

One approach for improving chunking is to make use of semantic differences over the length of the document. This unsupervised approach converts sentences into embedding vectors, calculates the cosine distance between adjacent sentences, and then splits the document into chunks using a threshold value for the cosine distance. This approach can be used directly within the RAPTOR framework in the first step.

Research from a related domain focuses on adapting deep learning models to support text segmentation. One example proposes using a BERT-based embedding model to convert sentences into sentence embeddings, then using those as inputs to a transformer model to identify breakpoints in the sentence embedding [61]. Another study proposes the use of a moving window inside the BERT model structure [124]. A PoNet-based model [99] was able to extend the context length from 512 tokens to 4096 tokens. Note that while these approaches were not designed specifically for use within the RAG framework, the goal is the same and the methods can be adapted for use with RAPTOR.

3.2.2.3 Chunking by language model

A relatively recent set of approaches consider the use of language models to support chunking. One example proposes splitting a document into chunks called 'propositions' and using prompt engineering pipelines to group propositions together using a node-tree structure [11]. Another example proposes the use of language models as decision-making machines that select different text chunks for RAG-based tasks [80]. While these approaches represent elegant solutions for chunking, the current limitation is the computational and time costs, which may make them less practical for downstream tasks and especially for application domains where consumer-level hardware is a constraint.

3.2.3 Embedding models in the RAPTOR framework

Embedding models can have a major impact on the performance of downstream tasks. RAPTOR uses embedding models in the construction of the tree and in the information retrieval process before the response generation. A recent study compared the performance of several embedding models for downstream tasks in the health domain, including general embedding models and specialised models trained using text data from health application domains [21]. The results show large variations in performance across the embedding models.

The context lengths of models can also affect performance on downstream tasks, even when used within an RAG framework. For example, BioBERT [50] is a popular model trained on

TABLE 3.1: Language model availability and size

Model Name	**Open-source**	**Parameter Size**
GPT-4o	No	Not Published
GPT-4 [3]	No	Not Published
Gemini 1.5 Pro [89]	No	Not Published
GPT-3.5-Turbo	No	Not Published
GPT-3 [8]	No	175 Billions
Llama-3-8B-Instruct [4]	Yes	8.03 billion
Mistral-7B-Instruct-v0.2 [39]	Yes	7.24 billion
Llama-2-7b-chat-hf [102]	Yes	6.74 billion

the biomedical text and has a maximum context length of 512. This compares to the more recently developed models jina-embeddings-v2-base-en (Jina) [30], BGE-M3 (BGE) [9], and text-embedding-ada-002 [69], which have a maximum context length of 8,192. The way this can affect performance is when shorter context length embeddings cut off sentences that exceed the maximum context length, leading to information loss.

Language models with between 1 billion and 10 billion parameters are considered to be 'medium scale' [65]. In 2024, the most powerful consumer-level GPUs available on the market have 24GB of GPU memory. This limits the scale of language models that can be implemented comfortably to these models with 10 billion parameters or fewer. The Llama-2-7B and Mistral-7B-Instruct-v0.2 models each have approximately 7 billion parameters (Table 3.1). Compared to large-scale models like GPT-4 and Gemini Pro, medium-scale models are more practical for use in application domains where local computing resources are required.

TABLE 3.2: Examples of simple & complex questions in the QuALITY and CTQA datasets

Question Type	Question Content
QuALITY, Simple	Which of the following most closely fits the theme of this article?
QuALITY, Complex	What does the author likely think will happen if democracy does not evolve?
CTQA, Simple	In the results of this trial, how many participants had serious adverse events in each study arm?
CTQA, Complex	In the design of this study, what intervention was used in the control arm?

3.3 Methodology

3.3.1 Datasets

Benchmarking was performed using two datasets. The QuALITY dataset [75] is commonly used to evaluate approaches in the RAG framework. This study introduce the Clinical Trials Question and Answer (CTQA) dataset as an example of a large dataset from biomedical application domains. Both datasets include opportunities to ask simple and complex questions (see examples below). Note that QuALITY has multiple-choice questions and answers and CTQA has short-answer questions and answers.

3.3.1.1 QuALITY dataset

QuALITY [75] is one of the three question-answering datasets used in the original evaluation of the RAPTOR framework [90]. The dataset includes three main sections: the main text documents, questions and correct answers. QuALITY was used as three subsets: training, development, and testing. The development subset included 230 article and multiple-choice questions, where each question was labelled as simple or complex 3.2. The length of the main text documents ranges from 2000 tokens to 6000 tokens. In the original study where the author proposed the QuALITY dataset, it states that the difficulty of the questions is labelled manually where the question is labelled as hard/complex question if the human annotator cannot answer the question within 45 seconds.

3.3.1.2 CTQA dataset

ClinicalTrials.gov is a registry for clinical trials and includes information about more than 500,000 trials and other studies, designed to provide public information about the design of studies before the study begins [96, 79]. Each study includes sections of text with a summary of the study, the population, interventions or exposures, and outcome measures. Some studies on ClinicalTrials.gov also include tables with numerical data describing the summary results of the study after it is completed. With a similar structure to the QuALITY dataset, CTQA includes main text documents (the registry entry), and pairs of short answer questions with their answers.

The simple question for the CTQA dataset was focused on information extraction, and the complex question requires both information extraction and to generate an appropriate response using contextual information (Table 3.2).

The CTQA dataset is useful because it is large and growing over time, has a complex structure including structured and unstructured data, and is an application domain representing a very large and expensive industry domain. Downstream tasks include those related to improving the efficiency of trial designs to avoid redundancy and avoid termination, synthesis and meta-analysis of trials that answer the same clinical question, and checking for reporting bias when results in published trial articles do not match what was registered.

For the simple question, the correct answer can be extracted at scale from studies in ClinicalTrials.gov that include a structured results section. The answer can be found under a structured result section labelled 'seriousNumAffected', which should indicate the number of patients that were analysed for serious adverse events. The complex question requires a contextual understanding of whether the trial is an interventional study, the structure of the trial design, including the number of study arms, and identifying which of the study arms is most likely to be used as a comparison for the intervention under investigation. It is often, but not always, a placebo, sham, or treatment as usual.

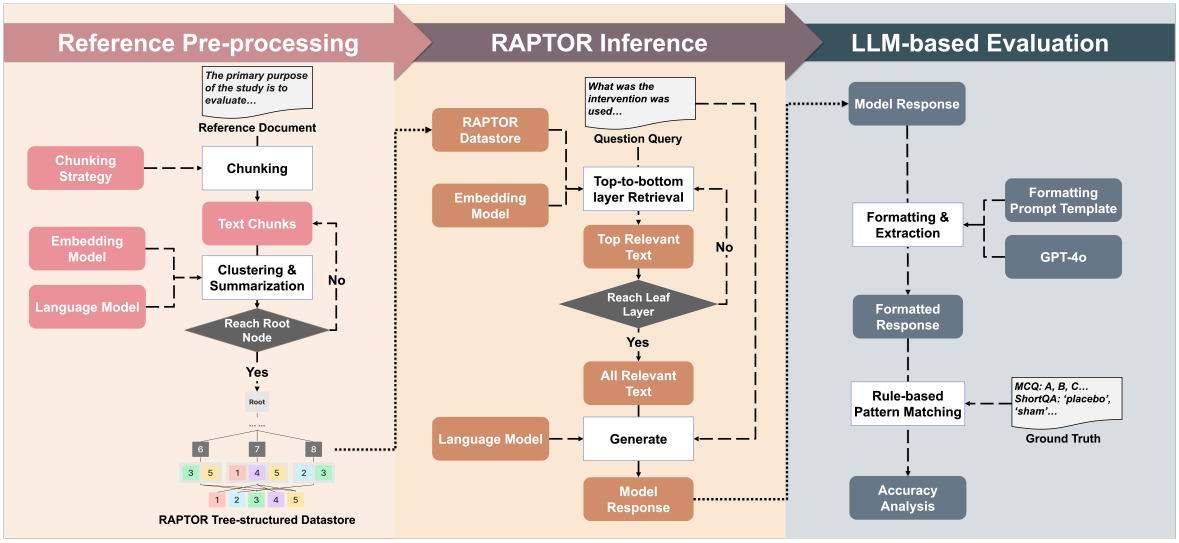

FIGURE 3.1: Overall structure diagram of the framework and evaluation process.

3.3.2 Chunking strategy configuration setup

The three chunking strategies included a character-based chunking strategy, a new semantic chunking strategy, and a chunking strategy that uses a deep learning method.

3.3.2.1 Naive character chunking

The default chunking strategy used in the RAPTOR framework is the naive character chunking strategy (character chunking), as the RAPTOR framework was not focused on this aspect. In this study, the default character-splitting chunking strategy will also be tested on both CTQA and QuALITY datasets as the reference.

3.3.2.2 Recursive moving percentile semantic chunking

This study introduces a modified semantic chunking strategy named Recursive Moving Percentile Semantic Chunking (RMP chunking). Rather than using a globally fixed value for the breakpoint between text sections, then calculate the threshold value dynamically with the moving percentile. The process is divided into the following steps:

- The character chunking algorithm is used to split the original document into the minimum chunks.
- Chunks are converted into sentence embedding vectors via a preset embedding model and calculate the cosine distances for each adjacent piece with the context padding.
- The moving percentile values are calculated based on the cosine distance with the preset window size and the preset percentile threshold value. All critical indices that are over its moving percentile threshold value are labelled.
- The document is broken into chunks where the moving percentile threshold value is exceeded, recursively, until there are no more indices within chunks that are over the moving percentile threshold value.

3.3.2.3 Deep learning based semantic chunking

PoNet [99] is a chunking strategy that uses a deep learning model and comes from the research area of text segmentation. While text segmentation models like hierarchical BERT [124] and PoNet were not designed to fit the needs of the chunking strategies in RAG-based frameworks, their motivations and structures are similar and can be used as chunking strategies in RAG-based frameworks.

3.3.3 Embedding model configuration setup

Four embedding models were used in the experiments 3.3. The default ‘baseline model is the text-embedding-ada-002 (OpenAI Embedding) model from OpenAI with 8192 context length on paper, and it is cloud-based deployed in the OpenAI servers. Since there is a potential need for the embedding model to deal with longer text, 2 local embedding models with long context length are introduced into the experiment: the BGE-M3 (BGE) Embedding model and the jina-embeddings-v2-base-en (Jina) Embedding model; where the BGE Embedding model is utilized for the RAG implementation according to their technical report, and Jina Embedding model has also been utilized to extend the context length from 512 to 8192 tokens. On top of all that, a traditional but biomedical domain embedding model, the BioBERT Embedding model, is also introduced to the experiment as well for comparison.

3.3.4 Language model configuration setup

For the summarisation and generation experiments, the more cost-effective GPT-35 model was tested as the baseline model to demonstrate the basic result. Language models GPT-35 and Mistral-7B-Instruct-v0.2 model (Mistral-7B) were used in the experiments. These 2 models should be able to represent both "very large language models" and "medium language models" accordingly, and these 2 models are also good examples of closed-source LLMs and open-source language models in the generalized domain.

To simulate a realistic low-setting environment, the experiments all used the 4-bit QLoRA quantization setting for the local models Mistral-7B.

3.3.5 Overall configuration setup

The RAPTOR framework contains 3 key components: the chunking strategy, the embedding model and the summarization/generation model. All combinations of the 3 components presented in Table 3.3 were implemented for both the CTQA and the QuALITY datasets as presented in Figure 3.1. Additionally, this study tested GPT-4o directly for both datasets to demonstrate the performance in terms of the task accuracy of the latest cloud-based language model. All experiments are implemented with respect to the zero-shot performance of the language models with the minimum hint.

3.3.6 Language model and rule-based evaluations

Since the model responses do not always match the correct format in terms of verbosity, even though they may still be correct (e.g., a response of "neither study arm had any serious adverse events" vs. the expected "Arm 1: 0; Arm 2: 0"), the pattern matching could be inaccurate, and the manual process would be time-consuming.

To overcome these issues, this study designed an LLM-based approach using LLM for format extraction and rule-based pattern matching for accuracy scoring. While the LLM evaluation is consistent to others [33], in that this study use LLM to judge the results using questionnaires

TABLE 3.3: Component Configurations

Component Type	**Name**	**Abbreviation**
Chunking Strategy	Naive Character Chunking	Character Chunking
	Recursive Moving Percentile Semantic Chunking	RMP Chunking
	PoNet Semantic Chunking	PoNet Chunking
Embedding Model	text-embedding-ada-002	OpenAI
	BGE-M3	BGE
	jina-embeddings-v2-base-en	Jina
	BioBERT	BioBERT
summarization and generation model	GPT-3.5-Turbo	GPT-35
	Mistral-7B-Instruct-v0.2	Mistral-7B

and collect scores in rule-based evaluation, the method is able to work for both MCQs and short QAs since this study focus on if the response matches the answer. The result confirmed that the LLM-based approach was accurate with a manual assessment of 500 examples of MCQ from the QuALITY dataset and 500 short QA examples from the CTQA dataset, achieving a match rate of 98.8%. The process is as follows:

- Generate model responses using the prompt templates provided in ZeroSCROLLS [93] and the framework described in the methodology section.
- Prompt the GPT-4o model to extract the well-formatted answer from the original model response in the previous step, using 3-shot prompt templates to provide formatting examples presented as table.1 in the Appendix section.
- Compare the extracted answer with the ground truth from the dataset using rule-based pattern matching and evaluate the accuracy.

Accuracy has been implemented as the evaluation metric for both datasets in this study. Although the F1-score may be better suited for an MCQ dataset like the QuALITY dataset, it cannot be used in the evaluation process for the short answer questions in the CTQA dataset since there are no "true positive" or "false negative" cases in this scenario. Therefore, accuracy is implemented as the sole evaluation metric for the purpose of direct comparison.

3.4 Results

3.4.1 Experimental results for the CTQA dataset

The varied across chunking strategies, embedding models, language models, and across simple and complex questions for the CTQA dataset (Table 3.4 and Figure 3.2).

In the experiment of the simple question, the results with the GPT-35 model show that the PoNet chunking strategy consistently outperformed the other two strategies 3/4 times with an average lead of 8.46% improvement, whereas the experiment results using the Mistral-7B model show that the RMP chunking strategy has outperformed the other two chunking strategies, with a difference of 10.88% in accuracy. In the complex question, the results in both models show that the PoNet chunking strategy can perform better compared to the other two with the leading average of 1.42% and 9.89%.

TABLE 3.4: Task accuracy of the RAPTOR framework for various configurations with the CTQA dataset

Language Model	Embedding	Simple Question (c=400, n=400)				Complex Question (c=400, n=400)			
		Character Chunking	RMP Chunking	PoNet Chunking	Average	Character Chunking	RMP Chunking	PoNet Chunking	Average
GPT-35	OpenAI	63.44	**64.37**	54.05	60.62	75.87	73.17	**77.12**	75.39
	BGE	56.37	64.25	**78.48**	66.37	70.38	77.70	**74.52**	74.20
	Jina	36.12	50.42	**54.61**	47.05	48.78	69.32	**72.55**	63.55
	BioBERT	37.00	21.59	**51.34**	36.64	63.41	67.82	**69.48**	66.90
	Average	48.23	50.16	59.62	52.67	64.61	72.00	73.42	70.01
Mistral-7B	OpenAI	55.05	**64.21**	29.37	49.54	68.42	**74.23**	74.03	72.23
	BGE	58.79	**72.27**	70.36	67.14	52.04	54.21	**69.53**	58.59
	Jina	39.52	**50.75**	35.81	42.03	45.62	52.11	**59.35**	52.36
	BioBERT	6.00	15.79	**23.96**	15.25	53.41	**60.53**	56.12	56.69
	Average	39.84	50.76	39.88	43.49	54.87	60.27	64.76	59.97
GPT-4o	-	-	-	-	93.25	-	-	-	52.26

*Value c indicates the number of long reports contained in the dataset, whereas value n indicates the number of questions contained in the dataset.

*Bold values show the highest value across the 3 chunking strategies with the same configurations; Underline values show the highest value across the 4 embedding models with the same configurations.

Among the 4 embedding models in the experiments, result values in both simple and complex questions show that the highest values are either coming from the OpenAI embedding model or the BGE embedding model. On average, the result of the simple question shows that, The

GPT-35 model with OpenAI and BGE embedding model scores at 60.62% and 66.37% on average, and the Mistral-7B model with the OpenAI and BGE embedding model scores at 49.54% and 67.14% on average; the result of the complex question shows that, The GPT-35 model with OpenAI and BGE embedding model scores at 75.39% and 74.20% on average, and Mistral-7B model with with OpenAI and BGE embedding model scores at 72.23% and 58.59% on average. All results showed that the OpenAI embedding model and BGE-embedding model outperformed the Jina embedding model and BioBERT embedding model.

In terms of the direct comparison between the GPT-35 and Mistral-7B models in the RAPTOR framework (and with the separate GPT-4o model), the results for the simple question show that the GPT-4o substantially outperformed the RAPTOR framework configurations. GPT-4o reached 93.25% accuracy, compared to an average of 52.67% for the GPT-35 model and 43.49% for the Mistral-7B model across the different configurations. The result was different for the complex question, where the GPT-4o mode achieved 52.26% accuracy, compared to an average of 70.01% for the GPT-35 model and 59.97% for the Mistral-7B model across the set of tested configurations within the RAPTOR framework.

3.4.2 Experimental results for the QuALITY dataset

In the QuALITY dataset experiments, the GPT-4o model outperformed RAPTOR with both model configurations in the simple and complex questions by around 30% (Table 3.5 & Figure 3.2).

The performance difference between the two models within the RAPTOR framework is relatively small for both questions. Accuracy for the simple question was 67.24% for the GPT-35 model and 66.87% for the Mistral-7B model. Accuracy for the complex question was 49.36% for the GPT-35 model and 48.86% for the Mistral-7B model. Performance across the chunking strategies showed relatively consistent differences. RMP chunking generally outperformed other strategies in the simple and complex questions, and with both GPT-35 and Mistral-7B language models. Average accuracy for the simple question using the RMP chunking strategy was 68.27% for GPT-35 and 68.12% for Mistral-7B. Average accuracy

TABLE 3.5: Task accuracy of the RAPTOR framework for various configurations with the QuALITY dataset

Language model	Embedding	Simple question (c=115, n=1021)				Complex question (c=115, n=1065)			
		Character Chunking	RMP Chunking	PoNet Chunking	Average	Character Chunking	RMP Chunking	PoNet Chunking	Average
GPT-35	OpenAI	69.44	**70.03**	69.79	69.90	49.86	51.27	**51.97**	51.03
	BGE	67.68	**68.27**	67.87	67.94	50.42	**52.96**	50.05	51.14
	Jina	**68.95**	68.66	68.17	68.59	50.23	**50.33**	47.98	49.51
	BioBERT	61.51	**66.11**	60.33	62.65	44.79	**49.30**	43.19	45.76
	Average	66.90	68.27	66.57	67.24	48.83	50.97	48.30	49.36
Mistral-7B	OpenAI	68.66	69.64	**69.70**	69.33	48.64	**52.58**	47.17	49.46
	BGE	67.48	67.68	**66.70**	67.29	50.33	**52.77**	49.30	50.80
	Jina	66.90	**69.93**	68.95	68.59	49.39	**51.36**	48.83	49.86
	BioBERT	60.72	**65.23**	60.82	62.26	43.47	**48.26**	44.23	45.32
	Average	65.94	68.12	66.54	66.87	47.96	51.24	47.38	48.86
GPT-4o	-	-	-	-	95.00	-	-	-	83.29

*Value c indicates the number of long stories contained in the dataset, whereas value n indicates the number of questions contained in the dataset.

*Bold values show the highest value across the 3 chunking strategies with the same configurations; Underline values show the highest value across the 4 embedding models with the same configurations.

for the complex question using the RMP chunking strategy was 50.97% for the GPT-35 model (2.14% higher than the second best performing chunking strategy) and 51.24% for the Mistral-7B model (3.28% higher than the second best performing chunking strategy).

The OpenAI and BGE embedding models outperformed Jina and BioBERT embedding models across all but one of the configurations (Table 3.5). Overall differences between OpenAI and BGE are relatively small, suggesting that the choice of embedding model between the two is less important than the choice of language model and chunking strategy.

3.5 Discussion

The results show that differences in the choice chunking strategy, embedding model, and the complexity or type of questions each appear to have an impact on performance within a RAPTOR framework. These choices are likely to be especially important in scenarios where implementation is restricted to local machines and access to high-performance cloud computing is restricted.

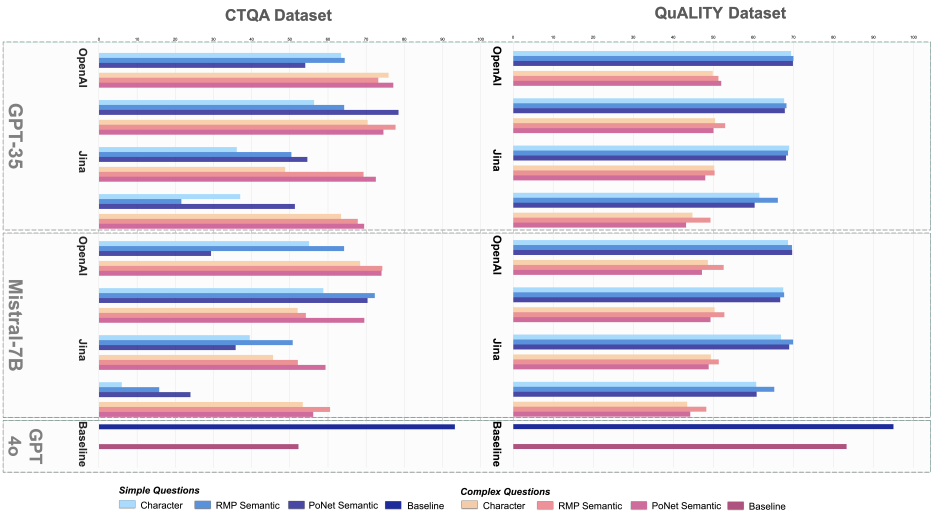

FIGURE 3.2: Task accuracy for the CTQA (left) and QuALITY (right) dataset across simple and complex questions and using GPT-35 and Mistral-7B, showing differences in performance compared to a GPT-4o baseline.

3.5.1 Chunking strategy

The results show that for a given model, semantic and deep learning chunking strategies generally improve the performance relative to standard character-level chunking in the CTQA dataset. These differences appeared for both simple and complex questions. The chunking strategy result may seem counterintuitive because the RAPTOR framework is designed to 'connect' all the information together in a tree structure. However, the main benefit of chunking strategies in the RAPTOR framework may come from improved summarisation in the leaf nodes, and this improvement then flows through the local and global information in the tree structure.

Lower performance was found for the PoNet chunking strategy using the OpenAI embedding model with both GPT-35 and Mistral-7B models in the simple question of the CTQA dataset. Because the PoNet chunking does not have a hard upper limit in the maximum chunk size, it could have larger initial chunks. This suggests that the OpenAI embedding model may not perform as well as the BGE embedding model in the long context embedding task.

3.5.2 Language model

Despite the relatively large performance difference between the GPT-35 model and the Mistral-7B model, the results show that the Mistral-7B model is able to reach the same level of performance as the GPT-35 with the BGE embedding model and the semantic chunking strategies.

GPT-4o outperformed all configurations of RAPTOR for the QuALITY dataset and for the simple question in the CTQA dataset, but did not perform as well for complex questions in the CTQA dataset. Note that when experts answer the complex CTQA question, they typically make use of the information from the title, brief and detailed summary, and contextual information about the structure and design of the trial. This suggests that the ability to synthesise local and global information is particularly important for the question, and may explain performance differences between GPT-4o and the top-performing RAPTOR configuration.

3.5.3 Embedding model

In terms of the embedding models, the results show that the text-embedding-ada-002 embedding model (OpenAI embedding model) from OpenAI and the BGE embedding model achieved similar performance, with few exceptions. The BGE model was designed to handle RAG-related tasks natively, which may explain the generally strong performance. When choosing an embedding model for use with the RAPTOR framework, the BGE embedding model appears to perform at least as well as the OpenAI embedding model and it is feasible to fine-tune the BGE model for domain adaptations.

3.5.4 Future work

The results suggest that applications of the RAPTOR framework and appropriate choices for language models, chunking strategies, and embedding models may still be relevant even as larger language models are developed. New evidence suggests that models with much

longer context lengths may still suffer degradation in performance [35, 57], which suggests that RAPTOR and other tree-structure frameworks could increase performance for applied tasks not only in environments with constrained computing resources, but also more broadly in larger models.

Another interesting aspect of this work was the use of simple and complex questions. An underlying assumption is that the RAPTOR framework may be better suited to questions that need to make use of local and global information to produce correct responses. Questions labelled as 'complex' in the QuALITY dataset do not always represent this same type of complexity. New datasets and question-answer pairs that examine this type of complexity in more detail are likely to be useful for future investigations.

ClinicalTrials.gov is underutilised. It is a large and public dataset, with a broad range of downstream tasks [60, 109, 19]. Many of these downstream tasks may benefit from RAG-based methods to support information extraction, synthesis, and classification. The CTQA dataset may be of value to the community for further benchmarking and used as test cases for the development and evaluation of new methods.

3.6 Conclusion

This benchmarking study showed that the RAPTOR framework varies in performance depending on configuration and the types of questions it is used to answer. The results showed that language models with fewer than 10 billion parameters can be used with the RAPTOR framework to overcome the long context problem, and these configurations are a feasible solution in scenarios where larger language models cannot be used. The results also showed that the use of a semantic chunking method improved the results compared to the standard character chunking method.

CHAPTER 4

CTLlama: Clinical Trial Specific Domain Open-source Large Language Model for Research

---

4.1 Introduction

ClinicalTrials.gov is a registry for clinical trials and includes information from more than 500,000 trials and other studies, developed to provide public information about the design of studies before the study begins [96, 79]. The goal is to explore how LLMs can improve the use of this data. While many studies have focused on clinical trial prediction and data mining with traditional methods like Random Forest, XGBoost or Logistic Regression models [109, 19, 59]. Recent work has shifted towards LLMs implementation for the ability to generalize across different downstream tasks. One work conducted and implemented based on the existing off-the-shelf LLMs [85] like GPT-4. Another work combined with external knowledge-graph-based RAG framework to optimize the performance [47]. Recent studies have not yet touched on how to optimize the LLM itself for the ClinicalTrials.gov database.

This study proposes a new pre-trained domain-specific model called CTLlama-8B. The objective of the CTLlama-8B model is to create an optimized open-source model that can be inference locally to understand and generate insights from clinical trial registration content. The model builds on the Llama-3.1-8B model and is continually pre-trained on over 100 billion tokens from the newly collected and preprocessed clinical trial registration dataset named CiTi, which focuses on the clinical trial registration data and their related publication abstract contents. The contributions to this study are as follows:

- Collected and preprocessed a large clinical trial registration dataset, named the CiTi dataset, consisting of 358870 clinical trial registration reports converted from JSON data to human-readable reports. The dataset also contains 1401401 PubMed abstracts mentioned in clinical trial registration data as background studies, result publications or related publications. Those studies are mentioned in citation format, and their abstract can be downloaded from PubMed. The dataset can use these abstracts as additional domain knowledge for the motivation or outcomes of the clinical trials, and extending the knowledge base from pure clinical trial context to the medical and health domain.
- Produced the domain-specific CTLlama-8B model using domain-adaptation continual pre-training for a better understanding of the clinical trial registration content and relevant medicine and health domain content. Additionally, a CTLlama-8B-demo model was produced during the experiment to fully demonstrate the impact of the continual pre-training on the language models.

4.2 Related Work

4.2.1 Open-source LLMs

The industry divided the LLMs into closed-source LLMs and open-source LLMs. The close-source LLMs are the GPT-4 [72], Claude-3.5 [78], Gemini [101] and other LLMs that only provide service by the founder company without providing the model parameters. The open-source LLMs like Mistral [39], Qwen [6] and Llama [4] have published the model parameters and welcome other researchers for downloading. More comparisons are listed in Table 3.1.

Using the Llama model series as an example of open-source LLMs, it has an extremely clear and transparent development roadmap. It started its journey from the original Llama model series in 2023 to the Llama-3.2 model series, released recently in September 2024. It also

covers a wide range of selections from 3 billion parameters to 405 billion parameters for different purposes.

Many studies are built successfully on top of those foundational open-source models for evaluation [125, 13, 55], domain shifting, and developing new strategies. For those interested in shifting these generalized LLMs into domain-specific models like biomedical or clinical domains, their main converting strategies are either implementing instruction fine-tuning for the downstream tasks or conducting continual pre-training for the domain knowledge injection.

4.2.2 Instruction fine-tuning

The concept of instruction fine-tuning was introduced in 2021 by the study of zero-shot learning [112]. It is a special training process that involves training the model using the instruction and output from the dataset. The instruction indicates the human instruction or the user query, and the output represents the desired output from the model developer.

Instruction fine-tuning can help the model to generate better results in the downstream tasks with appropriate instruction datasets. Many studies have adapted this idea to build their version of the domain-specific model from the open-source ones. ChatDoctor [55] is an excellent example of how only instruction fine-tuning could modify the generalized LLMs, such as the Llama model, into a medical domain model. It uses 100,000 patient-doctor dialogues as the instruction dataset for the training process to implement instruction fine-tuning, and the result shows that it outperformed the ChatGPT in a wide variety of novel medical tasks in question-answering scenarios. The ChatDoctor model demonstrated what a fine-tuned model could achieve in its target domain, and it used a large dataset, which consumed a relatively large computational resource during training.

In reality, not everyone can afford the resources to conduct full-size training on the model, and it is not cost-effective to do that for a large dataset. Therefore, a more efficient fine-tuning strategy is required to optimize the training process. Low-rank adaptation [36] (LoRA) is one of the most popular training strategies in the area. The idea behind the LoRA strategy is to

first freeze the model parameters, then construct and train the new parameters to the target model layers on the side of the model and adapt the new parameters to the original model.

The most obvious advantage of the LoRA strategy is the significantly lower cost compared to the full-size training. The original LoRA study indicated that the model could achieve 95% of full-size training performance with only 10% of parameters being trained. Another study shows that it has a relatively small catastrophic forgetting issue compared to full-size training and freeze-layer training. Besides that, because the LoRA strategy adopted the idea of the adaptor, which means the new parameters are stored separately from the model, it can be quickly swapped in and out for different domains and tasks, reducing the performance decrease by cross-domain. The Med42 [13] has demonstrated the LoRA strategy implementation in the instruction fine-tuning process. With limited resources, it can still surpass the GPT-3.5 model in multiple medical datasets.

4.2.3 Domain adaptation continual pre-training

Instruction fine-tuning can help the LLMs generate better responses to downstream tasks by training them on the relevant instruction dataset. However, it is not so helpful when the model lacks domain knowledge. For example, a pre-trained LLM can only be trained using the English dataset, and it would be hard for the model to understand the Spanish instruction; or if the model had only been trained on the general domain texts, it would not be optimal for the clinical domain tasks after.

The continual pre-training process can be more helpful in this scenario. It represents the pre-training process on top of the already pre-trained model [31]. Some empirical studies [43, 123] discussed the importance of the continual pre-training process when constructing the domain-specific models. PMC-Llama [113] demonstrated a good example of the continual pre-training process. It conducted the continual pre-training process first as a step of knowledge injection, then performed the instruction fine-tuning process to target the downstream tasks and the ability to "communicate". It stated that the continual pre-training process is essentially a process of knowledge injection, which helps the LLM develop more potential in the future.

In its comparison, it surpassed the ChatGPT in multiple medical-related tasks, and it also outperformed the previously mentioned Chat-Doctor by over 20% in those tasks.

The concerns of the continual pre-training process are similar to those of the instruction fine-tuning. The most obvious one is the requirement of an extensive number of computational resources for full-size training. Since continual pre-training and instruction fine-tuning are both casual model training, it is common to use similar strategies, i.e. LoRA training, to deal with resource consumption. The second issue is the catastrophic forgetting problem mentioned by multiple studies [54, 94]. The continual pre-training process would cause the model to perform worse in the previously trained domain while it gains better results in the new domain. These studies conclude that there is no good solution to completely solve the catastrophic forgetting problem. however, some studies have provided comments and solutions to mitigate the issue. For example, RationaleCL [117] proposed a contrastive rationale replay to deal with the problem. Similarly, Contunual-T0 [92] designed a memory buffer to replay the previous task. Also, a LoRA-based approach CURLoRA [24] is designed to stabilise the model performance.

4.3 Methodology

4.3.1 Dataset

I have collected and organized two main datasets for the continual pre-training in this study.

The first dataset collected is the clinical trial registration (CTR) dataset, which includes all clinical trial registrations that can be downloaded from ClinicalTrials.gov until March 2024. ClinicalTrials.gov is a registry website that stores over 500,000 detailed clinical trial registration data that could be in the states of proposing, ongoing, finished or termination. It is run by the United States National Library of Medicine at the National Institutes of Health and holds data from 221 countries. The total number of reports collected and included is 358870.

As demonstrated in Table 4.1, the original data are in the JSON format string. Compared to the website-displayed version, the JSON data contains extra unnecessary data like the keys and ids and it is different from the human-readable text. This study have analyzed all reports and compared them with the website-displayed version. I then preprocess the raw data and construct the text reports based on the JSON data. The final dataset for the clinical trial reports is the preprocessed version that looks like the website-displayed version, and they are much more readable compared to the original data. More details and comparisons are shown in Table 4.1.

TABLE 4.1: Comparison of clinical trial report between before and after preprocessing

Category	**Report Content**
Before	{'protocolSection': {'identificationModule': {'nctId': 'NCT04207931', 'orgStudyIdInfo': {'id': 'IRB00043796'}, 'organization': {'fullName': 'Wake Forest University Health Sciences', 'class': 'OTHER'}, 'briefTitle': 'Treatment Results for Patients With Central Centrifugal Cicatricial Alopecia (CCCA): a Multicenter Prospective Study', 'officialTitle': 'Treatment Results for Patients With Central Centrifugal Cicatricial Alopecia (CCCA)
After	# Introduction Title: Treatment Results for Patients With Central Centrifugal Cicatricial Alopecia (CCCA): a Multicenter Prospective Study ClinicalTrials.gov ID: NCT04207931 Information provided by: Wake Forest University Health Sciences Information Provider: Wake Forest University Health Sciences Last Update Posted: 2024-01-30...

The second dataset collected is the related publication dataset. It is noticeable that for some of the clinical registration reports, there is a related publication section that contains citations of background, derived results, and unlabeled studies related to clinical trial registration. Each related publication contains its PMID, which can be used to retrieve the abstract content from the PubMed website directly. There are in total, 1401401 related publications collected in the dataset for this study. I also converted the available data from the original clinical trial data into human-readable content. The detailed comparisons are listed in Table 4.2.

TABLE 4.2: Comparison of Related Publication Between Before and After Preprocessing

Category	**Report Content**
Before	'referencesModule': {'references': [{'pmid': '1739290', 'type': 'BACKGROUND', 'citation': "Sperling LC, Sau P. The follicular degeneration syndrome in black patients. 'Hot comb alopecia' revisited and revised. Arch Dermatol. 1992 Jan;128(1):68-74."},
After	BACKGROUND Reference - Abstract: Arch Dermatol. 1992 Jan;128(1):68-74. The follicular degeneration syndrome in black patients. 'Hot comb alopecia' revisited and revised....

The final dataset used in the continual pre-training process is the concatenated dataset from the above two datasets. I merged the clinical trial registration report with the related publication together based on their acid and separated them by special tokens which represent the beginning of reference content and the ending of the reference content: <begin_of_ref> and <end_of_ref>. I hope the model can identify the difference between the target report and the reference context from the training process. I named the final dataset the CiTi dataset. The example of the collected data is presented in Table 4.3.

TABLE 4.3: Final Organized CiTi Dataset Lookup

Category	**Content**
Report	# Introduction Title: Treatment Results for Patients With Central Centrifugal Cicatricial Alopecia (CCCA): a Multicenter Prospective Study ClinicalTrials.gov ID: NCT04207931 Information provided by: Wake Forest University Health Sciences Information Provider: Wake Forest University Health Sciences Last Update Posted: 2024-01-30...
Publication	<begin_of_ref>BACKGROUND Reference - Abstract: Arch Dermatol. 1992 Jan;128(1):68-74. The follicular degeneration syndrome in black patients. 'Hot comb alopecia' revisited and revised.... <end_of_ref>

During the instruction fine-tuning process, I implemented one dataset called alpaca-gpt4-en [77] for all 3 models. It is a public-available instruction dataset containing 52000 conversational instructions for training purposes. It is a general-purpose instruction dataset, and it does not contain any medical instruction.

During the evaluation process, I implemented 3 datasets for all 3 models. These 3 datasets are also public-available datasets implemented by the previous studies for training and evaluation purposes.

The first evaluation dataset is the PubMedQA dataset [41]. It is a biomedical research question-answering dataset collected from PubMed abstracts. It contains 1000 True or False questions in the evaluation subset. The speciality of this dataset is that it contains the context section. Therefore, the model could perform in-context learning from the information provided. The questions in this dataset are highly related to the CiTi dataset.

The second dataset is the MedQA dataset [40]. It is a multiple-choice question dataset collected from the professional medical board exams which is related to the medical domain. Only the test subset is used for the evaluation process. The test subset contains 1270 multiple-choice questions. Each question contains 4 options, and only one is the ground truth answer. The ground truth label is presented as A, B, C and D. It does not provide the related context for the question. Therefore, the model response relies on the model itself.

The second dataset is the MedMCQA dataset [74]. It is a multiple-choice question dataset collected from the AIIMS & NEET PG entrance exam, which is related to the healthcare and medical domain. I took the first 500 samples of its validation subset for the evaluation process. It contains over 4180 multiple-choice questions. Each question contains 4 options, and only one is the ground truth answer. The ground truth label is 0, 1, 2 and 3. Similar to the MedQA dataset, it does not have the context as well. Therefore, the model response also relies on the model itself for this dataset.

4.3.2 Training details

This study has implemented two steps in the training process: continual pre-training towards domain adaptation and instruction fine-tuning, as shown in Figure 4.1.

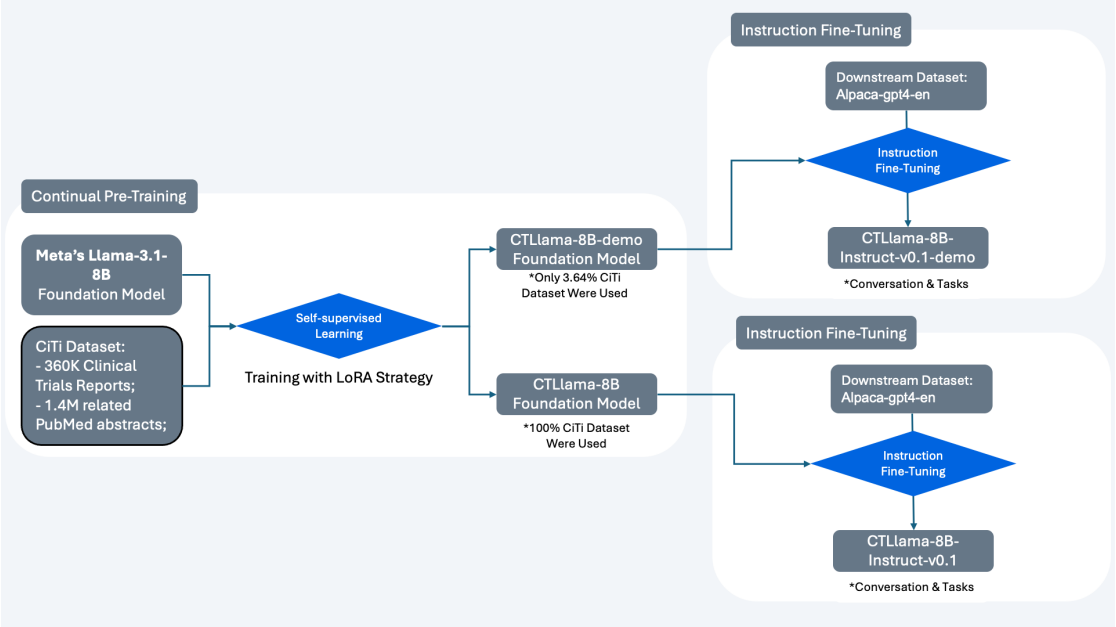

FIGURE 4.1: Model Training Pipeline

The base models used in this study are Llama-3.1-8B. Compared to its predecessor, the Llama-3-8B model, the Llama-3.1-8B has expanded the context length from 8K to 128K natively. Therefore, there is no need for us to implement progressive learning to further expand the context length to fit the clinical trial registration reports and potentially related publications. However, it is still a question whether the model can fully adopt the long context length after training.

This study implemented the novel CiTi dataset mentioned above for the continual pre-training process. The main purpose of the continual pre-training process is to inject knowledge into the model so that the model has a better understanding of the clinical trial domain. The trained model should perform better in the medicine and health domains and inference more efficiently.

To address the high training cost and catastrophic forgetting, the LoRA strategy was implemented and adjusted the training epoch to 1 in the continual pre-training process. The training log suggested that the total trainable parameters for the Llama-3.1-8B model are around 9.1 billion, and the LoRA strategy reduced the workload to around 1 billion parameters with only 11.8% of the total model parameters. It provides a feasible solution to therequirement in this study without any significant drawbacks. More details of training hyperparameters are presented in Table 4.4.

In the training experiment, I first selected a small portion of data from the CiTi dataset by setting "Max Sample" as 50,000. The selected subset represents only 3.64% of the whole CiTi dataset. This subset is then be used to construct a demo model named "CTLlama-8B-demo". Then, the full dataset was applied to the continual pre-training process and produced the fully trained model named "CTLlama-8B".

The purposes of this training process are one, to demonstrate the impact of the continual pre-training process due to the model; two, to provide a more detailed experiment result in the later evaluation process; and last, to test and make sure the training framework is working and compatible with the training platform.

TABLE 4.4: Model training hyperparameters during continual pre-training process

Training Parameter	**Demo model**	**Full model**
Fine-tuning Method	LoRA	LoRA
Target Layers	all	all
Max Samples	50000	100000000
Cutoff Length	8192	8192
Initial Learning Rate	0.00001	0.00001
Learning Rate Scheduler	cosine	cosine
Training Epochs	1	1

This study used the cloud-based AWS platform as the hardware environment for all training processes. The instance used for the continual pre-training process is the AWS ml.p4de.24xlarge instance includes 8 A100 GPUs.

Then I conducted the instruction fine-tuning process for all three models. The purpose of the instruction fine-tuning process is to give the model the ability to "communicate" with the users.

I did not collect the instruction question-answering dataset for the study because the purpose of this study is to construct the base model. However, it is essential to run the instruction fine-tuning process for evaluation. Therefore, I used the public-available Alpaca-gpt4-en dataset [77] for the instruction fine-tuning process in this study. The result models can conduct communications with users and generate human-readable responses accordingly. Because I only conducted the most basic instruction fine-tuning process, the evaluation result does not reflect the absolute performance of any of those models; instead, it could demonstrate the relative comparison between the models. The instance I use for the instruction fine-tuning process is the AWS ml.g5.12xlarge instance with 4 A10G GPUs.

Similar to the continual pre-training process, I implemented the same instruction fine-tuning process for the original Llama-3.1-8B model, the CTLlama-8B-demo model and the CTLlama-8B model. The purpose of repeating this step is to provide a fair and direct comparison of the model in the evaluation phase.

4.3.3 Evaluation

The evaluation process was implemented in the local environment that consisted of 1 RTX 3090 GPU with 24 GB of VRAM. The 4-bit quantization configuration was added to all models in the inference phase to eliminate the VRAM consumption overflow during the evaluation process. Since this evaluation aims to observe the relative performance differences among the three models, the performance drop due to the 4-bit quantization configuration is negligible in this case. The evaluation is then conducted for all three models using the PubMedQA, MedQA and MedMCQA datasets with F1-score as the evaluation metric. Similar to the previous study in Chapter 3, I also encountered the issue where the model response is not in the exact form I expected. Therefore, this study again implemented the same LLM-based evaluation for all evaluation results to extract and organize the answer in the desired form (Yes/No for PubMedQA dataset, and A/B/C/D for MedQA and MedMCQA datasets).

4.4 Results

The evaluation results of each model are presented in Table 4.5. The numbers represented the model performance in terms of accuracy in each evaluation dataset between the baseline Llama-3.1-8B model, the CTLlama-8B-demo and the CTLlama-8B model.

The result shows a stable relationship across different datasets and the 3 models. The continual pre-trained CTLlama-8B model has achieved the highest accuracy in all 3 datasets among the 3 models. The slightly trained CTLlama-8b-demo model reached second place in all 3 datasets. The baseline Llama-3.1-8B model only scores the lowest results in all 3 datasets.

Compared to the base model Llama-3.1-8B, the continual pre-training process provided a performance gain of 57.9%, 3.3% and 75% for the CTLlama-8B model with respect to the PubMedQA, MedQA and MedMCQA datasets. On average, it provided a 36.3% of performance gain to the CTLlama-8B model.

TABLE 4.5: Result Demonstration Across 3 Models

Model*	**PubMedQA Marco-F1**	**MedQA Marco-F1**	**MedMCQA Marco-F1**	**Average Marco-F1**
Llama-3.1-8B	0.228	0.459	0.264	0.317
CTLlama-8B-demo	0.327	0.459	0.377	0.388
CTLlama-8B	**0.360**	**0.474**	**0.462**	**0.432**

*All models have implemented 4-bit Quantization for the inference to save the GPU memory consumption.

4.5 Discussion

4.5.1 Result discussion

Overall, the result supports the assumption of the importance of the continual pre-training process. The result presented a clear positive relationship between the number of samples used in the continual pre-training process and the accuracy score presented in the table. The results also indicate that the CTLlama-8B model outperforms the original Llama-3.1-8B model in

the PubMedQA evaluation. The 3 models have shown similar gaps in the MedMCQA dataset, where the baseline Llama-3.1-8B model slightly falls behind the CTLlama-8B model. A similar situation is presented in the MedQA dataset, but the gaps between the 3 models are smaller.

The CTLlama-8B-demo model presented a closer result to the CTLlama-8B model in the PubMedQA dataset and a closer result to the other two datasets. This relationship indicates that even a small amount of continual pre-training could help the model gain better results in in-context learning. However, more knowledge is still needed in scenarios of zero-shot learning like MedQA or MedMCQA. The result in these two datasets demonstrated that there is a notable improvement for the CTLlama-8B-demo compared to the Llama-3.1-8B by only using 3% of the CiTi dataset in the training process, but the overall result reflected that it still needs more knowledge injection to fully adapt the clinical trial registration content.

The CiTi dataset contains information collected from ClinicalTrials.gov and PubMed websites. However, the alpaca-gpt4-en dataset does not contain medicine and health domain information, and the instruction fine-tuning process was implemented with the same training hyperparameters on all 3 models. Therefore, the evaluation process in this study has presented a fair comparison of the relative performance between the 3 models, and the increasing accuracy shown in the result can only come from the continual pre-training process, indicating its importance for domain-specific models.

4.5.2 Future works

The result indicates that continual pre-training is an important phase in formulating the domain-specific model. However, instruction fine-tuning is still highly relevant to the actual task performance of the models. Another interesting finding is that the result indicates that the model only needs a small amount of continual pre-training to benefit from the in-context learning. The future opportunity could be to evaluate the balance between training and performance so that the models can gain better results in the retrieval argumentation generation frameworks.

4.6 Conclusion

This study proposes a new domain-specific LLM for this study to better understand the clinical trial data. This study shows that generalized LLMs can benefit from continual pre-training in domain-specific tasks. The result indicates that the continual pre-training does look like a knowledge injection stage for the model because two out of three datasets are not directly related to the training dataset, and for the one dataset, PubMedQA, that is related to the training dataset, I did not fully collect the data from PubMed website as well. The evidence has suggested that continual pre-training is a great learning opportunity for the domain-specific model.

CHAPTER 5

Discussion

5.1 Summary of aim and findings

The primary aim and objectives of this thesis consist of three main areas. Firstly, it addresses the challenges of designing LLM-based clinical applications for local language models to process long clinical content. Secondly, the work explores the methods and pipeline of RAG frameworks, mainly through the RAPTOR framework. Additionally, this thesis wants to tackle and examine the insight of constructing a domain-specific language model for better and more efficient clinical application.

Chapter 3 introduced an external solution for the local language model implementation, which modified the existing RAG-based framework RAPTOR with new chunking strategies and embedding models to optimize the scenarios of adapting local language models to the framework. The experiment result in this study showed that, by modifying the chunking strategy, the framework improved 2-3% of accuracy on average for both the cloud-based GPT-3.5-Turbo model and the local inference Mistral-7B-Instruct-v0.2 model. For the result in complex questions of the CTQA dataset, it even surpasses the GPT-4o model by 7-8% of accuracy on average.

Chapter 4 explored the internal solution to the question by constructing a new domain-specific model, CTLlama-8B, that has a better understanding of the clinical trial registration data. The evaluation result indicated that the continual pre-training process in the study successfully helped the original model adapt to the domain knowledge. The number showed that the average accuracy of CTLlama-8B in the evaluation result is doubled compared to the

original Llama-3.1-8B. It presented the importance of the continual pre-training process when constructing a domain-specific large language model.

5.2 Implications of future research

This thesis has discussed and demonstrated a clinical question-answering framework modified from the RAPTOR framework for better local language model implementation. It also showcased a new domain-specific model optimized for the clinical trial registration data. A future opportunity would be first to continue the instruction fine-tuning and reinforcement learning procedures for the model to build a complete conversational large language model based on the successful continual pre-trained CTLlama-8B, and then proceed to the development of external RAG-based framework for quickly updating the domain information and knowledge.

The results in Chapter 3 showed that the RAPTOR framework with a locally implemented open language model outperformed the much larger general GPT-4, but only in certain complex questions. This was likely because of how RAPTOR synthesizes local and global information in a hierarchical form. Investigations into performance differences across different types of simple and complex questions are largely unexplored. Chapter 3 only proposed the fundamental way of distinguishing simple and complex questions, where the questions that take more steps in human logic would be considered complex questions. Future research could consider classifying the complexity of questions with more details before determining the specific configuration that is most suitable, where complex questions are answered using different configurations.

Chapter 4 showcased the language model that combines the existing generalized LLM with the data from ClinicalTrials.gov, and the evaluation result indicated that it could demonstrate some exciting performance in medicine and health domain questions. ClinicalTrials.gov, as an underutilised but important data source, would be an important future opportunity by itself. The development of LLMs based on it will have a huge potential to improve the design, efficiency, and synthesis of clinical trials.

CHAPTER 6

Conclusion

This thesis explored and discussed the challenges and solutions for building and optimizing local language models for domain applications using RAG-based frameworks and domain-specific LLMs. The introduction provides a general background of the electronic medical record data and the clinical trial registration data. It also discussed the challenges and opportunities of dealing with these databases, where researchers and general users would benefit from the LLMs to simplify messages, extract information and generate insights from the original data, but they would require the model as the local language model or "on-device AI" due to data privacy concerns or limited internet access in the workplace.

A clinical question-answering framework based on the existing RAPTOR framework was introduced in the study. It is designed to extract information and generate insights for complex questions using the local language models or "on-device AI", providing a suitable solution for users who have limited access to the cloud-based LLMs and who have concerns about violating the data safety agreement. During the experiment, the RAPTOR framework was deployed and tested locally to demonstrate the capability of local language models within the limited environment that prioritize data security issues and are disconnected from cloud-based LLMs. The study also introduced a new chunking strategy to the RAPTOR framework and switched the supported embedding models to the BGE-M3 model, optimized for RAG-related retrieval tasks. The benchmarking results showed that the modified RAPTOR framework achieved efficient retrieval of clinical information from the novel clinical trial registration dataset while maintaining high accuracy that surpass the GPT-4o in the complex questions. It also showed that the new chunking strategy and embedding models could help the RAPTOR framework achieve better results in both clinical and normal literature domains.

In the next study, I proposed a new modified model named CTLlama-8B. It is a domain-specific model specially designed and trained to target the clinical trial registration data. It was built on top of the existing Llama-3.1-8B model, where the continual pre-training process was implemented in the study to build the new model. The dataset used in this study was novelly collected and preprocessed from the ClinicalTrials.gov and PubMed websites. The evaluation was conducted using 3 public-available datasets that cover medicine and health domain questions related to the training dataset without duplication. The results showed that continual pre-training could help the model achieve domain adaptation without question, and it should be considered a necessary step to build a domain-specific LLM.

Bibliography

[1] URL: https://docs.llamaindex.ai/en/stable/examples/node_parsers/semantic_chunking/.

[2] A. Aali et al. *MIMIC-IV-Ext-BHC: Labeled Clinical Notes Dataset for Hospital Course Summarization (version 1.1.0)*. Accessed: 2024-10-22. 2024. URL: https://doi.org/10.13026/41et-8342.

[3] Josh Achiam et al. 'Gpt-4 technical report'. In: *arXiv preprint arXiv:2303.08774* (2023).

[4] AI@Meta. 'Llama 3 Model Card'. In: (2024). URL: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

[5] Akari Asai 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.

[6] Jinze Bai et al. *Qwen Technical Report*. 2023. arXiv: 2309.16609 [cs.CL]. URL: https://arxiv.org/abs/2309.16609.

[7] Suhana Bedi et al. 'Testing and Evaluation of Health Care Applications of Large Language Models: A Systematic Review'. In: *JAMA* (2024).

[8] Tom Brown et al. 'Language models are few-shot learners'. In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[9] Jianlv Chen et al. 'Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation'. In: *arXiv preprint arXiv:2402.03216* (2024).

[10] Lingjiao Chen, Matei Zaharia and James Zou. *FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance*. 2023. arXiv: 2305.05176 [cs.LG]. URL: https://arxiv.org/abs/2305.05176.

[11] Tong Chen et al. ‘Dense X Retrieval: What Retrieval Granularity Should We Use?’ In: *arXiv preprint arXiv:2312.06648* (2023).

[12] Yukang Chen et al. *LongLoRA: Efficient Fine-tuning of Long-Context Large Language Models*. 2024. arXiv: 2309.12307 [cs.CL]. URL: https://arxiv.org/abs/2309.12307.

[13] Clément Christophe et al. *Med42-v2: A Suite of Clinical LLMs*. 2024. arXiv: 2408.06142 [cs.CL]. URL: https://arxiv.org/abs/2408.06142.

[14] Gustavo Alberto Córdova González. ‘Electronic health records: its effects on the doctor-patient relationship and the role of the computer in the clinical setting’. In: *Health and Technology* 12.2 (2022), pp. 305–311.

[15] Badhan Chandra Das, M Hadi Amini and Yanzhao Wu. ‘Security and privacy challenges of large language models: A survey’. In: *arXiv preprint arXiv:2402.00888* (2024).

[16] Tim Dettmers et al. ‘Qlora: Efficient finetuning of quantized llms’. In: *Advances in Neural Information Processing Systems* 36 (2024).

[17] Abhimanyu Dubey et al. ‘The llama 3 herd of models’. In: *arXiv preprint arXiv:2407.21783* (2024).

[18] Darren Edge et al. ‘From local to global: A graph rag approach to query-focused summarization’. In: *arXiv preprint arXiv:2404.16130* (2024).

[19] Magdalyn E Elkin and Xingquan Zhu. ‘Predictive modeling of clinical trial terminations using feature engineering and embedding learning’. In: *Scientific reports* 11.1 (2021), p. 3446.

[20] Queen Elizabeth Enahoro et al. ‘The impact of electronic health records on healthcare delivery and patient outcomes: A review’. In: *World Journal of Advanced Research and Reviews* 21.2 (2024), pp. 451–460.

[21] Jean-Baptiste Excoffier et al. ‘Generalist embedding models are better at short-context clinical semantic search than specialized embedding models’. In: *arXiv preprint arXiv:2401.01943* (2024).

[22] Amar Fadillah, Nuke Athahirah and Kuan Ting Lai. 'Chunking Strategy for Retrieval Augmented Generation in Regulation Documents'. In: *2024 International Conference on Consumer Electronics-Taiwan (ICCE-Taiwan)*. IEEE. 2024, pp. 279–280.

[23] Masoomali Fatehkia, Ji Kim Lucas and Sanjay Chawla. 'T-RAG: lessons from the LLM trenches'. In: *arXiv preprint arXiv:2402.07483* (2024).

[24] Muhammad Fawi. *CURLoRA: Stable LLM Continual Fine-Tuning and Catastrophic Forgetting Mitigation*. en. 2024. DOI: 10.5281/ZENODO.12730055. URL: https://zenodo.org/doi/10.5281/zenodo.12730055.

[25] Yunfan Gao et al. *Retrieval-Augmented Generation for Large Language Models: A Survey*. 2023. arXiv: 2312.10997 [cs.CL].

[26] Aryo Pradipta Gema et al. *Parameter-Efficient Fine-Tuning of LLaMA for the Clinical Domain*. 2024. arXiv: 2307.03042 [cs.CL]. URL: https://arxiv.org/abs/2307.03042.

[27] Zelalem Gero et al. *Self-Verification Improves Few-Shot Clinical Information Extraction*. 2023. arXiv: 2306.00024 [cs.CL]. URL: https://arxiv.org/abs/2306.00024.

[28] Lawrence O Gostin, Laura A Levit and Sharyl J Nass. 'Beyond the HIPAA privacy rule: enhancing privacy, improving health through research'. In: (2009).

[29] Paromita Goswami et al. 'Investigation on storage level data integrity strategies in cloud computing: classification, security obstructions, challenges and vulnerability'. In: *Journal of Cloud Computing* 13.1 (2024), p. 45.

[30] Michael Günther et al. *Jina Embeddings 2: 8192-Token General-Purpose Text Embeddings for Long Documents*. 2023. arXiv: 2310.19923 [cs.CL].

[31] Kshitij Gupta et al. *Continual Pre-Training of Large Language Models: How to (re)warm your model?* 2023. arXiv: 2308.04014 [cs.CL]. URL: https://arxiv.org/abs/2308.04014.

[32] Paul Hager et al. 'Evaluation and mitigation of the limitations of large language models in clinical decision-making'. In: *Nature medicine* 30.9 (2024), pp. 2613–2622.

[33] Helia Hashemi et al. 'LLM-Rubric: A Multidimensional, Calibrated Approach to Automated Evaluation of Natural Language Texts'. In: *Proceedings of the 62nd*

Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2024, pp. 13806–13834.

[34] Amber E Hoek et al. 'Patient discharge instructions in the emergency department and their effects on comprehension and recall of discharge instructions: a systematic review and meta-analysis'. In: *Annals of emergency medicine* 75.3 (2020), pp. 435–444.

[35] Cheng-Ping Hsieh et al. 'RULER: What's the Real Context Size of Your Long-Context Language Models?' In: *arXiv preprint arXiv:2404.06654* (2024).

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

[37] Abigail E Huang et al. 'Clinical documentation in electronic health record systems: analysis of similarity in progress notes from consecutive outpatient ophthalmology encounters'. In: *AMIA Annual Symposium Proceedings*. Vol. 2018. American Medical Informatics Association. 2018, p. 1310.

[38] Ryan S Huang et al. 'The future of AI clinicians: assessing the modern standard of chatbots and their approach to diagnostic uncertainty'. In: *BMC Medical Education* 24.1 (2024), p. 1133.

[39] Albert Q. Jiang and et al. *Mistral 7B*. 2023. arXiv: 2310.06825 [cs.CL]. URL: https://arxiv.org/abs/2310.06825.

[40] Di Jin et al. 'What Disease does this Patient Have? A Large-scale Open Domain Question Answering Dataset from Medical Exams'. In: *arXiv preprint arXiv:2009.13081* (2020).

[41] Qiao Jin et al. *PubMedQA: A Dataset for Biomedical Research Question Answering*. 2019. arXiv: 1909.06146 [cs.CL]. URL: https://arxiv.org/abs/1909.06146.

[42] Jared Kaplan et al. *Scaling Laws for Neural Language Models*. 2020. arXiv: 2001.08361 [cs.LG]. URL: https://arxiv.org/abs/2001.08361.

[43] Zixuan Ke et al. *Continual Pre-training of Language Models*. 2023. arXiv: 2302.03241 [cs.CL]. URL: https://arxiv.org/abs/2302.03241.

[44] M Abdul Khaliq et al. 'RAGAR, Your Falsehood RADAR: RAG-Augmented Reasoning for Political Fact-Checking using Multimodal Large Language Models'. In: *arXiv preprint arXiv:2404.12065* (2024).

[45] Takeshi Kojima et al. *Large Language Models are Zero-Shot Reasoners*. 2023. arXiv: 2205.11916 [cs.CL]. URL: https://arxiv.org/abs/2205.11916.

[46] Karmela Krleža-Jerić, Mersiha Mahmić-Kaknjo and Khaled El Emam. 'Clinical trial registries, results databases, and research data repositories'. In: *Clinical Research Informatics*. Springer, 2023, pp. 329–363.

[47] Prerana Sanjay Kulkarni et al. *HeCiX: Integrating Knowledge Graphs and Large Language Models for Biomedical Research*. 2024. arXiv: 2407.14030 [cs.CL]. URL: https://arxiv.org/abs/2407.14030.

[48] Sunjun Kweon et al. *CAMEL : Clinically Adapted Model Enhanced from LLaMA*. https://github.com/starmpcc/CAMEL. May 2023.

[49] Harrison Lee et al. 'Rlaif: Scaling reinforcement learning from human feedback with ai feedback'. In: *arXiv preprint arXiv:2309.00267* (2023).

[50] Jinhyuk Lee et al. 'BioBERT: a pre-trained biomedical language representation model for biomedical text mining'. In: *Bioinformatics* 36.4 (2020), pp. 1234–1240.

[51] McInnes Leland, Healy John and Melville James. 'Uniform manifold approximation and projection for dimension reduction'. In: *arXiv preprint arXiv:1802.03426* (2018).

[52] Patrick Lewis et al. *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. 2021. arXiv: 2005.11401 [cs.CL]. URL: https://arxiv.org/abs/2005.11401.

[53] Patrick Lewis et al. 'Retrieval-augmented generation for knowledge-intensive nlp tasks'. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 9459–9474.

[54] Chen-An Li and Hung-Yi Lee. *Examining Forgetting in Continual Pre-training of Aligned Large Language Models*. 2024. arXiv: 2401.03129 [cs.CL]. URL: https://arxiv.org/abs/2401.03129.

[55] Yunxiang Li et al. *ChatDoctor: A Medical Chat Model Fine-Tuned on a Large Language Model Meta-AI (LLaMA) Using Medical Domain Knowledge*. 2023. arXiv: 2303.14070 [cs.CL]. URL: https://arxiv.org/abs/2303.14070.

[56] Fenglin Liu et al. 'Large Language Models Are Poor Clinical Decision-Makers: A Comprehensive Benchmark'. In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yaser Al-Onaizan, Mohit Bansal and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 13696–13710. DOI: 10.18653/v1/2024.emnlp-main.759. URL: https://aclanthology.org/2024.emnlp-main.759/.

[57] Nelson F. Liu et al. *Lost in the Middle: How Language Models Use Long Contexts*. 2023. arXiv: 2307.03172 [cs.CL]. URL: https://arxiv.org/abs/2307.03172.

[58] Yang Liu et al. 'Datasets for large language models: A comprehensive survey'. In: *arXiv preprint arXiv:2402.18041* (2024).

[59] Andrew W Lo, Kien Wei Siah and Chi Heem Wong. *Machine learning with statistical imputation for predicting drug approvals*. Vol. 60. 10.1162. SSRN, 2019.

[60] Bowen Long et al. 'Predicting Phase 1 Lymphoma Clinical Trial Durations Using Machine Learning: An In-Depth Analysis and Broad Application Insights'. In: *Clinics and Practice* 14.1 (2023), pp. 69–88.

[61] Michal Lukasik et al. 'Text segmentation by cross segment attention'. In: *arXiv preprint arXiv:2004.14535* (2020).

[62] Kun Luo et al. 'BGE Landmark Embedding: A Chunking-Free Embedding Method For Retrieval Augmented Long-Context Large Language Models'. In: *arXiv preprint arXiv:2402.11573* (2024).

[63] Bertalan Meskó and Eric J Topol. 'The imperative for regulatory oversight of large language models (or generative AI) in healthcare'. In: *NPJ digital medicine* 6.1 (2023), p. 120.

[64] David Mikhail et al. 'Performance of DeepSeek-R1 in Ophthalmology: An Evaluation of Clinical Decision-Making and Cost-Effectiveness'. In: *medRxiv* (2025), pp. 2025–02.

[65] Shervin Minaee et al. ‘Large language models: A survey’. In: *arXiv preprint arXiv:2402.06196* (2024).

[66] Tsendsuren Munkhdalai, Manaal Faruqui and Siddharth Gopal. ‘Leave no context behind: Efficient infinite context transformers with infini-attention’. In: *arXiv preprint arXiv:2404.07143* (2024).

[67] Humza Naveed et al. *A Comprehensive Overview of Large Language Models*. 2024. arXiv: 2307.06435 [cs.CL]. URL: https://arxiv.org/abs/2307.06435.

[68] Suvendu Kumar Nayak and Ananta Charan Ojha. ‘Data leakage detection and prevention: Review and research directions’. In: *Machine Learning and Information Processing: Proceedings of ICMLIP 2019* (2020), pp. 203–212.

[69] Arvind Neelakantan and et al. *Text and Code Embeddings by Contrastive Pre-Training*. 2022. arXiv: 2201.10005 [cs.CL]. URL: https://arxiv.org/abs/2201.10005.

[70] Harsha Nori et al. *Capabilities of GPT-4 on Medical Challenge Problems*. 2023. arXiv: 2303.13375 [cs.CL]. URL: https://arxiv.org/abs/2303.13375.

[71] Harsha Nori et al. ‘Capabilities of gpt-4 on medical challenge problems’. In: *arXiv preprint arXiv:2303.13375* (2023).

[72] OpenAI et al. *GPT-4 Technical Report*. 2024. arXiv: 2303.08774 [cs.CL]. URL: https://arxiv.org/abs/2303.08774.

[73] Long Ouyang et al. *Training language models to follow instructions with human feedback*. 2022. arXiv: 2203.02155 [cs.CL]. URL: https://arxiv.org/abs/2203.02155.

[74] Ankit Pal, Logesh Kumar Umapathi and Malaikannan Sankarasubbu. ‘MedMCQA: A Large-scale Multi-Subject Multi-Choice Dataset for Medical domain Question Answering’. In: *Proceedings of the Conference on Health, Inference, and Learning*. Ed. by Gerardo Flores et al. Vol. 174. Proceedings of Machine Learning Research. PMLR, July 2022, pp. 248–260. URL: https://proceedings.mlr.press/v174/pal22a.html.

[75] Richard Yuanzhe Pang et al. 'QuALITY: Question Answering with Long Input Texts, Yes!' In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 5336–5358. URL: https://aclanthology.org/2022.naacl-main.391.

[76] Jupinder Parmar et al. 'Reuse, Don't Retrain: A Recipe for Continued Pretraining of Language Models'. In: *arXiv preprint arXiv:2407.07263* (2024).

[77] Baolin Peng et al. *Instruction Tuning with GPT-4*. 2023. arXiv: 2304.03277 [cs.CL]. URL: https://arxiv.org/abs/2304.03277.

[78] Aman Priyanshu, Yash Maurya and Zuofei Hong. 'AI Governance and Accountability: An Analysis of Anthropic's Claude'. In: *arXiv preprint arXiv:2407.01557* (2024).

[79] *Protocol Registration Data Element Definitions for Interventional and Observational Studies*. Last updated on June 17, 2024. 2024. URL: http://https://clinicaltrials.gov/policy/protocol-definitions.

[80] Hongjin Qian et al. 'Are Long-LLMs A Necessity For Long-Context Tasks?' In: *arXiv preprint arXiv:2405.15318* (2024).

[81] Hongjin Qian et al. 'Grounding Language Model with Chunking-Free In-Context Retrieval'. In: *arXiv preprint arXiv:2402.09760* (2024).

[82] Alec Radford et al. *Improving language understanding by generative pre-training.(2018)*. 2018.

[83] Rafael Rafailov et al. 'Direct preference optimization: Your language model is secretly a reward model'. In: *Advances in Neural Information Processing Systems* 36 (2024).

[84] N Reimers. 'Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks'. In: *arXiv preprint arXiv:1908.10084* (2019).

[85] Michael Reinisch et al. *CTP-LLM: Clinical Trial Phase Transition Prediction Using Large Language Models*. 2024. arXiv: 2408.10995 [cs.CL]. URL: https://arxiv.org/abs/2408.10995.

[86] Douglas A Reynolds et al. 'Gaussian mixture models.' In: *Encyclopedia of biometrics* 741.659-663 (2009).

[87] Soumyadeep Roy et al. 'Beyond Accuracy: Investigating Error Types in GPT-4 Responses to USMLE Questions'. In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR 2024. ACM, July 2024, pp. 1073–1082. DOI: 10.1145/3626772.3657882. URL: http://dx.doi.org/10.1145/3626772.3657882.

[88] Adam Rule et al. 'Length and redundancy of outpatient progress notes across a decade at an academic medical center'. In: *JAMA Network Open* 4.7 (2021), e2115334–e2115334.

[89] Khaled Saab et al. 'Capabilities of gemini models in medicine'. In: *arXiv preprint arXiv:2404.18416* (2024).

[90] Parth Sarthi et al. 'RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval'. In: *International Conference on Learning Representations (ICLR)*. 2024.

[91] John Schulman et al. *Proximal Policy Optimization Algorithms*. 2017. arXiv: 1707.06347 [cs.LG]. URL: https://arxiv.org/abs/1707.06347.

[92] Thomas Scialom, Tuhin Chakrabarty and Smaranda Muresan. *Fine-tuned Language Models are Continual Learners*. 2022. arXiv: 2205.12393 [cs.CL]. URL: https://arxiv.org/abs/2205.12393.

[93] Uri Shaham et al. 'Zeroscrolls: A zero-shot benchmark for long text understanding'. In: *arXiv preprint arXiv:2305.14196* (2023).

[94] Shamane Siriwardhana et al. *Domain Adaptation of Llama3-70B-Instruct through Continual Pre-Training and Model Merging: A Comprehensive Evaluation*. 2024. arXiv: 2406.14971 [cs.CL]. URL: https://arxiv.org/abs/2406.14971.

[95] Jackson Steinkamp, Jacob J Kantrowitz and Subha Airan-Javia. 'Prevalence and sources of duplicate information in the electronic medical record'. In: *JAMA network open* 5.9 (2022), e2233348–e2233348.

[96] *Study Data Structure*. Last updated on April 01, 2024. 2024. URL: https://clinicaltrials.gov/data-api/about-api/study-data-structure.

[97] Hui Su et al. *Unraveling the Mystery of Scaling Laws: Part I*. 2024. arXiv: 2403.06563 [cs.LG]. URL: https://arxiv.org/abs/2403.06563.

[98] Soshi Takagi et al. 'Performance of GPT-3.5 and GPT-4 on the Japanese medical licensing examination: comparison study'. In: *JMIR Medical Education* 9.1 (2023), e48002.

[99] Chao-Hong Tan et al. 'Ponet: Pooling network for efficient token mixing in long sequences'. In: *arXiv preprint arXiv:2110.02442* (2021).

[100] Gemini Team et al. *Gemini: A Family of Highly Capable Multimodal Models*. 2024. arXiv: 2312.11805 [cs.CL]. URL: https://arxiv.org/abs/2312.11805.

[101] Gemini Team et al. 'Gemini: a family of highly capable multimodal models'. In: *arXiv preprint arXiv:2312.11805* (2023).

[102] Hugo Touvron and et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. 2023. arXiv: 2307.09288 [cs.CL]. URL: https://arxiv.org/abs/2307.09288.

[103] Hugo Touvron et al. 'Llama 2: Open foundation and fine-tuned chat models'. In: *arXiv preprint arXiv:2307.09288* (2023).

[104] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971 [cs.CL]. URL: https://arxiv.org/abs/2302.13971.

[105] Ehsan Ullah et al. 'Challenges and barriers of using large language models (LLM) such as ChatGPT for diagnostic medicine with a focus on digital pathology–a recent scoping review'. In: *Diagnostic Pathology* 19.1 (2024), pp. 1–9.

[106] Rheeya Uppaal, Yixuan Li and Junjie Hu. *How Useful is Continued Pre-Training for Generative Unsupervised Domain Adaptation?* 2024. arXiv: 2401.17514 [cs.CL]. URL: https://arxiv.org/abs/2401.17514.

[107] Paul Voigt and Axel Von dem Bussche. 'The eu general data protection regulation (gdpr)'. In: *A Practical Guide, 1st Ed., Cham: Springer International Publishing* 10.3152676 (2017), pp. 10–5555.

[108] Dandan Wang and Shiqing Zhang. 'Large language models in medical and healthcare fields: applications, advances, and challenges'. In: *Artificial Intelligence Review* 57.11 (2024), p. 299.

[109] Siyang Wang et al. 'Predicting publication of clinical trials using structured and unstructured data: model development and validation study'. In: *Journal of Medical Internet Research* 24.12 (2022), e38859.

[110] Jason Wei et al. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. 2023. arXiv: 2201.11903 [cs.CL]. URL: https://arxiv.org/abs/2201.11903.

[111] Jason Wei et al. *Emergent Abilities of Large Language Models*. 2022. arXiv: 2206.07682 [cs.CL]. URL: https://arxiv.org/abs/2206.07682.

[112] Jason Wei et al. 'Finetuned language models are zero-shot learners'. In: *arXiv preprint arXiv:2109.01652* (2021).

[113] Chaoyi Wu et al. *PMC-LLaMA: Towards Building Open-source Language Models for Medicine*. 2023. arXiv: 2304.14454 [cs.CL]. URL: https://arxiv.org/abs/2304.14454.

[114] Qianqian Xie et al. 'Me llama: Foundation large language models for medical applications'. In: *arXiv preprint arXiv:2402.12749* (2024).

[115] Yong Xie, Karan Aggarwal and Aitzaz Ahmad. 'Efficient Continual Pre-training for Building Domain Specific Large Language Models'. In: *Findings of the Association for Computational Linguistics: ACL 2024*. Ed. by Lun-Wei Ku, Andre Martins and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 10184–10201. DOI: 10.18653/v1/2024.findings-acl.606. URL: https://aclanthology.org/2024.findings-acl.606/.

[116] Honglin Xiong et al. *DoctorGLM: Fine-tuning your Chinese Doctor is not a Herculean Task*. 2023. arXiv: 2304.01097 [cs.CL]. URL: https://arxiv.org/abs/2304.01097.

[117] Weimin Xiong et al. *Rationale-Enhanced Language Models are Better Continual Relation Learners*. 2023. arXiv: 2310.06547 [cs.CL]. URL: https://arxiv.org/abs/2310.06547.

[118] Lingling Xu et al. *Parameter-Efficient Fine-Tuning Methods for Pretrained Language Models: A Critical Review and Assessment*. 2023. arXiv: 2312.12148 [cs.CL]. URL: https://arxiv.org/abs/2312.12148.

[119] Shi-Qi Yan et al. ‘Corrective retrieval augmented generation’. In: *arXiv preprint arXiv:2401.15884* (2024).

[120] Qimin Yang et al. *Fine-Tuning Medical Language Models for Enhanced Long-Contextual Understanding and Domain Expertise*. 2024. arXiv: 2407.11536 [cs.CL]. URL: https://arxiv.org/abs/2407.11536.

[121] Shunyu Yao et al. *ReAct: Synergizing Reasoning and Acting in Language Models*. 2023. arXiv: 2210.03629 [cs.CL]. URL: https://arxiv.org/abs/2210.03629.

[122] Antonio Jimeno Yepes et al. *Financial Report Chunking for Effective Retrieval Augmented Generation*. 2024. arXiv: 2402.05131 [cs.CL]. URL: https://arxiv.org/abs/2402.05131.

[123] Çağatay Yıldız et al. *Investigating Continual Pretraining in Large Language Models: Insights and Implications*. 2024. arXiv: 2402.17400 [cs.CL]. URL: https://arxiv.org/abs/2402.17400.

[124] Qinglin Zhang et al. ‘Sequence model with self-adaptive sliding window for efficient spoken document segmentation’. In: *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE. 2021, pp. 411–418.

[125] Jiean Zhu. ‘Cura-LLaMA: Evaluating open-source large language Model’s question answering capability on medical domain’. In: *Applied and Computational Engineering* 90 (2024), pp. 52–60.

1 Appendix

1.1 Evaluation Prompt Templates

Table .1 shows examples of 3-shot prompt templates used for response formatting and extraction purposes. These are demonstrated with 1 example for illustrative purposes; other examples are replaced by **example #N** in the table.

TABLE .1: 3-Shot prompt template examples

Task	Prompt
CTQA	You are a helpful assistant who can extract information. Response 'NaN' if answer is not existed. You always answer the question in the simplest way possible without adding any extra information. User: For the text below related to the Clinical Trials, What is the number of participants who experienced serious adverse events in the first arm/study? Text Content: In the results of this trial, no participants had serious adverse events in either study arm. Answer: Assistant: 0 **example #2** **example #3** For the following text in the bracket which related to the Clinical Trials, What is the number of participants who experienced serious adverse events in the first arm/study? Text Content:{response} Answer:
QuALITY	You are a helpful assistant who can extract information. Response 'NaN' if answer is not existed. You always answer the question in the simplest way possible without adding any extra information. User: Given the following model response and Options, I need you to extract the actual model response for me. The multiple choose question only have 4 answers: A,B,C,D. Model Response: "Answer: 6"; Options: ['8', '3', '6', '10']; Answer: Assistant: C **example #2** **example #3** User: The multiple choose question have 4 answers: A,B,C,D. Given the model response and the options, I need you to extract the actual model response for me. Model Response: {response}; Options: {options}; Answer: