Instruction Tuning for Domain Adaptation of Large Language Models

A Case Study in the Field of Education

Jiacheng Zhang

Delft University of Technology

TU Delft

Instruction Tuning for Domain Adaptation of Large Language Models

A Case Study in the Field of Education

by

Jiacheng Zhang

Jiacheng Zhang	5744814

Thesis committee:	Dr. Sole Pera	TU Delft, Thesis Advisor
	Dr. Jie Yang	TU Delft, Daily Supervisor
	Dr. Maliheh Izadi	TU Delft
	Gaole He	TU Delft, Daily Co-Supervisor
Project Duration:	11, 2023 - 7, 2024	
Faculty:	EEMCS, TU Delft	

Cover: Canadarm 2 Robotic Arm Grapples SpaceX Dragon by NASA under CC BY-NC 2.0 (Modified)

Preface

Artificial Intelligence is increasingly becoming an integral part of our daily lives. However, with its rapid advancement, a question arises: are human problems decreasing or increasing with the introduction of AI? This question can be a complex sociological, cognitive, and computer science problem. It may take a long time for humans to reach a point where AI, humans, and society coexist as equals. While there may not be much I can do to solve this now, it is time for me to leap forward. My journey at TU Delft has exposed me to the fascinating potential of LLMs. It seems these models could be a significant step toward solving many of humanity's problems. Motivated by this potential, I discussed my interests with Professor Jie Yang and Gaole He, and together we selected a topic that truly captivated me.

When I embarked on my research journey, I felt a little nervous due to my limited experience with tuning LLMs, a relatively new and complex field. However, as I delved deeper into learning and reading, I gradually gained confidence and expertise. This journey has been like sailing in an uncharted ocean; the tiny bright spot on the horizon could be just an island or a mainland. Through this thesis research, I have not only enhanced the learning capabilities of LLMs but also uncovered new areas for exploration in my future career. This process has been profoundly enlightening, revealing numerous possibilities for my professional growth and development. It has taught me resilience, adaptability, and the importance of continuous learning, which are invaluable lessons for my future endeavors.

I owe a great deal of gratitude to many people who have supported me throughout this journey. I am deeply grateful to Dr. Maria Soledad Pera for her invaluable guidance in helping me narrow down the research topic and for her willingness to answer my numerous questions. Her insights and expertise have been instrumental in shaping the direction of my research. I am deeply thankful to Professor Jie Yang for guiding me through the research process and for making me genuinely enjoy being part of this field. His dedication and expertise have been truly inspiring, making my journey in this research an incredibly rewarding experience. My sincere thanks go to Gaole He, my daily supervisor, whose unparalleled kindness and patience in answering my questions about the project have been truly invaluable. His constant support, thorough guidance, and remarkable dedication have been instrumental in my research journey and I am incredibly fortunate to have had his mentorship. I also want to extend my heartfelt thanks to my parents, who have supported and raised me for the past twenty years. Their encouragement and love have been the foundation of my journey, and I am deeply grateful for all they have done to help me reach this point. Special thanks to my best friend, Charlie, whom I met in middle school. His insights and perspectives have profoundly shaped my intellectual journey and fueled my passion for exploring new ideas. I am grateful for his enduring support and friendship. Lastly, I am grateful to my girlfriend, whose constant companionship and humor have been a source of joy and motivation. I am incredibly fortunate to have her by my side, and her presence has made this experience all the more meaningful and fulfilling.

Jiacheng Zhang
Delft, July 2024

Abstract

While most large language models (LLMs) are powerful, they are primarily designed for general purposes. Consequently, many enterprises and institutions have now focused on developing domain-specific models. In the realm of education, an expert LLM can significantly enhance students' ability to find information more effectively and reach their learning goals. Nevertheless, the training of such expert models in education remains largely unexplored. This study explores this research gap by developing a framework to transform semi-structured educational web data into structured datasets and perform instruction tuning on foundation models. Additionally, we conduct a comprehensive performance analysis to determine how various training factors affect model performance.

We first employed a systematic and cost-effective approach involving web data extraction, data cleaning, validation, task design based on student surveys, and automated instruction instance generation using LLMs. Human evaluations confirmed the quality, especially the relevance and accuracy of these datasets.

This study then investigates the impact of various training techniques on domain-specific educational large language models (LLMs) performance. Our experiments reveal that further pre-training enhances model performance, especially with domain-specific terminology, although the performance gains decrease as the dataset size increases. Furthermore, multi-task training also improves model relevance, accuracy, and clarity, but less correlated tasks and datasets can present challenges. These challenges include increased complexity and potential degradation in performance due to the model having to switch between diverse tasks. Lastly, this study conducts a comparative analysis of different models and it highlights trade-offs between computational resources and performance.

The findings demonstrate that a structured approach to dataset generation and strategic training can effectively develop domain-specific LLMs in education. This research benefits the development of educational LLMs and provides a foundation for future researchers to build more specialized models in various domains.

Contents

1 Introduction

1.1. Motivation

The field of Natural Language Processing (NLP) has seen remarkable advancements with the introduction of Large Language Models (LLMs) such as GPT-3 [10]. These models have significantly enhanced capabilities in text understanding and generation, making them valuable tools across various domains. Among these powerful LLMs, one of the most well-known commercial products is ChatGPT, which provides services to approximately 200 million users worldwide [11]. LLMs like ChatGPT are primarily designed for general-purpose applications. They excel in a wide range of tasks, from generating coherent text to answering complex questions and even creating code snippets. However, their versatility comes from being trained on a vast and diverse corpus of data, making them adept at handling a variety of subjects but not necessarily optimized for any specific domain. Additionally, ChatGPT may not have access to a company's internal data, such as product sales information or user interaction details. This limits its effectiveness in performing highly specialized tasks that require proprietary or domain-specific knowledge.

Recognizing the potential and limits of LLMs like GPT, several companies have begun to train and deploy their own domain-specific LLMs for internal data analysis and processing. These tailored models are fine-tuned on industry-specific datasets, which enables them to perform specialized tasks with greater accuracy and efficiency than general-purpose LLMs. For instance, financial institutions like JP-Morgan Chase have developed LLMs to analyze market trends and generate investment insights [34]. And healthcare organizations have created models to assist in medical diagnostics and research [38].

In contrast, the field of education, particularly within universities, has seen limited efforts towards developing domain-specific LLMs. Despite the significant potential benefits, such as improving educational content queries and recommendations, few studies or organizations have worked in this area. This gap highlights the attention for focused research and development to create domain-specific LLMs based on educational data. The domain-specific LLMs in education could help students utilize the university's resources more effectively when they are faced with overloading information from educational websites. The Stanford Alpaca project demonstrates a method to enhance the capabilities of LLMs by constructing training datasets through LLMs and performing instruction tuning on foundational models [40]. However, the Alpaca project remains a general-purpose fine-tuning method, so the question of how to generate education-specific datasets and train an expert model in the education field remains unexplored.

In all, this research aims to address this gap by focusing on the development of domain-specific LLMs for the educational sector. By leveraging instruction tuning and automating the creation of high-quality, domain-specific datasets, this work seeks to enhance the domain adaptation capabilities of LLMs in providing accurate, relevant, and contextually appropriate responses for educational purposes. Through this effort, the research aims to contribute to the university's online educational data, ultimately benefiting students and educators alike.

1.2. Problem Statement

To train domain-specific LLMs in education from foundation models, it is crucial for the model to access and learn from educational data. In this research, the educational data are sourced from TU Delft's course selection website and papers repository. Unlike the structured datasets used in the Alpaca project, which were written by human experts, educational webpages typically store and present data in HTML form, which is semi-structured. LLMs cannot directly parse and learn from these websites. Therefore, this research aims to explore how to extract and transform necessary information from educational web pages into structured instruction-tuning datasets.

Moreover, to perform instruction tuning on the model with educational data, developers must pre-define tasks related to the educational field that the model will be fine-tuned on in the later stages. It is essential to research what kind of tasks are needed and beneficial for users in the educational field. The process of transforming web data into high-quality, task-specific datasets also remains unexplored.

Lastly, assuming there is a structured instruction-tuning dataset in education, this research will focus on how training factors affect model performance. An evaluation process and empirical analysis are necessary to understand the relationship between these factors and the model's effectiveness.

Thus, this research addresses two main questions:

RQ1: How can we transform semi-structured educational website data into an instruction-tuning task dataset that the model can learn from?

Firstly, this research question involves exploring various techniques for web scraping and the use of automated tools to handle the structures of educational websites and convert the semi-structured HTML data into well-organized, structured datasets.

Moreover, it is also necessary to focus on identifying the specific educational tasks that are most beneficial for users, creating templates for these tasks, and generating instruction-tuning datasets. Ensuring the generated datasets are fact-based and meet the instructional needs of students is crucial.

RQ2: What are the impacts of different training techniques on the performance of domain-specific educational LLMs, and how do these techniques affect the models?

This question examines how various training methodologies influence the performance of educational LLMs. It includes investigating the effect of further-pretraining, comparing the performance of models trained on single versus multiple tasks, and evaluating different training strategies. The research aims to provide insights into optimizing the training process to enhance the effectiveness and accuracy of educational LLMs.

1.3. Challenges

In developing domain-specific language models for education, several key challenges must be addressed to ensure effective and efficient model performance. These challenges span data extraction, task design, privacy concerns, and resource optimization. Below are the main challenges identified in this research:

Complexity of Data: Extracting key information from educational websites poses a significant challenge due to the semi-structured nature of online data, often presented in HTML format. Developing a pipeline to extract structured datasets from this semi-structured data without losing any critical information is essential. This process requires sophisticated methods to accurately parse, clean, and transform the data into a format suitable for instruction tuning.

Desigining Appropriate Instructional Tasks: Identifying suitable instructional tasks for the later model learning process is both rigorous and difficult. It is necessary to conduct comprehensive surveys and a thorough review of related literature to ensure that the selected tasks are appropriate and effective for instruction tuning. This step is crucial to tailor the LLM's capabilities to meet the specific needs of the educational domain.

Trade-off between Privacy Measures and Performance: Much of the online educational data at TU Delft is subject to strict privacy measures due to university regulations. Utilizing this data in commercial online LLM APIs could potentially lead to privacy breaches. Therefore, this research must balance security concerns with task generation performance by exploring various large language models to ensure both privacy and efficiency are maintained.

Fine-tuning Under Limited Computing Resources: Fine-tuning large language models for the educational domain can consume substantial computing resources, such as GPU and memory. This research must address how to effectively fine-tune these models under limited computing resources. Ensuring that the models are efficiently fine-tuned to handle diverse educational queries while optimizing resource usage is a critical challenge.

1.4. Contributions

This research provides significant benefits to both the research and industry fields focused on training educational domain-specific LLMs. Firstly, it demonstrates an effective method for extracting information from both static and dynamic educational websites, ensuring the quality and accuracy of the extracted data. This work presents a practical approach to data extraction, and it addresses the challenge of handling semi-structured online data, transforming it into structured datasets suitable for instruction tuning.

Additionally, this research involves conducting surveys among university students to understand their specific needs and preferences regarding educational expert LLMs that possess key information from various educational websites. The insights gained from these surveys inform the design of instruction tuning tasks, ensuring they are relevant and beneficial to the end users. By aligning the instruction tuning tasks with student needs, the research enhances the practical applicability of the trained models.

The study also leverages open-source LLMs like Mixtral to generate instruction-tuning datasets that are cost-effective and low-cost in humans. By automating the creation of these datasets, Mixtral reduces the need for extensive human annotation, which is both time-consuming and costly. This process involved designing task-specific templates based on student surveys to ensure the tasks meet student need, and then using Mixtral to populate these templates with accurate and relevant information from educational web data. Compared to previous instruction dataset generation approaches, this method maintains data quality while minimizing associated costs compared to the previous. It brings feasibility for institutions to develop domain-specific LLMs on a budget without compromising accuracy. This method can be adopted by other researchers and practitioners to create large-scale, high-quality datasets tailored to their specific needs.

Furthermore, this research examines various training techniques and factors that influence model performance. It explores the combination of further pre-training and instruction tuning under limited computing resources, demonstrating how these methods can affect model performance. The study also investigates the impact of multi-task training on model responses, providing valuable insights into optimizing training processes for educational LLMs when there are various input data sources. These insights offer readers practical guidance on how to effectively design a strategy consisting of pre-training and multi-task instruction tuning to improve model outcomes, especially when operating under resource constraints.

Overall, this research contributes to the development and evaluation of domain-specific LLMs in education. By proposing a comprehensive framework from educational web data extraction, and task generation to fine-tuning, the research could ultimately benefit students, educators, and researchers in the educational field. It presents a clear, actionable framework that can be adapted and applied to various educational contexts, offering a structured approach to developing effective, domain-specific language models that address real-world educational needs.

Here is the current structure for this thesis research in later chapters.

1. Background: This section will delve into the current advancements of NLP and LLMs, with a special focus on instruction tuning. It also introduces the data sources of this research.
2. Related Work: This section will focus on the analysis of current work related to dataset generation and instruction tuning projects. It also covers the recent study on how LLMs could work in the field of education.
3. Methodology (2 Stages): This section details the two stages of the development of the data

transformation pipeline. It describes the process of automatically extracting educational data from the university's public website and converting educational resources into instruction-tuning datasets. The stages include data extraction, data preprocessing, and dataset preparation for training LLMs. The final stage involves processes and planned experiments of instruction tuning to optimize and analyze the performance of the models for specific educational tasks.

4. Experiment and Result: In this segment, the experiments conducted using the developed pipeline with educational data will be thoroughly documented. The evaluation will adopt a comprehensive framework involving human expert evaluations, and machine metrics to assess the performance of instruction-tuned models. This multi-faceted approach will provide a deeper understanding of the model's effectiveness in practical applications, going beyond traditional metrics like BLEU to include qualitative insights and user evaluation.
5. Conclusion: This section will summarize the key findings of the research, addressing how the methodologies applied in this study successfully transformed semi-structured educational data into effective instruction-tuning datasets. It will also discuss the impacts of different training techniques on the performance of domain-specific educational LLMs. And it offers insights into future research directions and potential improvements in model training and application.

2 Background

2.1. Large Language Models

Before we introduce the technique of instruction tuning, it is necessary to first study several advanced LLMs since LLMs are fast-evolving, and these models will serve as the foundation models for training in the methodology section. Understanding the architecture, training data, and innovations of these LLMs is crucial to effectively applying instruction tuning techniques. In this section, we will explore three prominent LLMs: Gemma, Llama3, and Mixtral 8x Series, examining their key features to provide a comprehensive foundation for their subsequent training and performance analysis.

2.1.1. Gemma

Developed by DeepMind and the Google AI Research team, Gemma has become one of the latest open-source large language models with multiple versions [41]. The design of Gemma is based on Google's early-stage Gemini model and it enhances the model's language generation, summarization, understanding, and reasoning abilities by introducing several new techniques based on transformer and attention mechanism.

Gemma is pre-trained in various data sources such as documents online, newspapers, and open-source coding projects. The total size of training data is around 6T tokens of text. Google currently releases two sizes of pre-trained model: 7 billion and 2 billion parameter versions for different computational resources so it not only supports running on GPU/TPU but also CPU solely devices. The team also releases the pretrain and fine-tuned checkpoints which allow future researchers to continue tuning the model.

The novelty of Gemma is the usage of Multi-Query Attention (MQA) and it helps to accelerate the inference time of both sizes of models compared to the original multi-head attention (MHA) mechanism applied in the previous Gemini model [37]. The classic MHA transformer architecture uses the key (K) and value (V) tensors that need to be loaded and reloaded with the model inference settings increased. This design can consume computing memory a lot and slow down the running time. MQA adapts this problem by sharing the single set of key(K) and value(V) among multiple heads[2, 41].

Moreover, Gemma is one of the first open-sourced models that applies RoPE (Rotary Positional Embeddings) [39]. During the pre-trained stages of LLMs where sentences are fed into the model, the model needs to have the embedding that represents each word for the input sentence. In particular, the sequence or order of the words that appear in a sentence should be stored so that positional embeddings are invented in which each word is transformed into a vector in the same size of dimension. However, classic positional embedding can reach a size limitation of input sentence sequence that undermines the model to memorize long paragraphs or determine the word positions beyond this limit[44]. In contrast, the RoPE mechanism lets the model understand and memorize a long sequence of input by rotating the sequence position in the vector space: it calculates an angle for the word's position in a sentence so the word rotates when the word has further positions.

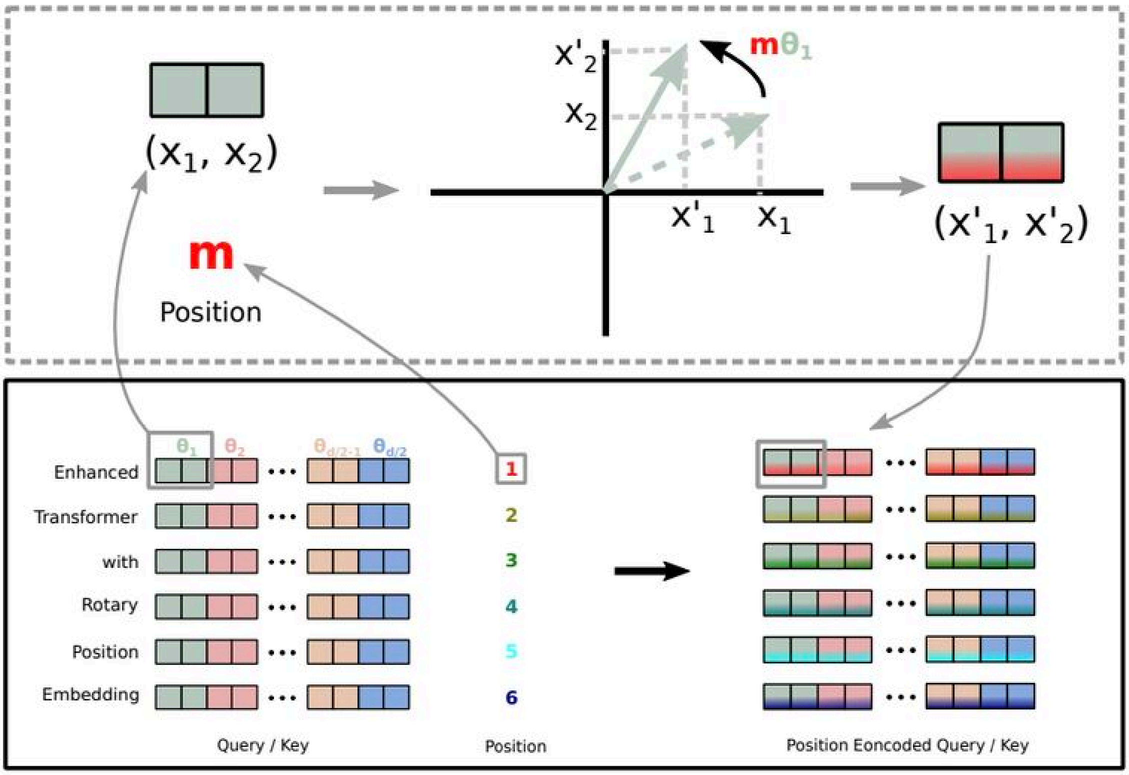


Figure 1: Implementation of Rotary Position Embedding(RoPE).

Figure 2.1: Illustration of the RoPE mechanism [39]

Due to several new techniques involved in Gemma, it can memorize large vocabulary and embedding weights and it performs well on multiple benchmarks such as HellaSwag, GSM8k, AGIEval compared to other LLMs [41]. Meanwhile, the fine-tuning and inferencing computation requirement is relatively low and the inferencing time is accelerated compared to Gemini.

2.1.2. Llama3

Meta Llama 3 is the latest open-source Large Language Model during the research of this thesis [18]. Building on the successes of its predecessors Llama2[42, 26], Llama 3 incorporates numerous innovations to enhance performance and usability. To accommodate various computational resources, Meta has released Llama 3 models in different sizes, including versions with 8 billion and 70 billion parameters, making it accessible for use on GPUs, TPUs, and even CPUs.

Llama 3 is trained on an extensive dataset comprising over 15 trillion tokens, sourced from a diverse array of publicly available texts, including documents, news articles, and open-source coding projects. This training dataset is seven times larger than that used for Llama 2, enabling the model to capture a broader and richer understanding of language. In addition, data-filtering techiqnues are involved in getting better pre-train datasets.

Built upon Llama2, the newer Llama is still implemented based on the previous version's decoder-only transformer architecture[42]. Moreover, LLama3 extends the size of the tokenizer's vocabulary to 128k. This reduces the need to break down common words or phrases into multiple subword tokens, which in turn minimizes tokenization loss and enhances the model's ability to understand and generate text more accurately [26]. A more comprehensive vocabulary also reduces the average number of tokens per sentence. Fewer tokens per sentence mean that the model can process text more efficiently during both training and inference phases, potentially reducing computational costs and speeding up processing times.

Another key innovation in Llama 3 is the adoption of Grouped Query Attention (GQA), which balances computational efficiency and model performance [54]. Traditional MHA, as used in previous models consumes substantial memory and slows down inference[9]. While MQA implemented in Gemma can improve inference efficiency, several new studies show MQA is still potentially hard to capture complex patterns. MQA computes single attention among the whole input sequence only once, so that it may not capture all important details when the input sequence is long and its topics are discrete. As result, the Meta team implements GQA which is an intermediate approach between MHA and MQA, it splits queries into groups, and in each one a single pair of key and value head is stored. Then MQA computes attention within each group so the mechanism improves machine understanding while not lose important information from a general view.

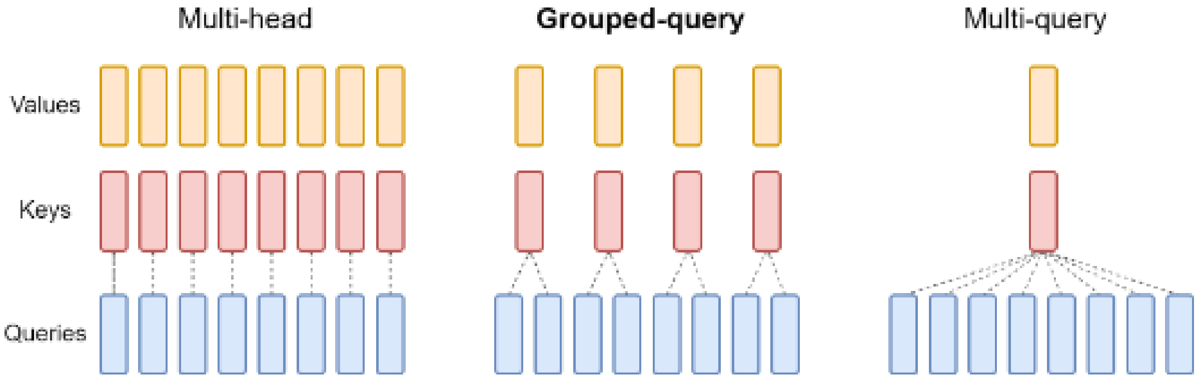


Figure 2.2: Architecture of Several Attention Mechanisms [36]

Llama 3 excels across multiple benchmarks, demonstrating state-of-the-art performance on a wide range of tasks, including reasoning, code generation, and instruction following. It significantly outperforms previous models. This remarkable performance is attributed to improvements in pretraining and post-training processes, which include advanced techniques like supervised fine-tuning (SFT), rejection sampling, proximal policy optimization (PPO), and direct preference optimization (DPO).

2.1.3. Mixtral 8x Series

In 2024, Mixtral 8x7B stands out as a pioneering open-source large language model with its sparse mixture of 8 expert models (SMoE) [19]. Mixtral is pre-trained on an extensive dataset containing multilingual text from various sources, including online documents, newspapers, and open-source coding projects, with a total size of 32,000 tokens. This comprehensive pre-training allows Mixtral to excel in understanding and generating text across multiple languages. The model is available in an 8x7B configuration, meaning it comprises 8 feedforward blocks (experts) per layer.

In particular, a key novelty in Mixtral is its Sparse Mixture of Experts architecture[32]. Unlike traditional multi-head attention mechanisms, Mixtral employs a router network at each layer to assign expert subnetworks to process each token. Every two out of eight experts are designed to handle a certain aspect of the input sequence data and then the router acts like a supervisor to determine how each pair of experts can contribute to the inference process [19]. This dynamic selection allows the model to utilize 13 billion active parameters during inference while having access to 47 billion parameters in total. This design significantly reduces computational load and memory usage, leading to faster inference times and improved model efficiency. Moreover, since each pair of experts is arranged to learn different parts of the input, this leads to the model having better generalization ability while remembering the important details of the input.

In addition to its architectural advancements, Mixtral demonstrates superior performance across a wide range of benchmarks, including mathematics, code generation, and multilingual tasks. It consistently outperforms Llama 2 70B and matches or exceeds the performance of GPT-3.5 on most metrics. Particularly in code generation and mathematical reasoning, Mixtral shows remarkable improvement while using five times fewer active parameters during inference.

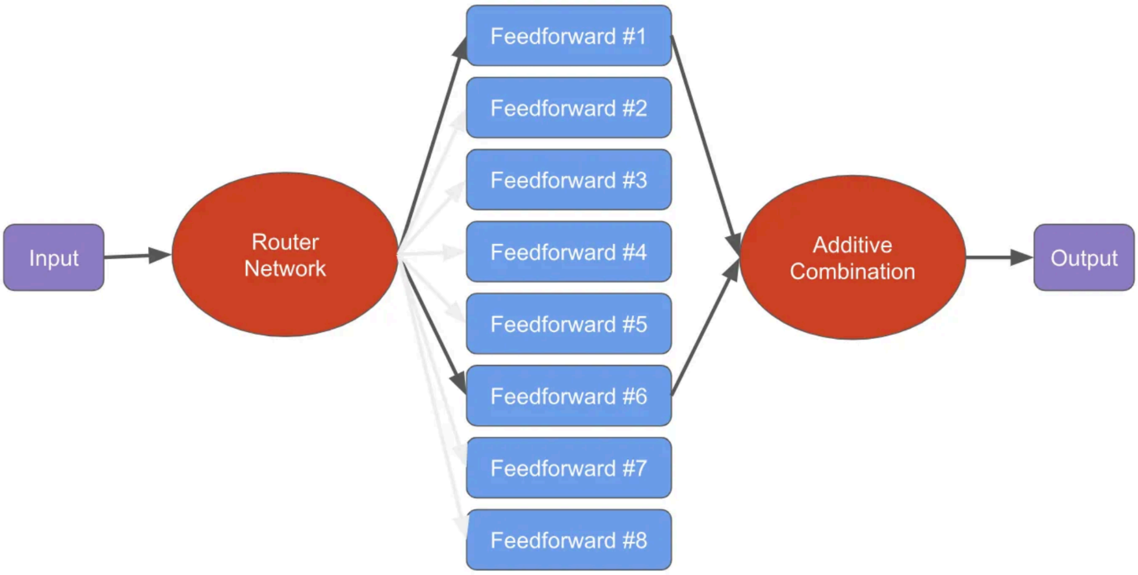


Figure 2.3: Illustration of the Spare Mixture Mechanism [21]

2.2. Rise of Instruction Tuning

2.2.1. Zero-shot Learning

Currently, it cannot be denied that the hallucination of LLMs is a tough problem, often leading to the generation of inaccurate or misleading information [53]. The primary cause of hallucinations is due to the inherent characteristics of LLMs [17]. When these models are inputed with queries or information outside their prior training data or domain knowledge, they can occasionally generate responses that are purely random and not grounded in factual knowledge. Addressing this challenge is crucial for improving the reliability and domain adaptation of LLMs in practical applications. The work presented by Google Research [47] presents a study on improving the domain-adaptation and zero-shot learning abilities of language models through a method called instruction tuning. Zero-shot learning abilities for Large Language Models (LLMs) refer to the capacity to understand and perform tasks they have never encountered during training, based solely on their pre-existing knowledge and logical understanding. This involves fine-tuning a large pre-trained language model (137B parameters) on a diverse collection of over 60 NLP tasks, each described via natural language instructions.

The research team's instruction tuning dataset involves converting existing text datasets into a format where tasks are described through natural language instructions given a specific context. This format includes creating multiple unique templates for each task, which guide the language model in understanding and performing the task as described in human language. These templates is not only based on the original task but also introduce variations. Furthermore, the instruction-tuned model, named Finetuned Language Net (FLAN), is evaluated on various unseen task types. The results show that FLAN significantly outperforms its unmodified models and even outperforms the zero-shot capabilities of the 175B parameter GPT-3 model on 20 out of 25 NLP tasks.

2.2.2. Pre-trained Finetuning

Pretrained fine-tuning has become an essential strategy for improving the performance of LLMs. In the early stage, the effective pre-training method on language models originated from BERT (Bidirectional Encoder Representations from Transformers) [13]. The pre-training technique of BERT takes advantage of masked language models as some tokens from the training data are masked and it allows the model to predict the content solely based on the context of that input data. More importantly, the prediction-masked language ability is independent of the order of the corpus. This method does not follow the classic unidirectional language model training as each token in the model is only computed with its previous token: the unidirectional model might be good for finding a sub-optimal solution within several sentences but it may not be guaranteed to reach the global optimal when the input data is relative long and complex. Thus BERT achieves state-of-the-art results on benchmarks such as the GLUE and SQuAD datasets compared to other contemporary ”left to right” language models like GPT-2. It shows the power of combining broad language understanding with focused task-specific training.

Building on this foundation, Liu et al. [24] developed RoBERTa, an optimized version of BERT. They enhanced the pretraining phase by using more data, longer training periods, and larger batch sizes. This improved pretraining, followed by fine-tuning, resulted in even better performance across multiple NLP expert asks. It helps to validate the effectiveness of the pre-trained fine-tuning method.

Another significant contribution is the T5 model by Raffel et al. [30], which proposed a unified framework for NLP by converting all tasks into a text-to-text format that includes many domain-specific knowledge tasks. This approach extends large language model T5 to a wide range of tasks without modify the inner structure of the model itself. It also simplifes the developing process of domain-specific language models and helps to achieve impressive results across various benchmarks. The extensive pretraining on a massive dataset, followed by task-specific fine-tuning, highlights the flexibility and strength of this method.

In all, pre-training is a non-trivial step of tuning a large language model as it can better understand the context and wording of domain-specific input as proved among several research works. It does not require the model to be modified a lot as long as it is built on masked language model structures.

2.2.3. Prompt Engineering

Prompt engineering has emerged as a critical technique for optimizing the performance of LLMs [7]. Prompt engineering usually happens during the conversation with the LLMs and the models are pre-trained and probably find-tuning for specific tasks at that stage. This approach involves designing and refining the prompts given to these models to produce more accurate and relevant responses. Prompt engineering is particularly important because the effectiveness of LLMs can vary significantly based on how the input queries are typed. Prompt engineering allows users to guide LLMs to perform better on specific tasks without modifying the inner model architecture or changing the parameters of the training model. In particular, those users are not required to have extensive knowledge of deep learning and model implementations. The introduction of this idea helps users yield a more effective large language model that aligns with human needs with low costs of developing and computing.

The process of prompting first requires the user to form clear prompts as input to the model. The prompts could be rules or hints that are highly aligned with specific human task requirements. For instance, given a piece of news article, if a user wants to generate a summary using an LLM, instead of a vague question like ”What is this article talking about?”, they might use a more directive prompt such as ”Summarize the key points of the following news text in three sentences.” This refined prompt provides clear instructions and a specific format and guides the model to produce results based on users’ demands. The idea of prompting stimulates the popularity of large language models such as ChatGPT as it could provide personalized results or even correct its mistakes if users provide hints or instructions during several conversations.

By experimenting with different formulations and structures, users can discover the most effective ways to produce accurate and contextually appropriate responses from the model. A notable example of prompt engineering’s impact is demonstrated in the GPT-3 model by Brown et al. [5]. The authors showed that by carefully designing prompts, GPT-3 could perform various tasks such as translation, question answering, and summarization with better accuracy. This approach leverages the model’s pre-trained knowledge and aligns it with the specific requirements of the task, thereby enhancing its zero-shot and few-shot learning capabilities.

Prompt engineering also plays a crucial role in reducing the occurrence of hallucinations in LLMs, which are instances where the model generates inaccurate or misleading information. According to a study by Yu et al. [50], the use of well-structured prompts significantly reduces the likelihood of hallucinations by providing clearer context and more precise instructions to the model. This method ensures that the model’s outputs are more grounded in factual knowledge, improving their reliability and trustworthiness.

2.2.4. Instruction Tuning on Foundation Models

While prompt engineering is effective for guiding LLMs to perform better on specific tasks, it has limitations when dealing with domain adaptation in larger data sizes or more complex tasks. In such cases, instruction tuning offers a more systematic and scalable approach. Instruction tuning stands as one of the most effective fine-tuning methods for reaching domain adaptation and enhancing the performance of pre-trained foundation models on specialized downstream tasks.

This section delves into the concept and methodology of instruction tuning for LLMs. Instruction tuning stands as one of the most effective fine-tuning methods for enhancing the performance of pre-trained foundation models to align human tasks. Foundation models in the context of this research, are a category of LLMs that are initially trained on extensive and diverse datasets to develop a comprehensive understanding of language patterns, contexts, and knowledge. These models, which serve as a solid base, can be further tailored through additional training or fine-tuning focused on task-specific datasets. Within the scope of this research, the term 'foundation models' specifically refers to LLMs such as Gemma, Mixtral, and Llama2/3.

The utility of instruction tuning is trying to tackle the limitations of pre-trained LLMs: they are not inherently optimized for tasks such as engaging in conversations or following specific instructions [51]. For instance, the foundation model Llama2 is primarily designed to excel in sentence completion tasks. Typically, LLMs do not directly respond to prompts but rather extend them by doing sentence completion [40, 42]. This characteristic can lead to outputs that, while linguistically correct, may not be contextually or functionally appropriate for specific interactions or user needs. Instruction tuning specifically addresses this challenge by refining the model's ability to generate text that is not only relevant but also directly useful in response to the instructions provided. By incorporating targeted instruction-based examples during the fine-tuning phase, instruction tuning effectively molds the model's outputs to be more aligned with the desired outcomes of specific tasks. This process significantly enhances the practicality of LLMs, transforming them from solely text completers into intelligent agents capable of executing tasks that require a higher level of understanding and responsiveness to user commands. This makes instruction tuning an invaluable strategy in leveraging the robust capabilities of foundation models to meet specific functional requirements across various applications.

At present most researchers store the instruction tuning dataset in the format of JSON file [40]. This is because JSON files offer a lightweight, easily readable, and widely supported format for structured data. JSON files are particularly well-suited for representing hierarchical data structures, which are common in instruction-tuning datasets. They make the training procedure easier to organize and access the multiple components of each training sample. In the instruction tuning dataset, each entry of the JSON file consists of three critical components:

1. Instruction: This is a natural language text input that clearly specifies a task to be performed. For instance, an instruction might say, “Summarize the following article,” directing the model to condense a longer text into a brief summary.
2. Additional Information: This optional component provides context relevant to the task at hand, enhancing the model's understanding and performance. For example, in a task involving sentiment analysis, the input might include a customer review, with the instruction to identify and categorize the expressed sentiment.
3. Desired Output: This is the target output or response for the given prompt, defined according to the instructions and contextual information provided. It serves as the ground truth against which the model's predictions are evaluated and optimized. For instance, if the task is to generate a list of synonyms, the desired output would include a precise list of synonyms corresponding to a given word, serving as a benchmark for the model's accuracy.

By integrating these elements into the instruction tuning process, the training dataset not only guides the model in what to produce but also how to approach and understand the task contextually. This structure enhances the model's ability to generate relevant, accurate responses across a variety of possible tasks, significantly improving its utility for specific applications.

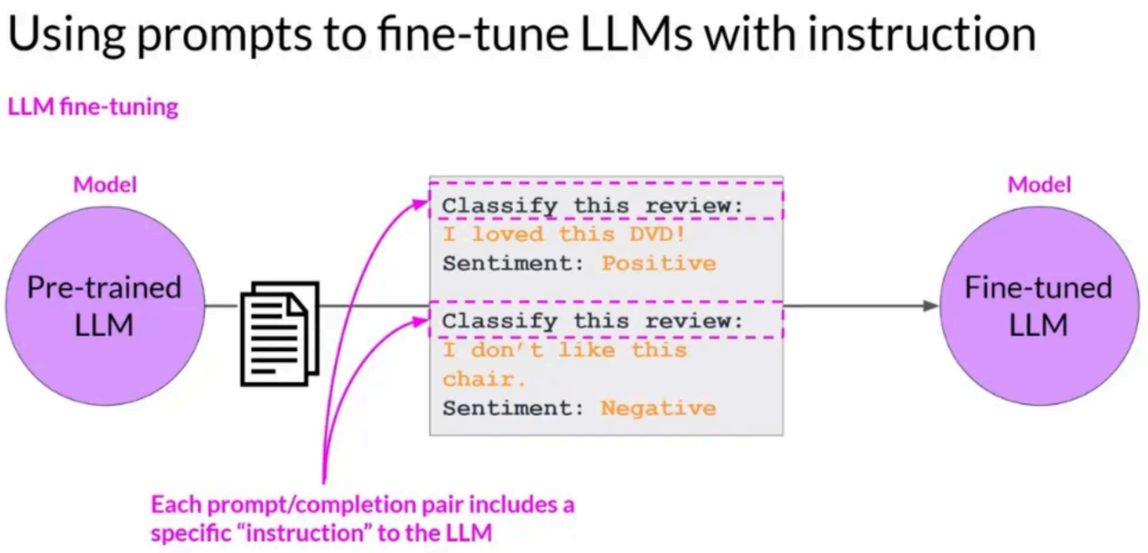


Figure 2.4: Illustration of the Instruction Tuning Technique [35]

2.3. TU Delft Online Education Resource

2.3.1. Study Guide

The course selection website Study Guide from TU Delft is an essential educational resource for students enrolled particularly within the Bachelor and Master's programs. This platform provides a comprehensive overview of the course offerings available, with detailed information included for each course.

The interface is organized in a user-friendly manner with a clear hierarchy that categorizes courses under different specialization tracks and relevant periods. Each course listing includes key details such as the course code, the responsible instructor(s), contact details, and the number of contact hours per week. This structure is beneficial for students planning their academic year, as it allows them to align their educational goals with the courses that best fit their career aspirations.

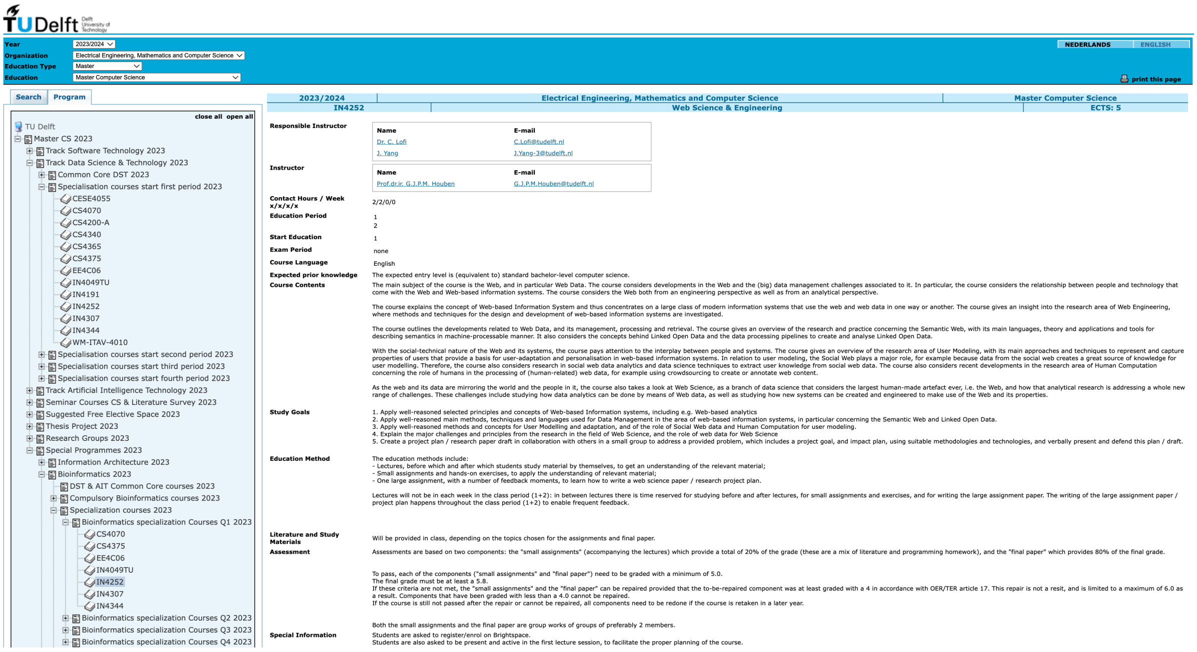
Moreover, the webpage includes essential details about each course, such as the expected prior knowledge, course contents, study goals, and education methods. This level of detail give the student a chance to understand the learning outcomes and teaching approaches before enrolling, which is crucial for managing expectations and achieving academic success.

2.3.2. Thesis Repository

Here illustrates the digital interface of TU Delft's education repository, a critical resource for students, researchers, and educators. This platform facilitates access to a wide array of academic materials, particularly focusing on student theses from various disciplines within the university.

TU Delft educational repository is an integral digital resource for managing and accessing a broad spectrum of student theses across various disciplines. This platform provides detailed views of individual projects. Moreover, each entry typically includes critical information such as the thesis title, author, contributors, degree-granting institution, and program of study, enriching the repository's utility for academic and research purposes. This interface allows users to filter results by collection, document type, subject, author, and date, enhancing the ease of locating relevant documents.

The repository is structured to facilitate efficient data retrieval and management, supporting a range of document types, including master theses, bachelor theses, and student reports. It categorizes these documents under multiple subjects, allowing for refined searches based on specific academic interests or fields of study. This structure supports the ongoing learning and research processes by providing a centralized platform for the accumulation and reuse of scholarly work. Furthermore, the inclusion of abstracts and keywords helps in quickly understanding the scope and focus of each thesis, enhancing the research experience within the academic community.


TU Delft
NEDERLANDS
ENGLISH
print this page
Search
Program
close all open all
TU Delft
Master CS 2023
Track Software Technology 2023
Track Data Science & Technology 2023
Common Core DST 2023
Specialisation courses start first period 2023
CESE4055
CS4070
CS4200-A
CS4340
CS4365
CS4375
EE4C06
IN4049TU
IN4191
IN4252
IN4307
IN4344
WM-ITAV-4010
Specialisation courses start second period 2023
Specialisation courses start third period 2023
Specialisation courses start fourth period 2023
Track Artificial Intelligence Technology 2023
Seminar Courses CS & Literature Survey 2023
Suggested Free Elective Space 2023
Thesis Project 2023
Research Groups 2023
Special Programmes 2023
Information Architecture 2023
Bioinformatics 2023
DST & AIT Common Core courses 2023
Compulsory Bioinformatics courses 2023
Specialization courses 2023
Bioinformatics specialization Courses Q1 2023
CS4070
CS4375
EE4C06
IN4049TU
IN4252
IN4307
IN4344
Bioinformatics specialization Courses Q2 2023
Bioinformatics specialization Courses Q3 2023
Bioinformatics specialization Courses Q4 2023

Year	2023/2024
Organization	Electrical Engineering, Mathematics and Computer Science
Education Type	Master
Education	Master Computer Science

2023/2024	Electrical Engineering, Mathematics and Computer Science	Master Computer Science
IN4252	Web Science & Engineering	ECTS: 5

Responsible Instructor

Name	E-mail
Dr. C. Lofi	C.Lofi@tudelft.nl
J. Yang	J.Yang-3@tudelft.nl

Instructor

Name	E-mail
Prof.dr.ir. G.J.P.M. Houben	G.J.P.M.Houben@tudelft.nl

Contact Hours / Week x/x/x/x 2/2/0/0

Education Period 1
2

Start Education 1

Exam Period none

Course Language English

Expected prior knowledge The expected entry level is (equivalent to) standard bachelor-level computer science.

Course Contents The main subject of the course is the Web, and in particular Web Data. The course considers developments in the Web and the (big) data management challenges associated to it. In particular, the course considers the relationship between people and technology that come with the Web and Web-based information systems. The course considers the Web both from an engineering perspective as well as from an analytical perspective.

The course explains the concept of Web-based Information System and thus concentrates on a large class of modern information systems that use the web and web data in one way or another. The course gives an insight into the research area of Web Engineering, where methods and techniques for the design and development of web-based information systems are investigated.

The course outlines the developments related to Web Data, and its management, processing and retrieval. The course gives an overview of the research and practice concerning the Semantic Web, with its main languages, theory and applications and tools for describing semantics in machine-processable manner. It also considers the concepts behind Linked Open Data and the data processing pipelines to create and analyse Linked Open Data.

With the social-technical nature of the Web and its systems, the course pays attention to the interplay between people and systems. The course gives an overview of the research area of User Modeling, with its main approaches and techniques to represent and capture properties of users that provide a basis for user-adaptation and personalisation in web-based information systems. In relation to user modeling, the Social Web plays a major role, for example because data from the social web creates a great source of knowledge for user modelling. Therefore, the course also considers research in social web data analytics and data science techniques to extract user knowledge from social web data. The course also considers recent developments in the research area of Human Computation concerning the role of humans in the processing of (human-related) web data, for example using crowdsourcing to create or annotate web content.

As the web and its data are mirroring the world and the people in it, the course also takes a look at Web Science, as a branch of data science that considers the largest human-made artefact ever, i.e. the Web, and how that analytical research is addressing a whole new range of challenges. These challenges include studying how data analytics can be done by means of Web data, as well as studying how new systems can be created and engineered to make use of the Web and its properties.

Study Goals
1. Apply well-reasoned selected principles and concepts of Web-based Information systems, including e.g. Web-based analytics
2. Apply well-reasoned main methods, techniques and languages used for Data Management in the area of web-based information systems, in particular concerning the Semantic Web and Linked Open Data.
3. Apply well-reasoned methods and concepts for User Modelling and adaptation, and of the role of Social Web data and Human Computation for user modeling.
4. Explain the major challenges and principles from the research in the field of Web Science, and the role of web data for Web Science
5. Create a project plan / research paper draft in collaboration with others in a small group to address a provided problem, which includes a project goal, and impact plan, using suitable methodologies and technologies, and verbally present and defend this plan / draft.

Education Method The education methods include:
- Lectures, before which and after which students study material by themselves, to get an understanding of the relevant material;
- Small assignments and hands-on exercises, to apply the understanding of relevant material;
- One large assignment, with a number of feedback moments, to learn how to write a web science paper / research project plan.

Lectures will not be in each week in the class period (1+2): in between lectures there is time reserved for studying before and after lectures, for small assignments and exercises, and for writing the large assignment paper. The writing of the large assignment paper / project plan happens throughout the class period (1+2) to enable frequent feedback.

Literature and Study Materials Will be provided in class, depending on the topics chosen for the assignments and final paper.

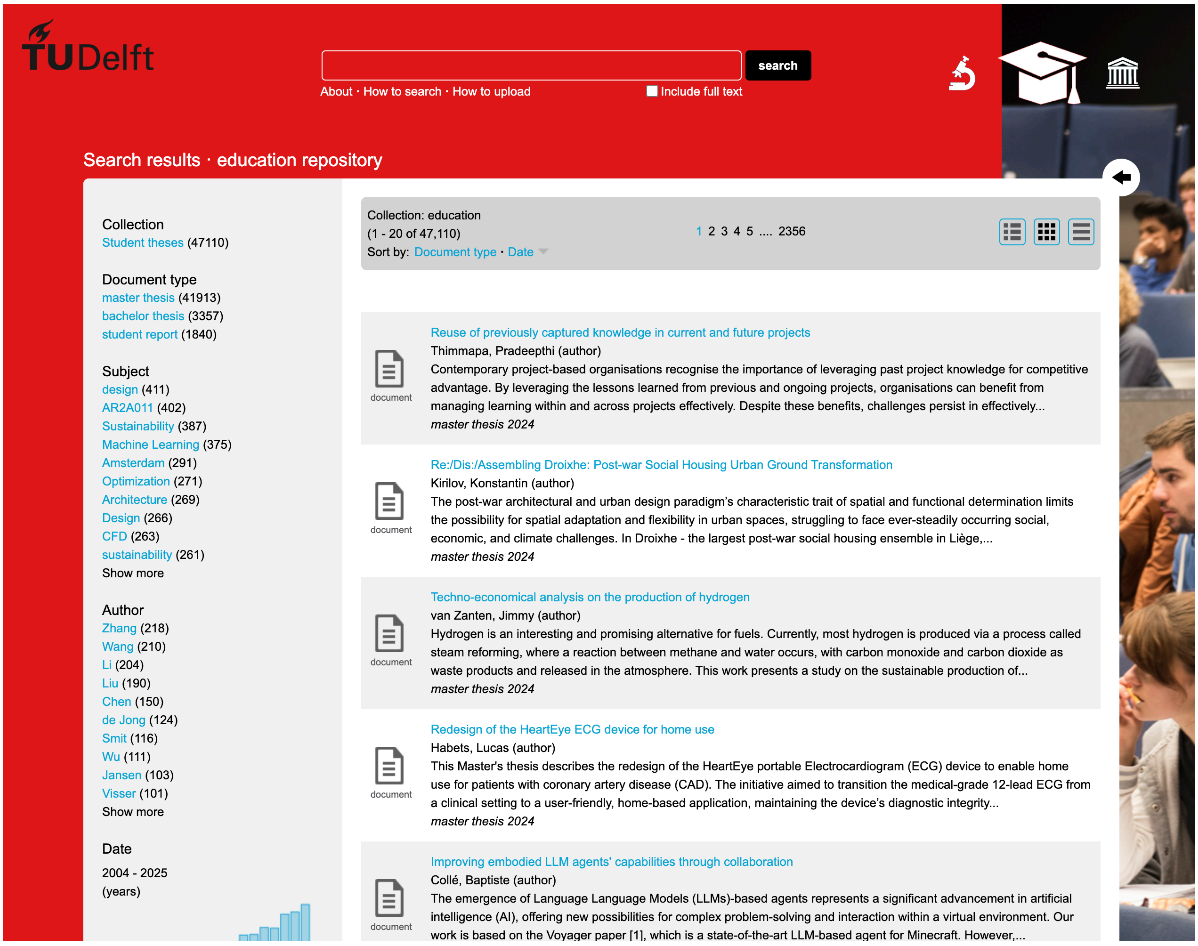
Assessment Assessments are based on two components: the "small assignments" (accompanying the lectures) which provide a total of 20% of the grade (these are a mix of literature and programming homework), and the "final paper" which provides 80% of the final grade.

To pass, each of the components ("small assignments" and "final paper") need to be graded with a minimum of 5.0.
The final grade must be at least a 5.8.
If these criteria are not met, the "small assignments" and the "final paper" can be repaired provided that the to-be-repaired component was at least graded with a 4 in accordance with OER/TER article 17. This repair is not a resit, and is limited to a maximum of 6.0 as a result. Components that have been graded with less than a 4.0 cannot be repaired.
If the course is still not passed after the repair or cannot be repaired, all components need to be redone if the course is retaken in a later year.

Both the small assignments and the final paper are group works of groups of preferably 2 members.

Special Information Students are asked to register/enrol on Brightspace.
Students are also asked to be present and active in the first lecture session, to facilitate the proper planning of the course.

Figure 2.5: Illustration of the Study Guide Website



Search results · education repository

Collection
Student theses (47110)

Document type
master thesis (41913)
bachelor thesis (3357)
student report (1840)

Subject
design (411)
AR2A011 (402)
Sustainability (387)
Machine Learning (375)
Amsterdam (291)
Optimization (271)
Architecture (269)
Design (266)
CFD (263)
sustainability (261)
Show more

Author
Zhang (218)
Wang (210)
Li (204)
Liu (190)
Chen (150)
de Jong (124)
Smit (116)
Wu (111)
Jansen (103)
Visser (101)
Show more

Date
2004 - 2025
(years)

Collection: education
(1 - 20 of 47,110)
Sort by: Document type · Date

1 2 3 4 5 2356

document
Reuse of previously captured knowledge in current and future projects
Thimmapa, Pradeepthi (author)
Contemporary project-based organisations recognise the importance of leveraging past project knowledge for competitive advantage. By leveraging the lessons learned from previous and ongoing projects, organisations can benefit from managing learning within and across projects effectively. Despite these benefits, challenges persist in effectively...
master thesis 2024

document
Re:/Dis:/Assembling Droixhe: Post-war Social Housing Urban Ground Transformation
Kirilov, Konstantin (author)
The post-war architectural and urban design paradigm's characteristic trait of spatial and functional determination limits the possibility for spatial adaptation and flexibility in urban spaces, struggling to face ever-steadily occurring social, economic, and climate challenges. In Droixhe - the largest post-war social housing ensemble in Liège,...
master thesis 2024

document
Techno-economical analysis on the production of hydrogen
van Zanten, Jimmy (author)
Hydrogen is an interesting and promising alternative for fuels. Currently, most hydrogen is produced via a process called steam reforming, where a reaction between methane and water occurs, with carbon monoxide and carbon dioxide as waste products and released in the atmosphere. This work presents a study on the sustainable production of...
master thesis 2024

document
Redesign of the HeartEye ECG device for home use
Habets, Lucas (author)
This Master's thesis describes the redesign of the HeartEye portable Electrocardiogram (ECG) device to enable home use for patients with coronary artery disease (CAD). The initiative aimed to transition the medical-grade 12-lead ECG from a clinical setting to a user-friendly, home-based application, maintaining the device's diagnostic integrity...
master thesis 2024

document
Improving embodied LLM agents' capabilities through collaboration
Collé, Baptiste (author)
The emergence of Language Language Models (LLMs)-based agents represents a significant advancement in artificial intelligence (AI), offering new possibilities for complex problem-solving and interaction within a virtual environment. Our work is based on the Voyager paper [1], which is a state-of-the-art LLM-based agent for Minecraft. However,...

Figure 2.6: Illustration of the Study Guide Website

3 Related Work

3.1. LLMs Studies in Educational Domain

In the reviewed papers, several authors focus on the survey Large Language Models (LLMs) to relate to online educational systems such as virtual teachers, homework helpers, and QA chatbot [45, 28]. In general, they explore the potential of LLMs in creating a next-generation intelligent education system by investigating their capabilities in various educational skills such as mathematics, writing, programming, reasoning, and knowledge-based question answering. For the scope of this master thesis, the scope is about reasoning and knowledge-based question answering.

A study by Cha et al. highlights the significant role AI-based systems, including LLMs, can play in course recommendations for students [6]. It emphasizes the potential for LLMs to help students discover relationships among course concepts and information across different departments and calls for future researchers to further explore this application. However, such study's data source is a structured form. While LLMs have advanced in understanding human language and reasoning, they often fall short in analyzing dynamic web data and making personalized responses, which are critical in educational settings. For instance, LLMs can process and generate language-based responses but struggle with understanding and adapting to the individual learning needs of students from the web information, which can vary widely.

Nonetheless, in some studies, the researchers present a way to introduce updated information by sampling multiple-choice questions [20]: during the beginning of each prompt with a LLM, it demonstrates the model selecting the most consistent answer in order to increase its reasoning and decision-making ability. However, this method results in a static and fully human-annotated process to prompt the model since the researchers need to prepare multiple unique questions for each different purpose of the conversation.

One significant advantage of LLMs is their ability to integrate extensive knowledge bases into their responses, thereby enhancing the quality and relevance of the information they provide. However, a study conducted by Xu et al. points out that LLMs still face challenges in terms of accuracy and the tendency to generate plausible but incorrect answers, known as ”hallucinations,” which can mislead learners [49]. The authors suggest that future research should focus on enhancing the reasoning capabilities of LLMs through methods such as supervised fine-tuning, prompt engineering, and hybrid methods that combine different tuning strategies to refine LLMs' educational applications. Moreover, they emphasize the importance of developing LLMs that can operate across various domains of knowledge effectively, using advanced algorithms that help mitigate the issues of hallucination and improve the precision of LLM outputs in educational settings.

3.2. Factors of Fine-tuning Performance

The relationship between the size of pre-training data and the performance of supervised fine-tuned (SFT) models has been a significant area of study. Several authors have explored how the pre-training data influences the effectiveness of fine-tuned models, particularly in LLMs [15, 25, 14].

In recent studies, researchers have demonstrated that the quantity of pre-training data directly impacts the performance of SFT models. For example, McKinzie et al. highlight the effect of increasing the size of pre-training datasets on model performance [25]. The results indicate that models pre-trained with larger datasets exhibit superior performance during fine-tuning stages compared to those pre-trained with smaller datasets. This improvement is evident in various tasks, including few-shot and zero-shot learning scenarios.

However, McKinzie's team primarily trained on general tasks like LLaVA-Complex and ChatQA. The effect of pre-training on education-specific data has not been thoroughly explored. This gap leaves questions about the models' performance and adaptability in educational domains where specific terminologies and contextual knowledge are crucial. Moreover, there is a lack of studies exploring the trade-offs between computational efficiency and performance, particularly in resource-constrained environments common in educational institutions.

Zhang et al. also investigate the practical application of fine-tuning LLMs for specific tasks rather than general ones [52]. They focus on the writing-assistant scenario with distinct writing tasks. By reformulating the training data into an instruction-following format, they fine-tune the LLaMA model to improve its performance on these writing tasks. Compared to larger models that are not fine-tuned for these tasks, their findings suggest that fine-tuning LLaMA on specific writing instruction data significantly enhances its performance. This paper also explores whether it is necessary to employ LLMs for a single targeted task, highlighting the potential efficiency and resource considerations.

However, Zhang's study does not explore how the number of similar tasks included in the fine-tuning process affects the overall performance of the model. Secondly, the study does not investigate the impact of introducing different types of datasets during the fine-tuning process. The influence of the multi-task training and dataset types on the model's performance remains unexplored. Future research should consider these factors to gain a more comprehensive understanding of how multi-task training can be optimized for different applications.

3.3. Instruction Tuning Dataset Creation

Recently Wang et al. [46] introduced a novel method called Self-Instruct for generating the instruction tuning dataset by pre-trained large language models like GPT-3. Currently, many instruction-tuning datasets are mostly written by human experts and they cost a lot since the data are based on facts [47, 3, 29]. However, the Self-Instruct method uses the large language model's generative abilities to create new instructional data. It starts with a small set of manually expert-written tasks and prompts the model to generate new instructions and corresponding input-output instances. These are then filtered for quality and uniqueness, and the valid tasks are used for further instruction tuning. The process is iterative, leading to a diverse and extensive set of instructional data.

When the instruction dataset is applied to GPT-3, the Self-Instruct method significantly improved its performance, achieving results nearly the same as InstructGPT-001's performance [29], a model trained with extensive human-written data. This approach demonstrates an efficient way to enhance language models' ability to follow instructions, with less reliance on human-labeled data. The paper also contributes a large synthetic dataset for future research in this area.

3.4. State-of-art Instruction Tuning Studies

3.4.1. Domain Specific LLMs

Many researchers have taken advantage of instruction tuning and transformed the general LLMs into an expert-level model that focuses on specific tasks. In the research of the K2, a geological large language model, led by Deng et al. [12] shows a significant focus on the creation of the instruction tuning dataset, GeoSignal. This dataset is crucial for aligning the pre-trained language model with user intentions in the geoscience domain.

The GeoSignal dataset is constructed through a semi-manual pipeline, combining general instruction tuning data from sources like Natural Instruction [27] and AI2 Reasoning Challenge [8] with geoscience expert-generated data. Additionally, the paper demonstrates the creation of GeoTool, a tool training dataset to enable K2 to use geoscience-specific tools. This dataset is crucial for training K2 to interact with geoscience academic search engines and other domain-specific tools.

However, the paper also has several challenges and has areas for future improvement. One significant issue is the resource-intensive nature of constructing the GeoSignal dataset, which requires substantial human expertise. In particular, the reliance on human experts for dataset construction and validation, while ensuring quality, adds to the cost and complexity of the process. Moreover, the size and diversity of the training dataset remain unexplored to the model's performance, indicating a need to study on the continuous expansion and diversification of the dataset.

3.4.2. Aplaca: Stanford's Fine-tuned LLaMA 7B Model

Due to OpenAI's decision not to disclose the specifics of models beyond GPT-3, it is hard for researchers to understand how instruction tuning contributes to enhancing the performance of such LLMs on unfamiliar tasks. However, the Stanford Center for Research on Foundation Models (CRFM) released Alpaca [40], a model fine-tuned from Meta's open-sourced LLaMA 7B model on 52K instruction tuning datasets. In particular, the Alpaca is trained on instruction-following demonstrations generated in the style of self-instruct [46] using text-davinci-003 [4]. Moreover, the lightweight training approach makes it cost-effective, and the full model's code and training dataset are published on GitHub, allowing accessible options for academic research.

Alpaca's project is trying to tackle two main challenges: obtaining a strong pre-trained language model and high-quality instruction-following data. The LLaMA models from Meta address the first challenge [42], while the self-instruct paper's method of using an existing strong language model to generate instruction data tackles the second [46]. The training process is efficient and only consumes RTX3090, making it suitable for academic budgets. In the end, Alpaca has been evaluated through human evaluation with text-davinci-003, the new model showing similar performance compared to GPT3.5 and GPT4 in some tasks[40]. However, the research on Alpacas requires more study on how instruction tuning helps to train a domain-specific model but not just general purpose one. And the scalability of the training dataset can be another focus to work on for future researchers.

3.4.3. Instruction Tuning on LLMs in Limited Resources

Fine-tuning large language models can be resource-intensive and it often requires researchers to have substantial computational power and memory. LoRA (Low-Rank Adaptation) offers a solution by introducing low-rank decomposition matrices into each layer of the Transformer architecture [16]. It significantly reduces the number of trainable parameters and the computational load during the fine-tuning process.

LoRA freezes the pre-trained model weights and only trains the low-rank matrices, which results in up to 10,000 times fewer trainable parameters compared to full fine-tuning. Specifically, LoRA decomposes the weight update matrix ΔW into two smaller matrices A and B, such that $\Delta W = A \times B$. Here, A is a low-rank matrix with dimensions $(d \times r)$ and B is a matrix with dimensions $(r \times k)$, where d is the input dimension, k is the output dimension, and r is the rank (with $r \ll d, k$). This decomposition ensures that the number of trainable parameters is significantly reduced to $(d \times r) + (r \times k)$.

This approach not only reduces the GPU memory requirement by three times but also maintains model performance across various tasks compared to the full parameter fine-tuning model. In practice, the low-rank matrices A and B are initialized to zero, and only these matrices are updated during fine-tuning, leaving the original model weights W unchanged. This makes the fine-tuning process much more efficient.

The key benefits of LoRA include:

- **Efficiency**: Reduces memory and storage usage, enabling fine-tuning on limited resources.
- **Scalability**: Allows efficient task-switching by only swapping low-rank matrices.
- **Performance**: Maintains high model quality without additional inference latency.

LoRA provides a cost-effective and efficient method for fine-tuning LLMs. It is an essential tool for deploying large models in resource-constrained environments without compromising model performance.

4 Stage 1: Instruction Tuning Dataset Generation

4.1. Overview

To continue answering the question **RQ1 (How can we transform semi-structured educational website data into an instruction-tuning task dataset that the model can learn from?)**, the first part of this methodology section converts the educational data into a structured format. Specifically, we focus on two main educational resources: the webpages of the TU Delft Study Guide and the TU Delft Thesis Repository. The Study Guide is relatively static, as course plan data are typically updated on an annual basis. Conversely, the Thesis Repository is updated frequently with new thesis submissions. Therefore, a dynamic and adaptable approach is required to efficiently handle new uploads and ensure the extracted data remains current and comprehensive. To achieve this, we implement Python Beautiful Soup and Selenium to extract data from these webpages based on their specific characteristics.

The second part of this section takes the approach to transforming CSV file data into instruction-tuning formats that LLMs can understand during the later fine-tuning stage. This process involves several key steps, including literature review and demand analysis through surveys for task formulation, template creation, and dataset generation using Mixtral 8x7b. Each of these steps is designed to ensure that the transformed data meets the specific needs of domain-specific educational LLMs.

To begin, a survey were conducted to identify the specific demands and expectations of students when interacting with educational websites in universities. This initial step is crucial in guiding the subsequent stages of task creation and data transformation.

Following this, various tasks were formulated to address the identified needs. Each task is carefully designed with specific content and goals in mind, ensuring that they align with the student's expectations and requirements. The tasks are then illustrated through different templates, which serve as fill-in-the-blank frameworks. These templates allow the LLMs to perform summarization and other relevant tasks by filling in the blanks, thereby generating new instruction instance datasets.

Finally, Mixtral 8x7b is employed to automate the dataset generation process. This involves using the templates to create instruction-tuning datasets, which are then integrated into the pipeline for later training LLMs. The resulting datasets are structured to enhance the chatbot's ability to provide academic guidance, research hints, and course selection advice, thereby improving the overall e-learning experience for students. Evaluations are conducted to ensure the quality of the generated datasets, especially verifying their similarity to the original ground truth data.

4.2. Data Sources Analysis

1. **TU Delft Study Guide:**
    - **Description**: The webpage is structured using HTML tables to organize and display educa-

tional information. Key elements include a header and navigation section that features form elements for filtering options such as academic year, organization, education type, and specific education programs. The main content area houses tables detailing courses, including columns for course codes, titles, and ECTS points. These tables use <tr>, <td>, and <div> elements to structure the data. Dropdown menus are provided for users to select filters, each containing <select> and <option> tags. Additionally, the webpage employs multiple JavaScript files to enable functionality like filtering and form submissions, while CSS files ensure the visual presentation is organized and user-friendly.

- **Update Frequency**: Most of the course information are uploaded annually, with occasional updates throughout the academic year.
- **Data Structure**: Primarily static HTML content, which simplifies the extraction process. However, periodic monitoring is necessary to capture any updates made throughout the academic year (see Table 4.1 for more details on the key elements of the study guide webpage)

Key	**Value Meaning**
Course Name	The name of the course
Course Code	The unique code assigned to the course
Responsible Instructor	The main instructor responsible for the course
Instructor	Other instructors involved in teaching the course
Contact Hours / Week	The number of contact hours per week, typically divided into lecture, lab, and self-study hours
Education Period	The period during which the course is taught
Start Education	The start period for the course
Exam Period	The periods during which exams are scheduled
Course Contents	A description of the topics and materials covered in the course
Study Goals	The objectives and learning outcomes that students are expected to achieve by the end of the course
Education Method	The teaching methods used in the course, such as lectures, lab assignments, and self-study
Literature and Study Materials	The books, articles, and other materials required or recommended for the course
Prerequisites	The prior knowledge or courses required before taking this course
Assessment	The methods used to evaluate student performance, such as exams, assignments, and their respective weightings

Table 4.1: Descriptions of Key Elements in the Study Guide Webpage for Course Information

2. **TU Delft Thesis Repository:**

- **Description**: The webpage is designed to display a repository of theses using a combination of HTML elements. The header includes navigation links and search functionality. The main content area displays a list of theses, each with details such as title, author, abstract snippet, and publication year. The thesis details are presented using structured elements like <div>, <ul>, <li>, and <table> tags. Additionally, specific thesis pages provide comprehensive metadata using <fieldset> and <span> elements, along with links for downloading and viewing the theses. The webpage also utilizes JavaScript for dynamic features and CSS for styling.
- **Update Frequency**: The repository is updated continuously with new theses being added regularly.
- **Data Structure**: Dynamic HTML content that requires a robust extraction process capable of handling frequent updates and ensuring data accuracy.

Key	Value Meaning
Title	The title of the thesis
Author	The name(s) of the author(s) of the thesis
Contributor	Individuals or entities who contributed to the thesis, such as mentors or committee members
Programme	The academic programme under which the thesis was submitted
Abstract	A brief summary of the thesis content
Subject	Keywords or subjects associated with the thesis

Table 4.2: Descriptions of Key Elements in the TU Delft Thesis Repository Webpage

4.3. Data Extraction

4.3.1. Extract Data from TU Delft Thesis Repository

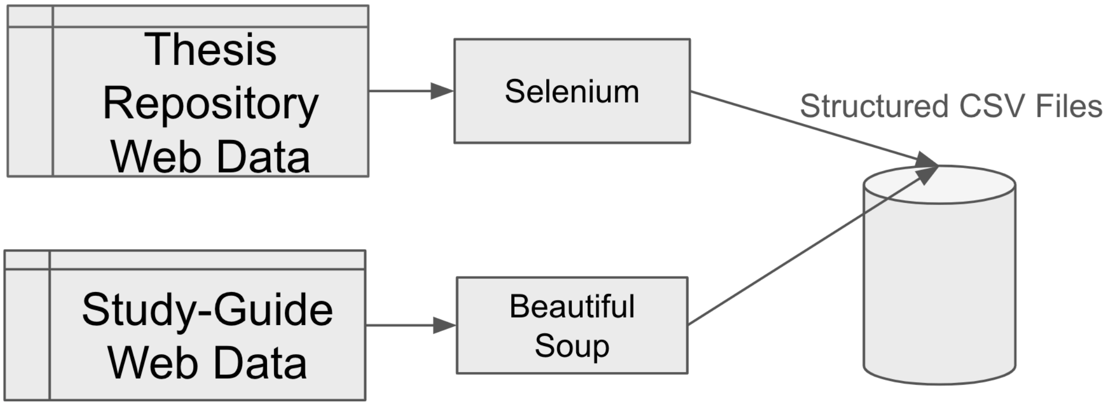


Figure 4.1: Design of the Data Collection

The methodology for extracting data from the TU Delft master's and PhD thesis repository involves an implementation of web scraping techniques using Selenium, which a powerful tool for automating web browsers. Selenium is widely used in the field of web scraping due to its ability to interact with web pages like a human user. It can handle HTML/JavaScript-heavy websites, navigate through multiple pages, refresh the page, and interact with webpages such as mouse clicking. By simulating those user actions, Selenium can effectively extract data from dynamic webpages that traditional scraping tools might struggle with. This makes it particularly suitable for extracting data from complex and frequently updated repositories like the TU Delft thesis repository.

Using Selenium offers an initial approach to addressing the research question of how to create instruction-tuning datasets from semi-structured educational websites. Selenium's automation capabilities significantly reduce the manual effort required to collect data, as it can be programmed to automatically traverse the repository, locate relevant information, and extract it systematically. This automated approach not only enhances efficiency but also ensures that the data is consistently captured in a structured format. By leveraging Selenium, the research can maintain up-to-date and accurate datasets from the thesis repository, supporting the goal of extracting comprehensive educational data with minimal manual intervention. Here is the pipeline from the initialization of Selenium to output a structured CSV file.

1. Initialization of Selenium WebDriver: The process begins with the initialization of the Selenium WebDriver [33], which is responsible for automating the interaction with the web browser. In this case, Chrome WebDriver is used, but it can be substituted with any compatible driver.
2. Navigating Through Pages: The script navigates through the repository pages, starting from a defined start page to an end page. This range is adjustable based on the specific data requirements. Each page is accessed via a constructed URL that includes the page number as a parameter.

3. Data Extraction: For each thesis listed on the index page, the script collects the URL and navigates to the detailed page. Here, it extracts the required information such as title, author, degree information, etc., using XPath selectors to locate the text based on the surrounding labels.
4. Writing to CSV: Extracted data for each thesis is written to a CSV file. This structured format is suitable for further processing and analysis. The CSV header includes columns for each piece of information being collected.
5. Handling Delays and Exceptions: To ensure the reliability of the extraction process and to mitigate the risk of being blocked by the website, the script includes deliberate delays between requests. Additionally, a helper function is used to handle exceptions gracefully, returning 'None' for any missing information.
6. Data Checking: The script includes a data checking step, where it randomly goes back to previously processed pages and compares the webpage data with the CSV data to ensure consistency and accuracy. This step helps in verifying that the data extraction process is functioning correctly and that the stored data matches the source.
7. Finalization: Once all pages within the specified range have been processed, the WebDriver is terminated, concluding the data extraction process.

The output of this process is a CSV file containing a comprehensive dataset of thesis information from the TU Delft repository. This file serves as a foundational component of the instruction tuning dataset, ready to be integrated into the pipeline for training LLMs.

4.3.2. Extract Data from TU Delft Study Guide

For extracting course description data from the TU Delft Study Guide, a methodology is utilized to leverage the Python library requests for HTTP requests and BeautifulSoup from bs4 [31] for HTML parsing. BeautifulSoup is an effective tool for parsing HTML and XML documents earlier than the Selenium web driver. BeautifulSoup extracts various forms of information from web pages straightforwardly. This approach enables the efficient gathering of detailed course information directly from the university's course description website.

This techinque offers an initial approach to addressing the research question of how to create instruction-tuning datasets from semi-structured educational websites, especially static ones. BeautifulSoup simplifies the process of parsing HTML content, allowing for the precise extraction of specific elements from web pages. It is particularly effective for navigating and extracting data from static HTML content, which is prevalent in many educational websites. This ensures that the data is accurately and consistently captured in a structured format, reducing the likelihood of errors and inconsistencies. Here is a pipeline to use BeautifulSoup to extract information from the course selection website.

1. HTTP Requests: The process initiates with sending HTTP requests to specific course detail URLs constructed by appending course IDs to the base URL. Each major's course IDs are obtained from the HTML main web pages of the department. This step is facilitated by the requests library, which fetches the HTML content of the page.
2. HTML Parsing: Upon receiving the HTML content, 'BeautifulSoup' is employed to parse and navigate the DOM tree. It extracts the required information by identifying HTML elements and their classes that correspond to the course details.
3. Data Extraction: Specific details such as the course code, course name, educational period, course contents, study goals, education method, literature and study materials, prerequisites, and assessment methods are meticulously extracted. This is achieved by defining precise selectors and parsing structures that match the layout of the Study Guide's course pages.
4. CSV Writing: The extracted information for each course is then written to a CSV file, providing a structured and easily accessible format for the collected data. This file includes headers for each of the details extracted, ensuring organized storage.
5. Iterative Processing: The script iterates over a list of course IDs, which are either predefined or dynamically extracted from a separate file. This iterative approach allows for the batch processing of multiple course pages in a sequence, maximizing efficiency.

6. Error Handling: Throughout the extraction process, error handling mechanisms are implemented to manage potential issues such as connection errors or missing data. This ensures the robustness of the data collection process and the integrity of the resulting dataset.

The outcome of this methodology is a detailed CSV file that captures a wide range of course information from the TU Delft Study Guide. By organizing this information systematically, the dataset becomes an essential resource for later instruction tuning dataset generation.

4.4. Task Preparation

4.4.1. Survey

To gain a comprehensive understanding of student needs on educational websites, we conducted a survey targeting TU Delft students. It invites 30 participants to provide insights into their perceptions of the educational websites at TU Delft. The respondents, primarily bachelor's and master's students, were familiar with the main educational resources at TU Delft, such as the study guide and paper repository.

The survey began by asking respondents how frequently they used these educational websites and what types of information they sought when browsing the webpages or using the website's search bar function. It then inquired about their experiences with the search functionality of these websites. Finally, the survey asked respondents about their expectations for integrating a large language model (LLM) into these websites and how it could help them find the information they need.

The survey results highlighted several key points. Some respondents noted that the search bar in the study guide relies on exact word matching, which they found inflexible. The top five desired features for an educational LLM, as showed by the survey result in figure 4.2, were:

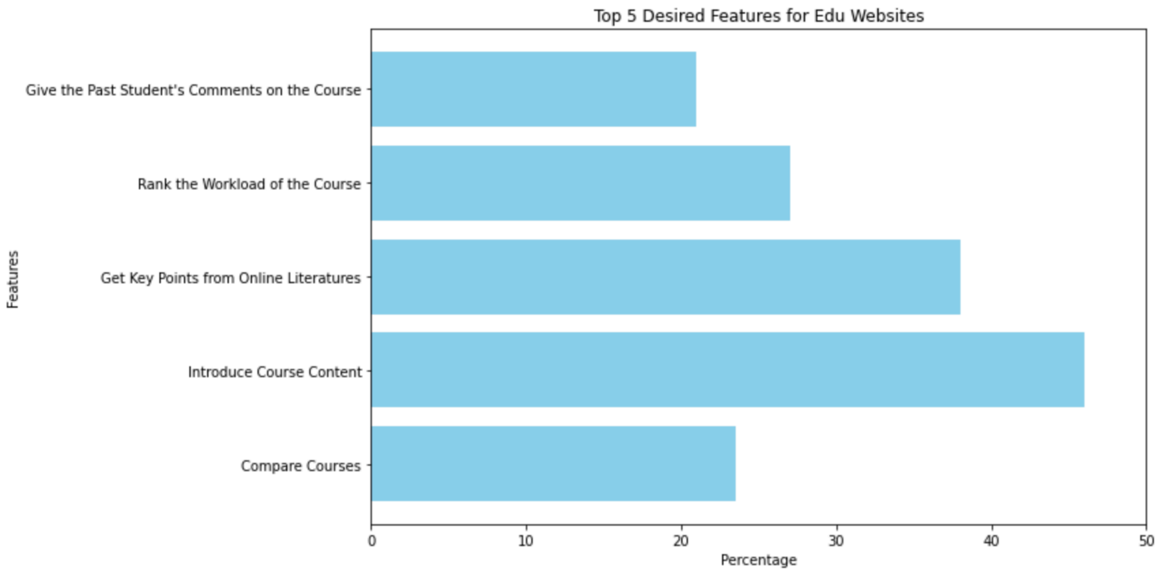


Figure 4.2: Survey of Student's Desired Features for Edu Websites

1. **Comparing Different Courses**: Students expressed a need for a tool that could provide a comprehensive comparison of different courses, helping them make informed decisions about their course selections.
2. **Introducing Course Content**: There was a significant demand for detailed course introductions, including information on teaching content, study goals, assessment methods, and more. This feature would help students better understand what each course offers and its requirements.
3. **Providing Key Information from Online Literature**: Students wanted an LLM to extract and summarize key points from academic literature, making it easier for them to grasp essential information quickly.

4. **Ranking the Workload of the Course**: Students wanted a feature that could rank the workload of different courses to help them plan their studies effectively. However, this study will not work on this feature since the data on the workload of each course is not public.
5. **Giving Past Student's Comments on the Course**: Students expressed interest in a feature that could provide insights from past students' comments to get a better understanding of what to expect from a course. However, this study will not work on this features since the student's comments on courses are not disclosed on the websites.

These insights from the survey were used to define the specific instructional tasks for the LLM. Moreover, some other literature, as mentioned in previous section, also points out the need for AI-based applications or LLMs can play a row in the educational field. By aligning the instruction tuning tasks with the actual needs and preferences of the students, the research aims to create a model that is both relevant and highly beneficial for the educational community at TU Delft.

4.4.2. Task Categories

Based on the results of the student survey and the study from current AI-education research, we then analyzed our data sources and identified relationships among the information. From this analysis, we developed 9 tasks that could lead to the creation of domain-specific instruction instances. These tasks are designed to cover a comprehensive range of educational needs and ensure the model can effectively assist students in various aspects of their academic journey. The tasks are detailed as follows:

1. **Course Content Introduction**:
   - **Task**: Introduce the course content to students based on the course name.
   - **Goal**: Provide a brief and concise summary of what the course covers, helping students understand the main topics and concepts of the course.
2. **Study Goal Answering**:
   - **Task**: Answer the study goal of a course based on the provided course name.
   - **Goal**: Explain the primary objectives and intended learning outcomes of the course to help students understand what they are expected to achieve upon completion.
3. **Assessment Method Answering**:
   - **Task**: Provide the assessment method of a course based on the provided course name.
   - **Goal**: Describe the evaluation and grading criteria used in the course, including exams, assignments, projects, and participation, to inform students how their performance will be measured.
4. **Prerequisites Introduction**:
   - **Task**: Provide the expected prior knowledge of a course based on the provided course name.
   - **Goal**: Inform students of the prerequisite knowledge or skills they should have before taking the course, ensuring they are adequately prepared.
5. **Teaching Materials Answering**:
   - **Task**: Provide the teaching materials of a course based on the provided course name.
   - **Goal**: List the textbooks, articles, and other educational materials that will be used in the course, giving students a clear idea of the resources required.
6. **Course Name Prediction**:
   - **Task**: Predict the name of a course based on a given description.
   - **Goal**: Help students identify courses that match their interests based on the description of the course content and objectives.
7. **Pairwise Courses Comparison**:

- **Task**: Compare two courses and provide reasons for the comparison.
- **Goal**: Highlight the similarities and differences between two courses, including their content, objectives, assessment methods, and relevance to the student’s interests, to help students make informed decisions about which course to choose.

8. **Manual scripts Summarization**:
    - **Task**: Summarize a research work based on its title.
    - **Goal**: Provide a concise summary of the research work, including its objectives, methodology, findings, and significance, to help students quickly grasp the essence of the research.
9. **Research Work Keywords Prediction**:
    - **Task**: Predict the keywords of a research work based on the abstract.
    - **Goal**: Identify the main topics and concepts of the research work to facilitate better indexing and searchability.

4.4.3. Task Templates

Tasks are illustrated through different templates, which serve as fill-in-the-blank frameworks. These templates allow the LLMs to perform summarization and other relevant tasks by filling in the blanks, thereby generating new datasets.

Instruction

You are an educational resource chatbot and you are introducing course content to students.

Input Prompt

Prompt: Would you please introduce the course **[Course Topic]** to me?.

Generated Response

Response: You might be interested in ’**[Course Name]**’ (**[Course Code]**), **[Course Summary]**, making it ideal for your interest in this area.

4.4.4. Example

Instruction

You are an educational resource chatbot and you are introducing course content to students.

Input Prompt

Prompt: Would you please introduce the course **Machine Learning** to me?

Generated Response

Response: You might be interested in ’**Machine Learning 1**’ (**ML101**), This course covers the fundamental concepts and techniques used in machine learning, including supervised and unsupervised learning, neural networks, and deep learning. Students will learn how to apply these techniques to real-world problems through practical examples and hands-on projects., making it ideal for your interest in this area.

4.5. Dataset Generation By Mixtral

In this section, we outline the workflow of utilizing the Mixtral 8x7b model to produce an instruction tuning dataset for various educational tasks. This process involves initializing the model, loading and processing course data, creating prompts, and generating responses for different tasks. Each task is designed to address specific student needs in the educational field.

This research aims to utilize the capabilities of LLMs to generate an instruction tuning dataset with minimal human effort. Finding an effective solution to automate this process is a key focus. Mixtral 8x7b is particularly suited for this task due to its advanced natural language processing capabilities, which allow it to understand, generate, and summarize text from the collected CSV files with high accuracy and relevance. By leveraging Mixtral 8x7b, the research aims to streamline the process of transforming educational data into structured formats suitable for instruction tuning.

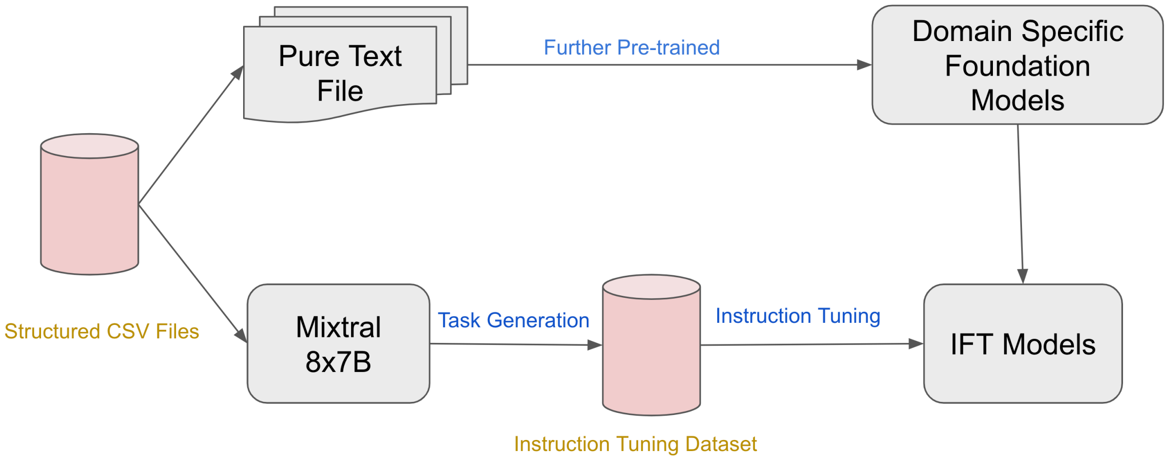


Figure 4.3: Design of Instruction Tuning Dataset Generation and Model fine-tuning

4.5.1. Why Mixtral 8x7b is Suitable: Tradeoffs between Privacy and Performance

Using a LLM like Mixtral 8x7b to generate instruction tuning tasks significantly reduces human effort, particularly in areas such as annotation. Previous instruction tuning projects, like Stanford's Alpaca, have utilized LLMs such as GPT-3.5 to create instruction-tuning datasets. However, this research involves educational data, which must adhere to strict security and privacy protocols. There are considerable privacy concerns associated with uploading educational data to GPT-APIs, as they can pose risks related to data breaches and unauthorized access [48].

To mitigate these privacy issues, we need to utilize open-source LLMs that can be processed on local machines. This approach allows developers to maintain complete control over the data flow, ensuring that educational data remains secure and protected from leaks. In particular, local processing also means that sensitive data never leaves the institution's secure environment, aligning with best practices for data privacy and security.

Choosing a suitable open-source LLM is critical to generating an instruction tuning dataset that is both high-quality and efficient. During the selection process, the pool of open-source LLMs considered included Gemma, Mixtral, and Llama2/3. Mixtral was ultimately selected due to its superior performance on summarization benchmarks, which is crucial as our generation tasks involve summarizing content from the original CSV files [1]. Summarization is a key task during this research's dataset generation because it enables the conversion of extensive educational content into concise and fact-based instruction-tuning examples.

Moreover, Mixtral 8x7b offers faster inference times compared to other models, which further supports its suitability for this research. This faster processing speed allows for more efficient data handling and task generation: the model could be an ideal choice for creating a large instruction tuning dataset within a reasonable timeframe. Faster inference times also mean that the model can be iteratively improved and tested more quickly, accelerating the overall research process.

4.5.2. Workflow of Workflow of Using Mixtral 8x7b for Dataset Generation

1. **Dataset Preparation and Model Initialization**: The process begins with preparing the dataset by loading the relevant educational information from a CSV file. This step involves using the pandas library to read the CSV file and select a manageable subset of data for initial processing and testing. Next, the Mixtral 8x7b model is initialized using the Ollama library. This step sets up the necessary environment for the language model to operate effectively. The initialization involves specifying the model to be used and ensuring that it is ready to process the prompts and generate responses.
2. **Template Creation**: For each task in the dataset, pre-created templates with blanks are used to let the model summarize the required information from the CSV file and fill in the blanks of the template. These templates are developed by the researchers to standardize the process. This involves defining user input, model output, and model instructions based on the tasks, ensuring relevance and consistency. The prompts generated from these templates are then used to produce responses from the Mixtral 8x7b model.
3. **Summarizing and Generating Content**: The Mixtral 8x7b model is utilized to fill in the blanks of the human-created templates for various tasks. This involves using the pre-defined prompts to instruct the model such as summarizing content, providing course comparisons, and answering other specific educational queries. By filling in the blanks of these templates, the Mixtral 8x7b model generates concise and relevant summaries and responses, making it easier for users to understand the main points and concepts covered.
4. **Filtering and Matching Data**: The educational data is filtered to find relevant matches based on the task requirements. This step involves using string-matching techniques to identify relevant information in the dataset. Once matching data is found, the response is generated and structured to include all relevant details, ensuring comprehensiveness.
5. **Instruction Tuning Dataset Generation**: The unique tasks are iterated over, creating prompts and generating responses using the Mixtral 8x7b model. Each entry in the dataset contains the instruction, input prompt, and the generated response. This structured approach ensures that the dataset is ready for instruction tuning, providing a robust foundation for fine-tuning LLMs.
6. **Saving the Dataset**: The final step involves saving the generated instruction tuning dataset to a JSON file. This step ensures that the dataset is stored in a simple and easy-to-use format, suitable for subsequent instruction fine-tuning.

The output of this process is JSON files containing instruction-tuning datasets with different tasks. This dataset is ready to be integrated into the next pipeline for LLMs fine-tuning.

4.6. Data Quality Assessment

4.6.1. Human Assessment

In this research, the ground truth is defined by the information in the study guide and papers repository, which appears on the education websites and is then stored in the CSV file from data extraction in Stage 1. For instance, the attributes ”course content” and ”abstract” are written by humans and verified by students, professors, or instructors. Therefore, we can treat the information from educational websites as ground truth. The instruction dataset created by Mixtral should have extracted key information from these attributes. In particular, Mixtral is primarily used to summarize attributes like course content and form new output instruction instances given the template.

The task of summarization in NLP has utilized the Rouge Score for a long period, and its effectiveness is well-proven. In contrast, machine metrics like the Rouge Score are effective for measuring the overlap of n-grams words, but they fall short of capturing the semantic similarity between the generated content and the ground truth. The Rouge Score may not adequately reflect the nuances of meaning and context, which are essential for evaluating educational content.

To address these limitations, human assessment is better than machine metrics and thus necessary to ensure a comprehensive examination of the instruction tuning dataset. this method involves inviting testers to assess how closely the instruction-tuning dataset aligns with the ground truth data in terms of semantic meaning. This process involved 5 testers, who are all students of TU Delft. Testers are randomly assigned samples from the instruction tuning dataset.

In particular, five testers are randomly given 50 samples from the instruction tuning dataset. Based on the task type of instruction, they will manually search for and compare the generated instruction instance with the corresponding ground truth from the study guide or paper repository. This manual comparison ensures a thorough evaluation of semantic similarity, which machine metrics alone cannot provide.

The human evaluation is based on five key metrics, each rated on a 5-point Likert scale, ranging from 0 to 5. These metrics are derived from a survey of human evaluation of automatically generated text to ensure a thorough assessment of the generated content [43]. The evaluation focuses on the following criteria:

1. **Relevance**: Measures how well the instruction instance aligns with the specific educational task or query. This ensures that the content is pertinent and useful for the intended purpose.
2. **Correctness**: Assesses the correctness and factuality of the information provided. This metric ensures that the generated content is reliable and factually accurate compared to the ground truth data on educational websites.
3. **Grammar**: Evaluates the syntax, punctuation, and overall fluency of the instruction dataset. Proper grammar is essential for maintaining the professionalism and readability of the content.
4. **Clarity**: Examines how easily the instruction dataset can be understood, focusing on the logical flow and simplicity of the language used. Clear content enhances the user's comprehension and learning experience.
5. **Usefulness**: Assesses the practical value and applicability of the instruction dataset to the user's needs. This metric ensures that the content is not only accurate and relevant but also beneficial and actionable for the user.

4.6.2. Assessment Result

Table 4.3: Human Assessment Scores for Tasks Design

Task (Model)	**Relevance**	**Accuracy**	**Grammar**	**Clarity**	**Usefulness**
Task Generation(Mixtral)	4.8	4.9	4.7	4.7	4.5

We collected the assessment scores from the five metrics and took an average. The assessment scores for the task generation using the Mixtral model demonstrate high performance across all five metrics. Both relevance and accuracy show strong results, which suggest that the generated content aligns well with specific educational tasks or queries. Moreover, the instruction instance is factually correct and reliable. This high degree of relevance and accuracy ensures that the users find the generated information trustworthy, which is crucial for educational purposes.

In addition to relevance and accuracy, the model also performs well in terms of grammar and clarity. The high scores in grammar suggest that the content produced by the model is grammatically correct, maintaining proper syntax, punctuation, and overall fluency. Similarly, the clarity score indicates that the content is easy to understand, with logical flow and simple language, which helps users comprehend the main points and concepts without confusion.

Lastly, the usefulness of the generated content is also rated highly. This means that the information provided by the model is practical and beneficial for students, meeting their needs in the educational sector. The combined high performance across relevance, accuracy, grammar, clarity, and usefulness metrics suggests that the Mixtral model could be a reliable tool for generating a high-quality educational instruction tuning dataset that is informative.

4.6.3. Krippendorff's Alpha Analysis

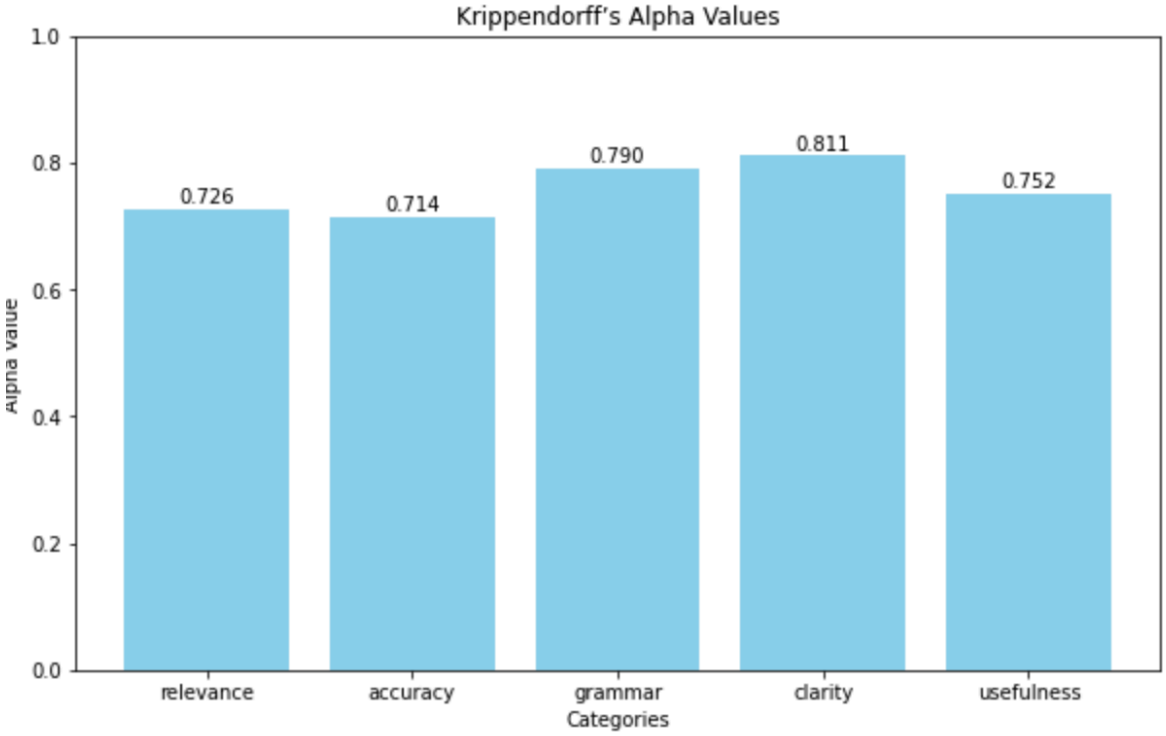


Figure 4.4: Krippendorff's Alpha Values

The Krippendorff's alpha values presented in Figure 4.4 provide insight into the agreement levels among different evaluators on various metrics used to assess the quality of the instruction-tuning datasets. Krippendorff's alpha is a statistical measure used to assess the reliability of agreement among raters, with values ranging from -1 to 1 [22]. Higher values of this coefficient indicate stronger agreement.

The relevance metric has an alpha value of 0.726, indicating moderate agreement among the evaluators. While the agreement is relatively strong, it is lower compared to some other metrics. This may be because different evaluators have slightly varied perspectives on what constitutes relevance in the context of educational content, leading to some discrepancies in their ratings.

The accuracy metric has the lowest alpha value at 0.714. This suggests that there is less agreement among evaluators regarding the factual correctness of the content. This could be due to differences in individual evaluators' knowledge levels. Thus they have various abilities to verify the factual accuracy of the provided information, so it results in varied assessments.

The grammar metric shows a higher alpha value of 0.790 thus it shows substantial agreement among evaluators. This suggests that grammatical correctness is relatively straightforward for evaluators to assess consistently, likely because grammar rules are more objective and standardized.

Clarity has the highest alpha value at 0.811, which means there is a strong agreement among evaluators. This suggests that evaluators find it relatively easy to assess how clearly the content is presented. Clear and logical flow of information is likely more universally understood and agreed upon.

The usefulness metric has an alpha value of 0.752, indicating moderate to strong agreement. Evaluators seem to have a fairly consistent understanding of how useful the content is for the intended educational purposes, though there may still be some subjective differences in how usefulness is perceived in the field of education.

Explanation for Differences

The variations in Krippendorff's alpha values can be attributed to several factors:

- **Subjectivity**: Metrics like relevance and usefulness are inherently more subjective than grammar and clarity. Evaluators may have different interpretations based on their individual experiences and expectations, leading to greater variability in ratings.
- **Knowledge and Expertise**: The accuracy metric likely shows the lowest agreement because it requires evaluators to have a certain level of expertise or knowledge to verify factual correctness. Differences in evaluators' knowledge levels can result in inconsistent assessments.
- **Standardization**: Grammar and clarity are more standardized metrics, with clear rules and guidelines that evaluators can follow. This reduces the room for subjective interpretation and results in higher agreement.
- **Complexity of Content**: The complexity and nature of the content being evaluated can also influence agreement levels. More complex or ambiguous content might lead to greater discrepancies in evaluations for relevance and accuracy.

In all, the Krippendorff's alpha values suggest that while there is a reasonable level of agreement among evaluators for all metrics, the highest agreement is found in more objective areas such as grammar and clarity. In contrast, metrics that require subjective judgment or specialized knowledge, like relevance and accuracy, show more variability in evaluations. This highlights the importance of clearly defining evaluation criteria and ensuring evaluators have adequate knowledge and training to assess content consistently.

4.7. Discussion

Table 4.4: Comparison of Domain Adaptation in Educational Field and Self-Instruct Approach

Dimension	**Self-Instruct Approach**	**Our Work (Educational Domain)**
Data Source and Type	- Small set of human-written seed tasks - General-purpose tasks (summarization, classification, QA)	- Semi-structured educational websites data - Domain-specific tasks in education
Task Generation Methodology	- Bootstrapping from human-written tasks using LLMs - Iterative process with steps like generating task instructions and filtering	- Automated data extraction from educational websites - Data cleaning and LLMs generating instruction-tuning datasets without human intervention during generation
Human Involvement	- Initial seed tasks written by humans - Human evaluation for filtering and data quality - High initial human effort	- Human involvement mainly in final evaluation for quality - Minimal human effort during data generation, relying on automation
Scalability and Cost	- Scales through iterative LLM generation - Costs associated with human-written tasks and GPT-API	- High scalability due to automation - Lower costs, avoiding extensive human annotation and using free LLM resources
Domain Adaptation Focus	- General-purpose domain adaptation - Broad, unspecific task generation	- Specific to educational domain - Tailored to educational content, improving LLMs' performance in educational tasks
Evaluation Metrics	- Human judgments and standard NLP metrics - Measures general instruction-following ability	- Domain-specific human assessment metrics (relevance, accuracy, grammar, clarity, usefulness) - Assesses model performance in generating educational content and practical applicability

To answer **RQ1 (How can we transform semi-structured educational website data into an instruction-tuning task dataset that the model can learn from?)**, the first part of this research demonstrates a comprehensive pipeline to extract and transform semi-structured web educational data into instruction-tuning dataset instances. This process involves several critical steps, including task formulation, template creation, data generation by LLMs, and human evaluation of dataset quality. By leveraging these instruction-tuning datasets to train domain-specific models in education, our study shows promising improvements compared to baseline general-purpose LLMs. Our findings align with previous works like Alpaca and Self-Instruct, indicating the suitability of using LLMs to generate instruction-tuning datasets, provided these datasets are verified to be factual and meet students' needs.

However, unlike approaches in earlier studies that often constructed instances from structured data sources, our source datasets are semi-structured in HTML form. This research first transforms the data to a structured form and introduces a novel approach by leveraging LLMs to fill the blanks in task templates, primarily utilizing the summarization capabilities of LLMs. In comparison to the self-instruct method, our approach specifically caters to the educational domain, focusing on creating training data for domain-specific LLMs. By surveying to understand and meet students' needs, we ensure that the generated content is relevant and practical.

Additionally, another insight for this method is the potential for reducing human effort in dataset generation. Like Self-Instruct and Alpaca, the traditional methods of creating datasets often require extensive human annotation, which is time-consuming and costly. We leverage the advantage of current educational web data, which has already been verified by humans and contains a wealth of valuable information. Moreover, our method produced high-quality instruction instances while minimizing the costs associated with LLMs inference. If this entire generation process were conducted using commercial LLMs API, similar to the approach taken by Alpaca, it could cost approximately 400$. Additionally, this cost would scale significantly as the size of the educational data increases.This automated approach ensures scalability and cost-efficiency, making it feasible for broader applications across various educational domains. Moreover, the comprehensive human evaluation process, supported by Krippendorff's alpha analysis, highlights the reliability and consistency of the generated datasets that form a solid foundation for the subsequent stage of fine-tuning expert models.

This study has several implications and recommendations for future research and practice.

First, the pipeline we developed enhances automation in the dataset generation process and it is scalable and cost-effective. Developing more sophisticated algorithms for identifying relevant educational content and prompting LLMs could streamline the pipeline.

Second, we focus on the technique of LLMs summarization and fact-based data generation. Our approach keeps the accuracy and relevance of the generated datasets from human assessments, which is crucial for educational applications. To improve the reliability of human assessments, researchers should explore more robust training programs for evaluators and incorporate additional metrics that capture the nuanced aspects of educational content. This could include domain-specific criteria that better reflect the quality and applicability of the generated datasets.

Third, the methodology introduced in this research can be applied to other semi-structured data sources beyond educational content. The approach leverages automated data extraction and task generation, which can be easily adapted to handle the unique structures and requirements of different domains. This opens up possibilities for creating domain-specific LLMs in various fields such as healthcare, finance, and legal sectors, where web information needs to be effectively transformed into training datasets.

In all, our study opens possibilities for further research into improving automated data extraction, validation processes, and even higher-quality datasets for instruction tuning in various domain-specific domain.

5

Stage 2: Model Instruction Tuning

5.1. Overview

In this chapter, we delve into the process of further pretraining and instruction fine-tuning of several foundation models using the datasets we have prepared. This section sets the stage for detailed analysis from different perspective in the next chapter, which aims to answer **RQ2: how can the different training techniques impact on the performance of domain-specific educational LLMs?**. Our chosen platform for the first half of this task is Llama Factory[55], a comprehensive framework designed to streamline and optimize the fine-tuning of large language models (LLMs). We will first explore the capabilities and features of Llama Factory, discuss our methodology for further pretraining, and detail the steps taken for instruction fine-tuning.

5.2. Platform: Llama Factory

Llama Factory is a unified framework that integrates a suite of cutting-edge efficient training methods and it allows for the flexible and customizable fine-tuning of over 30 LLMs [55]. The platform is designed to minimize the need for environment installation of different models through its built-in web UI, LLAMABOARD, which provides an intuitive interface for configuring and monitoring the training processes.

5.2.1. Features of Llama Factory

- **Model Compatibility:** Supports a wide range of models including LLaMA2/3, Mixtral, and Gemma.
- **Resource Efficiency:** Utilizes 32-bit full-tuning, 16-bit freeze-tuning, and various low-bit tuning methods like LoRA to optimize GPU memory usage.
- **Experiment Monitoring:** Supports experiment tracking and monitoring with tools like LlamaBoard, TensorBoard, Wandb, and MLflow.
- **Faster Inference:** Provides OpenAI-style API, Gradio UI, and CLI with vLLM worker for faster and more efficient inference.

5.3. Further Pretraining

Further pre-training involves continuing the training of a pre-existing model on additional data to enhance its generalization performance on specific tasks. This step leverages the extensive dataset we have created in stage 2, allowing the model to learn additional patterns and information before fine-tuning.

5.3.1. Main Workflow

1. **Model Initialization:** Models are loaded and initialized using the AutoModel API of Transformers. This ensures compatibility with various model architectures by establishing a model registry. For instance, the `Gemma` model is instantiated with preloaded configurations and weights. The model

configuration is loaded from cached files, and the model's architecture details, such as hidden size, number of layers, and attention heads, are pre-defined and set up.

2. **Adapter Attachment:** The adapter LoRA is attached to appropriate layers to facilitate efficient pretraining. The floating-point precision is managed based on the capabilities of the training devices. Typically, half-precision (float16) is used to optimize performance and memory usage without compromising model accuracy. The model's weights are initialized and configured to use float16 precision, and auto half-precision backend is utilized during training.
3. **Data Preparation:** The datasets are loaded, aligned, merged, and pre-processed to standardize them for training. Tokenization is performed using preloaded tokenizer configurations to convert textual data into model-readable tokens. For example, the tokenizer configuration, model, and special tokens are loaded from cache. The datasets are processed using multiple processing units to speed up the tokenization process. This step converts raw text into input IDs that the model can understand, ensuring that all tokens are correctly mapped.
4. **Training Execution:** The training process involves setting up batch sizes, gradient accumulation steps, and the total number of optimization steps. Real-time monitoring and adjustments are facilitated through LLAMABOARD, which provides detailed logs and loss value per step. The training is managed by `transformers.trainer`, which handles the optimization steps, logging of training loss, and learning rate adjustments.
5. **Model Saving:** Periodically, the model's state, including weights and configurations, is saved to checkpoints. This ensures that progress is not lost and allows for resuming training or performing evaluations at different stages. The model configuration and tokenizer files are saved in the checkpoint directory. It ensures that the model can be reloaded with the same settings for future use. Checkpoints are created at regular intervals and they capture the model's state at different stages of training.

5.4. Instruction Tuning

Instruction fine-tuning further tailors the model to follow specific instructions or perform particular tasks more effectively. This process enhances the model's ability to generate accurate and relevant responses based on the given instruction dataset.

5.4.1. Methodology

- **Data Worker:** A data processing pipeline is employed to load, align, merge, and preprocess the datasets, standardizing them into a unified format.
  - **Dataset Loading:** The datasets are loaded using the llamafactory data loader, ensuring compatibility and efficient handling of large datasets.
  - **Dataset Tokenization:** The loaded datasets are tokenized using a pre-trained tokenizer from the previous pre-trained model or hugging face. This includes loading necessary tokenizer files such as tokenizer.model, tokenizer.json, and special-tokens-map.json.
- **Training Execution:** The fine-tuning is carried out using state-of-the-art methods like LoRA, integrated into the Llama Factory framework.
  - **Model Download and Initialization:** The model weights and configuration are downloaded and initialized from its previous pre-trained version or directly from Hugging Face Hub. Gradient checkpointing and torch SDPA are used to optimize memory usage and training speed.
  - **Fine-Tuning with LoRA:** Technique of LoRA is employed, where trainable adapters are added to the model's layers. This allows for efficient fine-tuning by updating only a small subset of model parameters.
  - **Precision Management:** The model and trainable parameters are managed in mixed precision (float16 and float32) to leverage GPU capabilities and optimize performance.
  - **Training Configuration:** The training process is configured with specific settings, such as the number of epochs, batch size, gradient accumulation steps, and learning rate. Models are fine-tuned on task-specific datasets to improve their performance on structured tasks like

question answering, text summarization, and more. During each training step, the loss is calculated and backpropagated to update the model parameters. Metrics such as gradient norm and learning rate are monitored to ensure stable training. Real-time monitoring of training loss and other metrics like Bleu is performed.

- **Real-time Monitoring:** The system provides tools for monitoring training logs and loss curves in real time, offering insights into the training progress and allowing for timely adjustments.
- **Checkpointing and Model Saving:** Periodic checkpoints are saved during the training process to ensure that progress is not lost and to facilitate resumption in case of interruptions. Upon completion of training, the final model, along with its tokenizer and configuration files, is saved for future evaluation and deployment.

5.4.2. Human Evaluation

In this section, we outline the process for conducting human evaluation to assess the quality and effectiveness of the fine-tuned models. Human evaluation is critical to ensure that the model-generated content meets the standards of relevance, accuracy, and utility required for educational purposes.

Testers will interact directly with the fine-tuned models. For each output generated by the model, testers will search for the corresponding information on educational websites and compare the results. This interaction will provide a practical evaluation of how well the model performs in real-world scenarios.

The human evaluation is based on five metrics, derived from a survey of NLP tasks mentioned in previous chapter, to ensure a comprehensive assessment:

1. **Relevance**: Measures how well the generated content aligns with the specific educational task or query. This ensures that the content is pertinent and useful for the intended purpose.
2. **Accuracy**: Assesses the correctness and factuality of the information provided. This metric ensures that the generated content is reliable and factually accurate.
3. **Grammar**: Evaluates the syntax, punctuation, and overall fluency of the generated content. Proper grammar is essential for maintaining the professionalism and readability of the content.
4. **Clarity**: Examines how easily the generated content can be understood, focusing on the logical flow and simplicity of the language used. Clear content enhances the user's comprehension and learning experience.
5. **Usefulness**: Assesses the practical value and applicability of the generated content to the user's needs. This metric ensures that the content is not only accurate and relevant but also beneficial and actionable for the user.

The evaluation process will involve the following steps:

- **Introduction and Purpose**: The testers are introduced to the content of this research, the purpose of the evaluation, and how to interact with the model on LlamaBoard.
- **Output Generation**: The model generates responses based on the user query.
- **Information Retrieval**: Testers search for the corresponding information on educational websites.
- **Scoring Rubrics**: Scoring rubrics are presented to testers to guide them in scoring the tasks effectively.
- **Comparison and Rating**: Testers compare the model-generated content with the information retrieved from the websites and rate the outputs based on the five metrics.

By following this evaluation framework, we aim to obtain a detailed understanding of the model's performance and identify areas for further improvement.

5.5. Experiments Overview

In this section, we will provide an overview of the experiments designed to analyze and evaluate the performance of various domain-specific educational LLMs. These experiments are crucial for answering

RQ2: how can the different training techniques impact on the performance of domain-specific educational LLMs?

5.5.1. Experiment 1: Effect of Further Pre-training

This experiment focuses on evaluating the impact of further pre-training on the performance of the Llama2 model within the educational domain. The goal is to determine how additional pre-training on domain-specific educational data enhances the model's ability to generate accurate and relevant responses. By comparing models with and without further pre-training, we aim to quantify the improvements using ROUGE scores.

Understanding the effect of further pre-training is vital for assessing whether extending the training of a pre-existing model on domain-specific data can lead to significant performance improvements. This experiment will help in determining the necessity and effectiveness of this additional training phase.

5.5.2. Experiment 2: Effect of Multitask Training

This experiment examines how training a model on multiple tasks influences its performance on a specific task. We compare the performance of the Llama3 model trained exclusively on one task versus the model trained on three and five tasks. The objective is to evaluate whether multitasking training enhances the model's ability to handle individual tasks effectively and how the introduction of different types of data sources impacts performance.

In the educational domain, it is still unclear how multitask training can potentially affect a model's generalization capabilities and robustness. This experiment aims to explore the benefits and challenges of multitask training, particularly in handling correlated and diverse datasets within the educational domain.

5.5.3. Experiment 3: Performance Comparison of Different Models

This experiment compares the performance of several popular open-source models, including Llama2, Llama3, Gemma, and Mixtral, by performing instruction tuning on educational data. The goal is to identify which model performs best in generating relevant and accurate educational content. The evaluation involves both machine metrics and human assessments to provide qualitative insights into the models' performance.

Selecting the right model under limited physical resources is crucial for achieving optimal performance in domain-specific tasks. This experiment provides a comprehensive comparison of different models, highlighting their strengths and weaknesses, and guiding the selection of the most suitable model for educational applications.

6 Experiment and Results

This chapter provides a comprehensive analysis of the experiments conducted to evaluate the performance of various domain-specific LLMs in the educational domain. The primary focus is to answer **RQ2: how can the different training techniques impact on the performance of domain-specific educational LLMs?**. The experiments are designed to assess how further pre-training, multitask training, and model selection affect the models' ability to generate accurate, relevant, and clear educational content. The findings are based on both quantitative metrics and human evaluations, offering a detailed understanding of the effect of each training factor.

6.1. Experiments Setup

6.1.1. Datasets

In this section, we provide a detailed description of the datasets used for pretraining and fine-tuning the models. This includes their sources, key statistics, and preprocessing steps undertaken to prepare the data for training.

Dataset Sources

The datasets employed for this study were sourced from various educational resources, including course descriptions, study goals summaries, and other academic-related documents. The detailed statics information of these datasets is in . These sources provided a rich and diverse set of data suitable for training models aimed at educational purposes.

Key Statistics

The key statistics for the datasets used in pretraining and fine-tuning are as follows:

- **Number of Inputs:** The total number of Inputs included in the datasets.
- **Total Tokens:** The combined number of tokens across all documents.
- **Average Input Length:** The average number of tokens per entry.

Preprocessing Steps

The preprocessing of the datasets involved several key steps to ensure that the data was in a suitable format for training the models. These steps included:

- **Cleaning:** Both the pre-training raw text file and instruction tuning dataset were filtered to remove non-English content. Additionally, any data entries that were empty were removed to ensure the integrity and quality of the dataset.
- **Tokenization:** The text data was tokenized using a tokenizer compatible with the model architecture, ensuring that the data could be efficiently processed and the model could learn from the input.

Table 6.1: Key Statistics for the Datasets

Dataset	Number of Inputs	Total Tokens	Average Input Length	Unique Tokens
Course Content Introduction	1700+	500,000+	500+	10,000+
Course Study Goals Summarization	1400+	250,000+	500+	12,000+
Course Assessment Summarization	1200+	170,000+	500+	8,900+
Prerequisites Summarization	800+	110,000+	200+	3,200+
Teaching Material Answering	600+	90,000+	100+	5,700+
Course Name Prediction	1700+	500,000+	500+	10,000+
Courses Comparison	3000+	1,200,000+	500+	18,000+
Manual Scripts Summarizations	5000+	1,200,000+	500+	73,000+
Research Keywords Predictions	5000+	1,100,000+	500+	69,000+
Course Data Raw Text	5000+	/	/	/

- **Shuffling:** The datasets were shuffled to prevent any order bias during training, ensuring that the model could generalize well across different topics and formats.

6.1.2. Models

In this study, we utilized several state-of-the-art language models to evaluate their performance in various experiments by using the instruction tuning dataset in the educational domain. The models used include Llama2, Llama3, Gemma, and Mixtral. All models were obtained from Huggingface, a widely used repository for machine learning models. Some models, such as Llama3 and Mixtral, required access permissions that needed to be requested from the respective model maintainers.

6.1.3. Training Environment

The training environment for this study was designed to leverage high-performance computing resources to efficiently further pretrain and fine-tune the models. Below are the detailed hardware specifications, software frameworks, and configurations used during the experiments.

Hardware Specifications

The experiments were conducted on a system with the following hardware specifications:

- **GPU:** 1x NVIDIA A100 (40 GB SXM4), except for Mixtral which used 2x NVIDIA A100.
- **CPU:** 30 CPU cores
- **Memory:** 205.4 GB RAM
- **Storage:** 525.8 GB SSD

Software Enviroment

- **Operating System:** Ubuntu 20.04 LTS
- **Deep Learning Framework:** PyTorch 1.9.0
- **Transformers Library:** Hugging Face Transformers 4.9.2
- **Experiment Tracking:** LlamaBoard, TensorBoard

6.1.4. Training Procedure

The training procedure encompasses both the pretraining and fine-tuning processes, utilizing advanced methods such as LoRA to enhance model performance. Key hyperparameters are similar to the setting from the Alpaca project, and the steps are detailed below.

Pretraining and Fine-Tuning Process:

- **Data Loading:** The datasets are loaded from the specified directory, and preprocessing is performed using 16 parallel workers to ensure efficiency. The data is filtered to exclude non-English content and remove any empty entries, maintaining high data quality.
- **Model Initialization:** The base model is loaded and initialized with its pre-trained weights. Specific settings, such as enabling half-precision floating point (fp16) and automatic flash attention, are configured to optimize performance in limited computational resources.

- **LoRA Configuration:** LoRA (Low-Rank Adaptation) is employed for fine-tuning. The configuration includes parameters such as *lora_alpha* set to 16, *lora_dropout* set to 0, and *lora_rank* set to 8.
- **Training Configuration:** The training procedure uses a batch size of 8 per device, with gradient accumulation steps set to 8. The learning rate is configured at 0.0003, and the AdamW optimizer is used with a cosine learning rate scheduler. Gradient clipping is applied with a maximum gradient norm of 1.0 to prevent gradient explosion.
- **Training Execution:** The training process is executed for 3 epochs, with logging steps set to every 5 steps to monitor progress. The model undergoes both pretraining and fine-tuning stages, leveraging efficient techniques like LoRA to enhance adaptability and performance.
- **Model Saving:** The trained model and its configurations are saved at specified intervals (every 100 steps) to ensure checkpoints are available. The final model is stored in the designated output directory for future inference and evaluation.

6.2. Experiment 1: Effect of Further Pre-training

The purpose of this experiment is to evaluate the impact of further pre-training on the performance of the Llama2 model, specifically in the educational domain. This involves comparing the model's performance with and without additional pre-training steps under the same hyperparameter settings. The goal is to understand how further pre-training on educational data affects the model's ability to generate accurate and relevant responses. The evaluation focuses on machine metrics, namely BLEU and ROUGE scores, to quantify the improvements. For this part of the experiment, we only tested the task research key words prediction.

To conduct this experiment, the following steps were undertaken:

6.2.1. Dataset Preparation

- The dataset of both the instruction tuning dataset and raw data file was split into training and testing sets in a ratio of 70: 30.
- The test set was used to evaluate the model's performance. Specifically, 300 entries were randomly selected from the test set for the task research key words prediction.

6.2.2. Model Training

- **Baseline Model:** The Llama2 model without any additional pre-training or instruction tuning.
- **Pre-trained Model:** The Llama2 model subjected to further pre-training on a raw educational text file.
- **Instruction Tuned Model (IT):** The Llama2 model fine-tuned with instruction tuning on the educational dataset.
- **Combined Model (IT+Pretrained):** The Llama2 model that underwent both further pre-training and instruction tuning.

6.2.3. Evaluation Metrics

The performance of the models was evaluated using BLEU and ROUGE scores, which are standard metrics for assessing the quality of generated text against reference texts. For this experiment, BLEU scores indicate that how the generated keywords match the reference keywords in terms of precision. Moreover, ROUGE scores indicate that how the generated keywords recall the reference keywords.

6.3. Experiment 2: Effect of Multitasks

This experiment aims to evaluate how training a model on multiple tasks affects its performance on a specific task. Specifically, we compare the task performance of the Llama3 model trained exclusively on one task against the Llama3 model trained on three and five tasks. The purpose is to determine if multitasking training enhances the model's capability to handle individual tasks effectively and how introducing different types of data sources impacts performance.

The purpose of this experiment is to understand how instruction tuning with multiple tasks affects the model's performance on the specific task of introducing course content. The models evaluated include:

- **Llama3 (One Task):** Trained only on the task of introducing course content.
- **Llama3 (Three Tasks):** Trained on three tasks - introducing course content, answering the assessment method, and answering the study goals of a course.
- **Llama3 (Five Tasks):** Trained on five tasks - introducing course content, answering the assessment method, answering the study goals of a course, summarizing a thesis abstract, and predicting the keywords of a paper.

In particular, the first and second models' instruction tuning datasets are highly correlated as they are sourced from the course selection website and the tasks all introduce the content from different perspectives. However, the third involves an additional dataset from the paper repository. This setup helps us analyze: 1) How the number of similar tasks affects performance. 2) How the introduction of a different dataset type impacts performance.

6.3.1. Human Evaluation

The three models were evaluated by five students from TU Delft based on the following metrics:

1. **Relevance:** How well the generated content aligns with the specific educational task or query.
2. **Accuracy:** The correctness and factuality of the information provided.
3. **Grammar:** The syntax, punctuation, and overall fluency of the generated content.
4. **Clarity:** How easily the generated content can be understood, including logical flow and simplicity of the language used.
5. **Usefulness:** The practical value and applicability of the generated content to the user's needs.

To reduce bias, the testers were invited to interact with the models randomly, querying for course content. Each tester evaluated 25 outputs from each model, totaling 75 evaluations per tester and 300 evaluations overall across all testers. Each model's evaluation metrics were summed up and averaged to provide a comprehensive assessment of its performance.

6.4. Experiment 3: Performance of Different Models

This experiment compares the performance of various popular open-sourced models, including Llama2, Llama3, Gemma, and Mixtral by performing instruction tuning on educational data. The goal is to evaluate which model performs best in generating relevant and accurate educational content. The results are presented in section 7.7.

All models are trained with the same comprehensive instruction-tuning dataset comprising 9 tasks. They share similar training environments and hyperparameter settings as mentioned in section 7.2, except for Mixtral 8x7b, which uses 2xA100 GPUs due to its larger parameter size and higher virtual memory requirements during supervised fine-tuning.

Similarly, like the previous exp2, the experiment also involves human evaluators to assess the quality of the generated content from different models. Testers interact with the models, retrieve information from educational websites, and compare the results based on relevance, accuracy, grammar, clarity, and usefulness. The purpose is to obtain qualitative insights into the models' performance.

The evaluators randomly chatted with each model 18 times. This results in a total of 72 evaluations for each model and the scores are averaged to provide a comprehensive performance assessment.

6.5. Main Results

6.5.1. Effect of Further Pertaining

The results presented in Table 6.2 highlight the significant improvements in performance metrics for the Llama2 model when subjected to further pre-training and instruction tuning. The metrics include BLEU (1-4) and ROUGE (1, 2, L), which collectively offer a comprehensive view of the model's performance for keyword predictions.

Table 6.2: Performance Metrics for Llama2 with and without Further Pre-training

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-1	ROUGE-2	ROUGE-L
Llama2	17.06	13.77	10.00	7.55	9.51	1.92	4.84
Llama2 (Only Pretrained)	19.66	13.64	11.63	9.31	10.73	2.99	5.22
Llama2 (IT)	74.64	67.48	59.37	51.69	**52.00**	31.88	41.02
Llama2 (IT+Pretrained)	**76.37**	**69.12**	**62.48**	**52.22**	51.13	**33.01**	**42.57**

The baseline Llama2 model demonstrates low performance in both BLEU and ROUGE scores, indicating its inability to analyze the paper's abstract and predict related keywords directly. With further pre-training alone, slight improvements are observed. For example, BLEU-1 increases to 19.66 and ROUGE-1 rises to 10.73. This suggests that further pre-training on educational data enhances the model's content understanding and keywords prediction, though the improvements are not dramatic.

The introduction of instruction tuning (IT) yields substantial improvements across all metrics. The Llama2 (IT) model achieves a BLEU-1 score of 74.64 and a ROUGE-1 score of 52.00. These scores reflect a significant leap in performance, suggesting that instruction tuning greatly enhances the model's ability to follow specific tasks, learn from the pattern of the training dataset, and predict related keywords. The Llama2 model that undergoes both instruction tuning and further pre-training (IT+Pretrained) exhibits the highest performance among most metrics except slightly lower in the ROUGE-1 score. This indicates that further pre-training, followed by instruction tuning makes it more effective in predicting the accurate keywords of a literature abstract that it never learned before.

6.5.2. Effect of Multi-tasks

Table 6.3: Human Evaluation Scores for Different Models

Model	Relevance	Accuracy	Grammar	Clarity	Usefulness
Llama3 (One Task)	4.2	4.1	**4.8**	3.8	4.3
Llama3 (Three Similar Tasks)	**4.5**	**4.3**	4.7	**4.1**	**4.4**
Llama3 (Five Different Tasks)	4.3	3.9	4.7	4.0	**4.4**

The results in Table 6.3 provide insights into how training with multiple tasks affects the performance of the Llama3 model on the task of introducing course content.

Impact of Single Task Training: Llama3 trained on only one task (introducing course content) demonstrates good performance with high scores in grammar and usefulness, indicating that focused training on a single task allows the model to generate precise and coherent responses.

Impact of Three Tasks Training: Training on three tasks enhances the model's performance across all metrics. The Llama3 (Three Tasks) model shows improvements in relevance, accuracy, and clarity, suggesting that multitasking within a highly correlated dataset (course selection website) benefits the model's ability to handle individual tasks effectively.

Impact of Five Tasks Training: When training on five tasks, including datasets from the paper repository, the model's performance shows mixed results. While the scores in grammar remain high, slight decreases in relevance and accuracy are observed compared to the model trained on three tasks. This indicates that introducing less correlated datasets may introduce complexity, slightly affecting performance on specific tasks.

In summary, multitasking training with highly correlated datasets enhances overall performance. However, introducing datasets from different sources may require additional adjustments to maintain performance levels across all metrics.

6.6. Further Analysis

6.6.1. Pre-training time

To understand the relationship between pre-training size and the resulting performance metrics, we calculated the percentage increase in performance metrics and recorded the training time for different pre-training sizes.

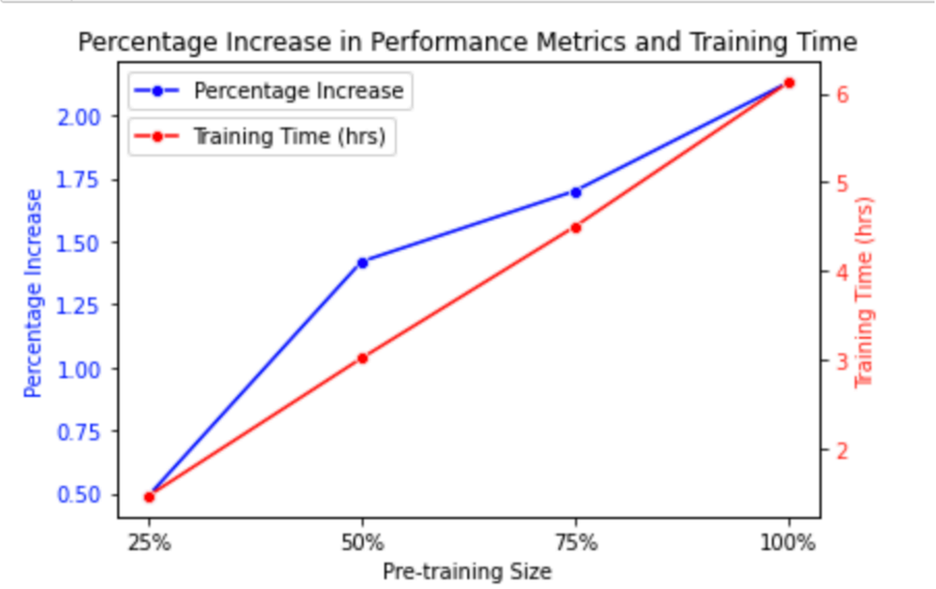


Figure 6.1: Line Chart of Percentage Increase and Training Size

The graph 6.1 shows the percentage increase in performance metrics and the associated training times for different pre-training sizes. The percentage increase is calculated by the average difference of all machine metrics of the two models. The data reveals that as the pre-training size increases from 25% to 100%, the percentage increase in performance metrics also rises, although at diminishing rates. For instance, the percentage increase is 0.49 at 25% pre-training size and 2.13 at 100% pre-training size. Correspondingly, the training time increases from 1.47 hours to 6.12 hours.

Moreover, the line chart shows that while performance continues to improve with larger pre-training sizes, the rate of improvement decreases. This suggests that as more training data is added, the additional performance gains become progressively smaller compared to the time and effort invested.

6.6.2. Performance on Different Models

Table 6.4: Human Evaluation Scores for Different Models

Model	Relevance	Accuracy	Grammar	Clarity	Usefulness
Llama2	3.4	3.8	**4.6**	4.0	4.5
Llama3	**4.3**	4.0	**4.6**	4.2	**4.6**
Gemma	4.2	**4.3**	4.5	3.9	4.5
Mixtral	**4.3**	**4.1**	4.7	**4.4**	4.3

The results from the human evaluation highlight several key differences in model performance. Llama3 and Mixtral demonstrated strong capabilities in relevance and clarity, with both models achieving high scores, suggesting their effectiveness in generating content that aligns well with the given tasks and is easy to understand. Gemma excelled in accuracy, indicating its strength in providing correct and factual information. However, Gemma did not perform as well in clarity, suggesting that its content might be harder to understand. In terms of grammar, Mixtral led the way, followed closely by Llama3 and Llama2, all of which produced grammatically sound content.

However, Llama2 lagged behind the other models in most metrics, showing a noticeable gap in performance, particularly in relevance and accuracy. Despite this, all models performed well overall which demonstrate their capabilities in handling expected educational tasks effectively.

6.6.3. Performance of Different Tasks

In this part, we examine the performance of each task using Llama3. Since most tasks achieve similar evaluation scores in grammar, clarity, and usefulness, we primarily present the average scores for relevance and accuracy for each task to provide more focused insights.

As from the line chart 6.2, the evaluation result shows that the tasks ”Prerequisites Introduction” and ”Research Work Keywords Prediction” scored highest, indicating strong performance in relevance and accuracy for these tasks. Conversely, ”Assessment Method Answering” and ”Manual Scripts Summarization” scored lowest, suggesting the model struggles to produce accurate and fact-based educational results. This finding leaves room for improvement in handling these specific tasks.

In all, this analysis highlights the varying performance levels across different educational tasks and it provides a clear understanding of where the model excels and where further refinement is needed.

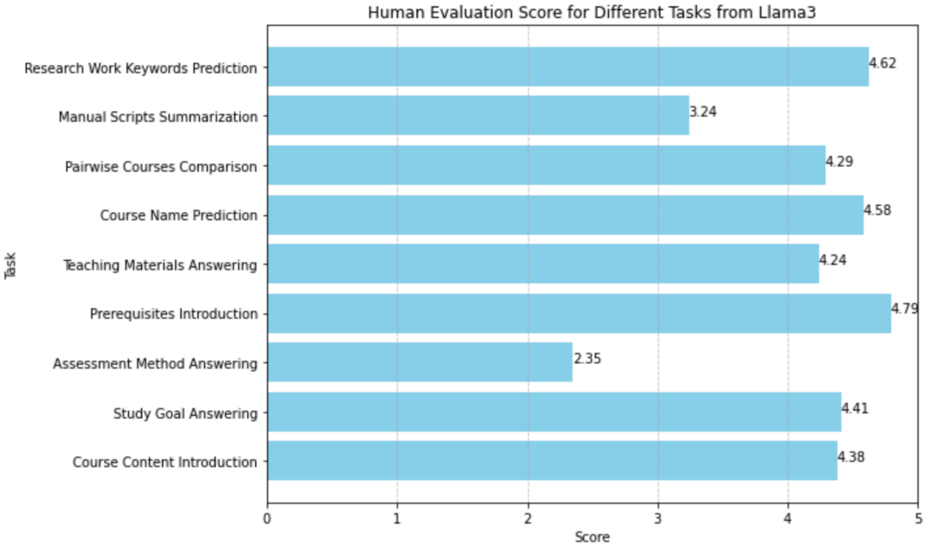


Figure 6.2: Line Chart of Percentage Increase and Training Size

6.7. Discussions

To address **RQ2 (How can different training techniques impact the performance of domain-specific educational LLMs)**, we analyze serveral factors and compare them with previous findings. We also discuss the implications of these findings and how they could influence the study of fine-tuning domain-specific LLMs.

6.7.1. Further Pre-training

The further pre-training experiments highlight improvements in model performance and emphasize the importance and effectiveness of this additional training phase. The finding matches the results observed in earlier studies of McKinzie work’s on the relationship between pre-training and SFT model performance [25].

In particular, further pre-training enables the model to acquire specialized knowledge not typically present in general datasets, crucial for tasks requiring a deep understanding of specific domains like educational content. Moreover, further pre-training enhances the model’s ability to understand and process the structure of educational webpage data. By learning from a domain-specific corpus, the model becomes adept at identifying correlations within the data, such as the relationship between an abstract and its corresponding keywords. Moreover, the results show that combining pre-training with instruction tuning yields the best performance across most metrics. This suggests that a two-step training process—starting with domain-specific pre-training followed by task-specific instruction tuning—can effectively maximize a model's performance.

However, compared to previous studies, this study observes that while performance continues to improve with larger pre-training sizes, the rate of improvement decreases. We speculate that this might be the reason that as the model is exposed to an increasingly larger dataset, the incremental gains in learning become smaller. Initially, additional data introduces new patterns and knowledge, leading to significant improvements. However, as the dataset grows, much of the new data may be redundant, repeating patterns the model has already learned, thus contributing less to overall performance enhancement. Moreover, a second speculation is that increased demand can lead to resource constraints and it potentially limits the efficiency of the training process. Additionally, the model might struggle to efficiently allocate its attention across a vast and diverse dataset, causing it to miss out on learning finer details and key information for specific educational data. In addition, it is important to acknowledge the resource-intensive nature of further pre-training. The computational demands and time required can be substantial. In this study, only three years thesis data was used, yet the training process was still resource-heavy even we deployed LoRA mechanism. Extrapolating this to larger datasets suggests that the time and computational resources required would scale accordingly.

In all, the findings from the further pre-training experiments underscore improvements in model performance, emphasizing the importance and effectiveness of this additional training phase. The observation of diminishing returns with larger pre-training datasets suggests that researchers should initially experiment with different sizes of small subsets to seek performance improvements. If resources such as time and computational power are limited, and a positive trend is observed with smaller datasets, it can justify expanding the dataset incrementally. This approach could ensure efficient use of resources while maximizing the benefits of pre-training and it makes a strategic recommendation for researchers aiming to fine-tune domain-specific language models on specific tasks.

6.7.2. Mutli-Tasks

The multitask training experiments reveal significant insights into how training models on multiple tasks affect their performance. These findings align with those of Zhang et al, and in our study, multitask training demonstrated similar benefits by exposing the models to a variety of tasks, allowing them to better capture the nuances and specific requirements of educational tasks [52]. Instruction tuning with multiple tasks enabled the models to learn a broader range of patterns and contexts, improving their ability to handle individual tasks effectively. This was particularly evident with the Llama3 model trained on three tasks, which outperformed the model trained on a single task in relevance, accuracy, and clarity. The diversity of tasks provided a richer training experience, leading to better generalization.

However, unlike Zhang's study, which did not explore the impact of task similarity and dataset diversity, our study found that introducing datasets from different sources, such as combining course selection data with paper repository data, presents a decrease in model performance. While the model trained on five tasks maintained high grammar scores, there were decreases in relevance and accuracy compared to the model trained on three tasks. This suggests that less correlated datasets and tasks may introduce complexity and require the model to learn different patterns simultaneously, which can affect performance. One reason for this issue could be that during learning, the model has to switch between different contexts and task requirements more frequently. This can lead to a dilution of the model's learning focus, making it harder for the model to achieve high performance in any single task. Another potential reason is that the model might develop overfitting tendencies on more prominent or easier patterns within the diverse datasets. It could neglect the harder, more complex relationships that are crucial for specific educational tasks.

The findings indicate that while multitasking training on similar data sources can enhance overall performance, it is crucial to optimize the selection and combination of tasks. Researchers should experiment with different task combinations to find the optimal balance that maximizes performance. Additionally, introducing new datasets gradually and monitoring the model's performance can help identify the point at which additional tasks no longer contribute to performance improvements or even cause degradation. This approach ensures that the model remains effective and efficient in handling educational tasks.

6.7.3. Performace of Different Models

The comparative performance evaluation of Llama2, Llama3, Gemma, and Mixtral models yields several important implications, highlighting the strengths and limitations of each model in handling educational content. The findings partly align with those of previous studies on the general performance comparison among those open-source models. In particular, Llama3 and Mixtral outperform the old Llama2 as they demonstrate strong capabilities in relevance and clarity, suggesting that training these models on multiple tasks can significantly enhance their ability to generate contextually appropriate and easy-to-understand content. This aligns with the architectural design of Llama3, which uses an extended vocabulary and GQA to balance computational efficiency and performance, enabling it to handle diverse and complex educational queries effectively. Similarly, SMoE architecture allows it to dynamically allocate computational processes, enhancing its ability to generate clear and relevant educational content.

However, although Gemma excelled in accuracy, its lower score in clarity suggests that there might be a trade-off between accuracy and clarity in some models. This implies that while some models can generate highly accurate content, it might be more challenging to ensure that this content is also clear and easily comprehensible. The MQA mechanism used in Gemma, although efficient, might contribute to this trade-off as it simplifies the attention process. It potentially misses out on necessary details for clarity. Future work should focus on balancing these two aspects to enhance overall performance.

Moreover, the need for advanced training techniques is evident from the performance of Llama2, which lagged behind the other models in most metrics, particularly in relevance and accuracy. This highlights the need for more advanced training techniques or further pre-training to improve its performance. While Llama2 is capable, it requires additional optimization and training to reach the performance levels of more advanced models like Llama3 and Mixtral. Llama2's performance could be significantly boosted by leveraging the innovations implemented in its successors, such as the larger tokenizer vocabulary and more efficient attention mechanisms. However, an advantage of Llama2 is its ability to be trained using A10 GPUs, which are more cost-effective compared to the higher-end GPUs required by more advanced models.

Additionally, the Mixtral model, which requires 2xA100 GPUs, demonstrates the trade-off between computational resources and model performance. While Mixtral shows high performance across multiple metrics, it also demands significant computational power. This implies that achieving top-tier performance may come at the cost of increased computational requirements.

In all, the findings from the comparative performance evaluation underscore the need for a strategic approach in selecting and optimizing models for educational content. Balancing accuracy, clarity, and computational efficiency is crucial to developing effective and practical educational LLMs. Future research should continue to explore these trade-offs and investigate innovative training techniques to enhance model performance across various domain-specific tasks.

6.7.4. Limitations

This study, while comprehensive, encountered several limitations that highlight areas for future improvement and exploration.

1. **Numerical Data Handling:** During our observations of interacting with the instruction-tuned models, we noted that the models performed poorly when dealing with numerical data. This problem is also shown in figure 6.2 and this issue particularly affected queries related to assessment methods, which were consistently the weakest among the nine instruction-tuning tasks. Assessment methods often include percentages of assignments and exams for final grades, which the models struggled to handle accurately. Additionally, the models demonstrated difficulty in memorizing course codes, sometimes hallucinating and providing incorrect codes. One potential solution to this problem is to combine the domain-specific model with a Retrieval-Augmented Generation (RAG) approach [23]. This could enhance the model's ability to retrieve accurate numerical data and course codes from a reliable database, thus improving its overall performance in these areas.

2. **Hallucinations:** Despite training on domain-specific instruction instances, the model still exhibits problems with hallucinations. During the human evaluation process, we observed that the model occasionally provides noticeably incorrect information, even when it was trained on specific instruction

instances. The problem could help to explain the low evaluation score of the task ”Manual scripts Summarization” as shown in figure 6.2. For example, when asked about a paper analyzing architectural design in San Francisco, the model might incorrectly state that the paper analyzes an architectural project in the Netherlands. Possible reasons for this issue include the model’s inability to accurately recall specific details or the presence of data. Since the paper repository data is mostly collected from the Libraries of TU Delft, it might be that the model is overfitting to certain aspects specific to that dataset. To address this, incorporating a more robust validation step in the training process could help. Additionally, integrating a feedback mechanism where users can flag incorrect responses may enable continuous improvement of the model’s accuracy and reliability.

3. **Scope and Scalability:** The scope of this research was limited to educational data from TU Delft’s course selection and paper repository websites. To gain a broader understanding of data preparation and training for domain-specific models in education, future research should consider involving a wider variety of educational websites. These could include online course platforms, educational forums, and other university repositories. By expanding the types of educational websites included, researchers can explore more diverse data structures and develop more robust models capable of handling a wider range of educational content. Additionally, developers could extend this approach to train domain-specific LLMs in other fields. Conducting experiments with larger datasets would further validate the models’ scalability and performance.

4. **Diversity of Evaluators:** Our evaluators were relatively limited in number, which may affect the robustness and reliability of our evaluation results. In future studies, expanding the pool of evaluators to include a more diverse range of perspectives and expertise could help achieve more comprehensive and reliable assessments of the models’ performance. This would ensure that the evaluations better capture the varied ways in which different users might interact with and perceive the models’ outputs, leading to more generalizable and actionable insights.

7 Conclusion

In this research, we develop a framework to train domain-specific LLMs in the educational field from semi-structured web data. We leverage LLMs to generate an instruction-tuning dataset based on data extracted from educational websites. Finally, we conduct a comprehensive performance analysis to determine how various training factors affect model performance. The study addresses two primary research questions: transforming semi-structured educational website data into instruction tuning task datasets (RQ1) and understanding the impacts of different training techniques on the performance of domain-specific educational LLMs (RQ2).

7.1. Transformation of Semi-structured Data (RQ1)

For the first research question, we investigated methodologies for converting semi-structured educational data into structured datasets suitable for instruction tuning.

We developed a systematic and cost-effective approach to handle semi-structured web data sources effectively. This involved cleaning and validating the data to ensure high quality, removing irrelevant content, and verifying accuracy. Task design and template creation were guided by student surveys, ensuring that the tasks aligned with their needs. By automating the dataset generation process with LLMs, we cost-effectively produced instruction-tuning instance datasets. Furthermore, comprehensive human assessments confirmed that the generated datasets met students' needs, maintained accuracy, and were relevant to the educational domain. Our approach demonstrates the suitability for transforming semi-structured web data into valuable training datasets for domain-specific LLMs.

7.2. Impact of Training Techniques (RQ2)

For the second research question, we focused on understanding the effects of various training techniques on the performance of domain-specific educational LLMs.

7.2.1. Further Pre-training

Further pre-training enabled the model to learn the structure of educational webpage data, improving accuracy and contextual relevance. This step is recommended, especially when the data source contains domain-specific terminology. However, the research showed that while performance gains from further pre-training are notable, the rate of improvement decreases as the size of the pre-training dataset increases. Researchers should start with smaller pre-training datasets and expand incrementally based on observed performance improvements.

7.2.2. Multitask Training

Instruction tuning a model on multiple tasks can significantly enhance its performance. The Llama3 model trained on three tasks outperformed the single-task model in relevance, accuracy, and clarity. However, introducing datasets from different sources, such as combining course selection data with paper repository data, presented challenges, indicating that less correlated datasets may affect performance. Researchers should carefully select tasks and datasets to ensure they complement each other and introduce new datasets gradually to monitor the model's performance.

7.2.3. Different Model Performance

The comparative performance evaluation of Llama2, Llama3, Gemma, and Mixtral models reveals key insights. Llama3 and Mixtral demonstrated strong capabilities in generating contextually appropriate and clear content. Gemma excelled in providing accurate information but at the expense of clarity. The study emphasizes the need for advanced training techniques and further pre-training to improve older models like Llama2. Additionally, Mixtral's high computational demands highlight the trade-off between computational resources and performance. Researchers must carefully evaluate the feasibility of resource-intensive models and balance the benefits of enhanced performance with available computational resources.

7.3. Future Work

Building on the findings and limitations of this study, several avenues for future research are identified to enhance the development and performance of domain-specific educational LLMs:

1. **Numerical Data Handling:** Future work should focus on improving the model's capability to handle numerical data accurately. Implementing a Retrieval-Augmented Generation (RAG) approach could enhance the model's ability to retrieve and accurately utilize numerical data and course codes from reliable databases, addressing the observed weaknesses in handling assessment methods and course codes.

2. **Reducing Hallucinations:** Addressing the issue of hallucinations is critical. Future research could incorporate more robust validation steps and user feedback mechanisms to flag incorrect responses, thereby continuously improving the model's accuracy and reliability. Additionally, ensuring a diverse training dataset to prevent overfitting to specific datasets, like the TU Delft repository, is essential.

3. **Expanding the Scope and Scalability:** To develop more versatile and robust models, future studies should include a broader variety of educational data sources with larger sizes. This could involve online course platforms, educational forums, and other university repositories, allowing the exploration of diverse data structures and improving the model's ability to handle a wider range of educational content. Additionally, developers could extend this approach to train domain-specific LLMs in other fields. Conducting experiments with larger datasets would further validate the models' scalability and performance.

4. **Increasing Evaluator Diversity:** Expanding the pool of evaluators in future research is crucial for achieving more robust and reliable assessments. Including a diverse range of perspectives and expertise will provide a comprehensive evaluation of the models' performance, capturing the varied ways different users interact with and perceive the models' outputs, leading to more generalizable and actionable insights.

References

[1] Artificial Analysis AI. *Mixtral-8x7B-Instruct: Advanced Language Models*. https://artificialanalysis.ai/models/mixtral-8x7b-instruct. Accessed: 2024-06-21. 2023.

[2] Joshua Ainslie et al. “Gqa: Training generalized multi-query transformer models from multi-head checkpoints”. In: *arXiv preprint arXiv:2305.13245* (2023).

[3] Stephen H. Bach et al. *PromptSource: An Integrated Development Environment and Repository for Natural Language Prompts*. 2022. arXiv: 2202.01279 [cs.LG].

[4] Jillian Bommarito et al. “Gpt as knowledge worker: A zero-shot evaluation of (ai) cpa capabilities”. In: *arXiv preprint arXiv:2301.04408* (2023).

[5] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL].

[6] Seungeon Cha, Martin Loeser, and Kyoungwon Seo. “The Impact of AI-Based Course-Recommender System on Students’ Course-Selection Decision-Making Process”. In: *Applied Sciences* 14.9 (2024). ISSN: 2076-3417. DOI: 10.3390/app14093672. URL: https://www.mdpi.com/2076-3417/14/9/3672.

[7] Banghao Chen et al. “Unleashing the potential of prompt engineering in Large Language Models: a comprehensive review”. In: *arXiv preprint arXiv:2310.14735* (2023).

[8] Peter Clark et al. *Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge*. 2018. arXiv: 1803.05457 [cs.AI].

[9] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. “Multi-head attention: Collaborate instead of concatenate”. In: *arXiv preprint arXiv:2006.16362* (2020).

[10] Robert Dale. “GPT-3: What’s it good for?” In: *Natural Language Engineering* 27.1 (2021), pp. 113–118.

[11] Brian Dean. *ChatGPT Statistics and Facts (2023)*. https://backlinko.com/chatgpt-stats. Accessed: 2024-06-21. 2023.

[12] Cheng Deng et al. *K2: A Foundation Language Model for Geoscience Knowledge Understanding and Utilization*. 2023. arXiv: 2306.05064 [cs.CL].

[13] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[14] Aleksandra Edwards et al. “Go Simple and Pre-Train on Domain-Specific Corpora: On the Role of Training Data for Text Classification”. In: *Proceedings of the 28th International Conference on Computational Linguistics*. Ed. by Donia Scott, Nuria Bel, and Chengqing Zong. Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 5522–5529. DOI: 10.18653/v1/2020.coling-main.481. URL: https://aclanthology.org/2020.coling-main.481.

[15] 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.

[16] Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685.

[17] Lei Huang et al. “A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions”. In: *arXiv preprint arXiv:2311.05232* (2023).

[18] Wei Huang et al. “How Good Are Low-bit Quantized LLaMA3 Models? An Empirical Study”. In: *arXiv preprint arXiv:2404.14047* (2024).

[19] Albert Q Jiang et al. “Mixtral of experts”. In: *arXiv preprint arXiv:2401.04088* (2024).

[20] Zhengbao Jiang et al. “How can we know when language models know? on the calibration of language models for question answering”. In: *Transactions of the Association for Computational Linguistics* 9 (2021), pp. 962–977.

[21] Tejaswi Kashyap. *Deciphering Mixtral-8x7B: Navigating the Sparse Expert Model Ensemble by Mistral AI*. https://medium.com/@tejaswi_kashyap/deciphering-mixtral-8x7b-navigating-the-sparse-expert-model-ensemble-by-mistral-ai-a39c0ba917eb. Accessed: 2024-04-24. 2023.

[22] Klaus Krippendorff. *Computing Krippendorff’s alpha-reliability*. 2011.

[23] 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.

[24] Yinhan Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: 1907.11692 [cs.CL].

[25] Brandon McKinzie et al. *MM1: Methods, Analysis Insights from Multimodal LLM Pre-training*. 2024. arXiv: 2403.09611.

[26] Meta AI. *Introducing Meta Llama 3: The most capable openly available LLM to date*. Apr. 2024. URL: https://ai.meta.com/blog/meta-llama-3/.

[27] Swaroop Mishra et al. “Natural instructions: Benchmarking generalization to new tasks from natural language instructions”. In: *arXiv preprint arXiv:2104.08773* (2021), pp. 839–849.

[28] Steven Moore et al. “Empowering education with llms-the next-gen interface and content generation”. In: *International Conference on Artificial Intelligence in Education*. Springer. 2023, pp. 32–37.

[29] Long Ouyang et al. “Training language models to follow instructions with human feedback”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 27730–27744.

[30] Colin Raffel et al. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. 2023. arXiv: 1910.10683 [cs.LG].

[31] Leonard Richardson. “Beautiful soup documentation”. In: *April* (2007).

[32] Carlos Riquelme et al. “Scaling vision with sparse mixture of experts”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 8583–8595.

[33] Sagar Shivaji Salunke. *Selenium Webdriver in Python: Learn with Examples*. 1st. North Charleston, SC, USA: CreateSpace Independent Publishing Platform, 2014. ISBN: 1497337364.

[34] Ronald Sebalamu. *JP Morgan’s DocLLM: Advanced Language Models in the Financial Sector*. https://medium.com/@ronaldssebalamu/jp-morgans-docllm-advanced-language-models-in-the-financial-sector-cb7646bfda2a. Accessed: 2024-06-21. 2023.

[35] Yash Shah. *Introduction to LLMs and the Generative AI, Part 3: Fine-tuning LLM with Instruction*. https://medium.com/@yash9439/introduction-to-llms-and-the-generative-ai-part-3-fine-tuning-llm-with-instruction-and-326bc95e07ae. Accessed: 2024-04-11. 2023.

[36] Max Shapp. *Grouped Query Attention (GQA) Explained with Code*. https://medium.com/@maxshapp/grouped-query-attention-gqa-explained-with-code-e56ee2a1df5a. Accessed: 2024-05-02. 2023.

[37] Noam Shazeer. *Fast Transformer Decoding: One Write-Head is All You Need*. 2019. arXiv: 1911.02150 [cs.NE].

[38] Karan Singhal et al. *Large Language Models Encode Clinical Knowledge*. 2022. arXiv: 2212.13138.

[39] Jianlin Su et al. “Roformer: Enhanced transformer with rotary position embedding”. In: *Neurocomputing* 568 (2024), p. 127063.

[40] Rohan Taori et al. *Stanford Alpaca: An Instruction-following LLaMA model*. https://github.com/tatsu-lab/stanford_alpaca. 2023.

[41] Gemma Team et al. *Gemma: Open Models Based on Gemini Research and Technology*. 2024. arXiv: 2403.08295 [cs.CL].

[42] Hugo Touvron et al. "Llama: Open and efficient foundation language models". In: *arXiv preprint arXiv:2302.13971* (2023).

[43] Chris van der Lee et al. "Human evaluation of automatically generated text: Current trends and best practice guidelines". In: *Computer Speech Language* 67 (2021), p. 101151. ISSN: 0885-2308. DOI: `https://doi.org/10.1016/j.csl.2020.101151`. URL: `https://www.sciencedirect.com/science/article/pii/S088523082030084X`.

[44] Yu-An Wang and Yun-Nung Chen. "What do position embeddings learn? an empirical study of pre-trained language model positional encoding". In: *arXiv preprint arXiv:2010.04903* (2020).

[45] Shen Wang et al. "Large language models for education: A survey and outlook". In: *arXiv preprint arXiv:2403.18105* (2024).

[46] Yizhong Wang et al. "Self-instruct: Aligning language model with self generated instructions". In: *arXiv preprint arXiv:2212.10560* (2022).

[47] Jason Wei et al. *Finetuned Language Models Are Zero-Shot Learners*. 2022. arXiv: `2109.01652 [cs.CL]`.

[48] Xiaodong Wu, Ran Duan, and Jianbing Ni. "Unveiling security, privacy, and ethical concerns of ChatGPT". In: *Journal of Information and Intelligence* 2.2 (2024), pp. 102–115. ISSN: 2949-7159. DOI: `https://doi.org/10.1016/j.jiixd.2023.10.007`. URL: `https://www.sciencedirect.com/science/article/pii/S2949715923000707`.

[49] Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. *Hallucination is Inevitable: An Innate Limitation of Large Language Models*. 2024. arXiv: `2401.11817 [cs.CL]`. URL: `https://arxiv.org/abs/2401.11817`.

[50] Chang Yu et al. *Seek for Incantations: Towards Accurate Text-to-Image Diffusion Synthesis through Prompt Engineering*. 2024. arXiv: `2401.06345 [cs.CV]`.

[51] Shengyu Zhang et al. "Instruction tuning for large language models: A survey". In: *arXiv preprint arXiv:2308.10792* (2023).

[52] Yue Zhang et al. *Multi-Task Instruction Tuning of LLaMa for Specific Scenarios: A Preliminary Study on Writing Assistance*. 2023. arXiv: `2305.13225`.

[53] Yue Zhang et al. "Siren's song in the ai ocean: A survey on hallucination in large language models". In: *arXiv preprint arXiv:2309.01219* (2023).

[54] Zhicheng Zhang, Haijun Zhao, and Huayue Chen. "Chinese Named Entity Recognition Based on BERT and Grouped-query Attention". In: *2024 IEEE 7th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*. Vol. 7. IEEE. 2024, pp. 397–401.

[55] Yaowei Zheng et al. *LlamaFactory: Unified Efficient Fine-Tuning of 100+ Language Models*. 2024. arXiv: `2403.13372 [cs.CL]`.